Е. П. Угрюмов

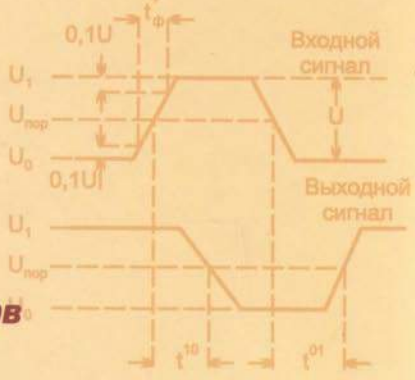
Master-serial mode

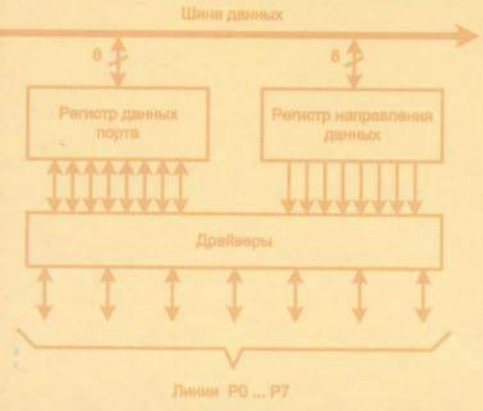
DRAM

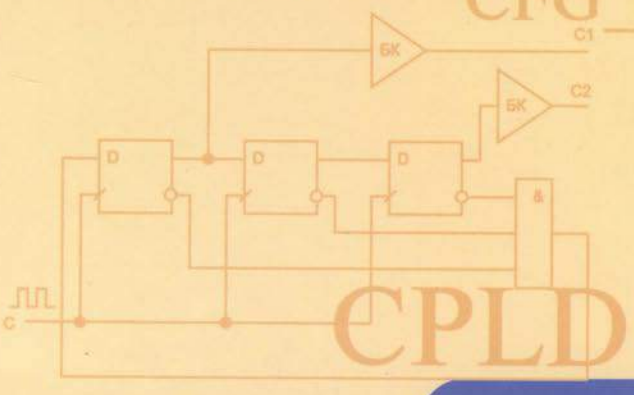
ЦИФРОВАЯ СХЕМОТЕХНИКА

2-Е ИЗДАНИЕ

- **ИС всех уровней интеграции: от простейших логических элементов до перспективных БИС/СБИС с программируемыми структурами**
- **Принципы построения цифровых узлов и их практическая реализация**
- **Методы проектирования цифровых устройств**







УЧЕБНОЕ ПОСОБИЕ

bhv

Угрюмов Е. П.

ЦИФРОВАЯ СХЕМОТЕХНИКА

Издание 2-е, переработанное и дополненное

Рекомендовано учебно-методическим объединением в области машиностроения и приборостроения в качестве учебного пособия для студентов направлений 654600 и 552800 — "Информатика и вычислительная техника" (специальность 220100 "Вычислительные машины, комплексы, системы и сети")

Санкт-Петербург
«БХВ-Петербург»
2007

УДК 681.3.06
ББК 32.973.26-04я73
У27

Угрюмов Е. П.

У27 Цифровая схемотехника: Учеб. пособие для вузов. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2007. — 800 с.: ил.

ISBN 978-5-94157-397-4

В книге рассматривается широкий круг вопросов, связанных с изучением, проектированием и применением цифровых элементов, узлов и устройств, микросхемы которых являются основой для реализации различных средств обработки информации — ЭВМ, систем цифровой автоматики, телекоммуникаций, измерений и др. Рассмотрены общие проблемы схемотехнической реализации цифровых устройств, их типовые функциональные узлы, структуры и схемотехника запоминающих устройств, простых микропроцессоров, интерфейсных схем и схем программируемой логики. По сравнению с первым изданием дополнены и расширены разделы, посвященные микроконтроллерам, программируемой логике, запоминающим устройствам, проблемам скоростной передачи данных и др. Изложена методика как "ручных", так и автоматизированных методов проектирования цифровых узлов и устройств.

Для студентов технических вузов и техникумов, а также специалистов, работающих в области создания цифровой аппаратуры

У35161

УДК 681.3.06
ББК 32.973.26-04я73

Рецензенты:
кафедра автоматики и вычислительной техники Санкт-Петербургского государственного технического университета (зав. кафедрой профессор В. Ф. Мелехин),
доцент С. Р. Иванов (Московский технический университет им. Н. Э. Баумана)

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Людмила Еремеевская*
Зав. редакцией	*Григорий Добин*
Редактор	*Юрий Рожко*
Компьютерная верстка	*Натальи Смирновой*
Корректор	*Наталия Першакова*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 29.01.07.
Формат 70×100$^{1}/_{16}$. Печать офсетная. Усл. печ. л. 64,5.
Доп. тираж 3000 экз. Заказ № 904
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 55.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02 от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в ОАО "Техническая книга"
190005, Санкт-Петербург, Измайловский пр., 29.

ISBN 978-5-94157-397-4

© Угрюмов Е. П., 2004
© Оформление, издательство "БХВ-Петербург", 2004

НТБ ВНТУ
м.Вінниця

Содержание

Предисловие 1

Введение 5

Глава 1. Схемотехнические проблемы построения цифровых узлов и устройств 7

§ 1.1. Простейшие модели и система параметров логических элементов 7
Простейшие модели логических элементов 7
Статические параметры логических элементов 10
Быстродействие логических элементов 11
Мощности потребления логических элементов 13
§ 1.2. Типы выходов цифровых элементов 13
Логический выход 13
Выход с тремя состояниями 15
Выход с открытым коллектором (стоком) 17
Выход с программированием ТС—ОС (с тремя состояниями или с открытым стоком) 21
Выход с открытым эмиттером 22
§ 1.3. Схемотехника входных цепей КМОП-элементов и режимы временно разомкнутых выводов 22
Выводы микросхем с pull-up- и pull-down-резисторами 22
Выводы микросхем с запоминанием последнего значения сигнала 24
§ 1.4. Паразитные связи цифровых элементов по цепям питания. Фильтрация питающих напряжений в схемах ЦУ 26
§ 1.5. Передача сигналов в цифровых узлах и устройствах. Помехи в сигнальных линиях. Сигнальные линии повышенного качества 28
Перекрестные помехи 28
Искажения сигналов в несогласованных линиях 29
Параллельное согласование волновых сопротивлений 32
Последовательное согласование волновых сопротивлений 36
Схемы с одновременным согласованием волновых сопротивлений в конце и начале линии 37
Линии передачи сигналов 37
Линии связи с гальваническими развязками 40
Линии передачи типа "токовая петля" 42
О стандартах сигналов ввода/вывода данных 43
Передача данных с двойной скоростью (технология DDR) 50
О разрядностях высокоскоростных шин. Блоки SERDES и CDR 50

§ 1.6. Вспомогательные элементы цифровых узлов и устройств ... 53
Элементы задержки ... 53
Формирование импульсов по длительности ... 55
Генераторы импульсов ... 56
Элементы индикации на светодиодах ... 58
Элементы индикации на жидких кристаллах ... 61
§ 1.7. О некоторых типовых ситуациях при построении узлов и устройств на стандартных ИС ... 64
Режимы неиспользуемых входов ... 64
Согласование уровней сигналов при сопряжении разнотипных элементов ... 66
Режимы неиспользуемых элементов ... 67
Наращивание числа входов ... 68
Снижение нагрузок на выходах логических элементов ... 68
Прошлое и настоящее малых и средних интегральных схем ... 69

Глава 2. Функциональные узлы комбинационного типа ... 73

§ 2.1. Введение в проблематику проектирования ЦУ комбинационного типа ... 73
§ 2.2. Двоичные дешифраторы ... 81
Схемотехническая реализация дешифраторов ... 83
§ 2.3. Приоритетные и двоичные шифраторы. Указатели старшей единицы ... 85
§ 2.4. Мультиплексоры и демультиплексоры ... 88
§ 2.5. Универсальные логические модули на основе мультиплексоров ... 91
Первый способ настройки УЛМ ... 92
Второй способ настройки УЛМ ... 93
Пирамидальные структуры УЛМ ... 95
§ 2.6. Компараторы ... 100
§ 2.7. Схемы контроля ... 103
Мажоритарные элементы ... 104
Контроль по модулю 2 ... 105
Схемы свертки ... 107
Передача данных с контролем по модулю 2 ... 109
Контроль логического преобразователя ... 109
Контроль с использованием кодов Хемминга ... 110
Схемы кодера и декодера для кода Хемминга ... 112
§ 2.8. Сумматоры ... 114
Одноразрядный сумматор ... 114
Последовательный сумматор ... 117
Параллельный сумматор с последовательным переносом ... 118
Параллельный сумматор с параллельным переносом ... 119
Сумматор с передачей сигнала переноса по цепочке замкнутых ключей ... 122
Сумматоры групповой структуры ... 124
Сумматор с условным переносом ... 127

§ 2.9. Арифметико-логические устройства и блоки ускоренного переноса....129
§ 2.10. Матричные умножители....132
Множительно-суммирующие блоки....133
Схемы ускоренного умножения....135
§ 2.11. Быстрые сдвигатели....138
Сдвигатель, управляемый кодом "1 из N"....139
Сдвигатель, управляемый двоичным кодом (логарифмический)....141

Глава 3. Функциональные узлы последовательностного типа (автоматы с памятью)....143

§ 3.1. Триггерные устройства (элементарные автоматы). Классификация. Основные сведения....143
Классификация триггеров....144
Времена предустановки и выдержки....147
Способы описания триггеров....147
§ 3.2. Схемотехника триггерных устройств....149
§ 3.3. Аномальные состояния триггеров....162
§ 3.4. Применение триггеров в схемах ввода и синхронизации логических сигналов и в генераторах синхропоследовательностей....162
Ввод логических сигналов от механических ключей....162
Синхронизаторы одиночных импульсов,....164
Ввод асинхронных данных....165
Формирование вторичных синхросигналов из опорной синхропоследовательности....166
§ 3.5. Введение в проблематику и методику проектирования автоматов с памятью....167
Проектирование автоматов....169
Пример проектирования....172
§ 3.6. Синхронизация в цифровых устройствах....179
Параметры тактовых импульсов....180
Структура устройств синхронизации....182
Размножение тактовых импульсов....183
Устройства типов PLL и DLL, улучшающие работу системы синхронизации....184
Однофазная синхронизация....188
Расчетные соотношения для проектирования однофазной синхронизации....190
Двухфазная синхронизация....192
§ 3.7. Регистры и регистровые файлы....194
Регистровые файлы....195
Сдвигающие регистры....197
Универсальные регистры....198
§ 3.8. Основные сведения о счетчиках. Двоичные счетчики....201
Классификация счетчиков....202
Двоичные счетчики....202
Счетчики с групповой структурой....206

§ 3.9. Двоично-кодированные счетчики с произвольным модулем 208
Построение счетчика методом модификации межразрядных связей....209
Построение счетчика методом управляемого сброса 212
§ 3.10. Счетчики с недвоичным кодированием 213
Счетчики в коде Грея 213
Счетчики в коде "1 из N" 214
Счетчики в коде "1 из N" на основе кольцевых регистров 215
Счетчики в коде "1 из N" на основе счетчиков Джонсона....218
§ 3.11. Полиномиальные счетчики....222
Схемы генераторов псевдослучайной последовательности (ГПСП) 224

Глава 4. Запоминающие устройства 227

§ 4.1. Основные сведения. Система параметров. Классификация 227
Важнейшие параметры ЗУ....228
Входные и выходные сигналы ЗУ 229
Классификация современных ЗУ 231
Классификация перспективных ЗУ....236
Преобладающие виды современной памяти 237
§ 4.2. Основные структуры запоминающих устройств....237
Структура 2D 237
Структура 3D 238
Структура 2DM....241
Структура блочных ЗУ....242
Память с последовательным доступом 244
Кэш-память....247
§ 4.3. Структурные методы повышения быстродействия запоминающих устройств 252
Кучность адресов и произвольный доступ....252
Быстрый страничный доступ 253
Пакетная передача данных и команд....253
Передача данных с удвоенной скоростью (технология DDR — Double Data Rate).
Технология QDR (Quad Data Rate) 254
Применение многобанковых структур....254
Конвейеризация трактов передачи данных....255
§ 4.4. Запоминающие устройства с рабочим режимом "только для чтения" (типа ROM(M), PROM, EPROM, EEPROM)....256
Масочные ЗУ 256
ЗУ типа PROM 259
ЗУ типов EPROM, EPROM-OTP и EEPROM 262
Импульсное питание ROM 268
§ 4.5. Флэш-память 269
Структуры с несимметричными и симметричными блоками....271
Структуры с ячейками ИЛИ-НЕ и И-НЕ....271
Способы улучшения параметров флэш-памяти....274
Команды управления флэш-памятью 275

О номенклатуре микросхем флэш-памяти 276
Флэш-память с несимметричной блочной структурой 277
Флэш-память с симметричной блочной структурой (файловая) 280
Флэш-память с многоуровневым хранением заряда в плавающих затворах (типа StrataFlash и др.) 283
Флэш-память с зеркальным битом 284
§ 4.6. Последовательные ЗУ типов EEPROM и Flash 285
§ 4.7. Использование программируемых ЗУ для решения задач обработки информации 287
Реализация логических (переключательных) функций 287
Реализация конечных автоматов 288
Воспроизведение арифметических операций и функциональных зависимостей 289
§ 4.8. Статические запоминающие устройства 291
Запоминающие элементы статических ЗУ 292
Внешняя организация и временные диаграммы статических ЗУ 293
Статические ЗУ повышенного быстродействия 297
Искусственная энергонезависимость статических ЗУ с резервным источником питания 299
Энергонезависимая память типа NV-SRAM 301
§ 4.9. Динамические запоминающие устройства — базовая структура 303
Запоминающие элементы 303
Усилители-регенераторы 307
Мультиплексирование шины адреса 307
Внешняя организация и временные диаграммы 308
Схема динамического ЗУ 309
§ 4.10. Регенерация данных в динамических запоминающих устройствах 311
§ 4.11. Динамические запоминающие устройства повышенного быстродействия 313
Структуры FPM, EDORAM, BEDORAM 314
Структура типа MDRAM 316
Структуры типа SDRAM 318
Структуры типа DDR SDRAM 320
Структуры типа RDRAM 324
Структура типа CDRAM 329
Динамические ЗУ с фрагментированной (блочной) базовой структурой 329
Структуры типа RLDRAM 331
Структуры типа FCRAM 332
Параметры динамических ЗУ 333
О конструктивных модулях памяти 334
§ 4.12. Перспективные запоминающие устройства (FRAM, PFRAM, MRAM, OUM) 335
ЗУ типа FRAM (ферроэлектрические) 335
ЗУ типа PFRAM (полимерно-ферроэлектрические) 337
ЗУ типа MRAM (магниторезистивные) 338

ЗУ типа OUM (с использованием фазовых переходов вещества) 340
Параметры перспективных микросхем памяти 341
§ 4.13. Заключительные замечания 342

Глава 5. Микропроцессорные БИС/СБИС и их применение в микропроцессорных системах 345

§ 5.1. Общие сведения. Структура и функционирование микропроцессорной системы 345
Структура микропроцессорной системы 349
§ 5.2. Управление памятью и внешними устройствами. Построение модуля памяти 354
Модули памяти 356
Сигналы управления 357
Вид обмена 358
§ 5.3. Структура и функционирование микропроцессора 359
Структура микропроцессора К1821ВМ85А 359
Блок регистров 361
Блок синхронизации и управления 362
Функции выводов и сигналов микропроцессора 363
Синхронизация и последовательность действий МП 367
Система прерываний 369
Система команд МП 373
Пример выполнения команды 378
§ 5.4. Схемы подключения памяти и внешних устройств к шинам микропроцессорной системы 380
Анализ нагрузочных условий 387
Согласование временных диаграмм МП и ЗУ 388
Схемы реализации безусловного программного ввода/вывода 392
Схемы реализации условного программного ввода/вывода 394

Глава 6. Интерфейсные и периферийные микросхемы 397

§ 6.1. Интерфейсы микропроцессорных систем 397
Интерфейсы начального периода развития МПС 399
Интерфейсы последующих поколений МПС, персональных компьютеров и других современных систем 400
§ 6.2. Шинные формирователи и буферные регистры 401
Шинные формирователи 401
Буферные регистры 403
§ 6.3. Параллельные порты и адаптеры 404
Схемотехника параллельных портов 404
Параллельные периферийные адаптеры 410
§ 6.4. Последовательные интерфейсы и программируемые связные адаптеры 419
Общие сведения 419
Программируемые связные адаптеры (ПСА) 424
Модули UART 433

Интерфейс SPI ... 434
Интерфейс I^2C ... 437
§ 6.5. Схемы обслуживания прерываний. Программируемые контроллеры прерываний ... 439
Аппаратный опрос источников прерываний ... 439
Программируемые контроллеры прерываний ... 440
Структура ПКП ... 443
Каскадное включение контроллеров ... 449
§ 6.6. Контроллеры прямого доступа к памяти ... 450
Структура и функции КПДП ... 451
Выводы и сигналы контроллера ... 457
Наращивание числа каналов ПДП ... 458
§ 6.7. Таймеры ... 459
Таймеры, входящие в состав микроконтроллеров ... 459
Программируемый интервальный таймер ВИ54 ... 464
§ 6.8. Схемотехника интерфейса JTAG ... 471
Интерфейс JTAG и граничное сканирование ... 471

Глава 7. Микроконтроллеры ... 479

§ 7.1. Основные сведения ... 479
§ 7.2. Структура микроконтроллера ... 482
§ 7.3. Организация памяти и функционирование микроконтроллера ... 486
Распределение памяти в MK AVR ... 486
Способы адресации, используемые в микроконтроллере ... 488
Выполнение команд при работе микроконтроллера ... 489
Режимы различного потребления мощности ... 490
Система прерываний ... 491
Программирование микроконтроллера ... 491

Глава 8. Программируемые логические матрицы, программируемая матричная логика, базовые матричные кристаллы ... 493

§ 8.1. Вводные замечания ... 493
§ 8.2. Программируемые логические матрицы и программируемая матричная логика (ПЛМ и ПМЛ) ... 494
Структура ПЛМ ... 494
Схемотехника ПЛМ ... 497
Подготовка задачи к решению с помощью ПЛМ ... 501
Программирование ПЛМ ... 501
Упрощенное изображение схем ПЛМ ... 502
Воспроизведение скобочных форм переключательных функций ... 503
Структура ПМЛ ... 506
Обогащение функциональных возможностей ПЛМ и ПМЛ ... 507
Примеры промышленных ПМЛ (серии К1556) ... 512
Пример подготовки задачи к решению с помощью ПМЛ ... 515
Примеры зарубежных ПМЛ с усложненными макроячейками ... 516

§ 8.3. Базовые матричные кристаллы (вентильные матрицы с масочным программированием) 520
Основные сведения 520
Классификация БМК 523
Параметры БМК 528
Этапы проектирования МАБИС 529

Глава 9. Архитектура и схемотехника БИС/СБИС с программируемыми структурами (CPLD, FPGA, смешанные структуры) 533

§ 9.1. Общие сведения 533
Классификация ПЛИС по конструктивно-технологическому типу программируемых элементов 535
О некоторых общих свойствах и возможностях применения ПЛИС 539
Области применения ПЛИС 541
§ 9.2. Сложные программируемые логические устройства (CPLD) 543
Структура CPLD 543
Функциональные блоки CPLD 544
Системы коммутации CPLD (программируемые матрицы соединений) 549
Блоки ввода/вывода CPLD 551
§ 9.3. Программируемые пользователем вентильные матрицы (FPGA) 553
Предварительные замечания 553
Логические блоки FPGA 554
Блоки ввода/вывода FPGA 559
Системы межсоединений FPGA 561
Обобщенная структура FPGA 567
§ 9.4. СБИС программируемой логики с комбинированными архитектурами 569
Структура СБИС ПЛ с комбинированной архитектурой 569
Логические элементы 571
Логические блоки и их коммутация 576
Встроенные блоки памяти 578
Блоки ввода/вывода 580
§ 9.5. Программируемые аналоговые и аналого-цифровые схемы 580
Общие сведения 580
Два варианта аналоговой схемотехники 581
Практические разработки 583

Глава 10. СБИС программируемой логики типа "система на кристалле" 591

§ 10.1. Общие сведения 591
IP-ядра и типы программируемых "систем на кристалле" 592
Сопоставление и возможности двух типов SOPC 594
Процессорные ядра SOPC 596
§ 10.2. СБИС ПЛ типа "система на кристалле" с синтезируемыми ядрами (с однородной структурой) 598
Семейства СБИС типа APEX 20K/KE/KC, APEX II 598
Семейство Stratix 601

Семейства СБИС типов Virtex E/EM, Virtex II, Virtex II Pro ... 604
SOPC с энергонезависимой памятью конфигурации ... 608
§ 10.3. СБИС ПЛ типа "система на кристалле" с аппаратными ядрами (блочные) ... 609
Блочные SOPC, не содержащие процессорных ядер ... 609
Блочные SOPC с процессорными ядрами ... 610
Семейство FPSLIC фирмы Atmel ... 611
Блочные SOPC фирмы Triscend ... 617
Блочные SOPC фирмы Altera ... 619
Блочные SOPC с аналоговыми и цифровыми блоками ... 620

Глава 11. Некоторые аспекты применения микросхем с программируемой структурой ... 623

§ 11.1. Конвертация проектов ... 623
§ 11.2. Конфигурирование микросхем ... 626
§ 11.3. Защищенность проектов от рассекречивания ... 629
§ 11.4. Оценка логической сложности и быстродействия ПЛИС ... 632
Оценка логической сложности ПЛИС ... 632
Оценка быстродействия ПЛИС ... 634

Глава 12. Методика и средства автоматизированного проектирования цифровых устройств ... 637

§ 12.1. Общее описание процесса проектирования ... 637
§ 12.2. Классификация интегральных схем по характеру их разработки, производства и применения ... 642
§ 12.3. Области применения ИС различных типов ... 647
§ 12.4. Место программируемой логики в процессе создания современной аппаратуры ... 649
§ 12.5. Инструментарий проектировщика ... 652
Средства системного этапа проектирования ... 653
Разработка специфических фрагментов проекта ... 654
Средства разработки процессорной части проекта ... 655
Средства разработки цифровой части проекта ... 658
Средства разработки аналоговых и аналого-цифровых фрагментов ... 660
Работы и средства этапа комплексной отладки проекта ... 661
Специфика конструирования и отладки проектов на ПЛИС и SOPC ... 662
§ 12.6. Системный этап проектирования цифровых устройств на базе ПЛИС ... 663
Выбор САПР ... 663
Представление проекта на блочно-функциональном уровне ... 664
Средства описания проекта ... 665
Средства описания автоматов ... 668
§ 12.7. Маршрут проектирования ПЛИС и возможности типовых САПР ... 670
Этапы проектных процедур с использованием САПР ... 670
§ 12.8. Основные сведения о языке VHDL ... 673
Синтаксические конструкции и основные понятия языка ... 674

Описание проекта на языке VHDL 675
Примеры поведенческих описаний элементов на языке VHDL 678
Язык VHDL для моделирования и синтеза 679
Структурный и поведенческий варианты описания проекта 680
О возможностях и средствах описания типовых узлов цифровой техники 681
Введение в язык VHDL-AMS 695
§ 12.9. Пример автоматизированного проектирования цифрового устройства с использованием языков описания аппаратуры 705
Первый этап. Рассмотрение ТЗ на разрабатываемое устройство 706
Второй этап. Разработка общей структуры операционного блока 707
Третий этап. Описание работы управляющего автомата 709
Пояснения к синтаксису AHDL и VHDL программ устройства управления 711
Четвертый этап. Компиляция проекта и основные параметры устройства 718
Пятый этап. Тестирование проекта 718
Шестой этап. Автоматическое определение временных характеристик устройства 720
Седьмой этап. Практическое использование результатов проектирования 720

Приложение 1. Сведения о некоторых популярных восьмиразрядных микроконтроллерах 721

Приложение 2. Функциональные ячейки библиотеки БМК 723

Приложение 3. Обзор продукции основных производителей микросхем программируемой логики 724
П.3.1. Однократно программируемые микросхемы 724
П.3.2. Микросхемы с программированием состояний плавающих затворов 725
П.3.3. Микросхемы с триггерной памятью конфигурации 726
П.3.4. Пример более подробного описания микросхемы программируемой логики 729

Глоссарий 731
Дополнение. Словарь иностранных сокращений и терминов 748

Принятые сокращения 756

Литература и источники в Интернете 761
Краткая библиография 761
Интернет-ресурсы 765

Предметный указатель 767

Предисловие

Информатизация общества — важнейшая задача, стоящая перед нашей страной и требующая интенсивного развития вычислительной техники и других средств обработки информации.

Все разнообразные средства цифровой техники: ЭВМ, микропроцессорные системы измерений и автоматизации технологических процессов, цифровая связь и телевидение и т. д. строятся на единой элементной базе, в состав которой входят чрезвычайно разные по сложности микросхемы — от логических элементов, выполняющих простейшие операции, до сложнейших программируемых кристаллов, содержащих миллионы логических элементов.

Создание современной элементной базы средств вычислительной техники — научно-техническая задача, решение которой по силам только наиболее развитым и экономически сильным странам. В России и странах СНГ развитие микроэлектроники в настоящее время находится в кризисном состоянии. Отставание от передовых стран (США, Японии) наметилось уже в СССР, когда переход на производство субмикронных интегральных схем не был освоен, что привело к нарастающему отставанию от уровня мировой техники. Переход к рыночной экономике, вызвавший ломку в экономике страны и поставивший на первый план конкурентоспособность продукции, привел к потере электронной промышленностью традиционных рынков и около 2/3 имевшихся мощностей. В течение приблизительно 10 лет после начала перестройки более или менее успешно функционирующими оставались единичные предприятия.

Начиная с 1999 г. стал отмечаться рост производства электронной промышленности на 30—45% в год. Однако выход на современный технологический уровень представляет для отечественной микроэлектроники довольно сложную задачу, поскольку требует вложения огромных средств. Имеются планы и программы развития российской электроники, свидетельствующие, по крайней мере, о возникшем понимании важности и неотложности ее поддержки. Государственная поддержка на требуемом уровне пока не реальна (даже крупнейшие мировые фирмы вынуждены объединяться в альянсы для аккумулирования средств, необходимых для создания новых схем памяти, микропроцессоров и др.). В то же время такая страна, как Россия, не может отказаться от промышленности высоких технологий, в первую очередь от развития собственного электронного приборостроения, как гражданского, так и, прежде всего, военного. Лауреат Нобелевской премии академик

Ж. И. Алферов говорит[1]: "Важно заниматься научными и технологическими исследованиями в области электроники, потому что именно она определяет технологический и даже социальный прогресс. Без собственных современных электронных технологий любые наши другие (те же космические) быстро перейдут во второсортные. Сейчас у нас два пути — либо становиться страной третьего мира, живущей за счет ресурсов, либо развивать наукоемкие отрасли". Крупнейший специалист в области информатики академик Е. П. Велихов в одной из своих статей[2] присоединился к тезису: *"Тот, кто умеет делать компьютеры, владеет миром"*.

Основа потенциальных возможностей развития российской электроники — подготовленные кадры. Промышленность СССР обладала мощным контингентом специалистов, однако он не только утерян количественно, но и ориентирован на работу в старых условиях, когда проекты выполнялись на основе стандартных элементов малого и среднего уровней интеграции. Сейчас необходимо создавать новый контингент специалистов, свободно владеющих работой на компьютере и английским языком (это требуется для доступа к информационным ресурсам и, кстати говоря, стало общим условием успешной работы для очень многих профессий), знающих современную элементную базу цифровой и аналоговой техники и способных эффективно использовать разнообразные средства систем автоматизированного проектирования (САПР).

Следует заметить, что для системотехников отсутствие отечественных микросхем современного уровня компенсируется сейчас доступностью зарубежной элементной базы, поэтому изучение цифровых узлов и устройств во всем их разнообразии имеет прямое практическое значение.

В данном учебном пособии рассмотрены принципы построения и проектирования функциональных узлов и устройств ЭВМ и цифровой автоматики и их практические реализации. Освоение материала пособия требует лишь знакомства с логическими операциями, двоичной системой счисления и действиями над двоичными числами, а также с принципами работы транзисторов и транзисторных переключательных каскадов.

Первое издание этого учебного пособия (издательство "БХВ-Петербург", 2000 г.) получило положительную оценку читателей, и в течение более чем трех лет спрос на него остается высоким. Это привело автора и издательство к решению о выпуске второго издания. За прошедший со времени выхода первого издания не очень большой срок в такой динамичной области, как цифровая схемотехника, уже произошло немало изменений, и во втором издании книги не могли не появиться новые материалы, особенно в разделах, посвященных программируемой логике, запоминающим устройствам,

[1] Газета "Известия", 15 марта 2001 г.
[2] Газета "Известия", 25 ноября 1999 г.

проблемам скоростной передачи данных и системам синхронизации в быстродействующих устройствах. Несмотря на стремление автора к согласованию объема книги с определенными лимитами времени на изучение излагаемого курса (около 100 лекционных часов), объем книги несколько увеличился.

В пособии рассмотрен широкий круг вопросов, связанных с изучением и применением современной элементной базы цифровой техники, что соответствует основному содержанию дисциплины "Схемотехника ЭВМ". Пособие предназначено для студентов вузов, обучающихся по специальности 2201 (Электронные вычислительные машины, комплексы, системы и сети) и другим специальностям, связанным с использованием и разработкой цифровых средств обработки информации. Оно может быть полезно также для работников промышленности и научных учреждений.

Содержание пособия основано на материале курса лекций, читаемого автором на кафедре "Вычислительная техника" Санкт-Петербургского государственного электротехнического университета "ЛЭТИ".

В работе над книгой участвовал к. т. н., доц. Р. И. Грушвицкий, которым написаны § 12.5—12.9 и совместно с автором § 6.8, 12.1—12.4.

Автор благодарит рецензентов за ряд ценных замечаний, способствовавших улучшению книги.

Введение

Элементную базу *цифровых устройств* (ЦУ) составляют *интегральные схемы* (ИС). Со времени их изобретения (США, 1959 г.) ИС постоянно совершенствуются и усложняются. Характеристикой сложности ИС является уровень интеграции, оцениваемый либо числом базовых логических элементов, либо числом транзисторов, которые размещены на кристалле.

Различия в уровне интеграции делят ИС на несколько категорий: МИС, СИС, БИС, СБИС (соответственно малые, средние, большие и сверхбольшие ИС). Практическое использование находят все категории, однако с течением времени все бóльшую долю используемых микросхем составляют схемы высокого уровня интеграции.

МИС реализуют простейшие логические преобразования и обладают универсальностью — даже с помощью одного типа логического элемента (например, И-НЕ) можно построить любое ЦУ. В виде СИС выпускаются в готовом виде такие схемы, как малоразрядные регистры, счетчики, дешифраторы, сумматоры и т. п. Номенклатура СИС должна быть более широкой и разнообразной, т. к. их универсальность снижается. В развитых сериях стандартных ИС насчитываются сотни типов СИС.

С появлением БИС и СБИС схемы с тысячами и миллионами логических элементов стали размещаться на одном кристалле (сейчас объявлено о возможности разместить на кристалле миллиард транзисторов). При этом проблема снижения универсальности для ИС с жесткой структурой обострилась бы чрезвычайно — пришлось бы производить огромное число типов ИС при снижении объема производства каждого из типов, что непомерно увеличило бы их стоимость, т. к. высокие затраты на проектирование БИС/СБИС относились бы к небольшому объему их выпуска.

Выход из возникшего противоречия был найден на пути переноса специализации микросхем в область программирования. Появились микропроцессоры и БИС/СБИС с программируемой структурой.

Микропроцессор способен выполнять команды, входящие в его систему команд. Меняя последовательность и состав команд (программу), можно решать различные задачи на одном и том же микропроцессоре. Иначе говоря, в этом случае структура аппаратных средств не связана с характером решаемой задачи. Это обеспечивает микропроцессорам массовое производство с соответствующим снижением стоимости.

В виде БИС/СБИС с программируемой структурой потребителю предлагается кристалл, содержащий множество логических блоков, межсоединения для которых назначает сам системотехник. Промышленность получает возмож-

ность производить кристаллы массовым тиражом, не адресуясь к отдельным потребителям. Системотехник сам программирует структуру ИС соответственно своему проекту. Разработан целый спектр методов программирования связей между блоками и элементами кристалла.

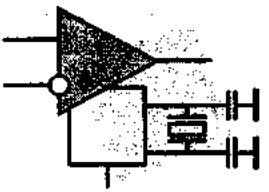
Два указанных метода имеют большие различия. Микропроцессоры реализуют последовательную обработку информации, выполняя большое число отдельных действий, соответствующих командам, что может не обеспечить требуемого быстродействия. В БИС/СБИС с программируемой структурой обработка информации возможна без разбиения этого процесса на последовательно выполняемые элементарные действия. Задача может решаться "целиком", ее характер определяет структуру устройства. Преобразование данных происходит одновременно во многих частях устройства. Сложность устройства зависит от сложности решаемой задачи, чего нет в микропроцессорных системах, где сложность задачи влияет лишь на программу, а не на аппаратные средства ее выполнения.

Таким образом, БИС/СБИС с программируемой структурой могут быстрее решать задачи, сложность которых ограничена уровнем интеграции микросхем, а микропроцессорные средства — задачи неограниченной сложности, но с меньшим быстродействием. Оба направления открывают ценные перспективы дальнейшего улучшения технико-экономических показателей создаваемой на них аппаратуры. Более того, современные кристаллы высшего уровня интеграции содержат одновременно и микропроцессоры, и большие массивы программируемой логики, обладая в силу этого большими функциональными возможностями. Подобная структура свойственна микросхемам класса "система на кристалле", важная роль которых в проектировании современной аппаратуры неоспорима.

С ростом уровня интеграции ИС в проектировании на их основе все больше усиливается аспект, который можно назвать интерфейсным проектированием. Задачей разработки становится составление блоков из субблоков стандартного вида путем правильного их соединения. Успешное проектирование требует хорошего знания номенклатуры и параметров элементов, узлов и устройств цифровой аппаратуры и привлечения *систем автоматизированного проектирования* (САПР) для создания сложных систем.

ИС широкого применения изготовляются по схемотехнологиям КМОП, ТТЛШ и др. Схемотехнологии КМОП и ТТЛШ стали основными. Современные элементы КМОП обладают рядом уникальных параметров (малая потребляемая мощность, особенно при невысоких частотах переключения, высокая помехоустойчивость, широкие допуски на величину питающих напряжений, высокое быстродействие, особенно при небольших емкостных нагрузках). Эти элементы доминируют в схемах внутренних областей БИС/СБИС. За ТТЛШ осталась пока область периферийных схем, где требуется передача сигналов по внешним цепям, испытывающим большую емкостную нагрузку.

Глава 1



Схемотехнические проблемы построения цифровых узлов и устройств

§ 1.1. Простейшие модели и система параметров логических элементов

Простейшие модели логических элементов

Даже самые сложные преобразования цифровой информации в конечном счете сводятся к простейшим операциям над логическими переменными 0 и 1. Такие операции реализуются логическими элементами в соответствии с формулами алгебры логики. В идеализированных схемах логические элементы могут быть представлены моделями вида (рис. 1.1, *а*), т. е. *условными графическими обозначениями* — прямоугольниками, в которых ставится символ выполняемой операции, а на линиях входных и выходных переменных могут изображаться кружки (индикаторы инверсии), если данная переменная входит в формулу зависимости выходной переменной от входных в инверсном виде.

Рис. 1.1. Обозначение идеализированного логического элемента (*а*) и модель логического элемента с фиксированной задержкой (*б*)

В реальных условиях логические переменные 0 и 1 отображаются, как правило, двумя различными уровнями напряжения: U_0 и U_1. Переход от логических переменных к электрическим сигналам ставит вопрос о логических соглашениях. Необходимо условиться, какой из двух уровней напряжения принять за U_0 и какой за U_1. Существуют *соглашения положительной и отрицательной логики*. В положительной логике $U_1 > U_0$, а в отрицательной $U_1 < U_0$. Один и тот же элемент, в зависимости от принятого логического соглашения, выполняет различные логические операции. Переход от операции в положительной логике к операции в отрицательной производится инвертированием всех переменных.

В дальнейшем, если не оговорено иное, будем пользоваться соглашением положительной логики.

Наряду с обозначениями U_1 и U_0 могут быть использованы и обозначения высокого и низкого уровней напряжения соответственно как H (High) и L (Low).

Одни и те же преобразования логических переменных можно задать в различных формах: с помощью операций И, ИЛИ, НЕ (булевский базис), операции И-НЕ (базис Шеффера), операции ИЛИ-НЕ (базис Пирса), а также многими другими способами. Выбор базиса зависит от простоты реализации той или иной операции с помощью электрических схем данной схемотехнологии. Чаще всего встречаются базисы Шеффера и Пирса. В развитых сериях стандартных ИС наряду с базовыми логическими элементами обычно имеется и ряд других, выполняющих другие логические операции.

Быстродействие или даже работоспособность ЦУ зависит от задержек сигналов в логических элементах и линиях связей между ними. Реальные переходные процессы в логических элементах достаточно сложны, и в моделях они отображаются с той или иной степенью упрощения. В простейшей модели динамические свойства элемента отражаются введением в его выходную цепь элемента задержки сигнала на фиксированное время $t_з$ (рис. 1.1, *б*). В силу простоты такая модель находит применение на практике, несмотря на то, что она является грубой и не учитывает ряд существенных факторов: технологического разброса задержек элементов, зависимости их от направления переключения элемента (из 0 в 1 или из 1 в 0), зависимости их от емкостной нагрузки, которая может быть резко выраженной (например, для элементов КМОП задержка пропорциональна емкости нагрузки) и т. д. Простейшая модель не учитывает также фильтрующих свойств реальных элементов, благодаря которым короткие входные импульсы, обладающие малой энергией, не способны вызвать переключение элемента, даже если их амплитуда велика.

Применение более точных моделей сигналов и задержек сопровождается усложнением расчетов при анализе работы ЦУ и характерно для САПР.

Пользуясь для описания сигналов алфавитом всего из двух символов {0,1}, нельзя, естественно, точно описать форму сигнала, т. к. в этом случае он может быть представлен только в виде идеально прямоугольного, тогда как реальные сигналы обязательно имеют интервалы переходных процессов при переключениях в направлениях 0 > 1 и 1 > 0 (если для упрощения линеаризовать участки переходных процессов, то сигналы можно представлять в форме трапеций). Для сигналов с областями переходных процессов *алфавит*, т. е. число символов, используемых для описания формы сигнала, расширится по меньшей мере до трех: 0, 1, X, где X обозначает неопределенное состояние во время переходного процесса. Часто к этим трем символам добавляется четвертый символ Z, отображающий присущий некоторым элементам режим "отключено" или, иначе, режим "высокого импеданса". Четырехсимвольный алфавит {0, 1, X, Z} применяется в ряде систем автоматизированного проектирования (САПР).

Для правильного представления режимов в схемах с источниками сигналов, имеющих существенно различные выходные сопротивления, вводится понятие *силы сигнала*, благодаря которому можно определить сигнал в точках соединения выходов, обладающих разной силой.

Вводя в четырехсимвольный алфавит символы слабого нуля L, слабой единицы H и слабого неопределенного состояния W, приходят к семи символам в составе алфавита. Добавив еще два символа ("не инициализировано" и "не важно"), необходимые для организации процесса моделирования работы функционально-логической схемы устройства, получают часто применяемый девятисимвольный алфавит. В некоторых случаях задачи правильного отображения сигналов решаются путем представления элемента задержки в виде цепочки нескольких динамических звеньев, каждое из которых отображает то или иное существенное свойство элемента с точки зрения динамики его поведения.

Расширение алфавита при описании сигналов позволяет приближать модельное описание процессов в цифровых схемах к реальным. В то же время усложнение моделей существенно увеличивает время моделирования и объем проводимых в его ходе вычислений. Поэтому в системах моделирования на разных этапах работы могут применяться модели сигналов разной сложности — простые для быстрого неточного моделирования и более сложные с расширенными алфавитами для более адекватного описания процессов в схемах.

Рассмотрение множества вариантов описания цифровых сигналов не входит в задачи этой главы, но о некоторых моментах стоит упомянуть. Цель усложнения моделей элементов — точнее отобразить временные соотношения сигналов в анализируемой схеме, выявить возможные временные состязания. В этом направлении важным достижением явилось предложенное Эйхельбергером троичное моделирование, выявляющее критические временные состязания для схем с произвольным соотношением задержек в элементах схемы. Поскольку реально задержки находятся в определенных пределах, а не в интервале от нуля до бесконечности, метод Эйхельбергера дает слишком пессимистические результаты и обнаруживает критические временные состязания в том числе и там, где фактически их не будет. Для устранения отмеченного недостатка были разработаны методы Δ-троичного моделирования, троичного моделирования с нарастающей неопределенностью и т. д. Для троичного моделирования по Эйхельбергеру используется алфавит {0, X, 1}, где X — неопределенное значение сигнала на интервале переходных процессов переключе-

ния элемента. Операции ИЛИ, И и НЕ троичной алгебры реализуются при этом согласно соотношениям, приведенным далее.

$0 \vee 0 = 0$	$0 \wedge 0 = 0$	$\bar{0} = 1$
$0 \vee 1 = 1$	$0 \wedge 1 = 0$	$\bar{1} = 0$
$0 \vee X = X$	$0 \wedge X = 0$	$\bar{X} = X$
$1 \vee 1 = 1$	$1 \wedge 1 = 1$	
$1 \vee X = 1$	$1 \wedge X = X$	

Для других операций троичной алгебры также существуют соотношения, выражаемые через обычные операции логического сложения, умножения и инверсии.

Приближенное к реальности описание сигналов дают *средства аналогового моделирования* процессов в логических элементах, использующие аппарат дифференциальных уравнений (например, известная программа SPICE), но объем вычислений при этом настолько возрастает, что подобное моделирование применяется для проектирования электрических схем уровня отдельных элементов или несложных фрагментов.

Статические параметры логических элементов

Для правильного проектирования и эксплуатации ЦУ необходимо знать систему параметров логических элементов (статических и динамических). В качестве важнейших статических параметров приводятся пять значений напряжений и пять значений токов.

Прежде всего указывается значение напряжения питания U_{CC} (величина и поле допуска).

Четыре значения напряжений задают границы отображения переменных (0 и 1) на выходе и входе элемента. Для нормальной работы элемента требуется, чтобы напряжение, отображающее логическую 1, было достаточно высоким, а напряжение, отображающее 0, — достаточно низким. Эти требования задаются параметрами $U_{вх.1.min}$ и $U_{вх.0.max}$. Входные напряжения данного элемента есть выходные напряжения предыдущего (источника сигналов). Уровни, гарантируемые на выходе элемента при соблюдении допустимых нагрузочных условий, задаются параметрами $U_{вых.1.min}$ и $U_{вых.0.max}$. Выходные уровни несколько "лучше" входных, что обеспечивает определенную помехоустойчивость элемента (рис. 1.2).

Для уровня U_1 опасны отрицательные помехи, снижающие его, причем допустимая статическая помеха (т. е. помеха любой длительности)

$$U^{-}_{пом} = U_{вых.1.min} - U_{вх.1.min}.$$

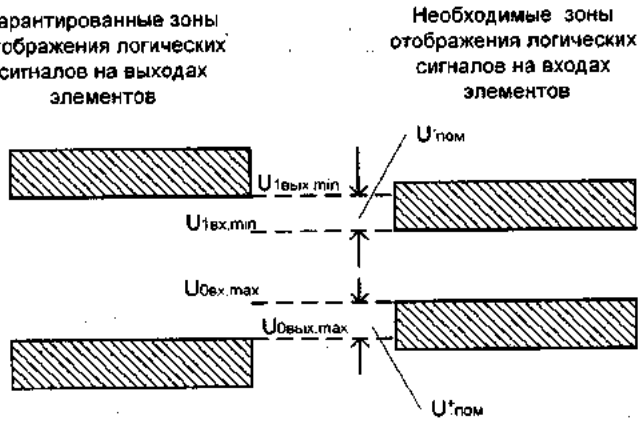


Рис. 1.2. Зоны отображения сигналов на входах и выходах логических элементов

Для уровня U_0 опасны положительные помехи, причем допустимая статическая помеха

$$U^{+}_{пом} = U_{вых.0.max} - U_{вх.0.max}.$$

Для токов в первую очередь указывается ток потребления, который нужен и для определения потребляемой элементом мощности, рассчитываемой как произведение напряжения питания элемента на потребляемый им ток.

Следующие четыре значения токов среди важнейших статистических параметров — входные и выходные токи в обоих логических состояниях. При высоком уровне выходного напряжения из элемента-источника ток вытекает, в то время как цепи нагрузки ток поглощают. При низком уровне выходного напряжения элемента-источника ток нагрузки втекает в этот элемент, а из входных цепей элементов-приемников токи вытекают. Зная токи $I_{вых.1.max}$ и $I_{вых.0.max}$, характеризующие возможности элемента-источника сигнала, и токи $I_{вх.1.max}$ и $I_{вх.0.max}$, потребляемые элементами-приемниками, можно контролировать соблюдение нагрузочных ограничений, обязательное для всех элементов схемы ЦУ.

Быстродействие логических элементов

Быстродействие логических элементов определяется скоростями их перехода из одного состояния в другое. Быстродействие ЦУ определяется задержками сигналов, как в логических элементах, так и в цепях их межсоединений.

Временные диаграммы переключения инвертирующего логического элемента (рис. 1.3) показывают длительности характерных этапов переходных про-

цессов, отсчитываемые по так называемым измерительным уровням. Моментом изменения логического сигнала считают момент достижения им порогового уровня. Часто за пороговый уровень принимают середину логического перепада сигнала, т. е. $0{,}5(U_0 + U_1)$. Иногда пороговый уровень указывается более точно в паспортных данных элемента. На временных диаграммах показаны задержки распространения сигнала при изменении выходного напряжения элемента от U_1 до U_0 и обратно (t^{10} и t^{01}). Очень часто для упрощения расчетов пользуются усредненным значением задержки распространения сигнала $t_з = 0{,}5(t^{10} + t^{01})$.

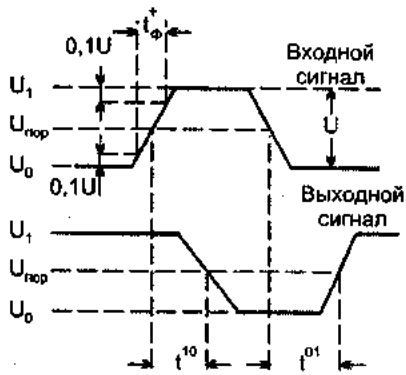


Рис. 1.3. Временные диаграммы процессов переключения логического элемента

Следует обратить внимание на то, что усреднение согласно приведенному соотношению не относится к технологическому разбросу задержек. Также следует заметить, что справочные данные о задержках соответствуют определенным условиям измерений, указанным в справочниках. Если условия работы элемента отличаются от условий измерения, то может потребоваться коррекция справочных данных.

На быстродействие ЦУ влияют также емкости, на перезаряд которых требуются затраты времени. В справочных данных приводятся входные и выходные емкости логических элементов, знание которых позволяет подсчитать емкости нагрузки в узлах схемы. Для подключаемой к выходу элемента емкости приводятся две цифры: номинальная емкость C_L (L от Load) и предельно допустимая емкость C_{max}. Первая емкость соответствует условиям измерения задержек сигналов, так что именно для нее справедливы значения задержек сигналов, приведенные в справочных данных. Если реальная нагрузочная емкость отличается от номинальной, то изменятся и значения задержек. Значения реальных задержек можно оценить с помощью соотношения $t_з = t_{з.н} + k\Delta C$, где $t_{з.н}$ — номинальное значение задержки; $\Delta C = C - C_L$;

C — фактическое значение нагрузочной емкости; k — коэффициент, величина которого задается для каждой серии элементов индивидуально.

Предельно допустимая емкость указывает границу, которую нельзя нарушать, поскольку при этом работоспособность элемента не гарантируется.

Разумеется, при подсчете емкостей в узлах ЦУ учитываются и емкости межсоединений (монтажные емкости).

Мощности потребления логических элементов

При разработке ЦУ требуется оценивать мощности их потребления, чтобы сформулировать требования к источникам питания, оценить температурный режим устройства и конструкцию теплоотвода. При этом суммируются мощности, рассеиваемые логическими и другими элементами схемы, а также межсоединениями.

Мощности, потребляемые элементами, делят на *статические и динамические*. Статическая мощность потребляется элементом, который не переключается. При переключении потребляется дополнительно динамическая мощность, которая пропорциональна частоте переключения элемента. Таким образом, полная мощность зависит от частоты переключения элемента, что и следует учитывать при ее подсчете, особенно для схем типа КМОП. При подсчете мощностей может не хватить данных, приведенных в обычных кратких справочниках, (отметим, что справочник под редакцией И. И. Петровского [26] предоставляет достаточные данные для расчета мощностей ЦУ на элементах КМОП для серии элементов КР1554).

§ 1.2. Типы выходов цифровых элементов

Цифровые элементы (логические, запоминающие, буферные) могут иметь выходы следующих типов:

- логические;
- с открытым коллектором (стоком);
- с третьим состоянием;
- с открытым эмиттером (истоком).

Наличие четырех типов выходов объясняется различными условиями работы элементов в логических цепях, магистрально-модульных системах и т. д.

Логический выход

Логический выход формирует два уровня выходного напряжения (U_0 и U_1). Выходное сопротивление логического выхода стремятся сделать малым, способным развивать большие токи для перезаряда емкостных нагрузок и,

следовательно, получения высокого быстродействия элемента. Такой тип выхода имеют большинство логических элементов, используемых в операционных устройствах.

Схемы логических выходов элементов ТТЛ(Ш) и КМОП подобны двухтактным каскадам — в них оба фронта выходного напряжения формируются с участием активных транзисторов, работающих противофазно, что обеспечивает малые выходные сопротивления при любом направлении переключения выхода (рис. 1.4, *а, б*).

Особенность таких выходов состоит в том, что их нельзя соединять параллельно. Во-первых, это создает логическую неопределенность, т. к. в точке соединения выхода, формирующего логическую единицу, и выхода, формирующего логический нуль, не будет нормального результата. Во-вторых, при соединении выходов, находящихся в различных логических состояниях, возникает их "противоборство". Вследствие малых величин выходных сопротивлений уравнительный ток при этом может достигать большой величины, что может вывести из строя элементы выходных цепей.

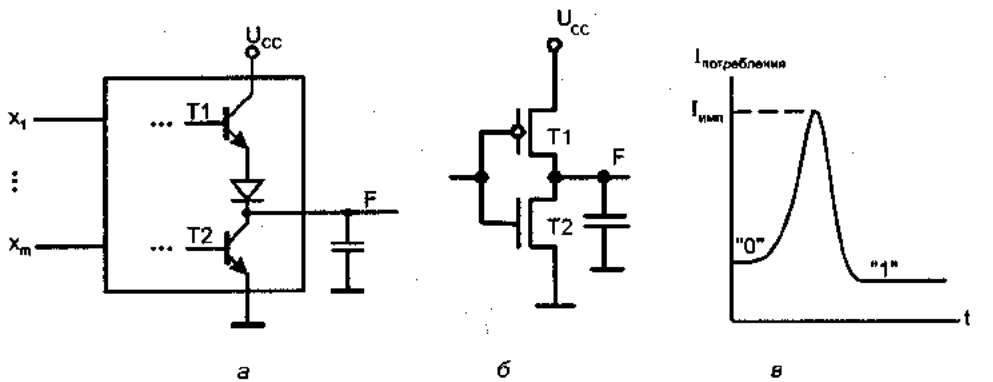


Рис. 1.4. Схемы логических выходов цифровых элементов (*а, б*) и график изменения потребляемого ими тока в процессе переключения (*в*)

Вторая особенность логического выхода двухтактного типа связана с протеканием через оба транзистора коротких импульсов тока при переключениях из одного логического состояния в другое. Эти токи протекают от источника питания на общую точку ("землю"). В статических состояниях таких токов быть не может, т. к. транзисторы Т1 и Т2 работают в противофазе, и один из них всегда заперт. Однако в переходном процессе из-за некоторой несинхронности переключения транзисторов возникает кратковременная ситуация, в которой проводят оба транзистора, что и порождает короткий *импульс сквозного тока* значительной величины (рис. 1.4, *в*). Следует особо отметить, что импульсные токи возникают в цепях питания логических эле-

ментов не только из-за описанного выше явления, но и *вследствие перезаряда емкостей* при их переключениях. Это важное обстоятельство свидетельствует о *неустранимости* такого неприятного факта, как наличие *импульсных токовых помех в цепях питания цифровых элементов*.

Выход с тремя состояниями

Элементы с тремя состояниями выхода (выходами типа ТС или Z) кроме логических состояний 0 и 1 имеют состояние "отключено", в котором ток выходной цепи пренебрежимо мал. В это состояние (третье или Z-состояние) элемент переводится специальным управляющим сигналом, обеспечивающим запертое состояние обоих транзисторов выходного каскада (Т1 и Т2 на рис. 1.4, *а, б*). Сигнал управления элементом типа ТС обычно обозначается как OE (Output Enable). При наличии разрешения (OE = 1) элемент работает как обычно, выполняя свою логическую операцию, а при его отсутствии (OE = 0) переходит в состояние "отключено".

На рис. 1.5 показан выходной каскад с третьим состоянием, используемый в схемотехнике КМОП. Высокий уровень сигнала OE означает разрешение работы, он открывает транзисторы Т3 и Т4 и, тем самым, позволяет нормально работать инвертору на транзисторах Т1 и Т2, через который данные DO передаются на выход. При низком уровне сигнала OE транзисторы Т3 и Т4 заперты и выход DO находится в состоянии "отключено".

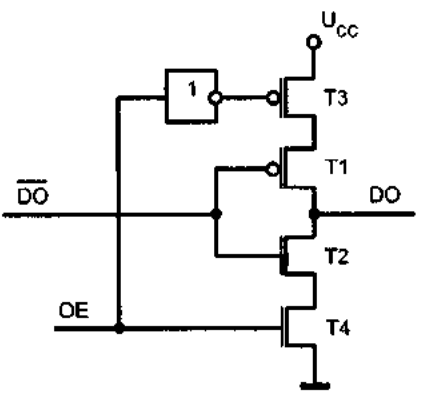


Рис. 1.5. КМОП-инвертор с тремя состояниями выхода

В ЦУ широко используются буферные элементы типа ТС для управляемой передачи сигналов по тем или иным линиям. Буферы могут быть неинвертирующими или инвертирующими, а сигналы OE — Н-активными или

L-активными, что ведет к наличию четырех типов буферных каскадов, показанных на рис. 1.6 в двух вариантах обозначений.

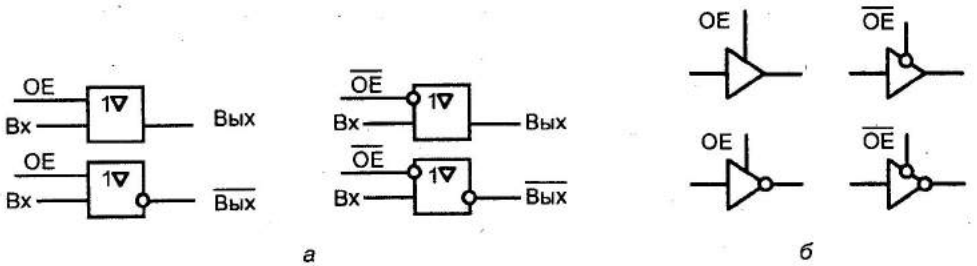


Рис. 1.6. Типы буферных каскадов с третьим состоянием

Выходы типа ТС отмечаются в обозначениях элементов значком треугольника, как на рис. 1.6, или буквой Z (при выполнении документации с помощью устройств вывода ЭВМ).

Выходы типа ТС можно соединять параллельно при условии, что в любой момент времени активным может быть только один из них. В этом случае отключенные выходы не мешают активному формировать сигналы в точке соединения выходов. Эта возможность позволяет применять элементы типа ТС в магистрально-модульных микропроцессорных и иных системах, где многие источники информации поочередно пользуются одной и той же линией связи.

Элементы типа ТС сохраняют такие достоинства элементов с логическим выходом, как быстродействие и высокая нагрузочная способность. Поэтому они являются основными в указанных применениях. В то же время они требуют обязательного соблюдения условия отключения всех выходов, соединенных параллельно, кроме одного, т. е. условия $OE_1 + OE_2 + ... + OE_n \le 1$ при объединении n выходов. Нарушение этого условия может привести даже к выходу из строя самих элементов. Кроме того, в отличие от выходов с открытым коллектором или стоком, выходы типа ТС не дают возможности выполнять операции монтажной логики.

Элементы типа ТС приобрели широкое распространение в современной схемотехнике, особенно в связи с массовым применением микропроцессорных систем с их магистрально-модульными структурами. Однако в самые последние годы в связи с появлением так называемых "систем на кристалле", в которых все части создаваемого устройства расположены на одном и том же кристалле, шины с тремя состояниями стали менее эффективными и задача поочередного использования одной и той же линии зачастую стала решаться иначе — с помощью мультиплексоров согласно принципу, показанному на рис. 1.7.

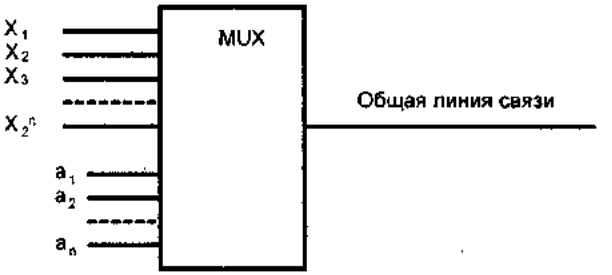


Рис.1.7. Схема мультиплексируемой линии передачи, используемой несколькими источниками сигнала в режиме разделения времени

В схемах типа рис. 1.7 несколько входных информационных сигналов X_i присутствуют на соответствующих входах ($i = 1, 2, \ldots, 2^n$). На выход мультиплексора пропускается лишь один из этих входов в зависимости от адресующего кода $a_1 \ldots a_n$. Подробнее работа мультиплексора рассмотрена в § 2.4.

Выход с открытым коллектором (стоком)

Элементы с открытым коллектором (типа ОК) или стоком (типа ОС) имеют выходную цепь, заканчивающуюся одиночным транзистором, коллектор которого не соединен с какими-либо цепями внутри микросхемы (рис. 1.8, *а*). Точно так же элементы с открытым стоком (типа ОС), используемые в схемотехнике КМОП, имеют выходную цепь в виде одиночного МОП-транзистора, сток которого разомкнут (рис. 1.8, *б*). В английской терминологии выход с открытым стоком обозначается как OD (Open Drain). Транзистор управляется от предыдущей части схемы элемента так, что может находиться в насыщенном (для МОП-транзистора просто открытом) или запертом состоянии. Насыщенное (открытое) состояние трактуется как отображение логического нуля, запертое — единицы.

С точки зрения логических состояний, схемы с открытым коллектором (ОК) и открытым стоком (ОС) равноценны, но их электрические режимы различны. В схемах типа КМОП входные токи элементов пренебрежимо малы, тогда как в схемах на биполярных транзисторах (ТТЛШ) с ними нельзя не считаться. Это обстоятельство делает анализ и проектирование для схем типа ОК более сложным, чем для схем ОС, поэтому ниже рассматривается схема ОК. Подход к расчету сопротивления внешнего резистора для схем с открытым стоком аналогичен изложенному для варианта ОК, но расчетные формулы могут быть упрощены, поскольку входными токами МОП-транзисторов можно пренебречь.

Насыщение транзистора в схеме с открытым коллектором обеспечивает на выходе напряжение U_0 (малое напряжение насыщения "коллектор-эмиттер" $U_{кэн}$). Запирание же транзистора какого-либо уровня напряжения на выходе

НТБ ВНТУ

элемента не задает, выход при этом имеет фактически неизвестный "плавающий" потенциал, т. к. он не подключен к каким-либо цепям. Поэтому для формирования высокого уровня напряжения при запирании транзистора на выходе элементов с открытым коллектором или стоком требуется подключать внешние резисторы (или другие нагрузки), соединенные с источником питания.

Несколько выходов типа ОК можно соединять параллельно, подключая их к общей для всех выходов цепочке U_{CC}—R (рис. 1.8, *в*). При этом можно получить режим поочередной работы элементов на общую линию, как и для элементов типа ТС, если активным будет лишь один элемент, а выходы всех остальных окажутся запертыми. Если же разрешить активную работу элементов, выходы которых соединены, то можно получить дополнительную логическую операцию, называемую операцией *монтажной логики*.

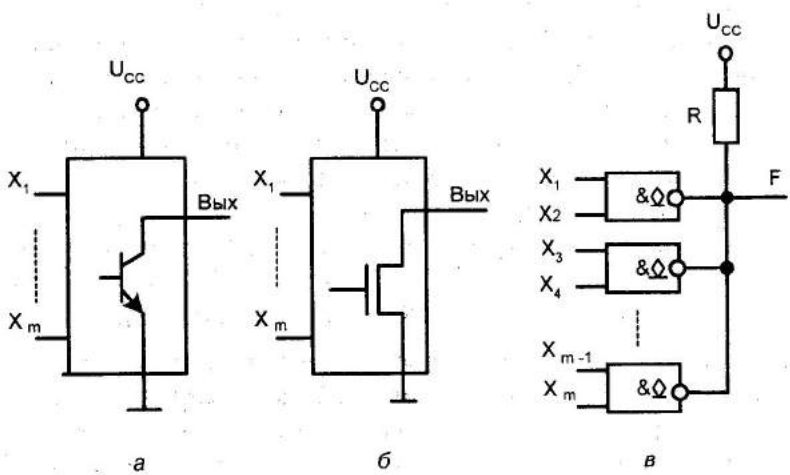


Рис. 1.8. Схема выходной цепи цифрового элемента с открытым коллектором (*а*), открытым стоком (*б*) и реализация монтажной логики (*в*)

При реализации монтажной логики высокое напряжение на общем выходе возникает только при запирании всех транзисторов, т. к. насыщение хотя бы одного из них снижает выходное напряжение до уровня $U_0 = U_{кэн}$. То есть для получения логической единицы на выходе требуется единичное состояние всех выходов: выполняется монтажная операция И. Поскольку каждый элемент выполняет операцию Шеффера над своими входными переменными, общий результат окажется следующим

$$F = \overline{x_1 x_2}\ \overline{x_3 x_4} \ldots \overline{x_{m-1} x_m} = \overline{x_1 x_2 \vee x_3 x_4 \vee \ldots \vee x_{m-1} x_m}.$$

В обозначениях элементов с ОК или ОС после символа функции ставится ромб с черточкой снизу.

При использовании элементов с ОК или ОС в магистрально-модульных структурах требуется разрешать или запрещать работу того или иного элемента. Для элементов типа ТС это делалось с помощью специального сигнала ОЕ. Для элементов типа ОК или ОС в качестве входа ОЕ может быть использован один из обычных входов элемента. Если речь идет об элементе И-НЕ, то, подавая 0 на любой из входов, можно запретить работу элемента, поставив его выход в разомкнутое состояние независимо от состояния других входов. Уровень 1 на этом управляющем входе разрешит работу элемента.

Положительной чертой элементов с ОК или ОС при работе в магистрально-модульных системах является их защищенность от повреждений из-за ошибок управления, приводящих к одновременной выдаче на общую шину нескольких слов, а также возможность реализации дополнительных операций монтажной логики. Недостатком таких элементов является большая задержка переключения из 0 в 1. При этом переключении выходная емкость заряжается сравнительно малым током резистора R. Сопротивление резистора нельзя сделать слишком малым, т. к. это привело бы к большим токам выходной цепи в статике при насыщенном состоянии выходного транзистора. Поэтому положительный фронт выходного напряжения формируется относительно медленно с постоянной времени RC. До порогового напряжения (до середины полного перепада напряжения) экспоненциально изменяющийся сигнал изменится за время 0,7RC, что и составляет задержку $t_з^{01}$.

При работе с элементами типа ОК проектировщик должен задать сопротивление резистора R, которое не является стандартным, а определяется для конкретных условий. Статические режимы задают ограничения R снизу и сверху. Сопротивление R выбирается в этом диапазоне с учетом быстродействия схемы и потребляемой ею мощности.

Ограничение сопротивления R снизу связано с тем, что его уменьшение может вызвать перегрузку насыщенного транзистора по току. На рис. 1.9 показан режим, в котором нулевое состояние выхода обеспечивается одним логическим элементом (ЛЭ) с ОК (или ОС).

Из этого рисунка видно, что через выход элемента протекает суммарный ток, складывающийся из токов резистора I_R, входных токов n логических элементов, подключенных к выходу элемента ЛЭ и токов (m − 1) запертых транзисторов элементов, включенных параллельно с ЛЭ, каждый из которых обозначен как I_z, т. е.

$$I_{вых.0} = I_R + nI_{вх.0} + (m - 1)I_z \approx I_R + nI_{вх.0},$$

где $I_{вх.0}$ — входные токи элементов — приемников сигнала при низком уровне входных напряжений; I_z — токи запертых выходов ОК (обычно пренебрежимо малые); $I_R = (U_{CC} - U_0)/R$. Чтобы ток выхода элемента 1 не превысил допустимого значения $I_{вых.0.max}$, следует соблюдать условие

$$R \geq (U_{CC} - U_0)/(I_{вых.0.max} - nI_{вх.0.max} - (m - 1)I_z).$$

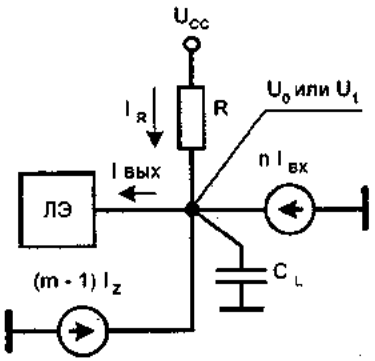


Рис. 1.9. Схема для расчета минимального и максимального значений сопротивления внешней цепи в каскадах с открытым коллектором (стоком)

Ограничение R сверху связано с необходимостью гарантировать достаточно высокий уровень напряжения U_1, формируемого в схеме при запертом состоянии всех выходов элементов с ОК. Из схемы видно, что $U_1 = U_{CC} - I_R R$, и для поддержания достаточно высокого уровня напряжения U_1 сопротивление R должно быть не слишком большим. Проанализировать режим схемы при единичном состоянии ее выхода можно по той же схеме (см. рис. 1.9), приняв во внимание, что токи I_z и входные токи элементов-нагрузок изменят направление на противоположное. В этом случае

$$I_R = mI_z + nI_{вх.1.max}.$$

Для допустимого значения R получим

$$R \leq (U_{CC} - U_{вых.1.min})/(mI_z + nI_{вх.1.max}),$$

где $U_{вых.1.min}$ — паспортный параметр элемента.

Имея границы диапазона значений сопротивления резистора R, полученные, как показано ранее, проектировщик должен выбрать некоторое конкретное его значение. Выбор вблизи нижней границы улучшает быстродействие схемы, а выбор вблизи верхней уменьшает потребляемую схемой мощность.

Выход с программированием ТС—ОС (с тремя состояниями или с открытым стоком)

Во многих современных микросхемах используются выходные каскады с возможностью их программирования (настройки) на работу в одном из двух вариантов — либо как каскада с открытым выходным электродом (коллектором или стоком), либо как каскада с третьим состоянием. Такие возможности часто весьма полезны, поскольку каждый из типов выходов имеет свои особенности. На рис. 1.10 приведена схема каскада с программированием ТС—ОС.

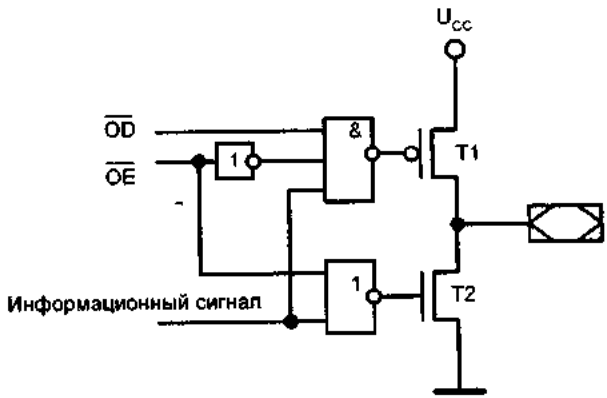


Рис. 1.10. Выходной каскад, программируемый на варианты ТС и ОС

При пассивном высоком уровне сигнала OD (Open Drain, "Открытый сток") выход работает как выход с третьим состоянием, поскольку при активном низком уровне сигнала OE (Output Enable, "Разрешение выхода") информационный сигнал поступает на управление выходным инвертором на транзисторах T1 и T2. Единичное значение информационного сигнала инвертируется элементами И-НЕ и ИЛИ-НЕ, отпирает транзистор T1 и запирает транзистор T2, что формирует на выходе сигнал логической единицы. Нулевое значение информационного сигнала приводит транзисторы в противоположное состояние и формирует на выходе сигнал логического нуля. При активном низком уровне сигнала OD на затворе транзистора T1 действует высокое напряжение, и он заперт, что вводит режим открытого стока для транзистора T2. При пассивном высоком уровне сигнала OE (отсутствии разрешения работы) на затвор транзистора T1 поступает высокий уровень напряжения, а на затвор транзистора T2 — низкий, т. е. оба транзистора заперты.

Выход с открытым эмиттером

Выход с открытым эмиттером характерен для элементов типа ЭСЛ (эмиттерно-связанная логика). Возможность соединять друг с другом выходы с открытым эмиттером приводит к получению дополнительной операции монтажной логики ИЛИ. Элементы ЭСЛ имеют противофазные выходы, на одном из которых (прямом) реализуется функция ИЛИ, на другом (инверсном) — ИЛИ-НЕ. Соединяя прямые выходы нескольких элементов, получают расширение по ИЛИ (входные переменные соединяемых элементов образуют единую дизъюнкцию). Соединяя инверсные выходы, получают операцию И-ИЛИ относительно инверсий входных переменных, т. к. при этом

$$F = \overline{x_1 \vee x_2} \vee \overline{x_3 \vee x_4} = \bar{x}_1\bar{x}_2 \vee \bar{x}_3\bar{x}_4.$$

Соединяя прямой выход с инверсным, можно получить функцию вида

$$F = x_1 \vee x_2 \vee \overline{x_3 \vee x_4} = x_1 \vee x_2 \vee \bar{x}_3\bar{x}_4.$$

§ 1.3. Схемотехника входных цепей КМОП-элементов и режимы временно разомкнутых выводов

Выводы микросхем с pull-up- и pull-down-резисторами

Рассмотрим ситуацию с выводами, которые в данное время не используются и не могут быть постоянно подключены к напряжению питания или общей точке схемы, поскольку это исключило бы возможность их последующего перевода в активный режим (подачу на них информационных сигналов). Оставлять высокоомные выводы разомкнутыми также нельзя, поскольку на них могут наводиться произвольные потенциалы, что чревато опасными последствиями. Произвольные потенциалы входов неприемлемы, потому что подобные потенциалы имитируют на входах непредусмотренные сигналы, а это не позволяет правильно задавать состояние элементов с помощью других выводов. Кроме того, напряжение на входе может принять некоторое промежуточное значение (в том числе близкое к пороговым напряжениям), при котором в схеме могут возникнуть статические сквозные токи и, следовательно, будет потребляться большая паразитная мощность. Поэтому принято *фиксировать потенциалы временно разомкнутых выводов слабыми сигналами,* определяющими потенциалы выводов в отсутствие информационных сигналов и не играющими заметной роли при их присутствии. На рис. 1.11 показаны фрагменты типичных цепей, применяемых во входных схемах КМОП-элементов, в том числе цепи, иллюстрирующие взаимодействие слабого и сильного сигналов.

На рис. 1.11, *а* показаны традиционные цепи защиты входа транзистора от перенапряжений. Благодаря включению диодов диапазон изменения напряжений на затворе транзистора ограничен уровнями, близкими к напряжению питания U_{CC} и "земли". Рис. 1.11, *б* иллюстрирует взаимодействие слабого и сильного сигналов. Слабым является сигнал, подаваемый в точку U через высокоомный резистор R1 (для примера сопротивление R1 принято равным 100 кОм). Сильный сигнал подан в точку U через низкоомную цепь (для примера сопротивление R2 принято равным 100 Ом). Напряжение на выходе цепи согласно принципу суперпозиции можно выразить следующим образом

$$U = (U_1R_2 + U_2R_1)/(R_1 + R_2)$$

или

$$U = (U_1R_2/R_1 + U_2)\cdot R_1/(R_1 + R_2).$$

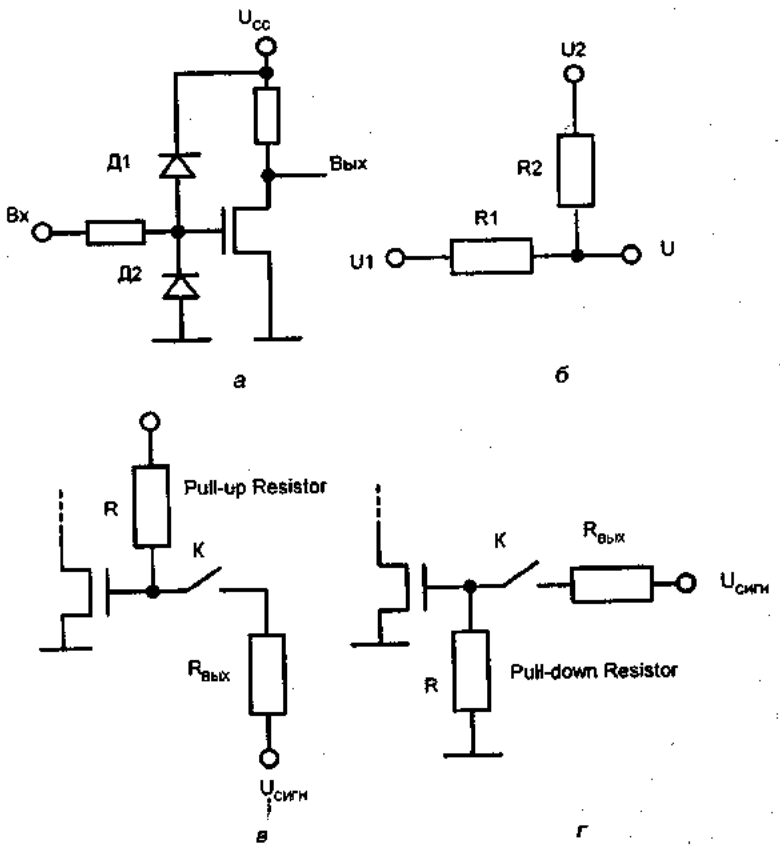


Рис. 1.11. Схемы защиты входа транзистора от перегрузки (*а*), взаимодействия сильного и слабого сигналов (*б*) и выводов с подтягивающим и заземляющим резисторами (*в*, *г*)

Для исключения паразитных потенциалов к точкам, в которых они могут возникать, подключают специальные резисторы, соединенные с источником питания или с общей точкой схемы. В первом случае это *"подтягивающие"* резисторы, называемые по-английски Pull-up Resistors, во втором — *"заземляющие"*, называемые Pull-down Resistors. Цепи с такими резисторами показаны на рис. 1.11, *в*, *г*. Если подтягивающие и заземляющие резисторы имеют большие сопротивления, то сигналы, формируемые с их участием, относятся к слабым. Когда к "подтянутым" или "заземленным" точкам подключаются сильные информационные сигналы, они преодолевают слабые сигналы, так что они практически не влияют на функционирование схем. Для взятых значений сопротивлений и напряжения питания схемы 5 В имеем следующее: в отсутствие информационных сигналов (ключ К разомкнут) на затворе МОП-транзистора напряжения равны U_{CC} или 0 для схем 1.11, *в*, *г* соответственно, а при подключении к выводу информационных сигналов (ключ К замкнут) сигнал на затворе в схеме с подтягивающим резистором составит $0{,}999U_{сигн} + 0{,}005$ В, а в схеме с заземляющим резистором $0{,}999U_{сигн}$. Как видно, сильный сигнал настолько доминирует над слабым, что его изменения из-за наличия цепочек фиксации потенциалов на разомкнутых выводах пренебрежимо малы. В то же время следует помнить, что цепи фиксации потенциалов создают дополнительные токовые нагрузки на источники информационных сигналов. Эти токи нагрузки в схеме с заземляющим резистором составляют U_1/R и U_0/R для единичного и нулевого состояний схемы, питающей рассматриваемый вывод (U_1 и U_0 — уровни кодирования 1 и 0). В схеме с подтягивающим резистором нагрузки для двух состояний схемы составляют соответственно $(U_{CC} - U_1)/R$ и $(U_{CC} - U_0)/R$.

В БИС/СБИС программируемой логики нередко к одним и тем же выводам подключают одновременно и подтягивающие, и заземляющие резисторы, последовательно с которыми включены ключевые транзисторы. При программировании выбирается вариант фиксации потенциала вывода и согласно этому выбору замыкается один из ключевых транзисторов.

Выводы микросхем с запоминанием последнего значения сигнала

Кроме варианта фиксации потенциалов временно неиспользуемых выводов с помощью подтягивающих и заземляющих резисторов применяют и схемы, которые позволяют сохранять на входном контакте прежнее значение потенциала после отключения контакта от источника сигнала (в английской терминологии такую схему называют Weak pin-keeper). Состояния контактов становятся предсказуемыми, потенциалы контактов соответствуют нормальным логическим уровням, схемы экономичны по потребляемой мощности. Схемы типа Weak pin-keeper основаны на применении обратной связи по слабому сигналу. Принцип их работы поясняется на рис. 1.12.

На рис. 1.12, *а* изображена цепь входа с автофиксатором. При подаче на вход логической единицы или нуля соответствующие напряжения дублируются сигналами на выходе второго инвертора. Если теперь отключить вход от информационного сигнала, состояние входа не изменится, т. к. дублирующее напряжение поддерживает само себя, поскольку пара инверторов, соединенных в кольцо, представляет собою бистабильную ячейку (простейший триггер), имеющую свойство памяти. Чтобы выход второго инвертора не шунтировал информационные сигналы, он должен быть высокоомным (в практических схемах это сотни килоом).

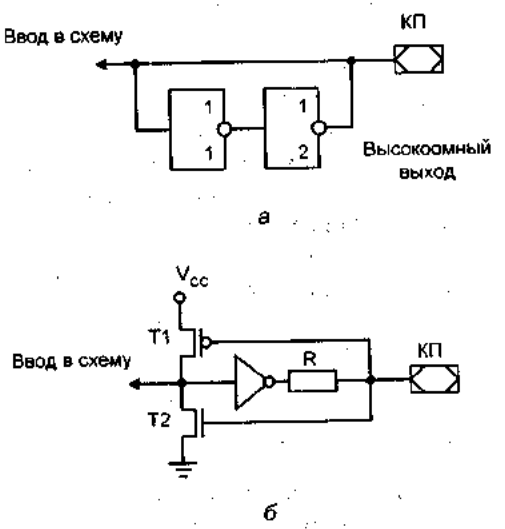


Рис. 1.12. Схемы входных цепей, обеспечивающие запоминание последнего значения информационного сигнала (а, б)

Другой вариант схемы типа Weak pin-keeper изображен на рис. 1.12, *б*. В этом варианте вводимые данные с контактной площадки подаются на затворы ключевых транзисторов T1 и T2. Высокий уровень напряжения открывает транзистор T2 с n-каналом и запирает транзистор T1 с p-каналом. При этом в линию входа подается нулевой сигнал, а в точку, связанную с контактной площадкой, — единичный, т. к. буфер является инвертирующим. Этот обратный сигнал подается на контактную площадку через высокоомное сопротивление R, благодаря чему он является слабым. Обычно сопротивление резистора R составляет не менее 100 кОм. При отключении источника сигнала от контактной площадки существовавший до отключения потенциал будет поддерживаться слабым сигналом, и вводимая величина не изменится. При подключении источника сигнала к контактной площадке он будет управлять состояниями транзисторов, преодолевая слабый

сигнал цепи "буфер—резистор". На преодоление действия слабого сигнала обычно затрачиваются токи порядка единиц микроампер.

§ 1.4. Паразитные связи цифровых элементов по цепям питания. Фильтрация питающих напряжений в схемах ЦУ

Одной из важнейших задач при проектировании и эксплуатации ЦУ является борьба со сбоями из-за помех. Типовой проблемой здесь является, в частности, наличие токовых импульсов в цепях питания ИС.

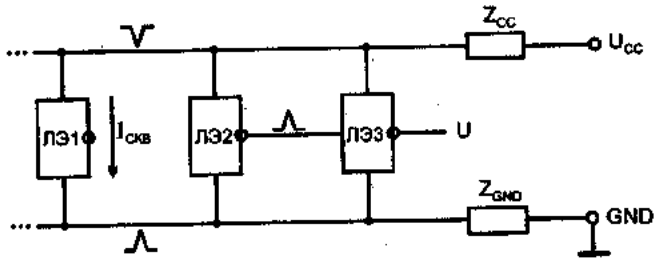
При переключениях элементов в цепях питания создаются кратковременные импульсные токи, благодаря чему сами элементы становятся источниками помех для соседних элементов. Токовые импульсы в цепях питания создаются упомянутыми в предыдущем параграфе сквозными токами выходных каскадов типов ТТЛ(Ш) и КМОП, а также токами перезаряда емкостей, что свойственно и всем другим типам элементов.

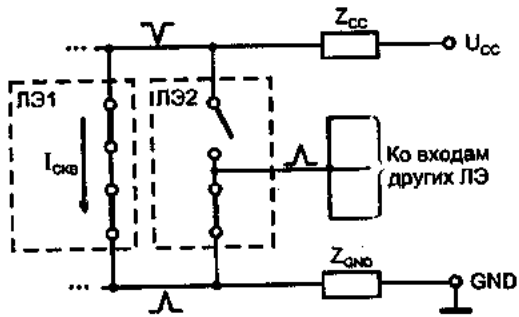
Для определенности далее будем говорить о сквозных токах, хотя практически то же самое можно говорить и о токах перезаряда емкостей.

Импульс сквозного тока переключающегося логического элемента 1 (ЛЭ1) (рис. 1.13, *а*) $I_{скв}$ протекает через транзисторы выходного каскада, условно изображенные замкнутыми ключами, от источника питания U_{CC} на общую точку схемы GND через линии, имеющие полные сопротивления Z_{cc}, и Z_{GND}. Главную часть сопротивлений составляют индуктивности линий, на которых выделяются напряжения $U_L = L\,di/dt$. Протекание сквозного тока создает на линии питания отрицательный импульс, а на линии общей точки ("земли") — положительный. Эти импульсы воздействуют на подключенный вблизи элемента ЛЭ1 элемент ЛЭ2. Если, как показано на рисунке, элемент ЛЭ2 находится в состоянии логического нуля, то его выход через насыщенный транзистор выходного каскада, отображаемый замкнутым ключом, связан с линией GND, следовательно, импульс с этой линии попадет на выход элемента ЛЭ2, откуда сможет распространяться и далее по обычным сигнальным цепям. При единичном состоянии элемента ЛЭ2 на его выход пройдет отрицательный импульс помехи с линии источника питания.

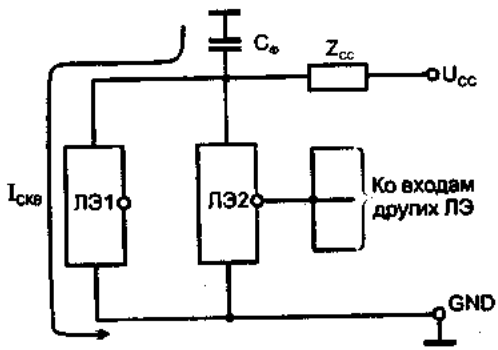
Для борьбы с этими опасными помехами нужны "хорошая земля" и фильтрация напряжений питания.

"Качество земли" улучшается конструктивными мерами, снижающими сопротивление Z_{GND}: шины "земли" делаются утолщенными, нередко для их реализации отводят целые плоскости многослойных конструкций (плат и кристаллов), систему "заземления" соединяют с несколькими выводами корпуса, чтобы сократить пути прохождения токов в этой системе и др.





а



б

Рис. 1.13. Схемы, поясняющие процесс возникновения импульсных помех при переключении цифрового элемента (*а*), и пути протекания сквозного тока при наличии в схеме фильтрующего конденсатора (*б*)

Для шин питания схемы наряду с конструктивными методами применяют и схемотехнические методы:

- в цепи выходных каскадов добавляют небольшие сопротивления, ограничивающие сквозные токи и токи перезаряда емкостей;

- используют элементы с управляемой крутизной фронтов для уменьшения производных сигнальных напряжений и токов;
- применяют развязывающие каскады на выходах ИС для ограничения емкостных нагрузок на этих выходах;
- используют фильтрацию питающих напряжений.

Для фильтрации напряжений питания между линиями U_{CC} и "землей" включают конденсаторы. Высокая эффективность этого метода борьбы с паразитными связями элементов через цепи питания связана со следующим обстоятельством. Цифровые узлы и устройства питают от высококачественных блоков питания со стабилизированным выходным напряжением. Такие источники имеют очень малые выходные сопротивления за счет применения глубоких отрицательных обратных связей в схемах блоков питания. Однако цепь обратной связи инерционна и не успевает отрабатывать короткие импульсные помехи. Поэтому для коротких помех выходное сопротивление источника не обеспечивает того низкого уровня, которое оно имеет в статике. Установка фильтрующих конденсаторов $C_ф$ создает путь (рис. 1.13, *б*), по которому замыкаются импульсы сквозного тока и токи перезаряда емкостей, минуя сопротивление Z_{CC}. Естественно, конденсаторы должны иметь малое сопротивление для высокочастотных сигналов, поэтому для фильтрации выбирают те типы конденсаторов, которые имеют малые паразитные индуктивности.

Рекомендации по числу, типу и емкости фильтрующих конденсаторов вырабатываются практикой и приводятся в руководствах по применению конкретных типов ИС.

§ 1.5. Передача сигналов в цифровых узлах и устройствах. Помехи в сигнальных линиях. Сигнальные линии повышенного качества

При работе ЦУ в межсоединениях (линиях связи) может возникнуть множество импульсных помех различного рода, способных нарушить нормальную работу схемы. К их числу относятся перекрестные помехи, электромагнитные наводки и паразитные колебания из-за несогласованности волновых сопротивлений линий связи.

Перекрестные помехи

Перекрестные помехи (Cross talks) порождаются взаимовлиянием близлежащих линий, передающих сигналы.

Пусть линия — источник помехи является близлежащей для линии, испытывающей воздействие помехи. Тогда между ними существует связь через

паразитную емкость $C_{пом}$ (рис. 1.14, *а*). Схема замещения рассматриваемой цепи может быть представлена в виде рис. 1.14, *б*, где

$$R = R_{вых.1}R_{вх.2}/(R_{вых.1} + R_{вх.2}).$$

Если считать фронт помехи линейным, изменяющимся по закону $U_{пом}(t) = at$, где

$$a = (U_1 - U_0)/t_ф = U/t_ф,$$

то напряжение помехи на входе элемента ЛЭ2 будет определяться соотношением (для времен от нуля до $t_ф$)

$$U_{вх.2}(t) = a\,[1 - \exp(-t/RC)]\,RC,$$

т. е. пропорционально крутизне фронта.

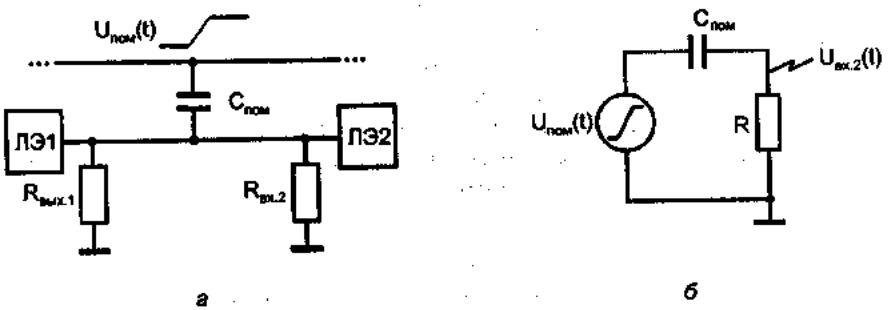


Рис. 1.14. Схема, поясняющая процесс возникновения перекрестных помех в цифровых устройствах (*а*), и схема замещения (*б*)

Борьба с перекрестными помехами осуществляется запрещением параллельного расположения близких и длинных сигнальных линий, размещением между такими линиями экранирующих заземленных проводников (так, в частности, поступают при применении плоских кабелей), применением коаксиальных кабелей, витых пар и др.

Электромагнитные наводки создаются внешними полями. Борьба с ними ведется конструктивными методами — экранированием устройства.

Искажения сигналов в несогласованных линиях

Паразитные колебания из-за несогласованности волновых сопротивлений возникают в связях, которые именуются *длинными*, причем речь не идет об абсолютных значениях длины, важно лишь соотношение длины линии и длины волны передаваемого сигнала.

Так как импульсные сигналы характеризуются широким спектром гармонических частот, говорить о длине волны сигнала для них затруднительно, и рекомендации по отнесению линий связи к коротким или длинным в значительной мере вырабатываются практикой. Например, граничную длину линии часто определяют по условию: время прохождения сигнала по линии должно быть на порядок меньше длительности передаваемого фронта. *Поведение длинной линии резко отличается от поведения короткой.*

Скорость распространения сигнала в линии равна $V = V_c/\sqrt{\varepsilon}$, где V_c — скорость света в вакууме (30 см/нс); ε — диэлектрическая постоянная среды, в которой распространяется сигнал. Практически V = 15—20 см/нс. Схема замещения длинной линии без потерь состоит из цепочки LC-звеньев, где L и C — погонные параметры индуктивности и емкости (т. е. приходящиеся на единицу длины). Такая линия (рис. 1.15, *а*) имеет волновое сопротивление $Z_0 = \sqrt{L/C}$.

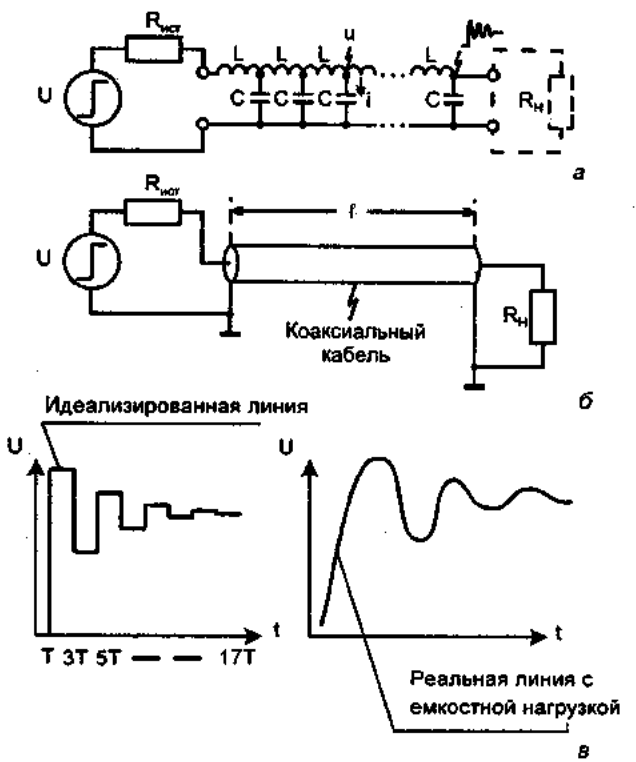


Рис. 1.15. Схема замещения длинной линии без потерь (*а*), схема с реализацией линии в виде коаксиального кабеля (*б*) и временные диаграммы сигналов в идеализированной и реальной линиях (*в*)

Величина Z_0 зависит от конструкции линии и обычно лежит в диапазоне от 50 до 100 Ом. На рис. 1.15, *б* в качестве примера показана реализация линии в виде коаксиального кабеля.

Физически волновое сопротивление соответствует отношению напряжения к току в точке линии, которой достигает распространяющаяся волна. Пока волна распространяется в линии, отношение $u/i = Z_0$ остается неизменным. В конце линии ситуация зависит от подключенного к линии сопротивления. Если в конце линии подключено сопротивление $R_H = Z_0$, то отношение u/i сохраняется, падающая волна не встречает неоднородности и целиком поглощается нагрузкой.

Если в конце линии $R_H \neq Z_0$, то отношение u/i, согласно закону Ома, сохраниться не может, и должно произойти искажение волны. Оно трактуется как появление отраженной волны, параметры которой таковы, что сумма падающей и отраженной волн соответствует условиям в конце линии. Отношение амплитуд отраженной и падающей волн равно *коэффициенту отражения*

$$\rho = (R_H - Z_0)/(R_H + Z_0).$$

Величина коэффициента отражения меняется от минус единицы (при $R_H = 0$, т. е. у короткозамкнутого конца линии) до плюс единицы (при $R_H = \infty$, т. е. у разомкнутого конца линии).

Отраженная волна распространяется обратно к началу линии. Если в начале линии подключено сопротивление, равное Z_0, то отраженная волна поглощается целиком, и режим линии устанавливается окончательно. В противном случае в начале линии также происходит отражение волны, которая вновь пойдет по линии от ее начала к концу. Амплитуды волн будут уменьшаться тем быстрее, чем меньше по абсолютной величине коэффициенты отражения в концах линии. Возможное многократное отражение способно затянуть переходные процессы в линии на время, равное десяткам T_0, где T_0 — время распространения сигнала по линии ($T_0 = \ell / V$, где ℓ — длина линии).

Графики напряжений в конце и начале идеализированной ненагруженной линии имели бы ступенчатый характер, поскольку напряжения в указанных точках менялись бы скачком в моменты очередного прихода волны к концу и началу линии. В реальных условиях скачков напряжения не происходит, в первую очередь из-за характерных для цифровых устройств емкостных нагрузок линий связи. Действующая в конце линии паразитная емкость C_L создает постоянную времени Z_0C_L, ограничивающую скорость нарастания напряжений на емкости. Сглаживанию скачков напряжений содействуют и другие факторы (наличие паразитных колебательных контуров и др.). Таким образом, графики напряжений в конце идеализированной ненагруженной линии и реальной линии будут иметь вид, показанный на рис. 1.15, *в*.

Для устранения недопустимых при передаче цифровых сигналов паразитных колебаний используют параллельное или последовательное согласование волновых сопротивлений.

Параллельное согласование волновых сопротивлений

При параллельном согласовании в конце линии включают резистор (называемый *терминатором*), чтобы сделать сопротивление нагрузки равным волновому. Это дает полное устранение паразитных колебаний, и время передачи сигнала становится равным T_0. Недостаток способа — потребление значительных токов от источника сигнала. После завершения переходных процессов на выходе линии устанавливается напряжение U_1 или U_0. Под этим напряжением находится резистор-терминатор, сопротивление которого равно Z_0 (входным сопротивлением приемника сигнала пренебрегаем, считая его большим и несоизмеримым с Z_0, в противном случае сопротивление терминатора подбирается так, чтобы параллельное соединение входного сопротивления приемника сигнала и резистора-терминатора давало величину Z_0). Сопротивление на выходе линии мало (типичные значения волновых сопротивлений линий передачи сигналов 50—100 Ом). Таким образом и в ходе переходных процессов, и после их завершения источник сигнала нагружен на малое сопротивление. Ток на выходе источника сигнала может оказаться неприемлемо большим.

Для поиска приемлемого варианта включения резисторов на выходе линии можно просмотреть несколько схемных вариантов (рис. 1.16). Для первого варианта (рис. 16, *а*) статические токи в зависимости от логического состояния источника сигнала равны U_0/Z_0 или U_1/Z_0, обычно достигают значений в десятки миллиампер, что превышает допустимые нагрузки логических элементов.

Для второго варианта (рис. 1.16, *б*) токи равны $(U_{CC} - U_0)/Z_0$ и $(U_{CC} - U_1)/Z_0$ и также могут быть неприемлемо велики (заметим, что для типичных элементов ТТЛ(Ш), находящихся в единичном состоянии, на пути тока, втекающего извне, включен диод в обратном направлении. Поэтому перегрузка током указанного направления не возникает, но отсутствие тока через резистор R_H приводит к повышению уровня логической единицы до U_{CC}).

В третьем варианте (рис. 1.16, *в*) на выходе линии включен делитель напряжения, составленный резисторами R_1 и R_2. Этот вариант обладает бóльшей гибкостью, т. к. позволяет варьировать нагрузку на линию, изменяя отношение сопротивлений R_1/R_2. В данном случае линия нагружена на сопротивление параллельного соединения резисторов R_1 и R_2, поэтому обязательным является условие $R_1R_2/(R_1 + R_2) = Z_0$ (здесь, как и ранее, не учитываем

входное сопротивление приемника сигнала, считая его достаточно большим). Переходя от сопротивлений к проводимостям, можно записать это же условие в более простом виде как

$$Y_1 + Y_2 = Y_0, \qquad (1.1)$$

где $Y_i = 1/R_i$ (i = 0, 1, 2).

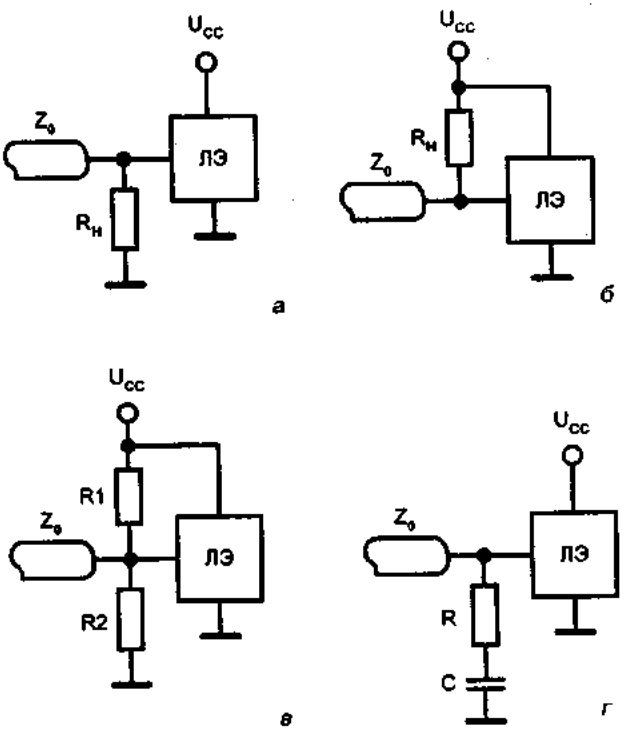


Рис. 1.16. Варианты параллельного согласования волновых сопротивлений линии передачи и нагрузки (*а*, *б*, *в*, *г*)

Условие (1.1) может быть выполнено при разных значениях сопротивлений резисторов. Если удастся подобрать такие значения R_1 и R_2, при которых токи источника сигнала в обоих логических состояниях не будут превышать допустимые значения, то задача согласования волновых сопротивлений линии и нагрузки будет успешно решена. Методику поиска приемлемых значений R_1 и R_2 (если они существуют) проиллюстрируем на примере схемы с КМОП-элементами, которая может быть представлена в виде, показанном на рис. 1.17, *а*.

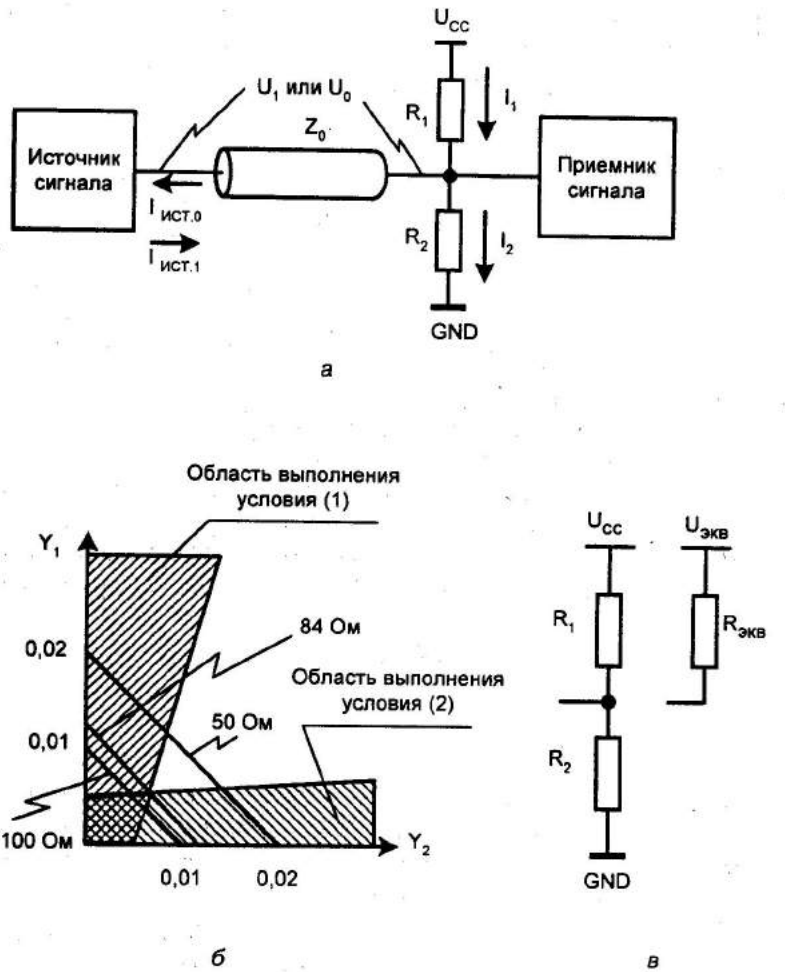


Рис. 1.17. К пояснению методики согласования волновых сопротивлений линии и нагрузки с помощью резистивного делителя напряжения (*а*, *б*, *в*)

Для этой схемы имеем

$$I_{ист.0} = I_1 - I_2 = (U_{CC} - U_0)Y_1 - U_0Y_2;$$

$$I_{ист.1} = I_2 - I_1 = U_1Y_2 - (U_{CC} - U_1)Y_1.$$

В свою очередь токи $I_{ист.1}$ и $I_{ист.2}$ не должны превышать допустимые для данного источника значения, т. е. должны удовлетворять условиям

$$I_{ист.0} \leq I_{ист.0\ доп.};$$

$$I_{ист.1} \leq I_{ист.1\ доп.}.$$

Из полученных равенств и неравенств легко получить условия работы источника сигнала без токовых перегрузок

$$Y_1 \le (I_{ист.0\ доп.} - U_0 Y_2)/(U_{CC} - U_0); \quad (1.2)$$

$$Y_1 \ge (U_1 Y_2 - I_{ист.1\ доп})/(U_{CC} - U_1). \quad (1.3)$$

Условия (1.1), (1.2) и (1.3) должны выполняться одновременно. Все они выражаются линейными зависимостями одной проводимости от другой, что удобно для графического представления решаемой задачи. В координатах Y_1, Y_2 условие (1.1) будет представлено диагональной линией, а неравенства (1.2) и (1.3) — областями (зонами). Приемлемая рабочая точка должна обязательно лежать на диагональной линии и находиться одновременно в обеих допустимых областях.

На рис. 1.17, *б* приведено графическое решение задачи согласования волновых сопротивлений с помощью делителя напряжения для элемента КР1554ЛА3 (элемента 2И-НЕ). Напряжение питания элемента принято равным 4,5 В, уровни U_1 и U_0 — отличающимися от U_{CC} и "земли" на 10%, допустимые токи в обоих логических состояниях 24 мА (допустимая норма для состояний с длительностью более 20 мс). При этих параметрах условия (1.1), (1.2) и (1.3) приобретают вид

$$Y_1 = Y_0 - Y_2;$$

$$Y_1 \le 0{,}0059 - 0{,}11 Y_2;$$

$$Y_1 \ge 9 Y_2 - 0{,}053.$$

Из рисунка видно, что при волновых сопротивлениях линии менее 84 Ом согласование невозможно. Для линии с волновым сопротивлением 100 Ом утолщенным участком диагональной линии показан отрезок, на котором можно выбирать пары значений Y_1 и Y_2.

Известно, что делитель напряжения по теореме Тевенина можно заменить эквивалентной цепью, как показано на рис. 1.17, *в*. При этом

$$R_{экв} = R_1 R_2/(R_1 + R_2);$$

$$U_{экв} = U_{CC} R_2/(R_1 + R_2).$$

На основании теоремы Тевенина полученные для параметров делителя данные можно использовать для схемы с одним резистором, имеющим сопротивление Z_0, если применить для схем согласования специальный источник питания с напряжением, равным $U_{экв}$. В последнее время такое решение в схемах БИС/СБИС встречается все чаще.

Наряду с чисто резисторными цепями пользуются также цепочками с включением последовательно с резистором емкости С, которая предотвращает потребление тока в статике (см. рис. 1.16, *г*). Неудобством при этом является появление такого элемента, как конденсатор. Емкость конденсатора должна

быть достаточно большой, чтобы постоянная времени Z_0C была многократно больше длительности такта передачи данных.

Последовательное согласование волновых сопротивлений

При последовательном согласовании в начале линии последовательно включается резистор $R_{доп}$, сопротивление которого совместно с выходным сопротивлением источника сигнала $R_{ист}$ дает величину Z_0 (рис. 1.18). На выходе линии включено высокое входное сопротивление элемента-приемника, следовательно, там коэффициент отражения приблизительно равен единице, и амплитуда отраженной волны приблизительно равна амплитуде падающей.

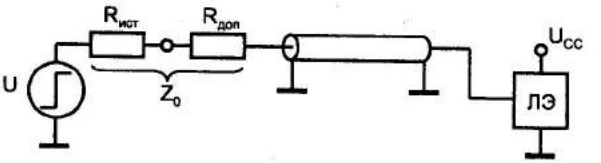


Рис. 1.18. Схема последовательного согласования волновых сопротивлений

Переходный процесс в этом случае протекает следующим образом.

Ступенчатое изменение напряжения источника сигнала U создает на входе линии перепад напряжения U/2 (т. к. $R_{ист} + R_{доп} = Z_0$). Перепад половинной амплитуды распространяется по линии и по истечении времени T_0 достигает ее конца. Коэффициент отражения в конце линии равен единице ($R_{вх} >> Z_0$ и влиянием $R_{вх}$ пренебрегаем). Амплитуда отраженной волны равна также U/2, в итоге в конце линии сразу устанавливается напряжение U. Отраженная волна возвращается к началу линии, где поглощается. Таким образом, на выходе линии процесс заканчивается через промежуток времени T_0, а на входе через $2T_0$.

При последовательном согласовании отсутствуют статические токи нагрузки на источник сигнала, характерные для параллельного согласования. Однако в переходном процессе вход линии проявляет себя как сопротивление, равное Z_0. Следовательно, в переходном процессе источник сигнала нагружен током $U/2 \cdot Z_0$, где U — перепад напряжения источника. Поэтому на высоких частотах передач и при последовательном согласовании, как и в схеме параллельного согласования, возникают проблемы токовой перегрузки выходных цепей источника сигнала.

Повышенное значение сопротивления в цепи передачи сигнала может уменьшать амплитуду передаваемых напряжений, если входное сопротивле-

ние приемника недостаточно велико, так что для схем на элементах с ощутимым входным током (ТТЛ(Ш)) требуется проконтролировать эту возможность. Если от линии связи берутся отводы в середине или начале линии, то задержка передачи сигнала может достигать величины $2T_0$.

Реальное положение в технике борьбы с отражениями в длинных линиях несколько сложнее, чем было описано, т. к. выходные сопротивления цифровых элементов непостоянны и зависят от логического состояния элемента, уровня сигнала и т. д. В частности, выходные сопротивления источников сигнала в состояниях логического нуля и логической единицы обычно неодинаковы. Поэтому последовательное согласование не может быть вполне правильным и приходится довольствоваться приближенным решением. То же самое можно сказать и о входных сопротивлениях элементов.

Схемы с одновременным согласованием волновых сопротивлений в конце и начале линии

Если требуется высокое качество передачи сигнала, т. е. высокая степень подавления его колебаний из-за несогласованности волновых сопротивлений (это характерно для линий передач высшего быстродействия), то применяют одновременно оба вида согласований — параллельное и последовательное. Это *резко улучшает форму передаваемых сигналов, но уменьшает их амплитуду в два раза.* Действительно, в этом случае происходят следующие процессы: в начале линии входной перепад делится пополам между согласующим сопротивлением и входным сопротивлением линии, которые равны. Половинный перепад распространяется по линии к приемнику сигнала, где падающая волна поглощается без отражения, поскольку нагрузка согласована с волновым сопротивлением линии. На этом все и заканчивается. Такой характер происходящих процессов исключает возможность включения линии между двумя элементами цифрового типа, т. к. сигнал половинной амплитуды непригоден для восприятия логическим элементом. Поэтому линия с одновременным согласованием сопротивлений в начале и конце должна работать на элемент аналогового типа (усилитель), который может восстанавливать нормальную амплитуду сигнала.

Заметим, что в схемотехнике используются и варианты, в которых применены одновременно более чем два согласования волновых сопротивлений. Такие схемы будут показаны далее.

Линии передачи сигналов

Для обеспечения работоспособности ЦУ следует уделять большое внимание линиям связи (межсоединениям элементов). Это важно при проектировании

печатных плат, и становится особенно острой проблемой в БИС/СБИС, где преобладающая часть площади кристалла, задержек сигналов и потребляемой мощности зачастую относится именно к системе межсоединений.

Ряд рекомендаций для разработки ЦУ высказан ранее ("качество земли", ограничения на параллельные размещения сигнальных линий, фильтрация питания, согласование волновых сопротивлений в длинных линиях). Отметим теперь особенности основных вариантов технической реализации межсоединений.

На платах межсоединения выполняются одиночными проводниками над "земляной" плоскостью, двумя проводниками, витыми парами, микрополосковыми линиями, коаксиальными кабелями малого диаметра и др.

Схема соединения одиночным проводником (рис. 1.19, *а*) изображена с учетом напряжения помехи, которая может возникать между "землями" двух элементов. В этом случае помеха передается на вход приемника сигнала.

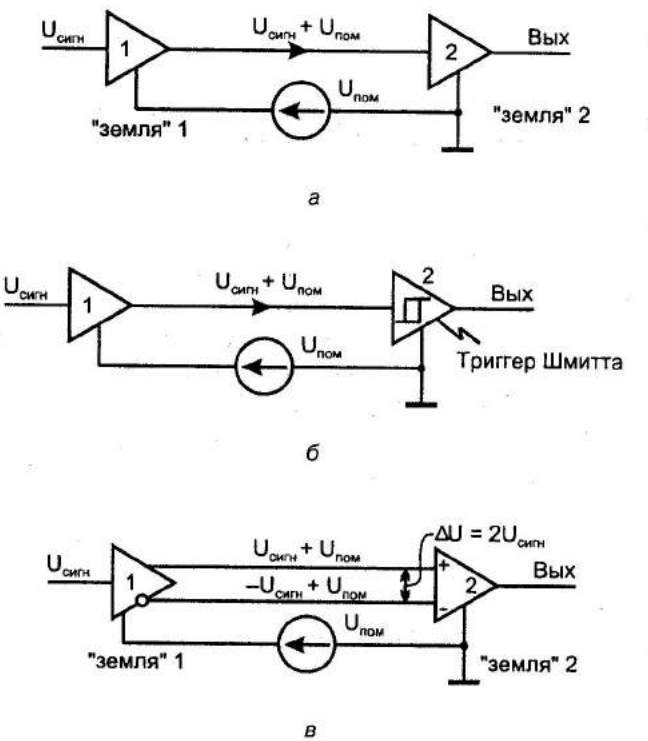
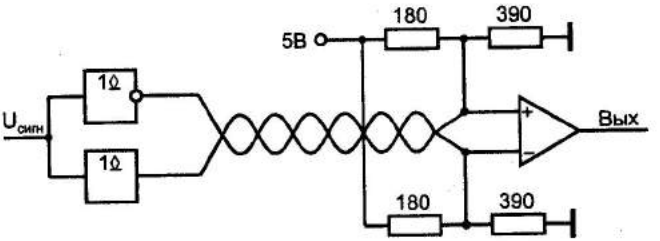
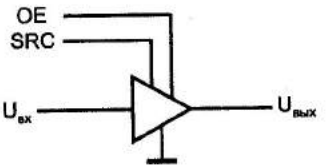


Рис. 1.19. Простейшая схема передачи цифрового сигнала (*а*), схема с гистерезисным приемником (*б*), передача сигнала дифференциальным способом (*в*)



г



д

Рис. 1.19. Пример схемы помехоустойчивой передачи сигнала (*г*), буфер с регулируемой крутизной фронта (*д*)

Помехоустойчивость передачи повышается, если элемент-приемник обладает гистерезисными свойствами, как, например, триггер Шмитта (рис. 1.19, *б*). Благодаря гистерезисной характеристике приемника для переключения в состояние логической 1 нужно подать на вход напряжение, значительно превышающее пороговое, а для переключения в 0 — значительно меньше, чем пороговое. Ясно, что это повышает уровень допустимых помех, причем тем больше, чем шире петля гистерезиса.

Значительное улучшение может дать передача парафазного сигнала по двум линиям (дифференциальная передача), показанная на рис. 1.19, *в*. Приемником сигнала в этом случае служит дифференциальный усилитель (или компаратор). На его верхнем входе действует напряжение $U_{сигн} + U_{пом}$, а на нижнем напряжение $-U_{сигн} + U_{пом}$. Дифференциальный приемник воспринимает разность напряжений между входами, которая равна $2U_{сигн}$ и не содержит напряжения помех. Перекрестные помехи в данном случае также значительно ослабляются, поскольку появляются в обоих проводниках близкими по величине, так что их разность, ощущаемая приемником, мала.

На рис. 1.19, *г* приведена схема помехоустойчивой передачи сигнала дифференциальным способом по витой паре. По волновому сопротивлению витая пара согласуется резистором-терминатором, выполненным в виде делителя из резисторов 180 и 390 Ом, эквивалентное сопротивление которого относительно выхода равно 120 Ом.

Витая пара, часто применяемая в ЦУ, представляет собою как бы упрощенную конструкцию коаксиального кабеля, в которой один из проводов можно рассматривать как некоторый аналог оплетки кабеля. Для примера укажем параметры витой пары проводников типа МНВ $2 \times 0{,}05$ мм^2: волновое сопротивление 100 Ом; сопротивление проводника постоянному току 0,35 Ом/м; коэффициент перекрестной помехи 0,15; время задержки сигнала 6 нс/м.

На рис. 1.19, *д* изображен буфер с третьим состоянием и регулировкой крутизны нарастания выходного сигнала. Введением/снятием третьего состояния управляет вход OE (Output Enable), крутизной фронтов — сигнал SRC (Slew Rate Control). Пологий фронт желателен, поскольку замедление изменений токов и напряжений снижает помехи из-за токовых импульсов в цепях питания, перекрестные помехи и др. В то же время в критичных для быстродействия устройства путях замедленные переключения элементов нежелательны, и поэтому в них устанавливают режимы крутых фронтов. Буферные каскады с регулировкой крутизны фронтов достаточно часто применяют в современных СБИС. В них встречаются и более изощренные способы регулировок скоростей изменения сигналов в буферных элементах по специально подобранным нелинейным законам.

Большие проблемы связаны с реализацией межсоединений в СБИС. Уменьшение размеров схемных элементов, одинаковое для размеров в плане и для толщин, ведет к уменьшению поперечного сечения проводников по квадратичной зависимости, что увеличивает их погонное сопротивление. Резистивность и емкости связей ограничивают гипотезу их эквипотенциальности. Распространение потенциала вдоль проводника подчиняется уравнению диффузии, чему соответствуют падение скорости распространения сигнала по мере удаления от источника и квадратичная зависимость задержки от длины проводника. Удвоение длины проводника приводит к учетверению задержки и т. д. Поэтому при использовании длинных связей иногда включают через определенные расстояния усилители-повторители сигнала. Для БИС/СБИС, реализованных по технологии с минимальным размером 0,25 мкм, задержки в связях становятся в среднем равными задержкам в логических элементах, а при минимальных размерах 0,18 мкм задержки в связях составляют около 70% общих задержек.

Линии связи с гальваническими развязками

Электрические токи могут протекать только в замкнутых цепях, для образования которых устройства, обменивающиеся данными, должны иметь общую схемную "землю" (общую точку). Объединение схемных земель взаимодействующих устройств — обычное простейшее решение, применяемое в подавляющем большинстве систем, но ему свойственны и известные недостатки. Не исключено, что до момента соединения друг с другом схемные земли двух устройств имели разные потенциалы. Тогда при соединении

схемных земель в образующемся замкнутом контуре могут возникнуть импульсные сигналы, небезопасные для соединяемых устройств. Далее, нередко электронные приборы управляют работой мощных высоковольтных агрегатов и при аварийных пробоях изоляции высокие напряжения могут полностью выводить их из строя. Кроме того, в электрически связанных цепях возникает множество помех — перекрестные помехи, помехи по цепям питания и др.

В силу отмеченных обстоятельств, т. е. по соображениям электробезопасности и борьбы с помехами, в некоторых ситуациях предусматривается применение линий связи с гальваническими развязками. В этих случаях в тракте передачи данных имеется участок, на котором данные передаются с помощью оптических средств без электрической связи между источником и приемником. Основными элементами обеспечения электрических развязок служат *оптроны* (оптические изоляторы) — приборы с размещением светодиода и фоточувствительного элемента (диода или транзистора) в одном корпусе. Оптический изолятор преобразует входной сигнал в выходной при полном гальваническом разделении цепей входа и выхода. Связь выходных цепей со входными осуществляется следующим образом — под воздействием входного сигнала генерируется электромагнитное излучение (обычно в инфракрасной области спектра), которое затем воспринимается фоточувствительным элементом, преобразующим его в электрический сигнал. С помощью такого процесса можно передавать по линии не только цифровые, но и аналоговые сигналы.

На рис. 1.20 показана схема цифровой связи источника сигнала (например, процессора) с приемником (например, внешним устройством микропроцессорной системы).

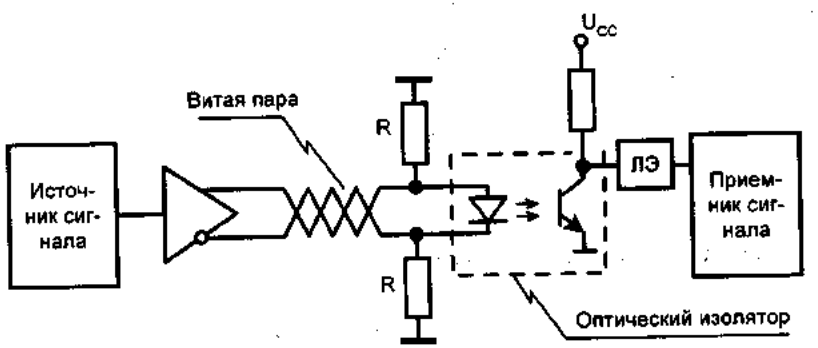


Рис.1.20. Схема линии связи с гальванической развязкой

Сигнал на оптический изолятор передается от источника через буфер по витой паре с согласованием волновых сопротивлений резисторами-термина-

торами R. Диодно-транзисторный оптический изолятор обеспечивает гальваническую развязку. Импульсы тока преобразуются светодиодом в импульсное излучение, воспринимаемое фоточувствительным транзистором, ток которого изменяется соответственно облучению базы. С коллектора транзистора снимается электрический сигнал и подается на приемник через логический элемент ЛЭ.

Линии с оптронными гальваническими развязками имеют ограниченное быстродействие, обусловленное инерционностью оптронов, однако с развитием технологий производства скоростные возможности оптронов возрастают. В номенклатуре современных оптронов имеются приборы со скоростью передачи до 20 Мбод.

Линии передачи типа "токовая петля"

Наряду с отображением логических сигналов уровнями напряжения при передаче последовательных данных применяют и отображение сигналов наличием или отсутствием тока с цепи. Работа с токовыми сигналами, не требующими формирования больших перепадов напряжения, имеет свои особенности, которые в соответствующих условиях можно с пользой применить для улучшения характеристик передающего тракта. В частности, вариант "токовая петля" используется при передачах на большие расстояния (до десятков километров), хотя и с относительно невысокой скоростью.

Линия связи с "токовой петлей" показана на рис. 1.21 на примере связи источника символьных данных с устройством (телетайпом), которое не только принимает или передает последовательные данные, но и имеет электромеханическое печатающее устройство, управляемое непосредственно токовыми сигналами. В телетайпе имеются соленоид печатающего устройства, зашунтированный диодом, и ключ передачи К. "Партнером" телетайпа является устройство (последовательный адаптер), выдающее последовательные данные на линию Tx и принимающее такие данные по линии Rx.

Нулевое значение сигнала на выходе Tx преобразуется инвертором в единичное, запирающее транзистор Т1 типа p-n-p. Единичный сигнал на выходе Tx, напротив, задает на базовую цепь транзистора Т1 низкий уровень напряжения, открывающий транзистор. Транзистор насыщается и через обмотку соленоида проходит ток. Ключ К в отсутствие передачи со стороны телетайпа замкнут, транзистор Т2 при этом насыщен подачей отрицательного напряжения на его базовую цепь, и на вход Rx через насыщенный транзистор поступает высокий уровень логической единицы, что является признаком отсутствия передачи. При размыкании ключа К транзистор Т2 запирается и через резистор с сопротивлением 1 кОм вход Rx подключается к нулевому потенциалу. Уровни токов в петле чаще всего 0 и 20 мА, имеются варианты с токами 0 и 40 мА.

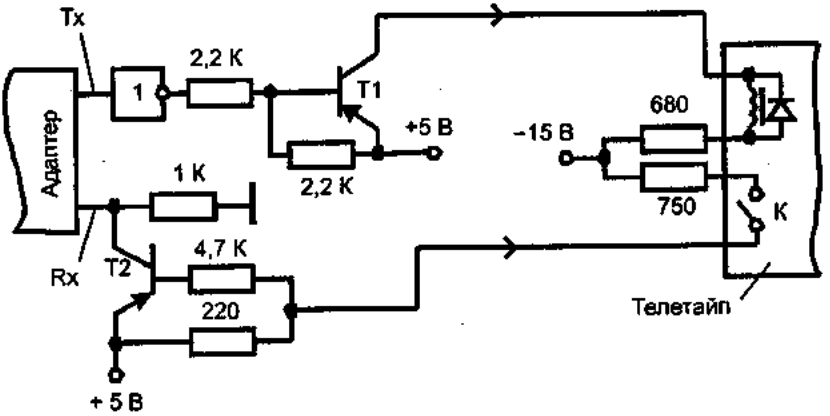


Рис. 1.21. Схема линии связи типа "токовая петля"

О стандартах сигналов ввода/вывода данных

С развитием элементной базы постоянно изменяются параметры сигналов в трактах передачи данных. Для получения скоростных линий передачи с высокими тактовыми частотами и малой потребляемой мощностью разработаны многие стандарты ввода/вывода, которые используются для интерфейса между различными блоками систем (процессорами, памятью, периферийными схемами, печатными платами). Очень широко используются современные стандарты ввода/вывода в БИС/СБИС программируемой логики, в которых для упрощения связей с разнообразными блоками систем применяются программируемые высокопроизводительные "многостандартные" буферы. Способность микросхем к поддержке многих промышленных стандартов ввода/вывода сокращает время разработки продукции и время ее выхода на рынок.

Скоростная передача сигналов, предусматриваемая множеством современных стандартов, требует тщательного согласования волновых сопротивлений в линиях связи. Для эффективного подавления отражений волн в линии применяются схемы, в которых используются одновременно комбинации параллельных и последовательных согласований. На рис. 1.22 показаны варианты с двойным параллельным согласованием (*а*), последовательным согласованием в начале линии и параллельным в ее конце (*б*) и последовательно-параллельным согласованием в начале линии и параллельным в ее конце (*в*).

Схема с двумя видами согласований, источником сигнала типа КМОП-инвертора и дифференциальным аналоговым приемником показана на рис. 1.22, *а*. Подобные схемы требуют питания от трех различных напряже-

ний: V_{CCIO} — напряжения питания выходного буфера (на схеме не показано), V_{TT} — напряжения, к которому подключаются резисторы-терминаторы и V_{REF} — опорного напряжения, подключаемого ко второму входу дифференциального приемника. На рис. 1.22, *б* изображена схема с двумя видами согласования — последовательным и параллельным. Свойства такой схемы уже рассмотрены ранее. Схема на рис. 1.22, *в* имеет три вида согласований — два в начале линии и одно в конце.

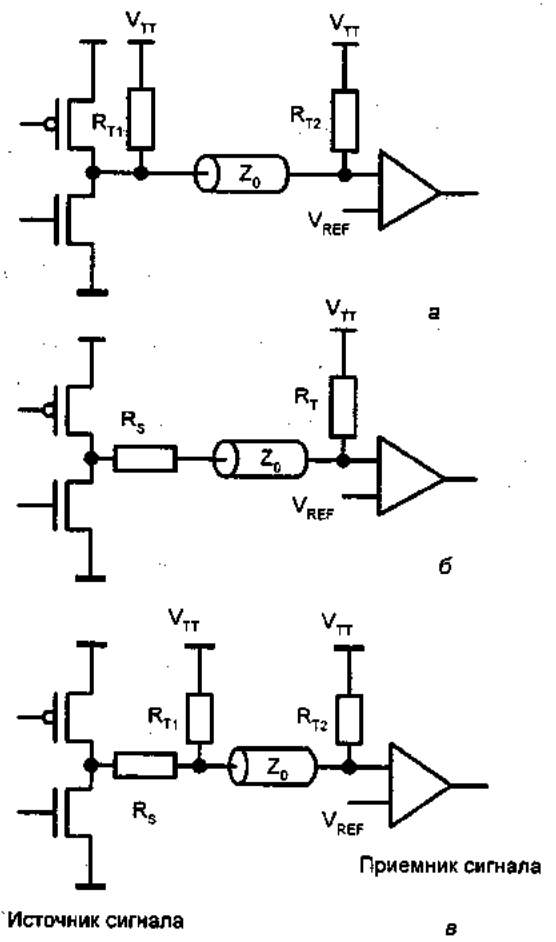


Рис. 1.22. Схемы с подавлением волновых отражений комбинированными методами

Существующие стандарты сигналов ввода/вывода отличаются друг от друга как величинами используемых напряжений и уровнями нагрузочных токов, так и составом схемных элементов. К числу стандартов ввода/вывода принадлежат следующие:

- LVTL — однополюсная линия широкого применения для устройств с напряжением питания 3,3 В, имеет выходной буфер двухтактного типа с выходным напряжением высокого уровня не менее 2,4 В. Не использует опорное напряжение и терминаторы. Допускает максимальное входное напряжение до 3,9 В. Имеет варианты с выходными токами в диапазоне 2—24 мА.
- LVCMOS — однополюсная линия для устройств с напряжением питания 3,3 В. Работа входного буфера соответствует стандарту LVTL, а от выходного буфера требуется формирование напряжения высокого уровня не менее V_{CCIO} − 0,2 В. При этом используется напряжение V_{CCIO} = 3,3 В, а опорные напряжения и терминаторы не применяются.
- 2,5 В — этот стандарт подобен стандарту LVCMOS, но использует напряжение питания 2,5 В.
- 1,8 В — этот стандарт подобен стандарту LVCMOS, но использует напряжение питания 1,8 В.
- 3,3V PCI (Peripheral Component Interconnect) — стандарт использует напряжение питания 3,3 В, на этом стандарте реализуются шины разрядностью до 64 при частоте тактирования 33 или 66 МГц. Основан на схемах стандарта LVTL.
- PCI-X — усовершенствованная версия стандарта PCI с поддержкой более высоких частот передачи сигналов и более жесткими требованиями к ним.
- LVDS — стандарт дифференциальной передачи, используется для реализации передач с очень высоким быстродействием и малым потреблением мощности. Имеются две версии промышленного стандарта. Стандарт от IEEE (Institute of Electrical and Electronic Engineers, Институт инженеров по электротехнике и электронике) поддерживает передачи с полосой пропускания 250 Мбит/с, а стандарт от ANSI/TIA/EIA (American National Standards Institute, Национальный институт стандартизации США/Telecommunications Industry Associastion, Ассоциация промышленности средств связи/Electronics Industries Association, Ассоциация электронной промышленности) поддерживает передачи с полосой пропускания в диапазоне от 644 до 840 Мбит/с. Используется напряжение V_{CCIO} = 3,3 В и резистор-терминатор с сопротивлением 100 Ом между двумя линиями дифференциальной передачи сигналов. Опорное напряжение не используется.
- LVPECL — стандарт дифференциальной передачи, сходный со стандартом LVDS.

- GTL+ — высокоскоростная стандартная шина, впервые использованная фирмой Intel для интерфейса с процессором Pentium Pro и часто применяемая для интерфейсов между различными процессорами и при коммуникациях на печатной плате. Требуется использование опорного напряжения 1,0 В и напряжений V_{TT} = 1,5 В. Сигнал на линию подается от каскада с открытым стоком. Минимальное значение питания цепей ввода V_{CCIO} должно быть не менее 3,0 В.
- SSTL-2 Class 1, Class 2 (Stub-Series Terminated Logic) — стандарт для систем с напряжением питания 2,5 В, требует опорного напряжения 1,25 В, напряжения V_{CCIO} = 2,5 В и напряжения V_{TT} = 1,25 В. Используется для интерфейсов с микросхемами высокоскоростной синхронной динамической памяти SDRAM (Synchronous Dynamic Random Access Memory).
- SSTL-3 Class 1, Class 2 (Stub-Series Terminated Logic) — стандарт для систем с питанием 3,3 В, требует опорного напряжения 1,5 В и напряжения V_{TT} = 1,5 В. Используется для интерфейсов с высокоскоростной динамической памятью SDRAM.
- HSTL (High Speed Transceiver Logic) — стандарт использует выходной буфер с напряжением питания V_{CCIO} = 1,5 В, опорное напряжение 0,75 В, напряжение V_{TT} = 0,75 В. Применяется для передач сигналов в цифровых ИС.
- AGP — соответствует спецификации Advanced Graphics Port Interface Specification, введен фирмой Intel для графических применений. Требует опорного напряжения 1,32 В и напряжения питания V_{CCIO} = 3,3 В. Не требует применения резисторов-терминаторов.
- CTT (Center-Tap-Terminated) — низковольтный высокоскоростной интерфейс для передач сигналов в цифровых ИС. Требует опорных напряжений 1,5 В, напряжений питания V_{CCIO} = 3,3 В и напряжений V_{TT} = 1,5 В. Приемники сигналов для этого стандарта совместимы с приемниками по стандартам LVTTL и LVCMOS. Передатчики, если они не используют резисторы-терминаторы, совместимы по переменному и постоянному току со спецификациями стандартов LVTTL и LVCMOS.

На рис. 1.23 приведены примеры схемных реализаций для некоторых стандартов сигналов ввода/вывода.

Остановимся несколько подробнее на схемотехнике скоростного интерфейса LVDS. Известны интерфейсы RS-422 и RS-485 с дифференциальными линиями связей для передачи относительно высоковольтных сигналов на сравнительно большие расстояния. Для передач высокой производительности между микросхемами или близко расположенными блоками при скоростях передач более 50 Мбит/с и при необходимости снижения потребляемой мощности "старые" интерфейсы уступают место интерфейсам LVDS или M-LVDS (Multipoint LVDS). Правда, очень высо-

кие скорости передачи можно получить с помощью схемотехники ECL (Emitter-Coupled Logic, эмиттерно-связанная логика) или PECL (Positive ECL), но в сравнении с LVDS это сопровождается повышением стоимости и потребляемой мощности.

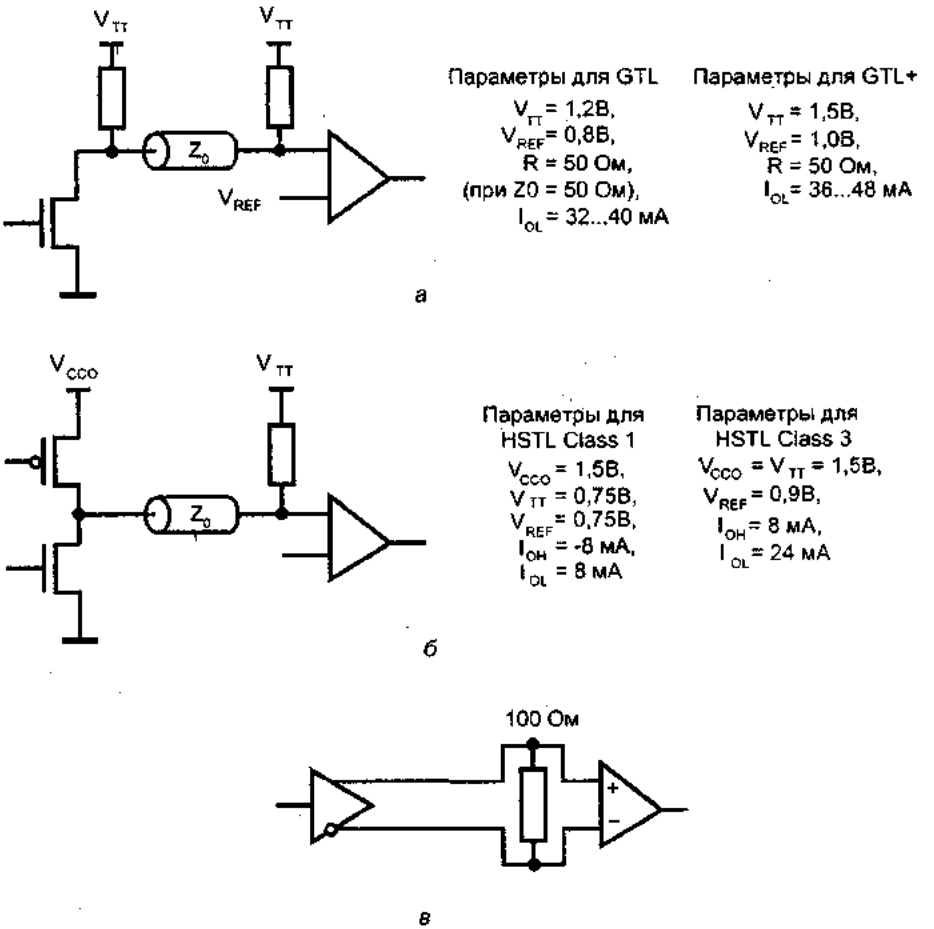


Рис. 1.23. Примеры схем, реализующих различные стандарты сигналов ввода/вывода. Стандарты GTL и GTL+ (*а*), стандарты HSTL класса 1 и 3 (*б*), стандарт LVDF (*в*)

Оценивая преимущества дифференциальных связей, напомним, что длительность $t_ф$ формирования фронтов цифровых (импульсных) сигналов при данном уровне токов перезаряда нагрузочных емкостей I пропорциональна

как величине емкостной нагрузки C, так и величине формируемого перепада напряжений ΔU:

$$t_ф = C\Delta U/I.$$

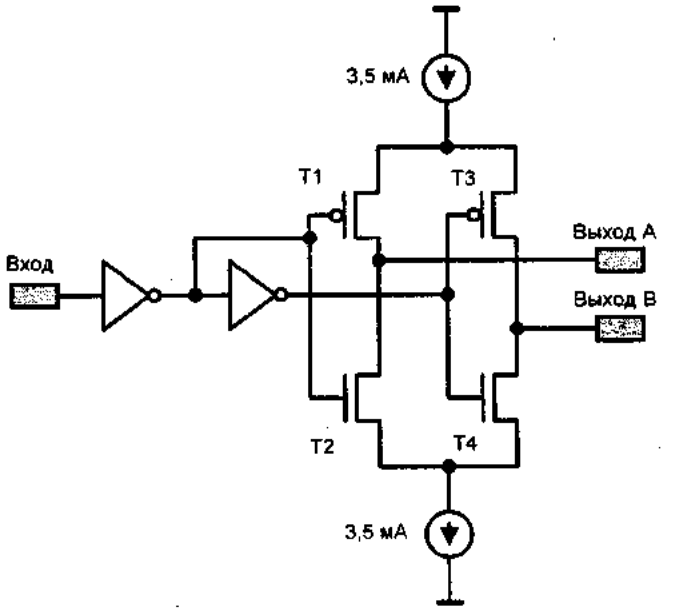
Поскольку помехозащищенность дифференциальных линий связи гораздо выше, чем у одиночных, амплитуду передаваемых сигналов ΔU в дифференциальных линиях можно существенно снизить. Емкостная нагрузка на линиях связи, не имеющих шинной структуры, также значительно меньше, поскольку эти линии соединяют две точки схемы и не имеют многочисленных источников и приемников сигналов, подключенных к ним, а также паразитных связей с параллельно идущими линиями, что имеет место в шинных трассах. Таким образом, в дифференциальных связях минимизируется время формирования фронтов.

Схема оконечного каскада для подачи сигналов в дифференциальный тракт LVDS показана на рис. 1.24, *а*. Каскад имеет токовый выход, напряжения большой амплитуды на выходе не формируются, что снижает как мощность, потребляемую схемой, так и ее зависимость от частоты переключений схемы. Питание каскада осуществляется от источников тока, направления токов в линиях выходов A и B противоположны друг другу и изменяются при передачах логических сигналов 0 и 1 под управлением входного сигнала, что легко проследить по схеме рис. 1.24, *а*.

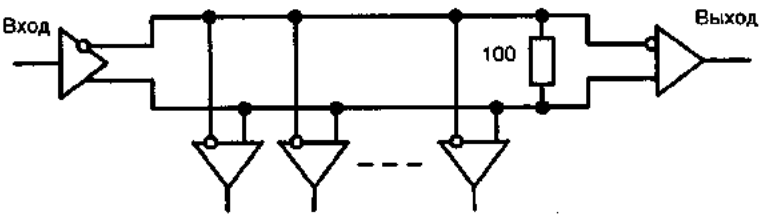
На рис. 1.23, *в* была приведена линия *симплексной* (т. е. односторонней) передачи, от одного источника одному приемнику (вариант Point-to-Point) в схемотехнике LVDS. Для варианта передач от одного источника нескольким приемникам (вариант Multidrop) схема строится согласно рис. 1.24, *б*, причем приемники должны располагаться близко к основным линиям передачи (отводы должны быть короткими).

Полудуплексные линии связей (рис. 1.24, *в*), также очень часто применяемые в цифровых устройствах, передают сигналы в двух направлениях, но поочередно (с разделением по времени). Показанный вариант предусматривает обмен только между двумя источниками и называется вариантом "точка-точка" (Point-to-Point). В такой схеме включены два согласующих резистора, что повышает токовую нагрузку на источники сигнала, от которых требуются выходные каскады повышенной мощности.

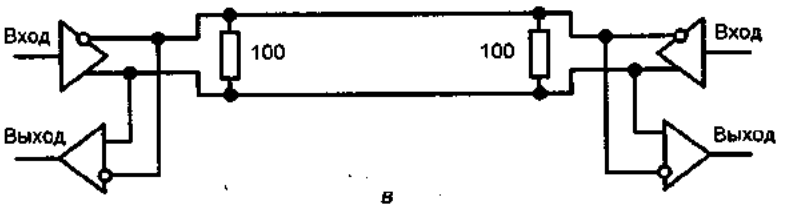
Схема полудуплексной связи многоточечного типа (Multipoint), в которой двусторонний обмен возможен между любыми источниками при разделении по времени, строится путем увеличения на каждом из концов линии числа пар усилителей, показанных на рис. 1.24, *в*. Для удобства создания многоточечных систем обмена разработаны специальные интерфейсные схемы M-LVDS. Существуют и варианты LVDS, называемые LVDM и LVDT. Вариант LVDM служит для работы на линии с двумя согласующими резисторами (терминаторами), требующей более мощных токовых выходов, а вариант LVDT имеет терминаторы, встроенные в кристалл.



а



б



в

Рис. 1.24. Линии связи типа LVDS

Передача данных с двойной скоростью (технология DDR)

Частота передач сигналов на печатной плате и кристалле ограничена параметрами этих конструкций и не может превышать определенных значений. Так, например, считается, что на обычной печатной плате максимальная частота передач составляет приблизительно 200 МГц. В то же время устройства обработки данных уже способны работать на значительно более высоких частотах. Пропускную способность линий передач удалось резко (в два раза) повысить в рамках так называемой технологии DDR (Double Data Rate). В обычном варианте передач (SDR — Single Data Rate) данные принимаются под воздействием активного фронта тактирующего сигнала, причем активным может служить положительный или отрицательный фронт. Интервал времени между приемами данных равен периоду тактирующих импульсов. Технология DDR состоит в приеме данных под воздействием фронтов обоих знаков. Такая возможность обеспечивается специальными схемотехническими мерами, а интервал между моментами приема соседних данных становится равным полупериоду тактирующих импульсов.

Быстродействующие низковольтные дифференциальные линии с двойной скоростью передачи данных, представителем которых является стандарт DDR LVDS, обладают максимальной на современном этапе скоростью передачи данных. Сейчас скорость передачи достигла для них более чем 3 Гбит/с в ближайшее время прогнозируется скорость более чем 6 Гбит/с. Преимущества дифференциальных линий связи рассматривались ранее. Кроме того, было показано, что в дифференциальных связях минимизируется время формирования фронтов. Если, к тому же, применена технология DDR и передача данных осуществляется каждым фронтом, т. е. дважды за период, то результатом будет реализация самых быстродействующих на сегодня линий связи. Конечно, все большее распространение быстродействующих линий передачи последовательных данных не означает отказ от многоразрядных шинных структур. В зависимости от конкретных условий преимущество имеет то или иное решение.

О разрядностях высокоскоростных шин. Блоки SERDES и CDR

Пропускная способность (производительность) шины зависит от частоты и разрядности передаваемых по ней слов и определяется их произведением. В прошлом пропускную способность шин традиционно повышали, увеличивая их разрядность. Но с ростом частот меняются параметры сигналов, увеличивается крутизна фронтов напряжений и токов, усложняется ситуация с воздействием помех на сигнальные линии, что особенно сказывается на многоразрядных шинах. В многоразрядных шинах труднее решать задачи

борьбы с перекрестными помехами, помехами по цепям питания и помехами из-за отраженных волн в несогласованных линиях. Поэтому для многоразрядных шин частота тактирования ниже, чем для малоразрядных. Вследствие указанных обстоятельств в современных условиях изменился и подход к выбору рациональной разрядности для шин с высокой пропускной способностью.

В последнее время в схемотехнике СБИС прослеживается *тенденция к замене многоразрядных шин на малоразрядные*, но работающие на более высокой частоте. Замена многоразрядных шин на малоразрядные может дать существенное упрощение и удешевление систем при сохранении или даже повышении производительности шины. Упрощение и удешевление систем могут быть весьма большими, т. к. сокращение числа контактов микросхем может оказаться многократным. Современные проектные нормы (около 0,1 мкм) позволяют получать логические элементы очень высокого быстродействия, что также является условием успешного перехода к малоразрядным шинам. Действительно, уменьшение разрядности передаваемых слов сопряжено с необходимостью последующего перехода к другим разрядностям, свойственным устройствам обработки получаемых по шине данных. Для этого нужны блоки преобразования последовательных данных в параллельные, и наоборот. Такие блоки называются *блоками SERDES* (Serializer-Deserializer). Благодаря достижениям современной схемотехники блоки SERDES при использовании дифференциальных линий связи уже могут работать на частотах в несколько гигагерц. Дифференциальные линии связей, естественно, требуют использования двух контактов для каждой линии.

Оценить получаемые при переходе на малоразрядные шины результаты можно на следующем примере. Пусть 64-разрядная шина заменяется на восьмиразрядную с дифференциальными связями. Тогда вместо 64 контактов для шины потребуется 16, и на каждом канале передачи, которых может быть несколько, будет сэкономлено 48 контактов, что существенно упрощает не только микросхему, но и монтаж на печатной плате. Частота формирования байтов, т. е. время накопления байтов, на которые была разделена 64-разрядная шина, будет приблизительно в восемь раз ниже, чем частота тактирования последовательных дифференциальных связей. Если, например, последовательные связи работают на частоте 1,6 ГГц, то в результате будет получен эквивалент 64-разрядной шины, работающей на частоте 200 МГц.

В настоящее время не все проблемы перехода на малоразрядные шины высшего быстродействия решены, в частности, высокоскоростные блоки SERDES потребляют еще слишком большую мощность, однако скоростные последовательные связи становятся обычным атрибутом для многих современных цифровых СБИС.

Работа последовательных линий на частотах в сотни мегагерц и до нескольких гигагерц предъявляет чрезвычайно высокие требования к синхрониза-

ции процессов в блоках, участвующих в передаче данных. В традиционных системах синхронизации используются внешние опорные синхросигналы и фазовые сдвиги информационных сигналов относительно опорных ограничиваются спецификациями, вырабатываемыми для того или иного типа канала связи. Наряду с этим вариантом сейчас получила распространение и работа по варианту CDR (Clock-Data Recovery), который исключает необходимость в жестких спецификациях на временные сдвиги сигналов, поскольку позволяет подстраивать друг к другу синхросигналы и передаваемые данные. Пример схемы приемного канала с CDR показан на рис. 1.25.

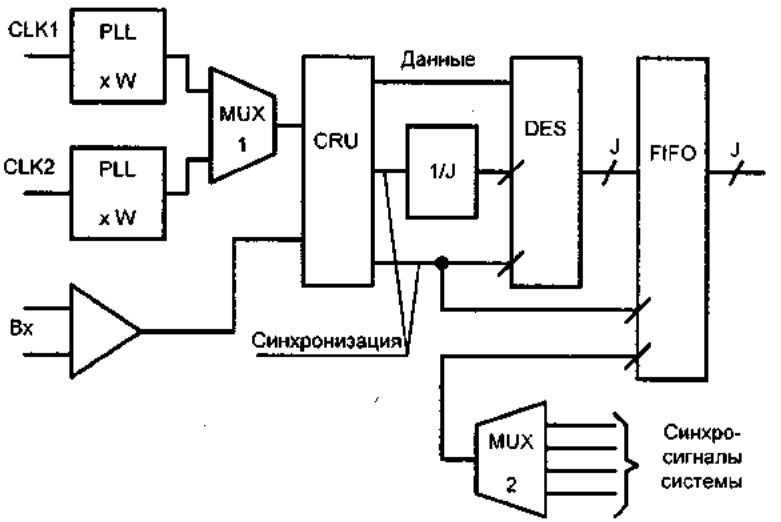


Рис. 1.25. Схема приемного канала с блоками CDR

Входы CLK1 и CLK2 приемного канала являются входами внешних синхросигналов, имеющихся в системе, частота которых ниже, чем частота передач по последовательному каналу. Блоки PLL *(см. § 3.6)* умножают частоту этих синхросигналов на W. Мультиплексор MUX1 позволяет выбрать для использования один из двух синхросигналов. Синхросигналы умноженной частоты, соответствующие частоте работы последовательного канала, как и передаваемые по этому каналу данные, поступают на входы блока CRU (Clock Recovery Unit). Блок CRU генерирует восстановленные синхросигналы, сфазированные с сигналами принимаемых данных. Восстановленные синхросигналы управляют работой блока DES (Deserialiser), принимающего последовательные данные в регистр типа SIPO (Serial Input — Parallel Output). Для выдачи из блока DES параллельных данных разрядности J используется синхросигнал из блока CRU, частота которого делится на J. Параллельные данные записываются в буфер FIFO (First In, First Out; первым прибыл,

первым обслужен), чтение из буфера осуществляется по синхросигналам системы, в которую вводятся принятые данные. Мультиплексор MUX2 позволяет выбирать одну из нескольких частот синхронизации системы.

Канал передачи данных из системы в последовательную линию связи реализуется проще, чем приемный канал — в нем отсутствует блок CRU. Параллельные данные из системы записываются в буфер FIFO при тактировании синхросигналами системы. Далее эти данные поступают в регистр типа PISO (Parallel Input — Serial Output) при тактировании частотами CLK1 × (W/J) или CLK2 × (W/J). Из регистра данные поступают в канал передатчика под управлением сигналов CLK1 × W или CLK2 × W.

§ 1.6. Вспомогательные элементы цифровых узлов и устройств

К числу вспомогательных отнесем элементы, не выполняющие логические операции или запоминание данных, но зачастую необходимые для построения ЦУ: элементы задержки, формирования и генерации импульсных сигналов, а также их визуальной индикации.

Элементы задержки

Задержки цифровых сигналов требуются прежде всего для временного согласования распространения сигналов по различным путям в ЦУ с целью борьбы с критическими временными состязаниями, нарушающими работоспособность автоматов с памятью.

Вариант технической реализации элементов задержки зависит от требуемых значений параметров задержки сигналов, а именно: величины, стабильности, регулируемости и т. д. На практике применяют следующие различные варианты реализации задержек:

- отрезки обычных или специальных коаксиальных кабелей;
- цепочки логических элементов;
- искусственные электромагнитные линии задержки;
- RC-цепочки;
- одновибраторы;
- схемы деления частоты тактовых сигналов.

Остановимся на самых типичных для ЦУ вариантах — цепочках логических элементов и RC-цепочках.

В первом случае используется естественная инерционность логических элементов. При составлении из нескольких логических элементов последова-

тельной цепочки можно суммировать задержки отдельных элементов. Для целей задержки естественно применять простейшие элементы-инверторы или повторители. Это удобный способ — в простейшем корпусе МИС уже размещены 6 инверторов или повторителей. Задержку можно регулировать дискретно, изменяя число элементов в цепочке. Если цепочка составлена из инверторов, то при четном их числе получается просто задержка сигнала, при нечетном — задержка с инверсией. Величины получаемых задержек обычно подходят к требуемым, т. к. требуется компенсация разновременности распространения сигналов в цепях, также составленных из логических элементов. Точность задержки в этом случае ограничивается разбросом собственных задержек элементов и поэтому невысока.

Задержку на большее время можно получить с помощью RC-цепочки, включаемой в цепь передачи сигнала (рис. 1.26), где она формирует экспоненциальные процессы перезаряда емкости через резистор R с постоянной времени RC. Если считать пороговым напряжением середину логического перепада, то время задержки $t_d = RC \cdot \ln 2 = 0{,}7RC$ (индекс d происходит от англ. *delay*, что означает задержку). После RC-цепочки в схеме включены три инвертора для формирования достаточно крутых фронтов на выходе элемента задержки.

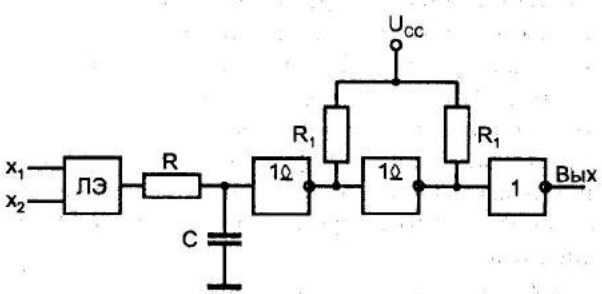


Рис. 1.26. Схема задержки с RC-цепочкой

Имеется существенная разница в условиях применения RC-цепочек в схемах на МОП-транзисторах и в схемах на биполярных приборах. В первом случае входные токи элементов пренебрежимо малы и включение на входе логического элемента даже большого сопротивления вполне допустимо. Во втором случае входные токи элементов значительны, поэтому в их входные цепи можно включать лишь малые сопротивления (иначе произойдут недопустимые изменения уровней напряжения U_0 и U_1 из-за падений напряжения на резисторе R). Нередко допустимые значения сопротивления резистора R составляют в этом случае величину порядка сотен Ом. При малых значениях сопротивления R постоянную времени придется увеличивать за счет больших емкостей C, что не всегда удобно по конструктивным соображениям.

С увеличением постоянной времени RC напряжение на емкости при переключениях становится все более пологим. При этом свойственный логическим элементам разброс пороговых напряжений будет вызывать все больший разброс задержек. Таким образом, чем больше задержка, тем менее точной она становится. Кроме того, для некоторых элементов (типа КМОП) слишком длительные фронты входных сигналов недопустимы по паспортным данным. Нежелательны затянутые фронты и для элементов ТТЛ(Ш) с их сквозными токами. Поэтому в схеме (см. рис. 1.26) первые элементы цепи формирования имеют выход с ОК или ОС, в котором не возникают сквозные токи.

Перед повторным срабатыванием схема должна восстановиться, для чего длительность постоянного уровня входного напряжения должна быть около 3RC.

В схемах ЦУ задержки на RC-цепочках могут составлять величины до единиц миллисекунд.

Цепочки RC используются не только непосредственно, но и в форме времязадающих цепей одновибраторов, которые также являются элементами, пригодными для использования в качестве задержек цифровых сигналов (фронтов). Одновибраторы имеют одно устойчивое состояние, которое является исходным. Входной сигнал переводит одновибратор в квазиустойчивое состояние, в котором он находится в течение времени, определяемого параметрами схемы одновибратора. Затем одновибратор возвращается в свое устойчивое состояние. При этом формируется фронт, который служит выходным сигналом. Значит, длительность квазиустойчивого состояния одновибратора, т. е. длительность формируемого им одиночного импульса и есть время задержки сигнала. Одновибратор является релаксационной схемой, способной формировать крутые фронты благодаря наличию в ней положительной обратной связи.

Задержку сигнала в ЦУ при наличии обычных для них синхросигналов можно получить с помощью счетчиков. При этом входной сигнал должен разрешать работу счетчика, находящегося в нулевом исходном состоянии. Счетчик начнет подсчитывать синхросигналы, а при его переполнении выработается выходной сигнал. Таким образом, осуществится задержка $t_d = NT$, где N — емкость счетчика, T — период синхроимпульсов.

С помощью элементов задержки и простых логических схем решаются задачи формирования импульсов по длительности и генерации импульсных последовательностей.

Формирование импульсов по длительности

К задачам формирования импульсов по длительности относятся расширение, сужение и стандартизация их длительности. Эти операции реализу-

ются схемой (рис. 1.27, *а*). Если конкретизировать функцию F, считая ее дизъюнкцией, то, как видно из временных диаграмм на рис. 1.27, *б*, схема будет расширять входной импульс на интервал, равный времени задержки t_d. Если понимать под функцией F конъюнкцию и рассмотреть временные диаграммы (рис. 1.27, *в*), то можно видеть, что схема дает сужение входного импульса на величину t_d. Если $F = x_1\bar{x}_2$, то будет выполнена стандартизация длительности импульса. Выходной импульс будет иметь длительность t_d независимо от длительности входного (при $t_{вх} > t_d$). Это иллюстрируется временными диаграммами рис. 1.27, *г*. Заметим, что схема при $F = x_1\bar{x}_2$ может быть заменена сочетанием обычного конъюнктора и инвертирующей задержки.

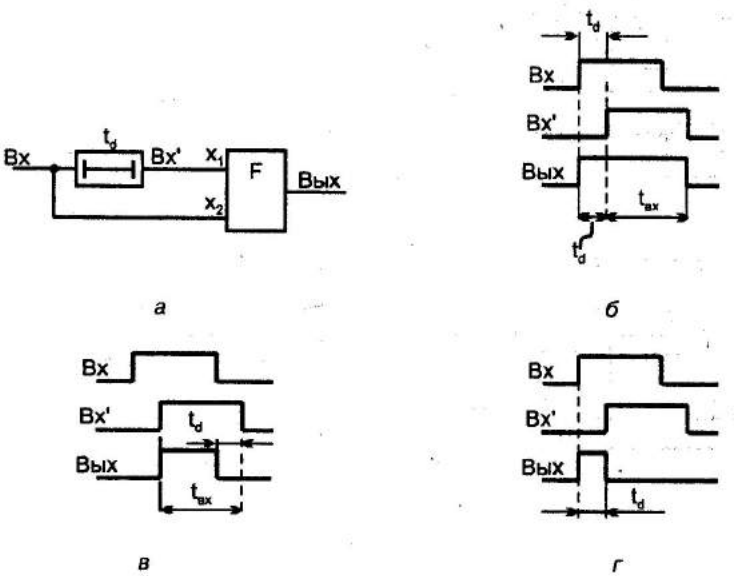


Рис. 1.27. Схема формирования импульса по длительности (*а*) и временные диаграммы реализации операций расширения (*б*), сужения (*в*) и стандартизации (*г*) импульсов

Генераторы импульсов

На элементах задержки и логических элементах строятся генераторы импульсных последовательностей. Простейший вариант показан на рис. 1.28, *а*. При нулевом значении сигнала управления Упр на выходе элемента И-НЕ имеется логическая единица, которая через обратную связь с задержкой на t_d передается на верхний вход элемента. Таким образом, в исходном состоянии верхний вход элемента И-НЕ находится в состоянии логической

единицы. Изменение управляющего сигнала является командой для начала работы генератора. Появление единицы на нижнем входе Упр элемента И-НЕ дает совпадение единиц на обоих входах, что переводит выход схемы в нулевое состояние. Это состояние длится в течение интервала t_d, т. к. после него нуль с выхода схемы по обратной связи пройдет на верхний вход элемента и поставит его в единичное состояние, которое также сохранится на время t_d, после чего изменится из-за воздействия по цепи обратной связи. Следовательно, схема будет генерировать симметричные импульсы с длительностями импульса и паузы, равными t_d (рис. 1.28, *б*).

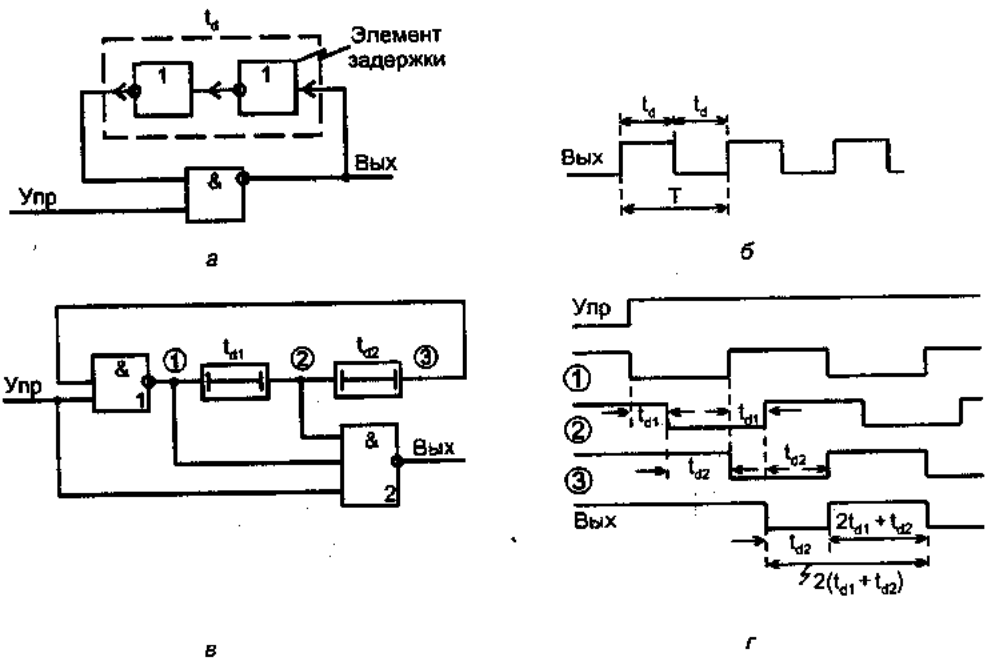


Рис. 1.28. Схемы генераторов симметричных (*а*) и несимметричных (*в*) импульсов и соответствующие временные диаграммы их выходных сигналов (*б*, *г*)

Очень часто требуются импульсы, в которых длительности импульса и паузы должны быть различны. На рис. 1.28, *в* показана схема, в которой возможно отдельное задание длительностей импульса и паузы.

Работу схемы легко уяснить из рассмотрения временных диаграмм на рис. 1.28, *г*. Видно, что длительность паузы устанавливается элементом задержки 2, после чего можно задать необходимую длительность импульса элементом задержки 1. При этом $t_п = t_{d2}$ и $t_и = 2t_{d1} + t_{d2}$. Здесь пауза короче импульса. Если требуется обратное соотношение, выходной сигнал можно проинвертировать.

На логических элементах и элементах задержки строят генераторы, к которым не предъявляются жесткие требования по стабильности частоты (допустимы отклонения порядка процентов).

Генераторами прямоугольных импульсов служат также типовые микросхемы мультивибраторов, стабильность частоты которых имеет тот же порядок, что и у ранее рассмотренных генераторов.

Для получения импульсных последовательностей с высокой стабильностью частоты применяют, как правило, *генераторы с кварцевыми резонаторами*, для которых даже без применения специальных мер нетрудно получить стабильность частоты с отклонениями порядка 10^{-5} или даже еще меньше.

Для синхронизации работы сложных устройств зачастую недостаточно иметь одну синхропоследовательность, в частности, вырабатываемую кварцевым генератором. Могут потребоваться две сдвинутые по фазе последовательности или несколько последовательностей с заданной относительной длительностью импульсов. В этом случае стабилизированная по частоте синхропоследовательность задающего генератора (обычно кварцевого) служит основой для генерации требуемого набора синхросигналов. Схема, формирующая двухфазные синхросигналы с относительными длительностями импульсов, равными 1/4, приведена в следующей главе как один из типичных примеров использования триггеров в цифровых устройствах.

Элементы индикации на светодиодах

Для общения с оператором ЦУ могут снабжаться средствами визуальной индикации символьных данных. Среди них имеются и сложные устройства, такие как экранные дисплеи, и простые, такие как светодиодные индикаторы или матрицы. Здесь же рассмотрим только простейшие индикаторы символов, которые могут встретиться проектировщику как объект самостоятельного изготовления.

Преобразование электрических сигналов в видимое изображение может быть основано на разных физических явлениях: светоизлучении полупроводниковых структур, оптических явлениях в жидких кристаллах, электролюминесценции, процессах в газовом разряде и др.

Светодиоды изготовляются на основе полупроводниковых материалов (арсенида галлия, фосфида галлия, арсенид-фосфида галлия и др.), пропускание тока через которые вызывает их свечение. Яркость свечения светодиода непосредственно зависит от величины тока. Обычно достаточны токи от единиц до приблизительно 20 миллиампер при падении напряжения на диоде около 1—2 В. Как правило, последовательно со светодиодом включается резистор, задающий и стабилизирующий ток диода.

Из нескольких диодов составляются индикаторы и матрицы, отображающие буквы и цифры. Широко применяются семисегментные индикаторы, в которых семь сегментов-диодов расположены так, что при зажигании определенной их комбинации высвечивается тот или иной символ (рис. 1.29, *а*).

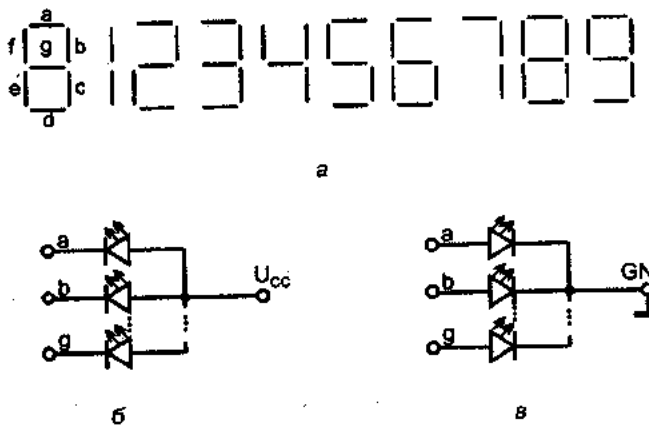


Рис. 1.29. Семисегментный индикатор и отображаемые им цифры (*а*), варианты индикатора с общим анодом (*б*) и общим катодом (*в*)

Выпускаются семисегментные индикаторы (ССИ) с общим анодом или общим катодом (рис. 1.29, *б*, *в*).

Для зажигания сегмента в схеме с общим анодом, подключенным к источнику питания U_{CC}, нужно снизить напряжение на его катоде (зажигание сигналом логического нуля). Для зажигания сегмента в схеме с общим катодом, подключенным к общей точке схемы, необходимо повысить напряжение на его аноде (зажигание сигналом логической единицы).

Для управления сегментами удобны элементы с выходом типа ОК или ОС, поскольку при их использовании имеется внешняя цепочка с резистором, сопротивление которого можно задать с учетом характеристик применяемых светодиодов.

В схеме (рис. 1.30, *а*) показано управление одним из сегментов ССИ. Диод зажигается, когда на выходе управляющего элемента напряжение равно U_0. Через диод будет протекать ток $I_д = (U_{CC} - U_д - U_0)/R$, следовательно, для его задания требуется условие $R = (U_{CC} - U_д - U_0)/I_д$. Для этой схемы требуются ССИ с общим анодом. Необходим управляющий элемент с достаточно большим выходным током в нулевом состоянии ($I_{вых.0} \geq I_д$).

В схеме (рис. 1.30, *б*) диод зажигается, когда выходной транзистор управляющего элемента запирается. Через диод течет ток $I_д = (U_{CC} - U_д)/R$,

откуда следует $R = (U_{CC} - U_д)/I_д$. Для этой схемы требуется ССИ с общим катодом. Выход управляющего элемента должен удовлетворять условию $I_{вых.0} \geq (U_{CC} - U_0)/R$.

Если выходные токи управляющих элементов недостаточны для управления диодом, между выходом элемента и сегментом индикатора можно включить буферный каскад на транзисторе. Примеры приведены на рис. 1.30, *в*, *г*.

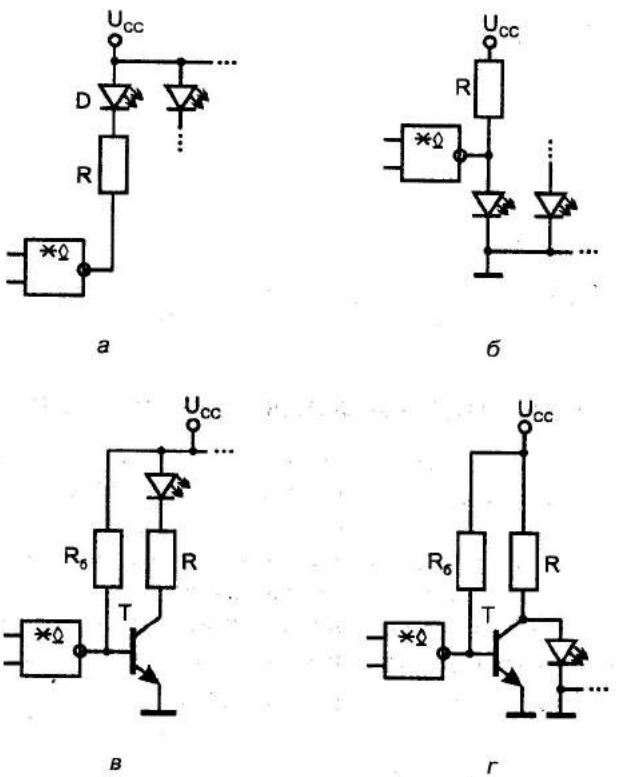


Рис. 1.30. Схемы управления сегментом индикатора с общим анодом (*а*), общим катодом (*б*) и использованием усилительных каскадов (*в*, *г*)

Для логического управления ССИ имеются стандартные ИС-дешифраторы ССИ, работающие согласно табл. 1.1. Таблица приведена для такой схемы управления светодиодами, в которой зажиганию сегмента соответствует единичное значение логического управляющего сигнала. Для схем управления, в которых светодиоды зажигаются нулевыми сигналами на выходе логического элемента, таблица видоизменяется — все значения функций для возбуждаемых сегментов с *a* по *g* должны быть проинвертированы.

Таблица 1.1

Десятичная цифра	Входной двоичный код	Возбуждаемые сегменты						
		a	*b*	*c*	*d*	*e*	*f*	*g*
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	1	0	1	1

Элементы индикации на жидких кристаллах

Второй тип индикаторов, имеющих обычные для ИС уровни управляющих сигналов, — *жидкокристаллические индикаторы* (ЖКИ). Ранее они применялись преимущественно в электронных часах, калькуляторах и измерительных приборах. С появлением портативных компьютеров с автономным питанием энергетическая экономичность жидкокристаллических индикаторов стала особенно важной, и с их использованием стали делать дисплеи — сложные периферийные устройства отображения информации ЭВМ.

Жидкий кристалл — вещество, которое при низких температурах представляет собою обычный твердый кристалл, при высоких температурах переходит в жидкое состояние, а в среднем диапазоне температур является веществом с анизотропными свойствами. Молекулы этого вещества под действием электрического поля приобретают упорядоченную ориентацию в пространстве, что изменяет оптические свойства вещества.

Индикатор ЖКИ реализуется в виде двух параллельно расположенных стеклянных пластин, на внутренних сторонах которых наклеены прозрачные электроды нужного рисунка и один общий электрод. Между пластинами размещается жидкий кристалл. Передний электрод всегда прозрачен, задний либо прозрачен (работа на просвет), либо является отражающим (работа на отражение). Если между электродами создается электрическое поле, то жидкокристаллическое вещество между ними теряет прозрачность и находящийся под напряжением электрод становится видимым. ЖКИ не излучает

свет и создаваемое в нем изображение не видно в темноте. Яркость изображения в ЖКИ тоже невелика. В то же время ЖКИ обладает большим достоинством — он потребляет очень малую мощность. Для управления этим индикатором достаточны напряжения в единицы вольт при токах порядка 10 мкА, что на несколько порядков меньше, чем токи, требующиеся для управления светодиодными индикаторами. Отсюда видна и высокая ценность ЖКИ для устройств с автономными источниками питания, в частности, для портативной аппаратуры с батарейным питанием. В ЖКИ используются семисегментные и другие индикаторы, подобные тем, которые рассмотрены выше применительно к индикаторам на светодиодах.

Напряжение, подаваемое на электроды ЖКИ, не должно содержать постоянной составляющей, т. к. это приводит к быстрому выходу ЖКИ из строя. В цифровой аппаратуре для возбуждения сегментов индикатора на них подается переменное напряжение в виде прямоугольных импульсов с частотой повторения в десятки герц. При такой частоте вследствие инерционности зрения мелькание сегментов уже не ощущается. Применение более высоких частот ведет к риску появления в напряжениях, питающих сегменты, постоянной составляющей. Действительно, для высокочастотных импульсов доля времени, приходящаяся на фронты, растет, а т. к. фронты импульсов фактически не идентичны (импульсы неидеальны по форме), и площадь положительной полуволны напряжения несколько отличается от площади отрицательной полуволны, возникающая постоянная составляющая пропорциональна частоте повторения импульсов.

Для получения переменного напряжения возбуждения сегментов ЖКИ используется фазовый метод управления. При этом на электроды передней и задней пластин ЖКИ подаются прямоугольные импульсы. Если фазы этих импульсов одинаковы, то между электродами напряжение будет отсутствовать, и сегмент не будет возбужден. Если же импульсы противофазны, то между электродами появляется переменное напряжение удвоенной амплитуды, и сегмент возбуждается. Схема управления для ЖКИ (рис. 1.31) имеет на одном из входов опорное напряжение $U_{оп}$, а на другом управляющее напряжение $U_{упр}$.

Опорное напряжение поступает на верхний вход элемента сложения по модулю 2, а управляющее — на нижний. Если управляющее напряжение равно нулю, то выходное напряжение элемента сложения по модулю 2 совпадает со входным (ХМ20 = 0), а при значении управляющего напряжения, соответствующем логической единице, выходное напряжение логического элемента инвертируется (ХМ21 = $\overline{X}$). В приведенных соотношениях М2 обозначает операцию сложения по модулю 2. На рис. 1.31 показаны также временные диаграммы действующих в схеме напряжений.

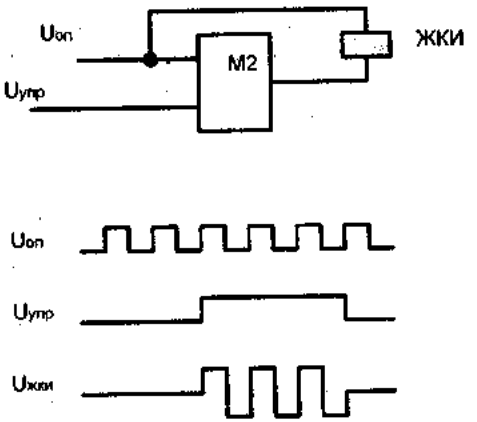


Рис. 1.31. Схема управления жидкокристаллическим индикатором

На основе светодиодов или жидкокристаллических индикаторов изготовляются как семисегментные изображения символов, так и более сложные, отображаемые возбуждением определенных сегментов из поля матрицы. Число строк и столбцов матрицы может быть различным. Для примера на рис. 1.32 показано поле размерностью 7×5, причем матрица неполная, из нее исключены 8 сегментов (дважды по 4), поскольку они не используются при отображении символов. Принципы формирования изображения при управлении сегментами матрицы те же, что и при управлении ССИ, а именно: входные коды специальным дешифратором преобразуются в сигналы возбуждения отдельных сегментов.

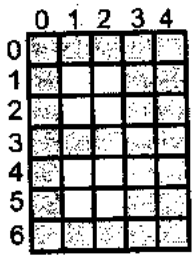


Рис. 1.32. Неполная матрица индикатора 7×5

При реализации так называемых *плоских дисплеев*, т. е. индикаторов многозначных символов, например, содержащих несколько ССИ, удобно использовать мультиплексное управление, при котором одни и те же управляющие схемы поочередно обслуживают различные ССИ, выбирая их в определенной последовательности. При этом каждый индикатор возбуждается импульсно, в течение времени 1/n, где n — число индикаторов. Иллюзия постоянного свечения всех символов создается из-за инерционности чело-

веческого зрения. Если частота возбуждения символов составляет десятки герц (современные средства визуальной индикации имеют частоты в 70—100 Гц), то мерцания изображений неощутимы.

§ 1.7. О некоторых типовых ситуациях при построении узлов и устройств на стандартных ИС

Разработанная проектировщиком функционально-логическая схема подлежит далее реализации на наборе стандартных ИС той или иной серии или на наборе библиотечных элементов той или иной БИС/СБИС с программируемой структурой. В обоих случаях возможны несовпадения элементов подлежащей изготовлению схемы и имеющихся для ее реализации. Типовыми ситуациями здесь являются наличие у имеющихся элементов "лишних" (неиспользуемых в данном случае) входов, наличие в корпусах ИС лишних элементов или, напротив, нехватка у имеющихся элементов необходимого числа входов или нагрузочной способности.

Режимы неиспользуемых входов

Вопрос о режиме "лишних" входов решается с учетом конкретного типа используемой схемотехнологии.

Пусть, например, нужно получить конъюнкцию (или ее инверсию) пяти переменных. В стандартных сериях нет соответствующих элементов с пятью входами, и поэтому придется взять элемент с восемью входами, у которого окажется три "лишних" входа. Принципиально возможно поступить следующим образом: не обращать внимания на "лишние" входы (т. е. оставить их разомкнутыми), подсоединить их к задействованным входам или подать на них некоторые константы. С точки зрения логических операций, все три возможности правомерны (рис. 1.33, *а*). Если же учесть особенности той или иной схемотехнологии, то выбор варианта действий становится определенным.

Для ЭСЛ (эмиттерно-связанная логика) решение такое: неиспользуемые входы остаются разомкнутыми. Это объясняется тем, что в схемах самих элементов уже предусмотрены специальные резисторы, связанные с источником питания, которые обеспечивают необходимые условия "лишним" входам.

Для КМОП и ТТЛ(Ш) неиспользуемые входы разомкнутыми не оставляют. Для КМОП это строгая рекомендация, т. к. у них очень велики входные сопротивления и, следовательно, на разомкнутые входы легко наводятся паразитные потенциалы, которые могут изменять работу схемы. Для ТТЛ(Ш)

строгого запрета на оставление разомкнутых входов нет, но это делать незачем, т. к. вследствие этого пострадают параметры быстродействия элемента. Подсоединение "лишних" входов к задействованным для КМОП и ТТЛ(Ш) принципиально возможно, но нежелательно, т. к. оно приводит к увеличению нагрузки на источник сигнала, что также сопровождается уменьшением быстродействия источника сигнала.

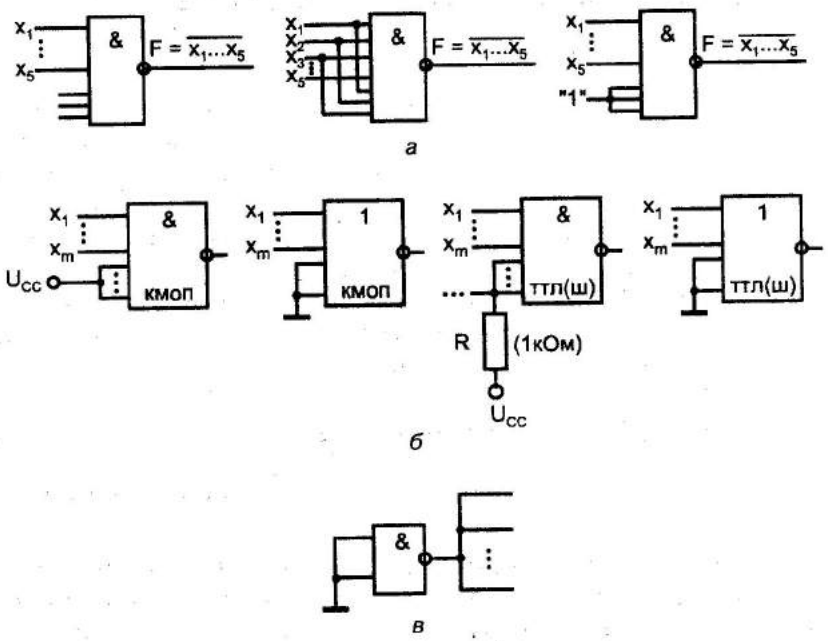


Рис. 1.33. Принципиально возможные (*а*) и рекомендуемые (*б*) режимы неиспользуемых входов логических элементов, схема формирования сигналов логической единицы (*в*)

Таким образом, для КМОП и ТТЛ(Ш) рекомендуемый режим неиспользуемых входов — подсоединение их к константам (логическим единицам или нулям), не изменяющим работу схемы для задействованных входов. При этом уровни напряжений U_1 и U_0 для КМОП близки к уровням U_{CC} и "земли", к которым и подключают неиспользуемые входы. У элементов ТТЛ(Ш) уровень U_1 на 1,5—2,5 В ниже уровня U_{CC}, поэтому в старых сериях ТТЛ(Ш), например, в серии К155 для предотвращения пробоев неиспользуемые входы подключали к источнику питания U_{CC} через резисторы R (обычно рекомендация была такой: R = 1 кОм и к одному резистору разрешается подключать до 20 входов). Для более поздних серий ТТЛ(Ш), таких как К555 и КР1533, разрешается подключать неиспользуемые входы непосредственно к напряжению питания.

Примеры, иллюстрирующие перечисленные способы подключения неиспользуемых выводов ИС, показаны на рис. 1.33, *б*. Сигналы логической единицы можно получать от специального элемента (рис. 1.33, *в*), причем, если это мощный элемент, то он может иметь коэффициент разветвления до 30.

Согласование уровней сигналов при сопряжении разнотипных элементов

Иногда в одних и тех же устройствах приходится по тем или иным соображениям применять элементы разных схемотехнологических типов. Самая типичная ситуация — одновременное использование элементов КМОП и ТТЛ(Ш). Различие элементов требует рассмотрения их совместимости по уровням напряжений, токов, быстродействию и т. д.

Согласование элементов КМОП и ТТЛ(Ш) отличается простотой. Выходные уровни U_1 и U_0 элементов КМОП близки соответственно к уровню питания и нулевому уровню, отличаясь от них на несколько процентов (например, в серии КР1554 при напряжениях питания 3—5 В $U_1 = U_{CC} - 0{,}1$ В, а $U_0 = 0{,}1$ В). При подключении к выходу элемента КМОП входов элементов ТТЛ(Ш) оказывается приемлемой прямая передача сигналов от элемента к элементу (рис. 1.34, *а*). При этом низкий нулевой уровень, поступающий от КМОП-элемента, оказывается более "хорошим", чем аналогичный уровень, получаемый от "своего" элемента ТТЛ(Ш). Высокий уровень логической единицы у элементов КМОП близок к напряжению питания, а у элементов ТТЛ(Ш) этот уровень приблизительно вдвое меньше. Повышение уровня логической единицы благоприятно для повышения помехоустойчивости схемы, но может быть опасно с точки зрения возможности пробоя входных цепей. Если повышение уровня U_1 допустимо, то прямое управление элементом ТТЛ(Ш) от элемента КМОП вполне приемлемо. В частности, такое управление рекомендуется для известных серий микросхем КР1533 и КР1554.

В сочетаниях элементов ТТЛ(Ш)—КМОП напряжение высокого уровня, формируемое выходным каскадом ТТЛ(Ш), обычно недостаточно для надлежащего управления элементами КМОП, и должно быть увеличено. В типовой схеме сопряжения (рис. 1.34, *б*) это выполняется с помощью цепочки U_{CC}—R. На первый взгляд схема рис. 1.34, *б* может показаться странной, поскольку в ней дополнительная цепочка U_{CC}—R должна воздействовать на выходное напряжение элемента ТТЛ(Ш). Выходные сопротивления элементов малы, что позволяет им работать на большие нагрузки и не поддаваться внешним воздействиям, жестко сохраняя выработанные сигналы. Поэтому, как правило, никакие сигналы на выходы элементов не подаются. Однако в рассматриваемом случае ситуация иная. Типичная выходная цепь элемента ТТЛ(Ш) изображена на рис. 1.34, *в*.

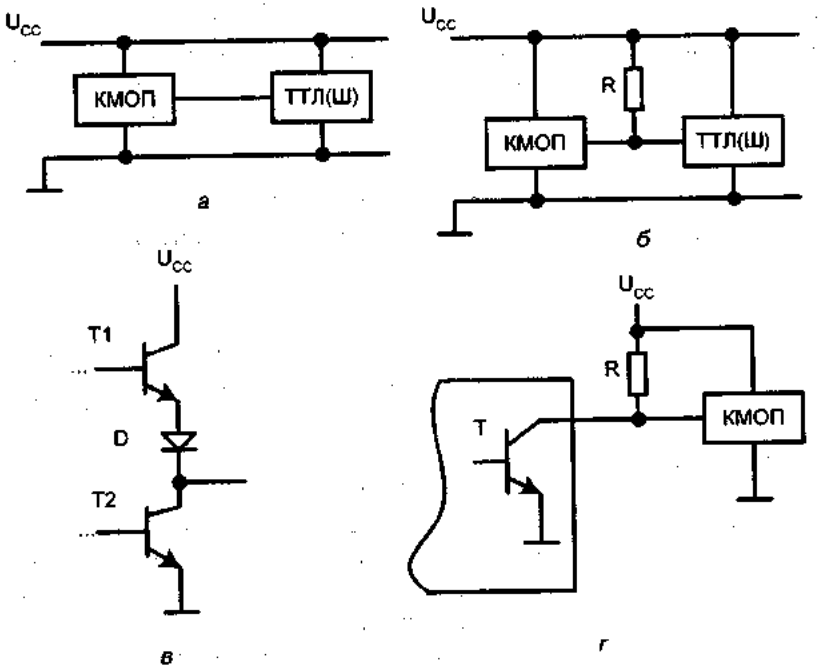


Рис. 1.34. Схемы согласования элементов КМОП и ТТЛ(Ш) (*а*, *б*) и пояснения к их работе (*в*, *г*)

При формировании высокого уровня напряжения транзистор T1 работает в схеме эмиттерного повторителя и создает малое выходное сопротивление для тока, *вытекающего из выходной цепи*. Ток, втекающий извне в выходной электрод, напротив, встречает чрезвычайно высокое сопротивление запертого транзистора T2 и сопротивление обратно включенного диода D. Такая резкая асимметрия выходных сопротивлений каскада для вытекающего тока (обычного рабочего режима) и втекающего, создаваемого цепочкой U_{CC}—R, позволяет этой цепочке определять напряжение в линии связи между элементами, задавая уровень U_1, приблизительно равный U_{CC}, что и требуется для элемента КМОП (рис. 1.34, *г*). Рекомендуемые для сопряжения элементов серий КР1533 и КР1554 значения сопротивления резистора равны приблизительно 5 кОм.

Режимы неиспользуемых элементов

Если не все элементы, имеющиеся в корпусе ИС, использованы в схеме, то неиспользованные также подключены к напряжению питания, которое является общим для всего корпуса. Если же мощности, потребляемые элемен-

тами в состояниях нуля и единицы, не равны, то имеет смысл поставить неиспользуемый элемент в состояние минимальной мощности, подав на какой-либо из его входов соответствующую константу.

Наращивание числа входов

Для элементов И и ИЛИ это не представляет трудностей: для получения нужного числа входов берется несколько элементов, выходы которых объединяются далее элементом того же типа. Наращивание числа входов для операций И-НЕ, ИЛИ-НЕ, в сущности, производится аналогичным методом, но в схеме появляются дополнительные инверторы (рис. 1.35, *а*). На этом рисунке звездочка обозначает операцию Шеффера или Пирса.

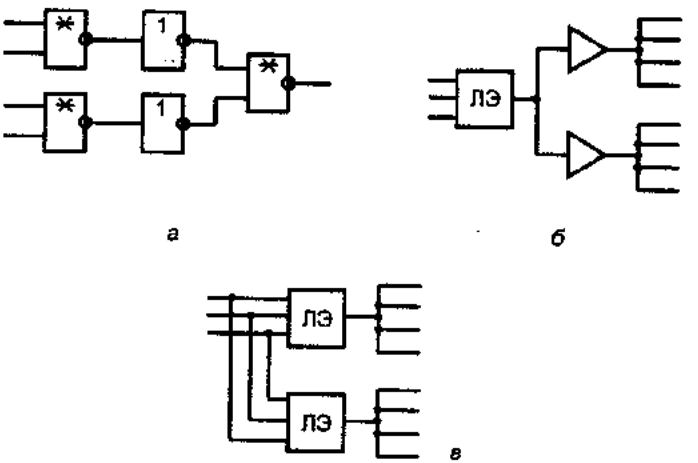


Рис. 1.35. Схемы наращивания числа входов (*а*) и снижения нагрузки на выходах логических элементов (*б*, *в*)

Снижение нагрузок на выходах логических элементов

Снижение нагрузок на выходах логических элементов может понадобиться, если нагрузки превышают допустимые значения, а также для повышения быстродействия схем, на которое нагрузки элементов оказывают самое непосредственное влияние. Чем больше у элемента — источника сигнала число нагрузок (или нестандартных внешних нагрузок), тем большее время тратится на достижение выходным сигналом порогового уровня при переключении, т. е. на изменение его логического состояния. Для предотвращения потерь

быстродействия из-за нагрузок на выходах сильно нагруженных элементов применяют буферизацию или разделение нагрузки (рис. 1.35, *б*, *в*).

Введение буферных каскадов ускоряет работу источника сигнала, но вносит собственную задержку в тракт передачи сигнала. Будет ли в конечном счете эффект ускорения, определяется конкретным расчетом.

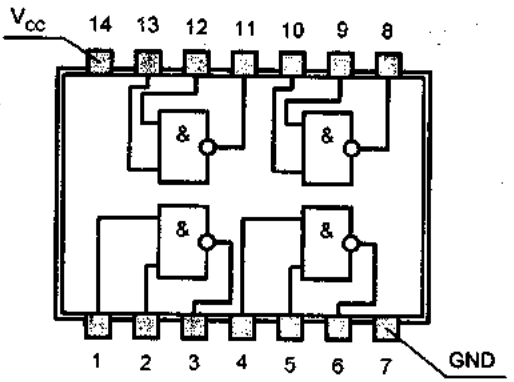
При разделении нагрузки новые элементы с задержками в тракт передачи сигнала не вводятся, но увеличивается нагрузка на тот источник сигнала, который питает рассматриваемую схему. Поэтому и здесь эффективность должна оцениваться конкретным расчетом.

Прошлое и настоящее малых и средних интегральных схем

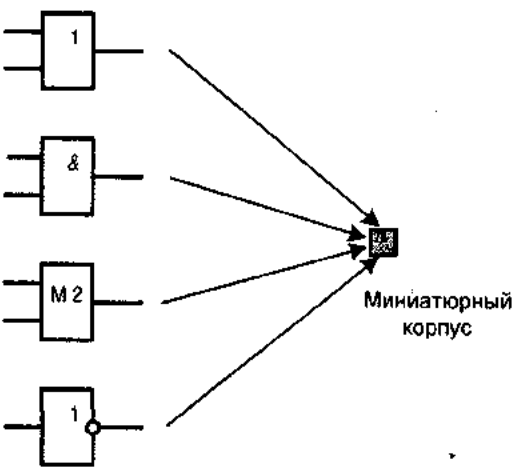
Малые и средние интегральные схемы (МИС и СИС) — старейшие на рынке цифровых элементов. Микросхемы ТТЛ появились в 1963 г., в последующие 10—15 лет были разработаны сотни их разновидностей. Позднее были освоены микросхемы типа КМОП, которые в то время проигрывали биполярным схемам по быстродействию, но всегда отличались высокой компактностью, энергетической экономичностью, высокой помехоустойчивостью, способностью работать при изменениях питающего напряжения в широких пределах. Элементы КМОП по мере повышения их быстродействия стали все более вытеснять микросхемы ТТЛ, оставляя за ними схемотехнику буферных, согласующих и других элементов, которые должны сохранять высокое быстродействие при больших нагрузках. Наряду с этими типами цифровых микросхем широко применялись и элементы ЭСЛ для особо скоростных устройств. Другие схемотехнические типы элементов применялись гораздо реже. В течение нескольких десятилетий микросхемы ТТЛ, КМОП, ЭСЛ выпускались в виде МИС и СИС, и проектировщики успели основательно привыкнуть к этому виду элементной базы.

Сейчас выпуск и использование ИС невысокого уровня интеграции не прекратились (МИС и СИС производятся девятью крупными фирмами), но их роль во многих случаях изменилась. Возможность размещать на кристалле все больше и больше транзисторов привела к развитию новых средств для построения цифровых устройств — микросхем высокого уровня интеграции с программируемой изготовителем или пользователем структурой. Новые кристаллы стали заменять все большее и большее число стандартных МИС и СИС. Основная работа в части логических преобразований информации стала выполняться большими и сверхбольшими интегральными схемами (БИС и СБИС). И где-то в середине 1990-х гг. разработка новых функциональных вариантов МИС и СИС фактически прекратилась (до этого времени серии стандартных элементов постоянно пополнялись новыми типами микросхем). Но МИС и СИС не исчезли из

сферы цифровых элементов (сохранились как их существование, так и их использование), в то время как принципы "корпусирования" под воздействием потребностей практики обогатились новыми подходами. Появилось так называемое *одновентильное корпусирование логических схем* (Single-gate Logic). Ранее, начиная с возникновения ТТЛ, проектировщики стремились получить в одном корпусе побольше элементов, и изготовители старались максимально наполнить имеющиеся в их распоряжении корпуса, размещая в них несколько идентичных элементов. Например, в корпусе с 14 выводами размещали 4 элемента 2И-НЕ (рис. 1.36, *а*).



а



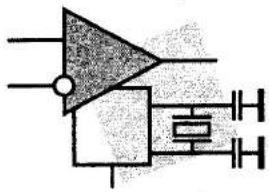
б

Рис. 1.36. Традиционное (*а*) и одновентильное (*б*) корпусирование малых интегральных схем

Такой подход естественен для ситуации, когда МИС и СИС были основой цифровых устройств. Сейчас положение иное — основная работа ложится на схемы высокого уровня интеграции. Однако ограничиться только БИС и СБИС при построении устройств и систем не удается. Практически всегда возникает потребность в реализации одиночных логических функций или преобразовании уровней сигнала или буферизации линий интерфейса и т. д. Для удовлетворения подобных потребностей *одновентильные схемы*, т. е. схемы, реализующие одну несложную функцию, размещают в сверхминиатюрные корпуса (площадью в единицы квадратных миллиметров) с поверхностным монтажом (рис. 1.36, *б*). Малые размеры микросхем в сочетании с совершенными технологическими процессами их изготовления позволяют получить элементы высокого быстродействия (задержки около 3 нс). Напряжения питания сверхминиатюрных МИС варьируются (5; 3,3; 2,5; 1,8 В с перспективами дальнейшего снижения).

Литература к главе: [2], [3], [26], [44], [46], [65].

Глава 2



Функциональные узлы комбинационного типа

§ 2.1. Введение в проблематику проектирования ЦУ комбинационного типа

Функциональные узлы, рассмотренные в этой главе, выполняют типовые для цифровых устройств операции над словами, обозначенные на иерархическом уровне языка регистровых передач.

Как и все цифровые устройства вообще, *функциональные узлы делятся на комбинационные и последовательностные.* В дальнейшем комбинационные узлы будем обозначать через КЦ (комбинационные цепи), а последовательностные через АП (автоматы с памятью). *Различия между КЦ и АП имеют фундаментальный характер.*

Выходные величины КЦ зависят только от текущего значения входных величин (аргументов). Предыстория значения не имеет. После завершения переходных процессов в КЦ на их выходах устанавливаются выходные величины, на которые характер переходных процессов влияния не оказывает. С этой точки зрения переходные процессы в КЦ не опасны. Но в ЦУ в целом КЦ функционируют совместно с АП, что кардинально меняет ситуацию. *Во время переходных процессов на выходах КЦ появляются временные сигналы, не предусмотренные описанием работы КЦ и называемые рисками.* Со временем они исчезают, и выход КЦ приобретает значение, предусмотренное логической формулой, описывающей работу цепи. Однако *риски могут быть восприняты элементами памяти АП, необратимое изменение состояния которых может радикально изменить работу ЦУ,* несмотря на исчезновение сигналов рисков на выходе КЦ.

Различают статические и динамические риски. Статические риски — это кратковременные изменения сигнала, который должен был бы оставаться неизменным (единичным или нулевым, соответственно говорят о 1-риске или 0-риске). Если согласно логике работы КЦ состояние выхода должно

измениться, но вместо однократного перехода происходят многократные, то имеет место динамический риск. При динамических рисках первый и последний переходы всегда совпадают с алгоритмическими, предусмотренными логикой работы схемы. Статический риск такого свойства не имеет и считается более неблагоприятным.

Простейший пример (рис. 2.1, *а*) соответствует выработке функции "константа 1" по формуле $F = \overline{x}\overline{\overline{x}} = 1$. В статике при любом значении x на одном из входов элемента И-НЕ имеется логический нуль, обеспечивающий единичное значение выхода. В переходных процессах возможен статический 1-риск.

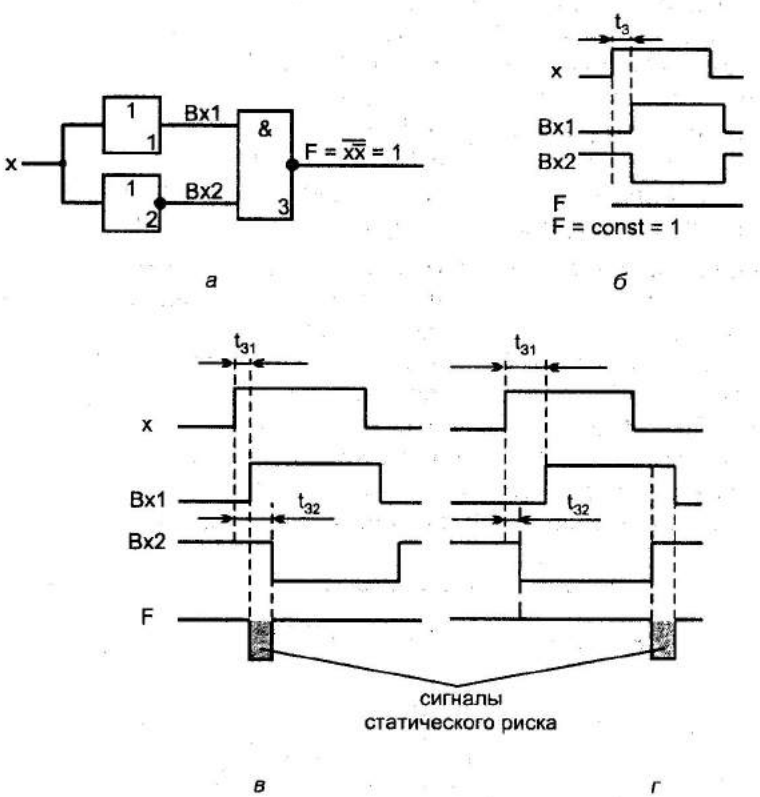


Рис. 2.1. Схема, иллюстрирующая механизм возникновения статического риска в комбинационной цепи (*а*), и временные диаграммы ее работы (*б*, *в*, *г*)

Не учитывая задержку элемента 3, которая здесь не играет роли, рассмотрим временные диаграммы переходных процессов для случаев равенства задержек элементов 1 и 2 ($t_{31} = t_{32}$) (рис. 2.1, *б*), а также их неравенства ($t_{31} < t_{32}$ и $t_{31} > t_{32}$), показанные на рис. 2.1, *в*, *г*. Видно, что при различных задержках

элементов возникает статический риск после положительного или отрицательного перепада входного сигнала в зависимости от того, задержка какого элемента больше.

Для исключения возможных сбоев в работе ЦУ из-за явлений риска имеются два пути.

Первый состоит в синтезе схем, свободных от рисков, и требует сложного анализа процессов в схеме и введения избыточных элементов для исключения рисков. Этот путь редко используется на практике.

Второй путь, основной для современной схемотехники, предусматривает *запрещение восприятия сигналов КЦ элементами памяти на время переходных процессов.* Прием информации с выходов КЦ разрешается только специальным сигналом синхронизации, подаваемым на элементы памяти после окончания переходных процессов в КЦ. Таким образом, исключается воздействие ложных сигналов на элементы памяти. Иными словами, *основная идея здесь может быть выражена словами "переждать неприятности". Соответствующие структуры называются синхронными.*

Для определения временного интервала, на котором проходят переходные процессы, следует оценить задержки на путях распространения сигналов от входов к выходам КЦ. Для примера рассмотрим рис. 2.2. Нужно взять пути с минимальной и максимальной задержками. Если на входе КЦ изменение аргументов произошло в нулевой момент времени, то по самому короткому пути до выхода F_3 сигнал может пройти за время $t_{з.2min.}$, которое и обозначит начало интервала переходных процессов. На самом длинном пути (до выхода F_1) сигнал задержится не более чем на время $t_з = t_{з.1.max} + t_{з.3.max} + t_{з.4.max}$, по истечении которого переходные процессы завершатся.

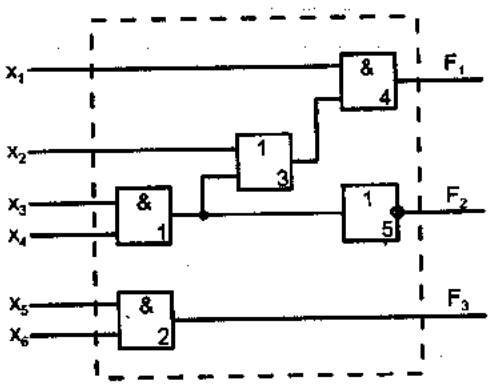


Рис. 2.2. Схема, иллюстрирующая расчет длительности переходного процесса в комбинационной цепи

В общем случае нужно оценить задержку сигнала на самом коротком пути как сумму минимальных задержек элементов, составляющих этот путь, и задержку на самом длинном пути как сумму максимальных задержек элементов этого пути.

Из приведенного примера видно, что для расчета переходных процессов в ЦУ нужны сведения о минимальных и максимальных значениях задержек элементов. К сожалению, изготовитель обычно указывает только максимальные значения задержек, иногда приводятся максимальные и типовые значения и *крайне редко имеются сведения о минимальных.* Наиболее полно описывались бы задержки статистическими характеристиками, но они, как правило, неизвестны.

Если даны только максимальные задержки, то теряется возможность сравнивать времена прохождения сигналов в разных цепях (в любой цепи задержка может быть сколь угодно малой), а это затрудняет оценку работоспособности схем и может вынудить принять не лучшие схемотехнические решения.

Для цепей из элементов с независимыми задержками отношение $t_{з.max}/t_{з.min}$ равно обычно 2—3, для элементов одного кристалла между задержками элементов возникает сильная корреляция, и отношение $t_{з.max}/t_{з.min}$ может существенно снижаться.

В состав ЦУ, как правило, входят типовые функциональные узлы и некоторое количество логических схем, специфичных для данного конкретного проекта. *Проектирование произвольной логики комбинационного типа производится по этапам.*

Прежде всего задается характер функционирования КЦ. Это может быть сделано различными способами, чаще всего пользуются таблицами функционирования (таблицами истинности), задающими значение искомых функций на всех наборах аргументов. От таблицы легко перейти к СДНФ искомых функций (СДНФ — совершенная дизъюнктивная нормальная форма, т. е. дизъюнкция конъюнктивных членов одинаковой размерности). Для этого составляют логическую сумму тех наборов аргументов, на которых функция принимает единичное значение.

Например, для подлежащей воспроизведению функции четырех аргументов, заданной табл. 2.1, получим

$$F = \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4 \vee \bar{x}_1\bar{x}_2\bar{x}_3x_4 \vee \bar{x}_1\bar{x}_2x_3\bar{x}_4 \vee \bar{x}_1\bar{x}_2x_3x_4 \vee x_1\bar{x}_2\bar{x}_3\bar{x}_4 \vee x_1\bar{x}_2\bar{x}_3x_4 \vee x_1x_2\bar{x}_3x_4 \vee x_1x_2x_3x_4.$$

Таблица 2.1

x_1	x_2	x_3	x_4	F	x_1	x_2	x_3	x_4	F
0	0	0	0	1	0	0	1	0	1
0	0	0	1	1	0	0	1	1	1

Таблица 2.1 *(окончание)*

x_1	x_2	x_3	x_4	F	x_1	x_2	x_3	x_4	F
0	1	0	0	0	1	0	1	0	0
0	1	0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	0	0	0
0	1	1	1	0	1	1	0	1	1
1	0	0	0	1	1	1	1	0	0
1	0	0	1	1	1	1	1	1	1

Дальнейшие действия зависят от средств реализации функций, к которым в современной схемотехнике относятся следующие.

- Логические блоки табличного типа (LUTs, Look-Up Tables).
- Логические блоки в виде последовательности матриц элементов И и ИЛИ (PLA, Programmable Logic Array; PAL, Programmable Array Logic).
- Универсальные логические блоки на основе мультиплексоров.
- Логические блоки, собираемые из логических элементов некоторого базиса (SLC, Small Logic Cells).

Если КЦ будет реализована на основе логических блоков табличного типа, то СДНФ явится окончательным выражением функции, и никаких дальнейших преобразований этой формы не потребуется. Дело в том, что табличный блок представляет собою память, в которой имеется столько ячеек, сколько необходимо для хранения всех значений функций, т. е. 2^m, где m — число аргументов функции. Набор аргументов является адресом той ячейки, в которой хранится значение функции на этом наборе (0 или 1). СДНФ как раз и содержит все адреса, по которым нужно хранить единичные значения функции. Если искомая функция выражена в какой-либо сокращенной форме, то следует перевести ее в СДНФ. Для этого конъюнктивные члены, не содержащие переменной x_j, умножаются на равную единице дизъюнкцию $x_j \vee \bar{x}_j$. Например,

$$F(x_1x_2x_3) = x_1 \vee \bar{x}_2\bar{x}_3 = x_1(x_2 \vee \bar{x}_2)(x_3 \vee \bar{x}_3) \vee x_2\bar{x}_3(x_1 \vee \bar{x}_1) =$$
$$= x_1x_2x_3 \vee x_1\bar{x}_2x_3 \vee x_1x_2\bar{x}_3 \vee x_1\bar{x}_2\bar{x}_3 \vee x_1x_2\bar{x}_3 \vee \bar{x}_1x_2\bar{x}_3.$$

Блок памяти для воспроизведения функции m переменных имеет вид рис. 2.3, *а*. Если требуется воспроизвести n функций, то в каждой ячейке нужно будет хранить n бит (по одному биту для каждой функции), и блок памяти будет организован, как показано на рис. 2.3, *б*.

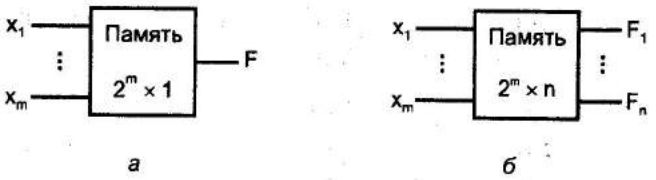


Рис. 2.3. Блоки памяти для воспроизведения одной (*а*) и нескольких (*б*) логических функций

Если размерность блоков табличного типа такова, что не позволяет получить искомую функцию с помощью одного блока, т. е. число входов блока памяти меньше числа аргументов функций, то появляется необходимость решения сложной задачи выражения искомой функции через подфункции с меньшим числом аргументов.

Если проект реализуется на логических блоках в виде последовательно включенных матриц элементов И и ИЛИ либо их эквивалента в другом базисе, то исходную СДНФ можно минимизировать, если, конечно, возникает такая необходимость. Логические блоки с матрицами И и ИЛИ воспроизводят системы переключательных функций и имеют параметры: число входов, выходов и термов. Число входов (аргументов воспроизводимых функций) и число выходов (самих функций) от формы выражения функций не зависят и предопределены заданием. Число термов (имеются в виду конъюнктивные термы) зависит от формы представления функций системы. Если число термов при данной форме представления функций превышает возможности логического блока, то возникает вопрос о минимизации функций. *Целью минимизации будет сокращение числа конъюнктивных термов в данной системе функций, т. е. поиск кратчайших дизъюнктивных форм.* Практически это сводится к поиску минимальных форм дизъюнктивных нормальных форм (ДНФ), о чем говорится далее, и отбору среди них вариантов с достаточно малым числом термов.

Как только находится форма с достаточно малым числом термов, поиск других форм можно прекратить, т. к. дальнейшее уменьшение числа термов системы эффекта не даст: сложность аппаратных средств воспроизведения системы уже не уменьшится. Разумеется, речь идет о реализациях на уже выбранных средствах, а не о том, что могут быть применены иные логические блоки — того же типа, но иной размерности.

Логические блоки на основе мультиплексоров рассмотрены в § 2.5 после ознакомления с самими ИС мультиплексоров.

Синтез КЦ на логических блоках типа SLC, т. е. на вентильном уровне, является самым традиционным и изученным (термином "вентиль" называют базовые логические ячейки, выполняющие простейшие операции, для многих ИС эту роль играют элементы И-НЕ с двумя-тремя входами).

В этом варианте проектирование КЦ содержит следующие этапы:

- минимизацию логических функций;
- переход к заданному логическому базису.

Минимизация в широком смысле слова — такое преобразование логических функций, которое упрощает их в смысле заданного критерия. Исторически первым было стремление минимизировать число логических элементов в схеме (элементы были наиболее дорогими компонентами устройств), что приводит к критерию сложности схемы в виде числа букв в реализуемых выражениях. Этот критерий учитывается так называемой ценой по Квайну — суммарным числом входов всех логических элементов схемы. Для минимизации по этому критерию разработано несколько методов, в их числе как аналитические, основанные на преобразованиях математических выражений, так и графические, основанные на применении специальных карт (карт Карно, диаграмм Вейча), удобных в использовании, если число аргументов функции не превышает 6.

С переходом на ИС и ростом уровня их интеграции критерием аппаратной сложности ЦУ стала площадь, затрачиваемая на их размещение. При этом для ИС, реализуемых непосредственно на кристалле, площадь имеет прямой физический смысл и измеряется чаще всего в квадратных миллиметрах. Для устройств, реализуемых на печатной плате, "площадь" измеряется числом корпусов в составе ЦУ. Так как корпуса ИС неодинаковы, их следует приводить к некоторым эквивалентным корпусам. Приведение учитывает число выводов корпуса, так, например, корпус с 24 выводами в 1,5 раза сложнее корпуса с 16 выводами. Понятно, что операции приведения соответствует *оценка суммарной площади корпусов ЦУ по общему числу всех выводов корпусов ИС.*

Минимизация по числу букв в реализуемом выражении перестала точно соответствовать новому критерию, хотя между обоими критериями сохраняется известная связь.

Следующий этап проектирования — переход к заданному логическому базису от исходных выражений, которые обычно получают в булевском базисе (И, ИЛИ, НЕ). Правила такого перехода известны, они основаны на применении теоремы де-Моргана. В частности, для перехода к базису И-НЕ используется соотношение

$$F(x_1,x_2,x_3,x_4) = x_1x_2 \vee x_3x_4 = \overline{\overline{x_1x_2 \vee x_3x_4}} = \overline{\overline{x_1x_2} \cdot \overline{x_3x_4}},$$

а для перехода к базису Пирса удобно вначале получить исходную булевскую форму для инверсии искомой функции, а затем от нее перейти к базису ИЛИ-НЕ по соотношениям

$$F = x_1x_2 \vee x_3x_4,\ \overline{F} = \overline{x_1x_2 \vee x_3x_4} = \overline{\overline{\overline{x_1x_2} \vee \overline{x_3x_4}}} = \overline{\overline{x_1} \vee \overline{x_2} \vee \overline{x_3} \vee \overline{x_4}}.$$

Традиционные методы минимизации функций алгебры логики приводят к каноническим их формам, соответствующим двухъярусной (если входные переменные заданы и прямыми, и инверсными значениями) реализации путем последовательного выполнения операций И и ИЛИ. Переход к базисам И-НЕ и ИЛИ-НЕ ярусность схем (их логическую глубину) не изменяет. Для построения простых схем или схем на некоторых видах программируемой матричной логики такое представление может служить в качестве окончательного варианта. Для некоторых задач каноническое представление может оказаться слишком громоздким. Для упрощения выражений можно применять к ним факторизацию (вынесение общих множителей за скобки и группирование членов), различного рода эквивалентные подстановки и др. Упрощение функций путем факторизации может дать большой эффект, но при этом увеличивается ярусность схем и, следовательно, возрастает задержка в выработке результата.

Возможные преобразования функций порождают необозримое множество вариантов, причем наиболее ценные отнюдь не лежат на поверхности. При поиске таких вариантов проектировщик не имеет теоретических подсказок и действует эвристически.

> В работе [33] сказано (с. 6): "Примером исчисления, которым широко пользуются в процессе синтеза логических схем, являются преобразования алгебры логики. Набор правил говорит лишь о том, как можно преобразовать исходное булевское выражение, но ничего не говорит о том, как нужно его преобразовать, чтобы на данном логическом базисе получить минимальную задержку или минимальное число корпусов, или некоторый компромисс между этими требованиями".

К проблематике проектирования ЦУ относится и вопрос о *критериях их качества*. Поскольку одну и ту же задачу можно решить многими способами, возникают альтернативные варианты проекта, которые нужно уметь сравнивать между собой. Объективная сложность сравнительной оценки вариантов обусловлена тем, что при этом имеет значение целый набор свойств для каждого варианта — *частных критериев* его качества. Каждый частный критерий имеет ясный, определенный смысл (аппаратная сложность, быстродействие, потребляемая мощность, помехоустойчивость и др.), но не может исчерпывающим образом охарактеризовать вариант. А чтобы учесть несколько частных критериев качества, нужно сформировать *общий критерий* (интегральный, многоцелевой, функцию качества, функцию ценности). Формирование такого критерия — чрезвычайно ответственная задача, не имеющая формального решения. *В любую форму общего критерия качества входят коэффициенты, назначаемые субъективно.* Таким образом, возникает ситуация, когда для оценки устройства применяется критерий, а для него самого оценки качества не существует. Поэтому в практике проектирования сложные общие критерии качества не популярны. Достаточно признанным можно, пожалуй, считать лишь критерий АТ, где А — аппаратная сложность устройства, Т — время решения задачи. Да и то здесь так же проявляется

общий недостаток, свойственный всем общим критериям — в них может происходить взаимная компенсация частных критериев, и уменьшение одного может быть скомпенсировано ростом другого, что формально равноценно, но не всегда разумно.

§ 2.2. Двоичные дешифраторы

Дешифраторы относятся к преобразователям кодов. Двоичные дешифраторы преобразуют двоичный код в код "1 из N". В кодовой комбинации этого кода только одна позиция занята единицей, а все остальные — нулевые. Например, код "1 из N", содержащий 4 кодовых комбинации, будет представлен следующим образом:

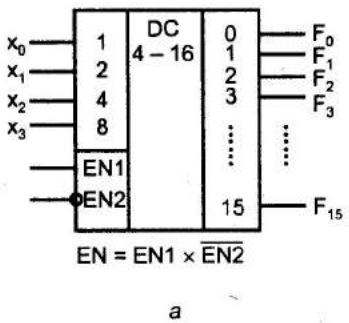
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Из сказанного видно, что двоичный дешифратор, имеющий n входов, должен иметь 2^n выходов, соответствующих числу разных комбинаций в n-разрядном двоичном коде.

В зависимости от входного двоичного кода на выходе дешифратора возбуждается одна и только одна из выходных цепей.

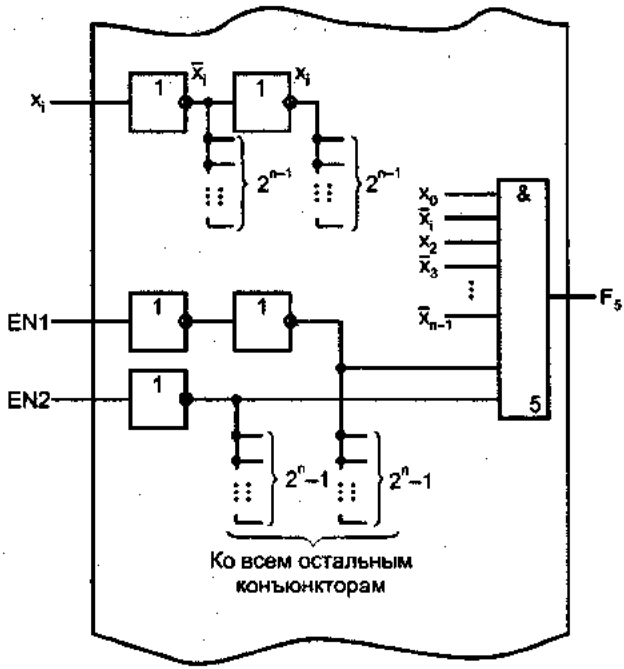
Если часть входных наборов не используется, то дешифратор называют *неполным*, и у него число выходов меньше 2^n.

В условном обозначении дешифраторов проставляются буквы DC (от англ. *Decoder*). Входы дешифратора принято обозначать их двоичными весами. Кроме информационных входов дешифратор обычно имеет один или более входов разрешения работы обозначаемых как EN (Enable).



а

Рис. 2.4. Условное обозначение двоичного дешифратора (*а*)



б

Рис. 2.4. Схемная реализация двоичного дешифратора (*б*)

При наличии разрешения по этому входу дешифратор работает описанным выше образом, при его отсутствии все выходы дешифратора пассивны. Если входов разрешения несколько, то сигнал разрешения работы образуется как конъюнкция сигналов отдельных входов. Условное графическое обозначение полного двоичного дешифратора показано на рис. 2.4, *а*. Часто дешифратор имеет инверсные выходы. В этом случае только один выход имеет нулевое значение, а все остальные — единичное. При запрещении работы дешифратора на всех его выходах будет присутствовать логическая единица.

Функционирование дешифратора описывается системой конъюнкций:

$$
\begin{aligned}
F_0 &= \bar{x}_0\bar{x}_1\bar{x}_2\ldots\bar{x}_{n-2}\bar{x}_{n-1}EN;\\
F_1 &= \bar{x}_0\bar{x}_1\bar{x}_2\ldots\bar{x}_{n-2}x_{n-1}EN;\\
F_2 &= \bar{x}_0\bar{x}_1\bar{x}_2\ldots x_{n-2}\bar{x}_{n-1}EN;\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
F_{2^n-2} &= x_0x_1x_2\ldots x_{n-2}\bar{x}_{n-1}EN;\\
F_{2^n-1} &= x_0x_1x_2\ldots x_{n-2}x_{n-1}EN.
\end{aligned}
$$

Схемотехническая реализация дешифраторов

Схемотехнически дешифратор представляет собою совокупность конъюнкторов (или элементов И-НЕ в дешифраторах с инверсными выходами), не связанных между собой. Каждый конъюнктор (или элемент И-НЕ) вырабатывает одну из выходных функций.

Заметим, что дешифраторы с прямыми выходами, построенные на основе конъюнкторов, удобнее других для рассмотрения и функционального описания, в силу чего и описаны в этом параграфе. В то же время для многих схемотехнических реализаций операция И-НЕ оказывается более удобной, чем операция И. Поэтому в стандартных сериях элементов дешифраторы с инверсными выходами встречаются значительно чаще, чем дешифраторы с прямыми выходами.

Кроме элементов для выработки выходных функций, дешифратор, как и многие другие ИС, снабжен схемами для выработки парафазных сигналов из однофазных (прямых), поступающих на входы ИС. Заметим, что входная прямая переменная непосредственно в схеме не используется, а вырабатывается повторно как двойная инверсия от входной. Это сделано для того, чтобы максимально разгружать линии внешних входов (здесь внешние входы нагружены только на один вход инвертора). Внешние линии входов максимально разгружаются всегда, поскольку они и без того нагружены емкостью из-за своей относительно большой длины и конструкции выводов корпуса ИС, что снижает скорость передачи сигнала по линии.

Сокращенная схема дешифратора с показом одной входной линии и одной выходной линии (для определенности взята линия с номером 5) дана на рис. 2.4, *б*. Время установления выходного сигнала дешифратора

$$t_{DC} = 2t_{з.инв.} + \max(t_з^{10}, t_з^{01}),$$

где $t_{з.инв.}$ — задержка сигнала в инверторе; $t_з^{10}$, $t_з^{01}$ — задержки переключений логического элемента.

Как известно, корпуса ИС с большим числом выводов изготовлять сложно, и они дороги. С этой точки зрения дешифраторы относятся к крайне неудачным схемам, т. к. у них при простой внутренней структуре и малом числе схемных элементов много внешних выводов. Для размещения в обычном недорогом корпусе годится только дешифратор с 4 информационными входами. Более "размерных" дешифраторов в сериях ИС нет.

Наращивание размерности дешифратора

Малоразрядность стандартных дешифраторов ставит вопрос о наращивании их разрядности. Из малоразрядных дешифраторов можно построить схему, эквивалентную дешифратору большей разрядности. Для этого входное слово делится на поля. Разрядность поля младших разрядов соответствует числу входов имеющихся дешифраторов. Оставшееся поле старших разрядов служит для по-

лучения сигналов разрешения работы одного из дешифраторов, декодирующих поле младших разрядов.

В качестве примера на рис. 2.5 приведена схема дешифрации пятиразрядного двоичного кода с помощью дешифраторов "3–8" и "2–4". Для получения нужных 32 выходов составляется столбец из четырех дешифраторов "3–8". Дешифратор "2–4" принимает два старших разряда входного кода. Возбужденный единичный выход этого дешифратора отпирает один из дешифраторов столбца по его входу разрешения. Выбранный дешифратор столбца расшифровывает три младших разряда входного слова.

Каждому входному слову соответствует возбуждение только одного выхода. Например, при дешифрации слова $x_4x_3x_2x_1x_0 = 11001_2 = 25_{10}$ на входе дешифратора первого яруса имеется код 11, возбуждающий его выход номер три (показано крестиком), что разрешает работу DC4. На входе DC4 действует код 001, поэтому единица появится на его первом выходе, т. е. на 25 выходе схемы в целом, что и требуется.

Общее разрешение или запрещение работы схемы осуществляется по входу EN дешифратора первого яруса.

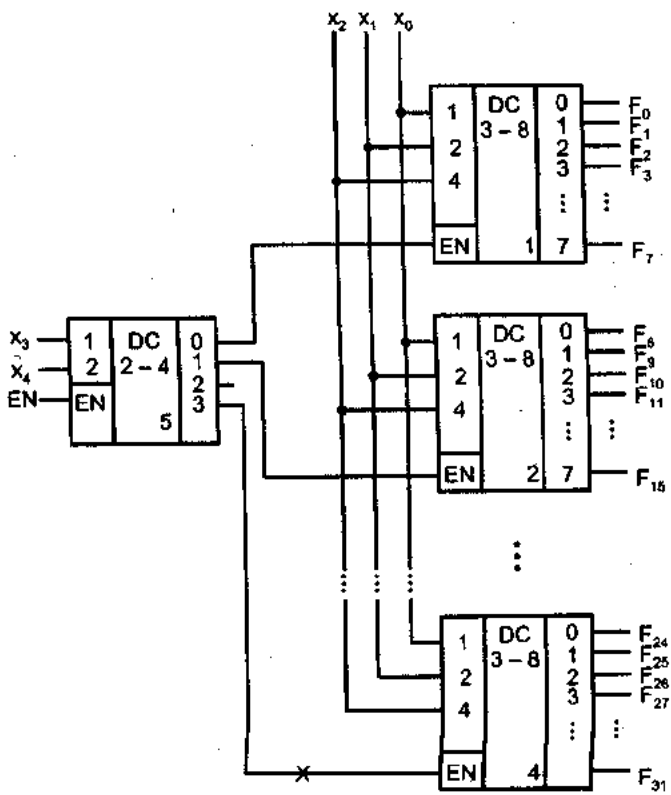


Рис. 2.5. Схема наращивания размерности двоичного дешифратора

Дешифраторы совместно со схемами ИЛИ можно использовать для воспроизведения произвольных логических функций. Действительно, на выходах дешифратора вырабатываются все конъюнктивные термы (конституенты единицы), которые только можно составить из данного числа аргументов. Логическая функция в СНДФ есть дизъюнкция некоторого числа таких термов. Собирая нужные термы по схеме ИЛИ, можно получить любую функцию данного числа аргументов.

На рис. 2.6 в качестве примера показана схема выработки двух функций $F_1 = \bar{x}_3\bar{x}_2 \vee x_3x_1$ и $F_2 = \bar{x}_3\bar{x}_2x_1 \vee x_2\bar{x}_1$. Такое решение может быть целесообразным при необходимости выработки нескольких функций одних и тех же аргументов. В этом случае для выработки дополнительной функции добавляется только один дизъюнктор. Заметим, что для проверки правильности схемы рис. 2.6 удобно перевести функции F_1 и F_2 в СДНФ.

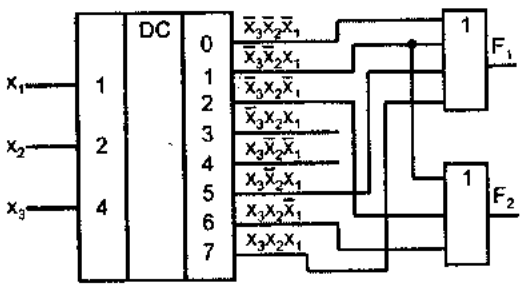


Рис. 2.6. Схема воспроизведения произвольных логических функций с помощью дешифратора и дизъюнкторов

§ 2.3. Приоритетные и двоичные шифраторы. Указатели старшей единицы

Двоичные шифраторы выполняют операцию, обратную по отношению к операции дешифратора: они преобразуют код "1 из N" в двоичный. При возбуждении одного из входов шифратора на его выходе формируется двоичный код номера возбужденной входной линии. Полный двоичный шифратор имеет 2^n входов и n выходов.

Приоритетные шифраторы выполняют более сложную операцию. При работе ЭВМ и в других устройствах часто решается задача определения приоритетного претендента на пользование каким-либо ресурсом. Несколько конкурентов выставляют свои запросы на обслуживание, которые не могут быть удовлетворены одновременно. Нужно выбрать того, кому предоставляется право первоочередного обслуживания. Простейший вариант решения ука-

занной задачи — присвоение каждому источнику запросов фиксированного приоритета. Например, группа из восьми запросов R_7—R_0 (R от англ. *request*) формируется так, что высший приоритет имеет источник номер семь, а далее приоритет уменьшается от номера к номеру. Самый младший приоритет у нулевого источника — он будет обслуживаться только при отсутствии всех других запросов. Если имеются одновременно несколько запросов, обслуживается запрос с наибольшим номером.

Приоритетный шифратор вырабатывает на выходе двоичный номер старшего запроса.

Легко видеть, что при наличии всего одного возбужденного входа приоритетный шифратор работает так же, как и двоичный. Поэтому в сериях элементов двоичный шифратор как самостоятельный элемент может отсутствовать. Режим его работы — частный случай работы приоритетного шифратора.

Указатели старшей единицы решают в сущности ту же задачу, что и приоритетные шифраторы, но вырабатывают результат в иной форме — в виде кода "1 из N". Таким образом, *при наличии на входах нескольких возбужденных линий (запросов) на выходе будет возбуждена лишь одна, соответствующая старшему запросу.* Число входов в этом случае равно числу выходов схемы. Указатели старшей единицы применяются в устройствах нормализации чисел с плавающей точкой, схемах аппаратного поллинга и т. д.

В промышленных сериях элементов имеются приоритетные шифраторы для восьмиразрядных и десятиразрядных слов. Функционирование их отображается в табл. 2.2.

Таблица 2.2

EI	R_7	R_6	R_5	R_4	R_3	R_2	R_1	R_0	a_2	a_1	a_0	G	EO
1	1	X	X	X	X	X	X	X	1	1	1	1	0
1	0	1	X	X	X	X	X	X	1	1	0	1	0
1	0	0	1	X	X	X	X	X	1	0	1	1	0
1	0	0	0	1	X	X	X	X	1	0	0	1	0
1	0	0	0	0	1	X	X	X	0	1	1	1	0
1	0	0	0	0	0	1	X	X	0	1	0	1	0
1	0	0	0	0	0	0	1	X	0	0	1	1	0
1	0	0	0	0	0	0	0	1	0	0	0	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	1
0	X	X	X	X	X	X	X	X	0	0	0	0	0

Таблица полностью характеризует работу приоритетного шифратора при всех возможных комбинациях сигналов: EI — сигнала разрешения работы

данного шифратора; EO — сигнала, вырабатываемого на выходе данного шифратора при отсутствии запросов на его входах для разрешения работы следующего (младшего) шифратора при наращивании размерности шифраторов; G — сигнала, отмечающего наличие запросов на входе данного шифратора; R_7—R_0 — запросов на входах шифратора; a_2—a_0 — значений разрядов выходного двоичного кода, формирующего номер старшего запроса. Все перечисленные сигналы формируются при условии EI = 1 (работа шифратора разрешена). При EI = 0 независимо от состояний входов запросов все выходные сигналы шифратора становятся нулевыми.

Из таблицы можно получить следующие выражения для функций a_2, a_1, a_0, EO, G

$$a_2 = (R_7 \vee \overline{R}_7 R_6 \vee \overline{R}_7 \overline{R}_6 R_5 \vee \overline{R}_7 \overline{R}_6 \overline{R}_5 R_4)EI;$$
$$a_1 = (R_7 \vee \overline{R}_7 R_6 \vee \overline{R}_7 \overline{R}_6 R_5 \overline{R}_4 R_3 \vee \overline{R}_7 \overline{R}_6 \overline{R}_5 R_4 \overline{R}_3 R_2)EI;$$
$$a_0 = (R_7 \vee \overline{R}_7 \overline{R}_6 R_5 \vee \overline{R}_7 \overline{R}_6 R_5 \overline{R}_4 R_3 \vee \overline{R}_7 R_6 \overline{R}_5 \overline{R}_4 \overline{R}_3 \overline{R}_2 R_1)EI;$$
$$EO = \overline{R}_7 \overline{R}_6 \overline{R}_5 \overline{R}_4 \overline{R}_3 \overline{R}_2 \overline{R}_1 \overline{R}_0 EI;$$
$$G = (R_7 \vee R_6 \vee R_5 \vee R_4 \vee R_3 \vee R_2 \vee R_1 \vee R_0)EI.$$

Повторным применением к каждой из функций a_i (i = 0, 1, 2) известного соотношения алгебры логики $a \vee F\bar{a} = a \vee F$ можно упростить их и получить выражения

$$a_2 = R_7 \vee R_6 \vee R_5 \vee R_4;$$
$$a_1 = R_7 \vee R_6 \vee \overline{R}_5 \overline{R}_4 R_2 \vee \overline{R}_5 \overline{R}_4 R_3;$$
$$a_0 = (R_7 \vee \overline{R}_6 R_5 \vee \overline{R}_6 \overline{R}_4 R_3 \vee \overline{R}_6 \overline{R}_4 \overline{R}_2 R_1,$$

которые определяют внутреннюю структуру шифратора приоритета в его основной части.

Наращивание размерности приоритетного шифратора

Условное обозначение шифратора приоритета показано на рис. 2.7, на котором изображено наращивание числа входов запросов вдвое (от 8 до 16). При этом показаны шифраторы с инверсными входами и выходами, как это свойственно большинству серий элементов.

Шифратор 2 — старший по приоритету, его работа всегда разрешена подачей нуля на вход EI2. Если на входах $\overline{R}_8$...$\overline{R}_{15}$ есть хотя бы один запрос, то разрешения на работу младшего шифратора 1 нет (EO2 = 1). Выходы шифратора 1 пассивны, т. е. имеют единичные значения. При этом элементы И-НЕ с номерами 1, 2, 3 играют роль инверторов для сигналов a_{i2} (i = 0, 1, 2). Поэтому на выходах a_0, a_1, a_2 схемы в целом формируются сигналы от нуля до семи в зависимости от номера старшего запроса в шифраторе 2, что вместе с единицей на выходе EO2 дает номера от 8 до 15.

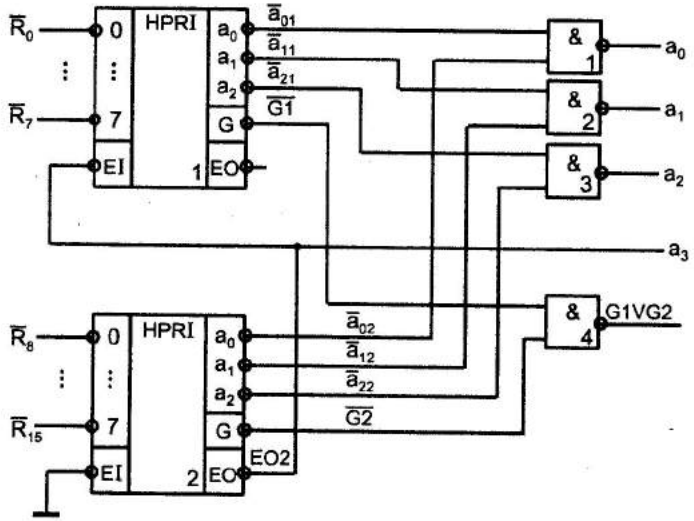


Рис. 2.7. Схема наращивания размерности приоритетного шифратора

Если на входах шифратора 2 запросов нет, он разрешает работу младшего, вырабатывая сигнал EO2 = 0, и приводит свои выходы a_0, a_1, a_2 в пассивное единичное состояние. Теперь на выходы a_i схемы в целом передаются инвертированные значения выходов a_{01}, a_{11}, a_{21} младшего шифратора, что вместе с нулем в разряде a_3 соответствует номерам от нуля до семи.

Таким образом, строится схема с 16 входами запросов, причем вход $\overline{R}_{15}$ имеет старший приоритет. Выход элемента 4 принимает единичное значение при наличии хотя бы одного запроса в любом из шифраторов и может использоваться как сигнал запроса на прерывания для процессора с последующим указанием процессору номера старшего запроса.

Указатели старшей единицы могут быть реализованы подключением двоичного дешифратора к выходу шифратора приоритета, но эту же задачу можно решить и с помощью специальной цепочечной схемы (дейзи-цепочки) путем последовательного опроса разрядов, начиная со старшего, и прекращения дальнейшего опроса при выявлении первой же единицы. Схема дейзи-цепочки рассмотрена в § 6.5 при описании устройств, обслуживающих процессы прерывания программы.

§ 2.4. Мультиплексоры и демультиплексоры

Мультиплексоры осуществляют подключение одного из входных каналов к выходному под воздействием соответствующего управляющего (адресующего)

слова. Разрядности каналов могут быть различными, мультиплексоры для коммутации многоразрядных слов составляются из одноразрядных.

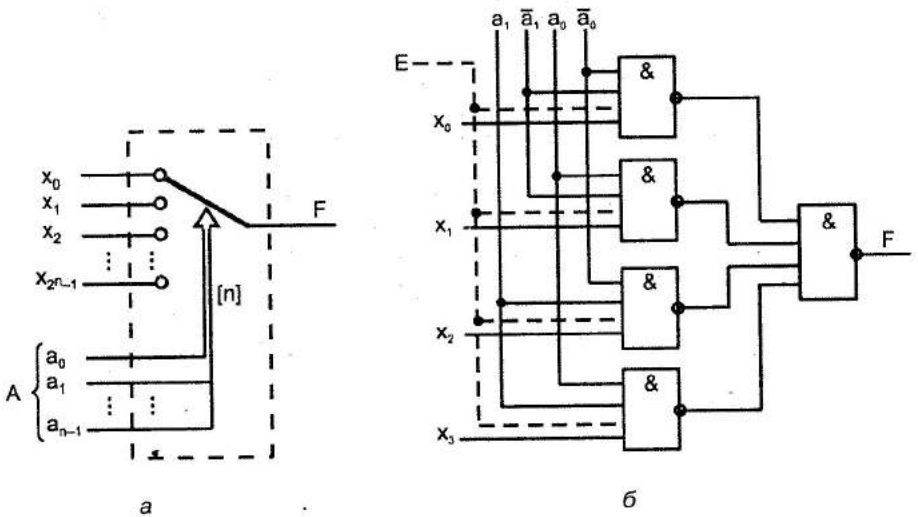


Рис. 2.8. Упрощенное представление мультиплексора многопозиционным ключом (*а*) и реализация мультиплексора на элементах И-НЕ (*б*)

Входы мультиплексора делятся на две группы: информационные и адресующие. Работу мультиплексора можно упрощенно представить с помощью многопозиционного ключа. Для одноразрядного мультиплексора это представлено на рис. 2.8, *а*. Адресующий код A задает переключателю определенное положение, соединяя с выходом F один из информационных входов x_i. При нулевом адресующем коде переключатель занимает верхнее положение x_0, с увеличением кода на единицу переходит в соседнее положение x_1 и т. д.

Работа мультиплексора описывается соотношением

$$F = x_0\bar{a}_{n-1}\bar{a}_{n-2}\ldots\bar{a}_1\bar{a}_0 \vee x_1\bar{a}_{n-1}\bar{a}_{n-2}\ldots\bar{a}_1 a_0 \vee \ldots \vee x_{2^n-1}a_{n-1}a_{n-2}\ldots a_1 a_0,$$

которое иногда называется *мультиплексной формулой*. При любом значении адресующего кода все слагаемые, кроме одного, равны нулю. Ненулевое слагаемое равно x_i, где i — значение текущего адресного кода.

Схемотехнически мультиплексор реализует электронную версию показанного переключателя, имея, в отличие от него, только одностороннюю передачу данных. На рис. 2.8, *б* показан мультиплексор с четырьмя информационными входами, двумя адресными входами и входом разрешения работы.

При отсутствии разрешения работы (E = 0) выход F становится нулевым независимо от информационных и адресных сигналов.

В стандартных сериях размерность мультиплексоров не более "16—1".

Наращивание размерности

Наращивание размерности мультиплексоров возможно с помощью пирамидальной структуры из нескольких мультиплексоров. При этом первый ярус схемы представляет собою столбец, содержащий столько мультиплексоров, сколько необходимо для получения нужного числа информационных входов. Все мультиплексоры столбца адресуются одним и тем же кодом, составленным из соответствующего числа младших разрядов общего адресного кода (если число информационных входов схемы равно 2^n, то общее число адресных разрядов равно n, младшее поле n_1 адресного кода используется для адресации мультиплексоров первого яруса). Старшие разряды адресного кода, число которых равно $n - n_1$, используются во втором ярусе, мультиплексор которого обеспечивает поочередную работу мультиплексоров первого яруса на общий выходной канал.

Пирамидальная схема, выполняющая функции мультиплексора "32—1" и построенная на мультиплексорах меньшей размерности, показана на рис. 2.9 (сокращение MUX от англ. *multiplexer*).

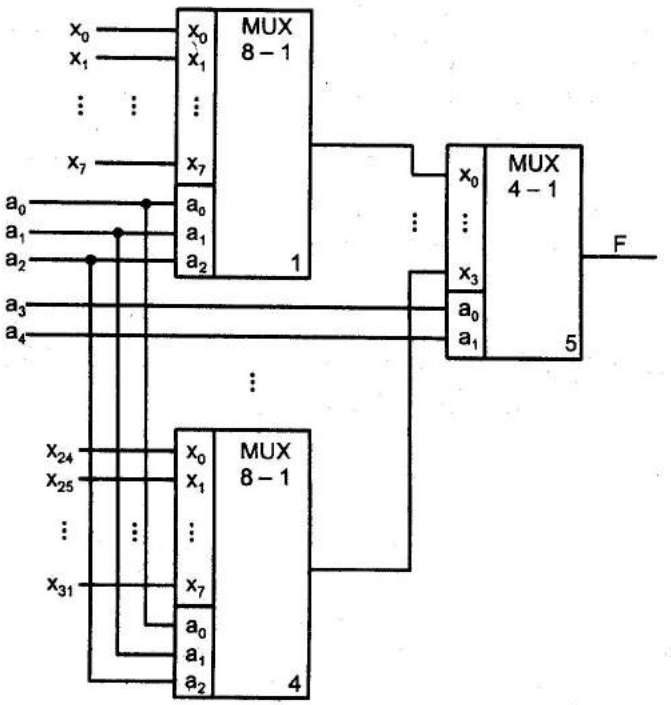


Рис. 2.9. Схема наращивания мультиплексоров

Демультиплексоры выполняют операцию, обратную операции мультиплексоров — передают данные из одного входного канала в один из нескольких каналов-приемников. Многоразрядные демультиплексоры составляются из нескольких одноразрядных. Условное обозначение демультиплексоров на примере размерности "1–4" показано на рис. 2.10.

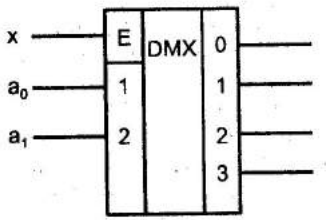


Рис. 2.10. Условное обозначение дешифратора-демультиплексора

Нетрудно заметить, что дешифратор со входом разрешения работы будет работать в режиме демультиплексора, если на вход разрешения подавать информационный сигнал. Действительно, при единичном значении этого сигнала адресация дешифратора (подача адресного кода на его входы) приведет к возбуждению соответствующего выхода, при нулевом — нет. А это и соответствует передаче информационного сигнала в адресованный выходной канал.

В связи с указанным в сериях элементов отдельные демультиплексоры могут отсутствовать, а *дешифратор со входом разрешения часто называется дешифратором-демультиплексором.*

§ 2.5. Универсальные логические модули на основе мультиплексоров

Универсальные логические модули (УЛМ) на основе мультиплексоров относятся к устройствам, настраиваемым на решение той или иной задачи. Универсальность их состоит в том, что для заданного числа аргументов можно настроить УЛМ на любую функцию. Известно, что общее число функций n аргументов выражается как 2^{2^n}. С ростом n число функций растет чрезвычайно быстро. Хотя практический интерес представляют не все существующие функции, возможность получить любую из огромного числа функций свидетельствует о больших перспективах применения УЛМ.

Первый способ настройки УЛМ

Первым способом настройки, используемым в УЛМ, является фиксация некоторых входов. Для этого способа справедливо следующее соотношение между числом аргументов и числом настроечных входов. Пусть число аргументов n и требуется настройка на любую из функций. Тогда число комбинаций для кода настройки, равное числу функций, есть 2^{2^n}. Для двоичного кода число комбинаций связано с разрядностью кода выражением 2^m, где m — разрядность кода. Приравнивая число воспроизводимых функций к числу комбинаций кода настройки, имеем для числа настроечных входов соотношение $m = 2^n$.

Полученному выражению отвечает соотношение между числом входов разного типа для мультиплексора. При этом на адресные входы следует подавать аргументы функции, а на информационные входы — сигналы настройки (рис. 2.11). Таким образом, *для использования мультиплексора в качестве УЛМ следует изменить назначение его входов.*

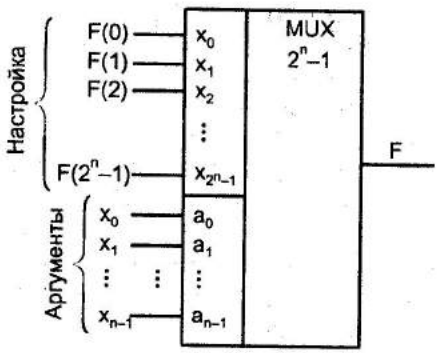


Рис. 2.11. Схема использования мультиплексора в качестве УЛМ

Рис. 2.11 иллюстрирует возможность воспроизведения с помощью мультиплексора любой функции n аргументов. Действительно, каждому набору аргументов соответствует передача на выход одного из сигналов настройки. Если этот сигнал есть значение функции на данном наборе аргументов, то задача решена. Разным функциям будут соответствовать разные коды настройки. Алфавитом настройки будет {0, 1} — настройка осуществляется константами 0 и 1. На рис. 2.12, *а* показан пример воспроизведения функции неравнозначности $x_1 \oplus x_2$ с помощью мультиплексора "4—1".

Большое число настроечных входов затрудняет реализацию УЛМ. Для УЛМ, расположенных внутри кристалла, можно вводить код настройки последовательно в сдвигающий регистр, к разрядам которого подключены входы на-

стройки. Тогда внешним входом настройки будет всего один, но настройка будет занимать не один такт, а 2^n тактов. Возможны и промежуточные последовательно-параллельные варианты ввода кода настройки.

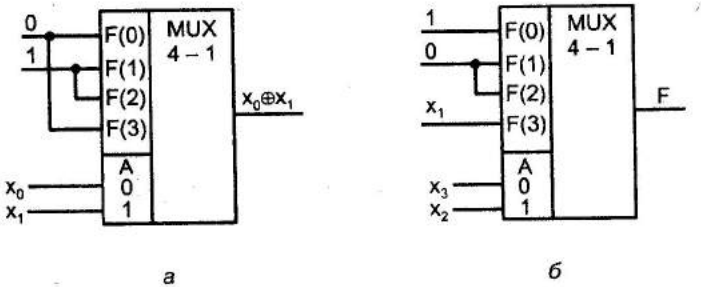


Рис. 2.12. Примеры воспроизведения функций при настройке константами (*а*) и при переносе одного аргумента в число сигналов настройки (*б*)

Второй способ настройки УЛМ

Большое число входов настройки наталкивает на поиск возможностей их уменьшения. Такие возможности существуют и заключаются в расширении алфавита настроечных сигналов. Если от алфавита {0, 1} перейти к алфавиту $\{0,1, \tilde{x}_i\}$, где $\tilde{x}_i$ — литерал одного из аргументов, то число входов аргументов сократится на единицу, а число настроечных входов — вдвое. Напомним, что под литералом переменной понимается либо сама переменная, либо ее инверсия. Перенос одного из аргументов в число сигналов настройки не влечет за собою каких-либо схемных изменений. На том же оборудовании будут реализованы функции с числом аргументов на единицу больше, чем при настройке константами.

Для нового алфавита код настройки находится следующим образом. Аргументы за исключением $\tilde{x}_i$ подаются на адресующие входы, что соответствует их фиксации в выражении для искомой функции, которая становится функцией единственного аргумента $\tilde{x}_i$. Эту функцию, которую назовем остаточной, и нужно подавать на настроечные входы.

Если искомая функция зависит от n аргументов и в число сигналов настройки будет перенесен один из аргументов, то возникает n вариантов решения задачи, т. к. в сигналы настройки может быть перенесен любой аргумент. Спрашивается, какой именно аргумент целесообразно переносить в сигналы настройки? Здесь можно опираться на рекомендацию: в настроечные сигналы следует переводить аргумент, который имеет минимальное число вхождений в термы функции. В этом случае будут максимально использованы как бы внутренние логические ресурсы мультиплексора, а среди

сигналов настройки увеличится число констант, что и считается благоприятным для схемной реализации УЛМ.

Проиллюстрируем сказанное примером воспроизведения функции трех аргументов $F = x_1x_2x_3 \vee \bar{x}_2\bar{x}_3$. Минимальное число вхождений в выражение функции имеет переменная x_1, которую и перенесем в число сигналов настройки. Остаточная функция определится табл. 2.3, *а*.

Таблица 2.3

x_2	x_3	$F_{ост}$
0	0	1
0	1	0
1	0	0
1	1	x_1

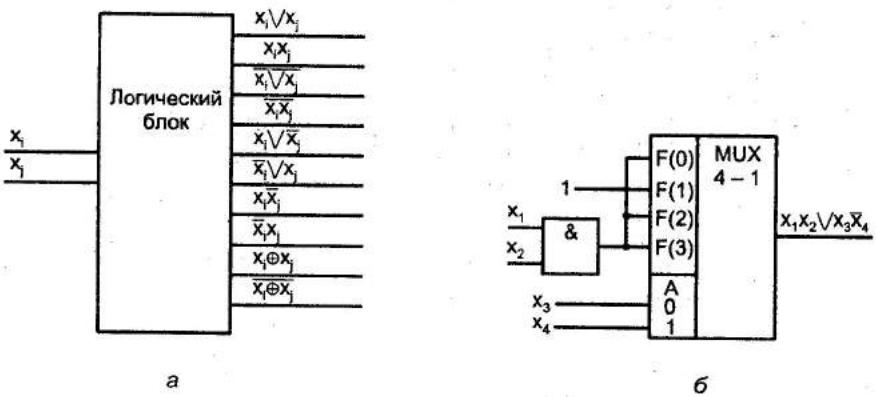
а

x_4	x_3	$F_{ост}$
0	0	$x_1 x_2$
0	1	1
1	0	$x_1 x_2$
1	1	$x_1 x_2$

б

Схема УЛМ приведена на рис. 2.12, *б*.

По пути расширения алфавита сигналов настройки можно идти и дальше, но при этом понадобятся дополнительные логические схемы, воспроизводящие остаточные функции, которые будут уже зависеть более чем от одного аргумента.



а *б*

Рис. 2.13. Логический блок выработки сигналов настройки УЛМ с переносом двух аргументов в сигналы настройки (*а*) и пример схемы воспроизведения функции четырех аргументов на мультиплексоре "4—1" (*б*)

Если в сигналы настройки перевести два аргумента, то дополнительные логические схемы будут двухвходовыми вентилями, что мало усложняет УЛМ и может оказаться приемлемым решением. В этом случае для сохранения универсальности УЛМ мультиплексору нужно предпослать блок выработки

остаточных функций, в котором формируются все функции двух переменных (за исключением констант 0 и 1 и литералов самих переменных, которые не требуется вырабатывать). Такой блок показан на рис. 2.13, *а*. Пример реализации функции $F = x_1x_2 \vee x_3\bar{x}_4$ при алфавите настройки $\{0,1, \bar{x}_1, \bar{x}_2\}$ показан на рис. 2.13, *б*. Таблица остаточной функции для этого примера приведена в табл. 2.3, *б*.

Пирамидальные структуры УЛМ

Дальнейшее расширение алфавита настройки за счет переноса трех и более переменных в сигналы настройки требует вычислений остаточных функций трех или более переменных.

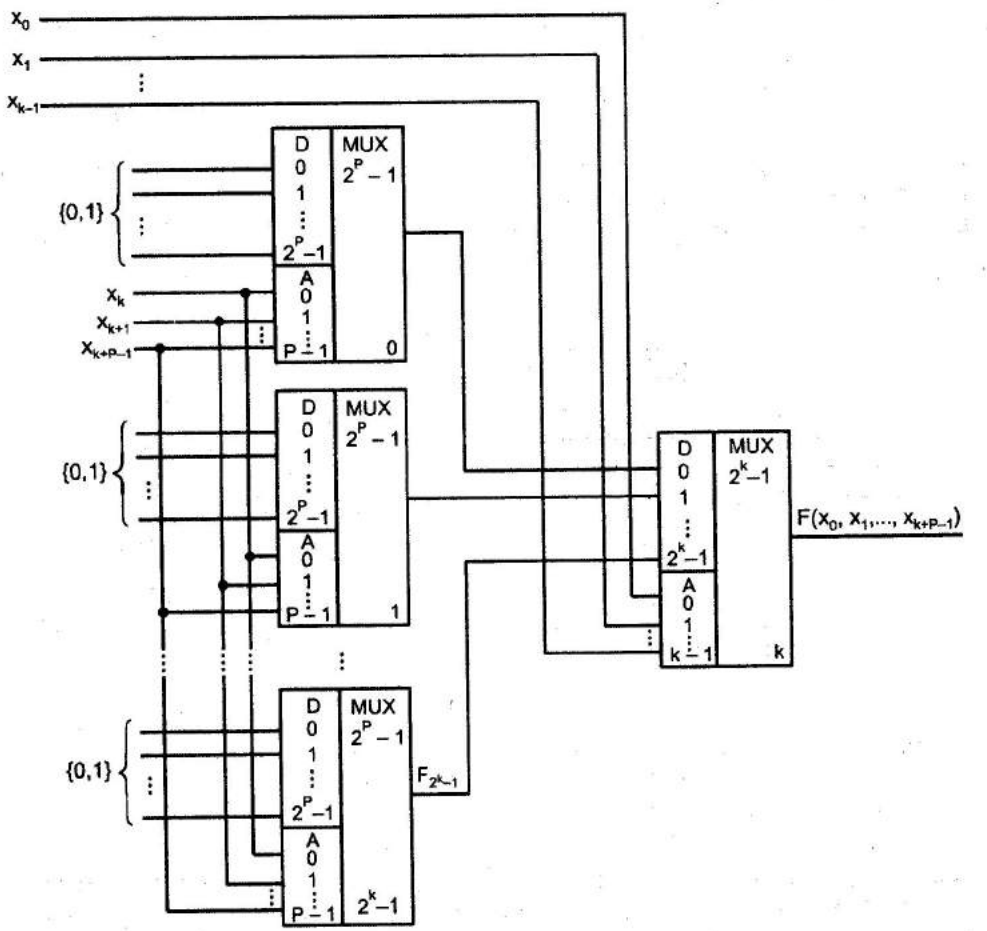


Рис. 2.14. Структура УЛМ, построенного на нескольких мультиплексорах

Вычисление таких остаточных функций с помощью мультиплексоров приводит к пирамидальной структуре (рис. 2.14), в которой мультиплексоры первого яруса реализуют остаточные функции, а мультиплексор второго яруса вырабатывает искомую функцию.

Показанная пирамидальная структура — каноническое решение, которое приводит к нужному результату, но не претендует на оптимальность. Дело в том, что варианты построения схем из нескольких мультиплексоров для воспроизведения функций многих переменных разнообразны, но алгоритм поиска оптимальной по затратам оборудования или какому-либо другому критерию отсутствует. Имеются работы, в которых найдены решения более высокого качества, но это результаты изобретений, касающиеся частных случаев и не относящиеся к регулярному методу поиска структур.

При чисто электронной настройке константами 0 и 1 схема воспроизводит функцию n аргументов, где $n = k + p$, причем k — число аргументов, подаваемых на мультиплексор второго яруса, p — число аргументов, от которых зависят остаточные функции, воспроизводимые мультиплексорами $0...2k - 1$ первого яруса.

Для уменьшения аппаратных затрат в схеме следует стремиться к минимизации числа мультиплексоров в столбце, т. е. минимизации k и, соответственно, максимальным p, поскольку их сумма $k + p$ постоянна и равна n.

Сигналы настройки для мультиплексоров первого яруса можно искать разными способами:

1. Подстановкой (фиксацией) наборов аргументов, подаваемых на адресные входы мультиплексоров для получения остаточных функций и, далее, сигналов настройки. Этот способ уже рассмотрен (см. табл. 2.3).

2. С помощью разложения функции по Шеннону. Это разложение можно произвести по разному числу переменных. По одному из аргументов разложение имеет вид:

$$F = (x_0, x_1,...,x_{n-1}) = \bar{x}_0 F(0, x_1,...,x_{n-1}) \vee x_0 F(1, x_1,...,x_{n-1}).$$

Справедливость такого разложения видна из подстановки в него значений $x_0 = 0$ и $x_0 = 1$, что дает непосредственно функции $F(0, x_1,..., x_{n-1})$ и $F(1, x_1,..., x_{n-1})$.

Разложение функции по двум аргументам

$$F = (x_0, x_1,...,x_{n-1}) = \bar{x}_0\bar{x}_1 F(0, 0, x_2,...,x_{n-1}) \vee \bar{x}_0 x_1 F(0, 1, x_2,...,x_{n-1}) \vee$$
$$\vee x_0\bar{x}_1 F(1, 0, x_2,...,x_{n-1}) \vee x_0 x_1 F(1, 1, x_2,...,x_{n-1})$$

и, наконец, разложение по k аргументам

$$F = (x_0, x_1,...,x_{n-1}) = \bar{x}_0\bar{x}_1...\bar{x}_{k-2}\bar{x}_{k-1}F(0, 0,..., 0, x_k,...,x_{n-1}) \vee$$
$$\vee \bar{x}_0\bar{x}_1...\bar{x}_{k-2}x_{k-1}F(0, 0,..., 0, 1, x_k,...,x_{n-1}) \vee ...$$
$$...\vee x_0x_1...x_{k-2}x_{k-1}F(1, 1,..., 1, x_k,...,x_{n-1}) =$$
$$= \bar{x}_0\bar{x}_1...\bar{x}_{k-2}\bar{x}_{k-1}F_0 \vee \bar{x}_0\bar{x}_1...\bar{x}_{k-2}x_{k-1}F_1 \vee ... \vee x_0x_1...x_{k-2}x_{k-1}F_{2^k-1},$$

где

$F_0 = F(0, 0,..., 0, x_k,..., x_{n-1})$,

$F_1 = F(0, 0,..., 0, 1, x_k,..., x_{n-1})$,

.....

$F_{2^k-1} = F(1, 1,..., 1, x_k,..., x_{n-1})$.

Структура формул разложения полностью соответствует реализации двухъярусных УЛМ. В первом ярусе реализуются функции F_i, ($i = 0, ..., 2^k - 1$), зависящие от $n - k$ аргументов, которые используются как настроечные для второго яруса, мультиплексор которого воспроизводит функцию k аргументов.

3. Сигналы настройки можно получить непосредственно из таблицы истинности функции. Для удобства просмотра таблицы ее следует записать так, чтобы аргументы, переносимые в сигналы настройки, играли роль младших разрядов в словах-наборах аргументов. Пусть имеется функция четырех переменных $x_3x_2x_1x_0$, и переменная x_3 считается старшим разрядом вектора аргументов. Пусть далее функция задана перечислением наборов аргументов, на которых она принимает единичные значения, причем заданы десятичные значения этих наборов: 3, 4, 5, 6, 7, 11, 15. Заметим, что аналитическое значение этой функции имеет вид $F = x_0x_1 \vee x_2\bar{x}_3$. Значения функции сведены в табл. 2.4.

Таблица 2.4

x_3	x_2	x_1	x_0	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1

Таблица 2.4 (окончание)

x_3	x_2	x_1	x_0	F
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

При электронной настройке УЛМ константами 0 и 1 требуется мультиплексор размерности "16—1", на настроечные входы УЛМ подаются значения самой функции из таблицы.

При переносе $\bar{x}_0$ в сигналы настройки (алфавит настройки $\{0, 1, \bar{x}_0\}$) требуется найти остаточную функцию, аргументами которой является вектор переменных $x_3x_2x_1$. Каждая комбинация этих переменных встречается в двух смежных строках таблицы. Просматривая таблицу по смежным парам строк, можно видеть, что остаточная функция соответствует другой таблице (табл. 2.5).

Таблица 2.5

x_3	x_2	x_1	$F_{ост}$
0	0	0	0
0	0	1	x_0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	x_0
1	1	0	0
1	1	1	x_0

Для реализации этого варианта УЛМ достаточен мультиплексор "8—1", но для перестройки на другую функцию потребуется не только смена кода настройки, но и коммутация входов настройки для подачи литералов переменной на другие настроечные входы.

При переносе в сигналы настройки двух переменных ($\bar{x}_0$ и $\bar{x}_1$) для поиска остаточных функций следует просмотреть четверки смежных строк таблицы с неизменными наборами x_2x_3 — аргументами, подаваемыми на адресные входы УЛМ. Этот просмотр приводит к следующей таблице (табл. 2.6).

Таблица 2.6

x_3	x_2	$F_{ост}$
0	0	x_1x_0
0	1	1
1	0	x_1x_0
1	1	x_1x_0

Из таблицы видно, что для воспроизведения функции достаточно использовать мультиплексор "4—1" с дополнительным конъюнктором для получения произведения x_1x_0. Но при перестройке на другую функцию потребуются и другие функции двух переменных, т. е. универсальный логический модуль должен включать в свой состав дополнительный логический блок (см. рис. 2.13, *а*).

Логические блоки на мультиплексорах используются в современных СБИС программируемой логики, выпускаемых ведущими мировыми фирмами. Эти блоки работают по изложенным выше принципам, однако зачастую универсальность в смысле воспроизводимости всех без исключения функций данного числа аргументов не преследуется, что упрощает схемы блоков, оставляя им в то же время достаточно широкие логические возможности.

В данном случае модули относятся к настраиваемым и характеризуются порождающей функцией, реализуемой модулем, когда все его входы используются как информационные (т. е. для подачи на них аргументов). Эта функция при введении настройки, когда часть входов занята под настроечные сигналы, порождает некоторый список подфункций, зависящих от меньшего числа аргументов в сравнении с порождающей функцией. Создается перечень практически важных подфункций для того или иного настраиваемого модуля.

На рис. 2.15, *а* показан логический блок, используемый в СБИС программируемой логики фирмы Actel (США). Изображены обозначения фирмы для мультиплексоров "2—1" (адресующие входы расположены сбоку). При $S = 0$ на выход передается сигнал верхнего входа, при $S = 1$ — нижнего. Функциональная характеристика (порождающая функция) для этого блока имеет вид $F = \overline{S_0} \vee S_1(\overline{S_A}A_0 \vee S_A A_1) \vee (S_0 \vee S_1)(\overline{S_B}B_0 \vee S_B B_1)$.

Варьируя подачу на входы блока констант и входных переменных, можно реализовать 702 практически полезные переключательные функции.

На рис. 2.15, *б* показан логический блок (вернее его комбинационная часть) фирмы Quicklogic (США) с более широкими логическими возможностями.

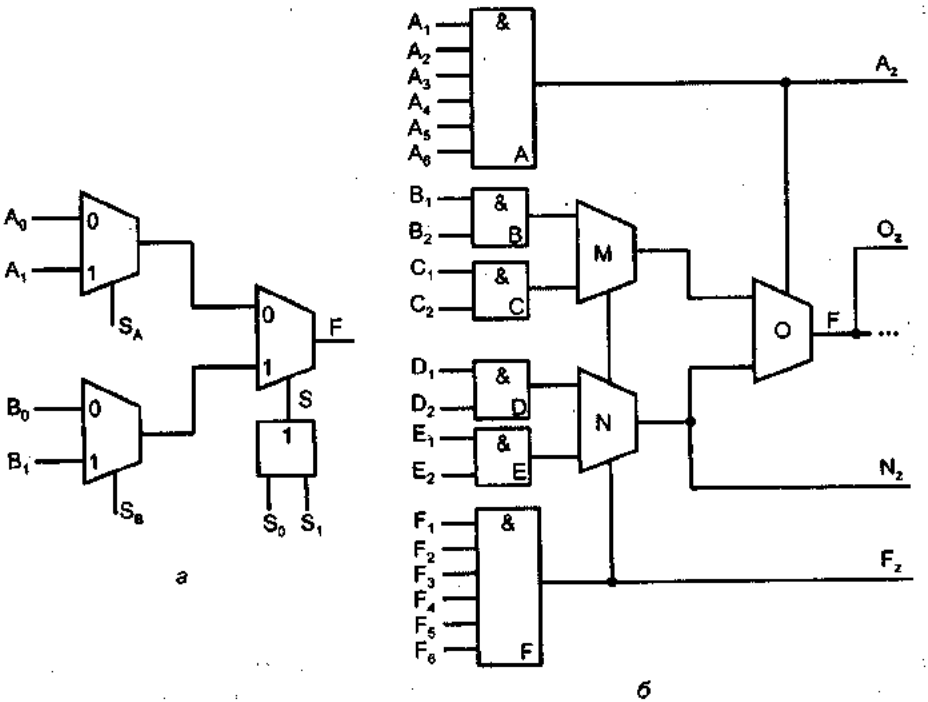


Рис. 2.15. Мультиплексорные логические блоки, используемые в микросхемах фирм Actel (*а*) и Quicklogic (*б*)

§ 2.6. Компараторы

Компараторы (устройства сравнения) определяют отношения между двумя словами. Основными отношениями, через которые можно выразить остальные, можно считать два — "равно" и "больше".

Определим функции, вырабатываемые компараторами, следующим образом: они принимают единичное значение (истинны), если соблюдается условие, указанное в индексе обозначения функции. Например, функция $F_{A=B} = 1$, если $A = B$ и принимает нулевое значение при $A \neq B$.

Приняв в качестве основных отношения "равно" и "больше", для остальных можно записать:

$$F_{A\neq B} = \overline{F}_{A=B};\ F_{A<B} = F_{B>A};\ F_{A\geq B} = \overline{F}_{B>A};\ F_{A\leq B} = \overline{F}_{A>B}.$$

Эти отношения используются как логические условия в микропрограммах, в устройствах контроля и диагностики ЭВМ и т. д.

В сериях цифровых элементов обычно имеются компараторы с тремя выходами: "равно", "больше" и "меньше" (рис. 2.16). Для краткости записей в индексе выходных функций указывается только слово А.

Рис. 2.16. Условное обозначение компаратора с тремя выходами

Устройства сравнения на равенство строятся на основе поразрядных операций над одноименными разрядами обоих слов. Слова равны, если равны все одноименные их разряды, т. е. если в обоих нули или единицы. Признак равенства разрядов

$$r_i = a_i b_i \vee \bar{a}_i \bar{b}_i = \overline{a_i \bar{b}_i \vee \bar{a}_i b_i} = \overline{\overline{a_i \bar{b}_i} \cdot \overline{\bar{a}_i b_i}} = \overline{a_i \oplus b_i}.$$

Признак неравенства разрядов

$$\bar{r}_i = a_i\bar{b}_i \vee \bar{a}_i b_i = \overline{a_i\bar{b}_i \vee \bar{a}_i b_i} = \overline{\overline{a_i\bar{b}_i}\cdot\overline{\bar{a}_i b_i}} = a_i \oplus b_i.$$

Признак равенства слов $R = r_{n-1}\, r_{n-2}\ldots r_0$.

Схема компаратора на равенство в базисе И-НЕ показана на рис. 2.17, *а*.

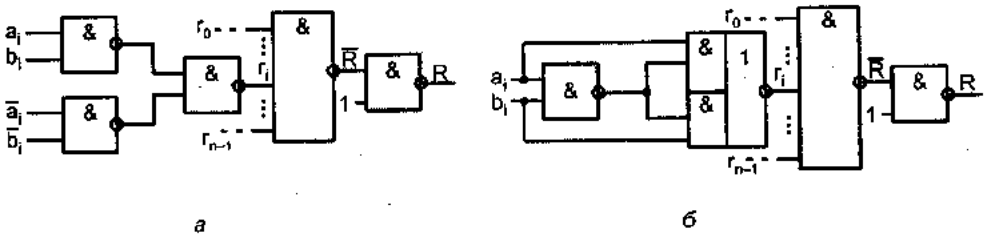


Рис. 2.17. Схемы компараторов на равенство

Схема без парафазных входов (рис. 2.17, *б*) основана на выражениях для $\bar{r}_i$, преобразованных следующим образом:

$$\bar{r}_i = \overline{a_i\bar{b}_i \vee \bar{a}_i b_i} = \overline{a_i(\bar{a}_i \vee \bar{b}_i) \vee b_i(\bar{a}_i \vee \bar{b}_i)} = \overline{a_i a_i b_i \vee b_i a_i b_i}.$$

Построение компаратора на "больше" для одноразрядных слов (табл. 2.7) требует реализации функции $F_{A>B} = a\bar{b}$.

Таблица 2.7

a	b	$F_{A>B}$
0	0	0
0	1	0
1	0	1
1	1	0

Функцию $F_{A>B}$ для многоразрядных слов проще всего получить на основе рассуждений. Пусть нужно сравнить двухразрядные слова. Если старшие разряды a_1 и b_1 не равны, то результат известен независимо от младших разрядов: при $a_1 = 1$ и $b_1 = 0$ имеем $A > B$, а при $a_1 = 0$ и $b_1 = 1$ имеем $A < B$. Если же $a_1 = b_1$, результат еще неизвестен, и требуется анализ следующего разряда по тому же алгоритму. Поэтому для двухразрядных слов можно записать $F_{A>B} = a_1\bar{b}_1 \vee r_1 a_0\bar{b}_0$.

Подобный же подход справедлив и для слов любой разрядности — к анализу следующего разряда нужно переходить только при равенстве предыдущих. Таким образом, для общего случая n-разрядных слов имеем

$$F_{A>B} = a_{n-1}\bar{b}_{n-1} \vee r_{n-1}a_{n-2}\bar{b}_{n-2} \vee \ldots \vee r_{n-1}r_{n-2}\ldots r_1 a_0\bar{b}_0.$$

Замечание

Правильно рассуждая, мы получили правильный результат. Однако цель минимизации формул при этом не ставилась и на самом деле выражения для $F_{A>B}$ не минимальны. В минимальном варианте признаки равенства r_i можно заме-

нить более простыми функциями $d_j = a_j \vee \overline{b_j}$. Однако для построения компаратора с тремя выходами ("равно", "больше" и "меньше") полученный нами вариант остается предпочтительным, поскольку функции r_i все равно нужны для сравнения на "равно", и для операций сравнения на "больше" они могут быть взяты в готовом виде.

Пример реализации компаратора с тремя выходами для двухразрядных слов приведен на рис. 2.18.

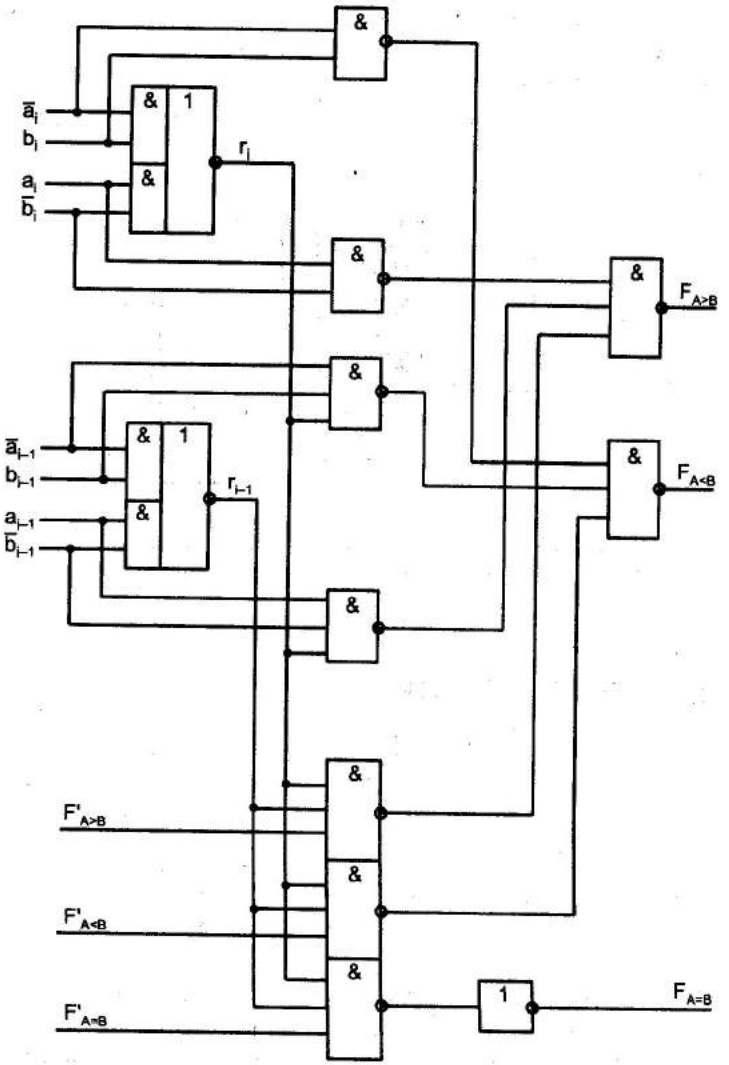


Рис. 2.18. Пример построения компаратора

Выработка признака A > B в этой схеме производится по соотношению (штрихом отмечены функции с выходов младшей группы)

$$F_{A>B} = a_i\bar{b}_i \vee r_i a_{i-1}\bar{b}_{i-1} \vee F'_{A>B} r_i r_{i-1} = \overline{\overline{a_i\bar{b}_i} \cdot \overline{a_{i-1}\bar{b}_{i-1}r_i} \cdot \overline{F'_{A>B} r_i r_{i-1}}}.$$

Компараторы для слов большой разрядности получают наращиванием размерности путем использования нескольких ИС компараторов, принцип наращивания соответствует показанному на рис. 2.18.

Примером гибкости цифровой схемотехники может служить возможность построения компаратора на равенство с помощью последовательного соединения демультиплексора и мультиплексора, как показано на рис. 2.19. В работе такого компаратора легко разобраться, помня как функционируют входящие в него блоки.

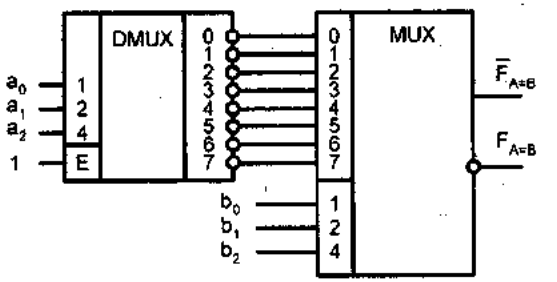


Рис. 2.19. Реализация компаратора на равенство с помощью демультиплексора и мультиплексора

§ 2.7. Схемы контроля

Сложность ЭВМ и других ЦУ определяет важность операций контроля и диагностики их функционирования. В некоторых случаях контроль жизненно важен (авиационные приборы, управление мощными энергетическими установками, мониторинг пациентов в клиниках и др.).

Причинами нарушения нормальной работы ЦУ могут быть отказы (т. е. нарушения из-за возникших неисправностей, имеющих постоянный характер) и сбои (т. е. нарушения из-за проявлений неблагоприятных факторов, в частности, помех, которые в дальнейшем могут и не проявиться). Независимо от этого дальше будем говорить об ошибках функционирования, поскольку для рассматриваемых далее вопросов конкретный характер ошибок несущественен.

Цели и задачи контроля, диагностики и исправления ошибок в ЦУ могут быть разными.

Можно ставить *задачу предотвращения ошибок* в работе ЦУ. Для этого необходимы такие меры, как применение высококачественных элементов схем, стабилизация условий окружающей среды и т. п. Но даже при всех стараниях вряд ли возможно полностью избавиться от ошибок.

Имея в виду неизбежность возникновения ошибок, следует позаботиться об их выявлении. *Задачи выявления ошибок* решаются разными методами. Можно, например, воспользоваться дублированием ЦУ и сравнением результатов работы двух идентичных устройств. Несовпадение результатов в этом случае рассматривается как признак ошибки (хотя вероятность того, что ошибка появилась в контролируемом устройстве, а не в контролирующем равна всего 50%). Для выявления ошибок используются специальные коды, более сложные, чем двоичные.

И, наконец, можно ставить *задачи маскирования (исправления)* ошибок. В этом случае наличие ошибок определенного типа и количества не нарушает работу устройства, поскольку их влияние устраняется автоматически. В этой области используется, например, троекратное резервирование устройств с выработкой результата путем "голосования" с помощью мажоритарных элементов. Эти элементы вырабатывают выходные данные "по большинству" входных. Если из трех устройств одно стало работать неправильно, это не скажется на результате. Только ошибка в двух из трех каналов проявляется в результате.

Отметим, что добавление к функциям устройств функций контроля всегда связано с избыточностью — платой за новые возможности будут дополнительные аппаратные или временные затраты.

Вводимая избыточность — это цена контроля. В частности, метод дублирования ценен своей универсальностью, но дорог, для него избыточность составляет около 100%.

В этом параграфе рассмотрены очень ограниченные вопросы контроля ЦУ, которому посвящаются специальные труды. Здесь затронуты темы, связанные с пониманием работы ИС, выпускаемых для использования в системах контроля. К таким схемам относятся мажоритарные элементы, схемы контроля по модулю 2 и схемы кодирования-декодирования для кодов Хемминга.

Мажоритарные элементы

Задача мажоритарного элемента — произвести "голосование" и передать на выход величину, соответствующую большинству из входных. Ясно, что мажоритарный элемент может иметь только нечетное число входов. Практически

выпускаемые элементы имеют по три входа или по пять входов. Функционирование мажоритарного элемента, на входы которого поступают величины F_1, F_2, и F_3 и по результатам голосования вырабатывается выходная величина F, представлено в табл. 2.8. Если имеется в виду контроль многоразрядных слов, то в каждом разряде ставится элемент рассматриваемого типа.

Таблица 2.8

F_1	F_2	F_3	F	a_1	a_0
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	1	0	1
1	0	0	0	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	1	0	0

Кроме выхода F, в таблице даны и выходы a_1, a_0 — старший и младший разряды двухразрядного кода, указывающего номер отказавшего канала (рис. 2.20).

Из таблицы легко получить функции, которые после несложных преобразований приводятся к следующим:

$$F = F_1F_2 \vee F_1F_3 \vee F_2F_3,\ a_1 = F_2 \oplus F_3,\ a_0 = F_1 \oplus F_3.$$

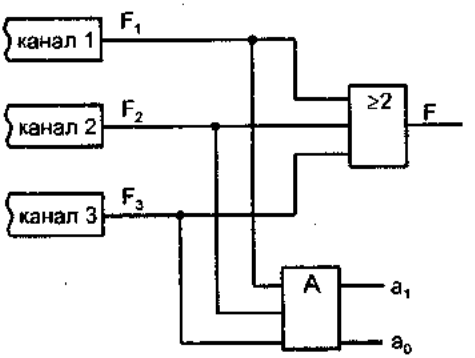


Рис. 2.20. Схема голосования с мажоритарным элементом

В схемах типа рис. 2.20 от мажоритарного элемента требуется особенно высокая надежность, т. к. его отказ делает бесполезной всю схему резервирования.

Контроль по модулю 2

Контроль правильности передач и хранения данных — важное условие нормальной работы ЦУ. В этой области простейшим и широко применяемым методом является контроль по модулю 2. Приступая к ознакомлению

с этим методом, следует остановиться на некоторых понятиях из теории построения помехоустойчивых кодов. *Кодовая комбинация* — набор из символов принятого алфавита. *Код* — совокупность кодовых комбинаций, используемых для отображения информации. *Кодовое расстояние* между двумя кодовыми комбинациями — число разрядов, в которых эти комбинации отличаются друг от друга. *Минимальное кодовое расстояние* — минимальное кодовое расстояние для любой пары комбинаций, входящих в данный код. *Кратностью ошибки* называют число ошибок в данном слове (число неверных разрядов).

Из теории кодирования известны условия обнаружения и исправления ошибок при использовании кодов:

$$d_{min} = r_{обн} + 1;\ d_{min} = 2r_{испр} + 1;\ d_{min} = 2r_{испр} + r_{обн} + 1,$$

где d_{min} — минимальное кодовое расстояние кода; $r_{обн}$ и $r_{испр}$ — кратность обнаруживаемых и исправляемых ошибок соответственно.

Существует также понятие *вес комбинации*, обозначающее число единиц в данной комбинации.

Для двоичного кода минимальное кодовое расстояние $d_{min} = 1$, поэтому он не обладает возможностями какого-либо контроля производимых над ним действий. Чтобы получить возможность обнаруживать хотя бы ошибки единичной кратности, нужно увеличить минимальное кодовое расстояние на 1. Это и сделано для кода контроля по модулю 2 (контроля по четности/нечетности).

При этом способе контроля *каждое слово дополняется контрольным разрядом, значение которого подбирается так, чтобы сделать четным (нечетным) вес каждой кодовой комбинации.* При одиночной ошибке в кодовой комбинации четность (нечетность) ее веса меняется, а такая комбинация не принадлежит к данному коду, что и обнаруживается схемами контроля. При двойной ошибке четность (нечетность) комбинации не нарушается — такая ошибка не обнаруживается. Легко видеть, что у кода с контрольным разрядом $d_{min} = 2$. Хотя обнаруживаются ошибки не только единичной, но вообще нечетной кратности, на величину d_{min} это не влияет.

При контроле по четности вес кодовых комбинаций делают четным, при контроле по нечетности — нечетным. Логические возможности обоих вариантов абсолютно идентичны. В зависимости от технической реализации каналов передачи данных, может проявиться предпочтительность того или иного варианта, поскольку один из вариантов может позволить отличать обрыв всех линий связи от передачи нулевого слова, а другой — нет.

Значения контрольного разряда ρ при контроле по четности ($\rho_ч$) и нечетности ($\rho_н$) приведены для четырехразрядного информационного слова в табл. 2.9.

Таблица 2.9

a_3	a_2	a_1	a_0	$\rho_ч$	$\rho_н$
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	0	1
0	1	1	0	0	1
.					.
.					.
.....					
1	1	1	1	0	1

Как видно из таблицы, $\rho_ч = a_3 \oplus a_2 \oplus a_1 \oplus a_0$; $\rho_н = \overline{a_3 \oplus a_2 \oplus a_1 \oplus a_0}$.

После передачи слова или считывания его из памяти вновь производится сложение разрядов кодовой комбинации по модулю 2 (свертка по модулю 2) и проверяется, сохранилась ли четность (нечетность) веса принятой комбинации. Если четность (нечетность) веса комбинации изменилась, фиксируется ошибка операции.

Из приведенного материала следует, что контроль по модулю 2 эффективен там, где вероятность единичной ошибки много больше, чем вероятность двойной (или вообще групповой).

В частности, для полупроводниковой основной памяти компьютеров такая ситуация справедлива, т. к. каждый бит слова хранится в своей собственной ячейке, и наиболее вероятны единичные ошибки. А для памяти на магнитных носителях информации (диски, ленты) дефекты таковы, что обычно затрагивают площадь, на которой размещено несколько бит данных, поэтому для этой памяти контроль по модулю 2 неэффективен.

Схемы свертки

Контроль по модулю 2 реализуется с помощью схем свертки. Типична многоярусная схема свертки пирамидального типа. На рис. 2.21, *а* показана схема свертки байта. Для оценки аппаратной сложности и быстродействия подобных схем при разрядности свертываемого слова 2^n (n — произвольное целое число) легко получить соотношения:

$$N_{лэ} = n/2 + n/4 + ... + n/n = n(1/2 + 1/4 + ... + 1/n) = n - 1;\ L = \log_2 n,$$

где $N_{лэ}$ — число логических элементов в схеме; L — ее логическая глубина.

Схемотехника сейчас сориентирована главным образом на работу с параллельными данными, однако не исключены ситуации обработки последовательных данных, когда слова передаются по одной линии последовательно разряд за разрядом. Для таких случаев целесообразно применять схему свертки (рис. 2.21, *б*), которая выдает результат всего лишь через одну задержку после поступления последнего разряда a_7.

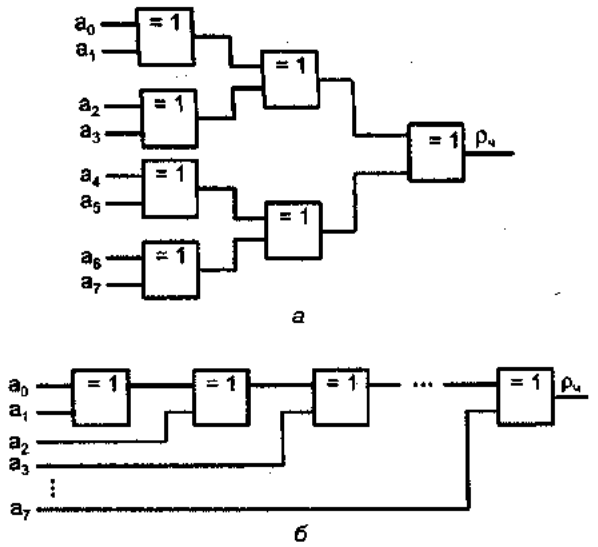


Рис. 2.21. Схемы свертки пирамидального (*а*) и последовательного (*б*) типов

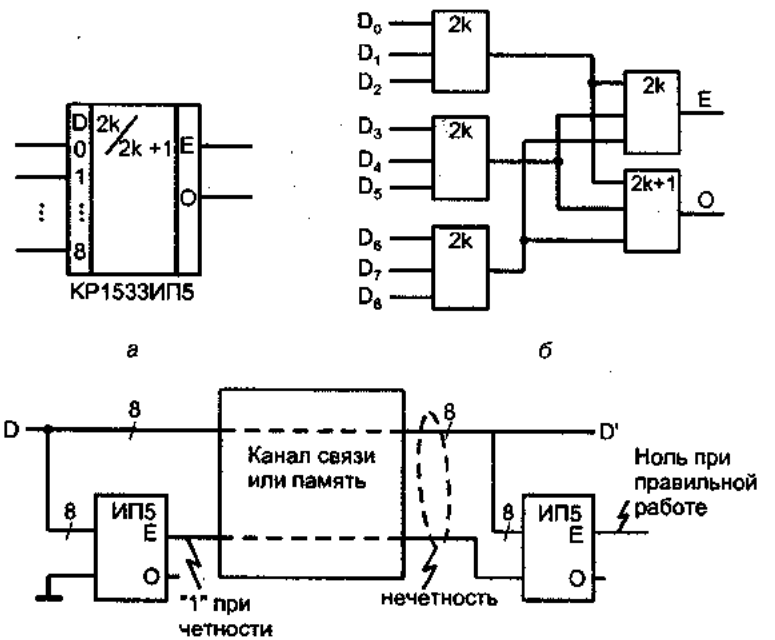


Рис. 2.22. Микросхема ИП5 (*а*, *б*) и ее применение в схеме контроля (*в*)

Примером ИС свертки по модулю 2 может служить микросхема ИП5 серии КР1533 (рис. 2.22, *а*). Схема имеет 9 входов, что допускает свертку байта с девятым контрольным разрядом. Двумя выходами схемы являются E (Even) и O (Odd). Если вес входной комбинации четный, то E = 1 и O = 0, и наоборот, если вес нечетный.

Схемотехнически ИС КР1533ИП5 представляет собою пирамидальную структуру из трехвходовых элементов типа четность/нечетность (рис. 2.22, *б*).

Передача данных с контролем по модулю 2

Передача данных или их запись/считывание (если речь идет о памяти) с контролем показаны на рис. 2.22, *в*. Входные данные обозначены через D, на выходе из канала связи или памяти данные обозначены через D', поскольку вследствие ошибок они могут измениться.

Контроль по модулю 2 применим не только для операций передачи и хранения слов, но и для некоторых более сложных операций. В этих случаях недостаточно просто добавить к информационному слову контрольный разряд, а требуются более развитые операции.

Контроль логического преобразователя

На рис. 2.23 показан пример контроля логического преобразователя ЛП, воспроизводящего систему переключательных функций от m переменных. Для осуществления контроля к системе добавляется еще одна функция $F_{доп} = F_1 \oplus F_2 \oplus ... \oplus F_n$, которая воспроизводится на индивидуальных элементах (во избежание маскирования контролируемых ошибок). Затем выработанные функции $F_1 — F_n$ свертываются по модулю 2, и результат сравнивается с дополнительной функцией $F_{доп}$. Ясно, что при отсутствии ошибок должны сравниваться одинаковые величины. Если они различны, то на выходе элемента сложения по модулю 2 возникнет сигнал ошибки.

Ряд операций контролируется в условиях, когда контрольный разряд не постоянен, а изменяется по определенному закону. Это возможно, если установлена закономерность изменения контрольного разряда при выполнении операции. Например, при работе счетчика его содержимое меняется по известному закону. Если к слову, содержащемуся в счетчике, добавлять контрольный разряд, также изменяющийся по известному закону, то свертка содержимого счетчика вместе с контрольным разрядом покажет единичную ошибку в работе счетчика. Такой же подход возможен для контроля сумматоров.

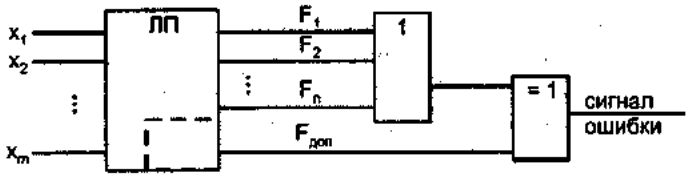


Рис. 2.23. Схема контроля
логического преобразователя по модулю 2

Контроль с использованием кодов Хемминга

Применение кодов Хемминга позволяет *исправлять единичные ошибки.* Добавление к коду Хемминга контрольного разряда, обеспечивающего четность/нечетность всей кодовой комбинации в целом, приводит к модифицированному коду Хемминга, с помощью которого можно исправлять единичные ошибки и обнаруживать двойные.

Методы контроля с помощью кодов Хемминга основаны на тех же идеях, что и контроль по модулю 2. Отсюда и область эффективного применения кодов Хемминга — устройства, в которых вероятность единичных ошибок много больше, чем вероятность групповых.

Для получения кодовой комбинации кода Хемминга *к информационному слову добавляется несколько контрольных разрядов.* Для простоты просмотра кодовых комбинаций с целью определения значений контрольных разрядов примем, что контрольные разряды занимают позиции с номерами 2^i (i = 0, 1, 2, ...).

Каждый контрольный разряд ассоциируется с некоторой группой разрядов кодовой комбинации и выводит вес группы, в которую он входит, на четность/нечетность.

Первый контрольный разряд входит в группу разрядов с номерами xx...xx1, где x означает произвольное значение. Иными словами, в первую группу входят разряды с нечетными номерами: 1, 3, 5, 7, 9, ...

Второй контрольный разряд входит в группу разрядов с номерами, имеющими единицу во втором справа разряде, т. е. номерами xx...x1x. Это номера 2, 3, 6, 7, 10, 11, ...

Третий контрольный разряд входит в группу, у которой номера разрядов имеют единицу в третьем справа разряде: xx...1xx, т. е. с номерами 4, 5, 6, 7, 12, 13, 14, 15, ...

Контрольные разряды выводят веса своих групп на четность/нечетность. Далее для определенности примем, что ведется контроль по четности. После

выполнения операции (например, считывания кодовой комбинации из памяти) производится столько проверок по модулю 2, сколько контрольных разрядов в кодовой комбинации, т. е. проверяется сохранение четности весов групп. Если в кодовой комбинации произошла ошибка, то в одних проверках она скажется, а в других — нет. Это и позволяет определить разряд, в котором произошла ошибка. Для восстановления правильного значения слова теперь остается только проинвертировать ошибочный разряд. Такова идея построения и использования кода Хемминга.

Пример составления кода Хемминга для четырехразрядного информационного слова $A = a_3a_2a_1a_0$ приведен в табл. 2.10. Через ρ в таблице обозначен общий контрольный разряд для всей кодовой комбинации, через ρ_1, ρ_2, ρ_3 — первый, второй и третий групповые контрольные разряды.

Таблица 2.10

8	7	6	5	4	3	2	1
ρ	a_3	a_2	a_1	ρ_3	a_0	ρ_2	ρ_1
0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1
0	0	0	1	1	0	0	1
0	0	0	1	1	1	1	0
1	0	1	0	1	0	1	0
0	0	1	0	1	1	0	1
1	0	1	1	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	1	0	1	0
.....	.....	.....	.....	.....	.....	.....	.....
1	1	1	1	1	1	1	1

$a_3a_2a_1a_0 = 0110$

Для коротких слов избыточность кода Хемминга получилась значительной (здесь на четыре информационных разряда приходится четыре контрольных), но это нетипично, поскольку реально контролируются слова большей разрядности, для которых избыточность (относительная) быстро уменьшается с ростом разрядности слов. Короткое слово взято, чтобы пример не был громоздким.

Рассмотрим теперь процесс исправления и выявления ошибок. Пусть, например, передавалось информационное слово $0110 = 6_{10}$. Не учитывая пока разряд ρ, получим, что правильная кодовая комбинация имеет вид:

7	6	5	4	3	2	1
0	1	1	0	0	1	1

Пусть во втором слева разряде произошла ошибка и принята комбинация:

7	6	5	4	3	2	1
0	0	1	0	0	1	1

Первая проверка (по группе разрядов с нечетными номерами) показывает сохранение четности, т. е. в этой группе ошибок нет, результат этой проверки отмечается нулем.

Вторая проверка (по разрядам 2, 3, 6, 7) обнаруживает нарушение четности веса комбинации, ее результат отмечается единицей.

Третья проверка (по разрядам 4, 5, 6, 7) также обнаруживает нарушение четности, ее результат отмечается единицей.

Результаты проверок образуют слово, называемое синдромом. Синдром указывает номер разряда, в котором произошла ошибка. Во взятом примере результаты проверок дают слово $110 = 6_{10}$. Проинвертировав разряд номер 6, возвращаемся к правильной кодовой комбинации — ошибка исправлена.

Минимальное кодовое расстояние обычного кода Хемминга равно трем. Добавление разряда проверки общей четности веса комбинации приводит к *модифицированному коду Хемминга* с минимальным кодовым расстоянием, равным 4 и, соответственно, добавляет возможность обнаружения двойной ошибки. Обнаружение двойной ошибки основано на сопоставлении наличия или отсутствия признаков ошибки в синдроме и общей четности. Если обозначить через S любое ненулевое значение синдрома, то возможные ситуации, используемые для обнаружения двойной ошибки, окажутся следующими (табл. 2.11).

Таблица 2.11

Синдром	Свертка кодовой комбинации	Характеристика результата
0	0	Все правильно, слово можно использовать
S	1	Была единичная ошибка, исправлена, слово можно использовать
S	0	Эти ситуации могут возникать только вследствие ошибок двойной или большей кратности, слово использовать нельзя
0	1	

Схемы кодера и декодера для кода Хемминга

На рис. 2.24 показана схема кодирования и декодирования для кодов Хемминга. Верхняя часть схемы показывает выработку контрольных разрядов

для составления кода Хемминга. Нижняя часть содержит три четырехразрядных схемы свертки для проведения групповых проверок (разрядов синдрома). Синдром поступает на дешифратор, который вырабатывает единичный сигнал на линии, соответствующей номеру ошибочного разряда. Эта единица выполняет инвертирование ошибочного разряда слова A, поступая на второй вход элемента сложения по модулю 2, через который данный разряд передается на выход схемы. Таким образом, нижняя часть схемы представляет собой декодирующее устройство для кода Хемминга.

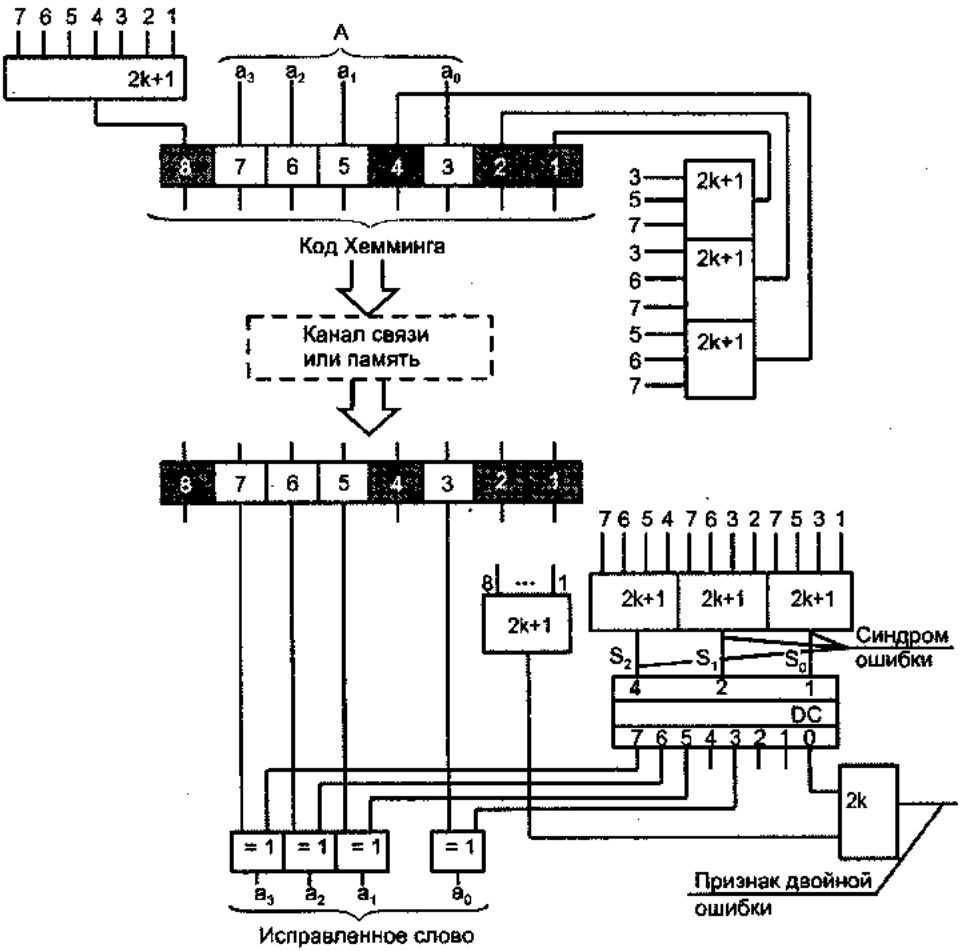


Рис. 2.24. Схема кодирования и декодирования для кодов Хемминга

Двойная ошибка обнаруживается элементом 2k согласно логике ситуаций, указанной ранее.

Кодирование-декодирование для 16-разрядных слов с формированием шести контрольных разрядов модифицированного кода Хемминга реализуется микросхемой ВЖ1 серий К555, 533. Время кодирования-декодирования составляет для этой ИС 50—60 нс.

Код Хемминга относится к числу простых. Есть много более сложных кодов с большими корректирующими возможностями (БЧХ, код Файра, код Рида—Соломона и др.).

§ 2.8. Сумматоры

Сумматоры выполняют арифметическое (в противоположность логическому) сложение и вычитание чисел. Имеют самостоятельное значение и являются также ядром схем арифметико-логических устройств (АЛУ), реализующих ряд разнообразных операций и являющихся непременной частью всех процессоров.

Аппаратная сложность и быстродействие сумматора являются очень важными параметрами и поэтому разработано множество вариантов сумматоров, которые имеют разветвленную классификацию. Выделяя главные варианты, остановимся на следующих типах сумматоров:

- одноразрядный сумматор;
- сумматор для последовательных операндов;
- сумматор для параллельных операндов с последовательным переносом;
- сумматор для параллельных операндов с параллельным переносом;
- сумматор с последовательным распространением переноса по цепочке замкнутых ключей;
- сумматор групповой структуры с цепным переносом;
- сумматор групповой структуры с параллельным межгрупповым переносом;
- сумматор с условным переносом;
- накапливающий сумматор.

Наряду с сумматорами могут быть реализованы вычитатели, однако это почти никогда не делается, поскольку вычитание выполняется посредством сложения с применением дополнительных либо обратных кодов.

Одноразрядный сумматор

Одноразрядный сумматор имеет три входа (два слагаемых и перенос из предыдущего разряда) и два выхода (суммы и переноса в следующий разряд).

Таблица истинности одноразрядного сумматора имеет вид, представленный в табл. 2.12.

Таблица 2.12

a_i	b_i	c_{i-1}	S_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Аналитические выражения функций суммы и переноса (сигнал переноса обозначен через С — от англ. *Carry*) имеют вид

$$S_i = \bar{a}_i\bar{b}_ic_{i-1} \vee \bar{a}_ib_i\bar{c}_{i-1} \vee a_i\bar{b}_i\bar{c}_{i-1} \vee a_ib_ic_{i-1},\ C_i = a_ib_i \vee a_ic_{i-1} \vee b_ic_{i-1}.$$

В базисе Шеффера функции S_i и C_i выражаются следующим образом:

$$S_i = \overline{\bar{a}_i\bar{b}_ic_{i-1} \cdot \bar{a}_ib_i\bar{c}_{i-1} \cdot a_i\bar{b}_i\bar{c}_{i-1} \cdot a_ib_ic_{i-1}},$$

$$C_i = \overline{a_ib_i \cdot a_ic_{i-1} \cdot b_ic_{i-1}}.$$

Непосредственное воспроизведение полученных формул на элементах двухступенчатой логики И-ИЛИ-НЕ приводит к применению элемента 2-2-2И-ИЛИ-НЕ для выработки сигнала переноса $\bar{C}_i$ и элемента 3-3-3-3И-ИЛИ-НЕ для сигнала суммы $\bar{S}_i$. Такое решение используется в некоторых сериях микросхем, но более популярно решение, приводящее к некоторому сокращению аппаратной сложности схемы при сохранении минимальной задержки по цепи переноса. Идея этого решения состоит в использовании полученного уже значения $\bar{C}_i$ в качестве вспомогательного аргумента при вычислении $\bar{S}_i$.

Из табл. 2.12 видно, что во всех ее строчках, кроме первой и последней, $S_i = \bar{C}_i$. Чтобы сделать формулу справедливой также в первой и последней строчках, нужно убрать единицу в строчке нулевых входных величин и добавить единицу в строчку единичных входных величин, что приводит к соотношению

$$S_i = \bar{C}_i(a_i \vee b_i \vee c_{i-1}) \vee a_ib_ic_{i-1}.$$

Схема сумматора, построенного по этому соотношению, показана на рис. 2.25, *а*.

Из табл. 2.12 видно, что и функция суммы, и функция переноса обладают свойством самодвойственности: при инвертировании всех аргументов инвертируется и значение функции, т. е.

$$S(\overline{X}) = \overline{S}(X),\ C(\overline{X}) = \overline{C}(X).$$

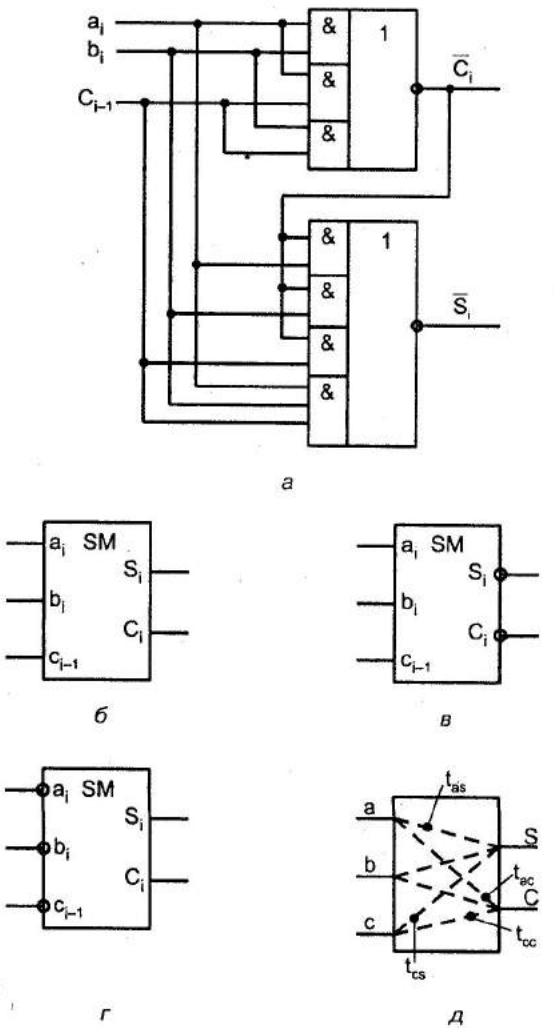


Рис. 2.25. Схема (*а*), условные обозначения (*б*, *в*, *г*) и пути распространения сигналов одноразрядного сумматора (*д*)

Условное обозначение одноразрядного сумматора показано на рис. 2.25, *б*. Для варианта с выработкой инвертированных значений суммы и переноса

на основании свойства самодвойственности можно пользоваться двумя вариантами обозначений для одной и той же схемы (рис. 2.25, *в*, *г*).

Быстродействие одноразрядного сумматора оценивается задержками по шести трактам распространения сигналов: от первого слагаемого до выхода суммы, от первого слагаемого до выхода переноса, от второго слагаемого до тех же выходов и от входа переноса до выхода переноса, от входа переноса до выхода суммы (рис. 2.25, *д*). Так как тракты от обоих слагаемых обычно одинаковы, то остаются четыре задержки, отмеченные надписями t_{as}, t_{ac}, t_{cc} и t_{cs} на рис. 2.25, *д*.

На рис. 2.26 показана схема сумматора, входящая в библиотеку схемных решений семейства программируемых СБИС фирмы Altera.

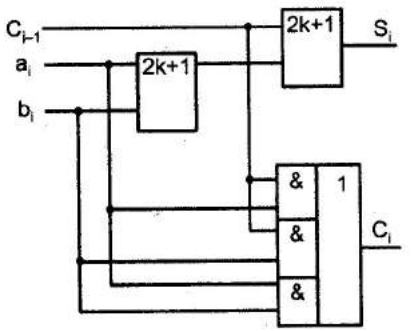


Рис. 2.26. Схема одноразрядного сумматора из библиотеки схемных решений для СБИС фирмы Altera

Последовательный сумматор

Сумматор для последовательных операндов содержит всего один одноразрядный сумматор, обрабатывающий поочередно разряд за разрядом, начиная с младшего. Сложив младшие разряды, одноразрядный сумматор вырабатывает сумму для младшего разряда результата и перенос, который запоминается на один такт.

В следующем такте складываются вновь поступившие разряды слагаемых a_1 и b_1 с переносом из младшего разряда и т. д. Схема сумматора последовательных операндов (рис. 2.27, *а*), помимо сумматора, содержит сдвигающие регистры слагаемых и суммы, а также триггер запоминания переноса. Регистры и триггер тактируются синхроимпульсами СИ.

На рис. 2.27, *б* показана временная диаграмма, соответствующая операции сложения двух операндов 101 + 110 = 1011 или в десятичном выражении 5 + 6 = 11.

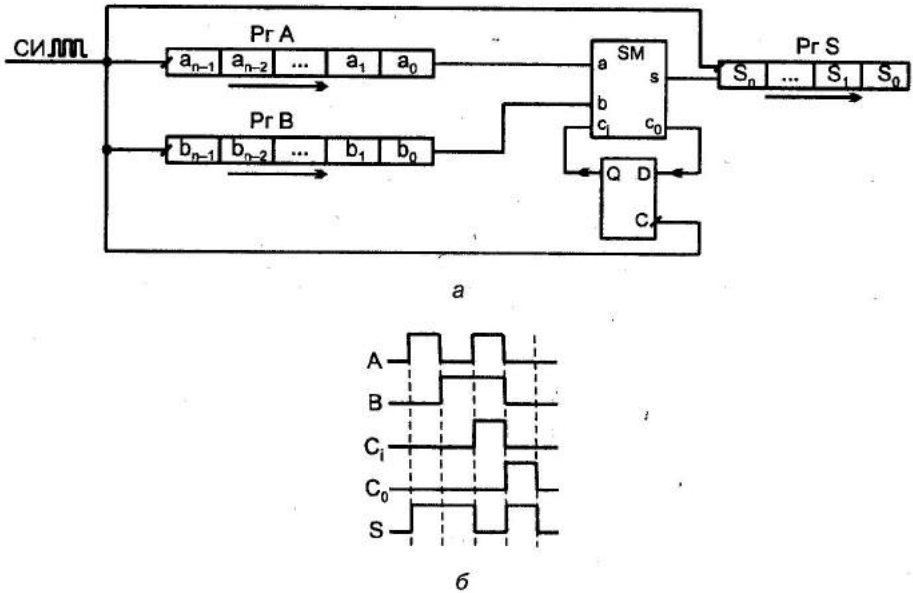


Рис. 2.27. Схема сумматора для последовательных операндов (*а*) и ее временная диаграмма (*б*)

Параллельный сумматор с последовательным переносом

Сумматор для параллельных операндов с последовательным переносом строится как цепочка одноразрядных, соединенных последовательно по цепям переноса. Для схемы с одноразрядными сумматорами, вырабатывающими инверсии суммы и переноса, такая цепочка имеет вид, приведенный на рис. 2.28, поскольку функции суммы и переноса самодвойственны. Там, где в разряд сумматора должны подаваться инверсные аргументы, в их линиях имеются инверторы, а там, где вырабатывается инверсная сумма, инвертор включен в выходную цепь. Важно, что инверторы не входят в цепь передачи переноса — они при этом не замедляют работу сумматора в целом.

Длительность суммирования для этой схемы в наихудшем случае распространения переноса по всей цепочке разрядов составит

$$t_{SM} = t_{ac} + (n - 2)t_{cc} + t_{cs},$$

где n — разрядность сумматора.

Как и в других схемах с последовательным распространением сигналов от разряда к разряду, здесь время суммирования практически пропорционально разрядности сумматора.

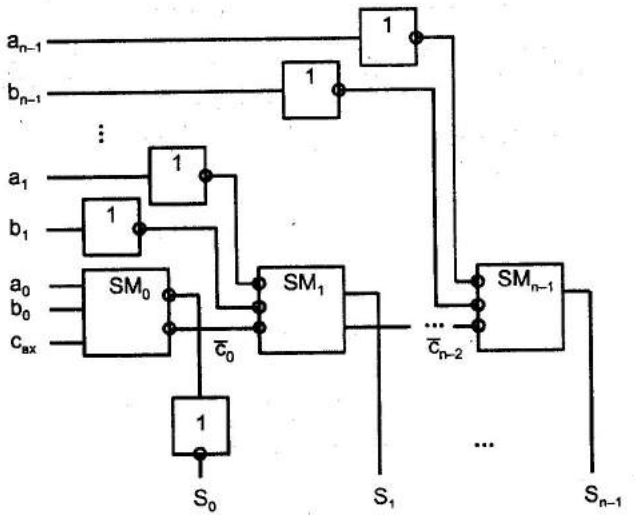


Рис. 2.28. Схема сумматора с последовательным переносом

Если одноразрядные сумматоры выполнены по схеме (см. рис. 2.25, *а*), то время суммирования для многоразрядного сумматора составит

$$t_{SM} = (n + 1)t_{ЛР},$$

где $t_{ЛР}$ — задержка элемента И-ИЛИ-НЕ, обозначенная индексом ЛР, поскольку именно эти буквы входят в маркировку элементов данного типа. Если одноразрядные сумматоры выполнены по схеме (см. рис. 2.26), то $t_{SM} = nt_{ЛР}$

Параллельный сумматор с параллельным переносом

Сумматоры для параллельных операндов с параллельным переносом разработаны для получения максимального быстродействия.

Подход к решению этой задачи требует пояснений. Дело в том, что рассматриваемые сумматоры — комбинационные схемы, и вырабатываемые ими функции могут быть представлены в нормальных формах, например, в ДНФ, что приводит к двухъярусной реализации при наличии парафазных аргументов и к трехъярусной при однофазных аргументах. Таким образом, предельное быстродействие оценивается (2—3) элементарными задержками. Однако реальные схемы таких пределов не достигают, т. к. построение сумматоров многоразрядных слов на основе нормальных форм дало бы неприемлемо громоздкие схемы. Реальные схемы имеют модульную структуру,

т. е. состоят из подсхем (разрядных схем), что резко упрощает их, но не дает предельно возможного быстродействия.

Сумматоры с параллельным переносом не имеют последовательного распространения переноса вдоль разрядной сетки. *Во всех разрядах результаты вырабатываются одновременно*, параллельно во времени. Сигналы переноса для данного разряда формируются специальными схемами, на входы которых поступают все переменные, необходимые для выработки переноса, т. е. те, от которых зависит его наличие или отсутствие. Ясно, что это внешний входной перенос $C_{вх}$ (если он есть) и значения всех разрядов слагаемых, младших относительно данного. Одноразрядные сумматоры, имеющиеся в разрядных схемах, здесь упрощены, т. к. от них выход переноса не требуется, достаточно одного выхода суммы (рис. 2.29). Обозначение CR происходит от слова carry (перенос).

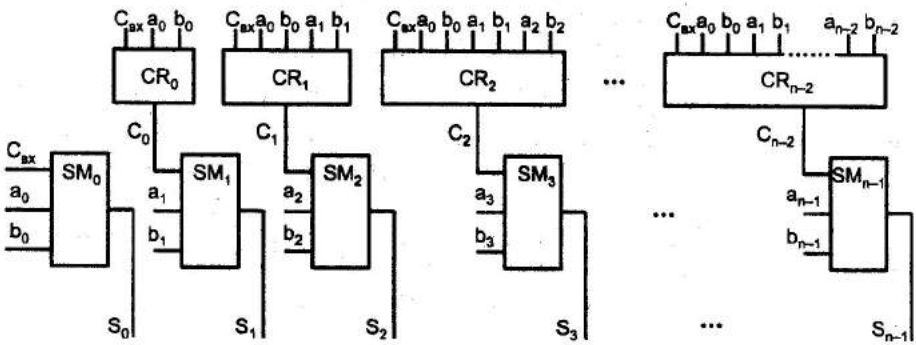


Рис. 2.29. Структура сумматора с параллельным переносом

Для перехода от идеи построения схемы к ее конкретному виду удобно ввести две вспомогательные функции: генерации и прозрачности.

Функция генерации принимает единичное значение, если перенос на выходе данного разряда появляется независимо от наличия или отсутствия входного переноса. Очевидно, что эта функция $g_i = a_i b_i$.

Функция прозрачности (транзита) принимает единичное значение, если перенос на выходе данного разряда появляется только при наличии входного переноса. Эта функция $h_i = a_i \vee b_i$. Строго говоря, $h_i = a_i\bar{b}_i \vee \bar{a}_i b_i$, но т. к. при $a_i = b_i = 1$, т. е. в ситуации, где между функциями ИЛИ и "исключающее ИЛИ" проявляется разница, перенос все равно формируется из-за $g_i = 1$, допустимо заменить функцию прозрачности на дизъюнкцию.

Теперь выражение для сигнала переноса можно записать в виде $C_i = g_i \vee h_i C_{i-1}$.

На основе полученного выражения выведем функции переноса С для нулевого, первого и второго разрядов с последующим их обобщением.

Перенос на выходе младшего разряда $C_0 = g_0 \vee C_{вх} h_0$, согласно чему он либо генерируется самим разрядом ($g_0 = 1$), либо пропускается через него ($h_0 = 1$ и $C_{вх} = 1$).

Аналогичным образом для переноса C_1 на выходе следующего разряда справедливо соотношение $C_1 = g_1 \vee C_0 h_1$.

Подставив в это соотношение выражение для C_0, получим

$$C_1 = g_1 \vee g_0 h_1 \vee C_{вх} h_1 h_0.$$

Для следующего разряда произведем те же действия

$$C_2 = g_2 \vee C_1 h_2 = g_2 \vee g_1 h_2 \vee g_0 h_2 h_1 \vee C_{вх} h_2 h_1 h_0.$$

Выведенные формулы имеют ясный физический смысл — перенос на выходе разряда сгенерируется в нем или придет от предыдущих разрядов при прозрачности тех, через которые он распространяется.

Для произвольного разряда с номером i можно записать

$$C_i = g_i \vee g_{i-1} h_i \vee g_{i-2} h_i h_{i-1} \vee \ldots \vee g_0 h_i h_{i-1} \ldots h_1 \vee C_{вх} h_i h_{i-1} \ldots h_0.$$

Функции переноса имеют нормальную дизъюнктивную форму и могут быть реализованы элементами И-ИЛИ (либо И-ИЛИ-НЕ, для $\overline{C}$, если это свойственно данной схемотехнике). Однако у этих элементов недостаточное число входов по И, требуемое для построения многоразрядного сумматора. Поэтому предпочтительна схема на элементах И-НЕ (у стандартных элементов имеется до восьми входов по И). Перевод полученных выражений в базис И-НЕ дает выражения

$$C_0 = \overline{\overline{g_0} \cdot \overline{C_{вх} h_0}} = \overline{\overline{a_0 b_0} \cdot \overline{C_{вх} h_0}};$$

$$C_1 = \overline{\overline{a_1 b_1} \cdot \overline{a_0 b_0 h_1} \cdot \overline{C_{вх} h_1 h_0}};$$

$$C_2 = \overline{\overline{a_2 b_2} \cdot \overline{a_1 b_1 h_2} \cdot \overline{a_0 b_0 h_2 h_1} \cdot \overline{C_{вх} h_2 h_1 h_0}}.$$

Схема сумматора (рис. 2.30) соответствует полученным выражениям.

Исходя из схемы, можно видеть, что время суммирования складывается из времени формирования функций прозрачности (одна задержка элемента И-НЕ, которую обозначим $t_{ЛА}$), времени формирования функций переноса ($2t_{ЛА}$) и задержки упрощенных одноразрядных сумматоров ($t_{ЛР}$), что в результате дает $t_{SM} = (4...5)\, t_{ЛА}$.

Длительность суммирования, полученная из рассмотрения логической схемы сумматора, не зависит от его разрядности, что является характерным признаком структур с параллельными переносами вообще, и не только сумматоров. Однако фактически это не совсем так, поскольку с ростом разрядности сумматора увеличивается нагрузка элементов схемы, что увеличивает их задержки *(см. § 1.1)*. В частности, коэффициент разветвления элементов,

вырабатывающих функции прозрачности, равен $n^2/4$, т. е. квадратично зависит от разрядности сумматора. Поэтому рост разрядности замедляет процесс суммирования.

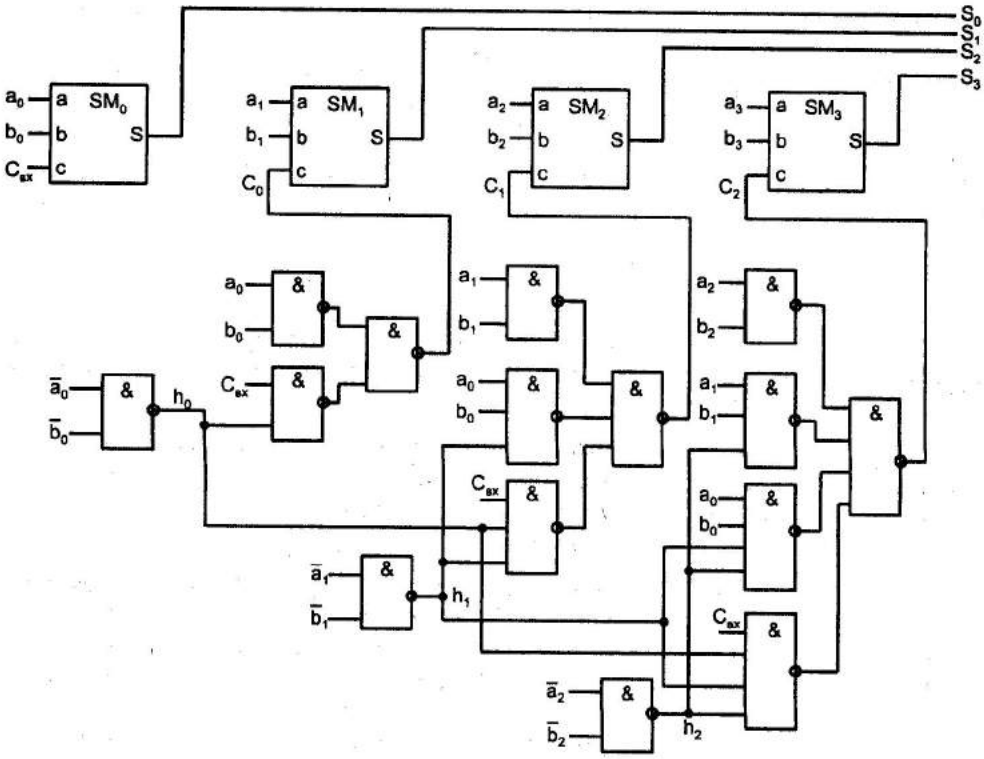


Рис. 2.30. Вариант схемы сумматора с параллельным переносом

Диапазон разрядностей, в которых проявляются достоинства сумматоров с параллельным переносом, невелик. До n = 3...4 преимущества имеют более простые схемы сумматоров с последовательным переносом, после n = 8 появляются перегруженные элементы и элементы с большим числом входов, что замедляет работу сумматора, требует введения развязывающих элементов с их задержками и т. п.

Сумматор с передачей сигнала переноса по цепочке замкнутых ключей

Среди компромиссных вариантов сумматоров, занимающих промежуточное положение между сумматорами с последовательным и параллельным пере-

носами, имеется интересный вариант, предложенный в работе [62], который в первом приближении оценивается как обладающий простотой сумматора с последовательным переносом при быстродействии, близком к быстродействию сумматора с параллельным переносом.

Предложенный подход предусматривает разделение задачи вычисления сигналов переноса на два этапа. На первом этапе для всех разрядов (параллельно во времени) задаются условия вычисления переносов, на втором происходит простая передача информации по образованной на первом этапе цепи.

При вычислении сигнала переноса C_i возможны два случая: $a_i = b_i$ или $a_i \neq b_i$. В первом случае перенос от предыдущего разряда не имеет значения (не влияет на сигнал переноса из данного разряда) и его можно исключить из рассмотрения. Действительно, при $a_i = b_i = 0$ перенос $C_i = 0$, а при $a_i = b_i = 1$ перенос $C_i = 1$. Следовательно, в качестве переноса может использоваться значение любого операнда (a_i или b_i). Примем, что $C_i = a_i$. Во втором случае сигнал переноса из данного разряда совпадает с сигналом переноса в данный разряд, т. е. $C_i = C_{i-1}$.

В обоих случаях после установления факта равенства или неравенства операндов дальнейших переключений элементов не требуется, происходит только передача информации по цепи, заготовленной на предыдущем этапе, что может происходить достаточно быстро.

Схема разряда описанного сумматора (рис. 2.31) содержит ключ, управляемый от элемента сложения по модулю 2 операндов a_i и b_i, который выявляет равенство или неравенство этих операндов. В зависимости от выходного сигнала этого элемента выходному переносу присваиваются значения a_i или C_i. Этот же элемент используется для вычисления совместно со вторым элементом сложения по модулю 2 значения суммы данного разряда по формуле: $S_i = a_i$ XOR b_i XOR C_{i-1}, где XOR (от англ. *exclusive OR*) — обозначение операции "исключающее ИЛИ" (синоним операции сложения по модулю 2).

Автор описанного метода построения сумматора проверил его работу при реализации схемы на элементах ТТЛ. Результат оказался эффективным. Например, для сумматоров с 2, 4 и 8 разрядами время сложения практически не зависело от разрядности, как это свойственно структурам с параллельным переносом. По скорости четырехразрядный сумматор выиграл у варианта с последовательным переносом в 2 раза, а восьмиразрядный сумматор в 3,5 раза. Естественно, рассматриваемая структура может быть реализована в рамках любой схемотехнологии, в том числе и на наиболее популярной схемотехнологии КМОП. Эффективность метода будет зависеть от задержки сигнала при его распространении через цепочку, состоящую из последовательного соединения замкнутых ключей, и в зависимости от параметров этих ключей будет меняться.

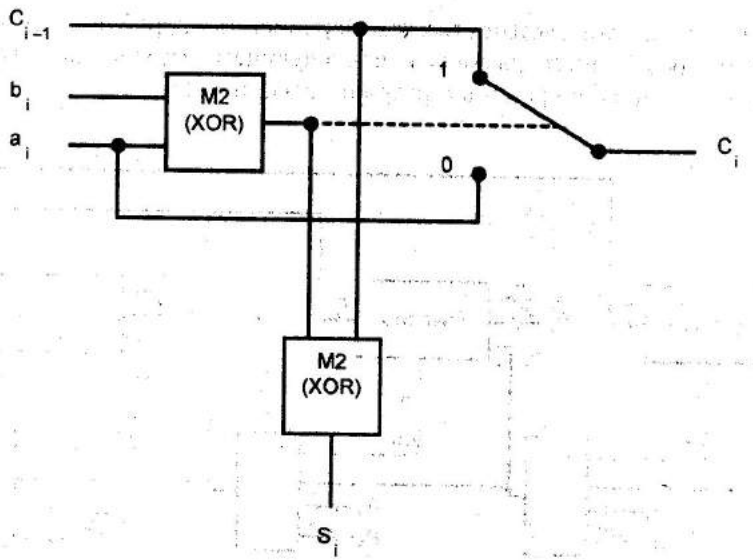


Рис. 2.31. Схема разряда сумматора с передачей сигнала переноса по цепочке замкнутых ключей

Сумматоры групповой структуры

В сумматорах групповой структуры схема с разрядностью n делится на ℓ групп по m разрядов ($n = \ell m$). В группах и между ними возможны различные виды переносов, что порождает множество вариантов групповых сумматоров. Ниже рассмотрены основные варианты: *с цепным (последовательным) и параллельным переносами между группами*. В самих группах перенос при этом может быть любым.

Групповой сумматор с цепным переносом при ℓ группах имеет $\ell - 1$ блок переноса. Блоки переноса включены последовательно и образуют тракт передачи переноса (рис. 2.32). Слагаемые разбиты на m-разрядные поля, суммируемые в группах. Результат также составляется из m-разрядных полей.

Блоки переноса $БП_i$ (i = 1...) анализируют слагаемые в пределах группы, и если из группы должен быть перенос, то он появляется на выходе блока для подачи на вход следующей группы и в цепочку распространения переноса от младших групп к старшим.

Переносы определяются формулами, полученными выше для сумматоров с параллельным переносом, но сумматоры благодаря делению на группы существенно упрощаются — у них все $БП_i$ имеют одинаковую сложность (все блоки анализируют m-разрядные операнды), тогда как в сумматоре с парал-

лельными переносами сложность схем переноса непрерывно возрастает при переходе от предыдущего разряда к последующему (последняя схема переноса требует анализа операндов с разрядностью n − 1).

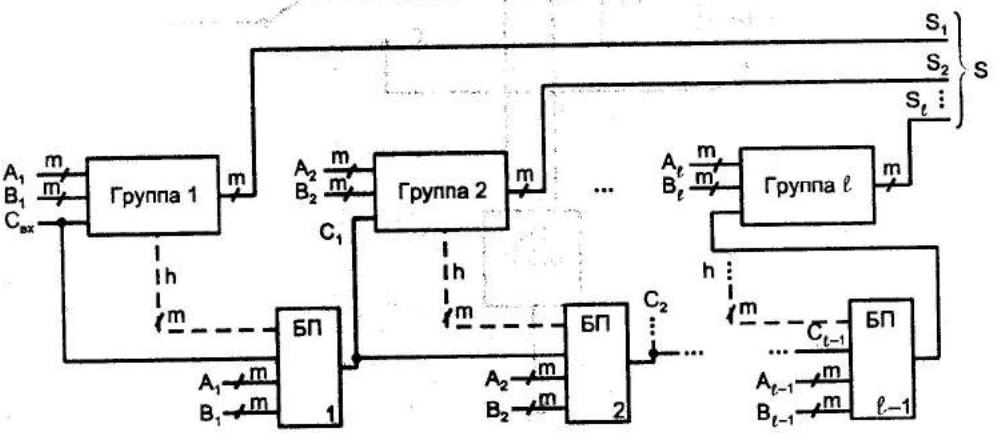


Рис. 2.32. Схема группового сумматора с цепным переносом

Максимальная длительность суммирования для варианта с цепным переносом $t_{SM} = (\ell - 1)t_{БП} + t_{гр}$.

Функции прозрачности разрядов, необходимые для блоков переносов, вырабатываются либо в этих блоках, либо уже имеются в группах, если в них организован параллельный перенос, и могут поступать из групп (штриховые линии на рис. 2.32). Имея в виду реализацию блоков переноса и групп, показанную ранее для базиса И-НЕ, формулу для времени суммирования можно представить в виде:

$$t_{SM} = t_h + (\ell - 1)2t_{ЛА} + (4...5)t_{ЛА} = (2\ell - 1)\,t_{ЛА} + (4...5)\,t_{ЛА}.$$

Для сумматора 16-разрядных слов, в частности, при его разбиении на 4 группы получим $t_{SM} = (11...12)\,t_{ЛА}$.

Сумматор с параллельными межгрупповыми переносами строится по структуре, сходной со структурой сумматора с параллельным переносом, в которой роль одноразрядных сумматоров играют группы.

Аппаратная сложность сумматоров с параллельными межгрупповыми переносами выше, чем сложность предыдущего варианта, но при больших разрядностях они дают преимущества по быстродействию.

При построении обычного сумматора с параллельными переносами каждый разряд характеризовался функциями генерации и прозрачности $g_i = a_i b_i$ и $h_i = a_i \vee b_i$. С помощью этих функций вырабатывался сигнал переноса по соотношению

$$C_i = g_i \vee g_{i-1}h_i \vee g_{i-2}h_i h_{i-1} \vee \ldots \vee g_0 h_i h_{i-1}\ldots h_1 \vee C_{вх} h_i h_{i-1}\ldots h_0.$$

В групповом сумматоре с параллельными межгрупповыми переносами роль одного "разряда" играет группа, которую также характеризуют функциями генерации и прозрачности. Обозначив эти функции большими буквами, можем записать выражение:

$$H = h_m h_{m-1} \ldots h_1,$$

согласно которому группа прозрачна при прозрачности всех ее разрядов, и

$$G_{гр} = g_m \vee g_{m-1} h_m \vee g_{m-2} h_m h_{m-1} \vee \ldots \vee g_1 h_m h_{m-1} \ldots h_2,$$

справедливость которого видна из предыдущего изложения способа построения сумматора с параллельным переносом.

Из групп собирается та же схема, что и из одноразрядных сумматоров, с параллельными межгрупповыми переносами согласно выражению для переноса на выходе группы с номером i:

$$C_i = G_i \vee G_{i-1} H_i \vee G_{i-2} H_{i-1} H_i \vee \ldots \vee G_1 H_2 \ldots H_i \vee C_{вх} H_1 H_2 \ldots H_i.$$

Схемы выработки переноса усложняются с ростом i. Структура группового сумматора с параллельными межгрупповыми переносами показана на рис. 2.33, где разрядность и число групп приняты равными 4. Функции прозрачности H_i могут вырабатываться как функции операндов или через использование функций прозрачности разрядов h, которые имеются в группах, если в них организован параллельный перенос (штриховые линии).

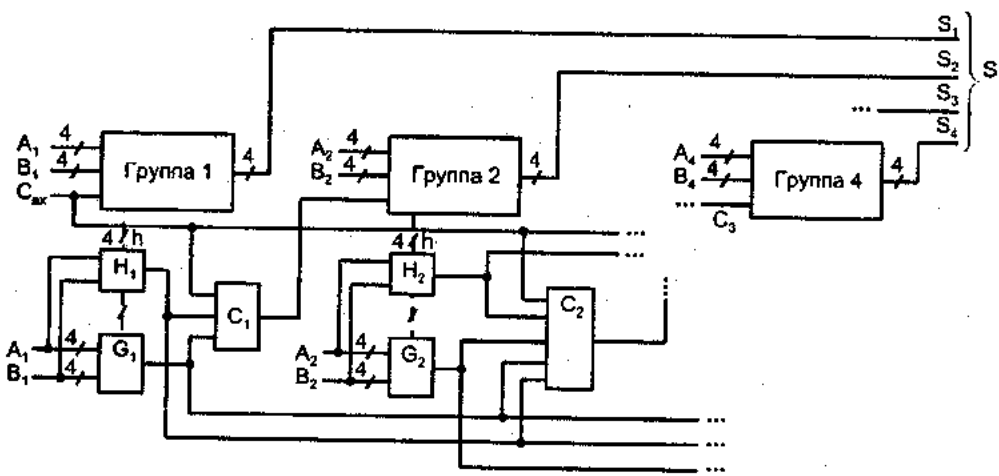


Рис. 2.33. Схема группового сумматора с параллельным переносом

Время суммирования для схемы (рис. 2.33)

$$t_{SM} = t_h + t_G + t_C + t_{гр}.$$

Для реализации, принятой нами в качестве примера, $t_{SM} \geq 10 t_{пА}$.

Так как групповой сумматор с параллельным межгрупповым переносом в отличие от негруппового можно без существенных трудностей построить и для достаточно большой разрядности, приведенная цифра является практической оценкой возможностей быстродействующих сумматоров вообще.

Если число разрядов очень велико, можно распространить способ организации параллельных переносов и на схему не с двумя, как рассмотрено, а тремя уровнями, считая групповой сумматор с двумя уровнями как бы новой группой и организуя параллельный перенос между новыми группами. Сумматоры разных типов подробно описаны в работе [33].

Сумматор с условным переносом

Сумматор с условным переносом — давно известная структура, которая со временем вышла из широкого применения, но сейчас возродилась в новейших СБИС программируемой логики. Эта структура улучшает быстродействие сумматоров с последовательным переносом. В СБИС программируемой логики, начиная с семейства микросхем FLEX 8000 фирмы Altera, была реализована цепь последовательных переносов с малыми задержками (1 нс на разряд, впоследствии задержка переноса была снижена до 0,2 нс на разряд). Это возродило интерес к структурам с последовательным переносом и, соответственно, к методам улучшения их быстродействия.

Идея построения сумматора с условным переносом такова. Имея сумматор с n разрядами, делят его на две равные группы с разрядностями n/2. Старшую группу дублируют, так что в схему входят три сумматора с разрядностью n/2. На одном суммируются младшие поля операндов $A_{мл}$ и $B_{мл}$. На втором — старшие поля операндов при условии $C_{вх} = 1$, в третьем — старшие поля операндов при условии $C_{вх} = 0$. После получения результата в младшем сумматоре становится известным фактическое значение переноса в старший сумматор, и из двух заготовленных заранее результатов выбирается тот, который нужен в данном случае. Цепь последовательного переноса здесь как бы укорачивается вдвое, т. к. обе половины сумматора работают параллельно во времени (рис. 2.34).

Накапливающий сумматор

Накапливающий сумматор обычно представляет собою сочетание комбинационного сумматора и регистра, работающее по формуле S := S + A, согласно которой к содержимому сумматора добавляется очередное слагаемое, и результат замещает старое значение суммы. Структура накапливающего сумматора показана на рис. 2.35. Очередное прибавление слагаемого тактируется синхроимпульсами СИ. Учитывая особенности функционирования, накапливающие сумматоры называют иногда аккумуляторами.

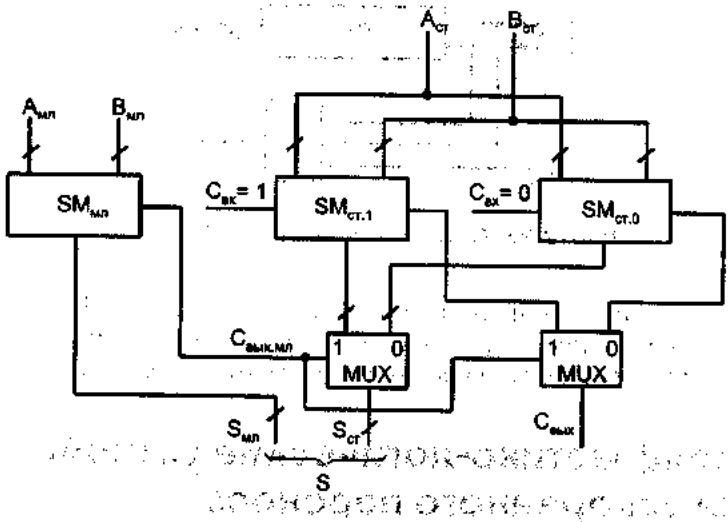


Рис. 2.34. Схема сумматора с условным переносом

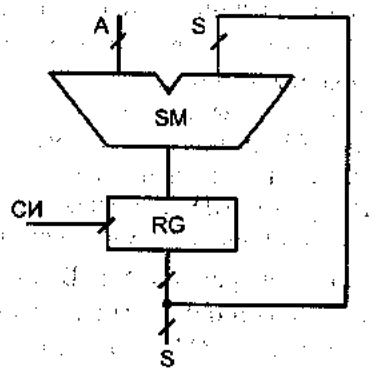


Рис. 2.35. Структура накапливающего сумматора

В сериях элементов имеются одноразрядные сумматоры, в том числе с дополнительной входной логикой, двухразрядные и четырехразрядные. Примером стандартных ИС сумматоров могут служить микросхемы ИМ3 серии К555, содержащие четырехразрядный сумматор с последовательным переносом и блок переноса (рис. 2.36), которые непосредственно пригодны для составления из них группового сумматора с цепным переносом.

Микросхемы четырехразрядных сумматоров можно также объединять в групповую структуру с межгрупповым параллельным переносом с помощью специальных блоков ускоренного переноса, которые рассмотрены в *§ 2.9*.

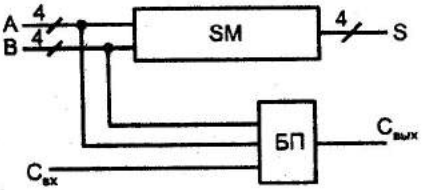


Рис. 2.36. Структура микросхемы К555ИМ3

В некоторых сериях элементов сумматоры отсутствуют. Причиной этого обычно является наличие арифметико-логического устройства (АЛУ), для которого режим суммирования есть один из возможных режимов.

§ 2.9. Арифметико-логические устройства и блоки ускоренного переноса

Арифметико-логические устройства АЛУ (ALU, Arithmetic-Logic Unit) выполняют над словами ряд действий. Основой АЛУ служит сумматор, схема которого дополнена логикой, расширяющей функциональные возможности АЛУ и обеспечивающей его перестройку с одной операции на другую.

Обычно АЛУ четырехразрядны и для наращивания разрядности объединяются с формированием последовательных или параллельных переносов. Логические возможности АЛУ разных технологий (ТТЛШ, КМОП, ЭСЛ) сходны. В силу самодвойственности выполняемых операций условное обозначение и таблица истинности АЛУ встречаются в двух вариантах, отличающихся взаимно инверсными значениями переменных.

АЛУ (рис. 2.37) имеет входы операндов A и B, входы выбора операций S, вход переноса $\overline{C}_i$ и вход M (Mode), сигнал которого задает тип выполняемых операций: логические (M = 1) или арифметико-логические (M = 0). Результат операции вырабатывается на выходах F, выходы G и H дают функции генерации и прозрачности, используемые для организаций параллельных переносов при наращивании размерности АЛУ. Сигнал $\overline{C}_0$ — выходной перенос, а выход A = B есть выход сравнения на равенство с открытым коллектором.

Перечень выполняемых АЛУ операций дан в табл. 2.13. Для краткости двоичные числа $s_3s_2s_1s_0$ представлены их десятичными эквивалентами. Под полужирными обозначениями 1 и 0 следует понимать наборы 1111 и 0000, входной перенос поступает в младший разряд слова, т. е. равен $000C_i$. Логические операции поразрядные, т. е. операция над словами A * B означает, что a_i * b_i при отсутствии взаимовлияния разрядов. При арифметических операциях учитываются межразрядные переносы.

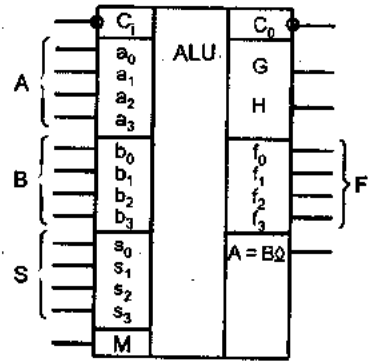


Рис. 2.37. Условное обозначение АЛУ

Таблица 2.13

S	Логические функции (M = 1)	Арифметико-логические функции (M = 0)
0	$\overline{A}$	$A + C_i$
1	$\overline{A \vee B}$	$A \vee B + C_i$
2	$\overline{A}B$	$A \vee \overline{B} + C_i$
3	0	$1 + C_i$
4	$\overline{A\overline{B}}$	$A + A\overline{B} + C_i$
5	$\overline{B}$	$A \vee B + A\overline{B} + C_i$
6	$A \oplus B$	$A + \overline{B} + C_i$
7	$A\overline{B}$	$A\overline{B} + 1 + C_i$
8	$\overline{A} \vee B$	$A + AB + C_i$
9	$\overline{A \oplus B}$	$A + B + C_i$
10	B	$A \vee \overline{B} + AB + C_i$
11	AB	$AB + 1 + C_i$
12	1	$A + A + C_i$
13	$A \vee \overline{B}$	$A \vee B + A + C_i$
14	$A \vee B$	$A \vee \overline{B} + A + C_i$
15	A	$A + 1 + C_i$

16 логических операций позволяют воспроизводить все функции двух переменных. В логико-арифметических операциях встречаются и логические, и арифметические операции одновременно.

Запись типа $A \vee \overline{B} + AB$ следует понимать так: вначале поразрядно выполняются операции инвертирования ($\overline{B}$), логического сложения ($A \vee \overline{B}$) и ум-

ножения (AB), а затем полученные указанным образом два четырехразрядных числа складываются арифметически.

При операциях над словами большой размерности АЛУ соединяются друг с другом с организацией последовательных (рис. 2.38, *а*) или параллельных (рис. 2.38, *б*) переносов. В последнем случае совместно с АЛУ применяют микросхемы — блоки ускоренного переноса (CRU, Carry Unit), получающие от отдельных АЛУ функции генерации и прозрачности, а также входной перенос и вырабатывающие сигналы переноса по формулам, приведенным в предыдущем параграфе.

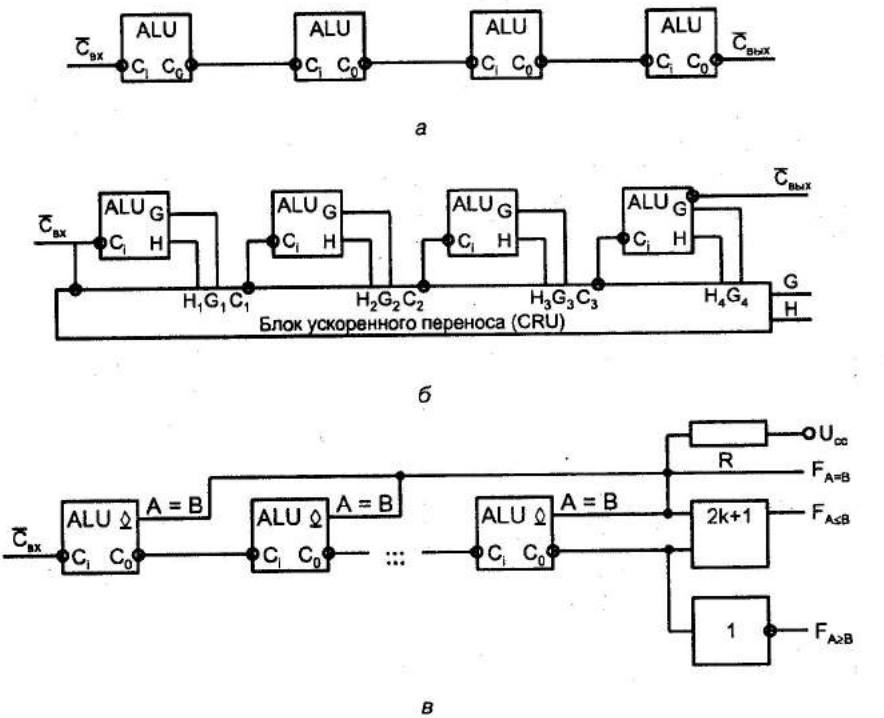


Рис. 2.38. Схемы наращивания АЛУ при последовательном (*а*) и параллельном (*б*) переносах и реализация функций компаратора для группы АЛУ (*в*)

Блок CRU вырабатывает также функции генерации и прозрачности для всей группы обслуживаемых им АЛУ, что при необходимости позволяет организовать параллельный перенос на следующем уровне (между несколькими группами из четырех АЛУ).

На рис. 2.38, *в* показаны способы выработки сигналов сравнения слов для группы АЛУ. Выход сравнения на равенство выполняется по схеме монтажной логики для выходов типа ОК. Комбинируя сигнал равенства слов с сигналом переноса на выходе группы при работе АЛУ в режиме вычитания, легко получить функции $F_{A\geq B}$ и $F_{A\leq B}$. Если A < B, то при вычитании возникает заем из старшего разряда и $F_{A\leq B} = 1$. Если заем отсутствует (A > B), то получим $F_{A\geq B} = 1$.

§ 2.10. Матричные умножители

Микросхемы множительных устройств появились в 1980-х гг., когда достигнутый уровень интеграции позволил разместить на одном кристалле достаточно большое количество логических элементов.

Структура матричных умножителей тесно связана со структурой математических выражений, описывающих операцию умножения.

Пусть имеются два целых двоичных числа без знаков $A_m = a_{m-1}...a_0$ и $B_n = b_{n-1}...b_0$. Их перемножение выполняется по известной схеме "умножения столбиком". Если числа четырехразрядные, т. е. m = n = 4, то

				×	a_3	a_2	a_1	a_0
					b_3	b_2	b_1	b_0
					a_3b_0	a_2b_0	a_1b_0	a_0b_0
+				a_3b_1	a_2b_1	a_1b_1	a_0b_1	
			a_3b_2	a_2b_2	a_1b_2	a_0b_2		
		a_3b_3	a_2b_3	a_1b_3	a_0b_3			
p_7		p_6	p_5	p_4	p_3	p_2	p_1	p_0

Произведение выражается числом $P_{m+n} = p_{m+n-1}\, p_{m+n-2} \cdots p_0$.

Члены вида a_ib_j, где i = 0... (m − 1) и j = 0... (n − 1) вырабатываются параллельно во времени конъюнкторами. Их сложение в столбцах, которое можно выполнять разными способами, составляет основную операцию для умножителя и определяет почти целиком время перемножения.

Матричные перемножители могут быть просто *множительными* блоками (МБ) или *множительно-суммирующими* (МСБ), последние обеспечивают удобство наращивания размерности умножителя.

МСБ реализует операцию $P = A_m \times B_n + C_m + D_n$, т. е. добавляет к произведению два слагаемых: одно разрядности m, совпадающей с разрядностью множимого, другое разрядности n, совпадающей с разрядностью множителя.

Множительно-суммирующие блоки

Множительно-суммирующий блок (МСБ) для четырехразрядных операндов без набора конъюнкторов, вырабатывающих члены вида a_ib_j, показан на рис. 2.39, *а*, где для одноразрядного сумматора принято обозначение (рис. 2.39, *б*).

Для построения МСБ чисел равной разрядности потребовалось n^2 конъюнкторов и n^2 одноразрядных сумматоров.

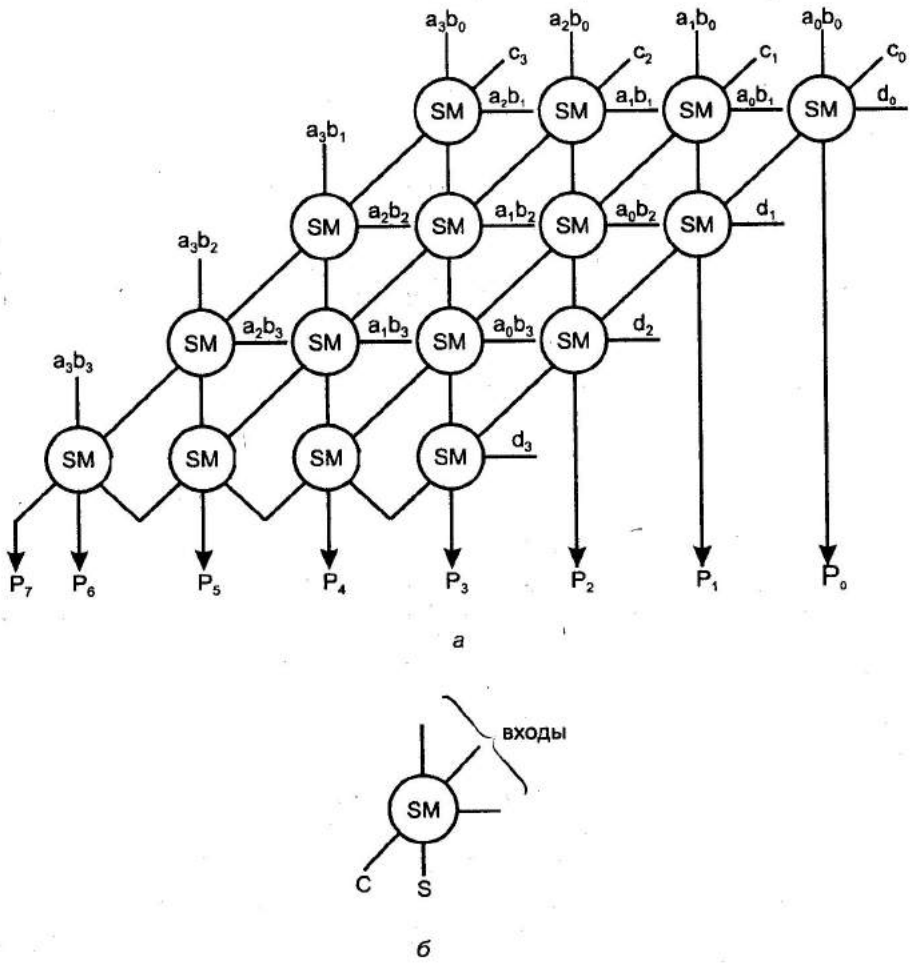


Рис. 2.39. Схема множительно-суммирующего блока для четырехразрядных сомножителей (*а*), обозначение одноразрядного сумматора для данной схемы (*б*)

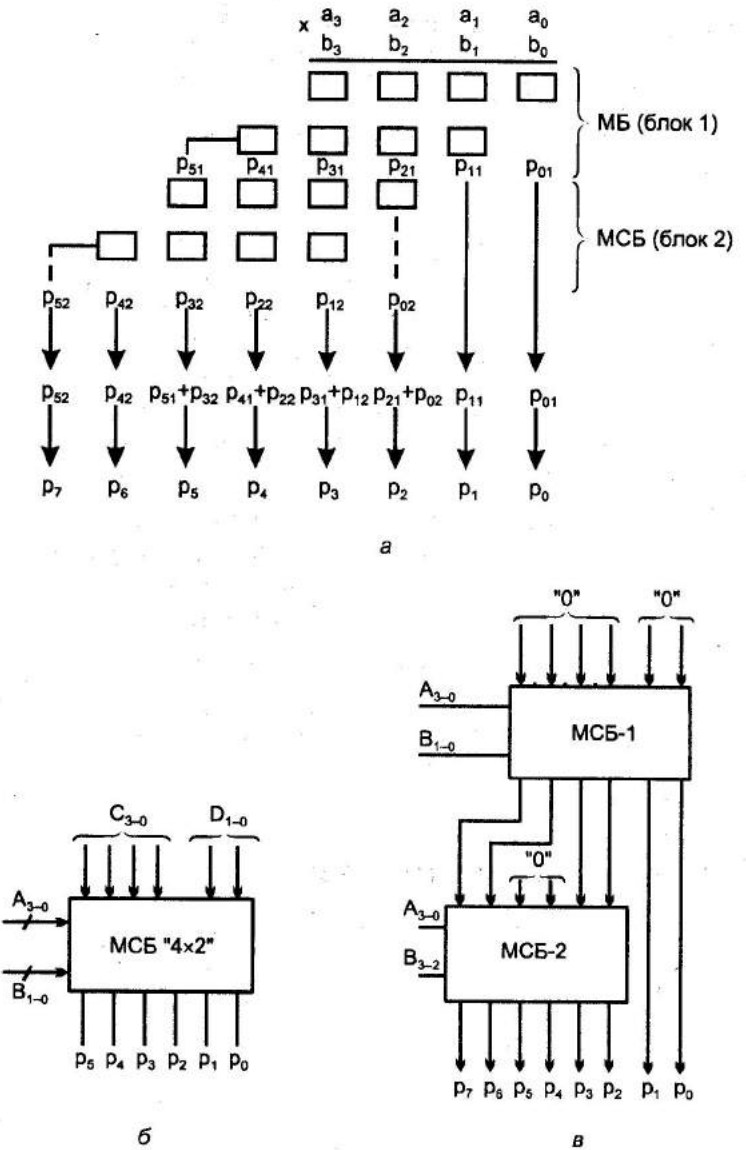


Рис. 2.40. К пояснению принципа наращивания размерности множительных устройств (*а*), условное обозначение множительно-суммирующего блока (*б*) и схема умножителя "4 × 4", построенная на множительно-суммирующих блоках "4 × 2" (*в*)

Максимальная длительность умножения — сумма задержек сигналов в конъюнкторах для выработки членов $a_i b_j$ и задержки в наиболее длинной

цепочке передачи сигнала в матрице одноразрядных сумматоров, равной $2n - 1$ ($m + n - 1$ в общем случае). Таким образом, $t_{MPL} = t_K + (2n - 1)t_{SM}$, где t_K — задержка конъюнктора.

Схема множительного блока отличается от схемы МСБ тем, что в ней отсутствуют сумматоры правой диагонали, т. к. при $C_m = 0$ и $D_n = 0$ они не требуются.

Построение умножителей большей размерности из умножителей меньшей размерности на основе МБ требует введения дополнительных схем, называемых "деревьями Уоллеса", которые имеются в некоторых зарубежных сериях. При использовании МСБ дополнительные схемы не требуются. Принцип наращивания размерности умножителя иллюстрируется на рис. 2.40, *а* на примере построения MPL "4 × 4" из МСБ "4 × 2". На поле частичных произведений выделены зоны, воспроизведение которых возможно на блоках размерности 4 × 2 (это две первые строки и две последние).

Перемножение в пределах зон дает частичные произведения p1 = = p51p41p31p21p11p01 и p2 = p52p42p32p22p12p02. Для получения конечного значения произведения эти частичные произведения нужно сложить с учетом их взаимного положения (сдвига одного относительно другого).

Схема, реализующая указанный принцип, изображена на рис. 2.40, *в*. В ней использовано условное обозначение МСБ (рис. 2.40, *б*). Для общности оба блока размерности 4 × 2 показаны как МСБ, хотя первый может быть просто множительным блоком, т. к. для него слагаемые C и D имеют нулевое значение.

Схемы ускоренного умножения

Для ускорения умножения разработан ряд алгоритмов, большой вклад в эти разработки внес Э. Бут (E. Boot). Рассмотрим процесс умножения по так называемому модифицированному алгоритму Бута *(умножение сразу на два разряда)*.

Из изложенного ранее видно, что основную задержку в процесс выработки произведения вносит суммирование частичных произведений. Уменьшение их числа сократило бы время суммирования. К этому приводит алгоритм, основанный на следующих рассуждениях.

Пусть требуется вычислить произведение

$$P = A \times B = A \times (b_{n-1}2^{n-1} + b_{n-2}2^{n-2} + \ldots + b_0 2^0). \qquad (2.1)$$

Непосредственное воспроизведение соотношения (2.1) связано с выработкой частичных произведений вида Ab_i2^i ($i = 0 \ldots n - 1$). Число таких произведений равно разрядности множителя n.

Выражение (2.1) можно видоизменить с помощью соотношения

$$b_i 2^i = b_i 2^{i+1} - 2b_i 2^{i-1}, \qquad (2.2)$$

справедливость которого очевидна.

Это соотношение позволяет разреживать последовательность (спектр) степеней в сумме частичных произведений. Можно, например, исключить четные степени, как показано на рис. 2.41, *а*.

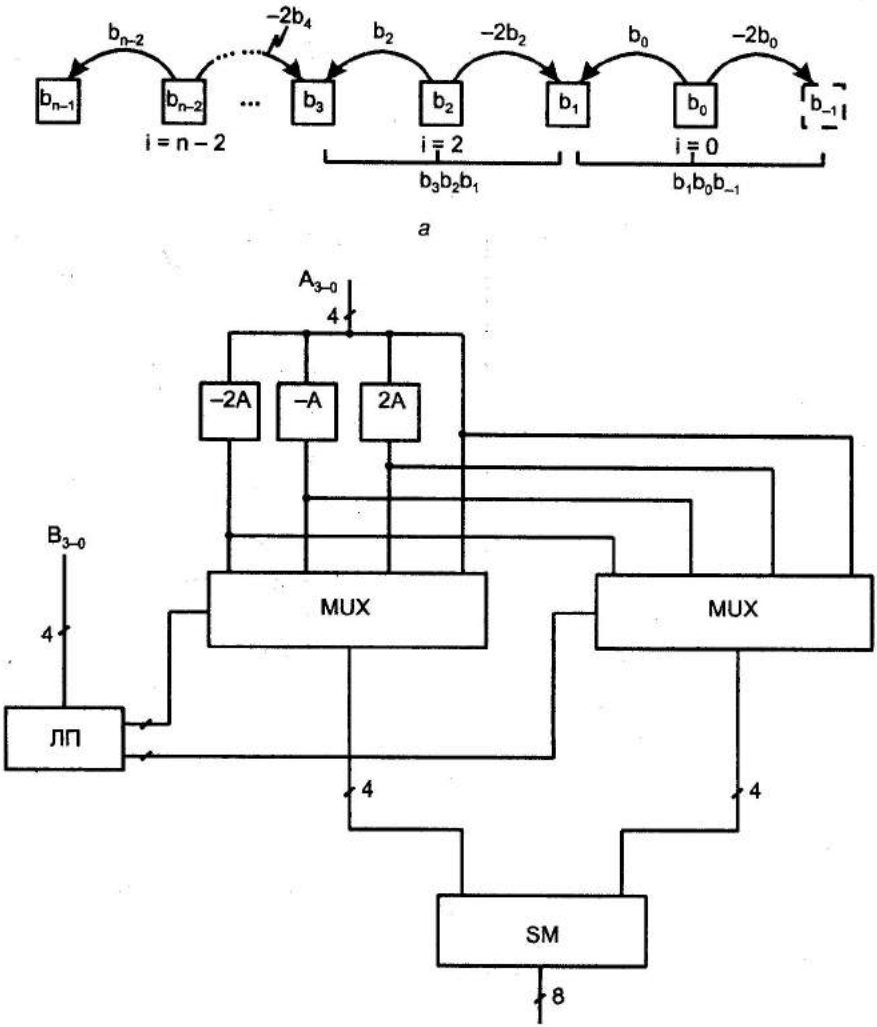


Рис. 2.41. К пояснению принципа быстрого умножения "сразу на два разряда" (*а*) и схема быстрого умножения (*б*)

Исключение четных (или нечетных) степеней не только изменяет значения оставшихся частичных произведений, но и сокращает их число примерно вдвое, что, в конечном счете ускоряет выработку произведения. Для того чтобы "разнести по соседям" член со степенью 2^0, расширим разрядную сетку, введя слагаемое $b_{-1}2^{-1}$ (нулевой разряд с номером -1).

Оставшиеся частичные произведения имеют вид

$$R_i = A(-2b_{i+1} + b_i + b_{i-1})2^i.$$

Так как число частичных произведений уменьшилось примерно вдвое, при применении этого алгоритма говорят об умножении сразу на два разряда.

Для всех возможных сочетаний b_{i+1}, b_i, b_{i-1} можно составить таблицу (табл. 2.14) частичных произведений.

Таблица 2.14

b_{i+1}	b_i	b_{i-1}	Значение скобки	$R_i/2^i$	Операция для получения $R_i/2^i$
0	0	0	0	0	Заменить A нулем
0	0	1	1	A	Скопировать A
0	1	0	1	A	Скопировать A
0	1	1	2	2A	Сдвинуть A влево
1	0	0	–2	–2A	Сдвинуть A влево и преобразовать в дополнительный код
1	0	1	–1	–A	Преобразовать A в дополнительный код
1	1	0	–1	–A	Преобразовать A в дополнительный код
1	1	1	0	0	Заменить A нулем

Пример

Пусть требуется умножить 1010_2 на 0111_2, т. е. 10×7. При разреживании частичных произведений оставим только нечетные, как показано на рис. 2.41, *а*. Расширив разрядную сетку множителя, имеем $B = b_4b_3b_2b_1b_0b_{-1}b_{-2} = 0011100$.

Первому частичному произведению соответствует тройка $b_0b_{-1}b_{-2} = 100$. Из табл. 2.14 получаем, что этой тройке соответствует частичное произведение — $-2A\ 2^{-1} = -A$, для получения которого требуется перевести A в дополнительный код. Сама величина A в пределах разрядной сетки произведения должна быть записана как 00001010, ее обратный код 11110101 и дополнительный код 11110110.

Второму частичному произведению соответствует тройка $b_2b_1b_0 = 111$, следовательно, второе частичное произведение равно нулю (табл. 2.14).

Третьему частичному произведению соответствует тройка $b_4b_3b_2 = 001$, следовательно, оно имеет вид $A\ 2^3 = 01010000$.

Для получения результата заданного умножения требуется сложить частичные произведения:

$$\begin{array}{l} 11110110 \\ 01010000 \\ \hline 01000110 = 2^6 + 2^2 + 2^1 = 64 + 4 + 2 = 70. \end{array}$$

Схема, реализующая алгоритм быстрого умножения сразу на два разряда, показана на рис. 2.41, *б*.

Множимое A поступает в этой схеме на ряд преобразователей, заготавливающих все возможные варианты частичных произведений (−2A, −A, 2A), кроме самого A и нуля, которые не требуют схемной реализации. Множитель B поступает на логический преобразователь ЛП, который анализирует тройки разрядов, декодирует их и дает мультиплексорам MUX сигналы выбора того или иного варианта частичных произведений. Окончательный результат получается суммированием частичных произведений с учетом их взаимного сдвига в разрядной сетке. Размерность умножителя 4 × 4.

Приведенные ранее примеры множительных устройств касались операций с прямыми кодами. В этом случае умножение знакопеременных чисел сведется только к выработке знакового разряда как суммы по модулю 2 знаковых разрядов сомножителей. Если же числа представлены не прямыми кодами со знаковыми разрядами, а, например, дополнительными кодами, то, имея рассмотренные ранее умножители, можно дополнить их преобразователями дополнительного кода в прямые на входах и преобразователем прямого кода в дополнительный на выходе или использовать схемы, непосредственно реализующие алгоритмы умножения дополнительных кодов (см., например, [37]).

Разработке матричных умножителей уделяют внимание многие фирмы. В отечественных сериях МИС/СИС имеются умножители малой размерности (2 × 2, 4 × 4, 4 × 2 и др.). В сериях БИС размерности умножителей значительно больше. В серии 1802, например, имеются умножители 8 × 8, 12 × 12, 16 × 16 (ВР3, ВР4 и ВР5 соответственно). В схемотехнике ЭСЛ выполнен умножитель 1800ВР1 (8 × 8 за 17 нс). Зарубежные фирмы разработали умножители (фирмы BIT, Hitachi и др.) размерностями 16 × 16 и более с временами умножения 3...5 нс. Несколько лет назад предприятие "Интеграл" (г. Минск) выпустило умножитель КА1843ВР1 размерностью 32 × 32 со временем умножения 250 нс в корпусе с 172 выводами.

§ 2.11. Быстрые сдвигатели

Сдвиги слов в разрядной сетке — типовая операция, используемая при вычислениях с представлением чисел с плавающей точкой, умножениях чисел на константу и в других случаях. Среди самых распространенных узлов

цифровых устройств (регистров) имеются сдвигающие, но с их помощью сдвиг на несколько разрядов можно реализовать только как результат нескольких одноразрядных, а это занимает много времени. Поэтому были разработаны так называемые быстрые сдвигатели, которые перемещают слово сразу на несколько разрядов в соответствии с указанным значением сдвига.

Схемотехника быстрых сдвигателей насчитывает ряд вариантов, из которых основными являются сдвигатели, управляемые кодом "1 из N" (Barrel Shifters) и сдвигатели, управляемые двоичным кодом (Logarithmic Shifters).

Сдвигатель, управляемый кодом "1 из N"

Структура этого варианта показана на рис. 2.42, *б* с обозначением ключевых транзисторов согласно рис. 2.42, *а*. Ключевые транзисторы (Pass transistors) для наглядности представления структуры изображены кружками, причем по прямой линии расположены выводы истока и стока (т. е. в эту линию включен управляемый ключ), а боковой вывод связан с затвором, и на него подается управляющее напряжение. Обозначение управляющего напряжения, замыкающего ключ, записывается в кружке в сокращенном виде. Если ключ замыкается сигналом "Сдвиг N", т. е. при команде сдвига на N разрядов, то в кружке ставится символ N.

Основа схемы сдвигателя — матрица ключевых транзисторов, управляемых кодами "1 из N" согласно табл. 2.15 (сдвиги производятся в сторону разрядов с меньшими номерами).

Таблица 2.15

Операция	Сигналы управления			
	Сдвиг 0	Сдвиг 1	Сдвиг 2	Сдвиг 3
Передача без сдвига	1	0	0	0
Сдвиг на 1 разряд	0	1	0	0
Сдвиг на 2 разряда	0	0	1	0
Сдвиг на 3 разряда	0	0	0	1

Требуемый результат получается за счет определенной схемы соединений ключей. В схеме рис. 2.42, *б* предусмотрено автоматическое повторение знакового разряда и на выходы B передаются следующие коды:

- при сдвиге на 0 разрядов: $A_3A_2A_1A_0$;
- при сдвиге на 1 разряд: $A_3A_3A_2A_1$;
- при сдвиге на 2 разряда: $A_3A_3A_3A_2$;
- при сдвиге на 3 разряда: $A_3A_3A_3A_3$.

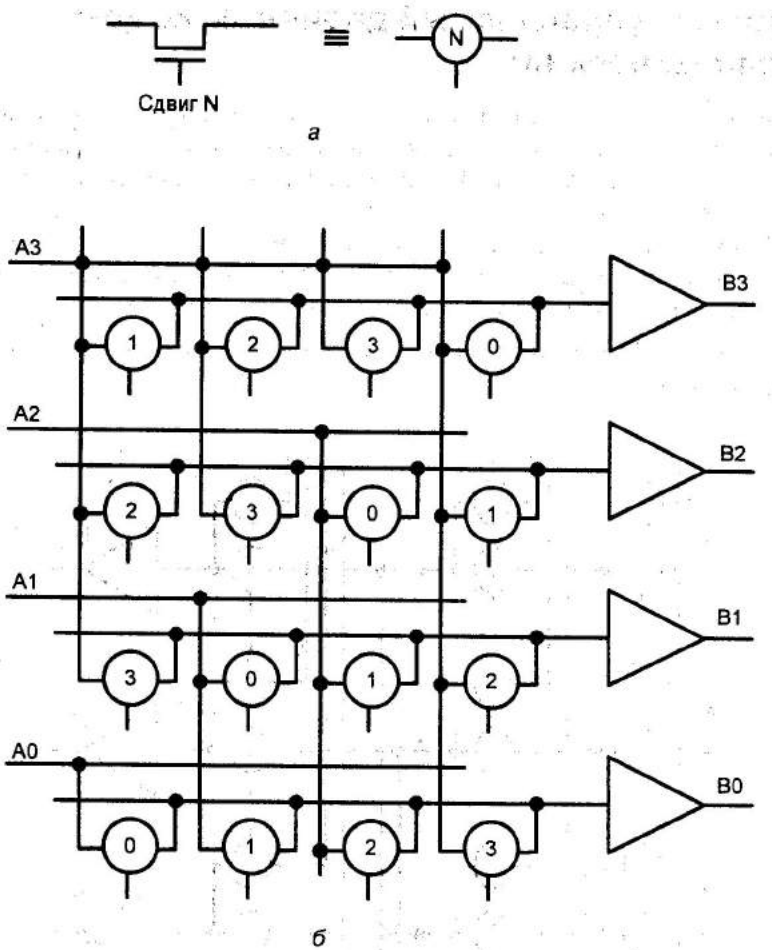


Рис. 2.42. Условное обозначение ключевых транзисторов, принятое при изображении структуры сдвигателя, управляемого кодом "1 из N" (*а*) и структура сдвигателя (*б*)

Сдвигатель типа Barrel Shifter отличается высоким быстродействием, т. к. сигнал сдвига замыкает для каждой передачи входного бита на выходной буфер цепь только из одного ключа. Однако это справедливо, если управляющий код является кодом "1 из N". В большинстве случаев управляющие коды исходно вырабатываются как двоичные, и перед сдвигателем требуется включать дешифратор, что увеличивает как сложность схемы, так и задержки в ее работе. В этих случаях, особенно при больших величинах максимально возможных сдвигов, преимущество по технико-экономическим показателям обычно оказывается на стороне сдвигателя, непосредственно управляемого двоичными кодами и называемого логарифмическим.

Сдвигатель, управляемый двоичным кодом (логарифмический)

В отличие от предыдущего варианта этот сдвигатель в качестве основы имеет не единую матрицу ключевых транзисторов, а схему, состоящую из нескольких каскадов (ступеней). Число ступеней N связано со значением максимального сдвига S зависимостью $N = \log_2 S$, что и объясняет название "логарифмический" применительно к данному типу сдвигателя. Требуемое значение величины сдвига образуется в этом сдвигателе как композиция нескольких сдвигов, каждый из которых равен числу 2^i, где i = 1, 2, 3.

Структура логарифмического сдвигателя показана на рис. 2.43 на примере четырехразрядного фрагмента.

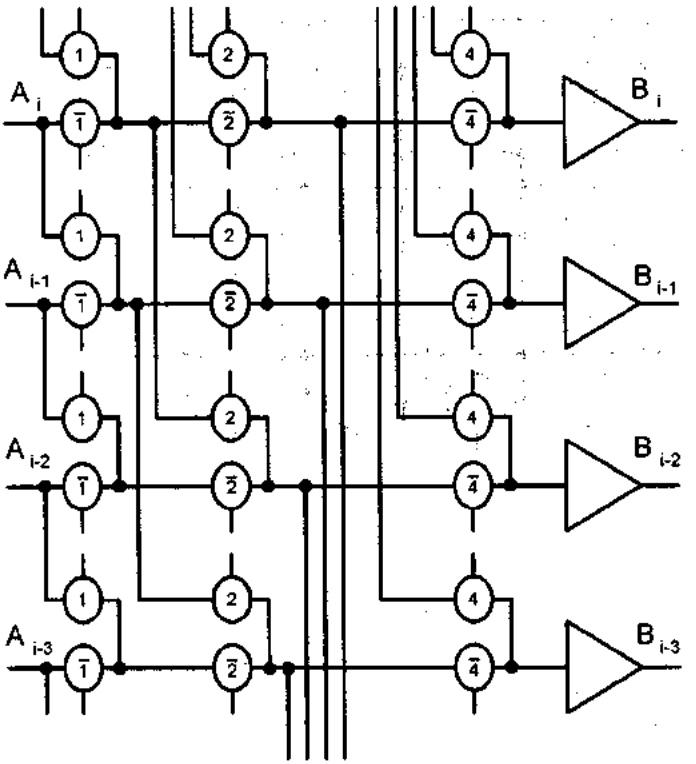


Рис. 2.43. Фрагмент структуры логарифмического сдвигателя

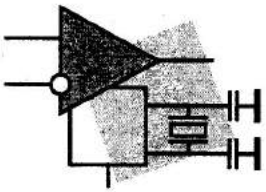
Обозначения сигналов управления ключевыми транзисторами совпадают с принятыми на рис. 2.42, кроме того, на рисунке показаны и инверсные сиг-

налы. Как видно из схемы, сигналы от разрядов входного слова до входов выходных буферов проходят через цепочку последовательно включенных транзисторов, число которых равно $\log_2 S$. Это замедляет процесс сдвига, а поскольку задержка цепочки из RC-звеньев, которая создается последовательно включенными пасс-транзисторами, зависит от их числа сильнее, чем по линейному закону, в длинные цепочки целесообразно вводить промежуточные буферные каскады, разбивая цепочки на более короткие части.

Для иллюстрации работы сдвигателя рассмотрим сдвиг на 5 разрядов. В двоичном коде число 5 представляется кодом $101 = 1\times2^2 + 0\times2^1 + 1\times2^0$, соответственно чему первая и третья ступени устанавливаются в режим сдвига, а вторая в режим пропуска сигнала без сдвига.

Литература к главе: [2], [3], [9], [12], [19], [25], [26], [30], [33], [38], [44], [45], [46], [47], [49], [62], [65].

Глава 3



Функциональные узлы последовательностного типа (автоматы с памятью)

§ 3.1. Триггерные устройства (элементарные автоматы). Классификация. Основные сведения

Триггеры — элементарные автоматы, содержащие собственно элемент памяти (фиксатор) и схему управления. Фиксатор строится на двух инверторах, связанных друг с другом "накрест", так что выход одного соединен с входом другого. Такое соединение дает *цепь с двумя устойчивыми состояниями* (рис. 3.1). Действительно, если на выходе инвертора 1 имеется логический нуль, то он обеспечивает на выходе инвертора 2 логическую единицу, благодаря которой сам и существует. То же согласование сигналов имеет место и для второго состояния, когда инвертор 1 находится в единице, а инвертор 2 — в нуле. Любое из двух состояний может существовать неограниченно долго.

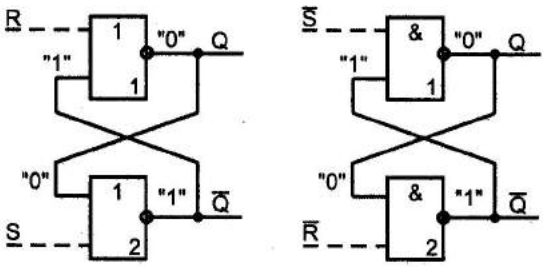


Рис. 3.1. Схемы фиксаторов с входами управления на элементах ИЛИ-НЕ и И-НЕ

Переходное состояние, в котором инверторы активны, неустойчиво. Это можно показать, имея в виду, что напряжения в любой цепи не являются идеально постоянными, а всегда имеют место флуктуации. Флуктуации обязательно приведут фиксатор в одно из двух стабильных состояний, т. к. из-за наличия в схеме петли положительной обратной связи любое изменение режима вызывает продолжение в том же направлении, пока фиксатор не перейдет в устойчивое состояние, когда петля обратной связи как бы разрывается вследствие потери инверторами усилительных свойств (переход в режимы отсечки и насыщения, свойственные устойчивым состояниям).

Чтобы управлять фиксатором, нужно иметь в логических элементах дополнительные входы, превращающие инверторы в элементы И-НЕ либо ИЛИ-НЕ. На входы управления поступают внешние установочные сигналы.

Установочные сигналы показаны на рис. 3.1 штриховыми линиями. Буквой R латинского алфавита (от Reset) обозначен сигнал установки триггера в нуль (сигнал сброса), а буквой S (от Set) — сигнал установки в состояние логической единицы (сигнал установки). Состояние триггера считывается по значению прямого выхода, обозначаемого как Q. Чаще всего триггер имеет и второй выход с инверсным сигналом $\overline{Q}$. Для фиксатора на элементах ИЛИ-НЕ установочным сигналом является единичный, поскольку только он приводит логический элемент в нулевое состояние независимо от сигналов на других входах элемента. Для фиксатора на элементах И-НЕ установочным сигналом является нулевой, как обладающий тем же свойством однозначно задавать состояние элемента независимо от состояний других входов.

Практически все серии цифровых ИС содержат готовые триггеры, и поэтому задача проектировщика — правильное использование имеющихся триггеров. Отсюда большое значение приобретает классификация триггеров, изучение их параметров и особенностей функционирования.

Классификация триггеров

Классификация триггеров проводится по признакам логического функционирования и способу записи информации (рис. 3.2).

По логическому функционированию различают триггеры типов RS, D, T, JK и др. Кроме того, используются комбинированные триггеры, в которых совмещаются одновременно несколько типов, и триггеры со сложной входной логикой (группами входов, связанных между собой логическими зависимостями).

Триггер типа RS имеет два входа — установки в единицу (S) и установки в нуль (R).

Одновременная подача сигналов установки S и сброса R не допускается (эта комбинация сигналов называется *запрещенной*).

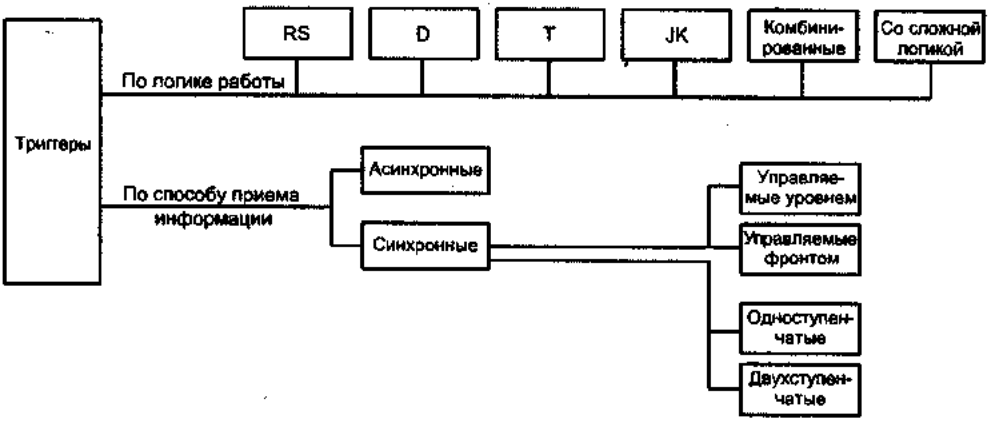


Рис. 3.2. Классификация триггеров, используемых в практической схемотехнике

Триггер типа D (от слова Delay — задержка) имеет один вход. Его состояние повторяет входной сигнал, но с задержкой, определяемой тактовым сигналом.

Триггер типа T изменяет свое состояние каждый раз при поступлении входного сигнала. Такой триггер имеет один вход и называется *триггером со счетным входом* или *счетным триггером.*

Триггер типа JK универсален, имеет входы установки (J) и сброса (K), подобные входам триггера RS. В отличие от последнего, допускает ситуацию с одновременной подачей сигналов на оба эти входа (J = K = 1). В этом режиме работает как счетный триггер относительно третьего (тактового) входа.

В комбинированных триггерах совмещаются несколько режимов. Например, триггер типа RST — счетный триггер, имеющий также входы установки и сброса.

Примером триггера со сложной входной логикой служит JK-триггер с группами входов $J_1J_2J_3$ и $K_1K_2K_3$, соединенными операцией конъюнкции:

$$J = J_1J_2J_3,\ K = K_1K_2K_3.$$

По способу записи информации различают асинхронные (нетактируемые) и синхронные (тактируемые) триггеры. В нетактируемых переход в новое состояние вызывается непосредственно изменениями входных информационных сигналов. В тактируемых триггерах, имеющих специальный вход, переход происходит только при подаче на этот вход тактовых сигналов. Тактовые сигналы называют также синхронизирующими, исполнительными, командными и т. д. Обозначаются они буквой C (от слова Clock).

По способу восприятия тактовых сигналов триггеры делятся на *управляемые уровнем* и *управляемые фронтом.* Управление уровнем означает, что при одном уровне тактового сигнала триггер воспринимает входные сигналы и реагирует

на них, а при другом не воспринимает и остается в неизменном состоянии. При управлении фронтом разрешение на переключение дается только в момент перепада тактового сигнала (на его фронте или спаде). В остальное время независимо от уровня тактового сигнала триггер не воспринимает входные сигналы и остается в неизменном состоянии. Триггеры, управляемые фронтом, называют также триггерами с *динамическим управлением*.

Динамический вход может быть прямым или инверсным. Прямое динамическое управление означает разрешение на переключение при изменении тактового сигнала с нулевого значения на единичное, инверсное — при изменении тактового сигнала с единичного значения на нулевое.

По характеру процесса переключения триггеры делятся на *одноступенчатые* и *двухступенчатые*.

В одноступенчатом триггере переключение в новое состояние происходит сразу, в двухступенчатом — по этапам. Двухступенчатые триггеры состоят из входной и выходной ступеней. Переход в новое состояние происходит в обеих ступенях поочередно. Один из уровней тактового сигнала разрешает прием информации во входную ступень при неизменном состоянии выходной ступени. Другой уровень тактового сигнала разрешает передачу нового состояния из входной ступени в выходную.

На рис. 3.3 показаны процессы, происходящие в синхронных (тактируемых) триггерах. На диаграммах тактовых импульсов отмечено содержание процессов на отдельных этапах, под диаграммами даны обозначения входов для соответствующих триггеров.

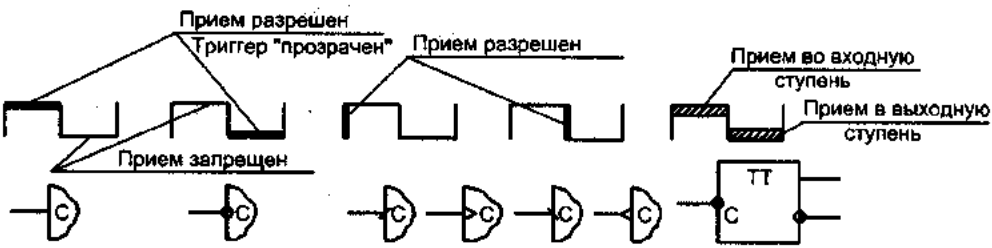


Рис. 3.3. Временные диаграммы, поясняющие работу синхронных триггеров, и условные обозначения тактирующих входов

В практике проектирования используется термин "триггер-защелка" (Latch). Под этим понимается триггер, который прозрачен при одном уровне тактового сигнала и переходит в режим хранения при другом.

Как видно из рисунка, двухступенчатый триггер обозначается двумя буквами Т. Двухступенчатые триггеры часто называют также триггерами типа MS (от англ. *Master-Slave*, т. е. "хозяин"—"раб"). Эта аббревиатура отражает характер

работы триггера: входная ступень вырабатывает новое значение выходной переменной Q, а выходная его копирует.

Времена предустановки и выдержки

С синхронизацией (тактированием) триггера связаны два важных параметра — *время предустановки* t_{SU} (Set-Up Time) и *время выдержки* t_H (Hold Time). Важность этих параметров обуславливается еще и тем, что они свойственны не только триггерам, но и другим устройствам. Время t_{SU} — это интервал до поступления синхросигнала, в течение которого информационный сигнал должен оставаться неизменным (рис. 3.4). Время выдержки t_H — это время после поступления синхросигнала, в течение которого информационный сигнал должен оставаться неизменным. Соблюдение времен предустановки и выдержки обеспечивает правильное восприятие триггером входной информации.

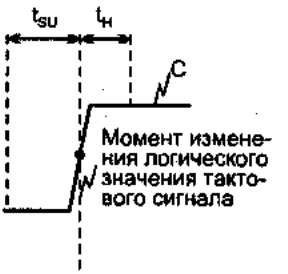


Рис. 3.4. К пояснению параметров предустановки и выдержки для синхронных триггеров

Ряд других временных параметров триггеров непосредственно связан с задержками сигнала при прохождении через триггер и не требует специальных пояснений.

Способы описания триггеров

Логическое функционирование триггеров описывается способами, принятыми для автоматов вообще: таблицами истинности, картами Карно, характеристическими уравнениями, диаграммами состояний, "словарями" (иной формой диаграмм состояний).

Ниже описывается логика работы наиболее распространенных триггеров JK и D. Работа триггера RS совпадает с работой триггера JK во всем за исключением наличия запрещенного состояния. Работа триггера T кратко характеризуется в таблице "словарей", приводимой далее.

Таблицу истинности триггера JK можно записать в полном (табл. 3.1) или сокращенном виде (табл. 3.2). Через Q_H обозначено новое состояние триггера (после переключения).

Таблица 3.1

J	K	Q	Q_H
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Таблица 3.2

J	K	Q_H
0	0	Q1
0	1	0
1	0	1
1	1	$\overline{Q}$

Карта Карно для JK-триггера показана на рис. 3.5. Из нее можно получить характеристическое уравнение триггера $Q_H = J\overline{Q} \vee Q\overline{K}$.

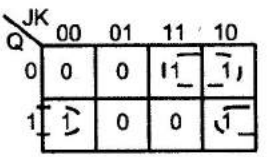
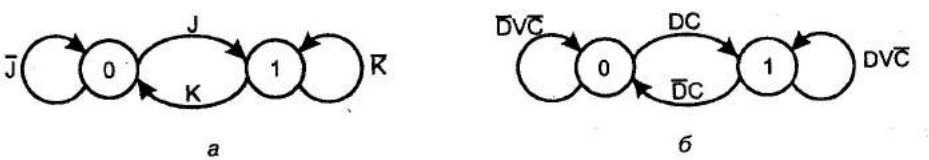


Рис. 3.5. Карта Карно для JK-триггера

Переведя уравнение в логический базис элементов, на которых строится триггер, получим *структурное уравнение* триггера, определяющее конфигурацию его схемы.

Диаграмма состояний (рис. 3.6, *а*) отражает наличие у триггера двух устойчивых состояний и условия перехода из одного состояния в другое. *Словарь триггера* (табл. 3.3) дает ту же информацию в аналитической форме и является инструментом проектирования схем, содержащих триггеры.



а *б*

Рис. 3.6. Диаграммы состояний (графы переходов) для триггеров JK (*а*) и D (*б*)

Для D-триггера сокращенная таблица истинности дана в табл. 3.4, а словарь — в табл. 3.5. Характеристическое уравнение триггера $Q_H = D$.

Таблица 3.3

Переход	J	K
0→0	0	X
0→1	1	X
1→0	X	1
1→1	X	0

Таблица 3.4

D	Q_H
0	0
1	1

Таблица 3.5

Переход	D
0→0	0
0→1	1
1→0	0
1→1	1

Диаграмма состояний с учетом синхросигнала C представлена на рис. 3.6, *б*. Синхросигнал C показан, т. к. триггеры типа D всегда тактируются.

Таблица 3.6

Переход	R	S	T
0→0	X	0	0
0→1	0	1	1
1→0	1	0	1
1→1	0	X	0

Словари триггеров RS и T имеют вид, представленный в табл. 3.6.

Важным способом описания функционирования триггеров (как и других автоматов) являются *временные диаграммы*, отражающие не только логическое функционирование схемы, но и ее поведение во времени. Это поведение другими способами описания работы триггеров не отображается, и поэтому в ряде случаев временные диаграммы незаменимы.

§ 3.2. Схемотехника триггерных устройств

Знание основ схемотехники триггерных устройств облегчает правильное их применение в различных условиях, в том числе и не всегда описанных в справочной литературе.

Прежде всего, отметим, что между триггерами RS и D, с одной стороны, и T и JK, с другой, имеется существенная разница. Первые имеют разомкнутую структуру (о внутренних обратных связях в схеме фиксатора сейчас речь не идет), а вторые используют выходные сигналы для воздействия на свои входы.

Возьмем за основу триггер RS и рассмотрим схемы информационных связей, создающих другие типы триггеров. Триггер типа D получается из триггера RS, если подавать на вход S значение D, а на вход R его инверсию (рис. 3.7, *а*). Триггер типа T образуется на основе триггера RS по схеме рис. 3.7, *б*. В этом случае роль счетного входа играет тактирующий вход. Действительно, при каждом разрешении приема информации по входу так-

тирования триггер по обратным связям принимает состояние, противоположное текущему, т. е. переключается. Триггер T аналогичным способом можно получить и на основе D-триггера (рис. 3.7, *в*).

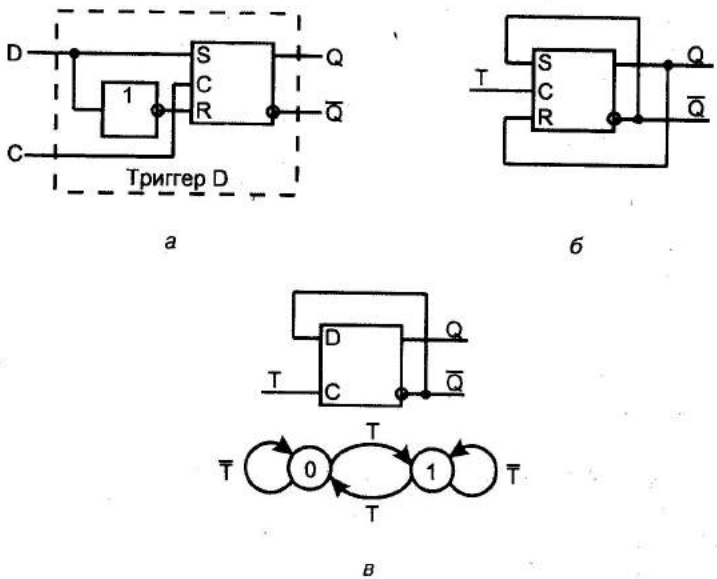


Рис. 3.7. Схемы информационных связей, образующих триггеры D (*а*) и T (*б*, *в*), и диаграмма состояний счетного триггера

Обратные связи с выхода триггера на его вход ведут к опасности появления в схеме режима генерации. Действительно, если применяется триггер с управлением уровнем, то при T = 1 триггеру, находящемуся в состоянии Q, разрешен прием состояния $\overline{Q}$, и он переключится. После этого, если значение T остается единичным, то повторяется та же ситуация — триггер примет состояние Q и вновь переключится. Таким образом, пока T = 1, схема ведет себя как генератор, что ясно видно из ее диаграммы состояний (см. рис. 3.7, *в*).

С режимом генерации можно бороться путем такого ограничения длительности сигнала T, при котором триггер успел бы переключиться всего один раз. Но практически это нереально из-за разброса параметров триггеров. Длительность сигнала, необходимая для однократного переключения медленного триггера, может оказаться достаточной для двукратного переключения быстрого. Практически работоспособность счетного триггера обеспечивается применением в рассматриваемой структуре *непрозрачных триггеров* (двухступенчатых или с динамическим управлением) или внутренних задержек.

Схему информационных связей, создающую структуру JK-триггера, определим формальным методом, пользуясь данными табл. 3.7.

Таблица 3.7

J	K	Q	Q_H	Переход	S	R
0	0	0	0	0→0	0	0
0	0	1	1	1→1	0	0
0	1	0	0	0→0	0	0
0	1	1	0	1→0	0	1
1	0	0	1	0→1	1	0
1	0	1	1	1→1	0	0
1	1	0	1	0→1	1	0
1	1	1	0	1→0	0	1

Левая часть таблицы показывает функционирование JK-триггера, а правые столбцы указывают сигналы RS-триггера, обеспечивающие необходимые переходы. Из этих столбцов получаем

$$S = J\overline{K}\,\overline{Q} \vee JK\overline{Q} = J\overline{Q},\ R = \overline{J}KQ \vee JKQ = KQ.$$

Схема информационных связей, образующих структуру JK-триггера, показана на рис. 3.8. Во избежание режима генерации, как и для счетного триггера, здесь требуется применять RS-триггер двухступенчатого типа или с динамическим управлением. Для триггера с двухступенчатой структурой и поочередным приемом информации во входную и выходную ступени диаграмма работы T-триггера изображена на рис. 3.9. Первая цифра кода состояния соответствует входной ступени, вторая — выходной. Видно, что в такой структуре режим генерации не возникает. То же относится и к JK-триггеру двухступенчатого типа. Работоспособные JK-триггеры строятся также на основе триггеров с динамическим управлением или внутренними задержками.

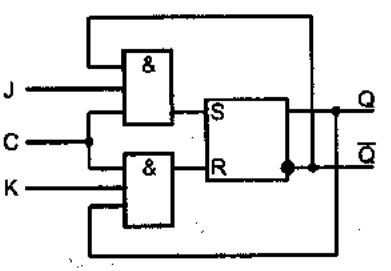


Рис. 3.8. Схема информационных связей, образующих JK-триггер

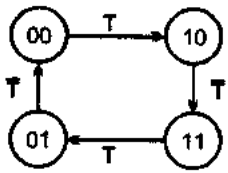


Рис. 3.9. Диаграмма состояний Т-триггера, построенного на двухступенчатых триггерах

Рассмотрим далее несколько схемотехнических вопросов работы конкретных вариантов триггеров.

Функционирование простейшего асинхронного RS-триггера (см. рис. 3.1) описано в начале этого параграфа. В дополнение к сказанному, оценим необходимую длительность входных сигналов триггера и причину существования для него запрещенной комбинации RS = 1.

Входной сигнал для обеспечения переключения триггера должен сохраняться до прихода по цепи обратной связи дублирующего его сигнала на втором входе соответствующего логического элемента, т. е. в течение времени, равного двум задержкам элементов, на которых собрана схема.

Комбинация входных сигналов R = S = 1 запрещена. Что же произойдет, если она возникнет? Видно, что в этом случае оба выхода триггера (для примера взята схема на элементах И-НЕ) станут единичными. Если после запрещенной комбинации входных сигналов 11 на входах появится комбинация 01 или 10, триггер перейдет в состояние, соответствующее этой комбинации. Если же после запрещенной комбинации 11 появится комбинация 00 (режим хранения), то возникнет непредсказуемая ситуация. Вначале оба элемента находятся в единичных состояниях, но, в конечном счете, схема перейдет в одно из устойчивых состояний, когда один из элементов имеет нулевое состояние, а другой — единичное. Происходит противоборство элементов, каждый из которых стремится навязать соседу свою "волю". Исход борьбы заранее неизвестен. Именно это заставляет считать комбинацию 11 запрещенной, т. к. пользоваться схемой, поведение которой непредсказуемо, если не говорить о специальных применениях, нельзя.

В схеме *синхронного RS-триггера* (рис. 3.10, *а*) при C = 0 на выходах элементов 1 и 2 действуют единичные сигналы, и фиксатор (элементы 3 и 4) хранит неизменное состояние. Если C = 1, то для сигналов S и R элементы 1, 2 становятся инверторами, и схема фиксатора получает нулевой сигнал установки или сброса от входа, на котором действует единичный сигнал. Таким образом, переключение разрешается только при C = 1. Условное обозначение триггера показано на рис. 3.10, *б*.

Триггеры типа D реализуют задержку сигнала с помощью тактирования, принимая сигнал только по разрешению тактового сигнала C.

Если на рис. 3.10, *а* заменить входы S и R входом D с введением в схему линий, показанных штрихами, то получится схема триггера-защелки типа D.

В этом триггере при C = 0 на выходах элементов 1 и 2 действуют единичные сигналы и фиксатор сохраняет свое состояние. При C = 1 состояние элементов определяется значением D. Если D = 1, то и на выходе Q установится единица, а при D = 0 и Q = 0.

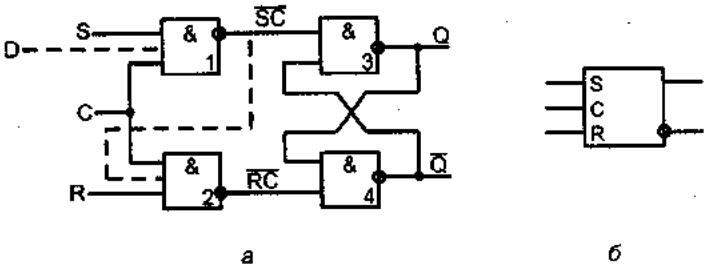


Рис. 3.10. Схема синхронного RS-триггера с управлением уровнем (*а*) и ее условное обозначение (*б*)

Схемотехническая *реализация триггеров на элементах КМОП* имеет некоторые особенности. К обычным элементам (инверторам, элементам И-НЕ, ИЛИ-НЕ и т. п.) добавляется так называемый *двунаправленный ключ* (рис. 3.11, *а*).

Ключ составлен из двух параллельно включенных транзисторов, имеющих взаимоинверсные управляющие напряжения затворов. Эти напряжения обозначены на рисунке через C и $\overline{C}$. Один из транзисторов имеет n-канал, другой — p-канал. Такая схема позволяет получать замкнутое состояние ключа с приблизительно постоянным и малым сопротивлением при разных значениях напряжения на переключаемом участке. Подавая на затвор транзистора с p-каналом низкое напряжение, а на затвор транзистора с n-каналом — высокое, можно привести ключ в замкнутое состояние, в котором оба транзистора открыты. Изменение напряжения на переключаемом участке при постоянстве напряжений на затворах транзисторов ведет к изменениям напряжений "затвор-исток" (или "затвор-сток", что безразлично ввиду симметричности конструкции МОП-транзисторов). Если бы ключ выполнялся на одиночном транзисторе, то изменение переключаемого напряжения резко меняло бы сопротивление ключа, в частности, возникали бы режимы с большим его сопротивлением, что неприемлемо. В ключе с параллельным включением разнотипных транзисторов при изменении переключаемого напряжения сопротивление одного из транзисторов растет, а другого падает, так что общее сопротивление ключа остается низким для всего диапазона переключаемых напряжений (т. е. напряжений на истоках и стоках транзисторов). При противоположных значениях напряжений на затворах транзисторов оба транзистора запираются, и ключ приводится в разомкнутое состояние.

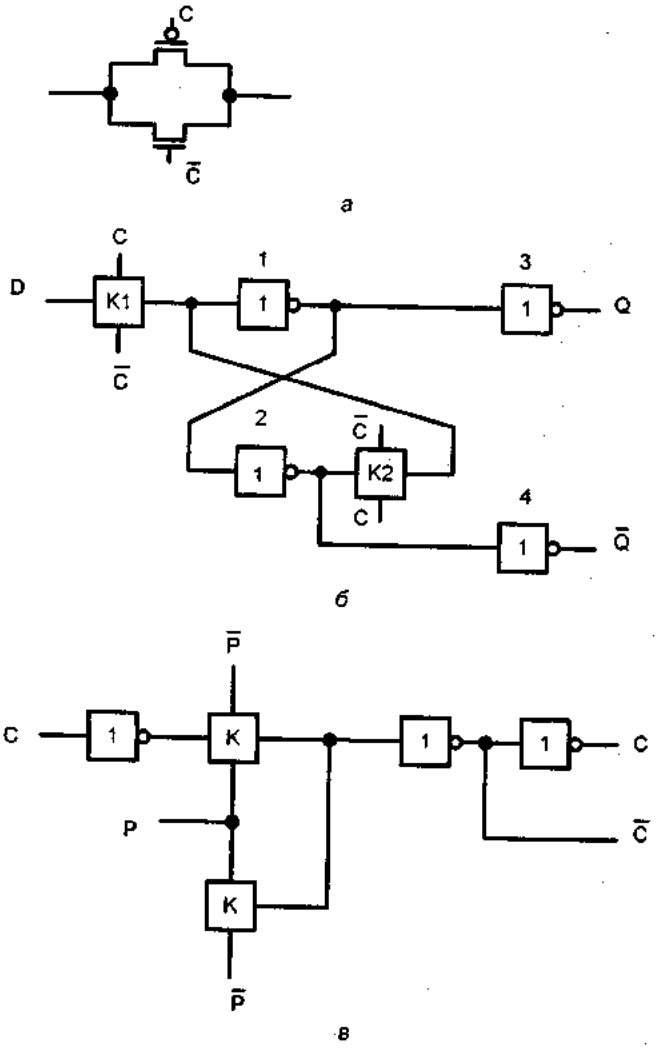


Рис. 3.11. Двунаправленный ключ (*а*), триггер-защелка (*б*) и триггер с выбором полярности тактирующего сигнала (*в*), реализованные по КМОП-схемотехнике

Схема триггера типа D с управлением уровнем (защелки) показана на рис. 3.11, *б*. В схему входят ключи K1 и K2, которые всегда находятся в противоположных состояниях. Когда ключ K1 разомкнут, а ключ K2 замкнут, схема представляет собою фиксатор на инверторах 1 и 2 с перекрестной обратной связью, выходные сигналы с которого снимаются через инверторы 3 и 4. Это состояние схемы соответствует режиму хранения. Когда ключ K2

разомкнут, а ключ K1 замкнут, то цепочка последовательно соединенных инверторов 1 и 2 подключается к входному напряжению так, что обеспечивается выработка выходных напряжений $Q = D$ и $\overline{Q} = \overline{D}$ (режим прозрачности триггера). При изменении тактирующих напряжений C и $\overline{C}$ на противоположные триггер переходит в режим хранения, "защелкивая" в фиксаторе последнее значение входного напряжения.

Применение ключей дает возможность вводить в схемы тактируемых триггеров простые цепи переключения полярностей тактирующих сигналов, т. е. выбирать тип этих сигналов (H-активные или L-активные) с помощью дополнительного управляющего сигнала. Пример такой цепи показан на рис. 3.11, *в*. В этой схеме ключи находятся в противоположных состояниях. Когда замкнут нижний ключ, то на выходе схемы формируются показанные на рисунке сигналы. Когда замкнут верхний ключ, то формируются сигналы противоположной полярности.

С помощью ключей легко реализуются схемы с третьим состоянием выхода, для чего достаточно включить в цепь выходного сигнала ключ, управляемый сигналом разрешения OE. Легко строятся также двухступенчатые триггеры, в которых для получения антисинхронного управления ступенями перед входной и выходной степенями достаточно ввести противофазно управляемые ключи.

Триггеры с динамическим управлением воспринимают информационные сигналы только в момент перепада тактового сигнала (точнее, в окрестностях этого момента).

Возможности построения таких триггеров удобно показать на примере так называемого шестиэлементного триггера (другое название — схема "трех триггеров"), показанного на рис. 3.12. Часть схемы, включающая в себя элементы 2, 3, 5, 6 без цепей перекрестных связей между элементами 2 и 3, образует синхронный RS-триггер с управлением уровнем (см. рис. 3.10), чувствительный к изменению информационных сигналов при C = 1. Чтобы получить такую чувствительность только во время фронта сигнала C, нужно блокировать цепи подачи входных сигналов сразу же после изменения синхросигнала с нулевого значения на единичное. Для достижения этого в схеме (рис. 3.12) входные сигналы подаются через элементы 1 и 4, которые и будут блокироваться в указанные моменты времени и сохранять блокировку до возвращения C к нулевому уровню. Нулевое значение C устанавливает единицы на выходах элементов 2 и 3 и приводит фиксатор в режим хранения до нового изменения синхросигнала от нуля к единице. В этом состоянии (при C = 0) выходы элементов 1 и 4 дают инверсии входных сигналов, передавая на элементы 2 и 3 значения S и R соответственно.

Что произойдет при поступлении C = 1? Если при этом S = R = 0, то сохранится режим хранения. Если же имеется единичный входной сигнал, то на входе одного из элементов (2 или 3) все входы окажутся единичными, а его выход — нулевым, что даст сигнал установки выходного триггера (элементы 5 и 6) в нужное состояние и, кроме того, отключит входной сигнал, вызвавший воздей-

ствие на схему, и также предотвратит возможное воздействие на выходной триггер по его второму входу (на элемент 6).

Три указанных действия вызываются сигналами логического нуля, подаваемыми по стрелкам 1, 2 и 3. Предполагается, что единичное значение имел вход S ($\overline{S} = 0$).

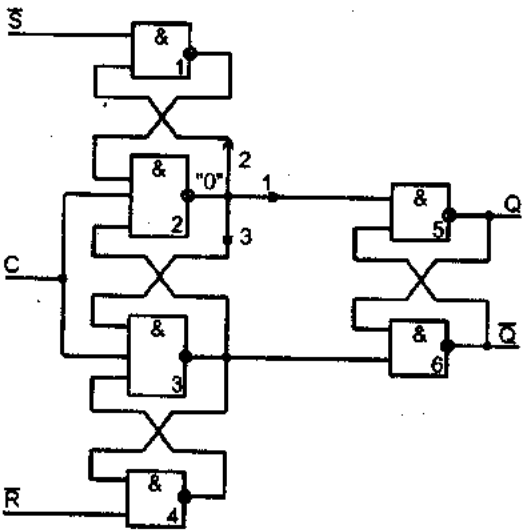


Рис. 3.12. Схема триггера с управлением фронтом

На основе рассмотренного шестиэлементного триггера строятся триггеры типов D, T и JK, свободные от некоторых недостатков, присущих данному RS-триггеру. Способы построения соответствуют схемам информационных связей (см. рис. 3.7, 3.8).

В *двухступенчатых триггерах* входная и выходная ступени тактируются "антисинхронно", прием информации в них разрешается поочередно. Следствие этого — отсутствие режима прозрачности триггера при любом уровне синхросигнала, что позволяет реализовать любые типы триггеров, свободные от режимов генерации, и дает возможность построения синхронных автоматов без опасных временных состязаний. В то же время схемы этих триггеров более сложные, чем схемы триггеров с динамическим входом, а их быстродействие несколько ниже.

Двухступенчатые триггеры строятся несколькими способами: *с разнополярным управлением ступенями* (рис. 3.13), *с инвертором* (рис. 3.14), *с запрещающими связями*.

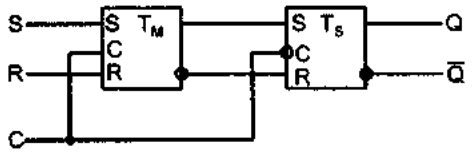


Рис. 3.13 Схема двухступенчатого триггера с разнополярным управлением

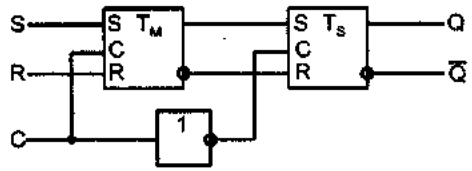


Рис. 3.14. Схема двухступенчатого триггера с инвертором

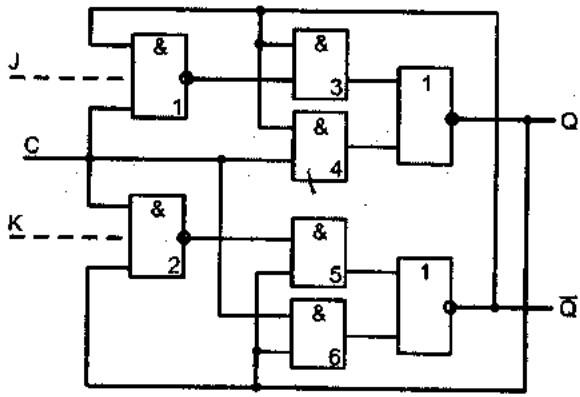
В первом варианте антисинхронное тактирование ступеней очевидно, поскольку ступени имеют соответствующие синхровходы. Во втором варианте ступени идентичны по синхровходам, а для их антисинхронного управления в цепь тактовых сигналов включен инвертор. В такой схеме возможны временные состязания сигналов: входной триггер состязается с инвертором. Если триггер переключится быстрее инвертора, то его новое состояние может успеть "проскочить" в выходной триггер, т. к. инвертор не успеет блокировать входы этого триггера. Несмотря на это, вариант с инвертором находит широкое применение, при его проектировании просто заботятся об обеспечении нужного соотношения задержек инвертора и входного триггера.

В связи с неоднозначностью трактовки функционирования двухступенчатых триггеров в литературе, уточним некоторые принятые здесь положения. Разрешающим уровнем тактового сигнала будем считать тот, который переносит информацию из входной ступени в выходную, т. к. именно при этом новая информация появляется на выходе триггера. Тип управления триггером (уровнем или фронтом) нужно определять с учетом конкретной схемы. Важнейшим качеством триггера с управлением фронтом (динамическим) является допустимость смены информационных сигналов при любом уровне тактового сигнала. Старые разновидности двухступенчатых триггеров из-за явлений "захвата единицы" и "захвата нуля" [33] таким свойством не обладали и не могли быть отнесены к триггерам с динамическим управлением. Новые разновидности свободны от явлений "захватов" и проявляют себя по существу как триггеры, управляемые фронтом.

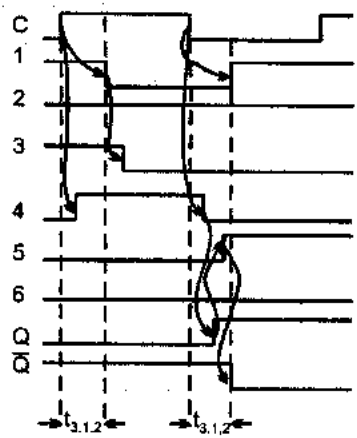
Двухступенчатые триггеры строятся также по схеме "с запрещающими связями", не имеющей инвертора в цепи подачи синхросигналов на вторую ступень.

Сигналы блокировки второй ступени берутся в этом случае со входов фиксатора первой ступени.

В последнее время преимущественное применение среди JK-триггеров получили одноступенчатые с внутренними задержками и шестиэлементные с управлением фронтом. Принцип работы шестиэлементных триггеров рассмотрен ранее. Функционирование одноступенчатого триггера с внутренней задержкой (рис. 3.15, *а*) можно рассмотреть с помощью временных диаграмм (рис. 3.15, *б*).



а



б

Рис. 3.15. Схема JK-триггера с внутренними задержками (а) и временные диаграммы ее работы (б)

Рассмотрим только счетный режим, т. к. работа входов сброса и установки проблем не создает. Так как в счетном режиме J = K = 1, соответствующие входы не влияют на работу элементов 1 и 2 и показаны штриховыми линиями. Исходное состояние триггера примем нулевым. Поскольку схема симметрична, достаточно рассмотреть только один процесс переключения (из нуля в единицу).

Работоспособность триггера обеспечивается только при условии $t_{з.1,2} > t_{и} + T_{или\text{-}не}$ (задержки вентилей 1 и 2 превышают суммарную задержку вентилей И и ИЛИ-НЕ), которое и отражено на временных диаграммах. Как видно из диаграмм, триггер переключается по отрицательному перепаду тактирующего сигнала.

На рис. 3.16 показаны наиболее популярные JK-триггеры. Триггер ТВ1 — двухступенчатый, причем триггеры серий К561, К1561, 564 выполнены на ступенях типа D и свободны от захватов.

Триггеры ТВ6, ТВ9, ТВ10 и ТВ11 — с внутренними задержками, одноступенчатые, переключаемые отрицательным фронтом синхросигнала. Они имеются в сериях КР1533, К555 и др. Особенность этого типа триггеров — нулевое время выдержки $t_{Н}$, что облегчает построение некоторых узлов на основе таких триггеров.

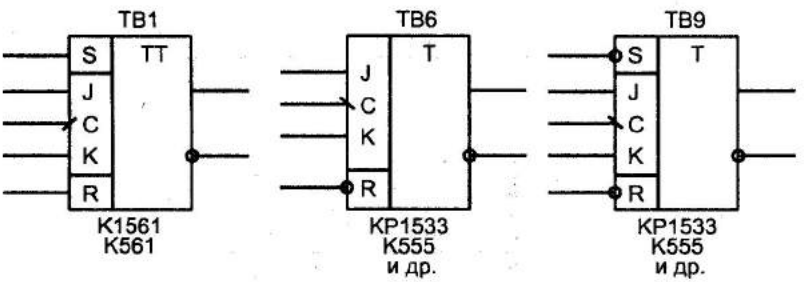


Рис. 3.16. Схемы стандартных триггеров типа JK

Асинхронные входы установки и сброса являются доминирующими, воздействие по ним осуществляется независимо от сигналов на других входах, которые при этом игнорируются.

Как следует из названия, время появления сигналов $\overline{S}$ и $\overline{R}$ может быть любым. Если эти сигналы снимаются, то обусловленное ими состояние триггера сохраняется до первого активного изменения синхросигнала, которое определит новое состояние триггера в соответствии с его информационными входами.

В справочниках функционирование триггеров чаще всего поясняется таблицей, в которой для каждого набора входных переменных и исходного со-

стояния триггера указывается его новое состояние. Функционирование триггера на примере микросхемы типа К561ТВ9 представлено табл. 3.8.

Таблица 3.8

$\bar{S}$	$\bar{R}$	C	J	K	$Q_н$	$\bar{Q}_н$	Режим работы
L	H	X	X	X	H	L	Асинхронная установка
H	L	X	X	X	L	H	Асинхронный сброс
L	L	X	X	X	X	X	Запрещенная комбинация
H	H	↓	H	H	$\bar{Q}$	Q	Счетный
H	H	↓	L	H	L	H	Загрузка нуля (сброс)
H	H	↓	H	L	H	L	Загрузка единицы (установка)
H	H	↓	L	L	Q	$\bar{Q}$	Хранение
H	H	H	X	X	Q	$\bar{Q}$	Хранение
H	H	L	X	X	Q	$\bar{Q}$	Хранение

Символ ↓ означает отрицательный перепад синхросигнала.

Для ориентировки в табл. 3.9 даны некоторые данные о быстродействии и потребляемой мощности для микросхем триггеров JKRS различных схемотехнологий (ТТЛШ, КМОП и ЭСЛ).

Таблица 3.9

Тип ИС	Схемотехнология	Число триггеров	Средняя задержка, нс	Максимальная частота, МГц	Потребляемая мощность, мВт	
К555ТВ9	ТТЛШ	2	15	30	40	
К531ТВ9	ТТЛШ	2	6,2	80	250	
К500ТВ135	ЭСЛ	2	3	170	290	
К1533ТВ15	ТТЛШ	2	12	40	24	
К561ТВ1	КМОП	2	170	3	0,02	статическая
К1554ТВ9	КМОП	2	10	140	0,035	статическая

На рис. 3.17 и рис. 3.18 показаны временные диаграммы, иллюстрирующие реакцию триггеров D разных типов и триггера RSD на показанные входные сигналы. Диаграммы могут служить полезным пособием для закрепления

материала этого параграфа по способам записи информации в триггеры разных типов.

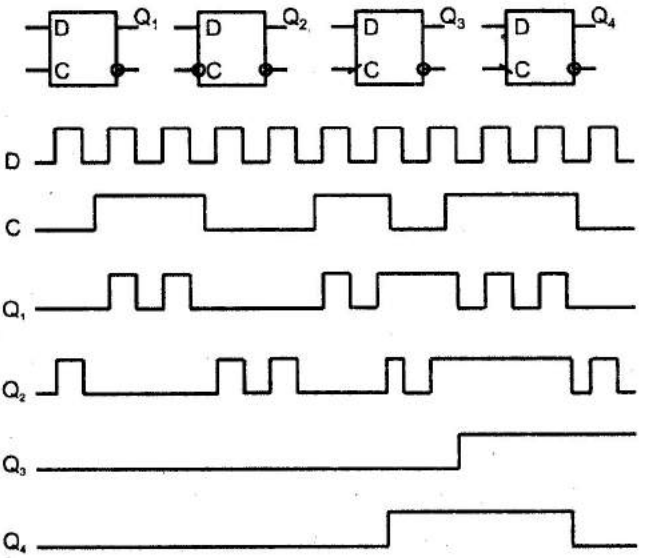


Рис. 3.17. Иллюстрация к работе триггеров типа D

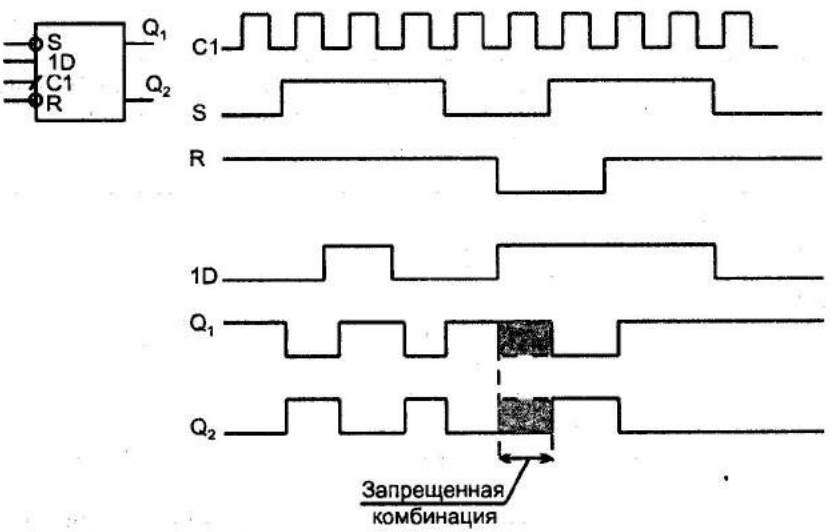


Рис. 3.18. Иллюстрация к работе триггера типа RSD

§ 3.3. Аномальные состояния триггеров

При использовании триггеров приходится сталкиваться с проблемой аномальных состояний. Триггер быстро принимает одно из своих устойчивых состояний при достаточно определенном воздействии на него. Чтобы избежать неопределенностей, для синхронных триггеров вводятся запретные зоны, в которых информационные сигналы не должны изменяться — времена предустановки и выдержки. Ясно, что при приеме по информационным входам асинхронных сигналов, появляющихся в произвольные моменты времени, соблюдать требования по временам предустановки и выдержки невозможно, и триггер может попадать под неопределенные воздействия. Например, триггер D может получить сигнал переключения по информационному входу одновременно с переходом синхросигнала в состояние запрета приема информации. Процесс переключения может начаться, но затем прекратиться в некотором промежуточном состоянии, т. к. синхросигнал отключит триггер от информационных входов. Триггер, предоставленный самому себе, рано или поздно перейдет в одно из устойчивых состояний (вернется в исходное состояние или перейдет в противоположное). Однако если его оставить в точке, очень близкой к равновесию, то выход из нее окажется аномально длительным, и триггер надолго "зависнет" в промежуточном состоянии. Аномалии разделяются *на метастабильные и колебательные*. В первом случае напряжения на обоих выходах триггера близки к пороговым напряжениям логических элементов, из которых собран триггер. Эти напряжения сохраняются почти неизменными в течение всего времени действия аномалии. Во втором случае выходные напряжения триггера медленно колеблются вокруг пороговых напряжений элементов.

Аномальные состояния — неустранимые явления, объясняющие неизбежность сбоев при работе с асинхронными сигналами. Следует лишь принимать меры для снижения частоты возникновения аномальных состояний и доведения уровня сбоев до минимальных значений.

§ 3.4. Применение триггеров в схемах ввода и синхронизации логических сигналов и в генераторах синхропоследовательностей

Ввод логических сигналов от механических ключей

Ввод логических сигналов от механических ключей — одно из типовых действий, позволяющее оператору воздействовать на цифровое устройство.

Механические ключи имеют упругость, их коммутация — сложный процесс. После первого соударения контактов происходит ряд упругих отскоков, называемых дребезгом контактов, поэтому вместо однократного перепада напряжения ключи создают целую серию импульсов (рис. 3.19, *а*).

Длительность упругих колебаний ключей зависит от их конструкции, обычно она лежит в диапазоне 1—10 мс. Такой сигнал нельзя вводить в цифровое устройство, т. к. он может создать множество ложных переключений.

Для получения "очищенного" от дребезга контактов сигнала принимают специальные меры — программные или схемные. Программные методы вводят паузу между каждым нажатием ключа и использованием формируемого ключом сигнала (серия пустых команд NOP в программе, выполняемой системой). В схемных методах борьбы с дребезгом контактов используются свойства триггеров.

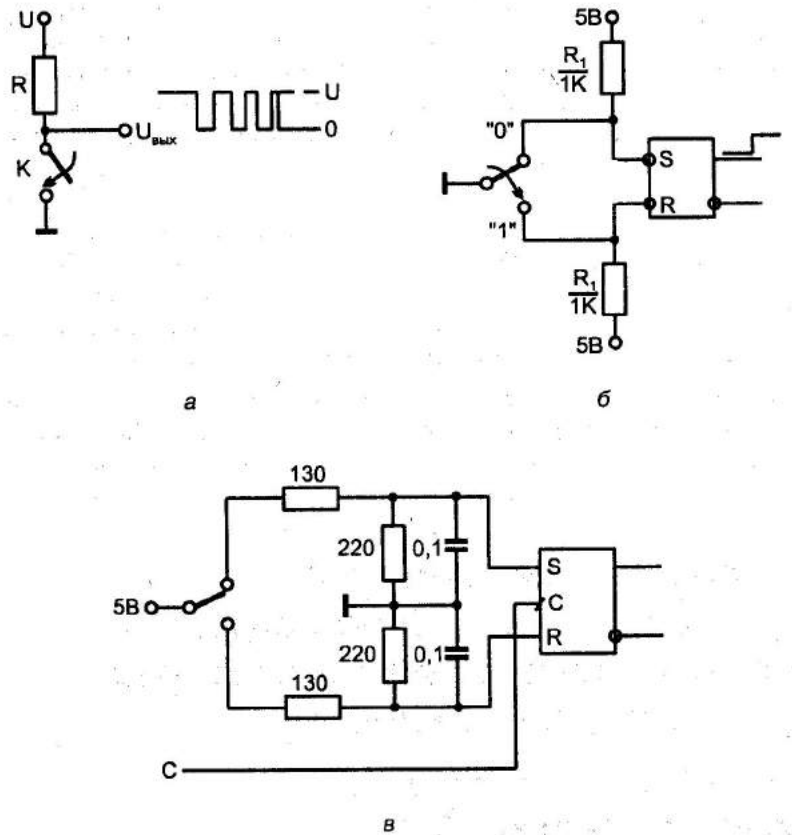


Рис. 3.19. Сигнал, формируемый механическим ключом (*а*), и схемы устранения дребезга контактов (*б*, *в*)

С помощью триггеров выходное напряжение ключа очищается от паразитных колебаний и превращается в стандартный логический сигнал. Для работы с перекидными ключами (однополюсными ключами на два положения) часто используется схема (рис. 3.19, *б*), в которой верхнее положение ключа устанавливает триггер (на входе $\overline{S}$ нулевое напряжение, на входе $\overline{R}$ — высокое, задаваемое от источника 5В через резистор R_1; номиналы напряжений и сопротивлений приводятся здесь с ориентацией на схемотехнику ТТЛ). Нижнее положение ключа ведет к сбросу триггера (на входе $\overline{R}$ нулевое напряжение, на входе $\overline{S}$ — высокое). При изменении состояния ключа возникают упругие отскоки от контактов. Первое же соударение приводит триггер в соответствующее состояние, а при отскоке ключа, когда он находится в воздухе, оба входа триггера получают пассивные сигналы логической единицы (высокие напряжения от цепочек "источник-резистор"), т. е. триггер попадает в режим хранения уже установленного правильного состояния. С помощью схемы (рис. 3.19, *б*) производится асинхронный ввод сигнала от механического ключа. Резисторы R_1 могут быть достаточно высокоомными, т. к. через них замыкается только относительно малый ток входной цепи триггера при единичном сигнале на нем $I_{вх.1}$.

Синхронизированный с тактовыми сигналами ввод (рис. 3.19, *в*) осуществляется с помощью тактирования триггера. Видоизменения схемы в сравнении с предыдущей объясняются тем, что синхронный триггер имеет прямые входы установки и сброса, и тем, что для повышения помехоустойчивости схемы добавлены конденсаторы. Принцип работы схемы сохраняется. Первый же тактовый импульс, пришедший после переключения ключа, формирует выходной сигнал. Резисторы низкоомны, поскольку через них замыкаются входные токи триггера при обоих значениях логического сигнала на них, в том числе и значительные по величине токи $I_{вх.0}$. Отношение сопротивлений плеч делителей обеспечивает подачу на входы триггера необходимых уровней U_1.

Иногда сигналы от механических ключей вводят с помощью более простых схем, содержащих интегрирующие RC-цепочки. В этом случае переходный процесс при идеальном замыкании ключа имел бы форму экспоненты, а в реальной ситуации эта экспонента будет с горизонтальными участками, соответствующими отскокам ключа, когда он находится в воздухе, и ток емкости нулевой. Так как подобная кривая монотонна, после достижения ею порогового значения логический элемент переключается однократно.

Синхронизаторы одиночных импульсов

Синхронизаторы одиночных импульсов вырабатывают под воздействием асинхронного входного сигнала импульс, принадлежащий тактовой последовательности ТИ. Такой импульс может понадобиться для запуска устройства, реализации пошагового режима его работы и т. д.

Привязка одиночного импульса к тактовой системе обязательна для правильного его восприятия синхронными цифровыми устройствами.

При реализации синхронизаторов следует организовать следующие процессы: разрешить прохождение очередного целого импульса ТИ на вход схемы и затем снять это разрешение после прохождения всего одного импульса. Этим требованиям соответствует структура: синхронизатор момента воздействия входного сигнала на триггер плюс сам триггер, устанавливаемый и сбрасываемый соседними фронтами ТИ (разнополярными). Простой вариант указанной структуры показан на рис. 3.20. Рассмотрение временных диаграмм его работы свидетельствует о сужении входного импульса относительно тактового на время $t_{тг.1}$. Входной сигнал по длительности должен превышать период тактовых импульсов T, иначе он может не дождаться фронта, который разрешает его прием в первый триггер, и, таким образом, остаться незамеченным.

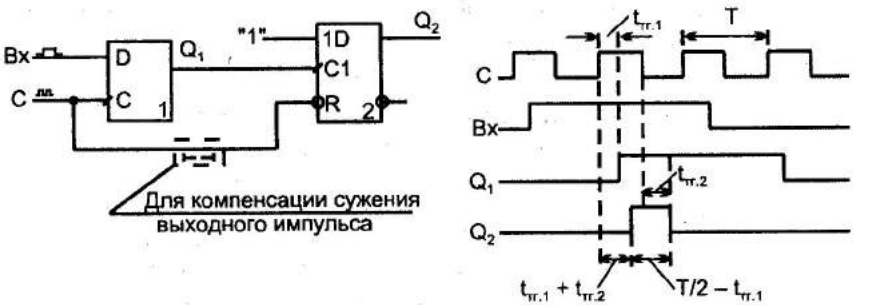


Рис. 3.20. Схема синхронизатора одиночных импульсов и временные диаграммы ее работы

Ввод асинхронных данных

При вводе асинхронных данных в систему они должны быть "привязаны" к системным тактовым сигналам, для чего данные пропускаются через синхронные D-триггеры (рис. 3.21). Известно, что запрещается изменение информационных сигналов на входе триггера в окне t_{SU}—t_H. Однако обеспечить это нет возможности, поэтому вероятности попадания триггеров в аномальные состояния не избежать. Временные диаграммы показывают ситуации с поступлением асинхронных входных данных до, после или во время тактового воздействия. В последнем случае возможно попадание в метастабильное состояние, что отмечено штриховкой. Длительность метастабильных состояний имеет вероятностный характер. Можно лишь сформулировать следующее положение: для X процентов появившихся метаста-

бильных состояний триггер с вероятностью Y выйдет из метастабильного состояния за время, равное Z нс.

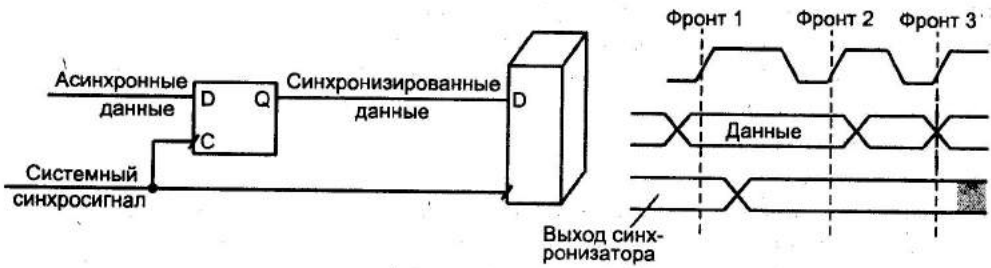


Рис. 3.21. Схема ввода асинхронных данных в цифровую систему и временные диаграммы ее работы

Формирование вторичных синхросигналов из опорной синхропоследовательности

Для управления работой синхронных цифровых устройств необходимы тактирующие импульсы.

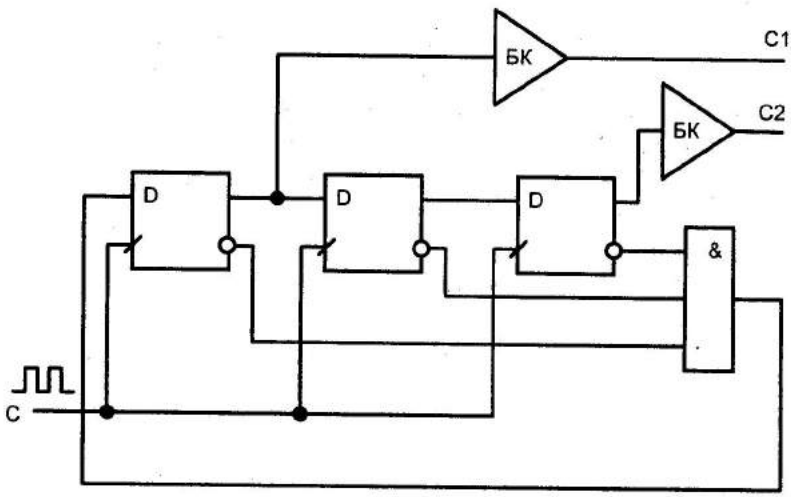


Рис. 3.22. Схема генератора вторичных синхросигналов

Выработка таких импульсов — задача, решаемая синхрогенераторами того или иного типа, для получения высокой стабильности частоты применяют генераторы с кварцевыми резонаторами. На выходе генератора, задающего

частоту синхросигналов, как правило, формируется одна синхропоследовательность с симметричными по длительности импульсом и паузой. В то же время для управления устройством могут понадобиться как несколько сдвинутых во времени синхропоследовательностей, так и несимметричные импульсы (обычно с длительностью паузы, в несколько раз превышающей длительность импульсов). Для получения таких синхросигналов (вторичных) из исходной (опорной) последовательности удобно применять схемы, содержащие триггеры.

На рис. 3.22 показана схема, вырабатывающая две синхропоследовательности с относительной длительностью импульсов 1/4 (т. е. со скважностью 4), сдвинутые относительно друг друга на 1/2 периода. Частота синхросигналов задается генератором опорной частоты (сигналом С).

§ 3.5. Введение в проблематику и методику проектирования автоматов с памятью

Узлы и устройства, которые содержат элементы памяти, относятся к классу автоматов с памятью (АП). Наличие элементов памяти (ЭП) придает АП свойство иметь некоторое внутреннее состояние Q, определяемое совокупностью состояний всех элементов памяти. В зависимости от внутреннего состояния (далее называемого просто состоянием), АП различно реагирует на один и тот же вектор входных сигналов X. Воспринимая входные сигналы при определенном состоянии, АП переходит в новое состояние и вырабатывает вектор выходных переменных Y. Таким образом, для АП $Q_н = f(Q, X)$ и $Y = \varphi(Q, X)$, где $Q_н$ и Q — состояния АП после и до подачи входных сигналов (индекс "н" от слова "новое").

Переходы АП из одного состояния в другое начинаются с некоторого исходного состояния Q_0, задание которого также является частью задания автомата. Следующее состояние зависит от Q_0 и поступивших входных сигналов X. В конечном счете, текущее состояние и выходы автомата зависят от начального состояния и всех векторов X, поступавших на автомат в предшествующих сменах входных сигналов. Таким образом, вся последовательность входных сигналов определяет последовательность состояний и выходных сигналов. Это объясняет название *"последовательностные схемы"*, также применяемое для обозначения АП.

Структурно АП отличаются от комбинационной цепи (КЦ) наличием в их схемах обратных связей, вследствие чего в них проявляются свойства запоминания состояний (полезно вспомнить схемы триггерных элементов, где указанная особенность проявляется очень наглядно).

Автоматы с памятью в каноническом представлении разделяют на две части: *память* и *комбинационную цепь*. На входы КЦ подаются входные сигналы и сигналы состояния АП. На ее выходе вырабатываются выходные сигналы и сигналы перевода АП в новое состояние.

Принципиальным является деление АП на *асинхронные* и *синхронные*. В асинхронных (рис. 3.23, *а*) роль элементов памяти играют элементы задержки, через которые сигналы состояния передаются на входы КЦ, чтобы совместно с новым набором входных переменных определить следующую пару значений Y и Q на выходе. Элементы АП переключаются здесь под непосредственным воздействием изменений информационных сигналов. Скорость распространения процесса переключений в цепях асинхронного автомата определяется собственными задержками элементов.

В синхронном АП (рис. 3.23, *б*) имеются специальные синхросигналы (тактирующие импульсы) С, которые разрешают элементам памяти прием данных только в определенные моменты времени. Элементами памяти служат синхронные триггеры. Процесс обработки информации упорядочивается во времени, и в течение одного такта возможно распространение процесса переключения только в строго определенных пределах тракта обработки информации.

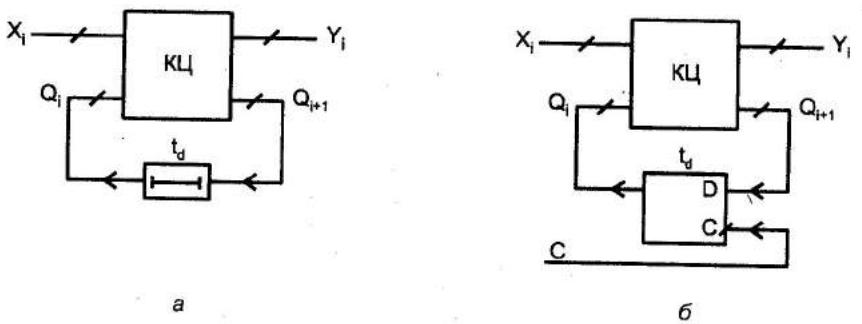


Рис. 3.23. Асинхронный (*а*) и синхронный (*б*) автоматы с памятью

Практическое применение асинхронных автоматов существенно затруднено сильным влиянием на их работу задержек сигналов в цепях АП, создающих статические и динамические риски, гонки элементов памяти (неодновременность срабатывания ЭП даже при одновременной подаче на них входных сигналов) и др. В итоге характерным свойством асинхронного автомата является то, что при переходе из одного устойчивого состояния в другое он обычно проходит через промежуточные нестабильные состояния. Нельзя сказать, что методы борьбы с нежелательными последствиями рисков и гонок в асинхронных АП отсутствуют, но все же обеспечение предсказуемого поведения АП — сложная проблема. В более или менее сложных АП асин-

хронные схемы встречаются очень редко, а в простейших схемах применяются. Примером могут служить асинхронные RS-триггеры.

В синхронных автоматах каждое состояние устойчиво и переходные временные состояния не возникают. Концепция борьбы с последствиями рисков и гонок в синхронных автоматах проста — прием информации в элементы памяти разрешается только после завершения в схеме переходных процессов. Это обеспечивается параметрами синхроимпульсов, задающих интервалы времени для завершения тех или иных процессов. В сравнении с асинхронными, синхронные АП значительно проще в проектировании.

На сегодняшний день и достаточно длительную перспективу основным путем построения АП следует считать применение тактирования, т. е. синхронных автоматов.

Отечественными и зарубежными учеными разрабатывается направление, называемое проектированием самосинхронизирующихся устройств, в которых тактовые импульсы следуют с переменной частотой, зависящей от длительности реального переходного процесса в схеме. Однако перспективность этого направления еще не вполне ясна.

В теории автоматов проводится их классификация по ряду признаков. Не вдаваясь в подробности, отметим, что в схемотехнике преобладают автоматы Мура, выходы которых являются функциями только состояния автомата. Для этого автомата $Q_н = f(Q, X)$ и $Y = \varphi(Q)$.

Зависимость выходов и от состояния автомата, и от вектора входных переменных свойственна автоматам Мили.

Некоторые функциональные узлы принадлежат к числу *автономных автоматов*, которые не имеют информационных входов и под действием тактовых сигналов переходят из состояния в состояние по алгоритму, определяемому структурой автомата.

Проектирование автоматов

Проектирование АП содержит следующие этапы:

- исходное задание функционирования;
- формализованное задание функционирования;
- минимизация состояний;
- кодирование состояний;
- составление таблицы переходов;
- определение функций возбуждения элементов памяти (триггеров);
- минимизация функций возбуждения триггеров;
- переход к базису выбранной для реализации схемотехнологии;

- составление логической схемы;
- сборка и проверка автомата.

Исходное задание функционирования может иметь различную форму, в том числе и словесную. От нее переходят к формализованному заданию — таблицам, формулам, диаграммам состояния и т. п. Далее выполняются минимизация и кодирование состояний автомата, в результате чего получается таблица переходов, на основании которой можно найти функции возбуждения триггеров.

Минимизация и кодирование состояний в общем случае задача, решение которой может потребовать значительных усилий, однако при проектировании узлов ЭВМ и цифровой автоматики она чаще всего проста, и ее решение подсказывается самой формулировкой задания на проектирование. Традиционно *широко применяется кодирование состояний автомата двоичными кодами*, при котором триггеры используются в схеме экономно. Для некоторых новых СБИС программируемой логики, снабженных большим числом триггеров, экономия их числа при построении автомата несущественна. Для таких случаев применение кодирования кодами "1 из N" может быть предпочтительным, т. к. приводит к более быстродействующим схемам, хотя и требующим значительного числа триггеров.

Функции возбуждения триггеров, обеспечивающие переходы АП из одного состояния в другое, реализуются его комбинационной частью. Они, как сказано в перечислении этапов проектирования, минимизируются и переводятся в базис выбранных средств реализации автомата. Это положение следует понимать в широком смысле, поскольку в зависимости от средств реализации КЦ требования к формам представления функций возбуждения могут существенно различаться *(см. § 2.1)*. Точнее, можно говорить о приведении функций к виду, удобному для воспроизведения данными средствами.

После выполнения указанных действий можно получить логическую схему АП. Заканчивается процесс проверкой работы узла с помощью моделирования или макетирования.

Рассмотрим более подробно *методику проектирования автоматов, содержащих триггеры* (рис. 3.24).

В тактируемых автоматах элементами памяти служат синхронные триггеры, причем любой автомат можно построить на любом типе триггера (D, JK, RS, T и др.).

При двоичном кодировании состояний автомата число триггеров в его схеме равно $n = \lceil \log_2 N \rceil$, где N — число состояний автомата и $\lceil\ \rceil$ — знак округления до ближайшего справа целого числа.

При кодировании кодом "1 из N" число триггеров равно числу состояний автомата: n = N, т. к. каждому состоянию соответствует один триггер в единичном состоянии при нулевом состоянии остальных.

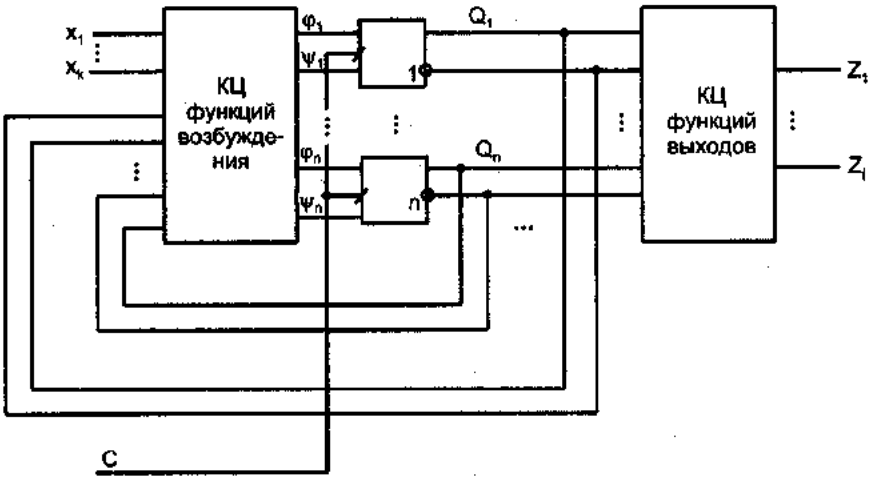


Рис. 3.24. Структурная схема автомата Мура

Будем считать, что закон функционирования автомата определен и кодирование его состояний произведено. Значит, известна последовательность состояний триггеров, принимаемых ими в каждом такте под управлением входных сигналов x_1, x_2, ..., x_k и текущего состояния Q_1, Q_2, ..., Q_n. Предмет синтеза — получение функций возбуждения φ_i и Ψ_j, для каждого входа всех триггеров, обеспечивающих необходимые переходы автомата.

Функции выхода для автоматов Мура зависят только от состояния автомата, поэтому нахождение выходов Z_1, Z_2, ..., Z_j осуществляется комбинационной схемой, на которую подаются только выходы триггеров Q_1, Q_2, ..., Q_n. Все триггеры тактируются общим синхросигналом C. После завершения выработки функций возбуждения комбинационной схемой поступает очередной тактовый сигнал C, переводящий триггеры в новое состояние.

Вид функций возбуждения зависит от логического типа триггеров. Поэтому одним из средств синтеза служат "словари" для триггеров *(см. § 3.1)*.

При поиске функций возбуждения триггеров вначале составляется таблица (табл. 3.10), содержащая следующие данные.

Таблица 3.10

Входы в момент времени t				Состояния триггеров								Необходимые сигналы на всех входах каждого триггера				
				Q (старое)				Q_H (новое)								
x_1	x_1	...	x_k	Q_1	Q_2	...	Q_n	Q_1	Q_2	...	Q_n	φ_1	Ψ_1	...	φ_n	Ψ_n

Столбцы φ_1, Ψ_1, ..., φ_n, Ψ_n определяют функции возбуждения триггеров.

Многовариантность реализаций автомата связана с выбором типа триггеров и комбинационной части.

Относительно наиболее распространенных типов триггеров JK и D можно отметить следующее. Триггер типа JK обладает более развитыми логическими функциями, поэтому для него функции возбуждения в среднем более просты, но число их вдвое больше, чем для триггера D. Что же даст более простое решение, заранее неизвестно.

Комбинационная часть автомата может быть построена на логических элементах, мультиплексорах, ИС программируемой памяти, программируемых логических матрицах и т. д.

Состояния автомата можно кодировать двоичными кодами, кодами "1 из N" и др.

Автомат можно построить, приспособив к необходимому функционированию типовую ИС среднего уровня интеграции (счетчик, сдвигающий регистр), добавив к ним специально спроектированную логическую часть.

Реализации автомата с использованием ИС памяти, программируемых логических матриц и тому подобных устройств поясняются далее (после рассмотрения соответствующих средств). Далее следуют примеры построения автоматов на уже изученных типовых элементах при кодировании состояний двоичными кодами и кодами "1 из N".

Пример проектирования

Пусть необходимо спроектировать автомат с двумя режимами работы, управляемый входным сигналом M. При M = 0 автомат работает как двоичный счетчик с модулем счета 8, при M = 1 как счетчик в коде Грея.

Примечание

Код Грея используется в системах контроля ЦУ, преобразователях механических перемещений в цифровой код и т. д. При переходе от предыдущей кодовой комбинации к следующей в коде Грея изменяется только один разряд. Первые восемь комбинаций кода Грея представлены в табл. 3.11.

Таблица 3.11

Десятичная цифра	Код Грея			Десятичная цифра	Код Грея		
0	0	0	0	4	1	1	0
1	0	0	1	5	1	1	1
2	0	1	1	6	1	0	1
3	0	1	0	7	1	0	0

Кодирование состояний автомата, являющегося автоматом Мура, определяется здесь самой постановкой задачи. Диаграмма состояний автомата показана на рис. 3.25. Изменение управляющего сигнала M сразу же ведет к изменению режима, т. е. следующее состояние будет принадлежать уже другому коду.

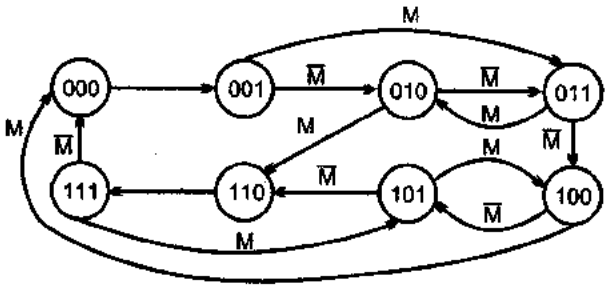


Рис. 3.25. Диаграмма состояний автомата для примера его проектирования

Вариант 1

Автомат, построенный на триггерах JK и логических элементах И-НЕ.

Таблица переходов автомата (табл. 3.12), соответствует диаграмме его состояний.

Таблица 3.12

Входной управляющий сигнал	Исходное состояние			Новое состояние			Входной управляющий сигнал	Исходное состояние			Новое состояние		
M	Q_2	Q_1	Q_0	$Q_{2н}$,	$Q_{1н}$,	$Q_{0н}$	M	Q_2	Q_1	Q_0	$Q_{2н}$,	$Q_{1н}$,	$Q_{0н}$
0	0	0	0	0	0	1	1	0	0	0	0	0	1
0	0	0	1	0	1	0	1	0	0	1	0	1	1
0	0	1	0	0	1	1	1	0	1	0	1	1	0
0	0	1	1	1	0	0	1	0	1	1	0	1	0
0	1	0	0	1	0	1	1	1	0	0	0	0	0
0	1	0	1	1	1	0	1	1	0	1	1	0	0
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	0	0	0	1	1	1	1	1	0	1

Синтез функций возбуждения для автоматов с триггерами JK имеет интересную особенность. Они могут быть получены не только указанным ранее путем, а и без поиска в таблицах функций J и K, столбцы которых можно в этом случае исключить.

Из данных о функционировании автомата можно получить функцию переходов каждого триггера

$$Q_{нi} = F(x_1, x_2, ..., x_k, Q_1, Q_2, ..., Q_n).$$

Эту функцию можно разложить следующим образом

$$Q_{нi} = f_i\overline{Q}_i \vee g_i Q_i, \qquad (3.1)$$

где функции f и g уже не содержат, соответственно, переменных Q_i и $\overline{Q}_i$.

Характеристическое уравнение триггера типа JK имеет вид:

$$Q_{нi} = J_i\overline{Q}_i \vee \overline{K}_i Q_i, \qquad (3.2)$$

Сопоставляя выражения (3.1) и (3.2), получим $f_i = J_i$, $\overline{g}_i = K_i$.

Следовательно, положив в функции переходов триггера $Q_i = 0$, сразу получаем функцию возбуждения для входа J:

$$Q_{нi}\Big|_{Q_i = 0} = f_i = J_i,$$

а приняв условие Q = 1, можно получить функцию возбуждения для входа K_i:

$$Q_{нi}\Big|_{Q_i = 1} = g_i = \overline{K}_i, \text{ т. е. } \overline{K}_i = g_i.$$

При этом члены формул для $Q_{н.i}$, не содержащие $\overline{Q}_i$, и Q_i, преобразуются путем умножения на $Q_i \vee \overline{Q}_i = 1$ в расширенную форму, в которой все слагаемые содержат переменные $\overline{Q}_i$ или Q_i.

В результате можно получить выражения:

$$J_2 = \overline{M}Q_1Q_0 \vee MQ_1\overline{Q}_0 = \overline{\overline{\overline{M}Q_1Q_0} \cdot \overline{MQ_1\overline{Q}_0}};$$

$$K_2 = \overline{M}Q_1Q_0 \vee M\overline{Q}_1\overline{Q}_0 = \overline{\overline{\overline{M}Q_1Q_0} \cdot \overline{M\overline{Q}_1\overline{Q}_0}};$$

$$J_1 = \overline{M}Q_0 \vee \overline{Q}_2Q_0 = \overline{\overline{MQ_0} \cdot \overline{\overline{Q}_2Q_0}};$$

$$K_1 = \overline{M}Q_0 \vee Q_2Q_0 = \overline{\overline{MQ_0} \cdot \overline{Q_2Q_0}};$$

$$J_0 = \overline{M} \vee \overline{Q}_2\overline{Q}_1 \vee Q_2Q_1 = \overline{M \cdot \overline{\overline{Q}_2\overline{Q}_1} \cdot \overline{Q_2Q_1}};$$

$$K_0 = \overline{M} \vee \overline{Q}_2Q_1 \vee Q_2\overline{Q}_1 = \overline{M \cdot \overline{\overline{Q}_2Q_1} \cdot \overline{Q_2\overline{Q}_1}}.$$

Схема автомата приведена на рис. 3.26.

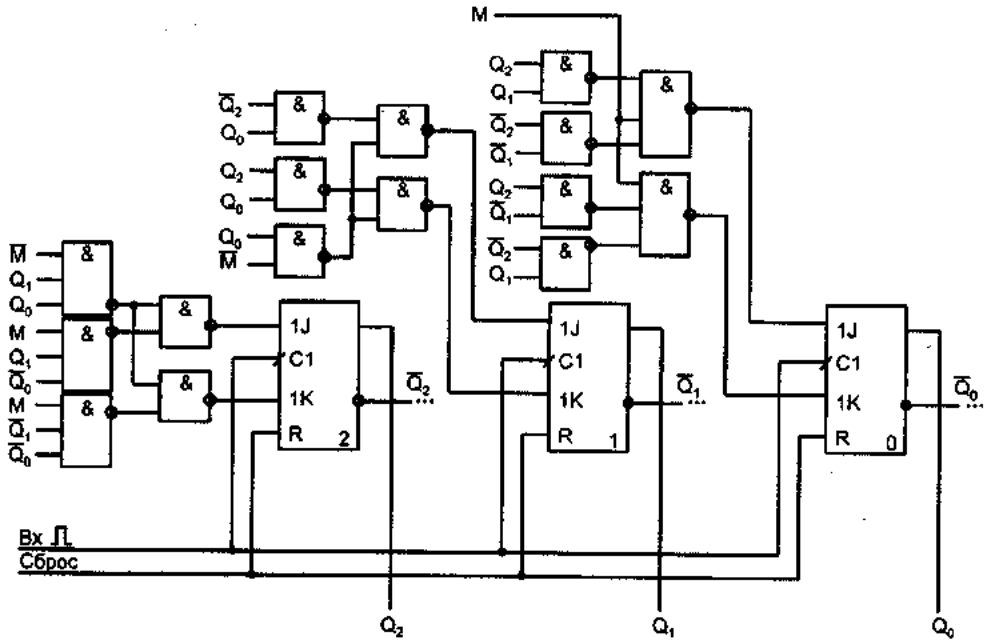


Рис. 3.26. Схема автомата на триггерах JK для примера проектирования

Вариант 2

Автомат, реализованный на триггерах с мультиплексным управлением.

Структура с мультиплексорами на входах триггеров отличается концептуальной простотой и наглядностью, для ее проектирования не требуется разработка логических преобразователей, обеспечивающих необходимые переходы автомата. Задача решается, в сущности, табличным методом. Переменные состояния, снимаемые с триггеров, и входные сигналы образуют слово, служащее для мультиплексора адресным входом. По этому адресу в каждом мультиплексоре выбирается переменная (0 или 1), необходимая для перевода триггера типа D в новое состояние. Ясно, что при этом данные для информационных входов мультиплексоров берутся прямо из таблицы переходов ($D_i = Q_{in}$). Достоинство структуры — легкость перестройки автомата на новый алгоритм работы, недостаток — быстрый рост размерности мультиплексоров с ростом числа состояний и входов автомата. Структура с мультиплексорным управлением триггерами показана на рис. 3.27. Входные сигналы $x_0...x_{m-1}$ и значения разрядов слова старого состояния $Q_0...Q_{n-1}$ образуют управляющее (адресное) входное слово мультиплексора, по которому выбираются значения разрядов нового слова состояния. Поступление тактового импульса ТИ вводит новое слово состояния в триггеры.

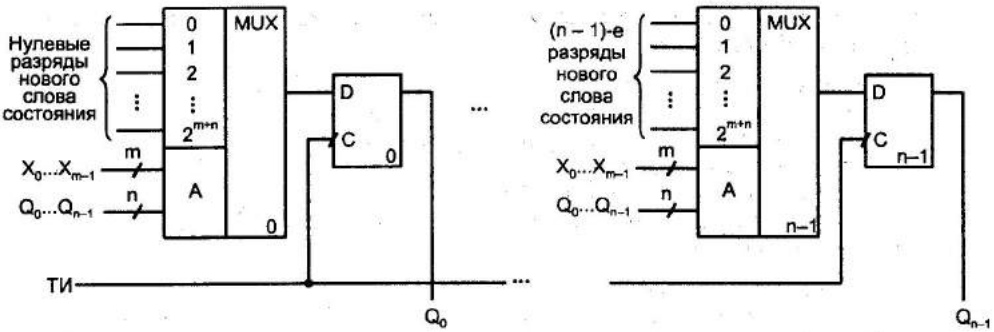


Рис. 3.27. Структура автомата на триггерах с мультиплексным управлением

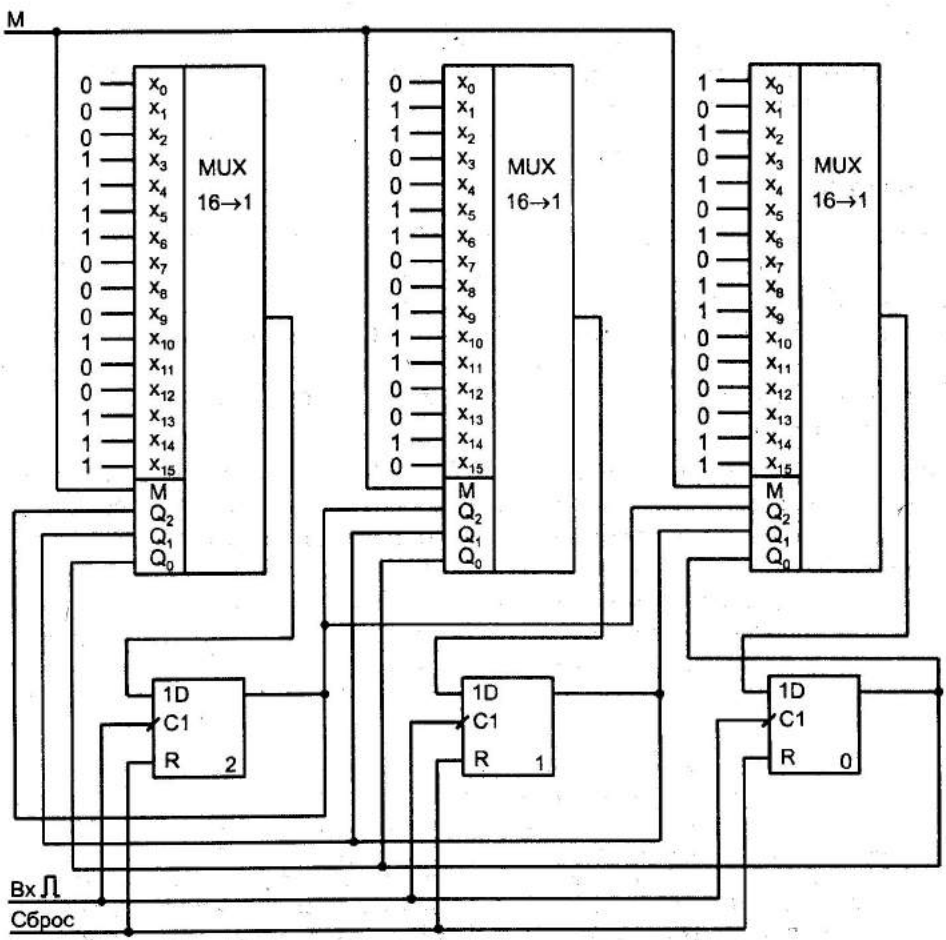


Рис. 3.28. Схема автомата є мультиплексорами на входах триггеров для примера проектирования

Конкретная реализация автомата для рассматриваемого примера (рис. 3.28) не требует особых пояснений.

При нулевых исходных состояниях триггеров и M = 0 на адресные входы мультиплексоров поступает код 0000 и на входах триггеров формируется комбинация сигналов 001. Поступление тактового импульса вводит эту комбинацию в триггеры. Теперь адресом для мультиплексоров становится комбинация 0001, по которой с них снимается комбинация 010, поступающая по разрешению следующего тактового импульса в триггеры (регистр состояния). Так реализуется режим двоичного счетчика.

Изменение управляющего сигнала M дает смену режима работы автомата. Если, например, при слове состояния 010 сигнал M становится единичным, то адрес мультиплексоров изменяется с 0010 на 1010 и с их выходов снимется комбинация 110, соответствующая следующему состоянию при работе счетчика в коде Грея.

Структура и работа автомата отличаются большой наглядностью, переход к другому алгоритму функционирования требует только смены сигналов на информационных входах мультиплексоров.

Вариант 3

Структура с кодированием типа "1 из N" содержит максимальное число триггеров, т. к. в ней для каждого состояния предусматривается специальный триггер, находящийся в активном состоянии (пусть это состояние 1) при пассивном состоянии всех остальных. Рост числа триггеров усложняет автомат, но, одновременно с этим, резко упрощается логика, обеспечивающая переходы автомата. Поэтому сложность автомата в целом может оказаться приемлемой.

Кодирование состояний автомата кодом "1 из N" (в английском языке ОНЕ, One-Hot Encoding) рекомендуется для ряда современных СБИС программируемой логики, т. к. дает простой метод построения автомата высокого быстродействия, имеющего во входных цепях триггеров мало уровней логики. Метод ОНЕ устраняет много логических схем, требуемых для декодирования состояний. Набор триггеров образует структуру типа сдвигающего регистра — узла, допускающего эффективное размещение и трассировку в топологии СБИС.

Для рассматриваемого примера автомат реализуется структурой (рис. 3.29) с числом триггеров 8. Переход в следующее состояние происходит как переход единственной единицы из триггера в соседний триггер, что осуществляется крайне просто сдвигающим регистром. В структуре должен быть шифратор для перевода кода "1 из N" в двоичный код либо в код Грея в зависимости от сигнала M. Этот сигнал выбирает один из двух шифраторов (шифраторы расположены в строке элементов ИЛИ-НЕ). При M = 0 получается двоичный код, при M = 1 — код Грея.

Автомат способен работать на высокой тактовой частоте — в цепях связи триггеров вообще нет каких-либо логических элементов.

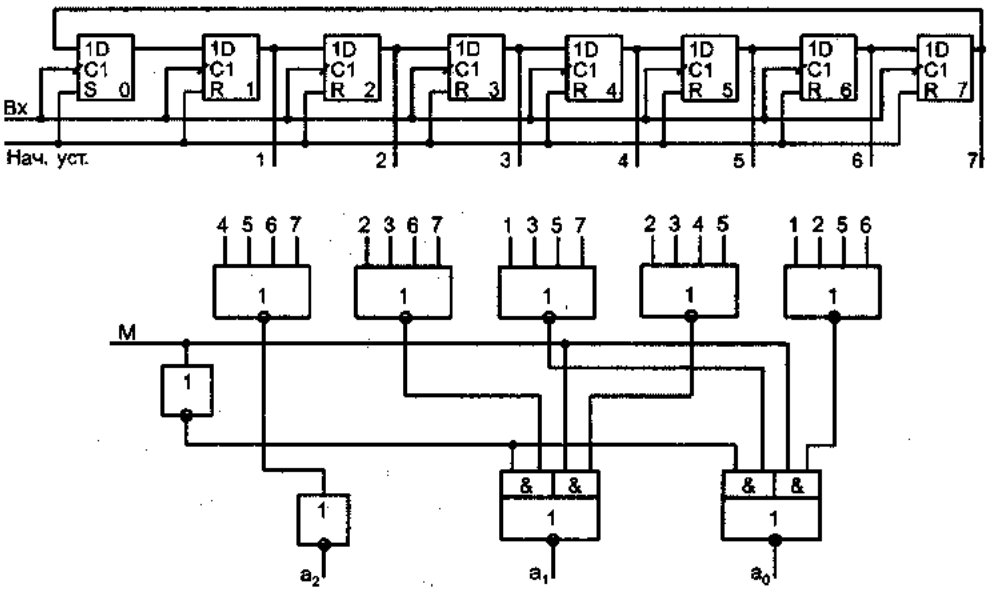


Рис. 3.29. Схема автомата с кодированием состояний в коде "1 из N" для примера проектирования

Специфическая ситуация может возникать при установке исходного состояния автомата. Если набор триггеров выполнен как единая ИС, то сброс сигналом "Нач. уст." переведет все триггеры в нулевое состояние, тогда как первый слева триггер должен получать единичное состояние. Для создания эквивалента нужной ситуации можно взять выход левого триггера с инверсного вывода, что после сброса даст на выходах регистра состояние 10000000. Чтобы не изменилось функционирование схемы, на вход левого триггера также следует подавать инвертированный сигнал (рис. 3.30).

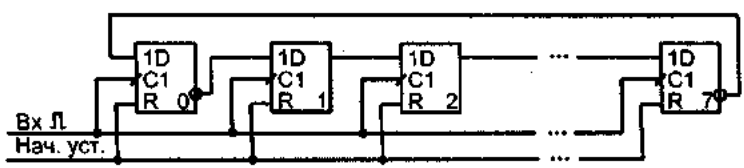


Рис. 3.30. Схема установки автомата в исходное состояние при использовании кода "1 из N"

§ 3.6. Синхронизация в цифровых устройствах

Как уже отмечалось, основным методом построения работоспособных цифровых устройств в настоящее время является синхронизация, устраняющая критические временные состязания сигналов.

Синхронизация осуществляется тактовым генератором, сигналы которого распределяются по всем частям устройства и разрешают прием данных элементам памяти — синхронным триггерам. Она упорядочивает во времени последовательность операций при обработке информации в ЦУ. Темп обработки задается частотой тактовых сигналов. Непосредственное использование одного тактового генератора для управления приемом информации во всех элементах памяти — прием, характерный для несложных систем. В сложных системах, содержащих большое число элементов памяти и/или разделенных на подсистемы, могут применяться и местные схемы тактрования для различных частей, генерирующие синхроимпульсы заданной фазности и скважности. Но и в этом случае синхросигналы общего генератора определяют ситуацию, поскольку местные синхросигналы вырабатываются из общего (опорного).

Обобщенный *тракт обработки информации* при синхронной организации процессов можно представить чередованием комбинационных цепей КЦ и элементов памяти ЭП, что отражает работу ЦУ как при пространственном чередовании КЦ и ЭП (рис. 3.31, *а*), так и при последовательном выполнении различных операций в разных временных тактах на одном и том же оборудовании (рис. 3.31, *б*).

При работе устройства КЦ преобразуют данные по тем или иным логическим зависимостям, а ЭП принимают их после окончания переходных процессов, т. е. установления на выходах КЦ истинных значений сигналов.

В КЦ пути от входов к различным выходам неидентичны. Для расчета системы синхронизации нужно оценить минимальную и максимальную задержки сигналов в КЦ. Для оценки минимальной задержки следует учесть минимальные задержки элементов (т. е. учесть разброс задержек для элементов данного типа) и найти самый короткий путь от входов к одному из выходов КЦ (короткий в смысле времени его прохождения сигналом, естественно). С учетом максимальных задержек элементов оценивается самый длинный путь сигнала к выходу КЦ. Таким образом, должны быть определены задержки $t_{кц.min}$ и $t_{кц.max}$.

Временная неидентичность путей к разным выходам КЦ затрудняет устранение критических временных состояний сигналов. С этой точки зрения одинаковость задержек для всех выходов КЦ была бы желательна.

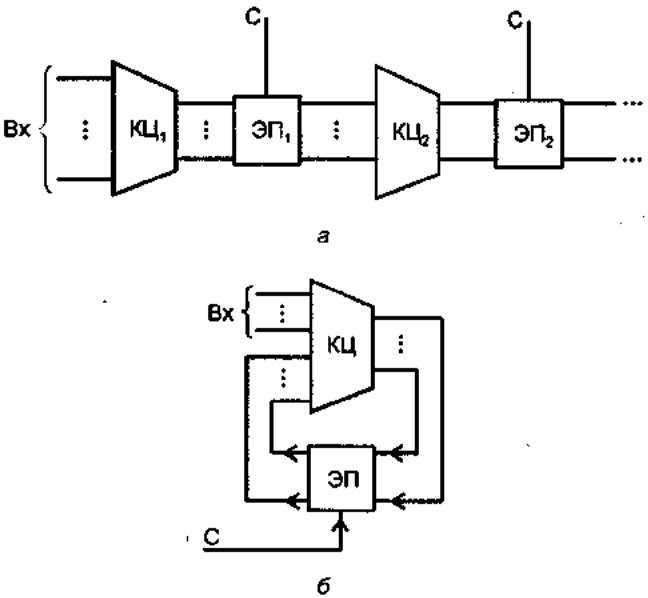


Рис. 3.31. Обобщенные структуры тракта обработки информации в цифровых устройствах (*а*, *б*)

Параметры тактовых импульсов

Период тактовых импульсов (синхроимпульсов) складывается из длительностей импульса и паузы. Длительность импульса должна быть достаточной для надежной записи информации в триггер, этот параметр задается в паспортных данных триггера. Обозначив его через t_{wc}, можем записать условие $t_и \geq t_{wc}$.

Новое состояние триггеры примут по истечении максимальной из задержек $t_з^{01}$ и $t_з^{10}$ их переключения. Параметры t_{wc} и $\max\{t_з^{01}, t_з^{10}\}$ зачастую близки, но могут и отличаться в два и более раз. Разность $\max\{t_з^{01}, t_з^{10}\} - t_{wc}$ обозначим через $\Delta t_{тг}$ (рис. 3.32).

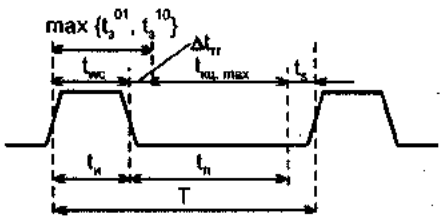


Рис. 3.32. Иллюстрация к определению параметров синхроимпульсов

Приняв новое состояние, триггеры тем самым формируют на входах КЦ новые значения сигналов. После этого до нового приема данных должно пройти время, достаточное для прохождения сигнала по самому длинному пути в КЦ плюс время предустановки t_s. Поэтому для длительности паузы имеем соотношение:

$$t_п \geq \Delta t_{тг} + t_{кц.max} + t_s.$$

Минимальный период тактовых импульсов $T_{min} = t_{и.min} + t_{п.min}$, а их частота $f_{max} = 1/T_{min}$.

На интервале от $t_{кц.min}$ до $t_{кц.max}$ после переключения триггеров выходные сигналы КЦ не соответствуют ни старому, ни новому значению (данные нестабильны).

Для многих схем, особенно для БИС/СБИС, большую роль играют задержки сигналов в линиях связи, которые следует оценивать с учетом топологии межсоединений. Поэтому на ранних стадиях проектирования расчет параметров синхросистемы может быть только ориентировочным.

В системах с постоянной тактовой частотой часто используют генераторы с кварцевой стабилизацией, позволяющие без затруднений обеспечить относительную нестабильность частоты порядка 10^{-4}—10^{-5}. В более простых генераторах нестабильность частоты существенно выше. Она, в конечном счете, приводит к потере быстродействия устройства. Действительно, частоту синхроимпульсов можно выразить соотношением: $f = f_0(1 \pm \delta f)$, где f_0 — номинальное значение частоты и $\delta f = \Delta f/f_0$ — ее относительный уход. Ширина поля допуска на частоту равна $2\delta f$. Даже максимальная частота не должна превышать допустимого значения. Если же частота будет равна нижнему пределу, то она окажется на $2\delta f$ ниже допустимой. То есть возможная потеря быстродействия устройства из-за нестабильности частоты синхроимпульсов составляет $2\delta f$.

Определенные требования предъявляются и к крутизне фронтов синхроимпульсов. Она не должна снижаться ниже допустимого предела. Причины этого ограничения заключаются в том, что при слишком пологих фронтах выходные цепи элементов могут слишком долго оставаться под действием сквозных токов и, во-вторых, то, что при малой крутизне фронтов синхроимпульсов разброс порогов срабатывания ЭП приводит к разбросу моментов их переключения. Особенно важно это обстоятельство для схем на элементах типа КМОП, для которых характерен повышенный разброс порогов срабатывания. Разброс моментов срабатывания (т. е. как бы разброс моментов поступления синхросигналов на разные элементы, питаемые одним и тем же синхросигналом), определяется выражением

$$t_2 - t_1 = (U_{пор2} - U_{пор1})/K,$$

где K — крутизна фронта синхроимпульса; $U_{пор2}$ и $U_{пор1}$ — пороговые напряжения элементов, для которых вычисляется эквивалентный сдвиг синхросигналов.

Для показа опасности сбоев из-за малой крутизны фронтов синхроимпульсов рассмотрим передачу данных в цепочке триггеров (сдвиг слова). В этой цепочке (рис. 3.33) поступление очередного синхроимпульса должно передавать состояние триггера соседу справа. Предположим, что пороговое напряжение триггера T_i минимально, а триггера T_{i+1} максимально. Тогда триггер T_i переключится раньше, чем придет сигнал приема данных для триггера T_{i+1}, и этот триггер не сможет принять старое состояние от соседа слева — информация будет утеряна.

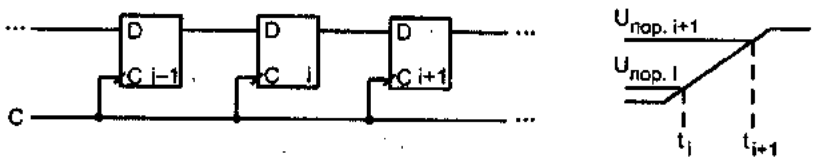


Рис. 3.33. Схема передачи данных в цепочке синхронных триггеров

Структура устройств синхронизации

Обобщенная структура устройства синхронизации (рис. 3.34) содержит следующие блоки: задающий генератор ЗГ, формирователь опорных сигналов ФОС и размножитель сигналов РС. Блок ФОС служит для выработки необходимого числа импульсных последовательностей заданной формы в зависимости от фазности системы синхронизации и временных параметров синхросигналов этих последовательностей. *Фазность* — важный признак системы синхронизации, определяемый числом синхроимпульсов в одном периоде синхронизации (иначе говоря, числом импульсных последовательностей, используемых для синхронизации устройства). Фазность зависит от типа триггеров, применяемых в устройстве, способа обмена между функциональными узлами, требований к быстродействию и аппаратурной сложности устройства.

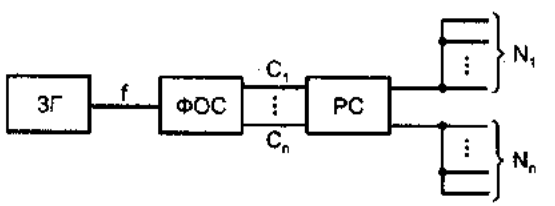


Рис. 3.34. Обобщенная структура блока синхронизации

Различают следующие системы синхронизации:

- ❒ однофазную;
- ❒ двухфазную;
- ❒ многофазную.

Размножение тактовых импульсов

Тактовыми импульсами обычно требуется обеспечить большое число элементов памяти. Тактовые импульсы исходно задаются одним генератором, а используются иногда тысячами и более элементов памяти. Попытка применить мощный генератор с разводкой от него синхросигналов по всем элементам памяти для сложных устройств оказывается, как правило, неудачной, в первую очередь из-за помех, вызываемых сильноточными цепями синхронизации.

Возможное решение — размножение тактовых импульсов с помощью разветвляющейся пирамидальной схемы (рис. 3.35), число ярусов которой зависит от числа тактируемых элементов памяти и коэффициентов разветвления задающего генератора и буферных каскадов БК. При определении числа ярусов целесообразно учитывать конструкцию устройства, ставя ярусы в соответствие каким-либо конструктивным единицам (ТЭЗам, панелям, рамам и т. п.). Такой подход типичен для традиционных конструкций, разводка синхросигналов на кристалле и применение специальных современных устройств для устранения временных сдвигов между тактирующими импульсами рассмотрены далее.

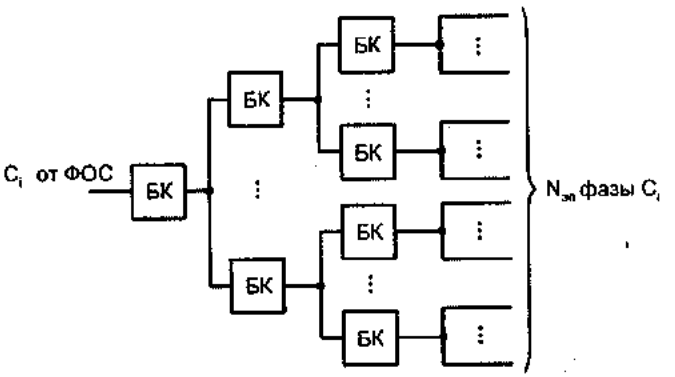


Рис. 3.35. Схема размножения тактовых импульсов

В каждом БК фронты импульсов задерживаются, причем из-за разброса задержек неодинаково. Если задержки обоих фронтов в БК идентичны, то при

прохождении БК длительность импульса не изменится, и сигналы разных выходов будут различаться лишь смещением во времени, причем максимальный сдвиг между сигналами произвольных выходов $\Delta t_{max} = m\Delta t_{бк}$, где m — число ярусов в схеме РС; $\Delta t_{бк} = (t_{бк.max} - t_{бк.min})$ — разброс задержек БК.

Временные сдвиги между синхроимпульсами, подаваемыми на различные ЭП, приводят к эффектам, равноценным сокращению одних интервалов и удлинению других. Для компенсации сокращений интервалов приходится увеличивать расчетное значение соответствующего интервала на входе схемы размножения, т. е. на выходе генератора. При этом увеличивается период синхроимпульсов и снижается быстродействие устройства. В связи с этим минимизации сдвигов уделяют большое внимание. Систему синхронизации иногда выполняют на специальных элементах повышенного быстродействия, применяют ограничение обменов данными между элементами, синхронизируемыми отдаленными выходами схемы размножения, тщательно подбирают длины соединительных проводников или вводят специальные задержки для выравнивания синхроимпульсов, используют специальные достаточно сложные устройства типа следящих систем для обеспечения синфазности синхросигналов.

Задержки синхросигналов возникают как в схемах их размножения, так и в цепях передачи.

Устройства типов PLL и DLL, улучшающие работу системы синхронизации

Проблема расфазирования тактовых импульсов в различных точках схемы для быстродействующих устройств настолько остра, что современные БИС/СБИС зачастую снабжаются специальными схемами коррекции временного положения синхросигналов, причем на одном кристалле могут быть установлены несколько таких схем, называемых в английской терминологии Phase Locked Loops (PLLs) или Delay Locked Loops (DLLs). Между PLL и DLL есть разница в технической реализации, но на них возлагаются идентичные задачи — *коррекция расфазирования синхросигналов и, при необходимости, умножение их частоты.*

Немного о терминах. Фазовый сдвиг синхросигналов называется Clock Skew, операция коррекции фазового сдвига называется функцией Clock Lock, получение умноженной частоты синхросигналов — функцией Clock Boost. Умножение тактовой частоты во внутренней области ЦУ относительно внешней частоты синхронизации часто используется в микропроцессорах и СБИС программируемой логики высокой сложности. Этот прием позволяет снизить частоту синхросигналов, передаваемых на те или иные модули, что важно, т. к. эта частота ограничена техническими возможностями линий передачи. Усложнение функций, улучшающих параметры систем синхронизации, привело к появлению блоков, называемых Clock Managers.

Благодаря введению схем PLL или DLL удается снижать расфазирование тактовых сигналов системы до очень малых значений.

Для более полного понимания роли и значения блоков PLL (DLL) рассмотрим вначале типичную схему синхронизации, использующую общий опорный синхросигнал и локальные схемы формирования синхросигналов для отдельных частей системы (рис. 3.36).

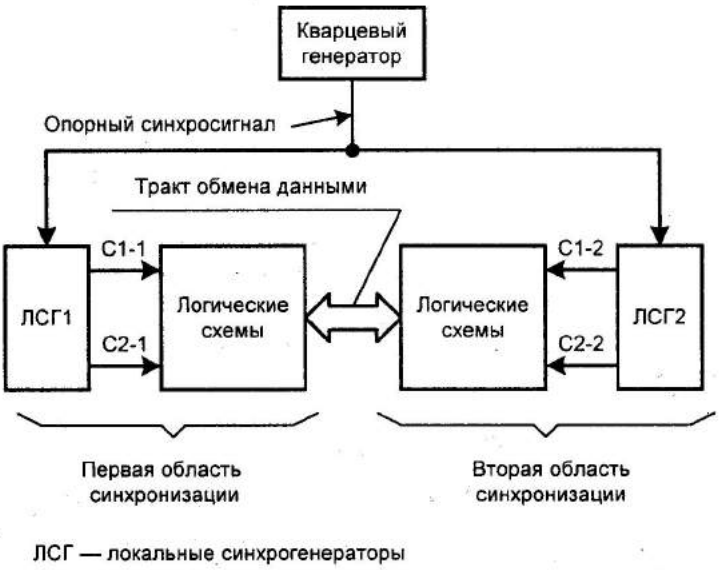


Рис. 3.36. Схема обмена данными между двумя областями синхронизации

В показанной схеме опорный синхросигнал от стабильного кварцевого генератора подается на различные области синхронизации (чаще всего различные кристаллы) по специальной ненагруженной линии *без ощутимых фазовых сдвигов*. Для синхронизации работы каждой области вырабатываются пары синхросигналов С1 и С2 с требуемой скважностью. Синхросигналы С1-1 и С1-2, а также С2-1 и С2-2 в силу ряда причин имеют заметные фазовые сдвиги относительно опорного сигнала и относительно друг друга. Эти фазовые сдвиги вносятся неидентичностью схем локальных синхрогенераторов ЛСГ1 и ЛСГ2 (в первую очередь их выходными буферами), разными задержками на входных контактах и межсоединениях. Задержки синхросигналов из-за указанных причин будут значительны. *Линии синхронизации сильно нагружены*, поскольку имеют значительную длину, и синхросигналы подаются на большое число элементов памяти. Результирующая емкостная

нагрузка в сложных устройствах оценивается сотнями или даже тысячами пикофарад, а это обуславливает большие времена переключения линий, т. е. существенные фазовые сдвиги синхросигналов, когда нагрузочные условия в разных областях синхронизации неодинаковы. При расфазировании синхросигналов в устройствах, обменивающихся данными, моменты приема данных отличаются от предусмотренных при проектировании системы, *что может создать критические временные состязания*. Для их предотвращения нужно будет снизить частоту синхросигналов и, соответственно, производительность системы. Применив блоки PLL или DLL, можно устранить фазовые сдвиги между контролируемыми синхросигналами, избежать снижения их частоты и получить выигрыш в производительности системы (например, в 30—35%).

Блоки PLL или DLL "привязывают" фронты синхросигналов в разных областях синхронизации к фронтам опорного сигнала, выравнивая тем самым их фазы (устраняя Clock Skew). Структура PLL приведена на рис. 3.37, *а*, где под блоком Л/А подразумевается преобразователь логического сигнала в аналоговый, а под блоком ГУН — генератор, управляемый напряжением.

Опорный и корректируемый синхросигналы подаются на фазовый детектор, определяющий фазовые соотношения между ними. Простейшим фазовым детектором может служить логический элемент "неравнозначность" (сложение по модулю 2). Если сигналы синфазны, то выходное напряжение элемента постоянно равно нулю. При фазовом сдвиге между синхросигналами появляются интервалы, на которых выход элемента становится единичным, причем длительности таких интервалов пропорциональны величине фазового сдвига. Среднее значение выходного напряжения пропорционально абсолютному значению фазового сдвига, а знак сдвига не определяется. Такой простой фазовый детектор пригоден для коррекции фазового сдвига синхросигнала, знак которого заранее известен. Подобные ситуации типичны, поскольку при сравнении фаз опорного синхросигнала и выработанного из него вторичного опорный сигнал всегда опережает вторичный. Однако подобные ситуации имеют место не всегда. Если, например, требуется устранить фазовый сдвиг между синхросигналами двух устройств, то знак фазового сдвига может быть любым. В этом случае фазовый детектор реализуется по схеме, близкой к схеме синхронизатора импульсов, рассмотренной в *§ 3.4*. Действительно, если один из синхросигналов использовать как тактирующий для триггера, принимающего информацию, а второй — как информационный сигнал, то при поступлении второго синхросигнала до первого он будет принят триггером, а при поступлении после первого — не будет. Таким образом, фазовый детектор типа синхронизатора импульсов будет вырабатывать логические сигналы, отображающие знак рассогласования фаз входных синхропоследовательностей. Именно такой фазовый детектор предполагается в схеме на рис. 3.37, *а*.

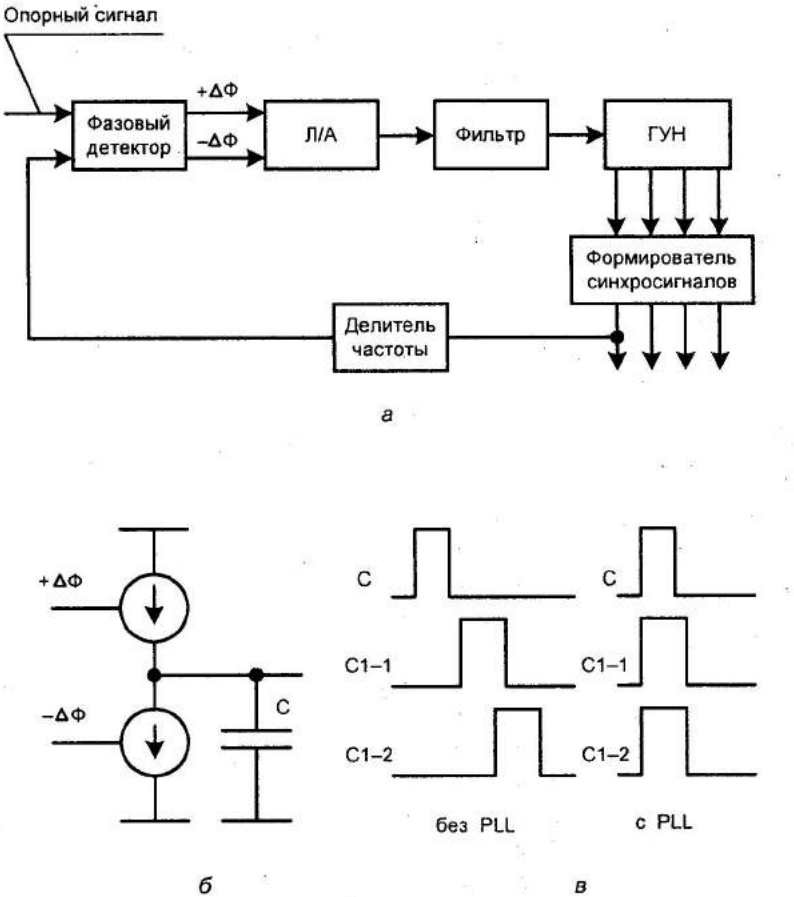


Рис. 3.37. Структура PLL (*а*), схема Charge Pump (*б*) и временные диаграммы синхросигналов при отсутствии и наличии PLL (*в*)

Для последующих блоков PLL нужны аналоговые сигналы, поэтому логические сигналы фазового детектора подаются на блок Л/А преобразования "логический—аналоговый", который в англоязычной литературе называют Charge Pump. В состав этого блока (рис. 3.37, *б*) входят генераторы токов и конденсатор. Логический сигнал от фазового детектора подключает к конденсатору тот или иной генератор тока в зависимости от знака рассогласования фаз синхросигналов (сигналы Up и Down), заставляя тем самым напряжение на емкости расти или уменьшаться.

Аналоговое напряжение используется в дальнейшем генератором, управляемым напряжением (ГУН), называемым также VCO (Voltage-Controlled Oscillator). ГУН вырабатывает импульсную последовательность переменной

фазы, подбираемой так, чтобы минимизировать фазовый сдвиг входных сигналов фазового детектора. Перед блоком ГУН включается фильтр низких частот, устраняющий высокочастотные составляющие напряжения, поскольку при отсутствии фильтра появляется дрожание фронтов выходного напряжения ГУН (Jitter). ГУН вырабатывает скорректированные по фазе синхросигналы, один из которых используется как входной для фазового детектора.

Для получения синхросигналов частоты, умноженной на N, в цепь обратной связи PLL вводится делитель частоты на N.

Блоки PLL и DLL — аналоговые замкнутые схемы, для которых серьезными проблемами являются обеспечение устойчивости, малой чувствительности к помехам и др. Возникают и альтернативные подходы к решению задач синхронизации процессов в сложных системах. В частности, имеются схемы, в которых *синхросигналы фактически извлекаются из самих передаваемых данных* (блоки CDR, Carrier Recovery Approach). О блоках CDR говорится далее в *главе 9*.

В качестве радикальной меры борьбы с расфазированием синхросигналов в сложных системах высокого быстродействия можно рассматривать *переход к так называемым самосинхронизирующимся схемам (Self-Timed Design)*, но пока что это мало применяется, т. к. цена введения самосинхронизации слишком велика (по дополнительным аппаратным затратам). Заметим также, что название "самосинхронизирующиеся" по существу неверно, т. к. фактически речь идет о разновидности *асинхронных систем*.

Однофазная синхронизация

Однофазная синхронизация использует минимальное число синхропоследовательностей и обеспечивает высокое быстродействие. В то же время ее применение сопровождается специфическими проблемами.

При однофазной синхронизации на все элементы памяти подаются одни и те же синхроимпульсы. Если бы устройство строилось на безынерционных элементах, то однофазная синхронизация была бы невозможна, т. к. в момент подачи синхроимпульса, т. е. команды на прием данных, эти данные исчезли бы. Это произошло бы потому, что при подаче синхроимпульса один и тот же элемент памяти должен одновременно принимать данные от предыдущего и снабжать данными последующий, что невозможно в безынерционной цепи, если только элементы памяти не обеспечивают за счет своей структуры присутствия в них одновременно "старой" и "новой" информации (это возможно в двухступенчатых триггерах).

Реальные элементы всегда инерционны, поэтому принципиальная возможность однофазной синхронизации появляется даже для систем с одноступенчатыми триггерами, но условия работоспособности могут оказаться трудновыполнимыми.

Рассмотрим однофазную синхронизацию для систем с простейшими триггерами — одноступенчатыми, управляемыми уровнем. Поступающие на входы триггеров синхроимпульсы должны иметь длительность, достаточную для их надежного переключения ($t_и \geq t_{wc}$). После переключения триггеров на входах КЦ появляются новые значения аргументов, а по истечении $t_{кц.min}$ изменятся сигналы на входах триггеров, но эти изменения не должны восприниматься триггерами. Если к указанному моменту синхроимпульсы еще не закончились, то состояния триггеров могут повторно измениться в одном и том же такте, что недопустимо. Поэтому должно соблюдаться следующее условие работоспособности

$$t_{wc} \leq t_и \leq t_{тг\ min} + t_{кц.min},$$

где $t_{тг\ min}$ — минимальное время переключения триггера.

Как видно, в данном случае необходимо строгое ограничение длительности импульсов снизу и сверху, т. к. за время существования импульса обязан переключиться даже самый инерционный триггер и, в то же время, информация не должна успеть пройти через самый быстродействующий каскад обработки данных (триггер плюс КЦ). Это условие должно соблюдаться во всем диапазоне изменений условий эксплуатации устройства. Расчету условий работоспособности данного варианта системы синхронизации препятствует также то, что сведения о минимальных задержках элементов могут отсутствовать.

Полученная формула определяет возможность применения однофазной синхронизации в схеме с одноступенчатыми триггерами, управляемыми уровнем, и показывает, что с ростом минимальной логической глубины КЦ реализация такой системы облегчается. Это обстоятельство подтверждает отмеченную ранее желательность выравнивания задержек сигналов в различных путях прохождения их на выход КЦ.

На практике однофазная синхронизация чаще всего применяется в схемах с триггерами, имеющими динамическое управление, или с двухступенчатыми триггерами.

При использовании триггеров с динамическим управлением (рис. 3.38) информация принимается по фронту синхроимпульса, а чувствительность триггера к информационным сигналам сохраняется лишь в малом интервале времени в окрестности фронта (в течение времени выдержки $t_Н$). Триггеры должны потерять чувствительность к изменениям информационных сигналов, прежде чем до их входов по кратчайшему пути может дойти такое изменение. Если это не обеспечивается, возможен сбой. Таким образом, и в этом варианте однофазной системы синхронизации требуется соблюдение определенного условия работоспособности: $t_{тг\ min} + t_{кц.\ min} \geq t_Н$.

Легко заметить, что обеспечить это условие работоспособности гораздо проще, чем предыдущее, т. к. величина $t_Н$ мала. Более того, для ряда тригге-

ров, в частности, для JK-триггеров, реализованных по схеме с внутренними задержками, $t_н = 0$. А это значит, что при их применении работоспособность систем с однофазной синхронизацией гарантирована.

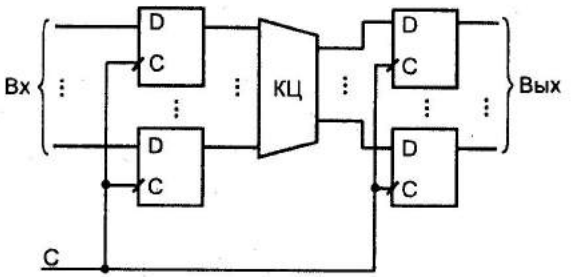


Рис. 3.38. Схема однофазной синхронизации триггеров с динамическим управлением

В системах однофазной синхронизации с двухступенчатыми триггерами высокий уровень синхросигнала открывает входные ступени триггеров, оставляя неизменными их выходные сигналы. При этом данные с предыдущих каскадов записываются во входные ступени следующих. Такую запись можно вести в течение необходимого времени без каких-либо опасностей временных состязаний сигналов. Переход синхросигнала на низкий уровень переносит состояния входных ступеней в выходные, изменяет тем самым переменные на входе КЦ, которые вырабатывают новые сигналы для триггеров следующего каскада. Этот процесс также можно вести достаточно длительное время без каких-либо опасений, поскольку входные ступени всех триггеров закрыты. Очередной переход синхросигнала на высокий уровень вновь запишет информацию во входные ступени триггеров и т. д. При правильном выборе параметров синхросигналов временные состязания сигналов в системе с двухступенчатыми триггерами вообще отсутствуют, работоспособность ее обеспечивается при сколь угодно малых минимальных задержках.

В то же время усложняются триггеры и увеличивается длительность паузы (необходимо дополнительное время на переключение выходных ступеней триггеров).

Расчетные соотношения для проектирования однофазной синхронизации

Такие соотношения для системы с триггерами, имеющими динамическое управление (для определенности — прямое), получим, приняв следующие условия. Частота синхроимпульсов постоянна (обоснованность этого условия связана с возможностью стабилизировать частоту генератора с точно-

стью, намного превышающей точность задания других параметров импульсов). Положение фронтов синхроимпульсов во времени задается с допусками Δ, т. е. при номинальном времени появления фронта t_0 фронт может появиться в интервале от $t_0 - \Delta$ до $t_0 + \Delta$. В этих допусках отражены все причины неточностей задания синхросигналов (сдвиги фронтов в схеме размножения синхросигналов, задержки в связях, разброс моментов срабатывания триггеров из-за разброса их пороговых напряжений и др.).

Цель расчета — *минимизировать период синхросигналов при соблюдении условий надежной работы устройства* и заданных разбросах параметров.

Объект расчета — система однофазной синхронизации с триггерами, имеющими динамическое управление, представляет собой важное практическое значение.

На временной диаграмме синхросигнала (рис. 3.39) отмечены следующие временные интервалы. Номинальный момент начала первого импульса $t = 0$ и номинальный момент начала второго импульса $t = T$, разбросы возможных моментов поступления импульсов относительно номинальных моментов Δ, времена предустановки и выдержки для используемого типа триггера t_S и t_H, суммарные длительности переключения триггера по цепи "синхровход—выход" и прохождения сигнала через комбинационную цепь $t_{тг} + t_{кц}$ для их максимального и минимального значений.

Чтобы соблюдалось требование неизменности информационного сигнала на интервале предустановки, входной сигнал триггера должен устанавливаться не позднее, чем в момент времени $-(\Delta + t_S)$ для первого импульса и в момент $T - \Delta - t_S$ для второго импульса. Изменение информационного сигнала становится допустимым не раньше момента времени $\Delta + t_H$ для первого импульса и $T + \Delta + t_H$ для второго. Наиболее позднее появление входного информационного сигнала в интервале между импульсами происходит в момент $\Delta + t_{тг\ max} + t_{кц.\ max}$, а наиболее раннее в момент $-\Delta + t_{тг\ min} + t_{кц.\ min}$.

Чтобы наиболее позднее поступление информационного сигнала оказалось в допустимой области, необходимо соблюдение условия

$$T - \Delta - t_S \geq \Delta + t_{тг\ max} + t_{кц.\ max}.$$

Из этого неравенства следует

$$T \geq 2\Delta + t_S + t_{тг\ max} + t_{кц.\ max}. \qquad (3.3)$$

Условие (3.3) обеспечивает неизменность информационного сигнала на входе триггера в течение интервала t_S при наихудшем случае разброса параметров.

Следует также обеспечить соблюдение неизменности информационного сигнала на интервале t_H. Чтобы это изменение оказалось в допустимом интервале, необходимо выполнить требование

$$-\Delta + t_{тг\ min} + t_{кц.\ min} \geq \Delta + t_H,$$

из которого следует условие

$$t_{кц.\ min} \geq 2\Delta + t_{н} - t_{тг\ min}. \qquad (3.4)$$

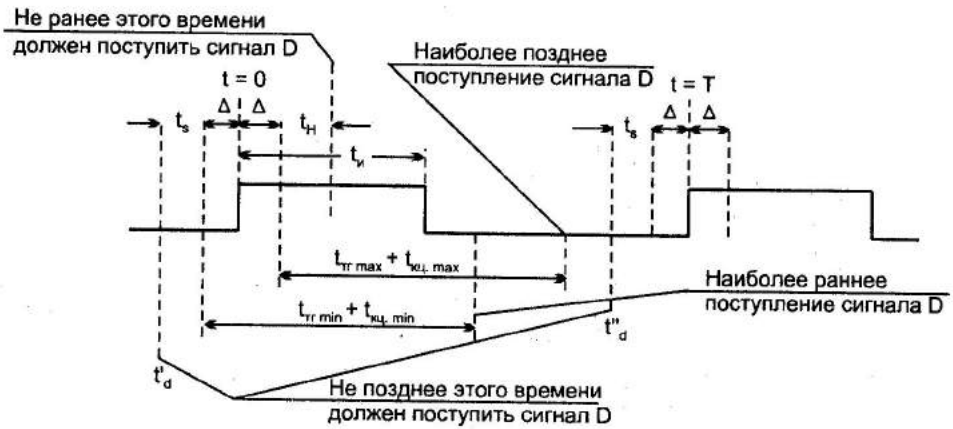


Рис. 3.39. Временная диаграмма синхросигнала однофазной системы синхронизации

Еще одним необходимым условием является требование длительности импульса, достаточной для надежного переключения триггера

$$t_{и} \geq 2\Delta + t_{и\ min}. \qquad (3.5)$$

Порядок определения параметров синхроимпульсов: выбор $t_{и}$ по условию (3.5), выбор T по условию (3.3), проверка выполнения условия (3.4).

Слагаемое 2Δ в выражении (3.5) отражает возможность запаздывания переднего и опережения заднего фронта синхроимпульсов. Нарушение условия (3.4) может потребовать введения задержек в соответствующие цепи, в частности, на выходах триггера. Задержки в связях в расчетных зависимостях отдельно не фигурируют — подразумевается их учет суммированием с задержками элементов.

Двухфазная синхронизация

Такая синхронизация характеризуется использованием двух последовательностей синхроимпульсов (рис. 3.40, *а*), сдвинутых во времени друг относительно друга. Интервал между импульсами обеих последовательностей отводится для работы комбинационных цепей. Соседние каскады получают разноименные серии синхроимпульсов (рис. 3.40, *б*).

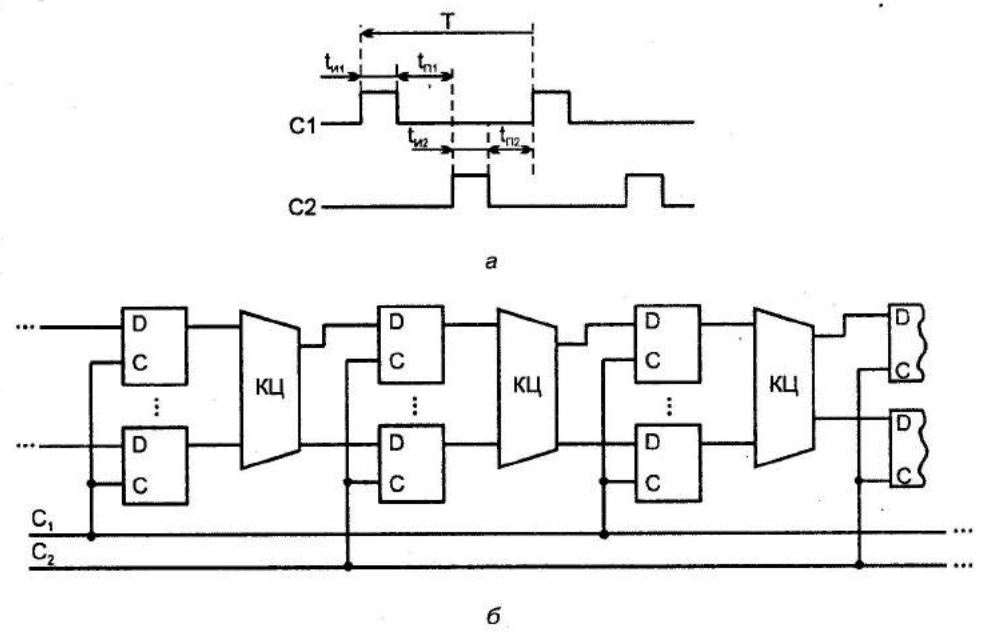


Рис. 3.40. Временная диаграмма синхросигналов (*а*) и схема тактирования элементов памяти для двухфазной системы синхронизации (*б*)

При возбуждении фазы С2 данные с триггеров фазы С1 через соответствующие КЦ передаются на триггеры фазы С2. При возбуждении фазы С1 триггеры этой фазы через КЦ принимают данные от триггеров фазы С2. Поочередное возбуждение фаз обеспечивает передачу данных по тракту их обработки без каких-либо временных состязаний, т. к. выдача данных производится триггерами, не изменяющими своих состояний в данной фазе, а прием данных осуществляется после завершения переходных процессов в КЦ.

Достоинством двухфазной системы является возможность применения простых одноступенчатых триггеров с управлением уровнем. В то же время наличие двух фаз синхроимпульсов усложняет схему устройства.

Расчет параметров синхроимпульсов для двухфазной системы основан на той же стратегии, что и расчет для однофазной, т. е. на обеспечении неизменности информационных сигналов на входах триггеров в интервалах t_S и t_H даже при наихудшем сочетании допусков на положения фронтов синхросигналов и задержек в КЦ.

Разные системы синхронизации встречаются в разработках ЦУ, выбор определяется конкретными условиями. В последнее время широко распространена однофазная система с триггерами, имеющими динамическое управление.

Многофазная синхронизация характеризуется использованием более чем двух серий синхроимпульсов и применяется для увеличения быстродействия систем путем организации работы их частей с разной скоростью. Это осуществляется разбиением периода основной частоты на части и использованием в отдельных блоках системы более высокочастотных синхросигналов. Для узлов и устройств применение многофазной системы синхронизации не характерно.

§ 3.7. Регистры и регистровые файлы

Регистры — самые распространенные узлы цифровых устройств. Они оперируют с множеством связанных переменных, составляющих слово. Над словами выполняется ряд операций: прием, выдача, хранение, сдвиг в разрядной сетке, поразрядные логические операции.

Регистры состоят из разрядных схем, в которых имеются триггеры и, чаще всего, также и логические элементы.

По количеству линий передачи переменных регистры делятся на однофазные и парафазные, по системе синхронизации на однотактные, двухтактные и многотактные. Однако главным классификационным признаком является способ приема и выдачи данных. По этому признаку различают *параллельные* (статические) регистры, *последовательные* (сдвигающие) и *параллельно-последовательные*.

В параллельных регистрах прием и выдача слов производятся по всем разрядам одновременно. В них хранятся слова, которые могут быть подвергнуты поразрядным логическим преобразованиям.

В последовательных регистрах слова принимаются и выдаются разряд за разрядом. Их называют *сдвигающими*, т. к. тактирующие сигналы при вводе и выводе слов перемещают их в разрядной сетке. Сдвигающий регистр может быть нереверсивным (с однонаправленным сдвигом) или реверсивным (с возможностью сдвига в обоих направлениях).

Последовательно-параллельные регистры имеют входы/выходы одновременно последовательного и параллельного типа. Кроме того, существуют варианты с последовательным входом и параллельным выходом (SIPO, Serial Input — Parallel Output), параллельным входом и последовательным выходом (PISO), а также варианты с возможностью любого сочетания способов приема и выдачи слов.

В параллельных (статических) регистрах схемы разрядов не обмениваются данными между собой. Общими для разрядов обычно являются цепи тактирования, сброса/установки, разрешения выхода или приема, т. е. цепи управления. Пример схемы статического регистра, построенного на триггерах типа D

с прямыми динамическими входами, имеющего входы сброса $\overline{R}$ и выходы с третьим состоянием, управляемые сигналом EZ, показан на рис. 3.41.

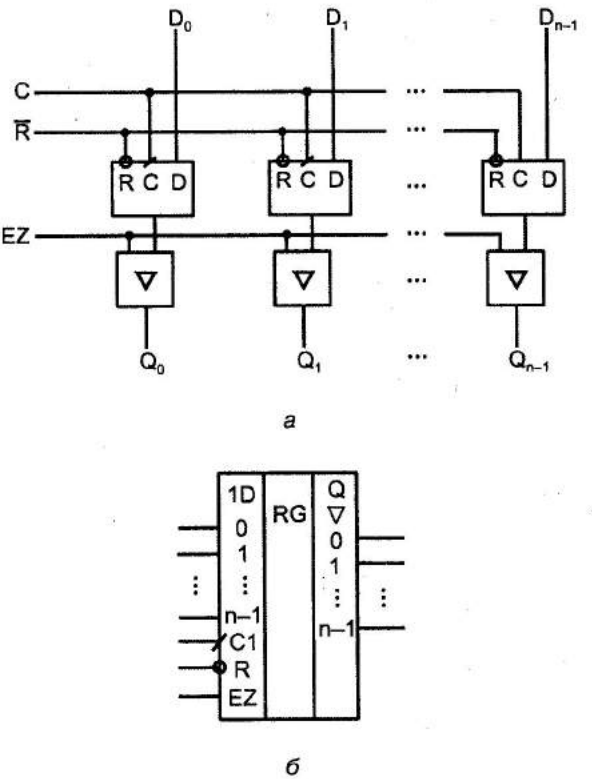


Рис. 3.41. Схема статического регистра (*а*) и его условное графическое обозначение (*б*)

Для современной схемотехники характерно построение регистров именно на D-триггерах, преимущественно с динамическим управлением. Многие имеют выходы с третьим состоянием, некоторые регистры относятся к числу буферных, т. е. рассчитаны на работу с большими емкостными и/или низкоомными активными нагрузками. Это обеспечивает их работу непосредственно на магистраль (без дополнительных схем интерфейса).

Регистровые файлы

Из статических регистров составляются блоки регистровой памяти — *регистровые файлы*. В микросхеме типа ИР26 (серий КР1533, К555 и др.) можно хранить 4 четырехразрядных слова с возможностью независимой и одновре-

менной записи одного слова и чтения другого. Информационные входы регистров соединены параллельно (рис. 3.42). Входы адресов записи WA и WB (от англ. *Write*) дают четыре комбинации, каждая из которых разрешает "защелкнуть" данные, присутствующие в настоящее время на выводах D_{1-4}.

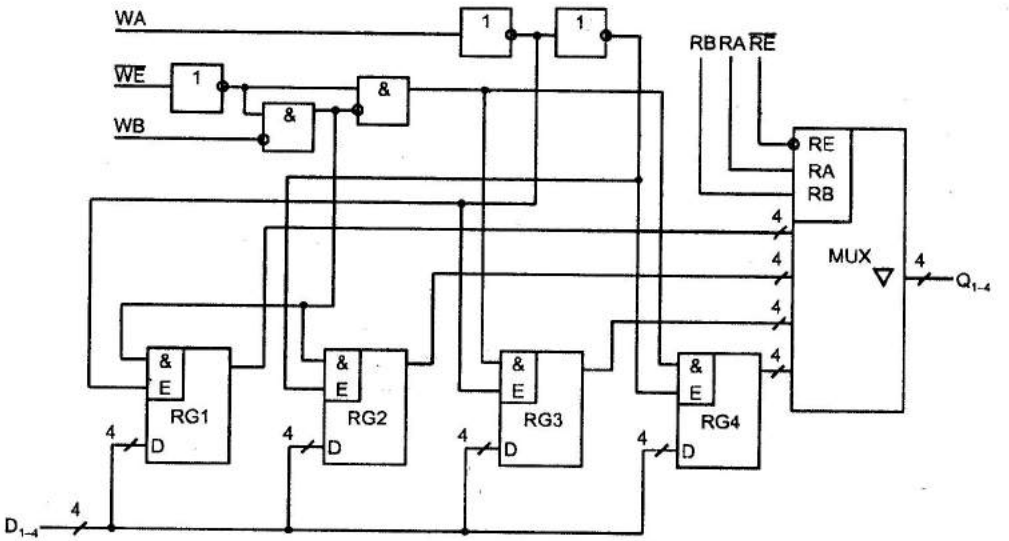


Рис. 3.42. Схема регистрового файла

Содержимое файла (регистра) вызывается на выходы блока Q_{1-4} с помощью дешифратора считывания (адресных входов мультиплексора MUX) адресами RA и RB (от англ. *Read*). Таких адресов четыре.

Если на входе разрешения записи $\overline{WE}$ (Write Enable) действует активный низкий уровень, то данные поступают в соответствующий регистр, при высоком уровне $\overline{WE}$ входы для данных и адресов запрещены.

Выходные данные выдаются в прямом коде.

Размерность регистровой памяти можно наращивать, составляя из нескольких ИС блок памяти. При наращивании числа хранимых слов выходы отдельных ИС с тремя состояниями соединяются в одной точке. Допускается соединять непосредственно до 128 выходов, что дает 512 хранимых слов. Ограничение на число соединяемых в одной точке выходов вызвано токовым режимом выхода, оно может быть преодолено при подключении к выходной точке специальных внешних резисторов. При наращивании разрядности слова соединяют параллельно входы разрешения и адресации нескольких ИС, выходы которых в совокупности дают единое информационное слово.

Сдвигающие регистры

Последовательные (сдвигающие) регистры представляют собою цепочку разрядных схем, связанных цепями переноса.

В однотактных регистрах со сдвигом на один разряд вправо (рис. 3.43, *а*) слово сдвигается при поступлении синхросигнала. Вход и выход последовательные (DSR — Data Serial Right). На рис. 3.43, *б* показана схема регистра со сдвигом влево (вход данных DSL — Data Serial Left), а на рис. 3.43, *в* иллюстрируется принцип построения реверсивного регистра, в котором имеются связи триггеров с обоими соседними разрядами, но соответствующими сигналами разрешается работа только одних из этих связей (команды "влево" и "вправо" одновременно не подаются).

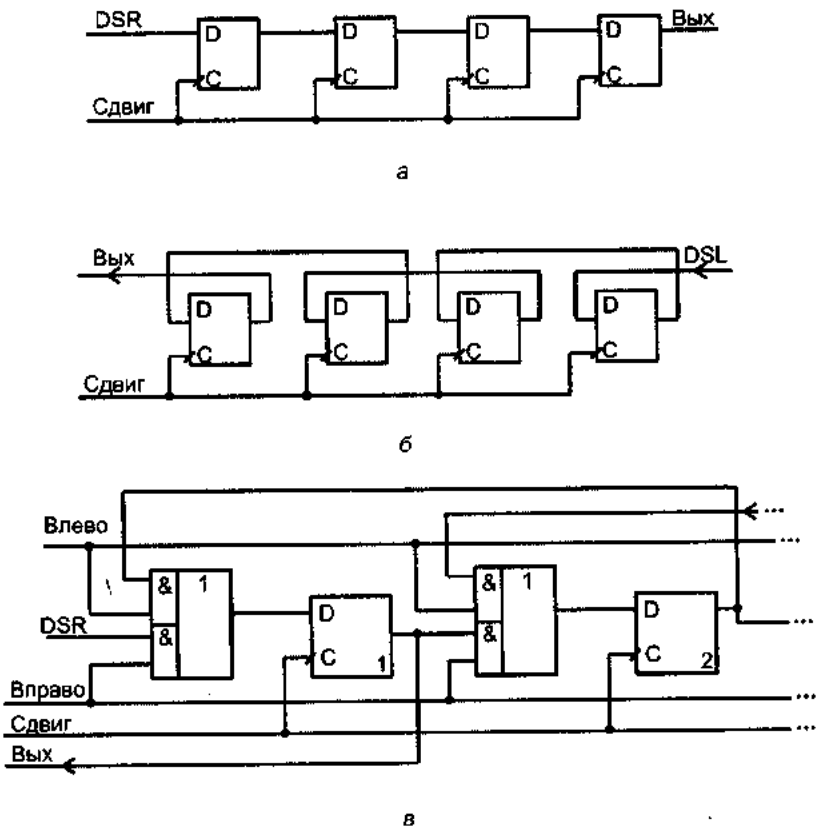


Рис. 3.43. Схемы регистров сдвига вправо (*а*), влево (*б*) и схемы регистров реверсивного сдвига (*в*)

Согласно требованиям синхронизации, рассмотренным в предыдущем параграфе, в сдвигающих регистрах, не имеющих логических элементов в межразрядных связях, нельзя применять одноступенчатые триггеры, управляемые уровнем, поскольку некоторые триггеры могут за время действия разрешающего уровня синхросигнала переключиться неоднократно, что недопустимо.

Триггеры с динамическим управлением или двухступенчатые могут быть использованы так, как описано в *§ 3.6.*

Появление в межразрядных связях логических элементов и, тем более, логических схем неединичной глубины упрощает выполнение условий работоспособности регистров и расширяет спектр типов триггеров, пригодных для этих схем.

Многотактные сдвигающие регистры управляются несколькими синхропоследовательностями. Из их числа наиболее известны двухтактные с основным и дополнительным регистрами, построенными на простых одноступенчатых триггерах, управляемых уровнем. По такту C1 содержимое основного регистра переписывается в дополнительный, а по такту C2 возвращается в основной, но уже в соседние разряды, что соответствует сдвигу слова. По затратам оборудования и быстродействию этот вариант близок к однотактному регистру с двухступенчатыми триггерами.

Универсальные регистры

В сериях ИС и библиотеках БИС/СБИС программируемой логики имеется много вариантов регистров (в схемотехнике ТТЛШ их около 30). Среди них многорежимные (многофункциональные) или универсальные, способные выполнять набор микроопераций. Многорежимность достигается композицией в одной и той же схеме частей, необходимых для выполнения различных операций. Управляющие сигналы, задающие вид выполняемой в данное время операции, активизируют необходимые для этого части схемы.

Типичным представителем многорежимных регистров является микросхема ИР13 серии КР1533 и других (рис. 3.44). Это восьмиразрядный регистр с возможностью двусторонних сдвигов с допустимой тактовой частотой до 25 МГц при токе потребления до 40 мА. Имеет также параллельные входы и выходы, вход асинхронного сброса $\overline{R}$ и входы выбора режима S_0 и S_1, задающие четыре режима (параллельная загрузка, два сдвига и хранение). Функционирование регистра определяется табл. 3.13.

Условное обозначение регистра ИР13 приведено на рис. 3.45.

Таблица 3.13

Режим	Входы							Выходы				
	C	$\overline{R}$	SO	S1	DSR	DSL	D_n	Q_0	Q_1	...	Q_6	Q_7
Сброс	X	L	X	X	X	X	X	L	L	...	L	L
Хранение	⌐	H	L	L	X	X	X	Q_0	Q_1	...	Q_6	Q_7
Сдвиг влево	⌐	H	H	L	X	L	X	Q_1	Q_2	...	Q_7	L
		H	H	L	X	H	X	Q_1	Q_2		Q_7	H
Сдвиг вправо	⌐	H	L	H	L	X	X	L	Q_0	...	Q_5	Q_6
		H	L	H	H	X	X	H	Q_0		Q_5	Q_6
Парал-лельная загрузка	⌐	H	H	H	X	X	D_n	D_0	D_1	...	D_6	D_7

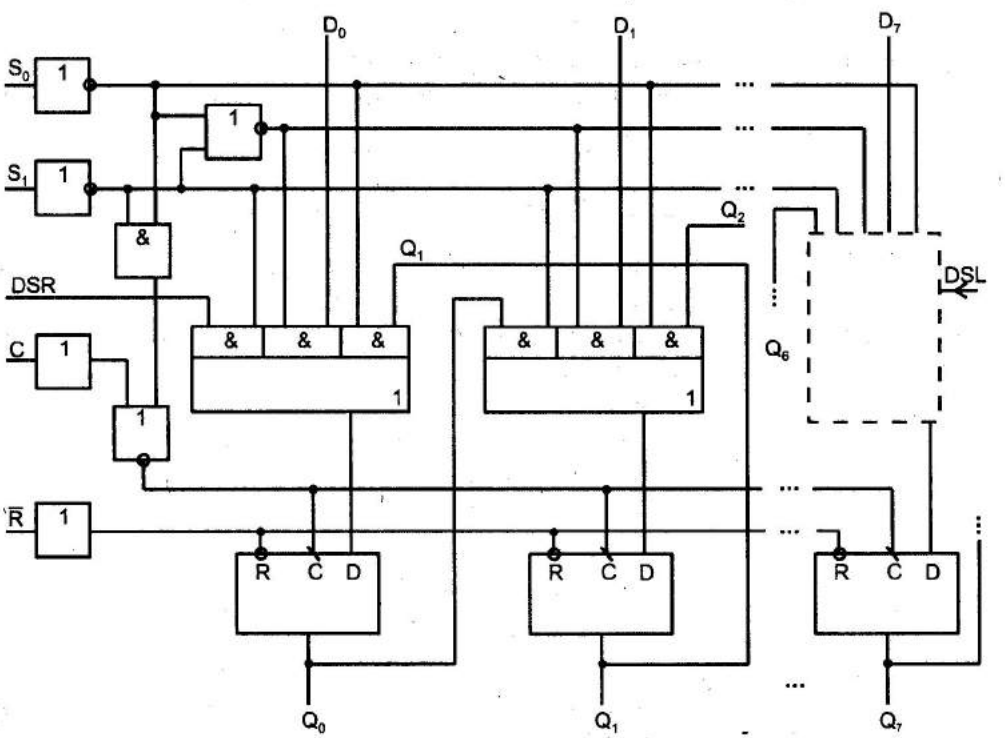


Рис. 3.44. Схема многорежимного регистра

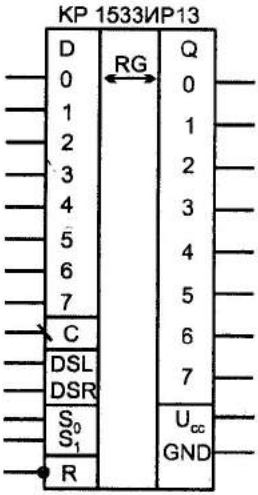


Рис. 3.45. Условное обозначение универсального регистра

Регистры, имеющие разнотипные вход и выход, служат основными блоками *преобразователей параллельных кодов в последовательные и обратно.* На рис. 3.46 показана схема преобразователя параллельного кода в последовательный на основе восьмиразрядного регистра типа SI/PI/SO. В этой схеме отрицательный стартовый импульс St, задающий уровень логического нуля на верхнем входе элемента 1, создает единичный сигнал параллельного приема данных на вход L (Load — загрузка), по которому в разряды 1...7 регистра RG загружается преобразуемое слово D_{1-7}, а в нулевой разряд — константа 0. На последовательный вход DSR подана константа 1.

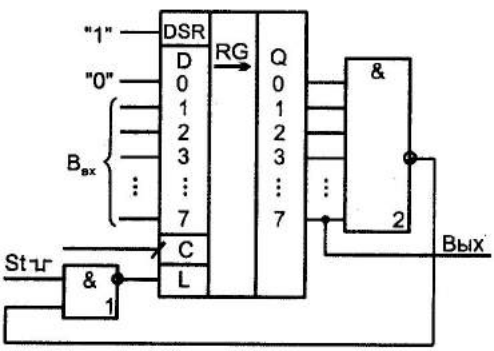


Рис. 3.46. Схема преобразователя параллельного кода в последовательный

Таким образом, после загрузки в регистре формируется слово 0D1D2...D7. Тактовые импульсы, поступающие на вход С, вызывают сдвиги слова вправо (для условного обозначения это соответствует сдвигу вниз). Сдвиги выводят слово в последовательной форме через выход Q7. Вслед за информационными разрядами идет нуль (константа "0"), после которого цепочка единиц. Пока нуль не выведен из регистра, на выходе элемента 2 действует единичный сигнал. После вывода нуля все входы элемента 2 становятся единичными, его выход приобретает нулевое значение и через элемент 1 формирует сигнал автоматической загрузки следующего слова, после чего цикл преобразования повторяется.

В перечне микроопераций, выполняемых регистрами, *были указаны поразрядные логические операции.* Современные регистры мало приспособлены для выполнения этих операций, однако при необходимости их можно выполнить, пользуясь регистрами на RS-триггерах. Для выполнения операции ИЛИ на S-входы статического регистра с исходным нулевым состоянием подается первое слово А, единичные разряды которого устанавливают соответствующие триггеры. Затем без сброса регистра на S-выходы подается второе слово В. Ясно, что в результате получим результат $Q = A \vee B$.

При выполнении поразрядной операции И в первом такте на S-входы регистра подается слово А, устанавливающее те разряды регистра, в которых слово А имеет единицы. Затем следует подать на регистр слово В. Чтобы в регистре сохранились единицы только в тех разрядах, в которых единицы имеют оба слова, слово В подается на входы R триггеров в инверсном виде.

Сложение по модулю 2 может быть выполнено схемой с триггерами типа Т в разрядах путем последовательной во времени подачи на нее двух слов А и В.

§ 3.8. Основные сведения о счетчиках. Двоичные счетчики

Понятие "счетчик" является очень широким. *К счетчикам относят автоматы, которые под действием входных импульсов переходят из одного состояния в другое, фиксируя тем самым число поступивших на их вход импульсов в том или ином коде.*

Специфичной для счетчиков операцией является изменение их содержимого на единицу (может быть и условную). Прибавление такой единицы соответствует операции инкрементации, вычитание — операции декрементации. Обычно счетчиками выполняются также и другие операции — сброс, установка, параллельная загрузка и др.

Счетчик характеризуется *модулем счета М (емкостью). Модуль определяет число возможных состояний счетчика.* После поступления на счетчик М входных сигналов начинается новый цикл, повторяющий предыдущий.

Классификация счетчиков

По способу кодирования внутренних состояний различают двоичные счетчики, счетчики Джонсона, счетчики с кодом "1 из N" и др.

По направлению счета счетчики делятся на суммирующие (прямого счета), вычитающие (обратного счета) и реверсивные (с изменением направления счета).

По принадлежности к тому или иному классу автоматов говорят о синхронных или асинхронных счетчиках (более подробную классификацию по этому признаку не затрагиваем, учитывая реальный состав микросхем счетчиков).

Счетчики строятся из разрядных схем, имеющих межразрядные связи. Соответственно организации этих связей различают счетчики с последовательным, параллельным и комбинированными переносами.

Возможные *режимы работы* счетчика:

- регистрация числа поступивших на счетчик сигналов;
- деление частоты.

В первом режиме результат — содержимое счетчика, во втором режиме выходными сигналами являются импульсы переполнения счетчика.

Быстродействие счетчика характеризуется временем установления в нем нового состояния (первый режим), а также максимальной частотой входных сигналов f_{max}.

Как и любой автомат, счетчик можно строить на триггерах любого типа, однако удобнее всего использовать для этого триггеры типа T (счетные) и JK, имеющие при J = K = 1 счетный режим.

Состояние счетчика читается по выходам разрядных схем как слово $Q_{n-1}Q_{n-2}...Q_0$, входные сигналы поступают на младший разряд счетчика.

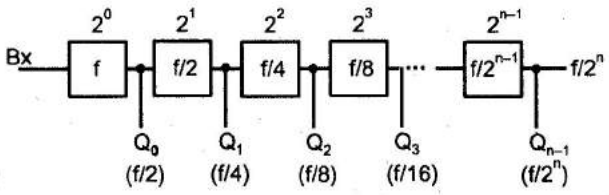
Двоичным счетчиком назовем счетчик, имеющий модуль $M = 2^n$, где n — целое число, и естественную последовательность кодов состояний (его состояния отображаются последовательностью двоичных чисел, десятичными эквивалентами которых будут числа 0, 1, 2, 3, ..., M−1).

Двоичные счетчики

Схему *двоичного счетчика* можно получить с помощью формального синтеза, однако более наглядным путем представляется эвристический. Таблица истинности двоичного счетчика — последовательность двоичных чисел от нуля до M − 1. Наблюдение за разрядами чисел, составляющих таблицу, приводит к пониманию структурной схемы двоичного счетчика. Состояния младшего разряда при его просмотре по соответствующему столбцу таблицы показывают чередование нулей и единиц вида 01010101..., что естественно,

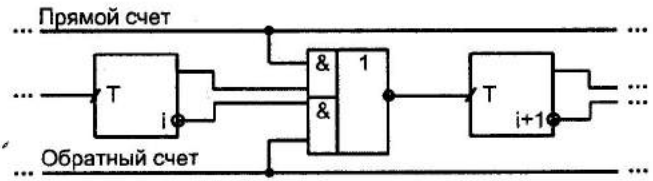
т. к. младший разряд принимает входной сигнал и переключается от каждого входного воздействия. В следующем разряде наблюдается последовательность пар нулей и единиц вида 00110011... . В третьем разряде образуется последовательность из четверок нулей и единиц 00001111... и т. д. Из этого наблюдения видно, что следующий по старшинству разряд переключается с частотой, в два раза меньшей, чем данный.

Известно, что счетный триггер делит частоту входных импульсов на два. Сопоставив этот факт с указанной выше закономерностью, видим, что счетчик может быть построен в виде цепочки последовательно включенных счетных триггеров (рис. 3.47, *а*). Заметим, кстати, что, согласно ГОСТу, входы элементов изображаются слева, а выходы справа. Соблюдение этого правила ведет к тому, что в числе, содержащемся в счетчике, младшие разряды расположены левее старших.



а

б



в

Рис. 3.47. Структура последовательного счетчика (*а*), ее реализация на триггерах с прямым динамическим управлением (*б*) и межразрядные связи реверсивного счетчика (*в*)

Представление счетчика цепочкой Т-триггеров справедливо как для суммирующего, так и для вычитающего вариантов, поскольку закономерность по соотношению частот переключения разрядов сохраняется как при просмотре таблицы сверху вниз (прямой счет), так и снизу вверх (обратный счет). Различия при этом состоят в направлении переключения предыдущего разряда, вызывающего переключение следующего. При прямом счете следующий разряд переключается при переходе предыдущего в направлении от 1 к 0, а при обратном — при переключении от 0 к 1. Следовательно, различие между вариантами заключается в разном подключении входов триггеров к выходам предыдущих. Если схема строится на счетных триггерах с прямым динамическим управлением, то характер подключения следующих триггеров к предыдущим для получения счетчиков прямого и обратного счета будет соответствовать рис. 3.47, *б*.

Из различия вариантов прямого и обратного счета следует также и способ построения *реверсивного счетчика* (рис. 3.47, *в*) путем переноса точки съема сигнала с триггера на противоположный выход под действием управляющего сигнала и с помощью элемента И-ИЛИ-НЕ, как показано на рисунке, либо элемента И-ИЛИ.

Полученные структуры относятся к *асинхронным счетчикам*, т. к. в них каждый триггер переключается выходным сигналом предыдущего, и эти переключения происходят не одновременно. Переключение одного триггера за другим есть не что иное, как распространение переноса по разрядам числа при изменении содержимого счетчика. В худшем случае перенос распространяется по всей разрядной сетке от младшего разряда к старшему, т. е. для установления нового состояния должны переключиться последовательно все триггеры. Отсюда видно, что время установления кода в асинхронном счетчике составит величину $t_{уст} \le nt_{тг}$. Другим названием асинхронного счетчика является название "последовательный счетчик".

Максимальная частота входных импульсов в режиме деления частоты ограничивается возможностями триггера младшего разряда, т. к. все последующие разряды переключаются с более низкими частотами.

Особенностью последовательных (асинхронных) счетчиков является возникновение в переходных процессах ложных состояний из-за задержек переключения триггеров. На рис. 3.48 показана временная диаграмма работы двухразрядного суммирующего счетчика на триггерах с прямым динамическим управлением, построенная с учетом задержек переключения триггеров $t_{тг}$. Читая состояние счетчика Q по потенциалам на выходах триггеров Q_0 и Q_1, видим, что после состояний 1 и 3 появляются ложные состояния 0 и 2 (показаны штриховкой). Опасность воздействия коротких ложных импульсов на ЦУ заставляет прибегать при необходимости к стробированию выхода счетчика.

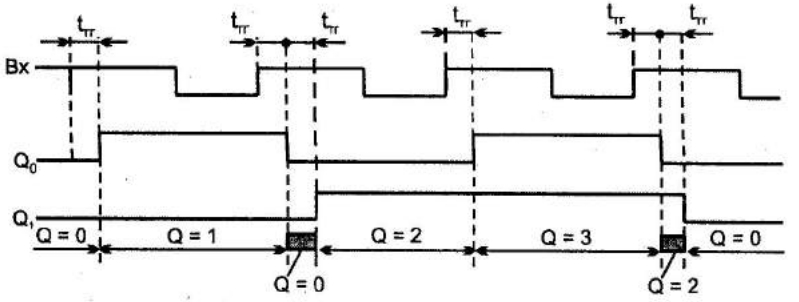


Рис. 3.48. Временные диаграммы работы последовательного двоичного счетчика

Максимальным быстродействием обладают *синхронные счетчики с параллельным переносом*, структуру которых найдем эвристически, рассмотрев процессы прибавления единицы к двоичным числам и вычитания ее из них, например:

$$\begin{array}{cccc|cccc} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ + & & & & & & & 1 \\ \hline 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{array} \qquad \begin{array}{cccc|cccc} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ - & & & & & & & 1 \\ \hline 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{array}$$

Результат всегда отличается от исходного числа только в нескольких младших разрядах, значения которых инвертируются. Для суммирующего счетчика требуется инверсия разрядов до первого разряда, равного логическому нулю, включая и его, а для вычитающего аналогично до разряда, равного логической единице. Таким образом, в суммирующем счетчике должны переключиться разряды, для которых все младшие единичны, для вычитающего — те, для которых все младшие находятся в нуле.

Эти задачи и должны решать счетчики. Время установления нового состояния для таких счетчиков (если не учитывать зависимости задержек элементов от нагрузок на них) не зависит от их разрядности и равно времени переключения триггера $t_{тг}$. Длительность цикла равна $t_{тг} + t_к$, где $t_к$ — задержка коньюнктора. Структура суммирующего синхронного счетчика с параллельным переносом, реализованного на триггерах с управлением фронтом, показана на рис. 3.49, *а*. Схема межразрядной связи для реверсивного счетчика с сигналом U/D (Up/Down, т. е. прямо/обратно) показана рис. 3.49, *б*.

С ростом числа разрядов реализация параллельных счетчиков затрудняется — требуются вентили с большим числом входов, растет нагрузка на выходы триггеров.

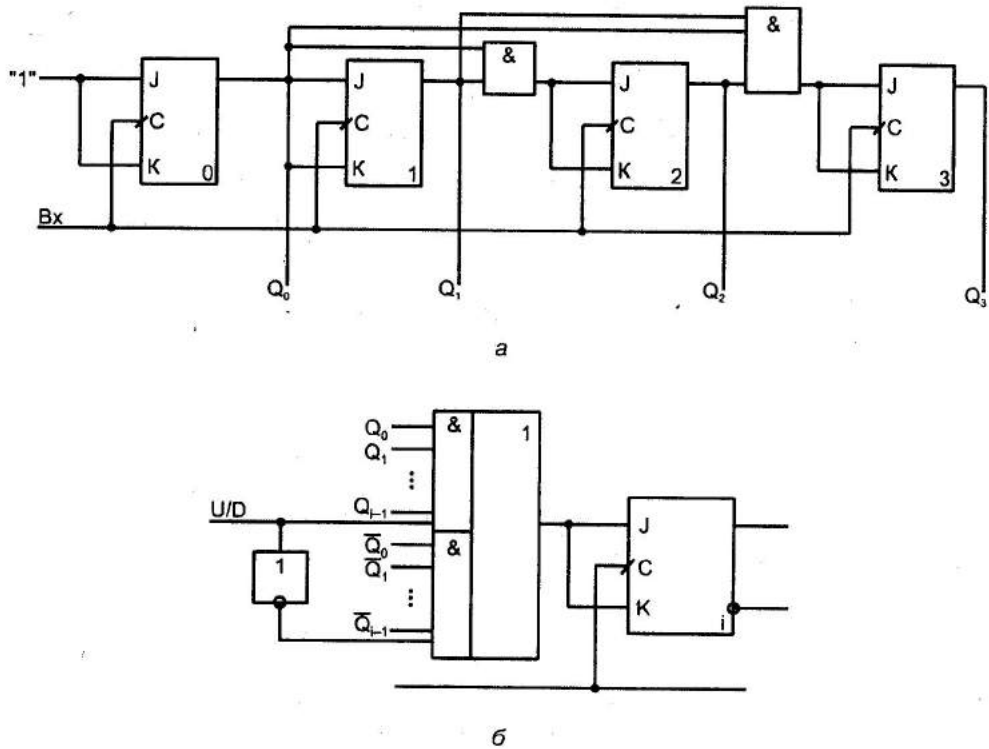


Рис. 3.49. Схемы параллельных счетчиков прямого счета (*а*) и реверсивного (*б*)

Счетчики с групповой структурой

В связи с ограничениями на построение параллельных счетчиков большой разрядности широкое распространение получили счетчики с групповой структурой, в которых счетчик разбивается на группы, связанные цепями межгруппового переноса (рис. 3.50, *а*). При единичном состоянии всех триггеров группы приход очередного входного сигнала создаст перенос из этой группы. Эта ситуация подготавливает межгрупповой конъюнктор к прямому пропусканию входного сигнала на следующую группу. В наихудшем для быстродействия случае, когда перенос проходит через все группы и поступает на вход последней,

$$t_{уст} = t_k(\ell - 1) + t_{гр},$$

где ℓ — число групп; $t_{гр}$ — время установления кода в группе.

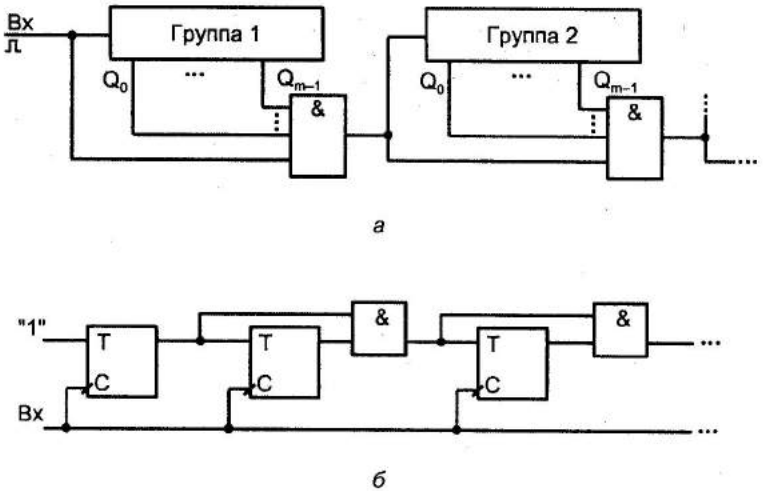


Рис. 3.50. Схемы счетчиков групповой структуры

Если уменьшить разрядность группы до единицы и использовать синхронные Т-триггеры, то получится схема *синхронного счетчика с последовательным переносом* (рис. 3.50, *б*). Схема относится к числу синхронных, т. к. все триггеры срабатывают одновременно под действием единого входного сигнала. В этом проявляется быстрая реакция схемы на входной сигнал, такая же, как и в счетчике с параллельным переносом. Однако по максимальной частоте входных сигналов эта схема существенно отличается от схемы с параллельным переносом, т. к. до подачи нового входного сигнала требуется предоставить время цепочке вентилей установиться в новое состояние путем их последовательного переключения.

В развитых сериях ИС обычно имеется по 5—10 вариантов двоичных счетчиков, выполненных в виде четырехразрядных групп (секций). Каскадирование секций может выполняться путем их последовательного включения по цепям переноса, организации параллельно-последовательных переносов или для более сложных счетчиков с двумя дополнительными управляющими входами разрешения счета и разрешения переноса путем организации параллельных переносов и в группах и между ними (см. например, [33], [38]).

Особенностью двоичных счетчиков синхронного типа является наличие ситуаций с одновременным переключением всех его разрядов (например, для суммирующего счетчика при переходе от кодовой комбинации 11...1 к комбинации 00...0 при переполнении счетчика и выработке сигнала переноса). *Одновременное переключение многих триггеров создает значительный токовый импульс в цепях питания* ЦУ и может привести к сбою в их работе *(см. § 1.4)*. Поэтому в руководящих материалах по использованию некоторых БИС/СБИС

программируемой логики, в частности, имеется ограничение на разрядность двоичных счетчиков заданной величиной k (например, 16). При необходимости применения счетчика большей разрядности рекомендуется переходить к коду Грея, для которого переходы от одной кодовой комбинации к другой сопровождаются переключением всего одного разряда. Правда, для получения результата счета в двоичном коде придется использовать дополнительно преобразователь кода, но это является платой за избавление от токовых импульсов большой интенсивности в цепях питания.

Пример условного обозначения счетчика приведен на рис. 3.51.

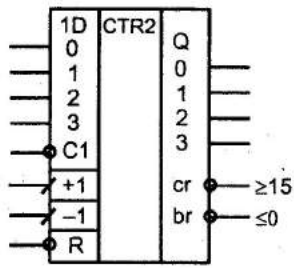


Рис. 3.51. Условное обозначение двоичного реверсивного счетчика со сбросом, параллельной загрузкой и выходами переноса и заема

§ 3.9. Двоично-кодированные счетчики с произвольным модулем

Счетчики с модулем, не равным целой степени числа 2, т. е. с произвольным модулем, реализуются на основе нескольких методов.

Для построения счетчика с произвольным модулем M берется разрядность $n = \lceil \log_1 M \rceil$, где $\lceil \rceil$ — знак округления до ближайшего справа целого числа. Иными словами, исходной структурой как бы служит двоичный счетчик с модулем 2^n, превышающим заданный и ближайшим к нему. Такой двоичный счетчик имеет $2^n - M = L$ лишних (неиспользуемых) состояний, подлежащих исключению.

Способы *исключения лишних состояний* многочисленны, и для любого M можно предложить множество реализаций счетчика. Исключая некоторое число первых состояний, получим ненулевое начальное состояние счетчика, что приводит к отсутствию естественного порядка счета и регистрации в счетчике кода с избытком. Исключение последних состояний позволяет сохранить естественный порядок счета. Сложность обоих вариантов принципиально одинакова, поэтому далее будем ориентироваться на схемы с естественным порядком счета. Состояния счетчиков во всех случаях предпо-

лагаем закодированными двоичными числами, т. е. будем рассматривать двоично-кодированные счетчики.

В счетчиках с исключением последних состояний счет ведется обычным способом вплоть до достижения числа M−1. Далее последовательность переходов счетчика в направлении роста регистрируемого числа должна быть прервана, и следующее состояние должно быть нулевым. При этом счетчик будет иметь M внутренних состояний (от 0 до M−1), т. е. его модуль равен M.

Остановимся на двух способах построения счетчиков с произвольным модулем: *модификации межразрядных связей* и *управлении сбросом*. При построении счетчика с модифицированными межразрядными связями последние, лишние, состояния исключаются непосредственно из таблицы функционирования счетчика. При этом после построения схемы обычным для синтеза автоматов способом получается счетчик, специфика которого состоит в нестандартных функциях возбуждения триггеров и, следовательно, в нестандартных связях между триггерами, что и объясняет название способа. Схема получается как специализированная, изменение модуля счета требует изменения самой схемы, т. е. легкость перестройки с одного модуля на другой отсутствует. В то же время реализация схемы счетчика может оказаться простой.

При управлении сбросом выявляется момент достижения содержимым счетчика значения M−1. Это является сигналом сброса счетчика в следующем такте, после чего начинается новый цикл. Этот вариант обеспечивает легкость перестройки счетчика на другие значения модуля, т. к. требуется изменять лишь код, с которым сравнивается содержимое счетчика для выявления момента сброса.

Построение счетчика методом модификации межразрядных связей

Построение счетчика методом модификации межразрядных связей проиллюстрируем примером для M = 5, начав с таблицы (табл. 3.14).

Таблица 3.14

Исходное состояние			Следующее состояние			Функции возбуждения					
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	0	0	0	X	1	0	X	0	X

При нахождении функций возбуждения триггеров использован "словарь" *(см. § 3.1)*. Имея в виду, что вместо символа произвольного сигнала X можно подставлять любую переменную (0 или 1), на основании таблицы запишем: $J_2 = Q_1Q_0$ (в столбце J_2 оставлена всего одна единица), $J_1 = Q_0$, $J_0 = \overline{Q}_2$. Для функций K_i (i = 0, 1, 2) выберем варианты с наибольшим числом констант, чтобы меньше нагружать источники сигналов. Примем, что $K_2 = 1$, $K_1 = J_1$ и $K_0 = 1$.

Схема счетчика приведена на рис. 3.52.

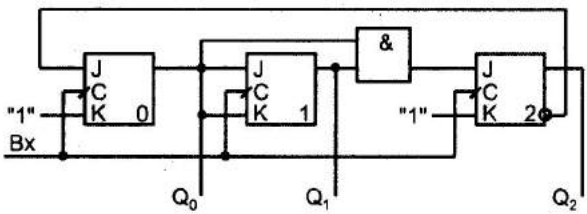


Рис. 3.52. Схема счетчика с модулем 5

В спроектированной схеме счетчика лишние состояния исключены в том смысле, что они не используются при нормальном функционировании счетчика. Но при сбоях или после подачи на схему напряжения питания в начале ее работы *лишние состояния могут возникать*. Поэтому полезно определить поведение схемы (автомата), в которой возникло лишнее состояние. Имея схему, можно полностью предсказать поведение схемы во всех возможных ситуациях. Сделаем это для полученной схемы счетчика с модулем 5.

Взяв каждое лишнее состояние, найдем для него функции возбуждения триггеров, определяющие их переходы в следующее состояние. При необходимости найдем таким же способом следующий переход и т. д. Для взятого примера лишними являются состояния 101, 110 и 111.

В состоянии 101 $Q_2 = 1$, $Q_1 = 0$ и $Q_0 = 1$. Зная функции возбуждения триггеров, находим, что $J_0 = 0$, $K_0 = 1$, $J_1 = K_1 = 1$, $J_2 = 0$, $K_2 = 1$. Следовательно, триггеры 0 и 2 сбросятся, а триггер 1 переключится в противоположное текущему состояние и из лишнего состояния 101 счетчик перейдет в состояние 010.

Аналогичным способом можно получить результаты для состояний 100 и 111. В итоге удобно построить диаграмму состояний счетчика (граф переходов), в которой будет учтен не только рабочий цикл (его состояния покажем кружками), но и поведение автомата, попавшего в неиспользуемые состояния (эти состояния показаны прямоугольниками). Такая диаграмма состояний показана на рис. 3.53. Из диаграммы видно, что рассматриваемый счетчик обладает *свойством самозапуска* (самовосстановления после сбоя) —

независимо от исходного состояния он приходит в рабочий цикл после начала работы. Этим свойством обладают не все схемы. В некоторых схемах автоматический вход в рабочий цикл не происходит.

При разработке некоторых схем в них вводят специальные элементы или подсхемы для придания свойств самозапуска.

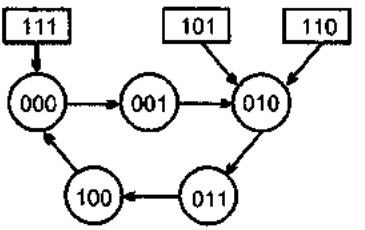


Рис. 3.53. Диаграмма состояний счетчика с модулем 5

Среди счетчиков с произвольным модулем особое место занимают двоично-десятичные, имеющие модуль 10. В сериях ИС нередко реализуют идентичные по прочим признакам счетчики с модулями 16 и 10. Счетчик с модулем 10 нетрудно построить формально проиллюстрированным выше методом.

Схема счетчика ИЕ2, которая появилась в самых первых сериях ИС и до сих пор повторяется во всех более новых (рис. 3.54), состоит фактически из двух секций с модулями 2 и 5, представленных триггером T_0 и группой $T_1T_2T_3$, на которой собран счетчик с модулем 5. Секции можно использовать по отдельности или соединять последовательно с помощью внешней коммутации выводов для получения двоично-десятичного счетчика. Имеется сброс по конъюнкции сигналов R_0R_1 и установка в состояние 1001 по конъюнкции сигналов S_0S_1. Режимы неиспользуемых входов не показаны.

Соединение счетчиков в порядке mod2—mod5 дает двоично-десятичный счетчик с естественной последовательностью счета, который в режиме делителя частоты формирует импульсы со скважностью 5. Соединение в порядке mod5—mod2 дает, естественно, тот же модуль счета, но состояния счетчика образуют последовательность чисел 0, 1, 2, 3, 4, 8, 9, 10, 11, 12, после которой цикл повторяется. В режиме делителя частоты формируются импульсы со скважностью 2. Таким образом, разбиение счетчика на две секции предоставляет возможность получить как обычный двоично-десятичный счетчик, так и два делителя частоты на 10 — с формированием узких импульсов ($t_и$ = T/5) и симметричных импульсов ($t_и$ = T/2), где T — период повторения импульсов.

Наряду с секционированным двоично-десятичным счетчиком в сериях ИС имеются и обычные с различными сочетаниями классификационных признаков (до 5—10 вариантов).

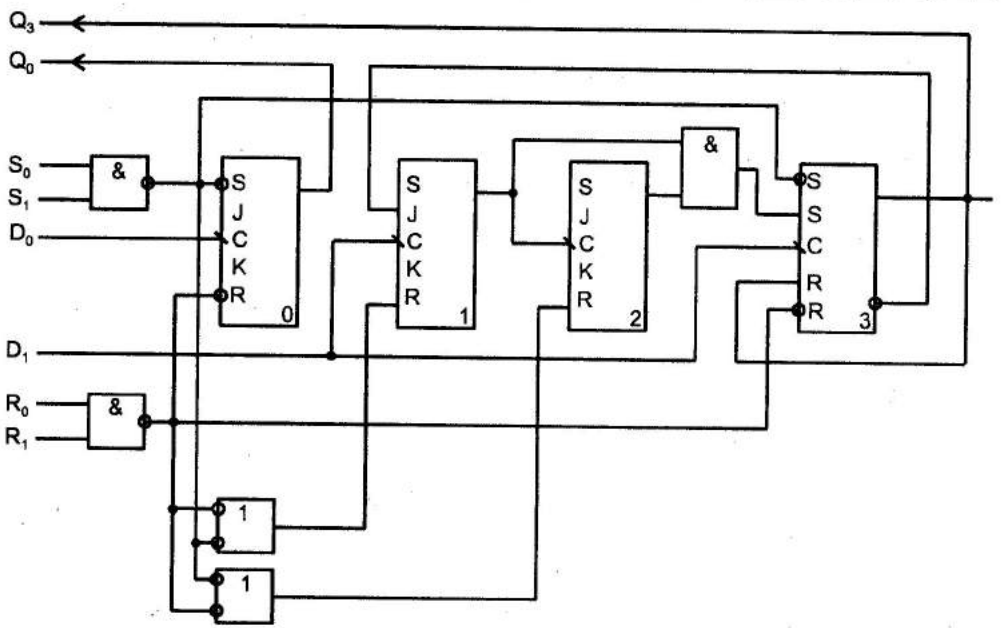


Рис. 3.54. Схема счетчика ИЕ2 серии КР1533

Построение счетчика методом управляемого сброса

Второй метод построения счетчиков с произвольным модулем — метод управляемого сброса — позволяет изменять модуль счета очень простым способом, не требующим изменений самой схемы счетчика.

Рассмотрим этот способ применительно к реализации синхронного счетчика с параллельным переносом. Функции возбуждения двоичного счетчика указанного типа, как известно, имеют вид $J_i = K_i = Q_0Q_1...Q_{i-1}$ (в младшем триггере $J_0 = K_0 = 1$). Введем в эти функции сигнал сброса R, изменив их следующим образом:

$$J_i = (Q_0Q_1...Q_{i-1})\,\overline{R},$$

$$K_i = J_i \vee R.$$

Пока сигнал сброса отсутствует (R = 0), функции J_i и K_i не отличаются от соответствующих функций двоичного счетчика. Когда сигнал R приобретает единичное значение, все функции J_i становятся нулевыми, K_i — единичными, что заставляет все триггеры сброситься по приходе следующего такта. Если сигнал R появится как следствие появления в счетчике числа M−1, то

будет реализована последовательность счета 0, 1, 2, ..., M−1, 0 ..., т. е. счетчик с модулем M.

Схемы всех разрядов счетчика с управляемым сбросом не зависят от модуля счета. Кроме разрядных схем, счетчик содержит один конъюнктор, вырабатывающий сигнал сброса при достижении содержимым счетчика значения M−1 (рис. 3.55, *а*).

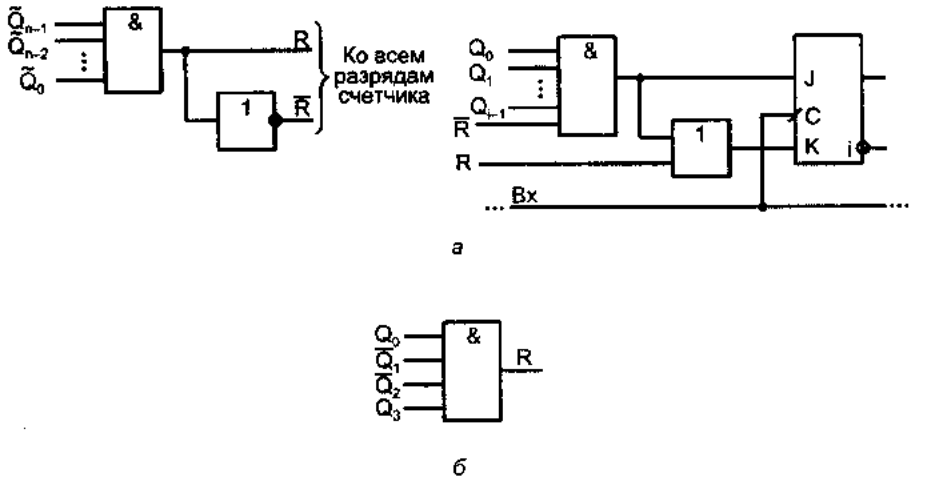


Рис. 3.55. Схема счетчика с управляемым сбросом (а) и схема выработки сигнала сброса для двоично-десятичного счетчика (б)

Если, например, имеется четырехразрядный счетчик, и на входы конъюнктора выработки сигнала сброса подключены выходы триггеров, как показано на рис. 3.55, *б*, то сброс произойдет после достижения счетчиком числа 1001 = 9, т. е. счетчик будет работать как двоично-десятичный.

§ 3.10. Счетчики с недвоичным кодированием

Наибольшее практическое значение среди счетчиков с недвоичным кодированием состояний имеют счетчики с кодом Грея, счетчики Джонсона и счетчики с кодом "1 из N".

Счетчики в коде Грея

Код Грея известен с 70-х годов XIX века, однако оказался связанным с именем Ф. Грея только в 50-х годах XX века, когда Ф. Грей применил его для

построения преобразователя угловых перемещений в цифровой код, обладающего явными преимуществами перед преобразователем с двоичным кодом. Код Грея относится к таким, в которых при переходе от любой кодовой комбинации к следующей изменяется только один разряд. В схемотехнике счетчиков это свойство устраняет одновременное переключение многих разрядов, характерное для двоичных счетчиков при некоторых переходах. Одновременное переключение многих элементов создает такие токовые импульсы в цепях питания схем, которые могут вызывать сбои в работе схемы *(см. § 1.4)*. В ряде БИС/СБИС применение двоичных счетчиков большой разрядности не разрешается, и они заменяются счетчиками с кодом Грея и последующим преобразованием кода Грея в двоичный.

Сложность счетчика с кодом Грея ненамного больше, чем сложность двоичного счетчика, преобразователь кодов также относительно прост. Нетрудно построить счетчик с кодом Грея формальным способом *(см. пример построения автомата в § 3.5)*, исходя из таблицы переходов счетчика. Последовательность кодовых комбинаций для кода Грея можно получить по соотношению $g_i = b_i \oplus b_{i+1}$, где g_i — значение разряда кода Грея; b_i — значение разряда двоичного кода, преобразуемого в код Грея. Разряд левее старшего для двоичного кода считается нулевым.

Схемы преобразователя кода Грея в двоичный приведены, в частности, в работе [36].

Счетчики в коде "1 из N"

Счетчики в коде "1 из N" находят применение в системах синхронизации, управления и других ЦУ. На их основе получают импульсные последовательности с заданными временными диаграммами. Для этого можно вначале разбить период временной диаграммы на части ("кванты"), соответствующие минимальному интервалу временной диаграммы, применив задающий генератор с частотой, равной m/T, где m — число "квантов" в периоде диаграммы T. Выходные импульсы задающего генератора затем распределяются во времени и пространстве так, что каждый "квант" появляется в свое время и в своем пространственном канале.

Счетчик в коде "1 из N" имеет один вход, на который подаются импульсы задающего генератора, и N выходов, причем первый импульс генератора передается на первый выход счетчика (канал), второй импульс во второй канал и т. д. Структура такого счетчика, называемого также распределителем тактов РТ, и временные диаграммы его работы показаны на рис. 3.56, *а, б, в*, причем временная диаграмма на рис. 3.56, *б* соответствует режиму *распределения уровней* (РУ) (паузы между активными состояниями каналов отсутствуют), а диаграмма на рис. 3.56, *в* — режиму *распределения импульсов* (РИ). Распределители импульсов не имеют самостоятельной схемотехники, они реализуются на основе распределителей уровней путем

включения в их выходные цепи конъюнкторов, на вторые входы которых подаются импульсы задающего генератора.

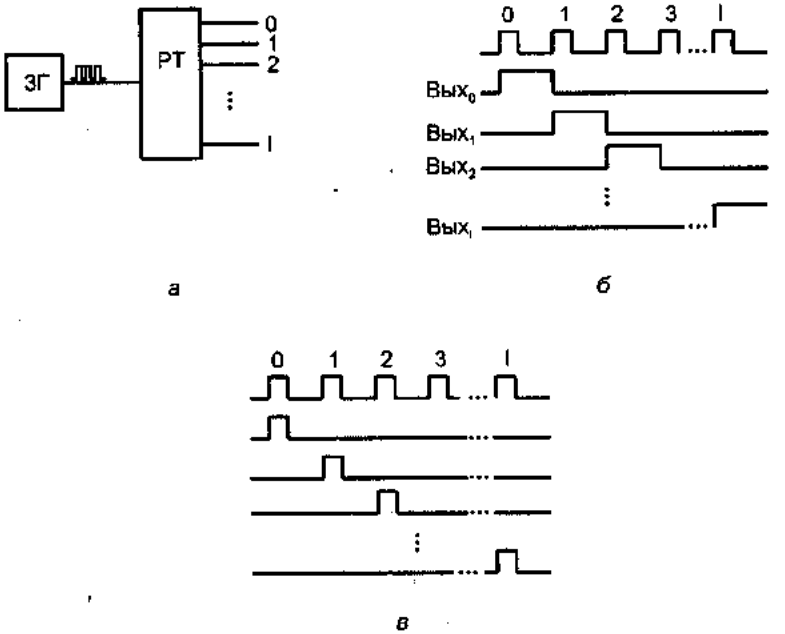


Рис. 3.56. Структура распределителя тактовых сигналов (а) и временные диаграммы распределения уровней (*б*) и импульсов (*в*)

Имея распределенные во времени и пространстве "кванты", можно по схеме ИЛИ собирать из них импульсные последовательности с необходимыми временными диаграммами. Часто нужны именно те последовательности, которые вырабатываются непосредственно распределителями тактов.

Счетчики в коде "1 из N" на основе кольцевых регистров

Распределителем тактов (РТ) является *сдвигающий регистр, замкнутый в кольцо*, если записанное в регистр слово содержит всего одну единицу. При сдвигах единица перемещается с одного выхода на другой, циркулируя в кольце. Число выходов РТ равно разрядности регистра. Недостаток схемы — потеря правильного функционирования при сбое. Если в силу каких-либо причин слово в регистре исказится, то возникшая ошибка станет постоянной. Схема не обладает свойством самозапуска.

Возможны варианты с *самовосстановлением* работы РТ на кольцевом регистре. Схема такого РТ с самовосстановлением за несколько тактов (рис. 3.57) основана на том, что на вход регистра подаются нули, пока в нем имеется хотя бы одна единица. Таким образом, лишние возникшие единицы будут устранены.

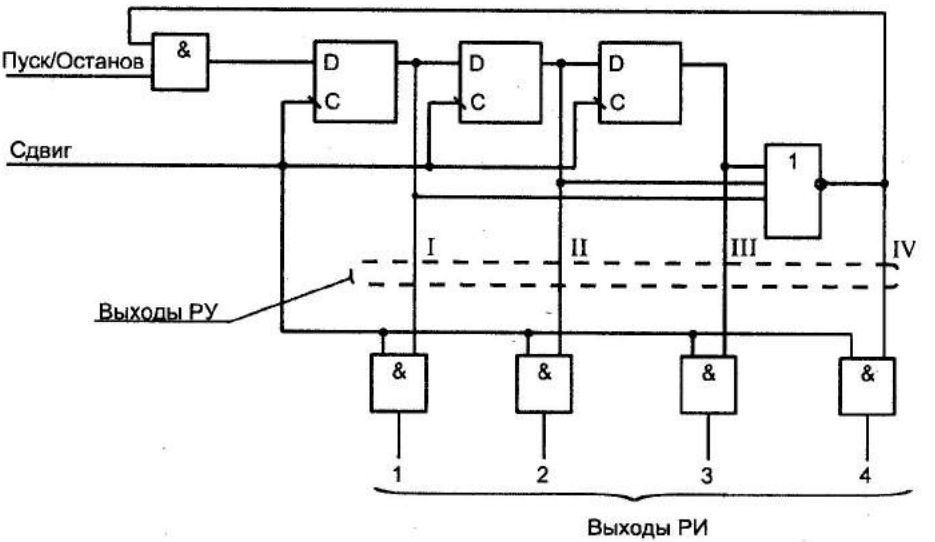
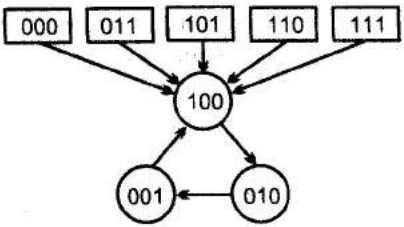


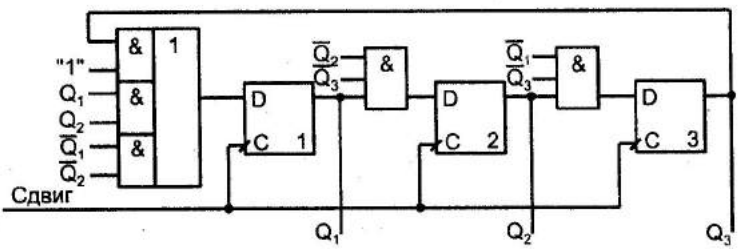
Рис. 3.57. Схема распределителя с автоматическим вхождением в рабочий цикл

Когда регистр очистится, сформируется сигнал записи единицы на его входе. Следовательно, потеря единственной единицы также будет исключена. Выход логического элемента, выполняющего самовосстановление схемы, дает еще один дополнительный канал. На схеме, приведенной на рис. 3.57, показаны также цепи пуска/останова РТ и два варианта выхода — для распределителя уровней (непосредственно с триггеров и логического элемента ИЛИ-НЕ) и распределителя импульсов (после стробирования сигналов распределителя уровней импульсами сдвига на цепочке конъюнкторов).

Можно поставить задачу более *быстрого исправления сбоев*, в том числе в ближайшем же такте. Для этого нужно задать и реализовать соответствующую диаграмму состояний распределителя. Сделаем это для трехканального распределителя. Диаграмма состояний с указанием рабочего цикла кружками и ложных состояний прямоугольниками приведена на рис. 3.58, *а*.



а



б

Рис. 3.58. Диаграмма состояний (*а*) и схема распределителя с автоматическим вхождением в рабочий цикл за один такт (*б*)

Ей соответствует следующая таблица истинности (табл. 3.15).

Таблица 3.15

Q_1	Q_2	Q_3	Q_{1H}	Q_{2H}	Q_{3H}	Q_1	Q_2	Q_3	Q_{1H}	Q_{2H}	Q_{3H}
0	0	0	1	0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0	1	1	0	0
0	1	0	0	0	1	1	1	0	1	0	0
0	1	1	1	0	0	1	1	1	1	0	0

Выбрав для построения схемы триггеры типа D, учтем, что функция возбуждения этого триггера $D = Q_H$. Исходя из таблицы, для функций $D_i = Q_{iH}$ имеем следующие соотношения:

$$D_3 = \overline{Q}_1\overline{Q}_2\overline{Q}_3,\ D_2 = Q_1\overline{Q}_2\overline{Q}_3 \text{ и } D_1 = Q_3 \vee Q_1Q_2 \vee \overline{Q}_1\overline{Q}_2.$$

Схема распределителя показана на рис. 3.58, *б*.

Распределители на кольцевых регистрах находят применение при малом числе выходных каналов, когда необходимость иметь по триггеру на каждый канал не ведет к чрезмерно большим аппаратурным затратам. Достоинством распределителей на кольцевых регистрах является отсутствие дешифраторов в их структуре и, как следствие, высокое быстродействие (задержка перехода в новое состояние равна времени переключения триггера).

Счетчики в коде "1 из N" на основе счетчиков Джонсона

Кольцевой регистр с перекрестной обратной связью (счетчик Джонсона, счетчик Мебиуса, счетчик Либау-Крейга) обладает обратной связью замкнутой на первый триггер от инверсии выходного сигнала (рис. 3.59, *а*). Он имеет 2n состояний, т. е. при той же разрядности вдвое больше, чем обычный кольцевой регистр. В то же время выход счетчика Джонсона представлен не в коде "1 из N", что требует преобразования кодов для получения выходов распределителя тактов. Такие преобразователи очень просты, что и обуславливает применение счетчиков Джонсона в составе распределителей.

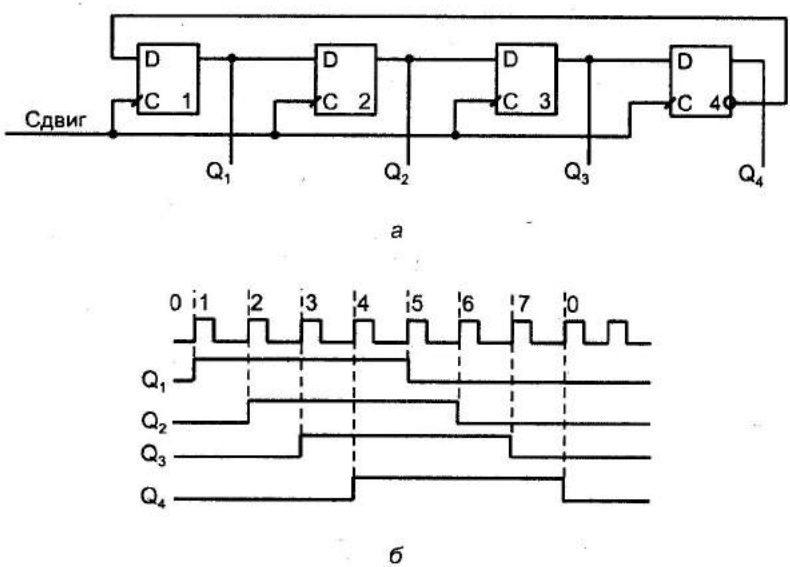


Рис. 3.59. Схема счетчика Джонсона (*а*) и временные диаграммы его работы (*б*)

Показанный на рисунке четырехразрядный счетчик Джонсона при начальном нулевом состоянии работает следующим образом. Первый тактовый

импульс "Сдвиг" установит первый триггер в единичное состояние ($Q_1 = 1$), т. к. $\overline{Q}_4 = 1$, в остальных разрядах будут нули как результат сдвига нулей от соседних слева разрядов. Второй импульс "Сдвиг" сохраняет единичное состояние первого триггера, т. к. по-прежнему $\overline{Q}_4 = 1$. Второй же разряд окажется в единичном состоянии ($Q_2 = 1$), поскольку примет единицу от первого триггера. Остальные разряды будут нулевыми. Последующие сдвиги приведут к заполнению единицами всех разрядов счетчика, т. е. "волна единиц", распространяясь слева направо, приведет счетчик в состояние 1111. Следующий импульс сдвига установит первый разряд в нуль, т. к. теперь $\overline{Q}_4 = 0$. Этим начинается процесс распространения "волны нулей". После восьми импульсов повторится состояние 0000, с которого было начато рассмотрение работы счетчика. Временные диаграммы описанных процессов показаны на рис. 3.59, *б*.

Особенность рассмотренной схемы — четное число состояний при любом n (2n — всегда число четное). Обычный кольцевой регистр такого ограничения не имеет.

Преобразование выходного кода счетчика Джонсона в код "1 из N" требует добавления всего одного двухвходового элемента И либо И-НЕ для каждого выхода распределителя тактов. Принцип дешифрации состоит в выявлении положения характерной координаты временной диаграммы — границы между зонами единиц и нулей (табл. 3.16).

Таблица 3.16

Номер состояния	Q_1	Q_2	Q_3	Q_4	Номер состояния	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0	4	1	1	1	1
1	1	0	0	0	5	0	1	1	1
2	1	1	0	0	6	0	0	1	1
3	1	1	1	0	7	0	0	0	1

В двух случаях (для слов, состоящих только из нулей или только из единиц) состояние выявляется анализом крайних разрядов. В остальных случаях анализируются разряды на границе зоны единиц и нулей.

Как видно из таблицы, преобразование выходного кода счетчика Джонсона в код "1 из N" осуществляется согласно выражениям

$$F_0 = \overline{Q}_1\overline{Q}_4,\ F_1 = Q_1\overline{Q}_2,\ F_2 = Q_2\overline{Q}_3,\ F_3 = Q_3\overline{Q}_4,$$

$$F_4 = Q_1Q_4,\ F_5 = \overline{Q}_1Q_2,\ F_6 = \overline{Q}_2Q_3,\ F_7 = \overline{Q}_3Q_4,$$

где F_i (i = 0...7) -- выходы распределителя тактов.

По полученным выражениям строится дешифратор. Рассмотрим дешифратор с элементами И-НЕ (с инверсными выходами). В таком дешифраторе можно дополнительно принять меры по предотвращению перекрытий импульсов в соседних каналах, возможных из-за различных задержек элементов. Используя элементы с тремя входами и "косыми связями" (рис. 3.60, *а*), можно запретить начало импульса в последующем канале до его завершения в предыдущем.

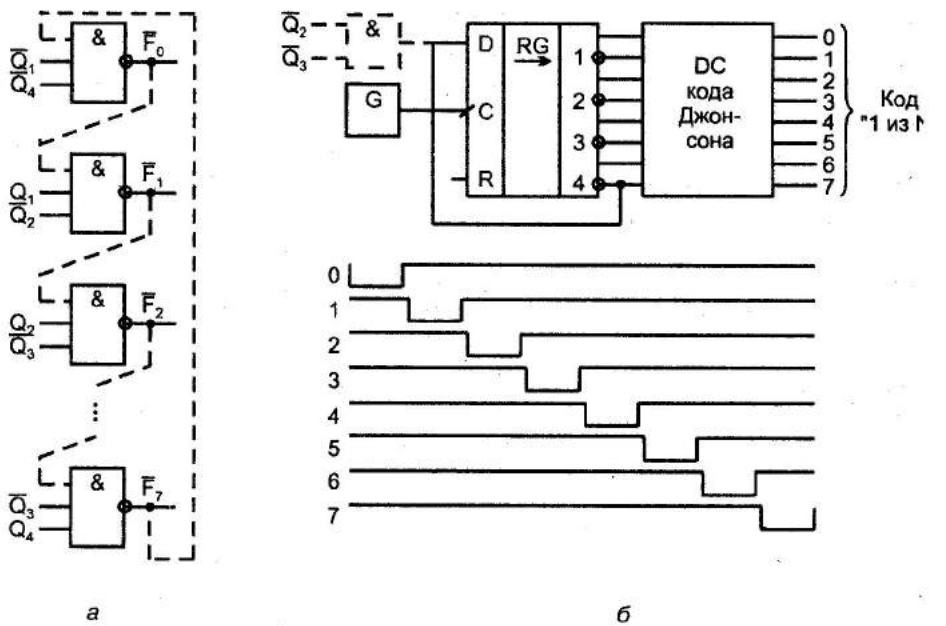


Рис. 3.60. Схемы преобразования кода Джонсона в код "1 из N" (*а*) и распределителя на основе счетчика Джонсона (*б*)

Распределитель тактов в целом (рис. 3.60, *б*) имеет выходные сигналы в коде "1 из N".

Для схем со счетчиками Джонсона могут возникнуть вопросы преодоления ограничения обязательной четности числа состояний счетчика и обеспечения автоматического вхождения его в рабочий цикл (свойства самозапуска).

Первую задачу можно решить в рамках подхода, применявшегося при построении счетчиков с произвольным модулем, т. е. исключением одного "лишнего" состояния.

Получить *схему с исключенным состоянием* можно уже не раз показанным способом, переходя от таблицы функционирования к функциям возбуждения триггеров и далее к схеме. Однако в данном случае нетрудно сократить этот путь, пользуясь простыми рассуждениями. Пусть исключению подлежит состояние 11...11. Чтобы его исключить, нужно перейти к следующему состоянию не от состояния "все единицы", а от предыдущего состояния 11...10, которое создает единицу в предпоследнем разряде счетчика, т. е. нуль на инверсном выходе этого разряда. Подавая этот нулевой сигнал на вход счетчика вместе с основным сигналом обратной связи через конъюнктор (показан на рис. 3.60, *б* штриховой линией), исключим состояние 11...11 и получим счетчик с нечетным числом состояний $2n-1$.

Задача *обеспечения вхождения распределителя на основе счетчика Джонсона в рабочий цикл* связана с тем, что базовая схема, рассмотренная ранее, свойством самозапуска не обладает. Например, распределитель с трехразрядным счетчиком Джонсона имеет общее число возможных состояний $2^3 = 8$, а число состояний в рабочем цикле $2 \cdot 3 = 6$. Неиспользуемыми являются два состояния: 010 и 101. Нетрудно видеть, что из состояния 010 счетчик перейдет в состояние 101, а из состояния 101 в состояние 010. Таким образом, наряду с замкнутым рабочим циклом существует и замкнутый цикл из двух неиспользуемых состояний, попав в который, схема без постороннего воздействия не сможет перейти в рабочий цикл.

Чтобы придать схеме свойство самозапуска, нужно модифицировать сигнал обратной связи, поступающий на вход счетчика. Понятно, что это можно сделать многими путями, поскольку траектория перехода из замкнутого цикла неиспользуемых состояний в рабочий неоднозначна. Одной из возможностей является выработка сигнала обратной связи согласно выражению

$$F_{oc} = D_1 = \overline{Q}_n \vee \overline{Q_{n-1} \cdots Q_2 Q_1}.$$

Распределители на основе счетчиков Джонсона характеризуются небольшими аппаратурными затратами (1/2 триггера и один двухвходовой вентиль на канал) и достаточно высоким быстродействием (время установления — сумма задержек переключения триггера и вентиля). Счетчики Джонсона реализованы, в частности, в сериях элементов типа КМОП (микросхемы ИЕ9 и ИЕ19 серии К561 и др.), причем одной из причин их применения является отсутствие импульсов помех в выходном напряжении и пониженный уровень токовых импульсов в цепях питания, создаваемых микросхемами. Распределитель в целом реализован в ИС К561ИЕ8.

Следует заметить, что распределители могут быть получены без применения специализированных схем в виде *сочетания обычного двоичного счетчика и дешифратора*. Такое решение наиболее очевидно. При большем числе выходных каналов эта структура может выигрывать у других, но при малом числе каналов преимущество по аппаратурной сложности и быстродействию, как правило, оказывается на стороне вариантов с кольцевыми регистрами или счетчиками Джонсона.

§ 3.11. Полиномиальные счетчики

Полиномиальные счетчики (сдвигающие регистры с линейными обратными связями, генераторы псевдослучайных последовательностей, линейные автоматы на основе сдвигающих регистров) используются в устройствах тестового диагностирования цифровых устройств, для решения математических задач методом Монте-Карло, при моделировании систем с учетом случайного разброса их параметров и в ряде других случаев, например, для предоставления слова тому или иному оратору при ограниченности возможностей заседания, как это было на съездах народных депутатов СССР после начала перестройки в период бурных дебатов.

Ряд названий, относящихся к полиномиальным счетчикам, связан с понятием линейных комбинационных функций, линейных автоматов и т. п. По определению к линейным переключательным функциям относятся те, для которых полином Жегалкина имеет степень не выше первой. Такие функции содержат конъюнкции единичной длины и могут быть представлены в виде

$$F(x_1, ..., x_n) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus ... \oplus a_nx_n,$$

где a_i принимают значения 0 или 1 ($i = 0...n$).

Автомат *линеен*, если схемы выработки функций выходов и функций возбуждения D-триггеров, образующих память автомата, линейны (эти схемы составлены только из сумматоров по модулю 2).

Возможная реализация автономного линейного автомата — сдвигающий регистр с обратными связями через сумматоры по модулю 2. Такие автоматы применяются в генераторах псевдослучайных последовательностей и устройствах тестового контроля ЦУ, где они обеспечивают получение циклических кодов.

Случайные сигналы могут быть аналоговыми или цифровыми. Цифровой сигнал при этом представляется случайной последовательностью, элементами которой могут быть символы 0 и 1 или многоразрядные числа. Первому варианту обычно присваивается наименование случайной последовательности, второму — случайных чисел.

Случайные сигналы характеризуются законами распределения, среди которых важное место занимает равномерный закон, т. к. сигналы с таким законом распределения имеют не только непосредственное применение, но и служат для получения сигналов с другими законами распределения путем определенной математической обработки.

Истинно случайные последовательности и числа генерируются на основе физических явлений (флуктуационных шумов в электронных приборах, радиоактивного излучения и др.), что сложно реализуется и не обеспечивает стабильности статистических характеристик.

При генерации *псевдослучайных последовательностей и чисел* сигнал детерминирован, но его характеристики близки к характеристикам истинно случайного сигнала. Генерация псевдослучайных сигналов проще и надежнее.

Структура сдвигающего регистра с линейной обратной связью показана на рис. 3.61. Выходная последовательность снимается с входа триггера D_1, она же повторяется со сдвигами в других точках тракта, образованного D-триггерами. Аргументами линейной функции являются величины, различающиеся только сдвигами на то или иное число тактов, что отмечается показателем степени при аргументе x. Структуре схемы ставится в соответствие полином

$$F(x) = x^n + \alpha_1 x^{n-1} + \ldots + \alpha_{n-1} x + 1.$$

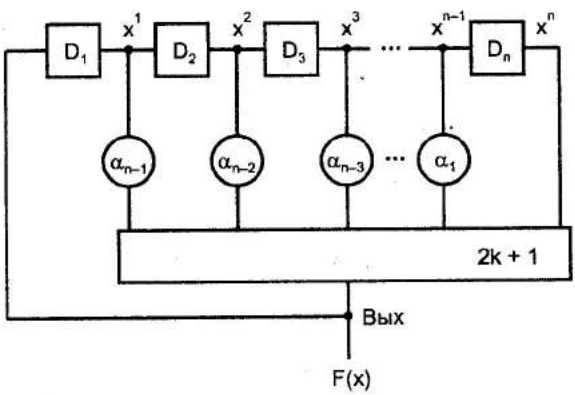


Рис. 3.61. Структура сдвигающего регистра с линейной обратной связью

Начав с любого исходного состояния, можно вычислить последующие. На входе левого триггера окажется значение функции F(x), соответствующее исходному состоянию, в других — результат сдвига на один разряд. Как только опять возникает состояние, идентичное исходному, все начнет по-

вторяться, т. е. устройство работает периодично. Период последовательности зависит от коэффициентов α_i. Обычно желателен максимальный период. Автомат с n-триггерами может иметь 2^n состояний. В данном автомате состояние всех нулей должно быть исключено, т. к. из него схема никогда не сможет выйти. Поэтому для данного автомата максимальный период составит 2^n-1, а соответствующая ему последовательность называется последовательностью максимальной длины или *М-последовательностью.*

К М-последовательности приводят многие варианты схемы. Их поиск основан на изучении полинома, соответствующего схеме. Чтобы генерировалась М-последовательность, полином должен быть неприводимым и примитивным. Таких полиномов много: при n = 8 их 16, при n = 16 уже 2048 и т. д. Среди множества полиномов целесообразно отыскать такие, у которых минимальное число ненулевых коэффициентов α_i, поскольку это упрощает схему.

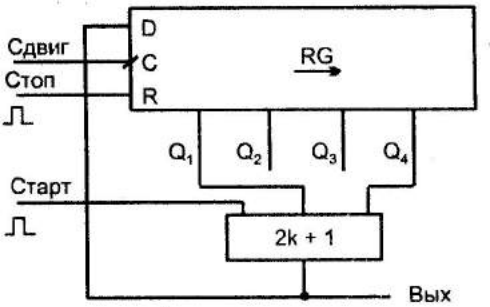
Для генерации М-последовательностей схемой с одним элементом сложения по модулю 2 рассчитаны таблицы. Элемент имеет два входа, один из которых подключен к выходу последнего триггера регистра (иначе его наличие в схеме теряет смысл), а второй подключен к разряду с номером i. Если перевести вход элемента с выхода разряда номер i на выход разряда номер n−i, то будет генерироваться последовательность с обратным порядком следования двоичных символов, поэтому в приводимой таблице (табл. 3.17) указаны номера разрядов i или n−i.

Таблица 3.17

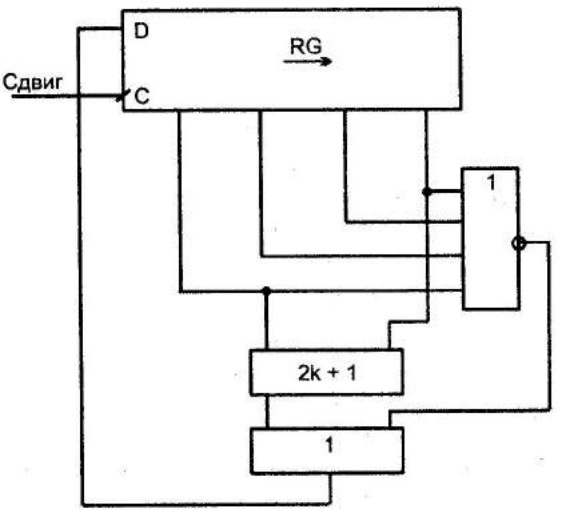
n	4	5	6	7	9	10	11	15	17	18	20
i или n–i	1	2	1	1,3	4	3	2	1, 4, 7	3	7	3

Схемы генераторов псевдослучайной последовательности (ГПСП)

Схема ГПСП, соответствующего первой строке таблицы (рис. 3.62, *а*), останавливается сбросом всех триггеров и запускается импульсом старта, записывающим единицу через элемент сложения по модулю 2 в левый триггер. На рис. 3.62, *б* показана схема такого же ГПСП, но обладающего свойством самозапуска. С выхода любого триггера ГПСП можно снять последовательность 111101011001000, соответствующую $M = 2^4 - 1 = 15$. Для схемы ГПСП с 20 разрядами M = 1048575. Если длина последовательности превышает емкость памяти системы, то псевдослучайную последовательность не отличить от случайной.



а



б

Рис. 3.62. Схемы четырехразрядных генераторов М-последовательностей: запускающегося стартовым импульсом (*а*) и самозапускающегося (*б*)

Генерируемые последовательности имеют число единиц, на единицу превышающее число нулей (следствие исключения состояния "все нули"); группы одинаковых символов появляются в них с той же частотой, что и в случайной последовательности равновероятных двоичных символов; любой набор из $m \leq n$ смежных элементов встречается с равной вероятностью (за исключением набора из m нулей); нормированная автокорреляционная функция качественно подобна этой функции белого шума $R(\tau) \approx 0$ при больших M и τ, не кратных M.

Генераторы псевдослучайных чисел (ГПСЧ)

Эти генераторы строят по последовательному, параллельному и смешанному способам. В первом случае число (слово) образуется за несколько тактов. Из образованной в регистре последовательности для получения m-разрядного слова получают результат путем S сдвигов, где $S \geq m$, что дает отсутствие корреляции между соседними словами. Период последовательности слов равен наименьшему общему кратному чисел S и M. Для получения максимального периода число S выбирается взаимно-простым к M.

В генераторах параллельного типа псевдослучайные числа генерируются в каждом такте. Очевидным решением было бы использование m генераторов псевдослучайных двоичных последовательностей для образования отдельных разрядов случайных чисел, однако существуют более простые решения.

Линейные автоматы на основе сдвигающих регистров используются также в *сигнатурных анализаторах*, являющихся средствами тестового диагностирования цифровых устройств, требующих подачи на них специальных воздействий, с помощью которых проверяется правильность работы устройства. С ростом сложности устройств длина тестовых последовательностей и объем оборудования, обеспечивающего генерацию тестов и анализ результатов, увеличиваются, и возникает задача сжатия информации при тестировании. Эта задача решается, в частности, с использованием линейных автоматов на основе регистров сдвига в сигнатурных анализаторах.

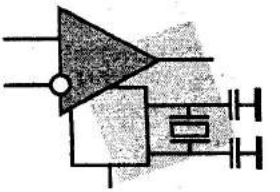
Если в схемах ГПСП ввести дополнительно внешний вход на элементы 2k + 1, то получится устройство для аппаратного выполнения операции деления полиномов по правилам арифметики по модулю 2. При этом входная последовательность, состоящая из нулей и единиц, трактуется как полином, содержащий те степени переменной x, которым соответствуют единицы. Например, последовательность 10010010 соответствует полиному $x^7 \oplus x^4 \oplus x$. Этот полином делится на так называемый порождающий полином, определяемый структурой схемы, как показано ранее. В результате деления в регистре записывается остаток R(x).

Проверка логической схемы производится следующим образом. На ее входы от генератора псевдослучайных чисел подается известная последовательность. Выход схемы подключается к узлу деления полиномов. После деления в регистре остается остаток (сигнатура). Сигнатура для исправной схемы известна. Сравнение полученного остатка с этой сигнатурой (эталонным остатком) позволяет сделать заключение о правильности работы схемы. С помощью специальных процедур наряду с обнаружением ошибок можно производить и их поиск.

Сдвигающие регистры с линейными обратными связями, выполняющие операции над полиномами, применяются также для построения и анализа циклических кодов, применяемых для обнаружения и коррекции ошибок в цифровых устройствах.

Литература к главе: [2], [3], [9], [12], [19], [25], [26], [30], [33], [38], [44], [45], [46], [47], [49].

Глава 4



Запоминающие устройства

§ 4.1. Основные сведения. Система параметров. Классификация

Запоминающие устройства (ЗУ) служат для хранения информации и обмена ею с другими ЦУ. Микросхемы памяти в общем объеме выпуска ИС занимают около 40% и играют важнейшую роль во многих системах различного назначения. Микросхемы и системы памяти постоянно совершенствуются как в области схемотехнологии, так и в области развития новых архитектур. В настоящее время созданы и используются десятки различных типов ЗУ.

Важнейшие параметры ЗУ находятся между собой в противоречии. Так, например, большая информационная емкость не сочетается с высоким быстродействием, а быстродействие в свою очередь не сочетается с низкой стоимостью. Поэтому системам памяти свойственна *многоступенчатая иерархическая структура*, и в зависимости от роли того или иного ЗУ его реализация может быть существенно различной.

В развитой иерархии памяти ЭВМ можно выделить следующие уровни:

- *регистровые ЗУ*, находящиеся в составе процессора, благодаря которым уменьшается число обращений к другим уровням памяти, реализованным вне процессора и требующим большего времени для операций обмена информацией;
- *кэш-память*, служащая для хранения копий информации, используемой в текущих операциях обмена. Работа процессора с кэш-памятью высокого быстродействия повышает производительность ЭВМ;
- *основная память* (оперативная, постоянная, полупостоянная), работающая в режиме обмена с процессором и по возможности согласованная с ним по быстродействию. Исполняемый в текущий момент фрагмент программы обязательно находится в основной памяти;
- *специализированные* виды памяти, характерные для некоторых специфических архитектур (многопортовые, ассоциативные, видеопамять и др.);

- *внешняя память*, хранящая большие объемы информации. Эта память обычно реализуется на основе устройств с подвижным носителем информации (магнитные и оптические диски, магнитные ленты и др.). В настоящем пособии устройства внешней памяти не рассматриваются.

Важнейшие параметры ЗУ

Информационная емкость — максимально возможный объем хранимой информации. Выражается в битах или словах (в частности, в байтах). Бит хранится запоминающим элементом (ЗЭ), а слово — запоминающей ячейкой (ЗЯ), т. е. группой ЗЭ, к которым возможно лишь одновременное обращение. Добавление к единице измерения множителя "К" (кило) означает умножение на $2^{10} = 1024$, а множителя "М" (мега) — умножение на $2^{20} = 1\,048\,576$. Производители микросхем памяти обычно оценивают их емкости в битах, а системотехники — в байтах или словах.

Организация ЗУ — произведение числа хранимых слов на их разрядность. Видно, что это произведение выражает информационную емкость ЗУ, однако при одной и той же информационной емкости организация ЗУ может быть различной, так что организация является самостоятельным важным параметром, выражаемым парой чисел. Примеры организации памяти: 32×8, 128К×8, 1М×1.

Быстродействие ЗУ оценивают временами считывания, записи, длительностями циклов чтения/записи и другими параметрами. *Время считывания* — интервал между моментами появления сигнала чтения и слова на выходе ЗУ. *Время записи* — интервал после появления сигнала записи, достаточный для установления ЗЯ в состояние, задаваемое входным словом. Минимально допустимый интервал между последовательными повторными операциями чтения или записи образует соответствующий *цикл*. Длительности циклов могут превышать времена чтения или записи, т. к. после этих операций до начала следующей может потребоваться время для восстановления необходимого начального состояния ЗУ.

Времена чтения, записи и длительности циклов — традиционные параметры, достаточные для оценки быстродействия простых структур ЗУ. Для многих современных ЗУ они должны быть дополнены новыми. Причиной является более сложный характер доступа к хранимым данным, когда обращение к первому слову некоторой группы слов (страницы, пакета) требует большего времени, чем обращение к последующим. Для таких режимов вводят параметры *времени доступа при первом обращении* (Latency) и *темпа передач* для последующих слов пакета. Темп передач в свою очередь оценивается двумя значениями — *предельным* (внутри пакета) и *усредненным* (с учетом Latency). С уменьшением пакета усредненный темп снижается, все более отличаясь от предельного.

Применительно к ЗУ используется также параметр, называемый *полосой пропускания* или *производительностью*, и определяемый как произведение числа считываемых (или записываемых) в секунду слов на их разрядность. Например, ЗУ с темпом передачи слов 50 МГц при их разрядности 8 бит имеет полосу пропускания (производительность) 400 Мбит/с.

Указанные параметры не исчерпывают всего множества сведений о быстродействии разных типов ЗУ, имеющих свою специфику. Для некоторых структур в число параметров, характеризующих быстродействие ЗУ, входят еще несколько показателей, о которых говорится далее в этой главе при описании конкретных разновидностей микросхем памяти.

Перечисленные динамические параметры являются *эксплуатационными* (измеряемыми). Кроме них существует ряд *режимных* параметров, обеспечение которых необходимо для нормального функционирования ЗУ, поскольку оно имеет несколько сигналов управления, сигналы адресации данных и самих данных и для них должно быть обеспечено определенное взаимное положение во времени. Для этих сигналов задаются длительности и ограничения по взаимному положению во времени.

Важным для микросхем памяти является *свойство энергонезависимости*, т. е. способность ЗУ сохранять данные при отключении напряжения питания. Энергонезависимость может быть *естественной*, т. е. присущей самим ЗЭ, или *искусственной*, достигаемой введением резервных источников питания, автоматически подключаемых к накопителю ЗУ при снятии основного питания или же дополнением схемы ЗУ специальными вспомогательными энергонезависимыми элементами памяти.

Кроме отмеченных параметров ЗУ характеризуются и другими (уровни напряжений, токи, емкости выводов, температурный диапазон и т. д.), которые в силу своей традиционности не требуют специального рассмотрения.

Входные и выходные сигналы ЗУ

Один из возможных наборов сигналов ЗУ (рис. 4.1, *а*) включает следующие сигналы (сигналы показаны для операции чтения, ЗУ обозначено буквой M от слова Memory):

- A — адрес, разрядность которого n определяется числом ячеек ЗУ, т. е. максимально возможным числом хранимых в ЗУ слов. Число ячеек ЗУ выражается целой степенью двойки. Адрес является номером ячейки, к которой идет обращение. Очевидно, что разрядность адреса n и число ячеек N связаны соотношениями $n = \log_2 N$ и $N = 2^n$. Например, ЗУ с информационной емкостью 64K слов имеет 16-разрядные адреса, выражаемые словами формата $A = A_{15}A_{14}A_{13}...A_0$;
- CS (Chip Select) или CE (Chip Enable), который разрешает или запрещает работу данной микросхемы;

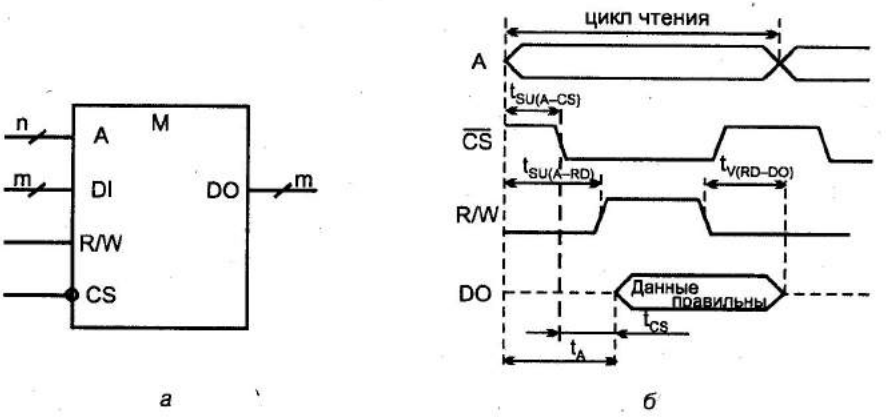


Рис. 4.1. Типичные сигналы простейшего ЗУ (*а*) и их временные диаграммы (*б*)

- R/W — (Read/Write) задает выполняемую операцию (при единичном значении — чтение, при нулевом — запись);
- DI и DO (Data Input) и (Data Output) — шины входных и выходных данных, разрядность которых m определяется организацией ЗУ (разрядностью его ячеек). В некоторых ЗУ эти линии объединены. При этом единая двунаправленная шина передачи данных обозначается как DIO (Data Input/Output).

Требования к взаимному временному положению двух сигналов (A и B) задаются временами предустановки, удержания и сохранения.

Время предустановки сигнала A относительно сигнала B $t_{SU\,(A-B)}$ есть интервал между началами обоих сигналов.

Время удержания $t_{H\,(A-B)}$ — это интервал между началом сигнала A и окончанием сигнала B.

Время сохранения $t_{V\,(A-B)}$ — интервал между окончанием сигнала A и окончанием сигнала B.

Длительности сигналов обозначаются как t_W (индекс от слова Width — ширина).

Для ЗУ характерна такая последовательность сигналов. Прежде всего, подается адрес, чтобы последующие операции не коснулись какой-либо ячейки, кроме выбранной. Затем разрешается работа микросхемы сигналом CS (CE) и подается строб чтения/записи R/W (взаимное положение сигналов CS и R/W для разных ЗУ может быть различным). Если задана, например, операция чтения, то после подачи перечисленных сигналов ЗУ готовит данные для чтения, что требует определенного времени. Задний фронт сигнала R/W, положение которого во времени должно обеспечивать установление правильных данных на выходе ЗУ, считывает данные.

Пример временной диаграммы для рассмотренного набора сигналов ЗУ и операции чтения приведен на рис. 4.1, *б*.

Индексом A (от слова Access) обозначаются времена доступа — интервалы времени от появления того или иного управляющего сигнала до появления информационного сигнала на выходе. *Время доступа относительно сигнала адреса* обозначается, если следовать правилу, как $t_{A(A)}$, но часто просто как t_A. Аналогично этому, *время доступа относительно сигнала CS*, т. е. $t_{A(CS)}$ часто обозначается просто как t_{CS}. Время t_A называют также временем выборки, а время t_{CS} — временем выбора.

Классификация современных ЗУ

Для классификации ЗУ (рис. 4.2, *а*) важнейшим признаком является способ доступа к данным.

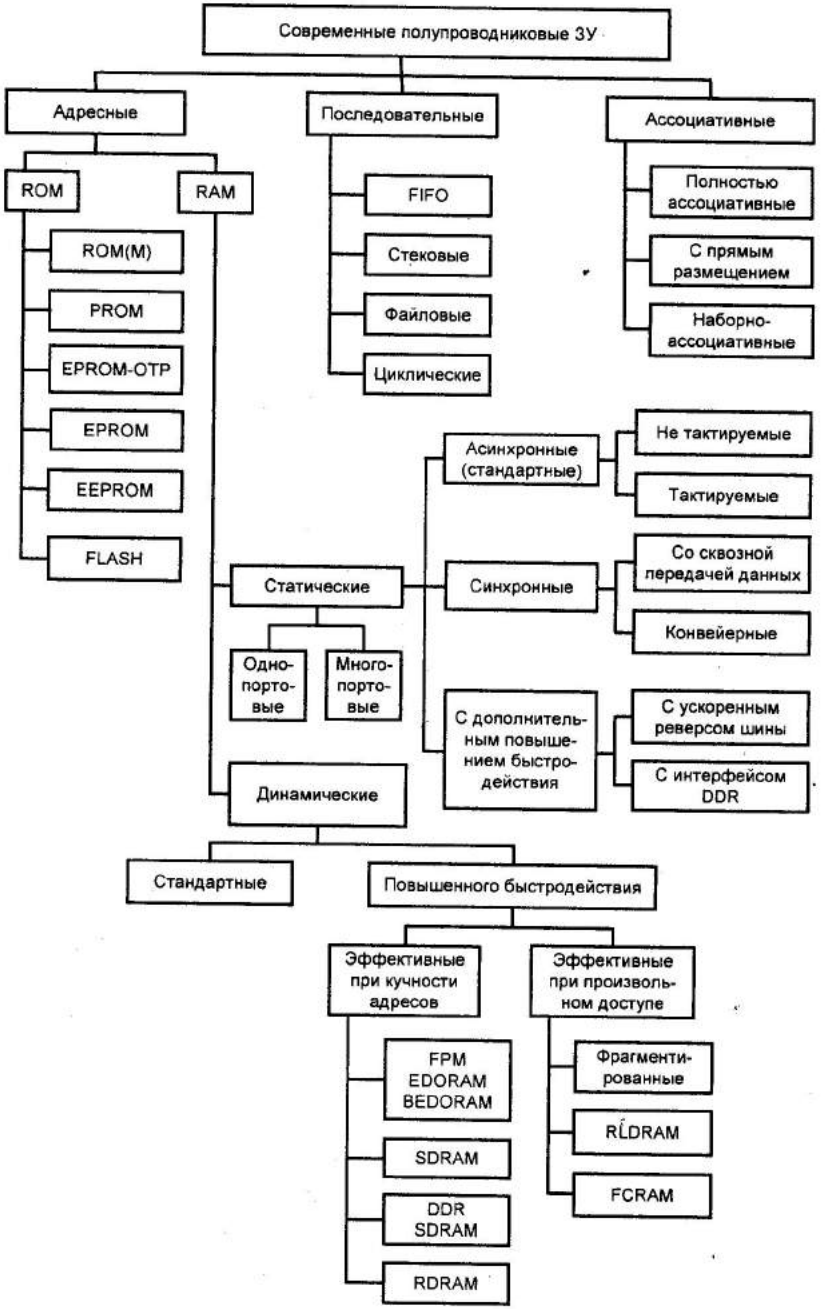
При *адресном* доступе код на адресных входах указывает ячейку, с которой ведется обмен. Все ячейки адресной памяти в момент обращения равнодоступны. Эти ЗУ наиболее разработаны, и другие виды памяти часто строят на основе адресной с соответствующими модификациями.

Адресные ЗУ делятся на RAM (Random Access Memory) и ROM (Read-Only Memory). Русские синонимы термина RAM: ОЗУ (оперативные ЗУ) или ЗУПВ (ЗУ с произвольной выборкой). Оперативные ЗУ хранят данные, участвующие в обмене при исполнении текущей программы, которые могут быть изменены в произвольный момент времени. Современные ОЗУ, как правило, не обладают энергонезависимостью (этим свойством обладают новые перспективные варианты ОЗУ, которые, возможно, вскоре начнут заменять существующие).

В ROM (русский эквивалент — ПЗУ, т. е. постоянные ЗУ) содержимое либо вообще не изменяется, либо изменяется, но редко и в специальном режиме. Для рабочего режима это *"память только для чтения"*.

Постоянная память типа Mask ROM, обозначенная как ROM(M), программируется при изготовлении методами интегральной технологии с помощью масок. На русском языке ее можно назвать памятью типа ПЗУМ (ПЗУ масочные). Для потребителя это в полном смысле слова постоянная память, т. к. изменить ее содержимое он не может.

В следующих четырех разновидностях ROM в обозначениях присутствует буква P (от Programmable). Это *программируемая пользователем память* (в русской терминологии ППЗУ). В память типов PROM и EPROM-OTP содержимое записывается однократно (OTP означает One Time Programmable). В ЗУ типов EPROM, EEPROM и FLASH содержимое может быть изменено путем стирания старой информации и записи новой. В памяти EPROM (Erasable Programmable ROM) стирание выполняется облучением кристалла ультрафиолетовыми лучами, ее русское название РПЗУ-УФ (репрограммируемое ПЗУ с УФ-стиранием).



а

Рис. 4.2. Классификация современных полупроводниковых ЗУ (*а*)

б

Рис. 4.2. Классификация перспективных ЗУ (*б*)

В EEPROM или, иначе, E²PROM (Electrically Erasable Programmable ROM) стирание производится электрическими сигналами, ее русское название РПЗУ-ЭС (репрограммируемое ПЗУ с электрическим стиранием) или ЭСППЗУ (электрически стираемое программируемое ПЗУ). Запись данных в EPROM, EEPROM и FLASH производится электрическими сигналами.

Программирование PROM, EPROM и EEPROM производится в обычных лабораторных условиях. Для ЗУ типов PROM и EPROM это делается с помощью программаторов. Для EEPROM возможно также использование специальных режимов для программирования без изъятия микросхемы из устройства, в котором она используется. Запоминающие элементы памяти типа FLASH принципиально подобны применяемым в EPROM и EEPROM, но эта память имеет структурные и технологические особенности, позволяющие выделить ее в отдельный вид.

RAM делятся на статические и динамические. В статических RAM запоминающими элементами являются триггеры, сохраняющие свое состояние, пока схема находится под питанием и нет новой записи данных. В динамических RAM данные хранятся в виде зарядов конденсаторов, образуемых элементами МОП-структур. Саморазряд конденсаторов ведет к разрушению данных, поэтому они должны периодически (каждые несколько миллисекунд) регенерироваться, что усложняет эксплуатацию ЗУ. В то же время плотность упаковки элементов динамической памяти в несколько раз превышает плотность упаковки, достижимую в статических RAM, поэтому динамические ЗУ имеют намного более высокую информационную емкость и в несколько раз дешевле более быстродействующих статических.

Разработаны также ЗУ с динамическими запоминающими элементами, имеющие внутреннюю встроенную систему регенерации, у которых внешнее поведение становится аналогичным поведению статических ЗУ. Такие ЗУ иногда называют *квазистатическими* (одна из фирм называет их даже статическими).

Статические ОЗУ в английской и международной терминологии называются SRAM (Static RAM), а динамические — DRAM (Dynamic RAM).

Статические ОЗУ разделены в классификации на *асинхронные* и *синхронные*. Асинронные ОЗУ названы также *стандартными*, т. к. до недавнего времени они были практически единственными представителями статических микросхем памяти и наиболее привычны для потребителя. В асинхронных ЗУ после произвольного по времени обращения к памяти до выдачи данных проходит определенное время, которое является параметром самой памяти, не связанным с параметрами системы синхронизации процессора. Ввиду отсутствия увязки моментов обращения к памяти и моментов выработки ею готовых данных с синхросигналами процессора могут возникать дополнительные задержки обмена данными между ЗУ и процессором.

Асинхронные статические ОЗУ можно разделить на *нетактируемые* и *тактируемые*. В нетактируемых сигналы управления могут задаваться как импульсами, так и уровнями. В тактируемых ЗУ некоторые сигналы обязательно должны быть импульсными (например, сигнал разрешения работы CS в каждом цикле должен переходить из пассивного состояния в активное).

В синхронных ОЗУ длительности этапов работы памяти жестко связаны с синхросигналами системы, и это позволяет исключить неоправданные потери времени при обмене данными между памятью и процессором, а также организовать конвейерную обработку данных. Таким образом, синхронность памяти является средством повышения ее быстродействия. Это важный способ повышения быстродействия, применяемый как в статических, так и в динамических микросхемах памяти (в динамических ОЗУ синхронные варианты появились раньше, чем в статических). Применение синхронных ОЗУ не является единственным способом повышения их быстродействия. Среди других методов повышения быстродействия статических ОЗУ можно назвать *ускорение реверса шины* при переходе от передачи данных в одном направлении к другому и *использование интерфейса DDR (см. главу 1)*.

Статические ОЗУ выполняются как *однопортовые* (обычные) и *многопортовые*. Многопортовые ЗУ специализированы для определенных применений. В них возможны одновременные обращения более чем к одной ячейке, например, в двупортовых ЗУ возможно считывание информации из одной ячейки и одновременная запись в другую. Подобные режимы полезны при разделении памяти между двумя или более абонентами.

Динамические ЗУ характеризуются наибольшей информационной емкостью и невысокой стоимостью, поэтому именно они используются как основная память ЭВМ. Базовая структура динамических ЗУ названа стандартной. Поскольку желательно получить от основной памяти ЭВМ максимально возможное быстродействие, разработаны многочисленные способы его повышения. Соответствующие архитектуры перечислены в классификации. *Подробнее эти архитектуры рассмотрены в § 4.11.*

Статические ЗУ в 4—5 раз дороже динамических и приблизительно во столько же раз меньше по максимально достижимой информационной ем-

кости. Их достоинством является высокое быстродействие, а типичной областью использования — схемы кэш-памяти, буферы FIFO и LIFO, память данных небольшой емкости для микроконтроллеров, быстродействующих коммуникационных устройств и т. п.

В *ЗУ с последовательным доступом* записываемые данные образуют некоторую *очередь*. Считывание происходит из очереди слово за словом либо в порядке записи, либо в обратном порядке. Моделью такого ЗУ является последовательная цепочка запоминающих элементов, в которой данные передаются между соседними элементами.

Прямой порядок считывания имеет место в *буферах FIFO* с дисциплиной "первый пришел — первый вышел" (First In — First Out), а также в файловых и циклических ЗУ.

Разница между памятью FIFO и файловым ЗУ состоит в том, что в FIFO запись в пустой буфер сразу же становится доступной для чтения, т. е. слово поступает в конец цепочки (модели ЗУ). В файловых ЗУ данные поступают в начало цепочки и появляются на выходе после некоторого числа обращений, равного числу элементов в цепочке. При независимости операций считывания и записи фактическое расположение данных в ЗУ на момент считывания не связано с каким-либо внешним признаком. Поэтому записываемые данные объединяют в блоки, обрамляемые специальными символами конца и начала (файлы). Прием данных из файлового ЗУ начинается после обнаружения приемником символа начала блока.

В *циклических ЗУ* слова доступны одно за другим с постоянным периодом, определяемым емкостью памяти. К такому типу среди полупроводниковых ЗУ относится *видеопамять* (VRAM).

Считывание в обратном порядке свойственно *стековым ЗУ*, для которых реализуется дисциплина "последний пришел — первый вышел". Такие ЗУ называют *буферами LIFO* (Last In — First Out).

Время доступа к конкретной единице хранимой информации в последовательных ЗУ представляет собою случайную величину. В наихудшем случае для такого доступа может потребоваться просмотр всего объема хранимых данных.

Ассоциативный доступ реализует поиск информации по некоторому признаку, а не по ее расположению в памяти (адресу или месту в очереди). В наиболее полной версии все хранимые в памяти слова одновременно проверяются на соответствие признаку, например, на совпадение определенных полей слов (тегов — от англ. *tag*) с признаком, задаваемым входным словом (теговым адресом). На выход выдаются слова, удовлетворяющие признаку. Дисциплина выдачи слов, если тегу удовлетворяют несколько слов, а также дисциплина записи новых данных могут быть разными. Основная область применения ассоциативной памяти в современных ЭВМ — кэширование данных.

Технико-экономические параметры ЗУ существенно зависят от их схемотехнологической реализации. По этому признаку также возможна классификация ЗУ, однако удобнее рассматривать этот вопрос применительно к отдельным типам памяти.

Классификация перспективных ЗУ

Даже впечатляющий рост емкостей и быстродействия современных ЗУ не снимает вопроса о поиске еще более эффективных решений, т. к. сейчас многие средства обработки информации требуют огромных емкостей памяти чрезвычайно высокого быстродействия. *Перспективные варианты новых типов ЗУ* указаны на рис. 4.2, *б*.

Наиболее зрелыми, уже достигшими уровня промышленного производства, являются ЗУ *ферроэлектрического типа* (FRAM, Ferroelectric RAM), сочетающие высокие емкости и быстродействие с полезным свойством энергонезависимости. Такое сочетание свойств близко к идеалу, его не имеют ни статические, ни динамические ЗУ, ни EEPROM, ни FLASH-память.

MRAM — это *магниторезистивные ЗУ* (Magnetoresistive RAM). В схемах MRAM запоминающим элементом является участок магнитного материала, способный сохранять приданное ему состояние намагниченности независимо от наличия или отсутствия питания схемы. Иными словами, физические свойства используемых материалов придают MRAM естественную энергонезависимость. Элементы MRAM обеспечивают неразрушающее чтение информации. Степень зрелости MRAM ниже, чем у FRAM, но относительно велика — их производство начинается, но до широкого внедрения необходимо решить еще ряд проблем.

Разновидностью ферроэлектрических ЗУ являются *полимерные ферроэлектрические ЗУ* (PFRAM, Polymeric Ferroelectric RAM). В этих ЗУ используются полимерные ферроэлектрические материалы (тонкие пленки), в которых образуются диполи. Участки с ориентированными диполями служат запоминающими элементами и в зависимости от направления поляризации хранят биты информации. Вследствие простоты запоминающего элемента и компактности конструкции в целом PFRAM имеют чрезвычайно высокие емкости при очень малой стоимости/бит. В то же время быстродействие PFRAM мало. Это исключает их применение в качестве ОЗУ, но для использования *вместо дисковой памяти* PFRAM весьма перспективны.

В памяти типа OUM (Ovonics Unified Memory) сочетаются интегральная технология и запоминающие элементы, свойственные компакт-дискам (CD, DVD), т. е. устройствам с подвижным носителем информации. В памяти типа OUM запоминающим элементом служит перемычка из халькогенидного сплава (GeSbTe), который может находиться в проводящем кристаллическом или непроводящем аморфном состоянии. Память считается весьма перспективной с точки зрения экономических показателей.

Преобладающие виды современной памяти

Для микросхем памяти, реализованных по преобладающей в настоящее время КМОП-технологии, доля различных типов ЗУ в 2003 г. составляет:

- DRAM ~ 58%;
- SRAM ~ 21%;
- FLASH ~ 13%;
- EEPROM ~ 5%.

Перечисленные типы ЗУ вместе занимают около 97% всего объема продаж на мировом рынке, далее в этой главе основное внимание уделяется именно этим типам ЗУ.

§ 4.2. Основные структуры запоминающих устройств

Многочисленные варианты ЗУ имеют много общего с точки зрения структурных схем, что делает рациональным изучение некоторых обобщенных структур с последующим описанием особенностей и запоминающих элементов для конкретных ЗУ.

Общность структур адресных ЗУ особенно проявляется для статических ОЗУ и памяти типа ROM. *Структуры динамических ОЗУ имеют свою специфику и дополнительно рассмотрены в гл. 4.* Для статических ОЗУ и памяти типа ROM характерны структуры 2D, 3D, 2DM и блочные структуры на их основе.

Структура 2D

В структуре 2D (рис. 4.3) запоминающие элементы ЗЭ организованы в прямоугольную матрицу размерностью $M = k \times m$, где M — информационная емкость памяти в битах; k — число хранимых слов; m — их разрядность.

Дешифратор адресного кода DC при наличии разрешающего сигнала CS (Chip Select) активизирует одну из выходных линий, разрешая одновременный доступ ко всем элементам выбранной строки, хранящей слово, адрес которого соответствует номеру строки. Элементы столбца соединены вертикальной линией — внутренней линией данных (разрядной линией, линией записи/считывания). Элементы столбца хранят одноименные биты всех слов. Направление обмена определяется усилителями чтения/записи под воздействием сигнала R/W (Read — чтение, Write — запись).

Структура типа 2D применяется лишь в ЗУ малой информационной емкости, т. к. при росте емкости проявляется несколько ее недостатков, наибо-

лее очевидным из которых является чрезмерное усложнение дешифратора адреса (число выходов дешифратора равно числу хранимых слов).

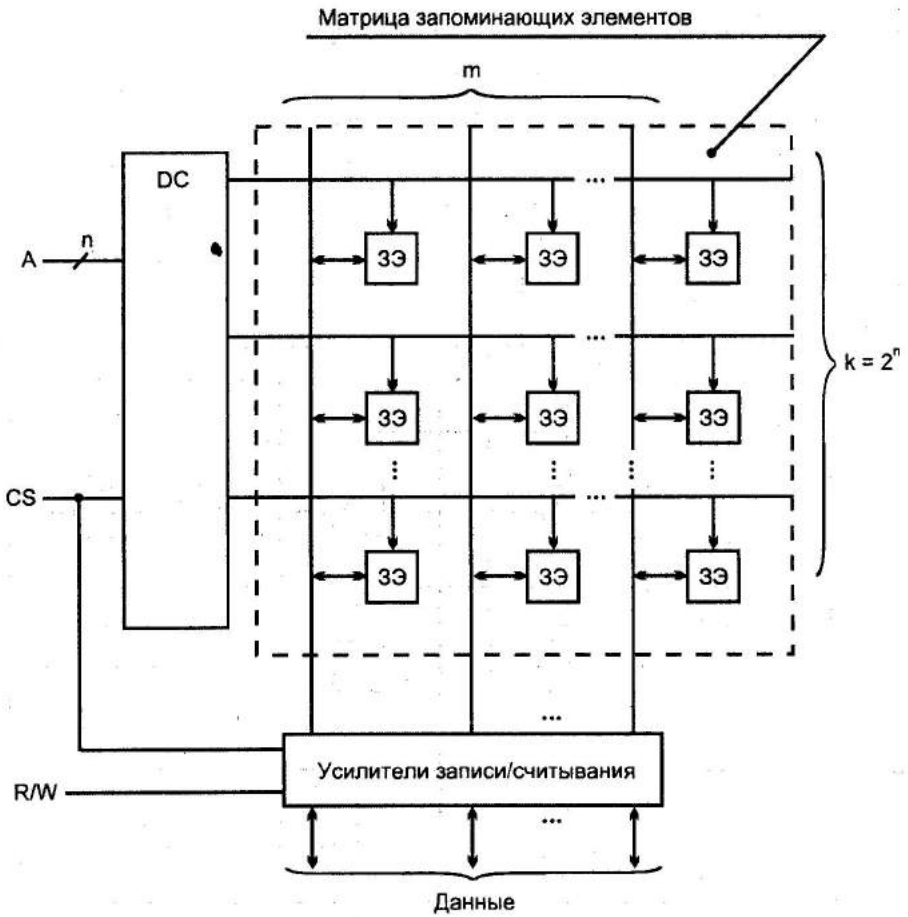


Рис. 4.3. Структура ЗУ типа 2D

Структура 3D

Структура 3D позволяет резко упростить дешифраторы адреса с помощью двухкоординатной выборки запоминающих элементов. Принцип двухкоординатной выборки поясняется на примере ЗУ типа ROM (рис. 4.4, *а*), реализующего только операции чтения данных.

Здесь код адреса разрядностью n делится на две половины, каждая из которых декодируется отдельно. Выбирается запоминающий элемент, находя-

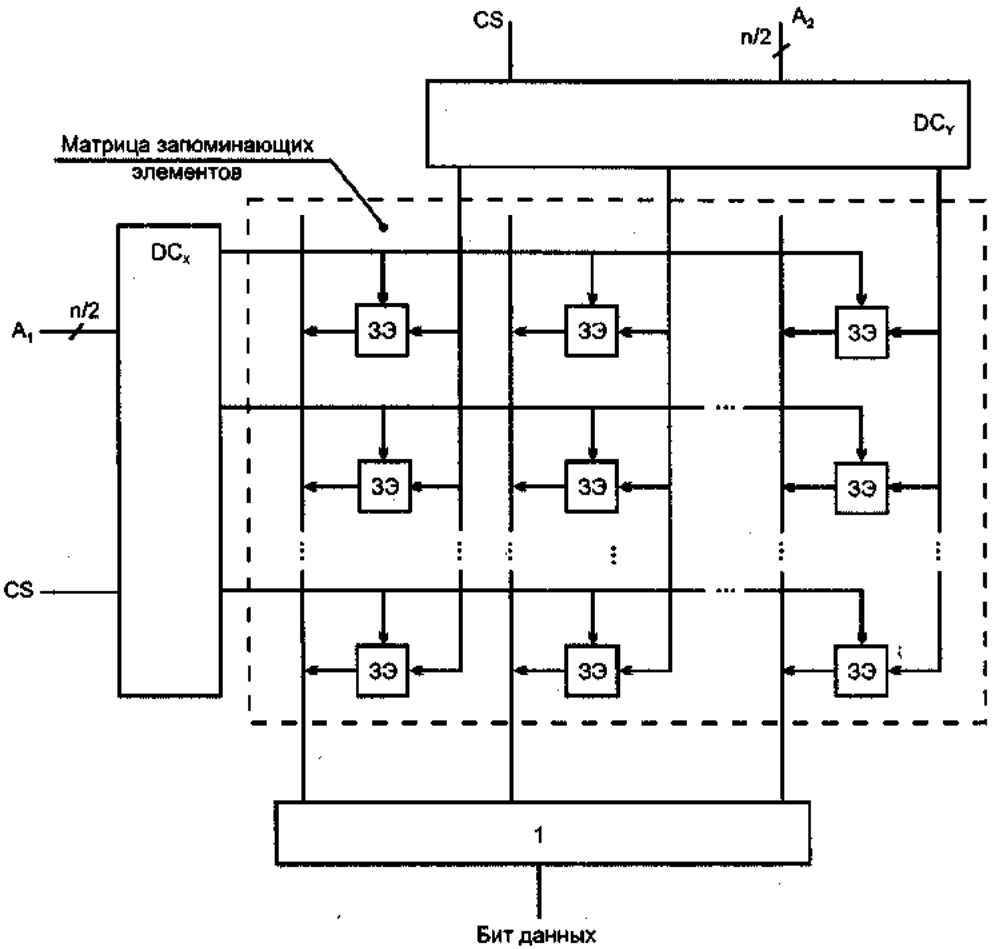
щийся на пересечении активных линий выходов обоих дешифраторов. Таких пересечений будет как раз

$$2^{n/2} \times 2^{n/2} = 2^n.$$

Суммарное число выходов обоих дешифраторов составляет

$$2^{n/2} + 2^{n/2} = 2^{n/2+1},$$

что гораздо меньше, чем 2^n при реальных значениях n.



а

Рис. 4.4. Структура ЗУ типа 3D с одноразрядной (*а*) организацией

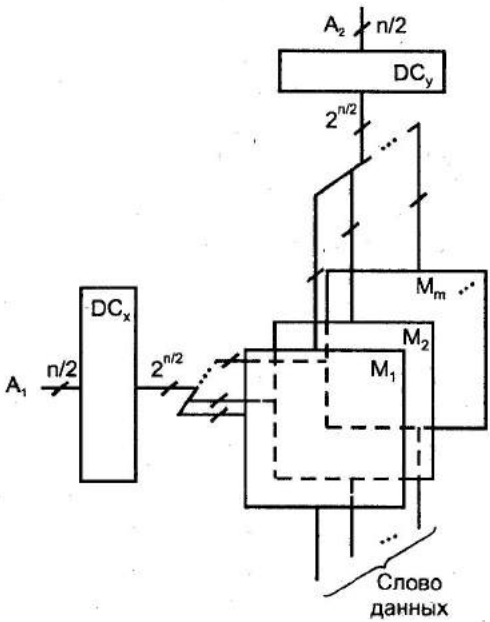


Рис. 4.4. Структура ЗУ типа 3D
с многоразрядной (*б*) организацией

Уже для ЗУ небольшой емкости видна эта существенная разница: для структуры 2D при хранении 1K слов потребовался бы дешифратор с 1024 выходами, тогда как для структуры типа 3D нужны два дешифратора с 32 выходами каждый. Недостатком структуры 3D в первую очередь является усложнение элементов памяти, имеющих двухкоординатную выборку.

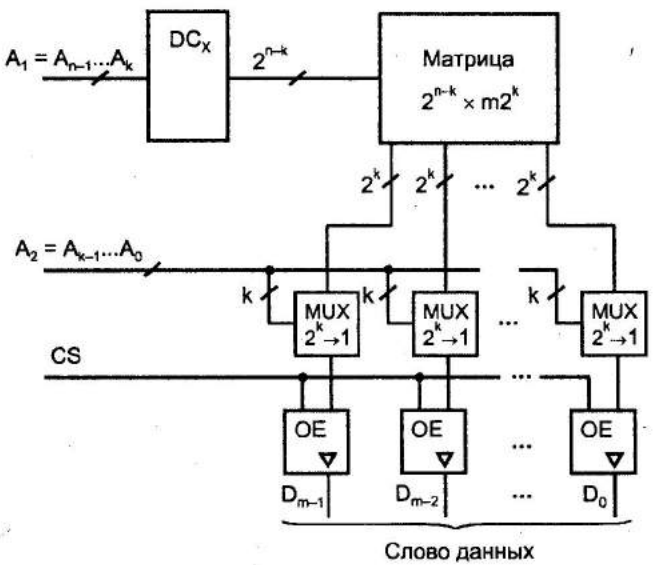
Структура типа 3D, показанная на рис. 4.4, *а* для ЗУ с одноразрядной организацией, может применяться и в ЗУ с многоразрядной организацией (рис. 4.4, *б*), приобретая при этом "трехмерный" характер. В этом случае несколько матриц управляются от двух дешифраторов, относительно которых они включены параллельно. Каждая матрица выдает один бит адресованного слова, а число матриц равно разрядности хранимых слов.

Структуры типа 3D имеют также довольно ограниченное применение, поскольку в структурах типа 2DM сочетаются достоинства обеих рассмотренных структур — упрощается дешифрация адреса и не требуются запоминающие элементы с двухкоординатной выборкой.

Структура 2DM

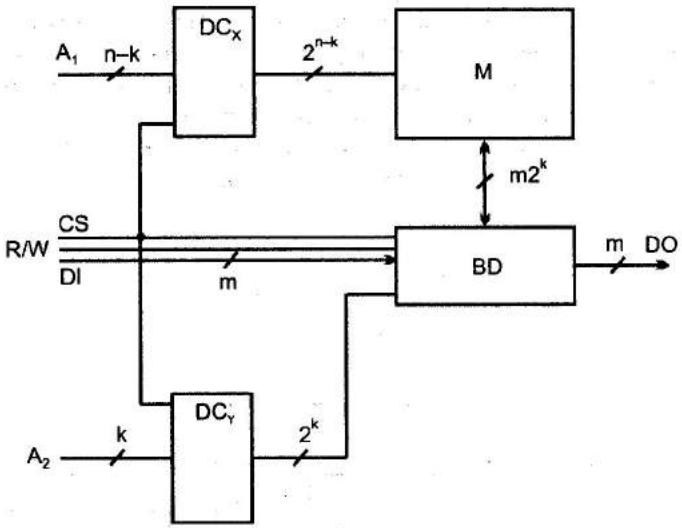
ЗУ структуры 2DM (2D модифицированная) (рис. 4.5, *а*) для матрицы запоминающих элементов с адресацией от дешифратора DCx имеет как бы характер структуры 2D: возбужденный выход дешифратора выбирает целую строку. Однако в отличие от структуры 2D, длина строки не равна разрядности хранимых слов, а многократно ее превышает. При этом число строк матрицы уменьшается и, соответственно, уменьшается число выходов дешифратора. Для выбора одной из строк служат не все разряды адресного кода, а их часть $A_{n-1}... A_k$. Остальные разряды адреса (от A_{k-1} до A_0) используются, чтобы выбрать необходимое слово из того множества слов, которое содержится в строке. Это выполняется с помощью мультиплексоров, на адресные входы которых подаются коды $A_{k-1}... A_0$. Длина строки равна $m2^k$, где m — разрядность хранимых слов. Из каждого "отрезка" строки длиной 2^k мультиплексор выбирает один бит. На выходах мультиплексоров формируется выходное слово. По разрешению сигнала CS, поступающего на входы OE управляемых буферов с тремя состояниями, выходное слово передается на внешнюю шину.

На рис. 4.5, *а* для большей наглядности структура 2DM показана на примере ROM. На рис 4.5, *б* структура 2DM в более общем виде показана для ЗУ типа RAM с операциями чтения и записи. Из матрицы M по-прежнему считывается "длинная" строка.



а

Рис. 4.5. Структура ЗУ типа 2DM для ROM (*а*)



б

Рис. 4.5. Структура ЗУ типа 2DM для RAM (*б*)

Данные в нужный отрезок этой строки записываются (или считываются из нее) управляемыми буферами данных BD, воспринимающими выходные сигналы второго дешифратора DC_Y, и выполняющими не только функции мультиплексирования, но и функции изменения направления передачи данных под воздействием сигнала R/W.

Структура блочных ЗУ

На примере предыдущих структур можно видеть, что разделение адреса на части, используемые различным образом, позволяет получить выигрыш в значениях параметров ЗУ. Так, в структуре 2D, где адрес выступает как единое слово, число строк матрицы равно числу хранимых слов, а число ее столбцов совпадает с разрядностью слов. При этом матрица имеет прямоугольную форму, далекую от оптимальной квадратной — ее высота, как правило, многократно превышает ширину. В структуре 2DM адрес разделяется на две части, это позволяет уменьшить число строк в матрице запоминающих элементов и увеличить число столбцов, приближая форму матрицы к квадратной. В квадратной матрице сокращается длина строк и столбцов, и, тем самым, уменьшается нагрузка на линии выборки и записи/считывания. Уменьшение нагруженности линий позволяет поддерживать необходимое быстродействие ЗУ.

До определенных пределов информационных емкостей архитектуры ЗУ с одной матрицей вполне приемлемы. Переход к более емким ЗУ ведет к серьезному снижению их быстродействия, поскольку удлинение строк и столбцов матрицы увеличивает электрическую емкость и сопротивление словарных и разрядных линий. Для ЗУ максимальных информационных емкостей используют *структуры с несколькими матрицами запоминающих элементов, которые назовем блочными* (рис. 4.6).

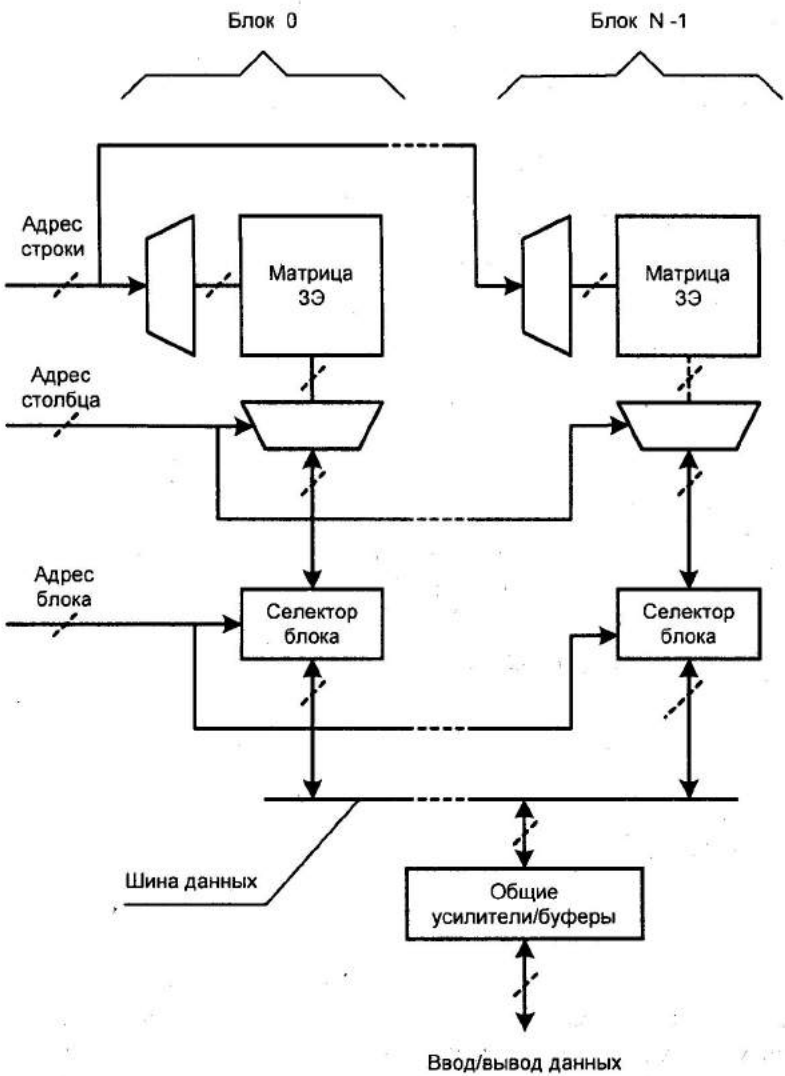


Рис. 4.6. Структура блочного ЗУ

В блочном ЗУ память разделяется на N идентичных блоков. На все блоки подаются одни и те же адреса строк и столбцов. В составе адреса помимо частей, адресующих строки и столбцы, выделяется и третье поле — адрес блока, по которому выбирается блок, которому разрешается выход на шину данных для чтения или записи. Такая архитектура дает преимущества как в быстродействии (в связи с сокращением длин словарных и разрядных линий в пределах блока), так и в возможности экономии мощности, поскольку можно затрачивать необходимую для активного режима мощность только для адресованного блока. В то же время схемная реализация ЗУ усложняется (в частности, вследствие увеличения числа декодеров в схемах адресации).

Память с последовательным доступом

Память с последовательным доступом строится либо с использованием продвижения данных в цепочке элементов (по подобию с регистрами сдвига), либо с хранением данных в адресном ЗУ при необходимом управлении адресом доступа.

Основными представителями этого вида памяти являются видеопамять, буфер FIFO и стек (буфер LIFO).

Видеопамять

Видеопамять работает циклично, на ее выходе последовательно в порядке сканирования экрана монитора лучом появляются коды, задающие параметры светимости (цвет, яркость) элементарных точек экрана — *пикселов*. Текущее изображение на мониторе — кадр — представлено последовательностью слов, длина которой равна числу пикселов экрана (кроме слов, воздействующих на пикселы, видеопамять генерирует и синхросимволы строчной и кадровой синхронизации, встроенные в выходную последовательность). Слово, соответствующее одному пикселу, может иметь различную разрядность в зависимости от требуемого числа возможных состояний пиксела. Если достаточно иметь всего два состояния пиксела — светится или не светится — то для воздействия на него нужны всего лишь однобитные слова. Если же речь идет о высококачественном воспроизведении цветных изображений, то используются 24-разрядные слова, что соответствует более чем 16 миллионам состояний пиксела.

При реализации на основе адресной памяти циклический доступ к данным обеспечивается счетчиком адреса с модулем, равным числу запоминаемых слов. При считывании после каждого обращения адрес увеличивается на единицу, обеспечивая последовательное обращение ко всем ячейкам ЗУ. При переполнении счетчика формируется сигнал начала кадра для управления монитором (для запуска кадровой синхронизации). Запись возможна в пакетном режиме или режиме одиночных записей. В первом случае сигнал переполнения счетчика и его переход на начальный адрес являются сигна-

лом начала передачи блока данных из основной памяти или видеобуфера. Во втором случае адрес изменяемой ячейки (номер пиксела) и данные сохраняются в буфере, а в момент совпадения этого адреса и содержимого счетчика выполняется один цикл записи нового слова. Все остальное время ЗУ работает обычным образом.

Построение циклических ЗУ с продвижением информации (рис. 4.7) для наглядности показано с представлением элементов хранения и перезаписи данных в виде статических регистров, хотя схемотехнически регистры кратковременного хранения реализуются более экономично. Для этого может быть использована динамическая память, имеющая большую емкость и невысокую стоимость. Ее применение в схемах видеопамяти облегчается тем, что постоянная перезапись содержимого элементов памяти одновременно выполняет и функции регенерации данных, необходимые для этого вида ЗУ.

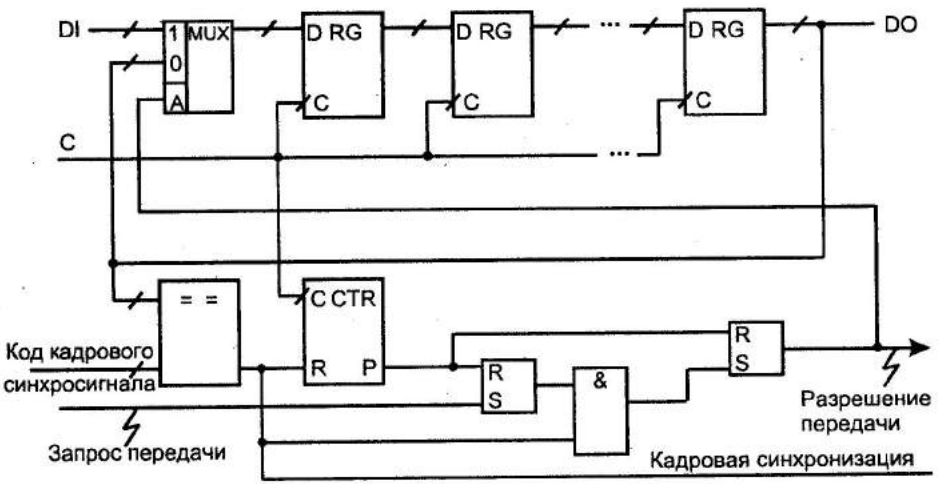


Рис. 4.7. Структура видеопамяти

При циклическом считывании хранимых данных выбран нижний канал мультиплексора MUX и записанные данные постоянно переписываются с выхода на вход цепочки запоминающих элементов. В последовательность данных вводятся специальные коды синхросигналов (кадровых и строчных, но на рис. 4.7 для пояснения принципа показан только кадровый). Появление кода синхросигнала на выходе обнаруживается компаратором и синхронизирует запуск развертки монитора.

Пакетная запись может начинаться после появления запроса передачи в момент прохождения кода кадрового синхросигнала. При этом вырабатывается сигнал разрешения передачи кадра из памяти ЭВМ на вход DI, а муль-

типлексор MUX переключается на верхний канал. После приема целого кадра счетчик CTR, емкость которого равна длине кадра, переполняется, и под воздействием сигнала переполнения ЗУ возвращается в режим циклической перезаписи.

При одиночных записях устройство должно иметь дополнительно схему сравнения кода счетчика и входного адресного кода (номера заменяемого кода пиксела). При их совпадении мультиплексор MUX переключается на верхний канал на один такт работы, чем обеспечивается замена всего одного слова.

Буфер FIFO

Буфер FIFO, пример структуры которого приведен на рис. 4.8, представляет собою ЗУ для хранения очередей данных (списков) с порядком выборки слов, таким же, что и порядок их поступления. Интервалы между словами могут быть совершенно различными, т. к. моменты записи слова в буфер и считывания из него задаются внешними сигналами управления независимо друг от друга.

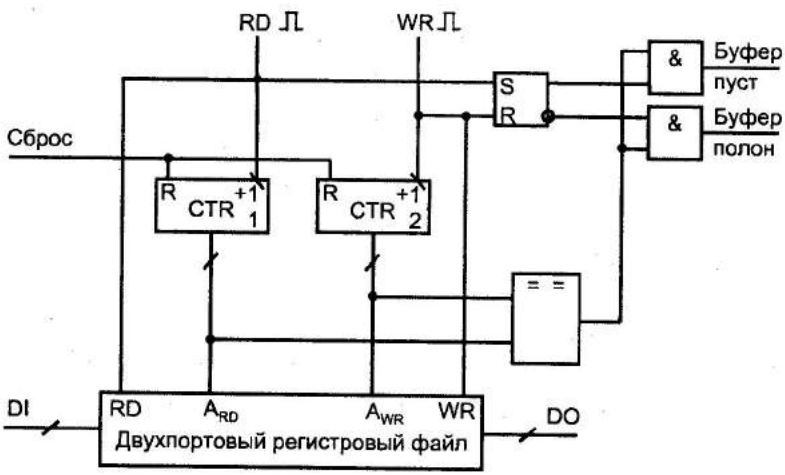


Рис. 4.8. Структура буфера FIFO

Возможность иметь разный темп приема и выдачи слов необходима, например, если приемник способен принимать данные, поступающие регулярно с некоторой частотой, а источник информации выдает слова в более быстром темпе и, может быть, к тому же не регулярно. Такие данные поступают в их темпе в буфер FIFO, а из него считываются регулярно с необходимой для приемника данных частотой. Новое слово ставится в конец очереди, считывание осуществляется с начала очереди.

В схеме (см. рис. 4.8) перед началом работы оба счетчика адресов CTR_1 и CTR_2 сбрасываются. При записи адреса увеличиваются на единицу при каждом обращении, т. е. возрастают, начиная с нулевого. То же происходит при чтении слов, так что адрес чтения всегда "гонится" за адресом записи. Если адреса сравняются при чтении, то буфер пуст. Если адреса сравняются при записи, то буфер полон (адресами занята вся емкость счетчика). Эти ситуации отмечаются соответствующими сигналами. Если буфер полон, то нужно прекратить прием данных, а если пуст, то нужно прекратить чтение. Очередь удлиняется или укорачивается в зависимости от разности чисел записанных и считанных слов. Переход через нуль осложнений не вызывает.

Задачу построения стека можно решить принципиально аналогичным способом. Эта задача встречается в дальнейшем изложении при рассмотрении структуры микропроцессора.

Кэш-память

Кэш-память запоминает копии информации, передаваемой между устройствами (прежде всего между процессором и основной памятью). Она имеет небольшую емкость в сравнении с основной памятью и более высокое быстродействие (реализуется на триггерных элементах памяти).

При чтении данных сначала выполняется обращение к кэш-памяти (рис. 4.9). Если в кэше имеется копия данных адресованной ячейки основной памяти, то кэш вырабатывает сигнал *Hit* (попадание) и выдает данные на общую шину данных. В противном случае сигнал *Hit* не вырабатывается и выполняется чтение из основной памяти и одновременное помещение считанных данных в кэш.

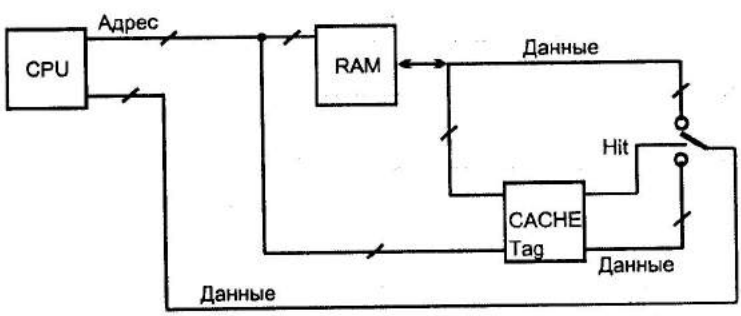


Рис. 4.9. Структура кэшированной памяти

Эффективность кэширования обуславливается тем, что большинство прикладных программ удовлетворяют принципу *локальности* или, иначе говоря, имеют *гнездовой характер обращений*, при котором адреса последовательных

обращений к памяти образуют, как правило, компактную группу. Поэтому после первого обращения к относительно медленной основной памяти повторные обращения (уже к кэшу) требуют меньше времени. К тому же при использовании процессором кэш-памяти основная память освобождается, и могут выполняться регенерация данных в динамическом ЗУ или использование памяти другими устройствами.

Объем кэш-памяти много меньше емкости основной памяти и любая единица информации, помещаемая в кэш, должна сопровождаться дополнительными данными (тегом), определяющими, копией содержания какой ячейки основной памяти она является.

В полностью ассоциативной кэш-памяти (FACM, Fully Associated Cache Memory), структура которой показана на рис. 4.10, каждая ячейка хранит данные, а в поле "Тег" — полный физический адрес информации, копия которой записана. При любых обменах физический адрес запрашиваемой информации сравнивается с полями "Тег" всех ячеек и при совпадении их в любой ячейке устанавливается сигнал *Hit*.

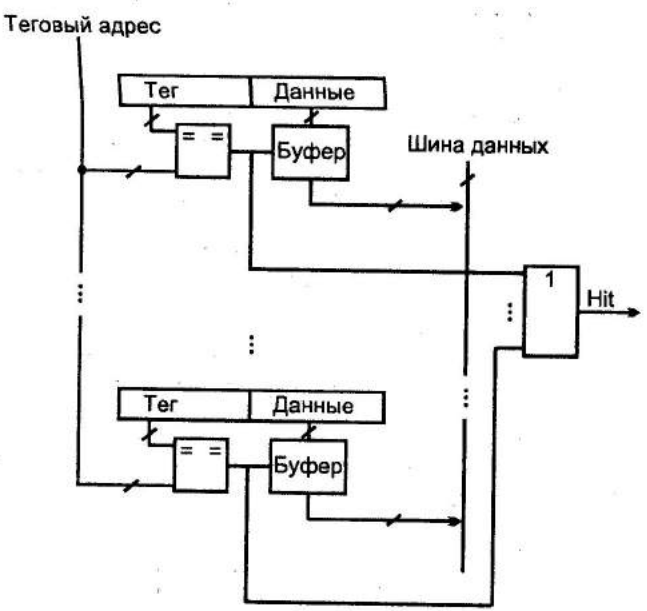


Рис. 4.10. Структура полностью ассоциативной кэш-памяти

При чтении и значении сигнала *Hit* = 1 данные выдаются на шину данных, если же совпадений нет (*Hit* = 0), то при чтении из основной памяти данные вместе с адресом помещаются в свободную или наиболее давно не используемую ячейку кэш-памяти.

При записи данные вместе с адресом сначала, как правило, размещаются в кэш-памяти (в обнаруженную ячейку при *Hit* = 1 и свободную при *Hit* = 0). Копирование данных в основную память выполняется под управлением специального контроллера, когда нет обращений к памяти.

Память типа FACM является весьма сложным устройством и используется только при малых емкостях, главным образом в специальных приложениях. В то же время этот вид кэш-памяти обеспечивает наибольшую функциональную гибкость и бесконфликтность адресов, т. к. любую единицу информации можно загрузить в любую ячейку кэш-памяти.

Сложность FACM заставляет искать иные структуры кэш-памяти, более экономичные по затратам аппаратных средств на их реализацию. К числу таких структур относятся кэш-память с прямым размещением и кэш-память с наборно-ассоциативной архитектурой (с ассоциацией по нескольким направлениям). До конкретного рассмотрения этих структур укажем, что главными параметрами кэш-памяти являются *размер строки* (Cache Line) и *число строк* (рис. 4.11). Строка представляет собою некоторый набор слов. Ее емкость будем считать соответствующей странице основной памяти.

Примечание

В большинстве работ по кэш-памяти, в частности, в работе [27] основными понятиями, отображающими ее функционирование, служат тег, индекс и блок. Принятая здесь терминология, хотя и является одним из используемых вариантов, отличается от указанной применением термина "страница". Хотя обычно этот термин связан с механизмами работы виртуальной памяти, здесь он сохранен применительно к работе кэш-памяти, поскольку в подобных ситуациях применялся и в предыдущих параграфах.

Рис. 4.11. Представление кэш-памяти в виде совокупности строк

В структуре FACM, называемой также структурой с произвольной загрузкой, любую страницу можно загрузить в любую строку кэш-буфера (рис. 4.12, *а*).

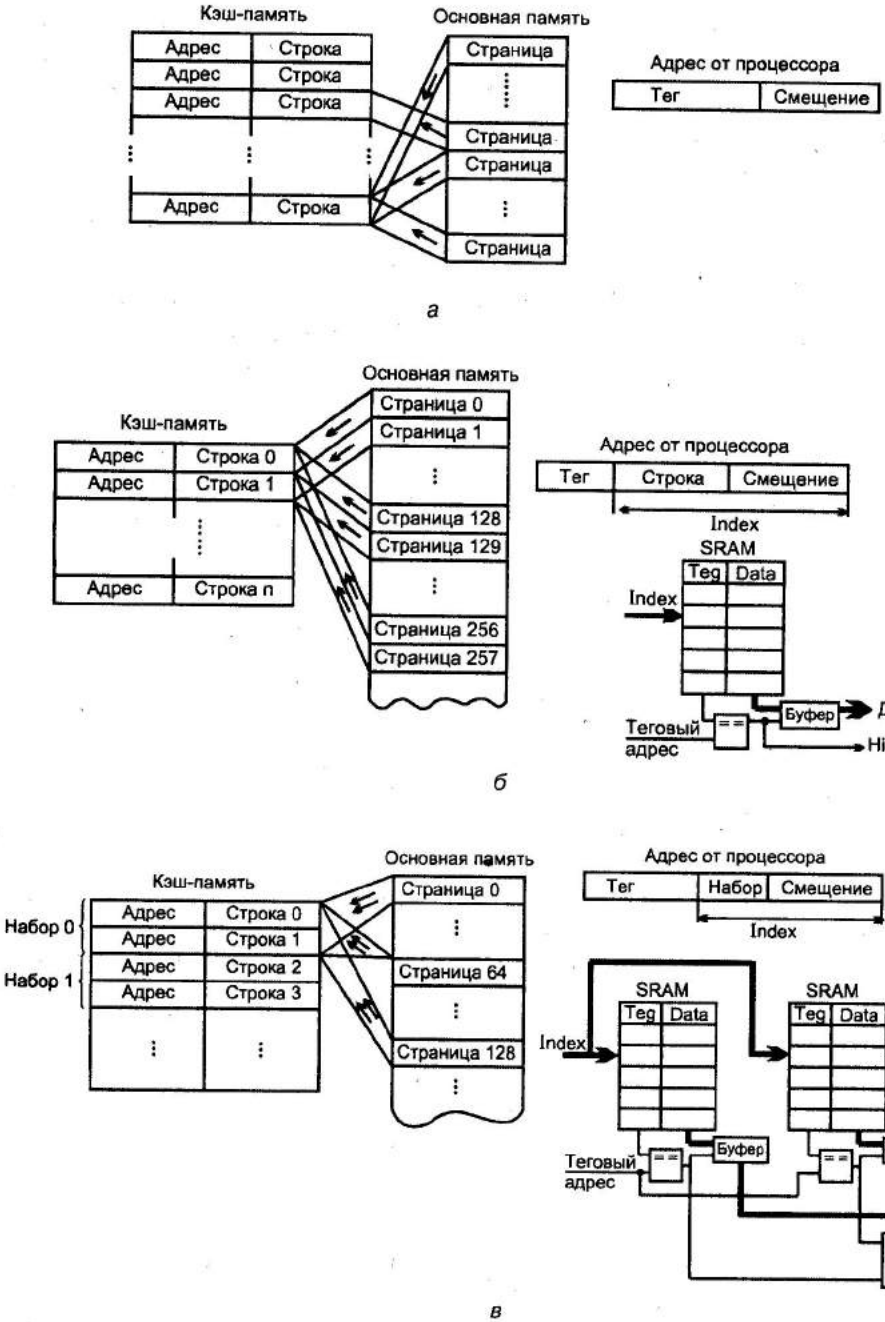


Рис. 4.12. Пояснения к организации кэш-памяти с произвольной загрузкой (*а*), с прямым размещением (*б*) и наборно-ассоциативной (*в*)

В качестве тега используется полный физический адрес, если речь идет об адресации отдельных слов, или старшие разряды этого адреса за вычетом младших (смещения), если смещение адресует слово в пределах строки.

Иными словами, в этом случае старшие разряды адреса рассматриваются как тег, тогда как младшие используются для адресации в пределах строки.

В кэш-памяти с прямым размещением (с прямым отображением) несколько страниц основной памяти строго соответствуют одной строке кэша (рис. 4.12, *б*). Так как занимать строку в одно и то же время может только одна страница, нужен специальный ее признак — тег. Адрес от процессора делится на три части. Младшие разряды (смещение) определяют положение слова в строке. Средние позволяют выбрать одну из строк кэш-памяти. Оставшиеся старшие образуют тег. По адресу строки производится считывание. Поле адресов считанной строки сравнивается с теговым адресом и, если есть совпадение, вырабатывается сигнал *Hit* выдачи информации и затем мультиплексированием из строки данных выбирается слово. При загрузке из внешней памяти заменяется вся строка. Здесь следует отметить, что блочные передачи в современных системах осуществляются достаточно быстро.

Тег для кэш-памяти с прямым размещением сильно сокращается по разрядности. Обычно номер строки есть адрес страницы по модулю, равному целой степени двойки. На рис. 4.12, *б* это 128. Достоинство кэша с прямым размещением — экономичность по аппаратурным затратам. Недостаток — ограничения на расположение страниц в кэше, что может не позволить сформировать в нем оптимальный набор страниц, т. к. передача в кэш страницы вызывает удаление из него другой, которая может быть нужна для формирования оптимального набора страниц.

Промежуточным по сложности и эффективности вариантом между структурами FACM и с прямым размещением является *кэш-память с ассоциацией по нескольким направлениям* (наборно-ассоциативная). В этом варианте несколько строк кэша объединяются в наборы, а средние разряды адреса памяти определяют уже не одну строку, а *набор* (рис. 4.12, *в*). Кэш-память делится на наборы с небольшим числом строк, кратным двойке, т. е. 2, 4, 8, ... и т. д. (на рисунке это 2). Страницу основной памяти можно поместить только в тот набор, номер которого равен адресу страницы по модулю (в данном случае модуль равен 64). Место страницы в наборе может быть произвольным. Сравнение тегов со старшими разрядами адреса производится только для строк, входящих в набор.

По числу строк в наборе кэш-памяти различают разнообразные структуры: двухвходовые, четырехвходовые и т. д.

Для взятого примера используются два отдельных блока памяти для четных строк и нечетных строк. Одновременно выбираются четные и нечетные строки (слова в них). Считывание идет от того блока, где имеется совпадение тега и тегового адреса. При этом из строки через смещение выбирается

адресованное слово. При отсутствии совпадений происходит обращение к основной памяти и замещение строки в одном из блоков кэша.

Блок-схема наборно-ассоциативного кэша показана на рис. 4.12, *в*. По сравнению с кэшем с прямым размещением, кэш наборно-ассоциативного типа имеет несколько удлиненный тег (во взятом примере всего на один разряд). Возможность свободного размещения страниц в наборе позволяет сформировать в кэше лучший состав страниц, т. к. имеется возможность выбрать ту или иную заменяемую страницу. В современных микропроцессорных системах кэш первого уровня, обозначаемый L1 (от английского слова Level (внутрипроцессорный)), обычно имеет наборно-ассоциативную структуру, а кэш второго уровня L2 (внешний) — структуру с прямым размещением.

Ряд фирм выпускают микросхемы ассоциативной памяти. Для построения кэш-памяти используют чаще всего обычные SRAM с ассоциативным доступом в сочетании с кэш-контроллерами.

В высокопроизводительном микропроцессоре Power 3 фирмы IBM использован кэш наборно-ассоциативного типа емкостью 32 Кбайта для команд и 64 Кбайта для данных на 128 направлений. Для связей с кэшем второго уровня L2 в системе Power 3 применена 256-разрядная шина. Емкость кэша L2 от 1 до 16 Мбайт.

§ 4.3. Структурные методы повышения быстродействия запоминающих устройств

Постоянно растущие потребности вызвали к жизни ряд методов повышения быстродействия микросхем памяти. При этом наряду с технологическим совершенствованием элементов, уменьшающим их задержки и позволяющим увеличивать рабочие частоты микросхем, идет и процесс модификации структур ЗУ, также существенно влияющий на их быстродействие. Многие *структурные методы*, разработанные вначале для конкретных типов ЗУ, стали распространяться и на другие их разновидности, приобретая достаточно универсальный характер. Такие методы полезно рассмотреть в обобщенном виде.

Кучность адресов и произвольный доступ

Эффективность и целесообразность применения различных структурных методов повышения быстродействия ЗУ зависят от характера изменения адресов при обращениях к памяти. При реализации компьютерных программ чаще всего наблюдается ситуация, которую можно назвать *кучностью адресов*. В этом случае после обращения по некоторому адресу следующее обращение вероятнее всего будет по соседнему адресу или близкому к нему. Так проявляется известное свойство *локальности (гнездования) программ*. После-

довательное изменение адресов при реализации программ нарушается командами переходов, но их доля среди команд программы невелика.

Для других алгоритмов, например, связанных с сетевыми задачами и процессами телекоммуникаций, кучность адресов не наблюдается. Адрес последующего обращения к памяти не коррелируется с адресом предыдущего. В этом случае говорят о работе памяти в режиме *произвольного доступа*.

Быстрый страничный доступ

Этот способ повышения быстродействия применяется при работе ЗУ в условиях кучности адресов. При произвольном доступе к памяти такой способ совершенно бесполезен. Дело в том, что в условиях кучности адресов при обращении по следующему адресу изменяется лишь часть разрядов адресного кода. Если обращение идет к близко расположенной ячейке, то старшие разряды адреса не изменятся. Адрес можно рассматривать как состоящий из двух частей — адреса страницы (старшие разряды) и адреса слова на странице (младшие разряды). Если страницей считать строку квадратной матрицы запоминающих ячеек, то адресный код будет разделен на две равные части — адрес страницы (номер строки) и адрес слова в пределах страницы (номер столбца).

Первоначальное обращение к памяти требует обработки обеих частей адреса, что занимает определенное время. Если следующее обращение происходит по адресу на той же странице, то процесс обращения к ячейке упрощается, поскольку обновляется только часть адреса, и цикл выборки ячейки сокращается. Действительно, неизменность части адресного кода означает, что часть схемы ЗУ останется в прежнем состоянии и не потребуется время на переключение ее элементов.

Быстрый страничный доступ в свое время появился в динамических ОЗУ и позволил повысить их быстродействие приблизительно на 40%.

Пакетная передача данных и команд

Этот способ близок к страничному доступу и также эффективен только при кучности адресов обращения к ЗУ. Пакетная передача адресов и команд означает, что при обращении к памяти последовательно извлекается не только первоначально адресованное слово, но и несколько соседних — пакет. Размер пакета может быть различным, часто он составляет два, четыре или восемь слов. Внутри пакета выборка слов идет быстро, т. к. адреса формируются внутри самой схемы (инкрементированием исходного адреса), и не требуется для каждого слова пакета передавать в микросхему его адрес или часть адреса (адрес на странице), что по сравнению со страничным доступом сокращает время выборки слов.

Работа ЗУ с быстрым страничным или пакетным доступом характеризуется цепочкой цифр, первая из которых характеризует время первоначального доступа к памяти, а последующие — времена доступа к следующим словам в пределах страницы или пакета. Например, такая цепочка может иметь вид 5-2-2-2 или 5-1-1-1 для пакетного доступа с размером пакета 4. Это означает, что первоначальный доступ занимает 5 некоторых единиц времени, а последующие 2 или 1.

В англоязычной литературе подобные цепочки называют *таймингами* (Timing). Этот же термин применяется и в русской терминологии, хотя следует иметь в виду, что он связан не только с данной ситуацией и в общем случае имеет гораздо более широкое толкование.

Передача данных с удвоенной скоростью (технология DDR — Double Data Rate). Технология QDR (Quad Data Rate)

Этот способ связан с повышением пропускной способности *интерфейса* памяти, т. е. тракта ввода/вывода данных. Такие тракты, реализованные на печатной плате или кристалле, имеют ограниченную техническими возможностями частоту тактирования передач. Традиционно данные воспринимались в моменты перепадов тактирующего сигнала в каком-либо одном направлении — по положительным или отрицательным фронтам синхросигнала (технология SDR, Single Data Rate). Технология DDR (Double Data Rate) предусматривает *восприятие данных по обоим фронтам синхросигнала*, а это удваивает скорость передачи. Заметим, что схемотехника внутри ЗУ может оставаться обычной (с синхронизацией передач фронтами одного знака) при согласовании пропускной способности интерфейса и ЗУ за счет разной разрядности трактов передачи данных. Заметим также, что технология DDR стала применяться не только в ЗУ, но и в интерфейсах других устройств.

Разработаны также ЗУ типа QDR (Quad Data Rate), в которых интерфейс DDR сочетается с двухпортовостью. В таких схемах за один такт выполняются четыре операции, поскольку одновременно производятся запись по одному адресу и чтение по другому.

Применение многобанковых структур

Применение многобанковых структур (Multibank Memory, Interleaving Memory) эффективно при кучности адресов обращения к памяти и для таких ЗУ, которые после выполнения операции чтения нуждаются в восстановлении исходного режима, т. е. длительность цикла которых включает в себя время восстановления начального состояния после обращении к ЗУ. Многобанковость позволяет исключать время восстановления из цикла об-

ращения к ЗУ. В многобанковых структурах память делится на части (банки), число которых может быть различным. Эффект появляется уже при делении памяти на два банка, с увеличением их числа эффективность метода возрастает, хотя и не линейно.

В двухбанковой структуре в одном банке размещаются ячейки с нечетными адресами, в другом — с четными. Если обращение идет по последовательным адресам, то *банки работают поочередно*, и каждый из них имеет свободный интервал для восстановления необходимого исходного состояния. Поэтому после завершения выборки данных из одного банка можно сразу же приступать к выборке из другого. Если следующий адрес не является соседним по отношению к предыдущему, то возможно повторное обращение к тому же самому банку, а это требует выполнения обычного (не сокращенного) цикла, поскольку придется тратить время на восстановление в активном банке исходного состояния. Чем больше банков в структуре памяти, тем реже будут возникать ситуации с соседними обращениями в тот же банк, т. е. тем выше будет выигрыш в быстродействии памяти. В некоторых ЗУ используется до 32 банков.

Необходимость в длительном интервале восстановления исходного состояния наиболее характерна для микросхем динамической памяти.

Конвейеризация трактов передачи данных

Сущность конвейеризации, которая широко применяется для повышения производительности различных средств обработки информации, заключается в разбиении трактов обработки информации на ступени. На рис. 4.13, *а* показан тракт обработки данных, содержащий входной и выходной регистры и логическую схему ЛСх между ними. Исходя из тезиса о возможности подачи новых входных данных только после окончания обработки старых, получим минимальный период тактовых импульсов для этой схемы:

$$T_{min} = t_{RG} + t_{кц} + t_{SU},$$

где t_{RG} — задержка входного регистра на пути "такт—выход"; $t_{кц}$ — задержка сигнала в комбинационной цепи (логической схеме); t_{SU} — время предустановки выходного регистра.

Уменьшения T_{min}, т. е. повышения частоты тактовых импульсов, можно добиться снижением $t_{кц}$ путем расщепления логической схемы на ступени, разделенные регистрами (рис. 4.13, *б*). Если логическая схема расщепляется по глубине ровно пополам, то новое значение минимального периода тактовых импульсов определится тем же соотношением, что и для схемы, показанной на рис. 4.13, *а*, однако численное значение задержки логической схемы уменьшится вдвое, т. е. допустимая частота тактирования схемы увеличится. Конвейеризация увеличивает темп передач данных по тракту их обработки, хотя время нахождения каждой единицы данных в этом тракте

не только не уменьшается, но даже увеличивается. Первая порция результатов в конвейеризованной системе появляется позднее, чем в сквозной, зато последующие поступают с более высокой частотой.

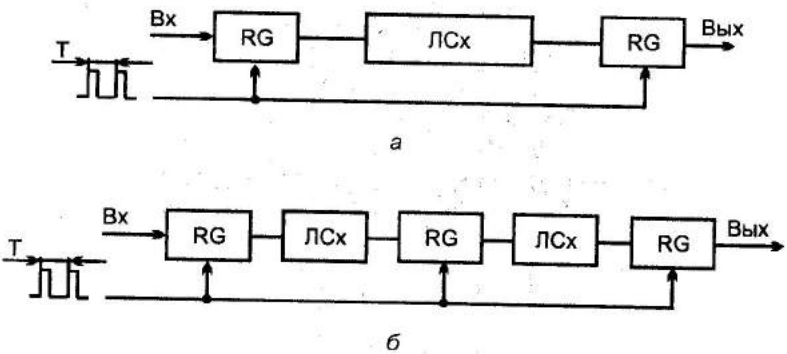


Рис. 4.13. Исходный (*а*) и конвейеризованный (*б*) тракты обработки информации

§ 4.4. Запоминающие устройства с рабочим режимом "только для чтения" (типа ROM(M), PROM, EPROM, EEPROM)

Запоминающие устройства типа "память только для чтения" хранят информацию, которая в зависимости от типа ЗУ либо вообще не изменяется, либо изменяется редко и в специальном режиме программирования. Программирование всех видов постоянной памяти заключается в том или ином размещении элементов связи между горизонтальными и вертикальными линиями матрицы запоминающих элементов.

Микросхемы памяти с рабочим режимом "только для чтения" имеют многоразрядную организацию (чаще всего 8-, 16- или 32-разрядную), их матрицы обычно выполняются по структуре 2DM. Простейшие ЗУ могут иметь структуру 2D. Технологии изготовления постоянных ЗУ разнообразны — диодные матрицы, ТТЛ(Ш), КМОП, n-МОП и др.

Масочные ЗУ

В масочные однократно программируемые ЗУ типа ROM(M) информация записывается на промышленных предприятиях с помощью шаблона (маски) на завершающем этапе технологического процесса.

Элементом связи (запоминающим элементом) в масочных ЗУ могут быть диоды, биполярные транзисторы, МОП-транзисторы и т. д.

В матрице диодного ROM(M) (рис. 4.14, *а*) горизонтальные линии являются линиями выборки слов, а вертикальные — линиями считывания разрядов.

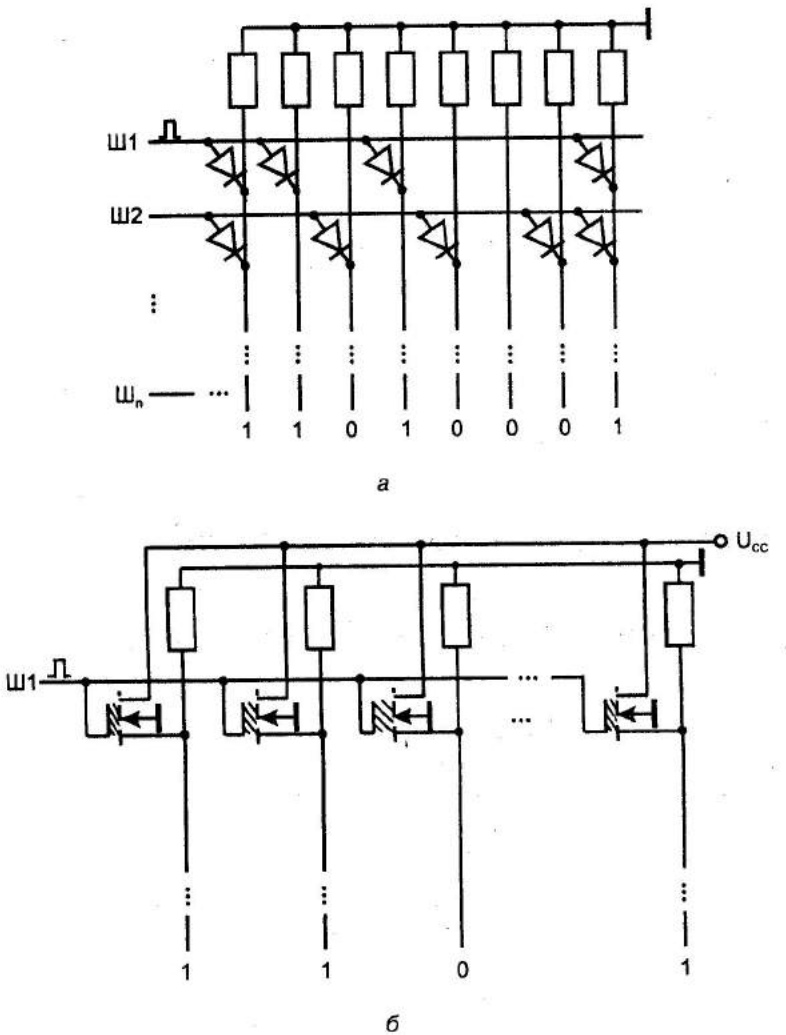


Рис. 4.14. Матрица диодных запоминающих элементов масочного ЗУ (*а*) и матрица МОП-транзисторных элементов (*б*)

Считываемое слово определяется расположением диодов в узлах координатной сетки. При наличии диода высокий потенциал выбранной горизонтальной линии передается на соответствующую вертикальную линию, и в данном разряде слова появляется сигнал логической единицы. При отсутствии

диода потенциал близок к нулевому, т. к. вертикальная линия через резистор связана с землей. В изображенной матрице при возбуждении линии выборки Ш1[1] считывается слово 11010001 (в ячейке номер один хранится это слово). При возбуждении линии Ш2 считывается слово 10101011 (оно хранится в ячейке номер 2). Шины выборки являются выходами дешифратора адреса, каждая адресная комбинация возбуждает свой выход дешифратора, что приводит к считыванию слова из адресуемой ячейки.

В матрице с диодными элементами в одних узлах диоды изготовляются, в других — нет. Схемы с диодными элементами просты, но их недостатком является отсутствие усилительных свойств ЗЭ, так что большие емкости разрядных шин нужно перезаряжать токами, потребляемыми от линий выборки.

Чтобы удешевить производство, при изготовлении масочных ЗУ стремятся варьировать только один шаблон, так чтобы одни элементы связи были законченными и работоспособными, а другие — незавершенными и как бы отсутствующими. Для этого, например, в МОП-транзисторах, соответствующих хранению нуля, увеличивают толщину подзатворного окисла, что ведет к увеличению порогового напряжения транзистора. В этом случае рабочие напряжения ЗУ не в состоянии открыть транзистор. Постоянно закрытое состояние транзистора аналогично его отсутствию. Матрица с МОП-транзисторами показана на рис. 4.14, *б*.

ЗУ с масочным программированием отличаются компактностью запоминающих элементов и, следовательно, высоким уровнем интеграции. Они имеют высокое быстродействие (времена доступа у стандартных микросхем составляют 25—70 нс). Однако при недостаточной тиражности ЗУ с масочным программированием затраты на проектирование и изготовление шаблонов для них окажутся чрезмерно высокими. Отсюда видна и область применения масочных ЗУ — хранение стандартной информации, имеющей широкий круг потребителей. В частности, масочные ЗУ используют как знакогенераторы, где они имеют в качестве "прошивки" коды букв алфавитов (русского и латинского), как таблицы типовых функций (синуса, квадратичной функции и др.), как стандартное программное обеспечение и т. п.

Примечание

Термином "прошивка" иногда называют содержимое постоянной памяти. Это название появилось во времена памяти на ферритовых сердечниках, когда информация заносилась в ЗУ путем пропускания провода через определенные сердечники.

[1] В литературе, посвященной памяти, "шинами" часто называют отдельные линии (в противоположность литературе по микропроцессорной технике).

Отечественные масочные ROM характеризуются в настоящее время следующими параметрами: информационной емкостью до 1 Мбит при временах доступа около 200 нс. Зарубежные масочные ROM имеют информационные емкости до 128 Мбит при временах доступа 40—100 нс.

ЗУ типа PROM

В однократно программируемые ЗУ типа PROM информация записывается потребителем в лабораторных условиях с помощью несложных программаторов.

Микросхемы PROM программируются удалением или созданием специальных *перемычек*. В исходной заготовке имеются (или отсутствуют) все перемычки. После программирования остаются (или возникают) только необходимые.

Удаление части перемычек свойственно ЗУ с плавкими перемычками (типа *fuse*). При этом в исходном состоянии ЗУ имеет все перемычки, а при программировании часть их ликвидируется путем расплавления импульсами тока достаточно большой амплитуды и длительности. Плавкие перемычки включаются в электроды диодов или транзисторов. Перемычки могут быть металлическими (вначале изготовлялись из нихрома, позднее из титано-вольфрамовых и других сплавов) или поликристаллическими (кремниевыми). В исходном состоянии запоминающий элемент хранит логическую единицу, логический нуль нужно записать, расплавляя перемычку.

Схемы с создаваемыми перемычками в исходном состоянии имеют непроводящие участки в виде пары встречно включенных диодов или тонких диэлектрических слоев, пробиваемых при программировании с образованием низкоомных сопротивлений. Схемы с тонкими пробиваемыми диэлектрическими перемычками (типа *antifuse*) наиболее компактны и совершенны. Их применение характерно для программируемых логических СБИС *(см. главу 9)*.

Сопротивление запоминающего элемента с двумя встречновключенными диодами в исходном состоянии практически равноценно разомкнутой цепи, и запоминающий элемент хранит логический нуль. Для записи единицы к диодам прикладывают повышенное напряжение, пробивающее диод, смещенный в обратном направлении. Диод пробивается с образованием в нем короткого замыкания и играет роль появившейся проводящей перемычки.

Запоминающие элементы с плавкими перемычками и парами диодов показаны на рис. 4.15, *а*, *б* в исходном состоянии и после программирования.

Матрица запоминающих элементов с плавкими перемычками в технике ТТЛ (микросхема К155РЕ3) показана на рис. 4.15, *в*. ЗУ имеет организацию 32×8. Матрица содержит 32 транзистора с 9 эмиттерами в каждом (8 рабочих и один технологический для уточнения режима прожигания, технологический

эмиттер на рисунке не показан). Высокий потенциал на какой-либо шине выборки активизирует соответствующий транзистор, работающий в режиме эмиттерного повторителя.

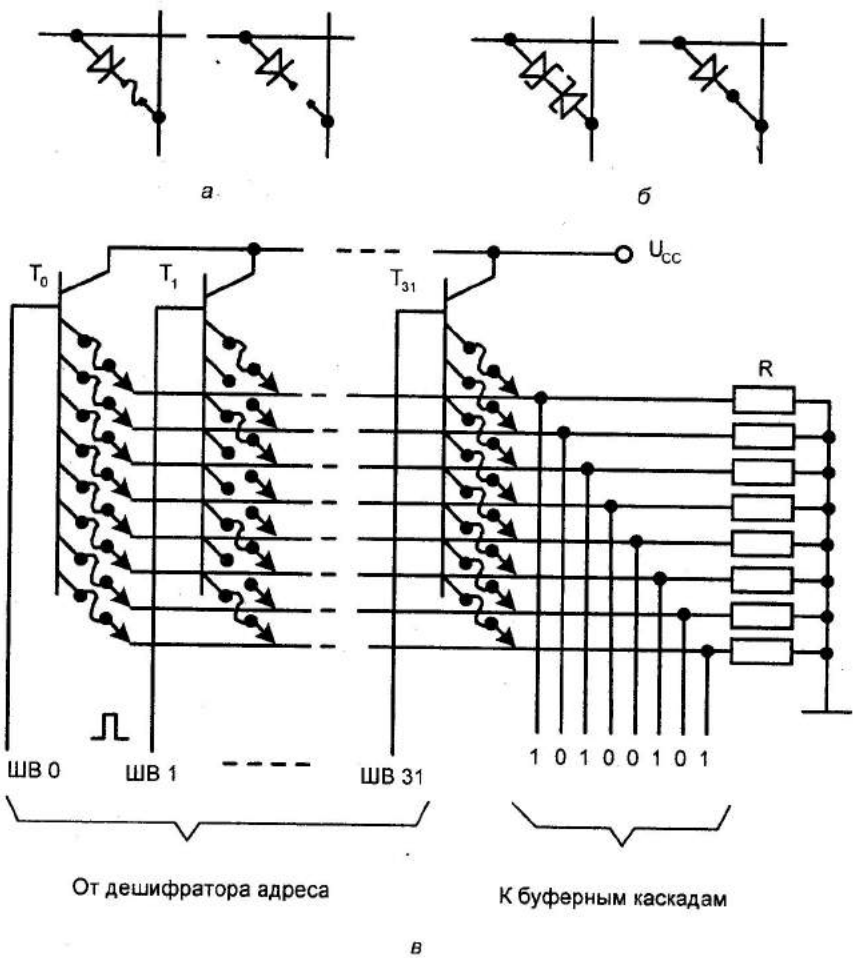


Рис. 4.15. Запоминающие элементы с плавкими перемычками (*а*), диодными парами (*б*) и матрица запоминающих элементов с плавкими перемычками в технике ТТЛ (*в*)

До программирования транзисторы передают высокий потенциал базы на все выходные (разрядные) линии, т. е. по всем адресам записаны слова, состоящие из одних единиц. Пережигание перемычки в цепи какого-либо эмиттера дает нуль в данном разряде слова, например, для ячейки с номером 1 показан вариант программирования для хранения по этому адресу

слова 10100101. Выходы матрицы связаны с внешними цепями через буферные каскады, имеющие выходы типа ОК или ТС. ЗУ имеет структуру 2D.

Программирование ЗУ с плавкими перемычками реализуется простыми аппаратными средствами и может быть доступно схемотехникам даже при отсутствии специального оборудования. На рис. 4.16 показан многоэмиттерный транзистор (МЭТ) с плавкими перемычками и дополнительными элементами, обеспечивающими программирование ЗУ. Выходы этого запоминающего элемента передаются во внешние цепи через буферные каскады с тремя состояниями, работа которых разрешается сигналом OE. При этом сигнал разрешения работы формирователей импульсов программирования OE_F отсутствует, и они не влияют на работу схемы. При программировании буферы данных переводятся в третье состояние (OE = 0), а работа формирователей F разрешается. Слово, которое нужно записать в данной ячейке, подается на линии данных $D_7...D_0$. Те разряды слова, в которых имеются единицы, будут иметь на выходах формирователей низкий уровень напряжения. Соответствующие эмиттеры МЭТ окажутся под низким напряжением и через них пройдет ток прожигания перемычки. При чтении отсутствие перемычки даст нулевой сигнал на вход буфера данных. Так как буфер инвертирующий, с его выхода снимется единичный сигнал, т. е. тот, который и записывался. Адресация программируемой ячейки как обычно обеспечивается дешифратором адреса, подающим высокий уровень потенциала на базу адресуемого МЭТ.

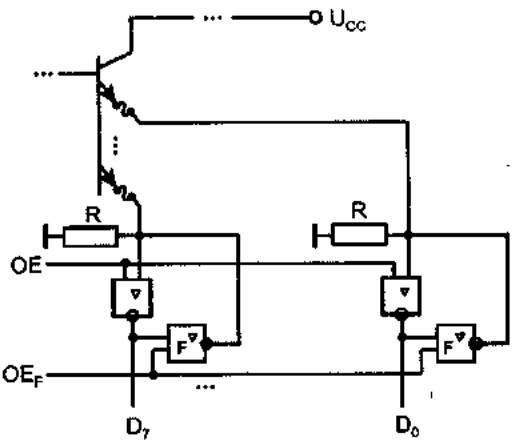


Рис. 4.16. Схема запоминающей ячейки с элементами программирования плавких перемычек

Для прожигания перемычек используются серии импульсов тока в десятки миллиампер (для большей надежности прожигания). Не все перемычки удается пережечь надлежащим образом, коэффициент программируемости для серии К556, например, составлял 0,5—0,7. В ЗУ с плавкими перемычками возможно восстановление проводимости перемычек через некоторое время из-за явления миграции в электроматериалах.

Плавкие перемычки занимают на кристалле относительно много места, поэтому уровень интеграции ЗУ с такими перемычками существенно ниже, чем у масочных ЗУ. В то же время простота программирования пользователем и невысокая стоимость в свое время обусловили широкое распространение ЗУ типа PROM (среди отечественных PROM имеются микросхемы серии К556, с информационной емкостью 1—64 Кбит и временем доступа по адресу 70—90 нс).

Внешняя организация памяти типов ROM(M) и PROM проста: входными сигналами для них служат адресный код и сигнал выбора микросхемы CS. Во времени последовательность сигналов следующая: вначале подается адресный код (чтобы произошла дешифрация адреса и было исключено обращение к непредусмотренной ячейке), затем поступает сигнал выбора микросхемы CS и после задержки, определяемой быстродействием схемы, на выходах данных устанавливаются правильные значения считываемых сигналов. Для повышения быстродействия в микросхемах постоянной памяти, как и в других типах ЗУ, может быть организован пакетный и конвейерный режимы доступа.

К микросхемам с однократным программированием относятся также ЗУ типа EPROM-OTP, о которых сказано далее, поскольку они являются упрощенным вариантом репрограммируемых ЗУ типа EPROM, рассматриваемых в следующем подпараграфе.

Отечественные микросхемы с плавкими перемычками характеризуются информационными емкостями до 64 Кбит и временами доступа около 80 нс.

ЗУ типов EPROM, EPROM-OTP и EEPROM

В репрограммируемых ЗУ типов EPROM и EEPROM (E^2PROM) возможно стирание старой информации и замена ее новой в результате специального процесса. Рабочий режим (чтение данных) — процесс, выполняемый с высокой скоростью. Замена же содержимого памяти требует выполнения гораздо более длительных и сложных операций.

В микросхемах EPROM (или РПЗУ-УФ) старая информация стирается ультрафиолетовыми лучами, в EEPROM (или РПЗУ-ЭС) — электрическими сигналами. Запоминающие элементы РПЗУ — транзисторы типа МНОП или транзисторы с плавающим затвором.

МНОП-транзистор отличается от обычного двухслойным подзатворным диэлектриком. На поверхности кристалла расположен тонкий слой двуокиси кремния SiO_2, далее более толстый слой нитрида кремния Si_3N_4 и затем уже затвор (рис. 4.17, *а*). На границе диэлектрических слоев SiO_2 и Si_3N_4 возникают центры захвата заряда. Благодаря туннельному эффекту носители заряда могут проходить через тонкую пленку окисла толщиной около 5 нм и скапливаться на границе раздела слоев. Этот заряд и является носителем информации, хранимой МНОП-транзистором. Заряд записывают созданием

под затвором напряженности электрического поля, достаточной для возникновения туннельного перехода носителей заряда через тонкий слой SiO_2. На границе раздела диэлектрических слоев можно создавать заряд любого знака в зависимости от направленности электрического поля в подзатворной области. Наличие заряда влияет на пороговое напряжение транзистора.

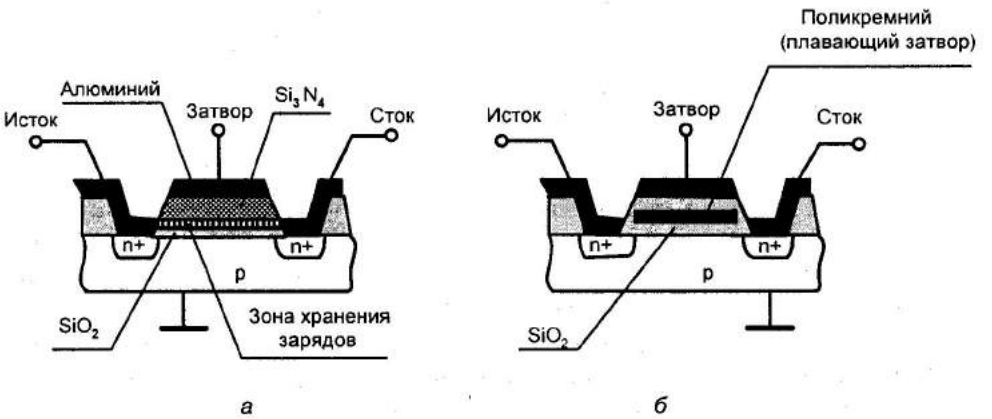


Рис. 4.17. Структура транзистора типа МНОП (*а*) и транзистора с плавающим и управляющим затворами (*б*)

Для МНОП-транзистора с n-каналом отрицательный заряд на границе раздела слоев повышает пороговое напряжение (экранирует воздействие положительного напряжения на затворе, отпирающего транзистор). При этом пороговое напряжение возрастает настолько, что рабочие напряжения на затворе транзистора не в состоянии его открыть (создать в нем проводящий канал). Транзистор, в котором заряд отсутствует или имеет другой знак, легко открывается рабочим значением напряжения на затворе. Так осуществляется хранение бита в МНОП: одно из состояний трактуется как отображение логической единицы, другое — нуля.

При программировании ЗУ используются относительно высокие напряжения. После снятия высоких напряжений туннельное прохождение носителей заряда через диэлектрик прекращается и заданное транзистору пороговое напряжение остается неизменным. Перед новой записью старая информация стирается записью нулей во все запоминающие элементы.

После 10^4—10^6 перезаписей МНОП-транзистор перестает устойчиво хранить заряд. РПЗУ на МНОП-транзисторах энергонезависимы и могут хранить информацию месяцами, годами и десятками лет.

Среди отечественных РПЗУ схемы на основе МНОП-транзисторов представлены достаточно широко, среди зарубежных их мало. Но следует отметить,

что конструкция с неоднородной областью диэлектрика, составленного из разных слоев, граница раздела между которыми играет роль ловушки для носителей электрического заряда, реализованная в МНОП-транзисторах, вполне *может получить новую жизнь* в перспективных энергонезависимых ЗУ будущего. Уже исследуются перспективные ЗУ подобного типа на основе новых материалов, обладающие чрезвычайно малыми размерами запоминающих элементов при выдающихся технических характеристиках.

Транзисторы с плавающим затвором имеют в подзатворном диэлектрике замкнутую проводящую область, которая называется *плавающим затвором* и в которую может быть введен электрический заряд. Вначале такие транзисторы были представлены вариантом, называемым ЛИЗМОП. Добавление ЛИЗ к обозначению МОП происходит от слов Лавинная Инжекция Заряда, так что аббревиатура в целом обозначает "МОП-транзистор с лавинной инжекцией заряда". Позднее появились транзисторы типа FLOTOX, в которых процессы ввода и удаления заряда происходят с помощью других механизмов (туннелирования электронов через тонкие слои диэлектрика). Еще позднее в транзисторах с плавающим затвором стали использовать разные механизмы для введения и удаления заряда (и лавинную инжекцию и туннелирование).

Плавающий затвор может быть единственным или вторым затвором транзистора (дополнительным к обычному управляющему затвору). Транзисторы с только одним плавающим затвором использовались в ЗУ типа EPROM, а транзисторы с двойным затвором пригодны для применения как в EPROM, так и в EEPROM, играющих в современных разработках основную роль. По принципу работы транзистор с двойным затвором (рис. 4.17, *б*) близок к МНОП-транзистору — здесь также между управляющим затвором и каналом имеется область, в которую при программировании можно вводить заряд, влияющий на величину порогового напряжения. Только область введения заряда представляет собою не границу раздела двух разнородных слоев диэлектрика, а окруженную со всех сторон диэлектриком проводящую область (обычно из поликристаллического кремния), в которую, как в ловушку, можно ввести заряд, способный сохраняться в ней в течение очень длительного времени. Эта область и является *плавающим затвором*.

При подаче на управляющий затвор и сток транзистора типа ЛИЗМОП положительных напряжений относительно большой величины в обратно смещенных p-n-переходах возникает лавинный пробой, область которого насыщается свободными электронами. Часть электронов, имеющих энергию, достаточную для преодоления потенциального барьера диэлектрической области, проникает в плавающий затвор. Снятие высокого программирующего напряжения восстанавливает непроводящее состояние диэлектрических областей транзистора и запирает электроны в плавающем затворе, где они могут находиться длительное время (десятки лет). Процесс ввода заряда в плавающий затвор саморегулируется — отрицательный заряд, накапливаю-

щийся в плавающем затворе, ослабляет электрическое поле в окисле и при определенном уровне заряда это поле становится неспособным ускорять "горячие" электроны для ввода их в плавающий затвор. Саморегулируемость заряда позволяет строить запоминающие элементы на одном ЛИЗМОП-транзисторе, что делает ячейки памяти EPROM компактными и дешевыми.

Заряженный электронами плавающий затвор увеличивает пороговое напряжение транзистора настолько, что в диапазоне рабочих напряжений проводящий канал в транзисторе не создается. При отсутствии заряда в плавающем затворе транзистор работает в обычном ключевом режиме.

Стирание информации в транзисторах с плавающим затвором может производиться двумя способами — ультрафиолетовым облучением или электрическими сигналами.

В первом случае (в памяти типа EPROM) корпус ИС имеет специальное прозрачное окошко для облучения кристалла. Двуокись кремния и поликремний прозрачны для ультрафиолетовых лучей. Эти лучи вызывают в областях транзистора фототоки и тепловые токи, что делает области прибора проводящими и позволяет заряду покинуть плавающий затвор. После стирания старой информации окошко в корпусе заклеивают, чтобы избежать воздействия света на поверхность кристалла. Операция стирания информации этим способом занимает десятки минут, информация стирается сразу во всем кристалле. В схемах с УФ-стиранием *число циклов перепрограммирования существенно ограничено* (10—1000 циклов у приборов разного качества), т. к. под действием ультрафиолетовых лучей свойства материалов постепенно изменяются.

Электрическое стирание информации стало осуществляться в транзисторах типа FLOTOX, затем появились и другие технологии (ETOX в разных модификациях и др.). Конструктивно эти транзисторы отличаются от предшественников более тонким слоем подзатворного диэлектрика (10 нм или меньше). При приложении к тонкому слою диэлектрика напряжений порядка 10 В электроны проходят через диэлектрик в том или ином направлении в зависимости от знака напряжения по механизму туннелирования Фаулера—Нордхайма (Fowler—Nordheim). Этим объясняется и название транзисторов FLOTOX — Floating-gate Tunneling Oxide (название ETOX происходит от Extremely Thin Oxide). Возможность пользоваться электрическими воздействиями, как для ввода, так и для вывода заряда весьма ценна, однако при этом оказалась потерянной саморегулируемость процесса заряда и возникла проблема его контроля. Выход был найден с помощью построения запоминающего элемента на двух транзисторах — одном с плавающим затвором и другом ключевом (обычного типа). Поэтому площадь запоминающего элемента памяти EEPROM больше, чем у памяти EPROM, а стоимость выше (в том числе из-за сложности изготовления тонких слоев подзатворного окисла).

Электрическое стирание имеет преимущества — можно стирать информацию не со всего кристалла, а выборочно (в приборах типа EEPROM индивидуально для каждого адреса). Длительность процесса "стирание-запись" значительно меньше, сильно ослабляются ограничения на число циклов перепрограммирования (допускается 10^4—10^6 таких циклов). Кроме того, перепрограммировать ЗУ можно, не извлекая микросхему из устройства, в котором она работает. В то же время схемы с электрическим стиранием занимают больше места на кристалле, в связи с чем уровень их интеграции меньше, а стоимость выше. Однако эти недостатки быстро преодолеваются и ЭС-стирание вытесняет УФ-стирание.

Предшественниками двухзатворных транзисторов были однозатворные, имевшие только плавающий затвор. Эти транзисторы изготовлялись обычно с p-каналом, поэтому (в противоположность транзисторам с n-каналом) введение электронов в плавающий затвор приводило к созданию в транзисторе проводящего канала, а удаление заряда — к исчезновению такого канала. Стирание информации производилось ультрафиолетовыми лучами.

Для стирания данных с помощью ультрафиолетовых лучей кристаллы EPROM помещают в *специальные корпуса с прозрачным окошком*, через которое облучается кристалл. Такие корпуса имеют относительно высокую стоимость. Отказ от сложных корпусов и переход к обычным дешевым пластмассовым корпусам без окошка привел к разновидности однократно программируемых ЗУ типа EPROM-OTP (OTP означает One Time Programmable). В этих ставших популярными ЗУ запись содержимого производится как в обычных EPROM электрическими сигналами путем заряда соответствующих плавающих затворов. Изменить содержимое памяти уже нельзя, т. к. отсутствие в корпусе окошка исключает возможность УФ-облучения кристалла.

> Заметим, что принципиально возможно получить репрограммируемые ЗУ и в пластмассовых корпусах без окошка, если применить стирание данных с помощью рентгеновского облучения. Более того, эта технология применялась, но оказалась, по крайней мере пока, недостаточно практичной.

Подключение двухзатворных транзисторов к линиям выборки строк и линиям чтения в матрицах ЗУ показано на рис. 4.18. Запись логического нуля осуществляется путем заряда плавающего затвора инжекцией "горячих" электронов в режиме программирования. Стирание информации, под которым понимается удаление заряда из плавающего затвора, приводит к записи во все запоминающие элементы логических единиц, т. к. в данном случае опрашиваемые транзисторы открываются и передают напряжение U_{CC} на линии считывания.

Среди отечественных EPROM (в маркировке они имеют буквы РФ) наиболее известна серия К573 с широким набором типономиналов, а среди

EEPROM (в маркировке имеют буквы РР) имеются серии КР558 (на основе n-МНОП), К1609, К1624, К1626 на ЛИЗМОП с двумя затворами.

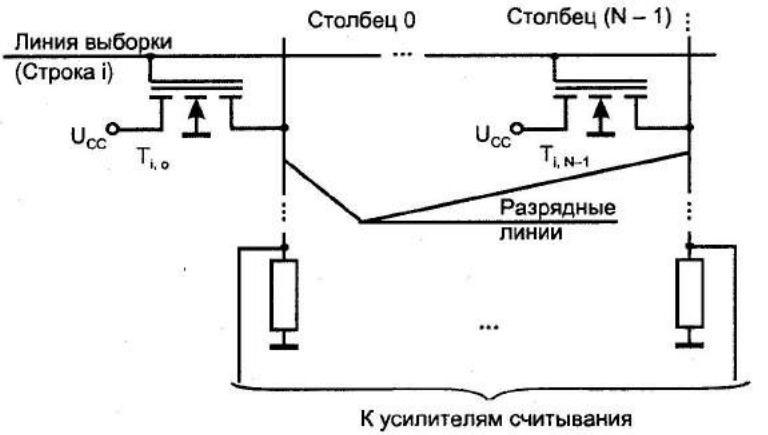


Рис. 4.18. Схема подключения транзисторов с двойным затвором к линиям выборки и считывания в РПЗУ

Основные параметры отечественных EPROM — информационная емкость до 1 Мбит и время доступа 350 нс, EEPROM — 64 Кбита и 250 нс соответственно.

На уровне мировой техники имеются ЗУ типов EPROM и EEPROM с информационной емкостью до 4 Мбит при тактовых частотах до 80 МГц и напряжениях питания 2,3—3,6 В.

На рис. 4.19 показана полная, хотя и несколько упрощенная, схема одного из современных ЗУ типа EPROM (на примере микросхем фирмы Cypress Semiconductor).

Ядром схемы является структура 2DM, содержащая блоки адресного дешифратора, матрицу запоминающих элементов и блок мультиплексоров. Это ядро дополнено регистром, принимающим данные по фронту синхросигнала CP, выходными буферами с третьим состоянием выходов и логикой управления буферами. Введение в схему регистра вносит в ее работу элементы параллелизма операций и повышает быстродействие ЗУ. Действительно, благодаря регистру сразу же после приема им данных можно изменять адрес и переходить к чтению следующего слова, не дожидаясь, пока предыдущее слово будет использовано устройством-приемником данных, т. к. во время выборки следующего слова предыдущее сохраняется регистром. Управление буферами с помощью двух сигналов разрешения E и E_S повышает гибкость использования микросхемы.

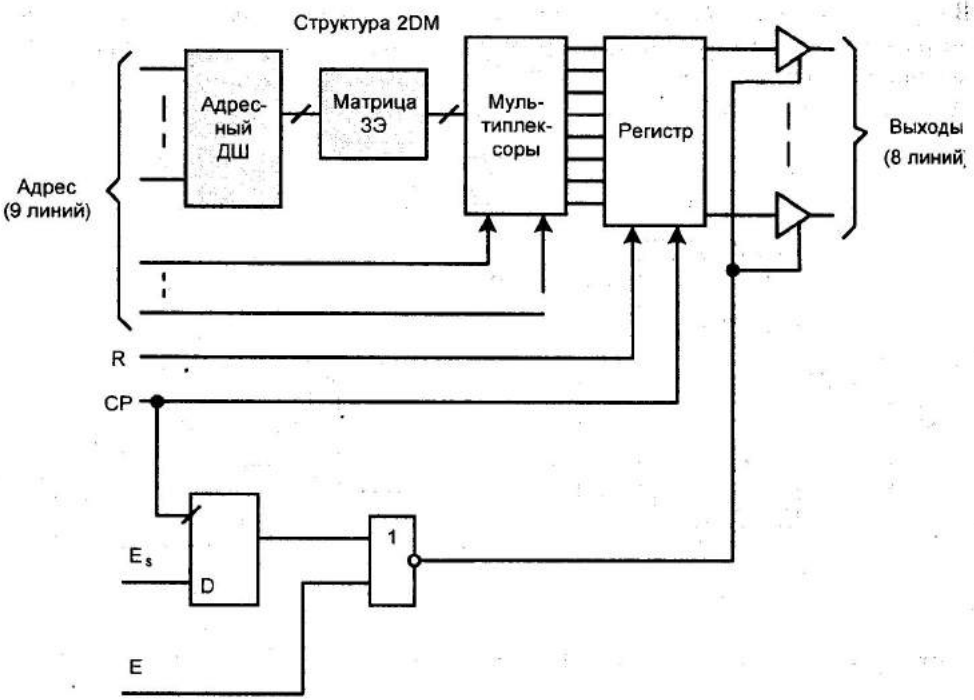


Рис. 4.19. Структура ЗУ типа EPROM

Конкретные цифры, приведенные на рис. 4.19, соответствуют микросхеме с информационной емкостью 4 Кбит при организации 512×8. В семействе микросхем, к которому она принадлежит, имеются представители с информационными емкостями от 4 Кбит до 256 Кбит. Микросхемы имеют напряжение питания 5 В, время доступа по адресу 25 нс и, как сказано в их описании, 100%-ую программируемость.

Импульсное питание ROM

Энергонезависимость всех ROM, сохраняющих информацию при отключении питания, открывает возможности экономии питания при их эксплуатации и, соответственно, улучшения их теплового режима, что повышает надежность схем. Питание можно подавать только на ИС, к которой в данный момент происходит обращение. На рис. 4.20 показан обычный вариант построения модуля памяти, состоящего из нескольких ИС, и вариант с импульсным питанием. В обычном варианте напряжение U_{CC} подключается ко всем ИС постоянно, а выбор адресуемой ИС осуществляется сигналом $\overline{CS}$.

В варианте с импульсным питанием работа всех ИС по входам $\overline{CS}$ постоянно разрешена, но питание подключается только к выбранной микросхеме с помощью ключа, управляемого от выходов адресного дешифратора, декодирующего старшие разряды адреса.

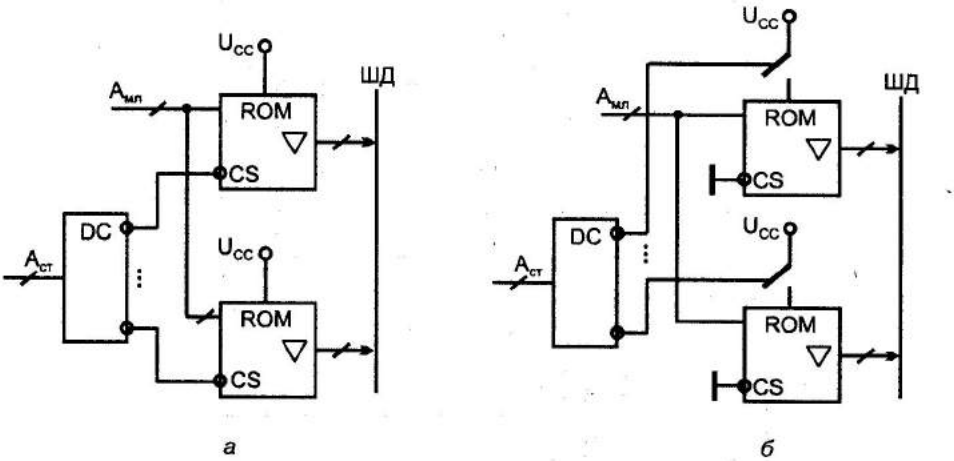


Рис. 4.20. Модули постоянной памяти с обычным (*а*) и импульсным (*б*) питанием

Режим импульсного питания может многократно уменьшить потребляемую модулем мощность, но одновременно увеличивает время обращения к ЗУ при одиночных произвольных обращениях, т. к. после включения питания необходимо время для установления режима ИС.

При чтении данных, расположенных по близким адресам, когда старшие разряды адреса остаются неизменными, потерь времени не возникает.

§ 4.5. Флэш-память

Флэш-память (Flash-Memory) по основным принципам работы подобна рассмотренным выше ЗУ с программированием плавающих затворов, но имеет и ряд особенностей. Разработка флэш-памяти считается кульминацией многолетнего развития схемотехники энергонезависимой памяти с электрическим стиранием информации.

В схемах флэш-памяти, как и схемах EEPROM, данные стираются электрическими сигналами, но *не предусмотрено стирание отдельных слов*, стирание осуществляется либо для всей памяти одновременно, либо для достаточно больших блоков, что позволяет исключить схемы управления записью/стиранием с разрешающей способностью до каждого слова (байта) и тем самым упростить схемы ЗУ, т. е. способствует достижению высокого

уровня интеграции и быстродействия при снижении стоимости. Технологически схемы флэш-памяти выполняются с высоким качеством и обладают очень хорошими параметрами. Запоминание данных осуществляется с помощью зарядов-разрядов плавающих затворов. Заряд плавающих затворов в большинстве вариантов флэш-памяти производится с помощью лавинной инжекции, при стирании используется туннелирование электронов через тонкий слой диэлектрика.

Объединение в схемотехнике флэш-памяти достоинств ее предшественников иллюстрируется рис. 4.21.

Рис. 4.21. Сопоставление схемотехнических свойств микросхем EPROM, EEPROM и Flash

Термин flash по одной из версий связан с характерной особенностью этого вида памяти — возможностью одновременного стирания всего ее объема. Согласно этой версии, еще до появления флэш-памяти при хранении секретных данных использовались устройства, которые при попытках несанкционированного доступа к ним автоматически стирали хранимую информацию и назывались устройствами типа flash (вспышка, мгновение). Это название перешло и к памяти, обладавшей свойством быстрого стирания всего массива данных одним сигналом.

Одновременное стирание всей информации ЗУ реализуется наиболее просто, но имеет тот недостаток, что даже замена одного слова в ЗУ требует стирания и новой записи для всего ЗУ в целом. Для большинства применений это неудобно. Поэтому стали широко применяться *схемы с блочной структурой*, в которых весь массив памяти делится на блоки, стираемые независимо друг от друга. В настоящее время микросхемы со стиранием сразу всего содержимого ЗУ практически полностью уступили место блочным (секторизованным) структурам.

Число циклов репрограммирования для флэш-памяти хотя и велико, но ограничено, т. е. ячейки при перезаписи хоть и очень мало, но "изнашиваются" (в этом отношении флэш-память несколько уступает памяти типа EEPROM). Чтобы увеличить долговечность флэш-памяти, в управлении ею используются специальные алгоритмы, способствующие "разравниванию" числа перезаписей по всем блокам микросхемы.

Структуры с несимметричными и симметричными блоками

Соответственно областям применения, флэш-память имеет архитектурные и схемотехнические разновидности. *Двумя основными направлениями эффективного использования флэш-памяти являются:*

- хранение не очень часто изменяемых данных (обновляемых программ, в частности);
- замена памяти на жестких магнитных дисках.

Для первого направления в связи с редким обновлением содержимого параметры циклов стирания и записи не столь существенны как информационная емкость и скорость считывания информации. Микросхемы первого направления имеют специализированные блоки *(несимметричные блочные структуры)*. По имени так называемых Boot-блоков, в которых информация надежно защищена аппаратными средствами от случайного стирания, эти ЗУ называют *Boot Block Flash Memory*. Boot-блоки хранят программы инициализации системы, позволяющие ввести ее в рабочее состояние после включения питания.

Микросхемы второго направления для замены жестких магнитных дисков *(Flash-File Memory)* имеют более развитые средства перезаписи информации и идентичные блоки *(симметричные блочные структуры)*.

Структуры с ячейками ИЛИ-НЕ и И-НЕ

Основой структуры флэш-памяти является накопитель (матрица запоминающих элементов). В схемотехнике накопителей развиваются два направления:

- на основе ячеек ИЛИ-НЕ (NOR);
- на основе ячеек И-НЕ (NAND).

Схемы обеих ячеек показаны на рис. 4.22. Эти схемы при использовании обычных транзисторов применяются в логических преобразователях, тогда как в ЗУ вместо обычных транзисторов используются транзисторы с плавающим затвором.

В ячейке ИЛИ-НЕ (рис. 4.22, *а*) транзисторы соединены параллельно и для формирования нулевого значения функции F достаточно включения хотя бы одного транзистора. При работе такой ячейки в матрице ЗУ все транзисторы, кроме адресованного, находятся в запертом состоянии, так что выходной сигнал определяется исключительно выбранным транзистором — если в его плавающем затворе имеется заряд, то транзистор под действием сигнала выборки не откроется, и на выходе схемы будет высокое напряжение. При

отсутствии заряда в плавающем затворе сигнал выборки откроет опрашиваемый транзистор, и выходное напряжение схемы окажется низким.

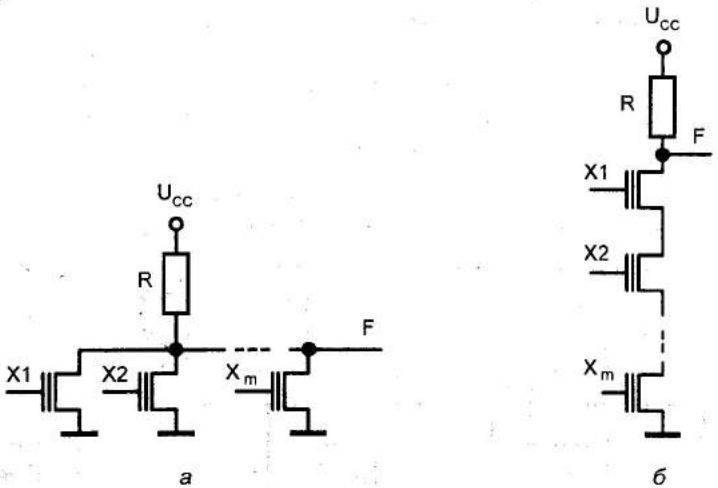


Рис. 4.22. Схемы на основе ячеек ИЛИ-НЕ (*а*) и И-НЕ (*б*)

В ячейке И-НЕ (рис. 4.22, *б*) транзисторы соединены последовательно и для формирования высокого уровня выходного напряжения достаточно наличия в цепочке транзисторов хотя бы одного запертого. При работе таких ячеек в составе ЗУ все транзисторы, кроме адресуемого, должны быть открыты, так что состояние выхода будет зависеть только от выбранного транзистора, т. е. наличия или отсутствия заряда в его плавающем затворе. При заряженном плавающем затворе опрашиваемый транзистор не откроется, и выходное напряжение схемы будет высоким, при разряженном затворе транзистор откроется сигналом выборки, и выходное напряжение схемы окажется низким.

Каждая из ячеек имеет свои особенности, здесь отметим лишь меньшее быстродействие и большую компактность ячеек И-НЕ. Повышенная компактность ячеек И-НЕ объясняется тем, что, в отличие от ячейки ИЛИ-НЕ, заземляющий контакт имеют в ней не все транзисторы, а только один (нижний). При этом области стока одного транзистора и истока соседнего, имеющие один и тот же тип проводимости, объединяются, также сокращая площадь, занимаемую ячейкой.

Накопители на основе ячеек ИЛИ-НЕ обеспечивают быстрый доступ к словам при произвольной выборке. Они приемлемы для разных применений, но наиболее бесспорным считается их использование в памяти типа Boot-Block. При этом возникает полезная преемственность с применявшимися ранее ROM и EPROM, сохраняются типичные сигналы управления, обеспечивающие чтение с произвольной выборкой. Структура матрицы накопите-

ля показана на рис. 4.23. Каждый столбец представляет собою совокупность параллельно соединенных транзисторов. Разрядные линии выборки находятся под высоким потенциалом. Все транзисторы невыбранных строк заперты. В выбранной строке открываются и передают высокий уровень напряжения на разрядные линии считывания те транзисторы, в плавающих затворах которых отсутствует заряд электронов, и, следовательно, пороговое напряжение транзистора имеет нормальное (не повышенное) значение.

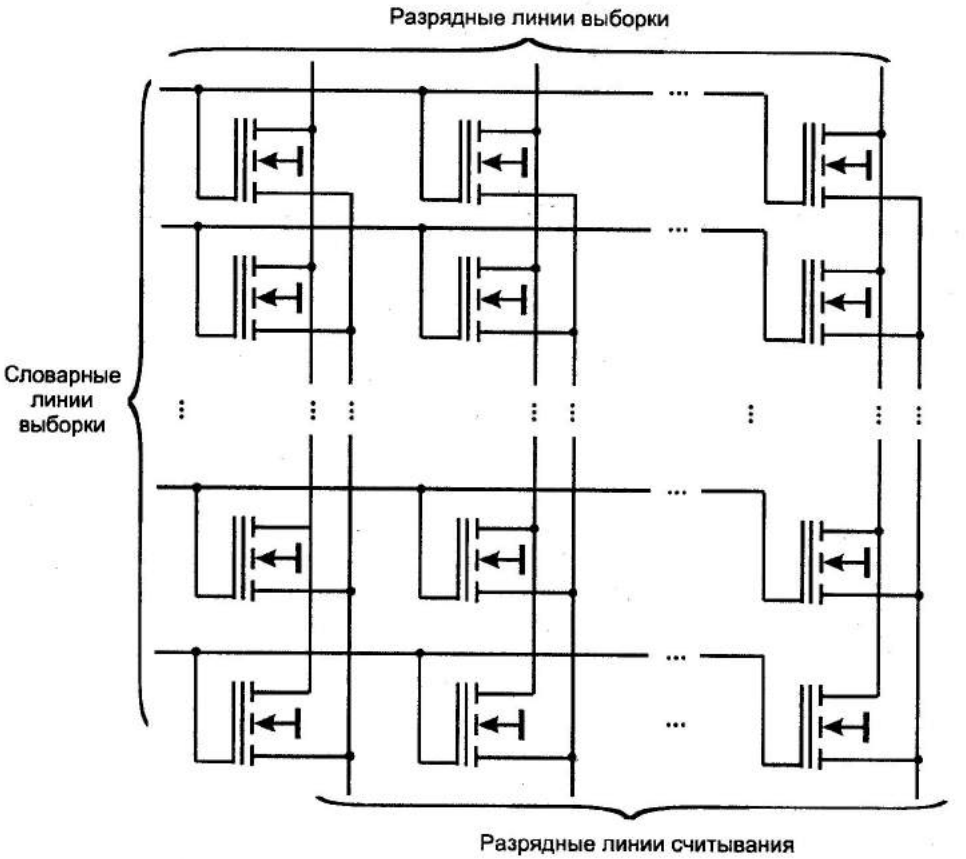


Рис. 4.23. Структура матрицы накопителя флэш-памяти на основе ячеек ИЛИ-НЕ

Накопители на основе ячеек ИЛИ-НЕ широко используются фирмой Intel. Имеются мнения о конкурентоспособности этих накопителей и в применениях, связанных с заменой жестких магнитных дисков флэш-памятью.

Структуры с ячейками И-НЕ более компактны, но не обеспечивают режима произвольного доступа и используются, прежде всего, в схемах замены маг-

нитных дисков. В схемах на этих ячейках сам накопитель компактнее, но увеличивается количество логических элементов обрамления накопителя. В целом же такие микросхемы имеют очень высокий уровень интеграции. При топологических нормах 0,13 мкм была достигнута емкость памяти в 1 Гбит. Во флэш-памяти с ячейками И-НЕ (NAND), как и в памяти с ячейками ИЛИ-НЕ (NOR), успешно применяют многоуровневое хранение заряда (при хранении в одной ячейке двух битов на той же площади кристалла была достигнута информационная емкость 2 Гбита). Последовательный характер доступа к данным в памяти с ячейками И-НЕ приводит к задержке чтения первого слова массива на время порядка 1 мкс при задержках 60—80 нс для чтения последующих слов массива. Обе разновидности флэш-памяти имеют хорошие перспективы, сейчас по объему производства доминирует вариант с ячейками ИЛИ-НЕ, имеющий произвольный доступ к ячейкам памяти (этот вариант занимает сейчас 80—90% общего рынка флэш-памяти). Относительно будущего имеются различные мнения, в том числе и прогнозирующие преимущественное развитие вариантов с ячейками И-НЕ.

К недостаткам флэш-памяти относятся несимметричность операций чтения/записи по временным параметрам (запись намного медленнее, чем чтение), хотя и большое, но все же ограниченное число циклов программирования.

Способы улучшения параметров флэш-памяти

Для улучшения параметров в схемах флэш-памяти применяются различные средства и приемы.

- Прерывание процессов записи при обращениях процессора для чтения (Erase Suspend). Без этого возникали бы длительные простои процессора, т. к. запись занимает достаточно большое время. После прерывания процесс записи возобновляется под управлением внутренних средств флэш-памяти.
- Внутренняя очередь команд, управляющих работой флэш-памяти, которая позволяет организовать конвейеризацию выполняемых операций и ускорить процессы чтения и записи.
- Программирование длины хранимых в ЗУ слов для согласования с различными портами ввода/вывода.
- Введение режимов пониженной мощности на время, когда к ЗУ нет обращений, в том числе режима глубокого покоя, в котором мощность снижается до крайне малых значений (например, ток потребления снижается до 2 мкА). Эти особенности очень важны для устройств с автономным (батарейным) питанием.
- Приспособленность к работе при различных питающих напряжениях (5 В; 3,3 В и др.). Сама схема "чувствует" уровень питания и производит необходимые переключения для приспособления к нему.

- Введение в структуры памяти страничных буферов для быстрого накопления новых данных, подлежащих записи. Два таких буфера могут работать в режиме, называемом "пинг-понг", когда один из них принимает слова, подлежащие записи, а другой в это время обеспечивает запись своего содержимого в память. Когда первый буфер заполнится, второй уже освободится, и они поменяются местами.
- Различные меры защиты от случайного или несанкционированного доступа.

Имея преемственность с ЗУ типов EEPROM и EPROM, разработанными ранее, схемы флэш-памяти предпочтительнее EEPROM по информационной емкости и стоимости в применениях, где не требуется индивидуальное стирание слов, а в сравнении с EPROM обладают тем преимуществом, что не требуют специальных условий и аппаратуры для стирания данных, которое происходит непосредственно в системе и гораздо быстрее, организуется поблочно, а также тем, что имеют менее жесткие ограничения на допустимое число циклов перезаписи. В итоге *флэш-память часто оказывается наиболее удобной для применения в самых разных условиях, особенно в портативной аппаратуре.* Если ее применение в начальный период развития сдерживалось высокой стоимостью, то в настоящее время положение изменилось к лучшему. Микросхемы флэш-памяти в 2003 году использовались в 90% всех персональных компьютеров, более чем в 90% сотовых телефонов, в 50% модемов и т. д.

Команды управления флэш-памятью

Нетрудно видеть, что в микросхемах флэш-памяти происходят более сложные процессы, чем в схемах ROM(M), EPROM-OTP или EPROM, из которых в рабочем режиме данные только читаются, для чего достаточно адресовать ячейку и разрешить ей выход на шину данных. В микросхемах флэш-памяти помимо этого непосредственно в системе выполняются операции стирания старых данных и записи новых, существенно отличающиеся от операции чтения, более сложные и длительные. Поэтому управление микросхемами флэш-памяти имеет более сложный характер. В отличие от традиционного управления схемами памяти с помощью адресных и управляющих сигналов флэш-память имеет и *управление словами-командами*, записываемыми процессором в специальный внутренний командный регистр. Слова-команды состоят из некоторого количества байтов и определяют работу внутреннего автомата управления, обеспечивающего для реализуемого режима необходимую последовательность действий.

Для слов-команд типичен следующий состав:

- две команды, относящиеся к операции стирания (подготовка стирания/стирание и проверка стирания);

- две команды, относящиеся к операции программирования (подготовка программирования/программирование и проверка программирования);
- две команды, задающие операцию чтения (данных и кодов идентификатора);
- команда, задающая операцию сброса.

По команде стирания содержимое памяти или блока стирается, после чего правильность операции стирания проверяется. Для этого содержимое ячеек читается. Чтение правильного кода (например, кода, состоящего из единиц) показывает, что все биты слова стерты. Если считывается иной код, выполняется повторная операция стирания. Затем проверка возобновляется с адреса последнего проверенного слова. Процесс проверки продолжается до достижения последнего адреса.

Программирование памяти ведется слово за словом (последовательно или при произвольном доступе). Цикл чтения от процессора выводит записанные данные, которые сравниваются с заданными. Равенство байтов свидетельствует об успешном программировании. После этого процесс программирования переходит к следующему слову.

Команда сброса является средством надежного устранения действия команд стирания/программирования. После каждой из этих команд в регистр команд можно записать код операции сброса, что устранит возможность каких-либо действий, связанных с указанными командами. Содержимое памяти не сможет изменяться. Для дальнейшего приведения схемы в желаемое состояние в регистр команд нужно записать соответствующую команду.

О номенклатуре микросхем флэш-памяти

В современной номенклатуре микросхем флэш-памяти остались только структуры блочного типа (структуры типа Bulk Erase с одновременным стиранием всего кристалла уступили место блочным структурам). Для хранения программ применяются структуры с несимметричными блоками (называемые Boot-Block Flash Memory или Parametric Flash Memory), для создания аналогов жестких дисков — с симметричными (Flash-File Memory).

С ростом информационной емкости микросхем увеличиваются и размеры блоков. В микросхемах раннего периода блоки были небольшими, их емкость составляла 64—256 байт. Для современных микросхем с симметричными блоками характерным размером блока является 64 Кбайта при числе блоков 64. В несимметричных блочных структурах наряду с большими (главными) блоками для хранения редко изменяемых программ традиционно выделялись как Boot-блоки со специальными мерами защиты от непредусмотренных операций стирания/записи, так и небольшие блоки параметров для хранения часто изменяемой информации. В последнее время получили распространение несимметричные структуры, в которых лишь

один из 64 блоков исходно симметричной структуры разбивается на 8 блоков специализированного характера.

Номенклатура микросхем флэш-памяти недавно пополнилась структурами типа Concurrent Flash Memory, допускающими выполнение операций чтения во время проведения в других блоках операций стирания/записи. Разработка таких структур явилась ответом на запросы ЦОС (цифровой обработки сигналов).

Флэш-память с несимметричной блочной структурой

Схемам этого типа присуще блочное стирание данных и несимметричная блочная архитектура. Блоки специализированы и имеют разные размеры. Среди них имеется так называемый Boot-блок, содержимое которого аппаратно и программно защищено от случайного стирания. В Boot-блоке хранится программное обеспечение базовой системы ввода/вывода микропроцессорной системы BIOS (Basic Input/Output System), необходимое для правильной эксплуатации и инициализации системы.

В составе блоков имеются также блоки параметров (БП), хранящие относительно часто меняемые параметры системы (коды идентификаторов, диагностические программы и т. п.) и главные блоки (ГБ), хранящие реализуемые программы и т. п. Блоки параметров и главные блоки не имеют средств защиты от непредусмотренной записи, свойственных Boot-блокам. На рис. 4.24 показана упрощенная структура и внешняя организация микросхемы флэш-памяти с несимметричными блоками.

Схема работает от одного источника питания 5 В как в режиме чтения, так и в режиме репрограммирования. Для выполнения сложных операций стирания/записи имеются внутренние схемы управления и таймер. Boot-блок имеет емкость 16 Кбайт, два блока параметров имеют емкости по 8 Кбайт, три главных блока — по 64 Кбайт и один 32 Кбайт. Для режимов стирания предусмотрены разные варианты — как поблочного стирания, так и стирания всего кристалла (за исключением Boot-блока или без такого исключения). Информационная емкость показанной для примера схемы рис. 4.24 равна 2 Мбайт при организации 256K×8.

Адресные входы $A_{17}A_0$ позволяют адресовать 256K ячеек памяти, линии входов/выходов $I/O_7I/O_0$ передают в микросхему или из нее информацию байтового формата, L-активные сигналы $\overline{CE}$ (Chip Enable), $\overline{OE}$ (Output Enable), $\overline{WE}$ (Write Enable) и $\overline{RESET}$ (сброса) участвуют в управлении работой схемы.

Чтение данных обеспечивается соответствующими сочетаниями сигналов $\overline{CE}$, $\overline{OE}$, $\overline{WE}$, как и при работе микросхем EPROM, EEPROM. Репро-

граммирование производится последовательностью операции полного стирания данных в блоке памяти, содержащем изменяемые байты, и затем программированием блока в режиме байт за байтом. Конец цикла программирования байта может индицироваться проверкой состояния одной из линий шины данных. Если при этом выявляется конец цикла, то можно начинать новый доступ для операций чтения или программирования.

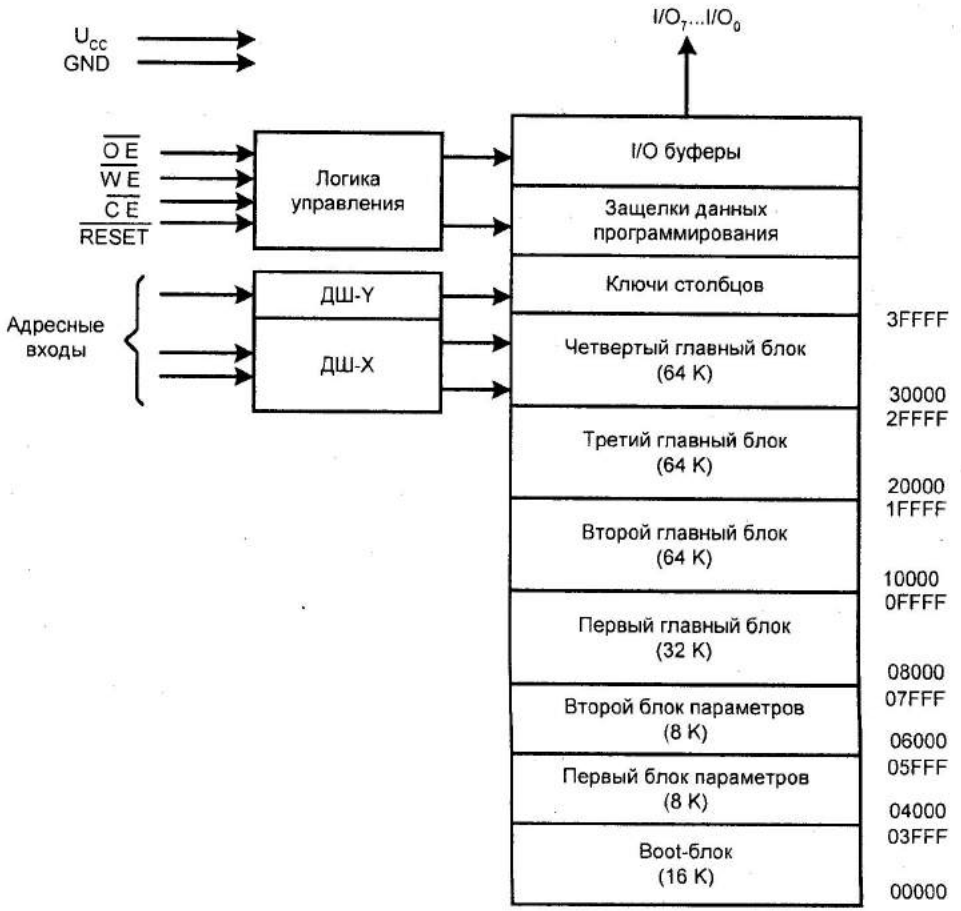


Рис. 4.24. Упрощенная структура микросхемы флэш-памяти с несимметричными блоками

Для инициализации операций стирания в микросхему вводится последовательность байтов (команда стирания), управляющая внутренними процессами приведения всех элементов памяти в стираемой области в состояние

логической единицы. После введения команды операция управляется внутренними средствами кристалла без использования внешних синхросигналов. При программировании состояние определенных элементов памяти будет изменено на нулевое. Данные в Boot-блоке могут быть защищены от изменений, что задается соответствующей командной последовательностью. При этом в микросхему вводится пароль пользователя.

Операция чтения задается сочетанием сигналов $\overline{OE} = \overline{CE} = L$, $\overline{WE} = H$. При этом данные, хранимые в адресованной ячейке, появляются на выходах микросхемы. Если $\overline{CE} = H$ или $\overline{OE} = H$, то выходы переводятся в третье состояние.

При включении питания схема окажется в режимах чтения или ожидания (Standby) в зависимости от состояния управляющих сигналов. Для перехода к другим режимам необходимо вводить командные последовательности, запись которых осуществляется нулевым импульсным сигналом, подаваемым на вход $\overline{WE}$ или $\overline{CE}$ при низком уровне $\overline{CE}$ или $\overline{WE}$ соответственно и $\overline{OE} = H$.

Вход $\overline{RESET}$ используется для упрощения некоторых системных операций. При высоком уровне напряжения на этом входе схема находится в обычном рабочем режиме. Переход сигнала $\overline{RESET}$ к низкому уровню останавливает текущие операции и переводит выходы схемы в третье состояние. Если такой переход произошел при выполнении стирания или программирования, то они не могут быть впоследствии продолжены и должны быть повторены с самого начала после возвращения сигнала сброса к высокому уровню. Возврат сигнала сброса к высокому уровню приводит схему в режим чтения или ожидания в зависимости от состояния управляющих сигналов. При подаче на вход $\overline{RESET}$ напряжения 12 В Boot-блок может быть репрограммирован даже если ранее был введен запрет этой операции.

Для операции стирания информации могут быть предусмотрены два варианта: стирание всего кристалла (Chip Erase) или стирание блока (Sector Erase). В первом варианте стирается содержимое главных блоков и блоков параметров, а воздействие на содержимое Boot-блока зависит от наличия или отсутствия запрета на его стирание. Если запрет был введен, то содержимое Boot-блока сохранится, если нет — будет стерто. Определить наличие или отсутствие запрета на репрограммирование Boot-блока можно программным способом — когда схема находится в режиме идентификации фирмы-производителя, чтение по соответствующему адресу отвечает на этот вопрос. После завершения операции стирания схема вернется к режиму чтения. Во время стирания кристалла все команды игнорируются. Операция стирания блока также выполняется под управлением внутреннего контроллера микросхемы после введения командной последовательности байтов, но применя-

ется только к блоку, адрес которого фиксируется в последнем цикле командной последовательности.

Побайтное программирование выполняется с использованием внутреннего регистра команд. Программирующие импульсы автоматически генерируются в самой схеме. Адреса и данные защелкиваются соответствующими перепадами сигналов $\overline{WE}$ и $\overline{CE}$. Программирование завершается по истечении определенного времени, для индикации завершения этого процесса может быть использована проверка состояния одного из разрядов шины I/O (сигнала $\overline{DATA}$). Например, если до завершения операции программирования этот разряд находится в состоянии, инвертированном относительно его значения во входном байте, а после окончания программирования принимает истинное значение, или же до завершения программирования попытка чтения этого разряда создает колебательный характер сигнала (переходы из 0 в 1 и обратно), а после окончания операции сигнал становится стабильным.

Микросхемы флэш-памяти с Boot-блоками предназначены для работы с разными микропроцессорами и для соответствия им имеют два варианта расположения этих блоков в адресном пространстве: вверху и внизу, что отображается в маркировке ИС буквами T (Top) или B (Bottom). На рис. 4.24 приведена схема с нижним расположением Boot-блока.

Флэш-память с симметричной блочной структурой (файловая)

С давних пор хранение больших объемов данных возлагается в микроЭВМ на хорошо отработанные и сравнительно недорогие внешние ЗУ на магнитных дисках. Во многих компьютерах система памяти была организована как сочетание жесткого магнитного диска (винчестера) с динамическим полупроводниковым ОЗУ.

Имея немалые достоинства, дисковые ЗУ как электромеханические устройства не свободны от ряда недостатков (чувствительность к ударам, вибрациям и загрязнениям, ограниченное быстродействие, значительное потребление мощности). Эти недостатки особенно сказываются в портативных устройствах с автономным (батарейным) питанием. Достаточно отметить, что дисководы потребляют в лучшем случае мощность около 3 Вт, что в системах с напряжениями питания 3,3—5 В означает потребление токов 0,6—0,9 А, быстро истощающих батарейки.

Файловая флэш-память (ФФП) ориентирована на замену твердых дисков, она в сотни раз сокращает потребляемую мощность, в той же мере увеличивает механическую прочность и надежность ЗУ, уменьшает их размеры и вес, на несколько порядков повышает быстродействие при чтении данных, сохраняя при этом программную совместимость со средствами управления

памятью. Вместе с тем за дисковой памятью остаются преимущества по информационной емкости и стоимости.

Использование ФФП в портативных компьютерах — один из важнейших факторов, способствующих их развитию. При этом традиционное сочетание "жесткий диск — динамическое ОЗУ" может заменяться сочетанием "флэш-память — статическое ОЗУ". Команды программы, хранимые в ФФП, в этом случае читаются процессором непосредственно из нее, результаты тоже записываются прямо в ФФП, а операции с интенсивными вычислениями, требующие быстрейшего доступа к памяти и записи данных с байтовой разрешающей способностью, выполняются с использованием быстродействующей статической памяти.

Накопитель ФФП делится на блоки, которые служат аналогами секторов магнитных дисков, присущих операционной системе MS-DOS. Разработаны программные средства, которые обеспечивают обмен с флэш-блоками, подобно тому как операционная система MS-DOS обеспечивает обмен с секторами диска.

Блоки ФФП идентичны и имеют одинаковую информационную емкость (симметричная блочная архитектура). Так как в ФФП операции записи производятся значительно чаще, чем в других разновидностях флэш-памяти, этим операциям уделяется большое внимание — вводятся страничные буферы, позволяющие с высокой скоростью накапливать некоторый объем данных, подлежащих записи, для их последующей передачи в накопитель с меньшей скоростью.

Внешняя организация ФФП с информационной емкостью $2^n \times m$ показана на рис. 4.25.

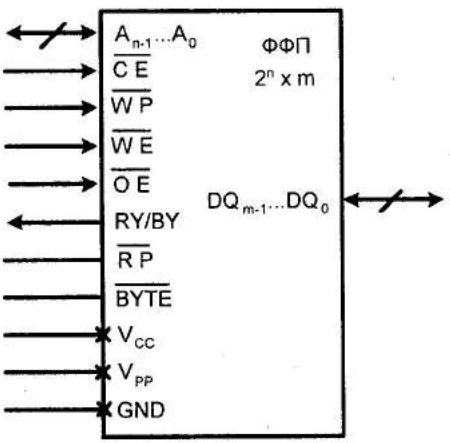


Рис. 4.25. Внешняя организация файловой флэш-памяти

Накопитель схемы разбит на некоторое число блоков, размеры которых у современных схем стали довольно большими (64 Кбайт, 128 Кбайт).

Выводы и сигналы, показанные на рис. 4.25, имеют следующий смысл. Линии старших разрядов шины адреса выбирают один из блоков, линии младших разрядов выбирают байт или слово в пределах блока (блок с емкостью N Кбайт содержит N/2 Кслов). Например, для микросхемы с информационной емкостью 128 Мбит и блоками по 128 Кбайт для указания блока будут использованы 7 старших разрядов адреса, а для указания байта — 17 младших. При словарной организации памяти разрядность адреса сокращается на единицу. Для вариантов с перестраиваемой разрядностью линия A_0 (младший бит адреса) при словарной организации памяти отключается, а при байтовой организации определяет старший и младший байты. При адресации блочных данных от процессора поступает начальный адрес блока, который запоминается в очереди адресов. Текущий адрес ячейки памяти формируется адресным счетчиком.

В 16-разрядной двунаправленной шине данных DQ_{15-0} линии DQ_{7-0} предназначены для ввода и вывода младшего байта данных, а также передачи в цикле записи команды так называемому *командному интерфейсу пользователя* CUI (Command User Interface), т. е. внутреннему автомату управления процессами стирания/записи. В циклах чтения линии DQ_{7-0} служат для вывода данных из буфера, регистров идентификатора или состояния. Линии DQ_{15-8} предназначены для передачи старшего байта при словарной организации памяти. По ним выводятся данные накопителя, буфера или идентификатора в соответствующем режиме чтения. Если кристалл не выбран или вывод запрещен, линии шины данных переходят в третье состояние.

Линия $\overline{CE}$ — вход разрешения кристалла, при высоком уровне сигнала кристалл не выбран, и потребление мощности снижается до уровня состояния покоя (Standby) после завершения текущей операции записи или стирания.

Сигнал $\overline{OE}$ открывает выходные буферы при низком уровне и переводит их в третье состояние при высоком.

Сигнал $\overline{WE}$ управляет доступом к командному интерфейсу пользователя CUI, страничным буферам, регистрам очереди данных и защелкам очереди адресов.

Сигнал $\overline{RP}$ (Reset/Power-Down) при низком уровне вводит схему в состояние глубокой экономии мощности, отключая все схемы, потребляющие статическую мощность. При выходе из этого состояния время восстановления достаточно велико (сотни наносекунд). При переходе к низкому уровню операции автомата записи прекращаются, схема сбрасывается.

Сигнал RY/BY (Ready/Busy) индицирует состояние внутреннего автомата записи. Низкий уровень означает занятость, высокий означает или готовность к новым операциям, или приостановление стирания, или состояние глубокой экономии мощности в зависимости от выполняемой операции.

Сигнал $\overline{WP}$ (Write Protect) имеет следующий смысл. Каждый блок содержит бит запрещения записи (Lock-bit). Низкий уровень $\overline{WP}$ разрешает защиту, т. е. запись или стирание в блоке могут выполняться только при Lock-bit = 0. При высоком уровне $\overline{WP}$ в блоках могут выполняться операции записи и стирания независимо от состояния блокирующих битов.

Сигнал $\overline{BYTE}$ низким уровнем вводит схему в байтовый режим, высоким — в словарный и выключает буфер линии A_0.

Микросхемы ФФП в настоящее время имеют информационную емкость от 4 Мбит до 4 Гбит при временах доступа при чтении 70—150 нс, напряжениях питания от 5 В до 1,8 В. Они имеют байтовую или программируемую разрядность 8/16 бит. Время программирования одного байта составляет около 10 мкс. Длительности процессов стирания и записи блоков в целом лежат в секундном диапазоне.

Флэш-память с многоуровневым хранением заряда в плавающих затворах (типа StrataFlash и др.)

В 1997 г. компания Intel представила новый вид флэш-памяти (StrataFlash), в которой в одном запоминающем элементе стали храниться не один, а два бита. Это обеспечивается тем, что в плавающем затворе транзистора фиксируется не только наличие или отсутствие заряда, но и определяется его величина, которая может иметь несколько значений. Различая четыре уровня, можно хранить в одном элементе два бита.

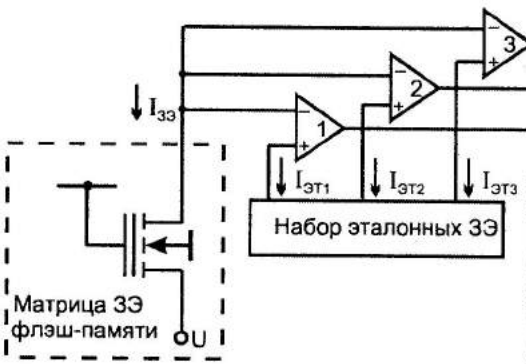
Ранее для увеличения емкости ЗУ шли путем уменьшения размеров схемных элементов и других усовершенствований технологических процессов литографии. StrataFlash ознаменовала другой подход к этой проблеме. Хранения двух битов добились практически в тех же запоминающих элементах, которые ранее хранили один бит, преодолев трудности ужесточения допусков на величины зарядов, вводимых в плавающий затвор. Во второй половине 1990-х гг. появились коммерческие образцы, в которых от емкости 32 Мбит разработанных к тому времени микросхем флэш-памяти перешли к емкости 64 Мбит в микросхемах StrataFlash без заметных изменений площади кристалла. Технология хранения в плавающих затворах нескольких уровней заряда была воспринята рядом фирм, выпустивших *"память на многоуровневых ячейках"*. Общим названием запоминающих элементов их микросхем можно считать MLC (Multilevel Cells).

Запоминающие элементы MLC памяти StrataFlash программируются введением в плавающий затвор одного из четырех возможных значений количеств заряда, каждое из которых соответствует паре двоичных цифр 11, 10, 01, 00. В зависимости от заряда запоминающий транзистор приобретает одно из четырех пороговых напряжений. При считывании информации к за-

твору транзистора прикладывают напряжение считывания. Ток запоминающего транзистора зависит от порогового напряжения. Определяя ток, можно выявить состояние плавающего затвора.

На рис. 4.26 показаны распределение пороговых напряжений в четырехуровневом запоминающем элементе (*а*) и схема чтения состояния запоминающего транзистора (*б*).

а



Заряд в ПЗ	Выходы усилителей чтения			Выходные данные	
	1	2	3		
Q11	1	1	1	1	1
Q10	0	1	1	1	0
Q01	0	0	1	0	1
Q00	0	0	0	0	0

б

Рис. 4.26. Графики распределения пороговых напряжений в четырехуровневом запоминающем элементе (*а*) и схема чтения данных из этого элемента (*б*)

Флэш-память с зеркальным битом

В 2001 г. фирма AMD предложила архитектуру флэш-памяти, названную схемой с зеркальным битом (Mirror Bit). В этих схемах, как и в ячейках с четырьмя уровнями хранимого заряда, можно удвоить емкость памяти по сравнению со стандартными вариантами, но достигается это иным путем. В одном запоминающем элементе хранятся два бита данных в виде индивидуальных зарядов, размещенных в разных местах подзатворного слоя одного

и того же транзистора. Иными словами, используется пространственное разделение двух зарядов, каждый из которых отображает свой бит хранимой информации (рис. 4.27).

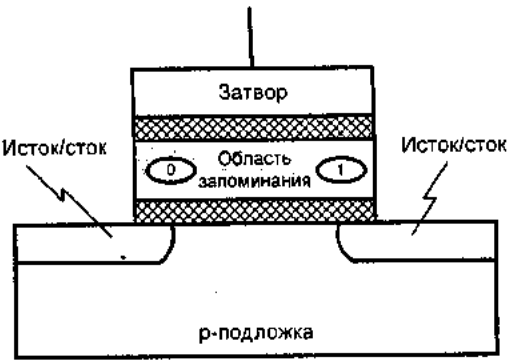


Рис. 4.27. Структура запоминающего элемента с зеркальным битом

Транзистор запоминающего элемента в схемах с зеркальным битом отличается от обычного полной идентичностью областей истока и стока. В обычных транзисторах исток и сток легируются различно, здесь они не отличаются друг от друга. Сама запоминающая область модифицирована так, чтобы группы электронов могли независимо храниться в обеих ее сторонах. При этом транзистор проявляет себя как два элемента памяти.

Поскольку многоуровневое хранение заряда в ячейке не используется, то проблема жестких допусков на величину вводимого заряда снимается, а это позволяет проводить процессы чтения, стирания и записи интенсивно с сохранением высокой скорости. По сравнению с вариантом многоуровневого хранения заряда повышается и надежность сохранения данных, поскольку при многоуровневом хранении допустимые потери заряда, естественно, значительно меньше, чем при двухуровневом. Указанные факторы следует считать достоинствами варианта с зеркальным битом. В то же время разработчики многоуровневых ячеек MLC прогнозируют возможность хранения в одном элементе более чем двух бит, говоря о достижимости хранения 4 бит, чего конструкция с зеркальным битом обеспечить не может.

§ 4.6. Последовательные ЗУ типов EEPROM и Flash

Для большинства проектируемых устройств потребности в энергонезависимой памяти удовлетворяются традиционными микросхемами EEPROM и

Flash. Наряду с этим существует и немало проектов, для которых требования к параметрам памяти достаточно скромны и не требуется слишком высокое быстродействие. В таких проектах целесообразно заменить традиционные микросхемы EEPROM или Flash-памяти значительно более простыми и дешевыми последовательными, получающими в последнее время все более широкое применение.

В микросхемах последовательной памяти исключаются многоразрядные шины адресов и данных. Адреса и данные вводятся в схемы по одной линии (через один контакт) последовательно, разряд за разрядом. Выходные слова также выводятся через один контакт в последовательной форме. *Время передачи адресов и данных при этом резко увеличивается, но столь же резко упрощается корпус микросхемы, растет ее компактность и снижается стоимость.* Корпус микросхемы последовательной памяти может иметь *всего 5 выводов*: для напряжения питания, "земли", входной информации, выходной информации и сигнала разрешения работы.

Современные последовательные ЗУ по информационной емкости перекрывают диапазон от сотен бит до мегабит и приобретают все более разнообразные свойства. Они могут обеспечивать работу с 8- или 16-разрядными словами, иногда и с возможностью перестройки организации памяти. Допустимое число циклов репрограммирования составляет для них 10^5—10^6, гарантированное время сохранения данных 10—100 лет. Микросхемы имеют весьма малые габариты, например, память с информационной емкостью 1 Кбит выпущена в корпусе SOT-23 с пятью выводами площадью всего 5 мм^2, ожидаются микросхемы в корпусах с площадью всего 2 мм^2. Очень мала и потребляемая мощность. Так, при напряжении питания 1,8 В токи потребления в активном режиме составляют приблизительно 1 мА, а в режиме покоя (Standby) всего несколько микроампер. Малые габариты и малая потребляемая мощность — важнейшие качества микросхем, предназначенных для портативной аппаратуры с батарейным питанием.

Передача информации в микросхемы последовательной памяти и из них обычно осуществлялась по шинам уже имевшегося типа (Microwire, I^2C, SPI). В последнее время получила применение модифицированная шина SPI (Serial Peripheral Interconnect) с частотой передач до 10 МГц, но это не устранило потребности в более скоростных интерфейсах, поскольку одновременно с ростом информационной емкости кристаллов памяти растут и требования сокращения времени доступа к данным. При той же частоте передачи слов темп передач битов должен быть пропорционален длине слов, поскольку при переходе к параллельным кодам за то же время в сдвигающие регистры преобразователей последовательного кода в параллельный и наоборот требуется вдвигать или выдвигать большее число битов. При тактовой частоте преобразователей 10 МГц для преобразователей потока битов в 8-разрядные слова требуется время, приблизительно равное 0,8 мкс, а для преобразования 16-разрядных слов необходимо время приблизительно в

1,6 мкс. Работе с памятью большей информационной емкости свойственно удлинение обрабатываемых слов, а это требует и повышения скоростей последовательных интерфейсов памяти.

Наряду с обычными базовыми интерфейсами для последовательной памяти реализованы и такие, в которых пакет байтов выходных данных формируется слитно после загрузки в память стартового адреса. Такой режим исключает задержки на загрузку нового адреса для каждого слова данных. Так, например, в микросхемах семейства AT93CXX фирмы Atmel после получения команды чтения и адреса выходной контакт выходит из третьего состояния, на выходе формируется нулевой бит, после которого выдвигается адресованное слово. При сохранении активного сигнала разрешения работы памяти адрес автоматически инкрементируется и выдвигается следующее слово данных. Этот процесс может продолжаться вплоть до достижения конца адресного пространства, после чего адрес становится нулевым и процесс чтения продолжается.

§ 4.7. Использование программируемых ЗУ для решения задач обработки информации

В предыдущих параграфах запоминающие устройства рассматривались с точки зрения задач хранения информации. Однако программируемая память является также и универсальным средством решения самых разных задач обработки информации. Применимость программируемой памяти в указанной области определяется возможностью представления решения задачи в табличной форме. Эта форма решения возможна для задач самого разного характера.

Для уяснения возможностей ППЗУ в области решения задач обработки информации целесообразно рассмотреть основные соотношения, связанные с воспроизведением логических и числовых функций.

Реализация логических (переключательных) функций

ППЗУ с организацией $2^m \times 1$ принимает m-разрядный адрес и выдает одноразрядный результат (0 или 1). Этот способ функционирования непосредственно воспроизводит переключательную функцию m переменных, т. к. для каждого входного набора можно при программировании ЗУ назначить необходимую выходную переменную. Например, ППЗУ с организацией 1024×1 может быть использовано для воспроизведения переключательной функции 10 аргументов.

ППЗУ с организацией $2^m \times n$ по поступающему на его вход m-разрядному адресу выдает n-разрядное выходное слово, хранящееся в ячейке с данным адресом. Такое ЗУ воспроизводит *систему переключательных функций*, число которых равно разрядности выходного слова. Действительно, на каждом выходе может быть воспроизведена любая переключательная функция m-аргументов, а совокупность выходов даст n различных функций.

В ППЗУ функции реализуются в совершенной дизъюнктивной нормальной форме, для каждой возможной конъюнкции имеется свое оборудование (выходная линия дешифратора адреса) и, следовательно, она может быть введена в выходную функцию. *Какой-либо минимизации функций при подготовке задачи к решению на основе ППЗУ не требуется*, более того, если функции уже минимизированы, то для удобства подготовки данных для программирования ЗУ их придется развернуть до самой громоздкой формы (СДНФ). Это делается либо заполнением карты Карно и последующей записью функции без какого-либо объединения единиц, либо введением в каждую конъюнкцию недостающих переменных x_i путем домножения конъюнкции на равные единице выражения $x_i \vee \bar{x}_i$ с последующим раскрытием скобок (x_i — вводимая переменная). Пример приведения функции в СДНФ:

$$F = x_1 \vee x_2 x_3 = x_1(x_2 \vee \bar{x}_2)(x_3 \vee \bar{x}_3) \vee (x_1 \vee \bar{x}_1)x_2 x_3 = x_1 x_2 x_3 \vee x_1 \bar{x}_2 x_3 \vee x_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 \bar{x}_3 \vee \bar{x}_1 x_2 x_3.$$

Для воспроизведения этой функции по пяти конъюнкциям-адресам в ППЗУ следует записать единицы, по остальным адресам — нули.

Реализация функции в СДНФ определяет большие затраты элементов памяти, однако цена элемента памяти значительно ниже цены логического элемента, поэтому даже при избыточности числа элементов памяти в несколько раз (в сравнении с числом логических элементов, необходимых для воспроизведения функции традиционным методом) реализация на ППЗУ может оказаться выгодной.

Особенности ППЗУ указывают на целесообразность его использования для реализации в первую очередь функций, не поддающихся существенной минимизации.

При этом время выполнения операции — это время считывания данных из ЗУ.

Реализация конечных автоматов

В канонической схеме автомата ППЗУ может заменить комбинационную цепь, поскольку оно способно воспроизводить переключательные функции. Поэтому структура автомата без потери общности может быть представлена также в виде, приведенном на рис. 4.28.

Начальная установка регистра задает исходное состояние элементов памяти (автомата). По этому состоянию и входным сигналам из памяти считывается

код нового состояния и функции выхода. В следующем такте эти процессы повторяются. В каждом очередном такте автомат переходит в новое состояние и вырабатывает выходные функции согласно таблицам переходов и выходов.

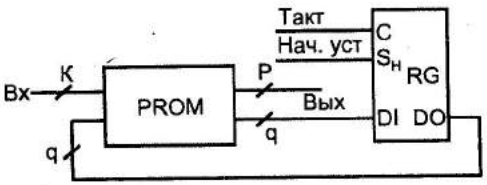


Рис. 4.28. Структура автомата, реализованного на основе микросхем памяти

Емкость ППЗУ определяется объемом таблиц, задающих функционирование автомата. Сведя таблицы переходов и выходов в одну, получим общее число входов $m = k + q$ и число выходов $n = p + q$, следовательно, для реализации автомата требуется емкость памяти $M = 2^{k+q}(p + q)$.

Воспроизведение арифметических операций и функциональных зависимостей

Арифметические операции и числовые (не логические) функции часто встречаются в качестве задач, решаемых цифровыми устройствами. Функции задаются аналитически или таблично.

Для функций одного аргумента объем памяти таблиц легко вычислить, зная разрядности аргумента и функции. При задании аргумента m-разрядным кодом число точек, в которых задана функция, составит 2^m (рис. 4.29, *а*). Если разрядность кода, представляющего функцию, равна n, то, очевидно, емкость памяти в битах будет равна $n2^m$.

С ростом числа аргументов объем памяти для запоминания таблиц функций быстро растет. Для функции двух аргументов разрядностей m число точек, в которых задана функция, определится как произведение чисел точек по каждой из координат и составит 2^{2m} (рис. 4.29, *б*). Объем памяти таблицы в этом случае составит $M = n2^{2m}$.

Для функций ℓ аргументов $M = n2^{\ell m}$.

Итак, с ростом разрядности слов и числа аргументов функций объем памяти таблиц быстро растет и чисто табличный метод решения задачи становится неприемлемым. В этих случаях часто очень полезны *таблично-алгоритмические методы*, в рамках которых можно существенно снизить объем таблиц, введя небольшое число простых операций над данными.

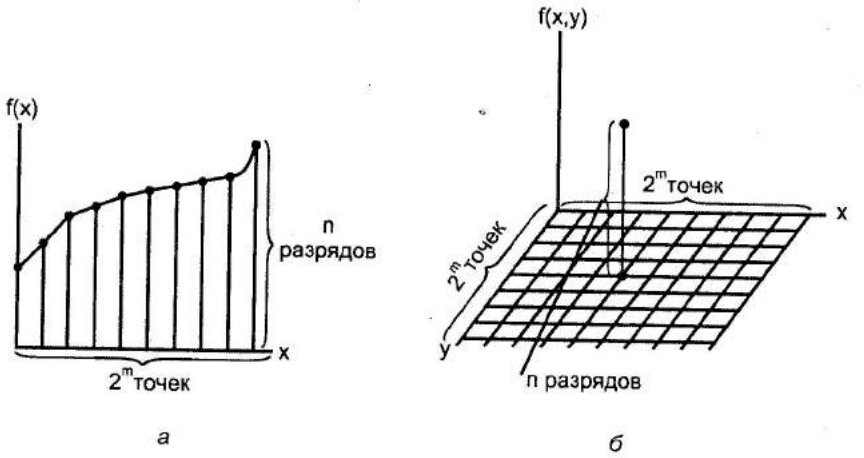


Рис. 4.29. К определению емкости памяти при воспроизведении табличным методом числовых функций одного (*а*) и двух (*б*) аргументов

Для произвольных функций f(x) простейший таблично-алгоритмический метод — *кусочно-линейная аппроксимация*, когда запоминаются только узловые значения функции, а в промежутках между узлами функция вычисляется в предположении, что на промежутках она изменяется линейно. Число узлов назначается по соображениям точности линейной аппроксимации функции на участках. Кусочно-линейной аппроксимации с постоянным шагом соответствуют следующие представления аргумента и функции:

$$x = x_i + \Delta x,\ f(x) = f(x_i) + \Delta f(x_i)\Delta x h^{-1},$$

где x_i — координата i-й узловой точки; Δx — разность значений x и координаты ближайшей слева узловой точки; $\Delta f(x_i)$ — приращение функции на участке от x_i до x_{i+1}; h — шаг аппроксимации (для удобства реализации цифровыми методами шаг берут равным целой степени числа 2).

Согласно приведенным формулам, структура функционального преобразователя с кусочно-линейной аппроксимацией имеет вид, приведенный на рис. 4.30.

Емкость памяти при переходе от табличного метода к таблично-алгоритмическому, как правило, существенно сокращается, а быстродействие остается довольно высоким.

Для функций двух переменных можно применить кусочно-плоскостные аппроксиматоры.

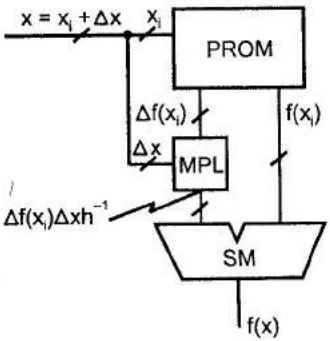


Рис. 4.30. Структура преобразователя с кусочно-линейной аппроксимацией функций

§ 4.8. Статические запоминающие устройства

Область применения дорогостоящих статических ОЗУ — память с высшим быстродействием. Они используются в кэш-памяти, которая при сравнительно малой емкости должна иметь максимальное быстродействие, при построении буферов FIFO и LIFO, а также для реализации небольших по емкости блоков памяти данных в микроконтроллерах.

Статические ОЗУ (SRAM) чаще всего имеют структуру 2DM (в том числе в составе блочных ЗУ), часть их при небольшой информационной емкости строится по структуре 2D, имеются и микросхемы со структурой 3D.

Запоминающие элементы статических ОЗУ — триггеры с цепями установки и сброса. Поэтому статические ОЗУ называют также *триггерными*. Триггеры можно реализовать по любой схемотехнологии (ТТЛ(Ш), $И^2Л$, ЭСЛ, n-МОП, КМОП, AsGa и др.), соответственно которым возможны разнообразные схемы ЗУ. На разных этапах развития на ведущие роли выдвигались различные схемотехнические типы ЗУ. Сейчас доминирующей стала КМОП-схемотехника, поскольку при субмикронной технологии КМОП-схемы, сохраняя свои традиционные достоинства, приобретают также высокое быстродействие и другие возможности, которые ранее были не достижимы.

В типичной номенклатуре SRAM зарубежных фирм имеются микросхемы с информационной емкостью в диапазоне от 64 Кбит до 8 Мбит и временами доступа 10—20 нс.

Среди отечественных микросхем хорошо развиты серии К537 технологии КМОП и К132 технологии n-МОП (как и другие отечественные микросхемы памяти, эти серии являются старыми разработками с соответствующим уровнем технических параметров).

Запоминающие элементы статических ЗУ

Запоминающий элемент ЗУ на n-МОП транзисторах (рис. 4.31) представляет собой RS-триггер на транзисторах Т1 и Т2 с ключами выборки Т3 и Т4. При обращении к данному ЗЭ появляется высокий потенциал на шине выборки ШВi (через i, j соответственно обозначены номера строки и столбца, на пересечении которых расположен $ЗЭ_{ij}$). Этот потенциал открывает ключи выборки (транзисторы Т3, Т4) по всей строке, и выходы триггеров строки соединяются с шинами считывания-записи столбцов. Одна из шин столбцов связана с прямым выходом триггера (обозначена через D_j), другая — с инверсным ($\overline{D}_j$). Через шины столбцов можно считывать состояние триггера (штриховыми линиями показано подключение дифференциального усилителя считывания). Через них же можно записывать данные в триггер, подавая низкий потенциал логического нуля на ту или иную шину.

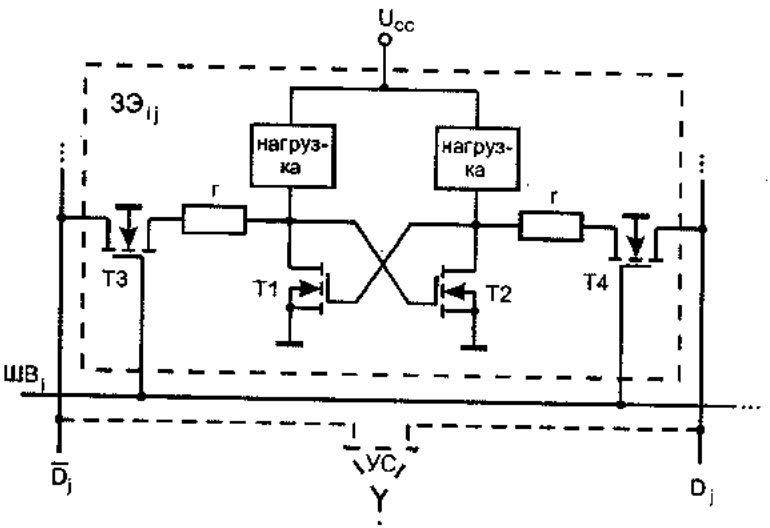


Рис. 4.31. Схема триггерного запоминающего элемента на n-МОП транзисторах

При подаче нуля на выход $\overline{D}_j$ снижается стоковое напряжение транзистора Т1, что запирает транзистор Т2 и повышает напряжение на его стоке. Это открывает транзистор Т1 и фиксирует созданный на его стоке низкий уровень даже после снятия сигнала записи. Триггер установлен в состояние логической единицы. Аналогичным образом нулевым сигналом по шине D_j можно установить триггер в нулевое состояние. При выборке строки со своими шинами столбцов соединяются все ее триггеры, но с выходными цепями считывания или входными цепями записи может связываться лишь

та или иная часть столбцов в соответствии с адресной информацией (речь идет о структуре 2DM). Резисторы r служат для уменьшения емкостных токов в моменты открывания ключевых транзисторов и реализуются как части диффузионных областей этих транзисторов.

В качестве нагрузки могут быть использованы n-МОП-транзистор со встроенным каналом и нулевым напряжением затвора или высокоомные нагрузочные резисторы, изготовленные из поликристаллического кремния и пространственно расположенные над областью транзисторов, что придает схеме как режим микротоков, так и высокую компактность. Режим микротоков нужен для ограничения потребляемой мощности, прежде всего для кристаллов высокого уровня интеграции, но создает и маломощность выходных сигналов. Эта особенность требует применения высокочувствительных усилителей считывания, что объясняет использование в статических ЗУ так называемых усилителей-регенераторов (ранее они были характерны только для динамических ЗУ).

Запоминающий элемент статических ОЗУ, выполненных по КМОП-технологии, показан на рис. 4.32 в обозначениях США. Эти элементы построены так же, как и элементы на n-МОП-транзисторах, и их работа не требует дополнительных пояснений. Следует лишь напомнить, что в КМОП-схемах статические токи практически отсутствуют, и мощность потребляется ими только в процессах переключения.

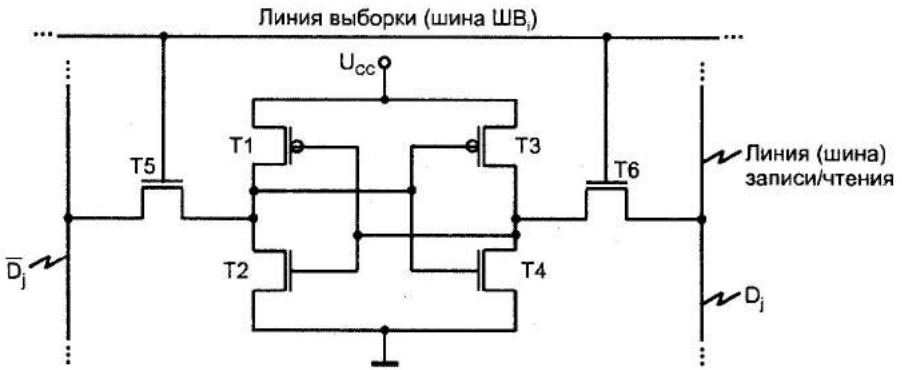


Рис. 4.32. Схемы триггерного запоминающего элемента в схемотехнике КМОП

Внешняя организация и временные диаграммы статических ЗУ

В номенклатуре статических ЗУ представлены ИС с одноразрядной и многоразрядной организацией. Внешняя организация статического ЗУ емкостью

64 Кбит (8K×8) показана на рис. 4.33. Состав и функциональное назначение сигналов адреса A12—0, выборки кристалла $\overline{CS}$, чтения/записи R/W соответствуют рассмотренным выше сигналам аналогичного типа. Входы и выходы ИС совмещены и обладают свойством двунаправленных передач. Имеется также вход $\overline{OE}$ разрешения по выходу, пассивное состояние которого ($\overline{OE}$ = H) переводит выходы в третье состояние. Работа ЗУ отображается таблицей (табл. 4.1).

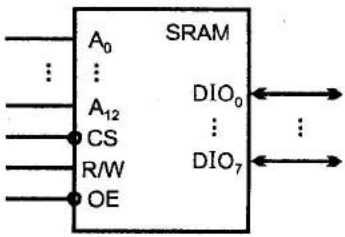


Рис. 4.33. Пример внешней организации статического ЗУ

Таблица 4.1

$\overline{CS}$	$\overline{OE}$	R/W	A	DIO	Режим
1	X	X	X	Z	Хранение
0	X	0	A	DI	Запись
0	0	1	A	DO	Чтение

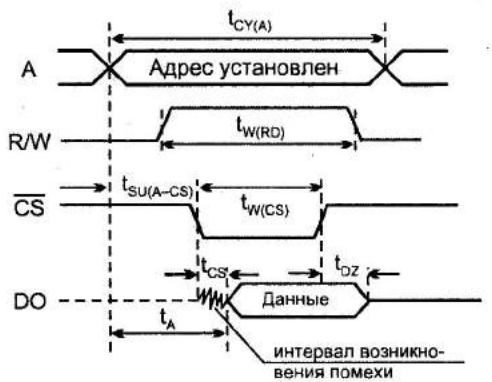
Функционирование ЗУ во времени регламентируется временными диаграммами, устанавливаемыми изготовителем. В основу кладутся определенные требования. Например, чтобы исключить возможность обращения к другой ячейке, рекомендуется подавать адрес раньше, чем другие сигналы, с опережением на время его декодирования. Адрес должен держаться в течение всего цикла обращения к памяти.

Затем следует подать сигналы, определяющие направление передачи данных и если предполагается запись, то записываемые данные, а также сигналы выборки кристалла и, при чтении, разрешения выхода. Среди этих сигналов будет и *стробирующий*, т. е. выделяющий временной интервал непосредственного выполнения действия. Таким сигналом для разных ЗУ может служить как сигнал R/W, так и сигнал $\overline{CS}$.

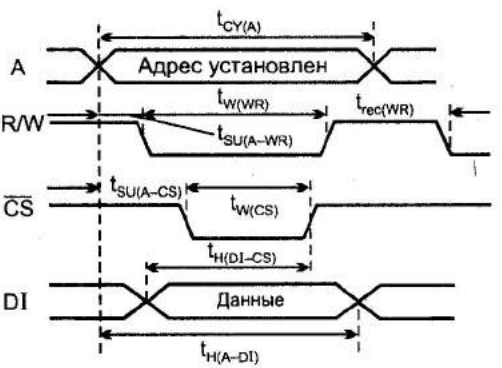
Стандартные (асинхронные) ЗУ могут быть нетактируемыми или тактируемыми. В тактируемых ЗУ к определенным сигналам (как правило, к сигналу

CS) предъявляется требование импульсного характера, согласно которому после активизации сигнала он обязательно должен вернуться к пассивному уровню и только после этого возможна его активизация в следующем цикле обращения к памяти. В нетактируемых ЗУ такое требование отсутствует и, например, разрешение работы может производиться постоянным уровнем $\overline{CS}$ = L на протяжении множества циклов обращения к памяти.

Пример временных диаграмм для процессов чтения и записи в асинхронном статическом ЗУ показан на рис. 4.34, *а*, *б*.



а



б

Рис. 4.34. Временные диаграммы процессов чтения (*а*) и записи (*б*) в асинхронном статическом ЗУ

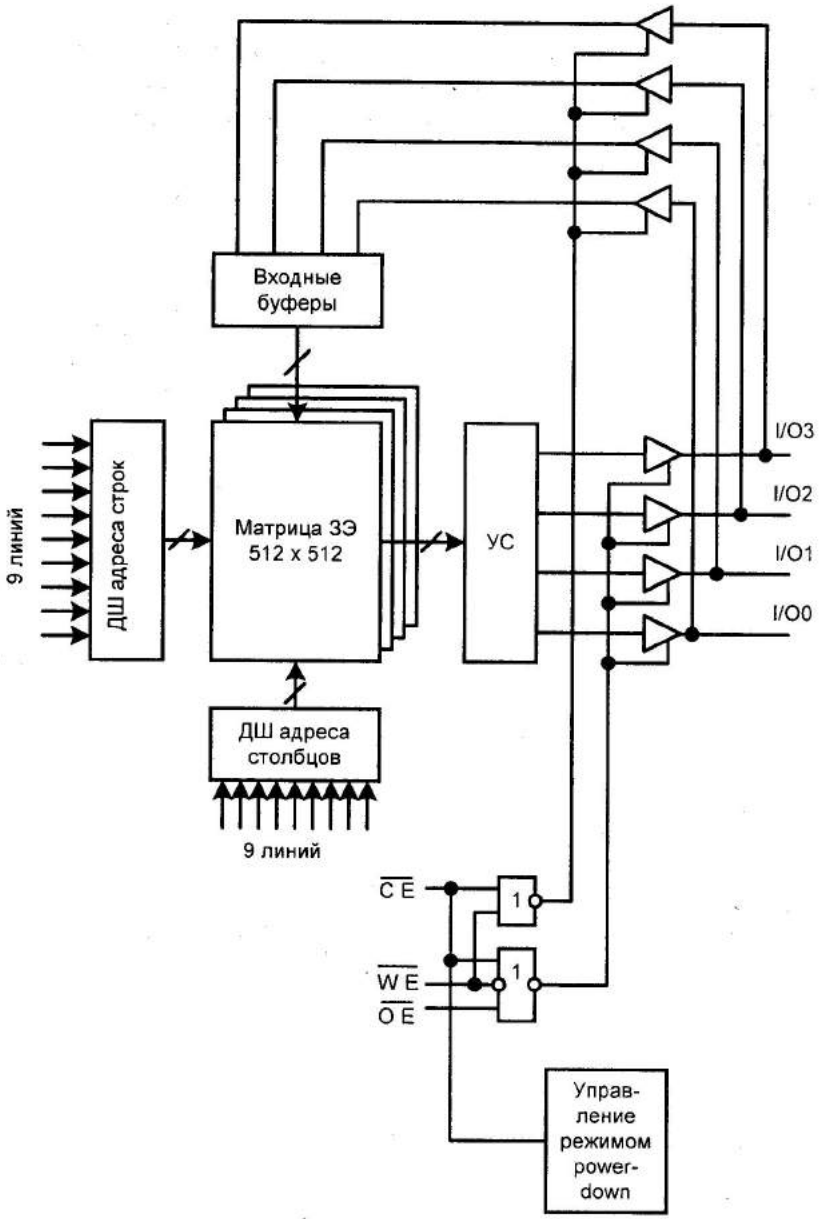


Рис. 4.35. Структура асинхронного статического ОЗУ

На них показаны времена выборки относительно адреса t_A и выбора t_{CS}, длительности импульсов t_W различных сигналов и цикла адреса $t_{CY(A)}$, за-

держка t_{DZ} перехода выхода из активного состояния в состояние "отключено", времена предустановки t_{SU} и удержания t_H с указанием сигналов, для которых они отсчитываются. Приведено время восстановления $t_{rec(WR)}$, отсчитываемое как необходимая пауза между повторениями активных интервалов сигнала WR.

Для правильного проектирования модулей памяти и использования в них конкретных микросхем необходимо также знать емкости их входов C_I, выходов C_O и предельно допустимую емкость нагрузки $C_{L\,max}$.

На рис. 4.35 показана микросхема одного из современных статических ОЗУ с информационной емкостью 1 Мбит, построенного по структуре 3D при организации 256K×4. Дополнительно к обычным для структуры 3D блокам, схема имеет блок управления режимом глубокого понижения мощности power-down, который снижает потребляемую мощность более чем на 65%, когда схема не выбрана, т. е. сигнал $\overline{CE}$ имеет высокий уровень. Запись данных производится при $\overline{CE} = \overline{WE} = L$. При этом данные с выводов I/O0I/O3 записываются по адресу, поданному на соответствующие входы микросхемы. Чтение происходит при $\overline{CE} = \overline{OE} = L$ и $\overline{WE} = H$. Это сочетание сигналов передает на выходы схемы содержимое ячейки, определяемой заданным адресом. При $\overline{CE} = H$ выводы I/O0I/O3 переходят в состояние "отключено".

Статические ЗУ повышенного быстродействия

Синхронные статические ЗУ. Стандартные (асинхронные) статические ЗУ с типичными временами доступа 10—20 нс позволяют организовать пакетный обмен данными без циклов ожидания процессора до частот 33 МГц. Для более высоких частот потребовались синхронные SRAM. Вначале синхронные структуры применялись для динамических ЗУ, где дали значительный положительный эффект, а вслед за тем нашли применение и в структурах статических ЗУ.

Синхронные статические ЗУ имеют модифицированный интерфейс для *согласования работы ЗУ с синхросистемой процессора* и могут работать со *сквозной* или *конвейерной* передачей данных. Согласование временных диаграмм интерфейса (процессора) и ЗУ позволяет *исключить непроизводительные потери времени*, возможные в асинхронных ЗУ. В синхронных ЗУ моменты изменения всех сигналов точно известны, они фиксируются фронтами синхросигнала CLK, вырабатываемого процессором. В частности, время приема данных от памяти при чтении точно известно, и в соответствующем синхротакте процессор может без каких-либо потерь времени выполнить операцию приема данных.

Сквозная передача данных соответствует обычному режиму работы памяти, когда между обращениями к ней выдерживается временной интервал, в течение которого формируется результат (например, появляются читаемые данные). В конвейеризованных устройствах положение иное: *не ожидая выработки конечного результата для данного цикла можно начинать следующий.* Сущность конвейеризации была рассмотрена в *§ 4.3.* При конвейеризации SRAM между усилителями чтения и выходными буферами ЗУ добавляются регистры, позволяющие увеличить частоту тактирования системы и расширить полосу пропускания памяти (следует напомнить, что полосой пропускания называют произведение частоты передач данных на разрядность памяти). Повышению производительности систем с синхронными ЗУ способствует и наличие в синхронных микросхемах памяти регистров-защелок на входах и выходах, что позволяет процессору после выдачи для ЗУ адреса и сигналов управления сразу же переходить к другим операциям.

В структурах синхронных ЗУ отображается их ориентация на обмен данными, свойственный кэш-памяти (пакетный обмен).

Синхронные ЗУ с пакетным обменом (Synchronous Burst SRAM) имеют внутренний счетчик адресов и в дополнение к обычным для асинхронных ЗУ управляющим сигналам следующие сигналы:

- синхросигнал процессора (системы) CLK;
- стробы, которыми процессор (или контроллер) записывают начальный адрес пакета во внутренний регистр микросхемы и инициируют очередной цикл обращения к памяти (цикл может быть не только пакетным, но и одиночным);
- сигнал для перехода к следующему адресу в пакете. Следующий адрес может идти последовательно относительно предыдущего или же обеспечивать чередование банков микросхемы.

Синхронные конвейеризованные ЗУ с пакетным обменом (Pipelined Burst SRAM). В эту структуру введен дополнительный регистр в тракт передачи данных. При этом, как и свойственно конвейеру, для первой передачи время увеличивается на дополнительный такт, но для последующих передач темп ускоряется. Для тех же данных по быстродействию, что и ранее, в конвейеризованных структурах для передач внутри пакета были получены задержки "синхросигнал—выход" около 5—8 нс и работа без тактов ожидания на частотах более 75 МГц (внутри пакета).

Статические ЗУ с ускоренным реверсом шины. В номенклатуре SRAM появились архитектуры с *уменьшенным временем реверсирования направления передач по шине данных* (временем, затрачиваемым на реверс шины при изменениях режимов "запись—чтение"). К таким архитектурам относятся микросхемы с маркировками ZBT (Zero Bus Turnaround), NoBL (No Bus Latency), NtRAM (No Turnaround RAM) и др. В этих архитектурах удалось

исключить свободный такт, обычно возникающий при переходе от операции чтения к операции записи и наоборот. Для современных DDR SRAM указанного типа (с отсутствием шинной задержки) приводятся параметры — информационная емкость 8 Мбит и тактовая частоты 400 МГц. Тестовая схема фирмы Intel (2003 г.) имеет информационную емкость 16 Мбит при тактовой частоте 900 МГц.

В современных быстродействующих синхронных SRAM *сочетаются многие способы повышения быстродействия*. Например, в микросхемах фирмы Cypress Semiconductor используется низковольтное питание (1,8 или 2,5 В), конвейеризованная структура, технология DDR, пакетная передача адресов с применением двухразрядного внутреннего счетчика, двухпортовость и т. д. В результате при тактовой частоте 300 МГц на каждом такте передаются 4 слова с разрядностью 18 или 36 для разных представителей семейства.

Искусственная энергонезависимость статических ЗУ с резервным источником питания

Статические ОЗУ энергозависимы — при снятии питания информация в триггерных запоминающих элементах теряется. Можно придать им искусственную энергонезависимость с помощью резервного источника питания. Это наиболее пригодно для ЗУ на элементах КМОП, т. к. они в режиме хранения потребляют чрезвычайно малую мощность.

Для подключения к накопителю ЗУ резервного источника питания разработчики памяти рекомендуют схему, приведенную на рис. 4.36, *а*. В этой схеме напряжение резервного источника несколько ниже напряжения основного источника U_{CC}. В рабочем режиме накопитель питается от напряжения U_{CC}, при этом диод D1 проводит, а диод D2 заперт. При снижении рабочего напряжения к накопителю автоматически подключается источник резервного питания. При этом проводит диод D2, а диод D1 запирается, т. к. при малых значениях U_{CC} он попадает под обратное смещение.

При использовании в микропроцессорных системах вариант, показанный на рис. 4.36, *а*, недостаточно надежен в связи со следующим обстоятельством. Напряжение питания системы U_{CC} вырабатывается источником, на выходе которого обычно имеется сглаживающий фильтр со значительной инерционностью. Поэтому при аварии питания напряжение U_{CC} не исчезает сразу, а снижается относительно медленно. На начальном этапе этого процесса система продолжает работать, но в ее работе возможны ошибки. Желательно быстрее отреагировать на аварию питания. Это достигается с помощью специальной схемы подключения (рис. 4.36, *б*).

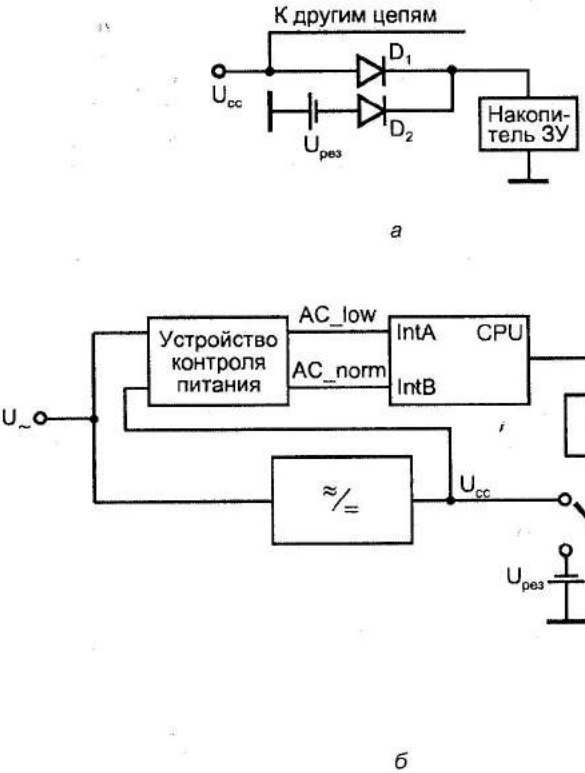


Рис. 4.36. Схемы подключения резервных источников питания к накопителям ЗУ

Здесь нарушение нормальной работы источника питания обнаруживается раньше, чем в предыдущей схеме, с помощью *контроля напряжения переменного тока* (AC — Alternate Current). Нарушение можно выявить за один-два периода переменного напряжения, пока постоянное напряжение U_{CC} еще практически не изменилось. Признак нарушения AC_low служит запросом прерывания для процессора CPU. Получив запрос, процессор выполняет подпрограмму обслуживания прерывания A (Interrupt A), в ходе которого передает содержимое своих регистров в стек накопителя (выполняет так называемое контекстное переключение) и заканчивает подпрограмму установкой триггера T, что воздействует на обмотку реле, управляющего ключом. В результате память подключается к резервному источнику. При восстановлении нормального питания от основного источника признак AC_norm вызывает свою программу обслуживания прерывания B, в ходе которой из стека возвращаются в процессор данные для регистров процессора и сбрасывается триггер, что подключает память к основному источнику питания.

Энергонезависимая память типа NV-SRAM

Память типа NV-SRAM (Non-Volatile SRAM) — вид "неразрушаемой" статической оперативной памяти, способной сохранять свое содержимое при снятии питания. Это свойство обеспечивается *добавлением к ОЗУ специального РПЗУ-ЭС*. Такое РПЗУ служит для автоматического копирования в него данных из ОЗУ при снижении напряжения питания и обратной передачи данных в ОЗУ при восстановлении нормального уровня питающего напряжения. Получающаяся *комбинация двух типов памяти* дает сочетание SRAM, необходимого для скоростной оперативной работы с процессором, и постоянного ЗУ для хранения данных при снятии питания.

Технологические особенности выпускаемых в настоящее время схем NV-SRAM состоят в следующем. Запоминающие элементы РПЗУ-ЭС фактически являются МНОП-транзисторами, а статическое ОЗУ выполнено на основе четырехтранзисторных триггерных запоминающих элементов с высокоомными нагрузками. Такое решение обеспечило как приемлемую плотность размещения запоминающих элементов на кристалле, так и достаточно высокое быстродействие ОЗУ. В каждый элемент ОЗУ встраиваются два элемента РПЗУ-ЭС для сохранения присущего триггерным запоминающим элементам дифференциального представления и дифференциальных передач данных, дающих надежную работу ЗУ в широком диапазоне изменения температуры.

Копирование данных в РПЗУ-ЭС происходит параллельно во всех элементах памяти. При этом вначале РПЗУ-ЭС, будучи отключенным от ОЗУ, полностью очищается. Затем элементы РПЗУ-ЭС и ОЗУ соединяются и происходит копирование данных в энергонезависимую память. Необходимые для стирания и программирования уровни напряжений вырабатываются встроенными преобразователями. В большинстве микросхем NV-SRAM реализован режим автоматического копирования данных в энергонезависимую память при снижении напряжения питания ниже заданного порогового уровня. Для копирования требуется около 10 мс и нужно, чтобы за это время напряжение питания снизилось не более чем на допустимую величину (приблизительно 0,7 В). Если требуемое условие не соблюдается, можно специально замедлить спад напряжения питания, включив, например, на входе питания микросхемы электролитический конденсатор с емкостью порядка 100 мкФ или более и развязав его от источника питания диодом с малым прямым сопротивлением (диодом Шоттки).

Обратная передача данных осуществляется автоматически при появлении нормального питания и также происходит параллельно для всех элементов и в два этапа (очистка и копирование). Восстановление данных осуществляется за время в несколько сотен микросекунд после достижения напряжением питания порогового уровня. При необходимости процесс восстановления данных может быть инициирован программой. Некоторые микросхемы NV-SRAM имеют специальный вывод для аппаратной инициализации процесса

сохранения данных путем подачи на этот вывод соответствующего уровня напряжения. В рабочем режиме элементы РПЗУ-ЭС отключены от оперативного ЗУ, и память работает как обычная SRAM.

Микросхемы NV-SRAM выпускаются с 1996 г., их емкости составляют сейчас 16—256 Кбит при временах доступа в режиме ОЗУ около 20 нс.

Статические ЗУ типа БиКМОП

Триггерные ЗУ — одно из основных направлений применения БиКМОП-технологии, в которой стремятся объединить достоинства схем на основе биполярных приборов и МОП-структур. Применительно к SRAM это реализация триггеров на схемах КМОП, а цепей выдачи данных, имеющих значительную емкостную нагрузку, с которой элементы типа КМОП справляются плохо, — на биполярной схемотехнике (ЭСЛ или ТТЛШ). Повышенная сложность изготовления БиКМОП-схем и их удорожание могут быть скомпенсированы более высоким их быстродействием, эффективной работой на длинные линии и другими факторами.

На рис. 4.37 показана для примера ячейка двухпортового ЗУ с организацией 4К×1 и временем доступа 4 нс (следует заметить, что эта ячейка разработана относительно давно, и приводимые цифры не характеризуют возможностей современной технологии). Ячейка выполнена по БиКМОП-технологии. Запоминающий триггер построен на транзисторах Т1—Т4. Его выход подключен к базе биполярного транзистора Т6, который совместно с опорным транзистором Т7, общим для всех ячеек столбца, образует схему токового переключателя, характерного для ЭСЛ и способного с большой скоростью коммутировать ток из одного плеча в другое. Показанный условно источник тока реально выполняется так же, как и в обычных схемах ЭСЛ. Возможность быстро формировать сигналы в нагруженных цепях линий записи-считывания позволяет сохранить быстродействие на уровне, соответствующем внутренним частям ЗУ, в которых КМОП-схемы работают в условиях малых нагрузок.

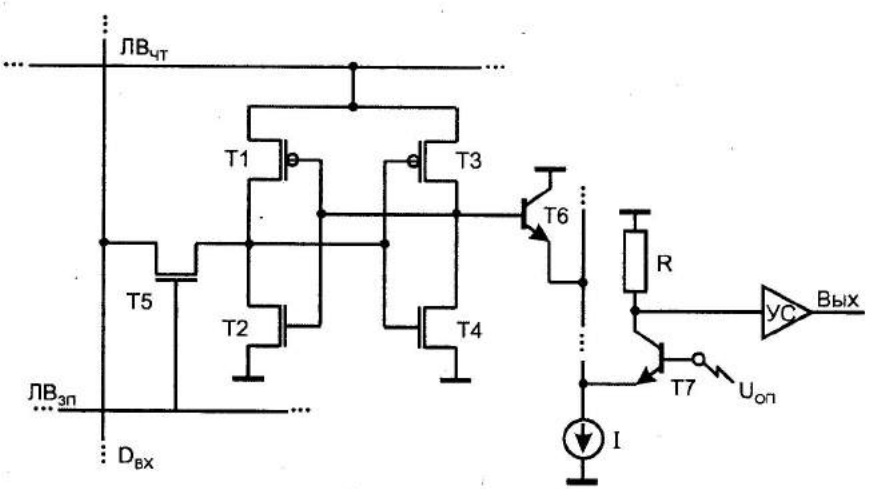


Рис. 4.37. Схема ячейки статического ЗУ в схемотехнике БиКМОП

Ячейка имеет две линии выборки — для чтения ($ЛВ_{чт}$) и для записи ($ЛВ_{зп}$). Это позволяет записывать данные в невыбранные для чтения элементы одновременно со считыванием из других элементов, что характерно для двухпортовой памяти.

Питанием ячейки служит потенциал линии $ЛВ_{чт}$. В отсутствие выборки для чтения этот потенциал невысок и любые переключения триггера не могут настолько повысить потенциал базы Т6, чтобы он открылся. Запись данных производится сигналом $D_{вх}$ при выборке ячейки по линии $ЛВ_{зп}$. Транзистор Т5 изготовляется как низкоомный, что позволяет ему диктовать состояние триггера.

Для чтения напряжение на линии $ЛВ_{чт}$ повышают на 0,55 В. Если триггер хранит единицу, то Т3 открыт, а Т4 заперт. Так как при этом перепад напряжения на $ЛВ_{чт}$ передается на базу Т6, он открывается, и ток I переключается из опорного транзистора Т7 в транзистор Т6. Напряжение на коллекторе Т7 повышается, что и служит входным сигналом чтения единицы для последующих каскадов усилителя чтения, обозначенных как УС. Если триггер хранит логический нуль, то Т3 заперт и Т4 открыт. Ясно, что в этом случае перепад напряжения на линии $ЛВ_{чт}$ никак не повлияет на потенциал базы Т6, переключения тока I не возникнет и перепада выходного напряжения схемы не будет.

§ 4.9. Динамические запоминающие устройства — базовая структура

В динамических ЗУ (DRAM) данные хранятся в виде зарядов емкостей МОП-структур и основой ЗЭ является просто конденсатор небольшой емкости. Такой ЗЭ значительно проще триггерного, содержащего 4 или 6 транзисторов, что позволяет разместить на кристалле намного больше ЗЭ (примерно в 4—5 раз) и обеспечивает динамическим ЗУ максимальную емкость. В то же время из-за токов утечки конденсатор со временем неизбежно теряет свой заряд, и хранение данных требует их периодической регенерации (через каждые несколько миллисекунд).

Запоминающие элементы

Известны конденсаторные ЗЭ разной сложности. В последнее время практически всегда применяют однотранзисторные ЗЭ — лидеры компактности, размеры которых настолько малы, что на их работу стали влиять даже α-частицы, излучаемые элементами корпуса ИС.

Электрическая схема и конструкция однотранзисторного ЗЭ показаны на рис. 4.38. Ключевой транзистор отключает запоминающий конденсатор от линии записи-считывания или подключает его к ней. Сток транзистора не имеет внешнего вывода и образует одну из обкладок конденсатора. Другой обкладкой служит подложка. Между обкладками расположен тонкий слой диэлектрика — оксида кремния SiO_2.

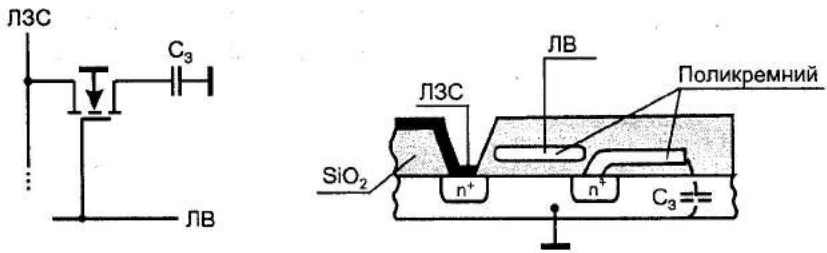


Рис. 4.38. Схема и конструкция запоминающего элемента динамического ЗУ

В режиме хранения ключевой транзистор заперт. При выборке данного ЗЭ на затвор подается напряжение, отпирающее транзистор. Запоминающая емкость через проводящий канал подключается к линии записи-считывания и в зависимости от заряженного или разряженного состояния емкости различно влияет на потенциал линии записи-считывания. При записи потенциал линии записи-считывания передается на конденсатор, определяя его состояние.

Процесс чтения состояния запоминающего элемента. Здесь и ниже основное внимание уделяется операции чтения данных, т. к. именно чтение лимитирует быстродействие динамического ЗУ. При чтении требуется перезаряжать длинную разрядную линию с большой емкостной нагрузкой через маломощный транзистор адресованного запоминающего элемента. При записи ситуация облегчается, т. к. записываемые данные можно подавать на линии записи/считывания через мощные буферные каскады (драйверы).

Фрагмент ЗУ (рис. 4.39) показывает ЗЭ, усилитель считывания УС, а также ключи К1 и К0 соответственно для записи единицы и нуля. К линии записи-считывания (ЛЗС) подключено столько ЗЭ, сколько строк имеется в запоминающей матрице. Особое значение имеет емкость ЛЗС $C_л$, в силу большой протяженности линии и большого числа подключенных к ней транзисторов многократно превышающая емкость ЗЭ.

Перед считыванием производится предварительный заряд (предзаряд) ЛЗС. Имеются варианты ЗУ с предзарядом ЛЗС до уровня напряжения питания и до уровня его половины. Рассмотрим последний вариант в силу его большей схемной простоты. Итак, перед считыванием емкость $C_л$ заряжается до уровня $U_{CC}/2$. Будем считать, что хранение единицы соответствует заряженной емкости $C_з$, а хранение нуля — разряженной.

При считывании нуля к ЛЗС подключается емкость $C_з$, имевшая нулевой заряд. Часть заряда емкости $C_л$ перетекает в емкость $C_з$, и напряжения на них уравниваются. Потенциал ЛЗС снижается на величину ΔU, которая и является сигналом, поступающим на усилитель считывания. При считывании единицы, напротив, напряжение на $C_з$ составляло вначале величину

U_{CC} и превышало напряжение на ЛЗС. При подключении $C_з$ к ЛЗС часть заряда стекает с запоминающей емкости в $C_л$ и напряжение на ЛЗС увеличивается на ΔU. Графики сигналов при считывании нуля и единицы показаны на рис. 4.40.

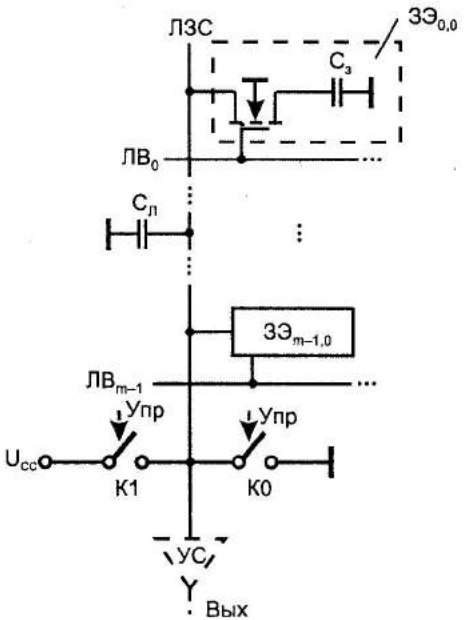


Рис. 4.39. Фрагмент схемы динамического ЗУ

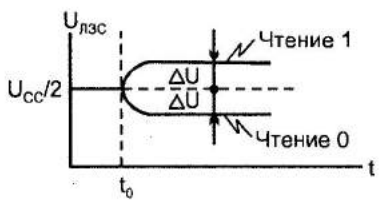


Рис. 4.40. Временные диаграммы сигналов при считывании данных в динамических ЗУ

Значение ΔU нетрудно вычислить на основе анализа любого из процессов — считывания нуля или считывания единицы. Для считывания нуля справедливы следующие рассуждения. До выборки ЗЭ емкость ЛЗС имела заряд $Q = C_л U_{CC}/2$.

После выборки ЗЭ этот же заряд имеет суммарная емкость $C_л + C_з$ и можно записать следующее соотношение:

$$Q = (C_л + C_з)(U_{CC}/2 - \Delta U).$$

Приравнивая выражения для одного и того же значения заряда Q, получим соотношение

$$C_л U_{CC}/2 = (C_л + C_з)(U_{CC}/2 - \Delta U),$$

из которого следует выражение

$$\Delta U = U_{CC}C_з/[2(C_з + C_л)] \approx U_{CC}C_з/2C_л.$$

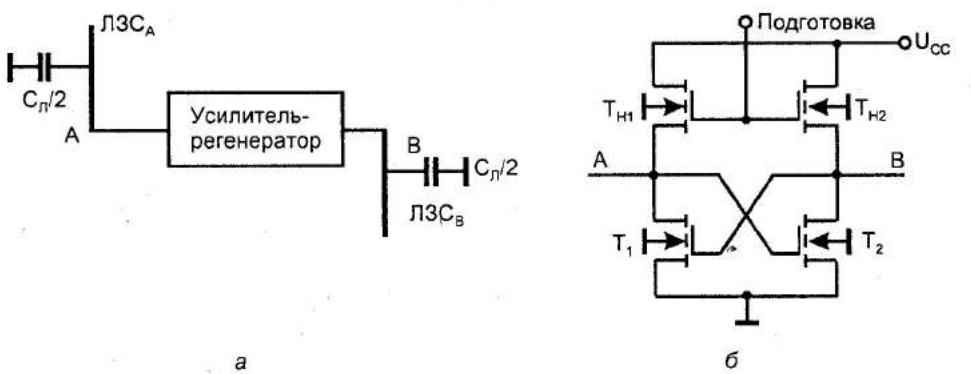
В силу неравенства $C_з << C_л$ сигнал ΔU оказывается слабым.

Кроме того, считывание является разрушающим — подключение запоминающей емкости к ЛЗС изменяет ее заряд.

Мерами преодоления отмеченных недостатков служат увеличение емкости $C_з$ (без увеличения площади ЗЭ), уменьшение емкости ЛЗС и применение усилителей-регенераторов для считывания данных.

В направлении увеличения $C_з$ можно указать разработку новых диэлектриков, имеющих бóльшую диэлектрическую постоянную (бóльшую, чем SiO_2). Это позволяет при той же емкости сократить площадь ЗЭ или увеличить $C_з$ даже при уменьшении ее площади. Имеются и варианты с введением в ЗЭ токоусиливающих структур, что также эквивалентно увеличению емкости ЗЭ.

Уменьшения емкости ЛЗС можно достичь "разрезанием" этой линии на две половины с включением дифференциального усилителя считывания в разрыв между половинами ЛЗС (рис. 4.41, *а*). Очевидно, что такой прием вдвое уменьшает емкость линий, к которым подключаются запоминающие емкости, т. е. вдвое увеличивает сигнал ΔU.



а *б*

Рис. 4.41. Схема включения усилителя-регенератора в разрыв линии записи-считывания динамического ЗУ (*а*) и вариант схемной реализации усилителя-регенератора (*б*)

Усилители-регенераторы

Усилители-регенераторы строятся на основе триггерных схем. Один из возможных вариантов (рис. 4.41, *б*) основан на введении в схему дополнительного сигнала "Подготовка" для управления нагрузочными транзисторами T_{H1} и T_{H2}. Вначале сигнал "Подготовка" имеет низкий уровень и нагрузочные транзисторы заперты. В этом состоянии усилитель-регенератор воспринимает слабые сигналы считывания с линий ЛЗС. Одна из половин ЛЗС, к которой не подключается C_3, сохраняет напряжение предзаряда $U_{CC}/2$, напряжение на другой половине, к которой подключается выбранный ЗЭ, отклоняется от напряжения предзаряда на ΔU в ту или иную сторону в зависимости от того, считывается единица или нуль. Неравенство напряжений в точках A и B вносит неидентичность проводимостей транзисторов T_1 и T_2. Для считывания и регенерации данных сигнал "Подготовка" переводится на высокий уровень. Транзисторы T_{H1} и T_{H2} открываются, и возникает схема триггера, находящегося в неустойчивом состоянии, близком к симметричному. Такой триггер в силу своих свойств быстро перейдет в устойчивое состояние, предопределенное начальной несимметрией его режима. На выходах триггера сформируются полные напряжения высокого и низкого уровней. Так как одни и те же точки A и B являются одновременно и входами, и выходами усилителя-регенератора, то после своего срабатывания он восстанавливает на емкости C_3 полное значение считанного сигнала. Тем самым автоматически осуществляется регенерация данных в ЗЭ. Состояние триггера определяет также сигналы, выводимые во внешние цепи в качестве считанной информации.

Мультиплексирование шины адреса

Особенностью почти всех динамических ЗУ является мультиплексирование шины адреса. Адрес делится на два полуадреса, один из которых представляет собою адрес строки, а другой — адрес столбца матрицы ЗЭ. Оба полуадреса подаются на одни и те же выводы корпуса ИС поочередно. Подача адреса строки сопровождается стробом $\overline{RAS}$ (Row Address Strobe), а адреса столбца — стробом $\overline{CAS}$ (Column Address Strobe). Причиной мультиплексирования адресов служит стремление уменьшить число выводов корпуса ИС и тем самым удешевить ее, а также то обстоятельство, что полуадреса и сигналы $\overline{RAS}$ и $\overline{CAS}$ в некоторых режимах и схемах используются различно (например, в режимах построчной регенерации адрес столбца вообще не нужен). Сокращение числа внешних выводов корпуса для динамических ЗУ особенно актуально, т. к. они имеют максимальную емкость и, следовательно, большую разрядность адресов. Например, ЗУ с организацией 16M×1 имеет 24-разрядный адрес, а мультиплексирование сократит число адресных линий на 12. Мультиплексирование шины адреса — ситуация, типичная для динамиче-

ских ЗУ и до недавнего времени почти обязательная. Однако в последнее время появились и варианты DRAM с одновременной передачей в микросхему всех разрядов адреса. Такой подход — проявление отказа от оптимизации схемы по критериям аппаратной простоты в пользу стремления к максимальному ее быстродействию.

Внешняя организация и временные диаграммы

На рис. 4.42 показаны внешняя организация и временные диаграммы стандартного динамического ОЗУ.

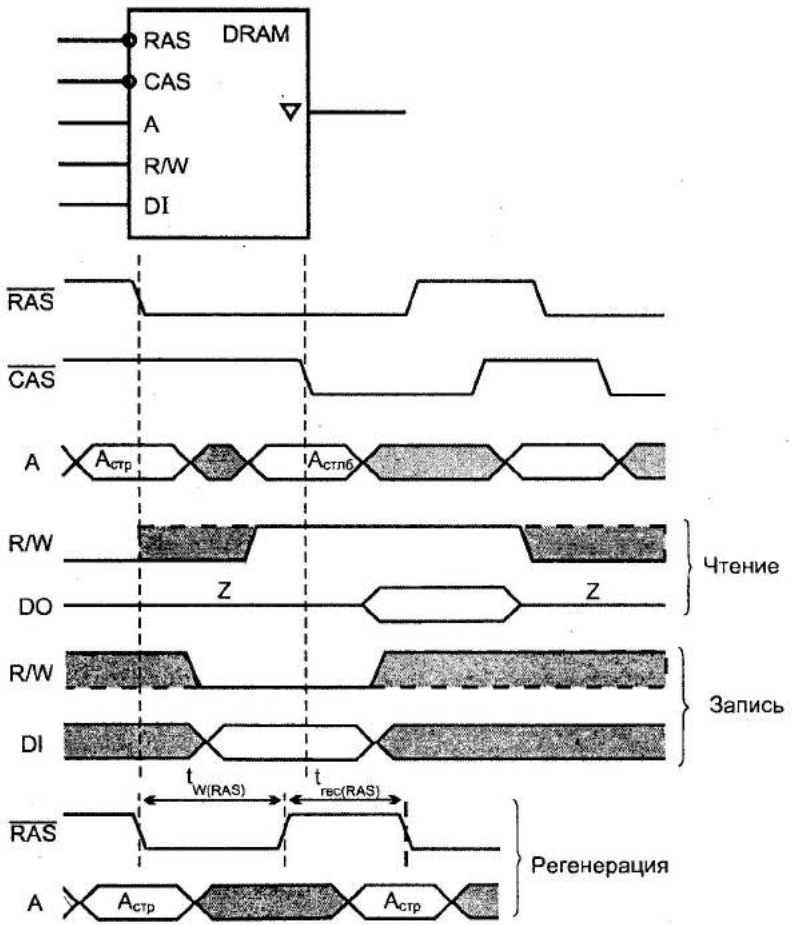


Рис. 4.42. Пример внешней организации и временных диаграмм динамического ЗУ

Циклы обращения к ЗУ начинаются сигналом $\overline{RAS}$ и запаздывающим относительно него сигналом $\overline{CAS}$. Отрицательным фронтам этих сигналов соответствуют области подачи на адресные линии ЗУ полуадресов, адресующих строки и столбцы матрицы соответственно. Согласно указанию выполняемой операции (сигналу R/W), либо вырабатываются выходные данные DO, либо принимаются входные данные DI. В циклах регенерации подаются только импульсные сигналы $\overline{RAS}$ и адреса строк. Области безразличных значений сигналов на рисунке заштрихованы.

Схема динамического ЗУ

В схеме динамического ЗУ (рис. 4.43) один из столбцов матрицы раскрыт полностью, другие столбцы аналогичны ему. Ключевые транзисторы для простоты изображения представлены кружками, как пояснено в левом верхнем углу рисунка. Обозначения блоков стандартны за исключением обозначения ФТС — формирователь тактирующих сигналов.

В исходном состоянии (до обращения к ЗУ) сигнал $\overline{RAS}$ пассивен, т. е. имеет высокий уровень, который замыкает ключи 1 и подает напряжение $U_{CC}/2$ на полушины записи-считывания $ЛЗС_а$ и $ЛЗС_в$ для их предзаряда. При обращении к ЗУ активизируется сигнал $\overline{RAS}$ одновременно с подачей по шине адреса А первого полуадреса (адреса строки). При этом ключи 1 размыкаются и линии записи-считывания изолируются от источника напряжения $U_{CC}/2$, а формирователь ФТС1 вырабатывает пару последовательных сигналов Ф1 и Ф2. Тактирующий сигнал Ф1 разрешает загрузку регистра РгХ и работу дешифратора ДШХ, одна из выходных линий которого возбуждается и выбирает все ЗЭ строки, адрес которой содержится в регистре РгХ.

В разрыв между секциями $ЛЗС_а$ и $ЛЗС_в$ включен усилитель-регенератор, для которого подключение ЗЭ, хранящего единицу или нуль, создает дисбаланс входных сигналов.

Второй тактирующий сигнал Ф2 снимает сигнал "Подготовка" с усилителей-регенераторов, и они срабатывают, формируя в своих точках входов/выходов полные уровни сигналов, что восстанавливает состояния ЗЭ выбранной строки.

Для последующих операций чтения или записи требуется наличие сигнала $\overline{CAS}$, разрешающего формирователю ФТС2 формирование второй пары тактирующих сигналов Ф3 и Ф4. Сигнал Ф3 загружает в РгY адрес столбца, а Ф4 активизирует дешифратор ДШY, вследствие чего открываются ключи 2 выбранного столбца.

В зависимости от сигнала R/W линии ЛЗС подключаются либо к выходной шине данных (через ключ 4 при R/W = 1), либо к линии входных данных (через ключи 3 при R/W = 0).

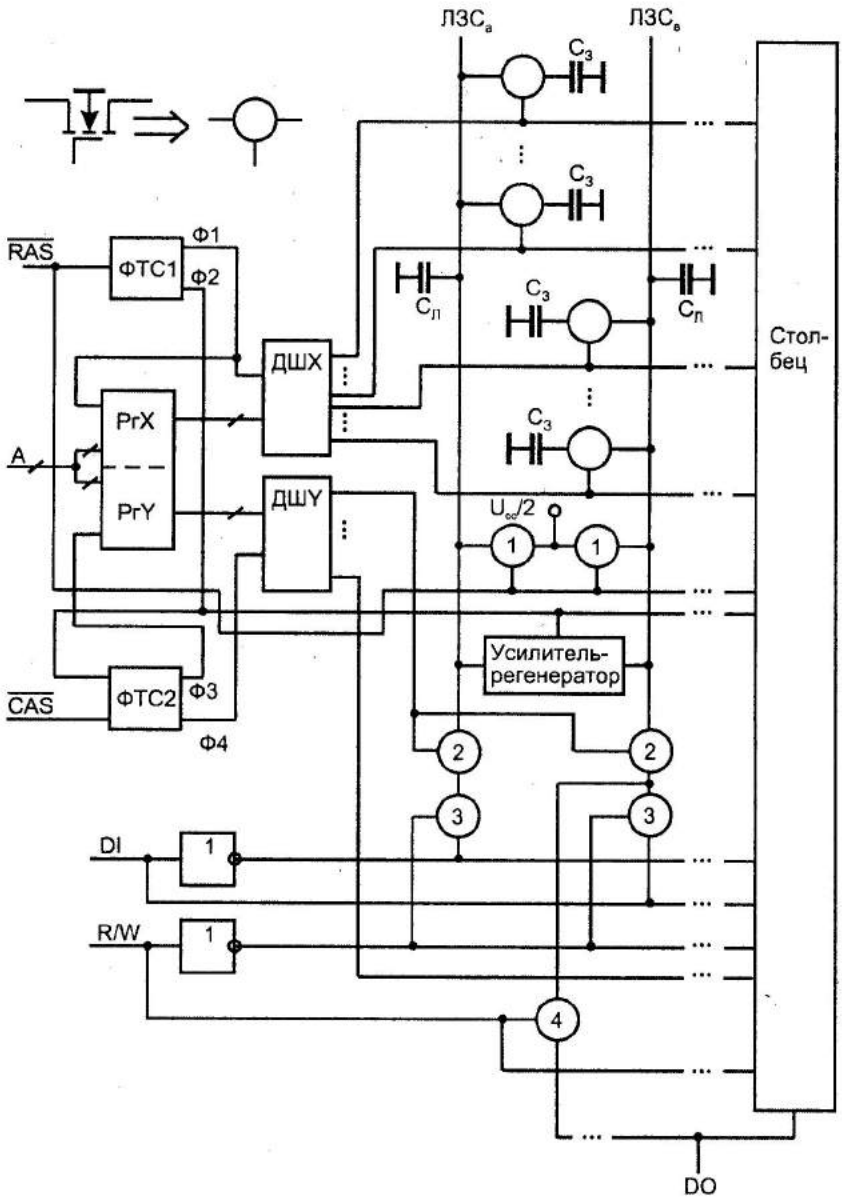


Рис. 4.43. Схема динамического ЗУ

Для операции регенерации, целиком проходящей внутри ЗУ, связь с внешними выводами не требуется, поэтому для нее достаточно подачи только сигнала $\overline{\text{RAS}}$ (совместно с адресами регенерируемых строк) и выработки только тактирующих сигналов Ф1 и Ф2.

Кроме режимов записи и считывания, в динамических ЗУ иногда организуют дополнительные режимы, в частности, *режим "считывание-модификация-запись"*. В этом режиме в одном цикле слово считывается и вновь записывается по тому же адресу, но может быть изменено (модифицировано). Такой режим используется в ЗУ с коррекцией ошибок, например, с применением кодов Хемминга. В этом случае слово с контрольными разрядами считывается, проверяется контрольной схемой и при необходимости исправляется и вновь записывается по старому адресу. Длительность цикла режима "считывание-модификация-запись" больше циклов записи и считывания, но меньше их суммы, поэтому время на коррекцию содержимого ЗУ сокращается.

§ 4.10. Регенерация данных в динамических запоминающих устройствах

Во избежание потери информации динамические ЗУ нуждаются в постоянной регенерации. Без обновления информация в виде зарядов конденсаторов может сохраняться только в течение нескольких миллисекунд (в современных ИС это интервал от 1 до 15 мс).

Традиционным режимом регенерации является режим строчной регенерации путем осуществления циклов чтения по всем строкам матрицы ЗЭ. При этом процесс не сопровождается выдачей данных на выходные буферы, а целиком проходит внутри ЗУ. Используются только адреса строк, а адреса столбцов не требуются.

Если длительность цикла чтения t_{CY}, а число строк матрицы ЗУ $N_{стр}$, то на регенерацию данных потребуется время $t_{рег} = t_{CY}N_{стр}$. Относительные потери времени на регенерацию составят величину

$$\tau_{рег} = (t_{рег}/T_{рег}),$$

где $T_{рег}$ — период повторения операции регенерации.

Например, в ЗУ емкостью 1 Мбит с организацией 1М×1, для которого длительность цикла чтения равна 100 нс, а период регенерации составляет 5 мс, потери времени на регенерацию составят

$$\tau_{рег} = (100 \cdot 10^{-9} \cdot 2^{10}/5 \cdot 10^{-3}) \cdot 100\% = 2\%$$

($2^{10} = 1024$ — число строк в квадратной матрице, содержащей 1М запоминающих элементов).

Пример структуры контроллера регенерации, управляющего этим процессом, приведен на рис. 4.44. Модуль памяти составлен из одноразрядных микросхем, число которых равно разрядности хранимых в ЗУ слов. Относительно входных сигналов все микросхемы включены параллельно. В рабочем режиме модулем управляет процессор, в режиме регенерации — контроллер. В рабочем режиме триггеры Т1 и Т2 сброшены. Нулевое значение выхода Т2

сбрасывает счетчик CTR, блокирует передачу через элемент И-ИЛИ строба RAS_{per} и по адресному входу A мультиплексора MUX2 обеспечивает передачу на выход этого мультиплексора адресов от мультиплексора MUX1.

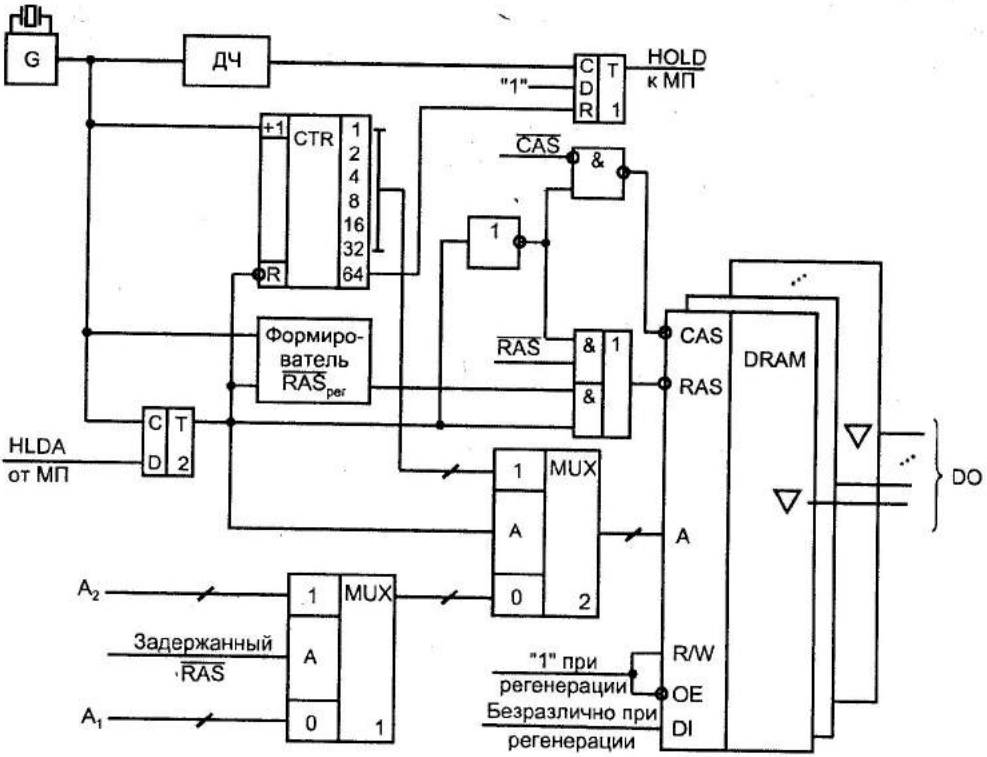


Рис. 4.44. Схема контроллера динамического ОЗУ

При этом модуль памяти получает сигналы $\overline{RAS}$ и $\overline{CAS}$, соответствующие рабочему режиму, адреса A_1 и A_2 строк и столбцов, выдаваемые процессором в сопровождении стробов $\overline{RAS}$ и $\overline{CAS}$, а также сигналы управления R/W и $\overline{OE}$. При записи модулем памяти воспринимаются входные данные DI, при чтении выдаются выходные данные DO. Так реализуется рабочий режим.

Генератор G непрерывно генерирует последовательность импульсов, период повторения которых равен длительности цикла чтения ЗУ. Делитель частоты ДЧ понижает частоту импульсов генератора так, что на его выходе период повторения импульсов будет равен периоду регенерации T_{per} (составит несколько миллисекунд). Таким образом, с периодом T_{per} на выходе ДЧ появ-

няется импульс, что заставляет триггер T1 принять единичное состояние и инициировать режим регенерации. Единичное значение сигнала HOLD является сигналом запроса на управление памятью со стороны контроллера. Этот сигнал поступает на соответствующий вход процессора. Процессор не может остановиться мгновенно, т. к. для прерывания выполняемой им программы требуются определенные операции. Произведя эти операции, процессор вырабатывает сигнал HLDA, разрешающий переход к операции регенерации ЗУ. Сигнал HLDA устанавливает триггер T2, в результате чего блокируется передача стробов $\overline{RAS}$ и $\overline{CAS}$ на модуль памяти, разрешается передача на вход $\overline{RAS}_{рег}$, вырабатываемого формирователем контроллера, мультиплексор MUX2 переключается на передачу адресов со счетчика CTR на адресный вход ЗУ. Одновременно с этим триггер T2 снимает сигнал асинхронного сброса со входа $\overline{R}$ счетчика, и он начинает перебирать адреса строк от нулевого до максимального (конкретно в показанной схеме таких адресов 64). Появление импульса переполнения счетчика сбрасывает триггер T1, обозначая этим окончание операции регенерации и снимая сигнал HOLD. В ответ процессор снимает сигнал HLDA, после чего очередной импульс генератора сбрасывает T2, возвращая схему в рабочий режим.

В последнее время разработаны совмещенные контроллеры кэш-памяти и динамических ЗУ.

В некоторых ЗУ схемы регенерации данных реализованы на самом кристалле памяти, и от разработчика не требуется специальных мер по организации этого процесса. Такие ЗУ называют *квазистатическими*. Примером квазистатических ЗУ служат микросхемы фирмы Mosys (Monolithic Systems), называемые 1T-SRAM и трактуемые как статические ЗУ с однотранзисторными запоминающими элементами, что по существу неправильно, т. к. запоминающим элементом является все-таки конденсатор.

§ 4.11. Динамические запоминающие устройства повышенного быстродействия

Современные микропроцессоры характеризуются высоким быстродействием. Это требует и увеличения скорости работы ОЗУ, обменивающихся информацией с процессорами. Эта задача важна и для разработчиков динамических ОЗУ, которые благодаря максимальной информационной емкости и низкой стоимости занимают ведущее место в составе основной памяти компьютеров.

В ходе развития предложен ряд вариантов динамических ОЗУ повышенного быстродействия. Методы ускорения работы, использованные в этих ОЗУ, можно разделить на две группы. Методы первой группы *основаны на предпо-*

ложении о кучности адресов при обращениях к ОЗУ, справедливом, когда адреса последующих обращений к ОЗУ вероятнее всего расположены близко к адресу текущего обращения. Если такое предположение не оправдывается, то методы первой группы становятся неэффективными, и основанные на них ЗУ работают с той же скоростью, что и обычные стандартные архитектуры. Методы второй группы *дают ускорение работы памяти даже при произвольном доступе к ее ячейкам*, когда адреса предыдущих и последующих обращений к памяти никак не взаимосвязаны.

Разработка DRAM повышенного быстродействия началась с освоения методов первой группы и лишь недавно появились варианты, эффективные при произвольном доступе к хранимым данным. Кроме того, производятся и микросхемы с комбинированием методов обеих групп. В них реализуются в основном методы первой группы, благодаря которым темп передач в пакете выше, чем вне пакета, но одновременно с этим приняты меры к сокращению времени доступа к первому слову пакета, так что схемы стали более эффективными и при произвольном доступе к данным.

Структуры FPM, EDORAM, BEDORAM

Вариант FPM (Fast Page Mode, быстрый страничный доступ) эффективен, если после обращения к некоторому ЗЭ следующее обращение будет к ЗЭ той же строки матрицы запоминающих элементов. Сравним такую ситуацию с обращением по произвольному адресу.

При чтении по произвольному адресу старший полуадрес выбирает строку, затем младший полуадрес выбирает столбец в матрице ЗЭ. При этом сначала требуется перезарядить шину выборки строки, а затем шину выборки столбца, что сопровождается соответствующими задержками.

При выборке строки (страницы) происходят процессы, соответствующие первым фазам полного цикла, описанного выше при рассмотрении работы базовой схемы ЗУ. Это процессы, связанные с исходным состоянием и затем активизацией строба $\overline{\text{RAS}}$. В ходе этих процессов производится предзаряд линий считывания, подключение к ним запоминающих элементов строки и срабатывание усилителей-регенераторов. После завершения перечисленных процессов ЗУ готово к выполнению очередных фаз, связанных с активизацией строба $\overline{\text{CAS}}$ и выборкой из строки необходимого столбца.

При обращении к данным в пределах одной страницы адрес строки остается неизменным, изменяются только адреса столбцов в сопровождении строба $\overline{\text{CAS}}$. Изменяет состояние фактически только группа ключей 3 и 4 (см. рис. 4.43). Пока не изменился номер страницы, в циклах обмена исключены первые этапы, что сокращает длительность циклов.

Временные диаграммы для режима FPM представлены на рис. 4.45. Видно, что время доступа к данным при неизменности адреса строки RA и изменениях только адреса столбца CA1...CA_m сокращается в сравнении со временем доступа при полном цикле (временем доступа при первом обращении к ЗУ). Характерную пропорциональность времен первого и последующих обращений к ЗУ можно записать следующим образом: 5-3-3-...

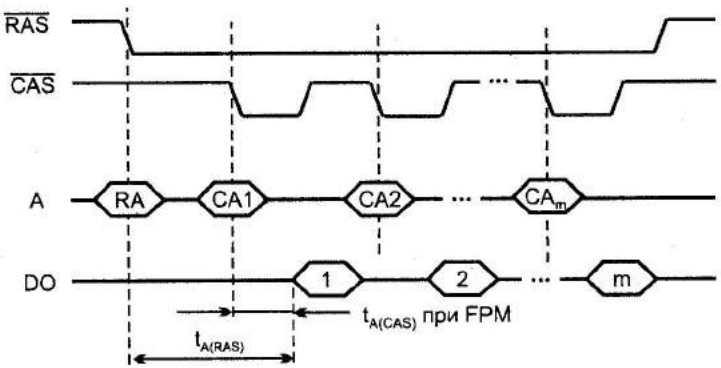


Рис. 4.45. Временные диаграммы режима FPM динамических ОЗУ

В компьютерной технике FPM DRAM пришли на смену стандартным DRAM в середине 1990-х гг. Режим FPM — начало линии развития методов повышения быстродействия динамических ЗУ. По быстродействию его возможности уже намного превышены более поздними разработками. Дополнительные средства для организации режима FPM просты: требуется лишь проверять принадлежность очередного адреса текущей странице (строке), что позволяет выполнять цикл страничного режима. В противном случае требуется выполнение обычного (полного) цикла. Разработанные ОЗУ типа FPM обеспечивали времена обращения к ЗУ в пределах страницы около 30—40 нс, что допускало их работу с процессорными шинами на тактовой частоте до 33 МГц.

Структуры типа EDORAM (Enhanced Data Out RAM, т. е. ОЗУ с усовершенствованным выводом данных) близки к структурам FPM и отличаются от них модификацией процесса вывода данных. В EDORAM данные в усилителях-регенераторах не сбрасываются по окончании строба $\overline{CAS}$. При этом на кристалле как бы появляется статический регистр, хранящий строку. При обращениях в пределах строки (страницы) используется чтение данных из регистра, т. е. быстродействующей статической памяти. По-прежнему используется только сигнал $\overline{CAS}$, но длительность его может быть сокращена в сравнении с режимом FPM. Это увеличивает быстродействие ЗУ. В случае

применения памяти типа EDORAM характерная пропорциональность времен первого и последующих обращений к словам в пределах страницы будет следующей: 5-2-2-...

В микросхемах EDORAM присутствуют элементы конвейеризации — содержимое выбранной ячейки защелкивается в регистре, и уже в это время можно подавать на входы матрицы адрес следующей ячейки. Возникает совмещение операций во времени, увеличивающее скорость считывания данных в пределах страницы.

Разработанные EDORAM допускали работу на частотах до 50 МГц. Такие ЗУ получили в свое время широкое распространение, в частности, из-за тесной преемственности с разработанными ранее ЗУ типа FPM, замена которых на EDORAM требует лишь небольших изменений в схеме и синхросигналах ЗУ.

В структуре типа BEDORAM (Burst EDORAM, т. е. с пакетным расширенным доступом) содержится дополнительно внутренний счетчик адресов столбцов. При обращении к группе слов (пакету) адрес столбца формируется обычным способом только в начале пакетного цикла. Для последующих передач адреса образуются быстро с помощью инкрементирования внутреннего счетчика. Характерный *тайминг* 5-1-1-1 (имеется в виду часто применяемый вариант с длиной пакета, равной 4). Память BEDORAM не получила широкого распространения из-за появления почти одновременно с нею сильного конкурента с таким же таймингом 5-1-1-1, а именно синхронных DRAM (SDRAM), в которых достигается не только доступ к словам внутри пакета с единичным интервалом времени (в каждом такте), но и сами длительности такта существенно сокращаются.

Структура типа MDRAM

В структурах MDRAM (Multibank DRAM, многобанковые ОЗУ) память делится на части (банки). При кучности адресов обращение к банкам будет преимущественно поочередным, что исключит задержки на подготовку памяти к очередному обращению (на перезаряд шин). Пока считываются данные из одного банка, другие имеют "передышку" на подготовку, после которой появляется возможность обращения к ним без дополнительного ожидания. При нарушении очередности и повторном обращении к тому же банку выполняется полный цикл обращения к памяти. Чем больше банков имеет память, тем меньше будет последовательных обращений в один и тот же банк.

Процессор чаще всего считывает данные по последовательным адресам, поэтому эффект ускорения работы ЗУ достигается уже при делении памяти всего на два блока, а именно на один с нечетными адресами, другой — с четными. В микросхемах максимального быстродействия число банков ис-

числяется десятками. Банки ЗУ типа MDRAM могут строиться на обычных DRAM без каких-либо схемных изменений.

Дополнительные сведения о системе параметров, характеризующих быстродействие ЗУ

После ознакомления с такими особенностями памяти, как мультиплексирование шины адреса, страничный доступ к данным, многобанковость, наличие в структуре ЗУ внутренних адресующих счетчиков и др. систему параметров следует рассмотреть подробнее, чем это было сделано в начале главы. Для примера остановимся, прежде всего, на параметрах микросхем SDRAM, играющих сейчас важную роль в компьютерных и других системах с высокими требованиями к подсистемам памяти.

- Параметр T_{RAS} называется *временем доступа* и определяется временем, проходящим от момента обращения к памяти (активизации строба RAS) до момента считывания данных. В течение этого интервала проходят полностью все действия, необходимые для первого считывания, и он приблизительно одинаков для всех вариантов современных DRAM (составляет около 50 нс). Время доступа характеризует быстродействие памяти при произвольном доступе к данным.

- *Время цикла.* При блочных передачах (внутри пакета) время доступа не характеризует работу памяти, т. к. время между двумя последовательными обращениями в пределах открытой страницы меньше времени доступа. Это время называют *временем цикла* T_{RC}. В приведенных ранее таймингах 5-3-3-3, 5-2-2-2 и 5-1-1-1 первая цифра соответствует времени доступа, а последующие — временам циклов. Сумма всех цифр цепочки дает время, необходимое для получения всех элементов пакета, длина которого равна 4. Для первого тайминга это 14 тактов, для второго 11, для третьего 8.

- Параметр T_{RCD} (RAS to CAS delay) определяет измеряемую числом *тактов задержку подачи строба CAS относительно подачи строба RAS*. Подача строба CAS открывает доступ к адресованному столбцу матрицы элементов памяти. После этого с задержкой T_{CL} *(CAS latency)* на шине данных появится слово, которое может быть считано процессором, последующие слова, входящие в пакет, появляются в каждом очередном такте. Сумма задержек T_{RCD} и T_{CL} составляет интервал, называемый *латентностью памяти*.

- Время деактивации. В конце процесса обращения к банку памяти должна быть подана команда деактивации банка (эта команда подается за $T_{CL} - 1$ такт перед выдачей последнего слова пакета). *Время деактивации* обозначается как T_{RP} (RAS Precharge).

- *Тайминги.* Три последних параметра часто записывают в виде тайминга T_{CL}-T_{RCD}-T_{RP}. Для микросхем SDRAM этот тайминг составляет цепочку 2-2-2 или 3-3-3. Тайминг $T_{CL} + T_{RCD} + T_{RP}$ является основной характеристикой быстродействия SDRAM, через составляющие его времена выражаются и времена доступа T_{RAS} и цикла T_{RC}. Например, при данном тайминге 3T + 3T + 3T, где T — длительность такта, время доступа определится как 3T + 3T + T = 7T, поскольку до появления данных на шине пройдет время $T_{RCD} + T_{CL}$ = 6T, а между моментом появления данных на шине и их считыванием процессором проходит один такт. Полное время цикла составит при этом 3T + 3T + 3T = 9T. Внутри пакета темп передач соответствует одному такту.

Перед началом работы схемы SDRAM должны быть настроены загрузкой специальных регистров, содержимое которых задает длину пакета, порядок счета адресов в пакете, CAS-латентность и тип операции. Настройка памяти на возможные режимы устанавливается в BIOS, причем для современных SDRAM уже имеются специальные микросхемы памяти настроек, из которых данные могут переноситься в BIOS.

Длина пакета BL (Burst Length) задает число адресов, формируемых в пакете с помощью внутреннего счетчика микросхемы. Длина пакета может быть равной 1, 2, 4, 8 или "полной странице" (Full Page), причем полная страница обычно содержит 256 адресов.

Порядок счета адресов в пакетном цикле может быть последовательным или поочередным (Memory Interleaving).

CAS-латентность может составлять 2 или 3 такта. Тип операции задается как либо нормальный (с пакетными режимами при чтении и записи), либо специальный (с пакетным режимом чтения и одиночными операциями записи).

Структуры типа SDRAM

Хотя переход от базовой структуры DRAM к архитектурам FPM, EDORAM и BEDORAM повысил быстродействие памяти, это оказалось недостаточным для современных компьютеров и графических систем. Архитектуры асинхронных ЗУ типов FPM, EDORAM, BEDORAM для таких (но не для всех) применений уже устарели. В качестве компьютерной памяти с высокой производительностью используются сейчас микросхемы SDRAM (Synchronous DRAM) и RDRAM (Rambus DRAM). Оба вида памяти применяются и с интерфейсом DDR.

В SDRAM синхросигналы памяти тесно увязаны с тактовой частотой процессора (системы), в них используется конвейеризация тракта продвижения информации, применяется многобанковая структура и буферизация (защелкивание) адресов и данных.

В одной из первых работ по тематике SDRAM [67] в 1997 г. были предложены двухбанковые структуры с трехступенчатым конвейером, которые при U_{CC} = 3,3 В и топологической норме 0,5 мкм работали на частоте 125 МГц. Причем площадь кристалла (113,7 мм^2) практически не отличалась от площади кристаллов обычных DRAM той же емкости.

Сущность процесса конвейеризации трактов обработки информации была рассмотрена ранее (см. рис. 4.13 и текст к нему).

В микросхемах SDRAM внешние управляющие сигналы фиксируются фронтами тактовых импульсов системы и их комбинации используются для генерации команд, управляющих процессами в ЗУ. Команда ACT (Active)

связана с выбором строки по соответствующему адресу. Команда RED (Read) определяет адрес первого столбца для чтения данных. Команда PRE (Precharge) связана с этапом предзаряда шин.

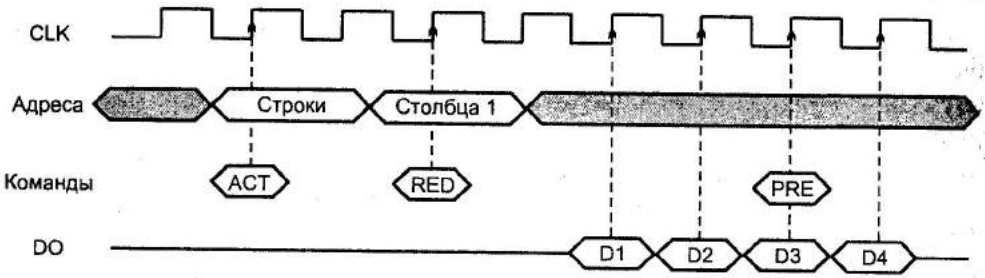
Входными сигналами SDRAM являются следующие:

- CLK — синхросигнал;
- CKE (Clock Enable) — сигнал разрешения синхронизации, отсутствие которого переводит схему в специальные режимы (экономии мощности, приостановки работы и др.);
- $\overline{CE}$ — сигнал, разрешающий декодирование команд (при его отсутствии начатая команда выполняется, новая не декодируется);
- $\overline{RAS}$, $\overline{CAS}$ и $\overline{WE}$ — сигналы, определяющие выполняемую операцию (код команды);
- BS0, BS1 — сигналы, выбирающие банк, к которому относится команда;
- A12—A0 — мультиплексируемая шина адреса, в цикле активизации банка задает адрес строки, в цикле чтения/записи линии шины, кроме линии A10, задают адрес столбца. Линия A10 связана с управлением автопредзарядом и предзарядом шин;
- DQM — сигнал, маскирующий входы данных.

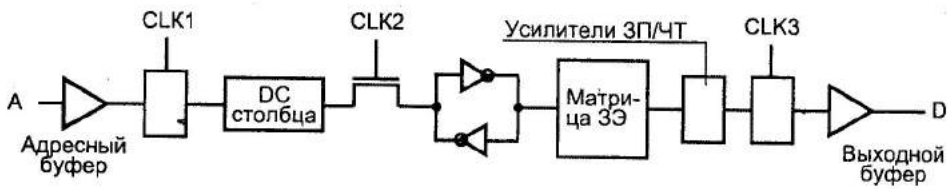
При записи данные первой передачи из состава пакета устанавливаются одновременно с командой WR, остальные передачи осуществляются в последующих тактах. При чтении первое слово после формирования команды появляется с запаздыванием на несколько тактов (Access Latency). Время доступа при этом "обычное", т. е. такое, каким бы оно было в стандартном ЗУ. Адреса следующих слов формируются внутренним счетчиком, и слова появляются в каждом такте (рис. 4.46, *а*). Чтобы ускорить темп появления слов, в пакете организуется трехступенчатый конвейер (рис. 4.46, *б*). Работу конвейера можно определить как параллельное функционирование последовательно активизируемых блоков. В соответствии с управлением тактами каждый сегмент схемы столбца работает в параллель с другими (рис. 4.46, *в*).

В микросхемах SDRAM предусматривают регулировку запаздывания первого доступа с целью приспособления памяти к частотным требованиям системы, а также длины пакета, в котором слова после всего одной команды читаются или записываются в каждом такте.

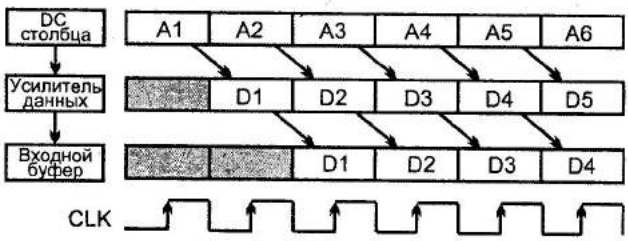
К достоинствам SDRAM, в частности, относится и отсутствие больших проблем по согласованию взаимного положения во времени входных сигналов, что в иных случаях может быть сложным. Здесь же положение облегчается, т. к. входные сигналы фиксируются (защелкиваются) фронтами тактовых импульсов, жестко задающими моменты их появления и исчезновения.



а



б



в

Рис. 4.46. Временные диаграммы (*а*), трехступенчатый конвейер (*б*) и временные соотношения обработки информации (*в*) для синхронных динамических ОЗУ

Структуры типа DDR SDRAM

Память типа DDR SDRAM по своему строению и функционированию совпадает с памятью типа SDRAM во всем за исключением лишь одного момента — характера синхронизации процессов ввода/вывода данных. Ввод/вывод данных тактируется обоими фронтами синхроимпульсов, и это удваивает пропускную способность интерфейса памяти по тракту передачи данных. При работе с тактовыми частотами 100 и 133 МГц микросхемы памяти маркируются уже как DDR200 и DDR266, а для варианта маркировки

с указанием пропускной способности — как DDR PC1600 и DDR PC2100, поскольку при 64-разрядных шинах пропускная способность этих микросхем составляет 1600 Мбайт/с и 2128 Мбайт/с соответственно. Появилась также память DDR PC3200. Во внутренних блоках микросхем DDR SDRAM тактирование ведется обычным способом (по одному фронту), а пропускная способность согласуется с процессами ввода/вывода с помощью увеличения ширины внутренних шин.

Синхронизация по технологии DDR при высоких тактовых частотах предъявляет очень высокие требования к точности временных диаграмм, для поддержания которой принимаются специальные меры:

- синхросигналы генерируются и передаются в дифференциальной форме (одновременно имеются сигналы CLK и $\overline{\text{CLK}}$);
- для синхронизации данных в интерфейс вводится специальный сигнал DQS (DQ Strobe), вырабатываемый самим источником данных (памятью при чтении, контроллером памяти при записи), фронты которого центрируются различным образом (по фронту или центру импульсов) для чтения и записи данных;
- для автоподстройки задержек между синхросигналами в состав микросхем вводятся блоки DLL.

На рис. 4.47 приведена структура DDR SDRAM емкостью 128 Мбит (фирма Samsung). Организация микросхемы может выбираться в вариантах 8M×16, 16M×8, 32M×4, конкретный набор сигналов показан для организации 16M×8, шины с разрядностями, зависящими от выбранного варианта — для вариантов 8M×16 и 16M×8.

Микросхема имеет 4 банка, для рассматриваемого варианта каждый банк имеет емкость 32 Мбит при организации 2M×16. В составе адреса выделяются три части — двухразрядное поле выбора банка и поля адресов строки и столбца, поступающие на соответствующие декодеры через буферные схемы. В буфере адреса строк размещен также счетчик адресов, используемый при регенерации данных путем перебора строк в режиме чтения. Входные данные принимаются регистром со схемой изменения разрядностей данных, который передает на блок банков данные удвоенной разрядности, для согласования пропускной способности областей с DDR и с обычным тактированием, при котором информация воспринимается только по положительным фронтам синхроимпульсов. На выходе блока банков данные, снимаемые с усилителей, подвергаются обратному изменению разрядности в связи с переходом из области обычного тактирования в область DDR. Передачи данных стробируются сигналом DQS. Для повышения качества работы синхросистемы использован блок DLL.

Внешние управляющие сигналы поступают на регистр управления (Timing Register). Их набор определяет выполняемую команду. Содержащаяся в на-

боре сигналов информация используется для выработки определенной последовательности внутренних сигналов управления (их обозначения начинаются с буквы L), обеспечивающих выполнение требуемых действий.

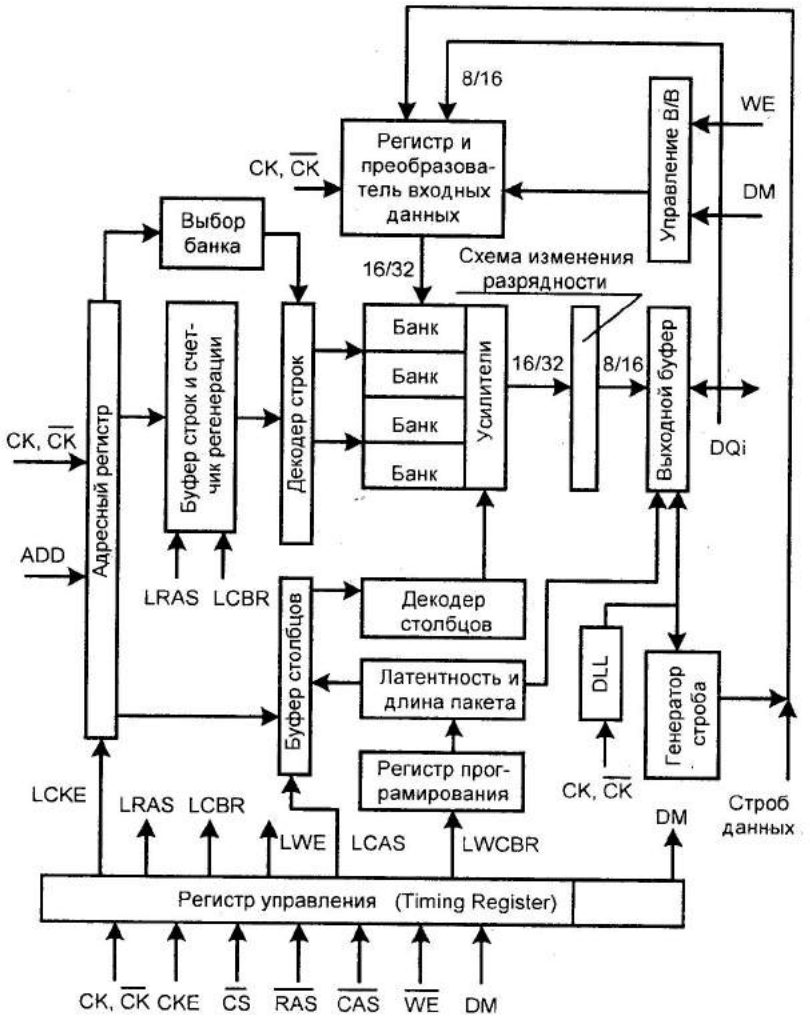


Рис. 4.47. Структура микросхемы DDR SDRAM

Интерфейсные сигналы микросхемы имеют следующий смысл:

- CK и $\overline{CK}$ — дифференциальные синхросигналы. Адреса и сигналы управления воспринимаются блоками микросхемы по положительным фрон-

там CK и, соответственно, отрицательным фронтам $\overline{CK}$. Данные — по обоим фронтам. Внутренние синхросигналы вырабатываются из CK и $\overline{CK}$;

- CKE — (Clock Enable) разрешает или запрещает внутреннее тактирование и функционирование входных и выходных буферов. Снятие синхросигналов обеспечивает проведение операций предзаряда (Precharge), снижения мощности (Power-Down) и саморегенерации (Self Refresh) или некоторых других действий;
- $\overline{CS}$ — разрешает или запрещает декодирование команд, при высоком уровне этого сигнала все команды маскируются, сигнал является частью командного кода;
- $\overline{RAS}$, $\overline{CAS}$, $\overline{WE}$ вместе с $\overline{CS}$ определяют вводимую команду;
- DM — маскирует входные сигналы при записи данных, при чтении его состояние безразлично;
- ADD ($A_{11}A_0$) — адресные входы, задающие адрес строки при командах ACTIVE и адрес столбца и бит предзаряда при командах READ/WRITE, чтобы выбрать ячейку памяти в матрице соответствующего банка. Один из разрядов адреса используется при командах предзаряда для определения того, относится эта команда к одному или всем банкам. При выполнении команды Mode Register Set на адресные входы поступает код операции;
- DQ — шина данных для их ввода/вывода;
- DQS — строб данных, выходной сигнал при чтении данных и входной при их записи. При чтении он согласован с данными по фронтам, при записи отмечает центры битовых интервалов. Служит для захвата записываемых данных.

Кроме сигнальных выводов микросхемы имеют ряд выводов для напряжений общего питания 2,5 В и схемной земли (по три вывода), напряжений питания 2,5 В и схемной земли цепей передачи данных (по пять выводов) и опорного напряжения, соответствующего стандарту SSTL-2 (составляет половину напряжения питания цепей передачи данных).

В состав команд микросхемы кроме указанных ранее Active, Precharge, Read, Write входят также команды загрузки регистра управления (Mode Register Set), регенерации (Refresh), введения режима понижения мощности и др., причем, как правило, в нескольких модификациях.

Рабочие частоты микросхем семейства лежат в диапазоне от 100 до 166 МГц. Латентность при чтении — 2; 2,5 периода тактовых сигналов, длины пакетов 2, 4 или 8, типы пакетов — последовательный или с перемежением адресов (Interleave).

Структуры типа RDRAM

Микросхемы RDRAM (Rambus DRAM) названы по имени фирмы-разработчика — Rambus. Эти микросхемы обладают наивысшей среди динамических ЗУ производительностью, превосходящей, хотя и не радикально, производительность своих ближайших конкурентов — синхронных динамических ЗУ SDRAM. В то же время работа на очень высоких частотах удорожает RDRAM, и по стоимости они заметно проигрывают микросхемам SDRAM.

Ядро микросхем RDRAM выполнено на той же схемотехнической основе, что и ядра других современных динамических ЗУ, запоминающие элементы и организация матриц RDRAM и их предшественников принципиально идентичны. Отсюда можно заключить, что причины высокого быстродействия RDRAM кроются в интерфейсе. Действительно, в RDRAM реализованы как сверхскоростные линии связи, так и специальная структура объединения нескольких микросхем в подсистему памяти. Микросхемы RDRAM отличаются малоразрядностью шин ввода/вывода информации (в первых вариантах использовалась 8-разрядная шина, затем применили и 16-разрядную).

В сужении шин проявляется общая и важная тенденция, характерная для быстродействующих трактов передачи сигналов не только в памяти, но и в цифровой технике вообще. Сужение шин является ответом на противоречие между параметрами частоты передач и разрядности шины. При большой разрядности шины росту частоты передач особо ощутимо препятствует возникновение помех. Связь уровня помех с разрядностью шины видна хотя бы из того, что в линиях шины возможно одновременное изменение всех сигналов, и при широкой шине одновременное переключение многих элементов создает большие помехи. Если сужение шины в N раз позволит увеличить частоту передачи сигналов более чем в N раз, то это приведет к увеличению пропускной способности шины. Такие ситуации как раз и наблюдаются в современной цифровой технике.

Микросхемы RDRAM имеют высокую производительность, но, одновременно, и высокую стоимость. Их область применения — компьютеры с процессорами Pentium 4 или более производительными. Применение памяти типа RDRAM активно поддерживается фирмой Intel, однако в силу меньшей стоимости микросхемы SDRAM имеют сейчас более широкий круг потребителей.

Существуют три варианта микросхем RDRAM (Base, Concurrent, Direct), из которых последний наиболее интересен своей максимальной производительностью и минимальной латентностью. Предшественники варианта Direct RDRAM имели большую латентность. Интерфейс у микросхем RDRAM с байт-последовательной передачей данных вначале имел всего 13 сигнальных линий, что значительно меньше, чем у традиционных микросхем памяти. В последующем число линий интерфейса несколько возросло,

но осталось намного меньшим, чем у SDRAM и других ОЗУ. В интерфейсе нет специализированных адресных линий. Вместо обычной адресации с применением широких адресных шин по интерфейсу посылаются пакеты, включающие в себя сигналы управления и адреса. Вначале посылается пакет запросов, на который память отвечает пакетом подтверждения, после чего идет пакет данных. Из-за такого процесса первый доступ к данным оказывается сильно запаздывающим. В первой разработке при темпе передачи байтов 500 МГц (2 нс на байт) запаздывание составляло 128 нс. Поэтому при чтении отдельных слов RDRAM была совершенно неэффективна. Средняя частота передачи байтов зависела от длины пакета данных. При обмене пакетами по 256 байт средняя частота будет 400 МГц (к 2 нс добавляется 0,5 нс на байт), при пакетах по 64 байта — 250 МГц и т. д.

Для варианта Direct RDRAM с уменьшенной латентностью было принято наименование DRDRAM (Direct Rambus DRAM), однако со временем букву D в этой аббревиатуре стали опускать, понимая под термином RDRAM именно вариант DRDRAM, что и принято далее.

В подсистему памяти RDRAM входят:

- собственно микросхемы;
- канал;
- контроллер памяти.

Микросхема RDRAM в упрощенном виде показана на рис. 4.48 (вариант фирмы Samsung).

По верхнему краю рисунка размещены сигнальные выводы микросхемы. Две линии записи/чтения последовательных данных SIO_{1-0} используются для ввода/вывода данных в регистры управления. Ввод/вывод в регистры управляется сигналами двух других линий SCK (Serial Clock) и CMD (Command), функции которых состоят не только в управлении передачами данных — они используются также и для управления потребляемой микросхемой мощностью. Два буфера (1 и 2) принимают дифференциальные синхросигналы CTM/CTMN (Clock To Master) и CFM/CFMN (Clock From Master). Смысл направлений To Master и From Master можно уяснить при рассмотрении структуры канала, в который устанавливаются несколько микросхем (см. далее). Из дифференциальных синхросигналов скоростной передачи по внешним линиям формируются однополюсные синхросигналы TCLK и RCLK для внутренней синхронизации процессов передач и приема данных. Три линии управления доступом к строке $ROW_{2..0}$ и пять линий управления доступом к столбцу $COL_{4..0}$ образуют шину для передачи в микросхему пакетов управляющей и адресной информации, необходимой для осуществления транзакций. Две девятиразрядные шины служат для приема/выдачи в канал байтов A и B (в байте восемь информационных разрядов плюс контрольный, который можно использовать и как информационный).

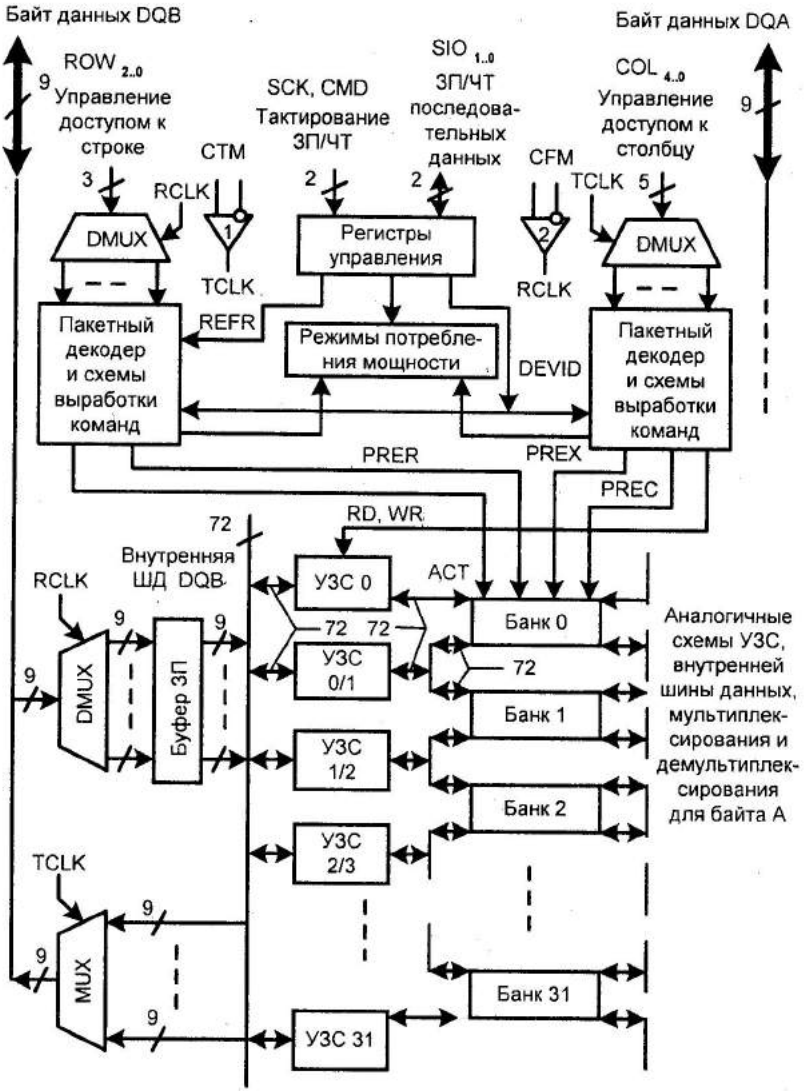


Рис. 4.48. Упрощенная схема RDRAM

Информация, получаемая от шин $ROW_{2..0}$ и $COL_{4..0}$, демультиплексируется, сформированные при этом коды вместе с кодами DEVID и REFR, поступающими от регистров управления, являются основой для формирования команд ACT, PRER, PREX, PREC и RD, WR. Пятиразрядный код DEVID определяет адрес данной микросхемы в канале RDRAM, код REFR связан с выполнением операций регенерации данных в памяти (отслеживает по-

следнюю регенерированную строку). Команды организуют работу ядра микросхемы.

Для упрощения рис. 4.48 на нем изображена лишь часть ядра, а именно банки и левая сторона, обслуживающая передачи байта B в ядро и из него. Для обслуживания передач байта A справа от банков имеются схемы, зеркально повторяющие схемы левой стороны ядра. Эти-то схемы и не показаны на рисунке, хотя, естественно, при описании работы микросхемы их наличие подразумевается.

В ядро микросхемы входят 32 банка памяти, 34 усилителя записи/считывания УЗС (УЗС состоит из двух частей — слева от банка и справа от него) и две 72-разрядные внутренние шины данных для байтов A и B. Банки имеют информационные емкости по 1 Мбайт, общая емкость памяти составляет 32 Мбайт. Каждый банк имеет 512 строк. Строка состоит из 16-байтных наименьших адресуемых единиц информации (Dualocts), число которых в строке составляет 128.

Каждый УЗС содержит 1 Кбайт быстродействующей регистровой памяти — по 512 байт для обеих частей УЗС — левой и правой, соединенных с внутренними шинами данных для правого байта (шина DQA) и левого байта (шина DQB). Поскольку емкость строки равна 2 Кбайт, УЗС может принимать данные половины строки. В УЗС можно загружать любую из 1024 полустрок банка, связанного с этой строкой. Равным образом из УЗС можно передавать данные в любую полустроку ассоциированного со строкой банка. Почти все УЗС (кроме 0, 15, 16 и 31-го) находятся в совместном пользовании двух соседних банков, что ограничивает одновременный доступ в смежные банки.

Команда ACT (от Activate) вызывает загрузку одной из 512 строк выбранного банка памяти в усилители записи-считывания, связанные с этим банком. Команда PRER (от Precharge, что означает "предзаряд") заставляет выбранный банк освободить связанную с ним пару УЗС, позволяя активизироваться другой строке этого банка или смежным банкам. Команды PREC и PREX реализуют другие механизмы операции предзаряда. Команда RD вызывает передачу на выводы DQA/DQB из УЗС одной из минимально адресуемых единиц информации (Dualocts). Команда WR вызывает загрузку полученной из канала по шинам DQA/DQB минимально адресуемой единицы информации в буфер записи (в буфере имеется место и для некоторых сведений об адресах банка и столбца. Наличие буфера записи уменьшает задержку при реверсе внутренней шины данных DQA/DQB.

Взаимодействие ядра микросхемы с быстродействующими шинами DQA/DQB происходит следующим образом. Ядро работает на частоте в 1/8 от частоты шин DQA/DQB. При этом данные в УЗС или из них поступают порциями по 72 разряда на каждую из двух внутренних шин данных или от них (по 8 девятиразрядных байтов). В интервале между передачами много-

разрядных данных они с помощью мультиплексоров разбиваются на 8 байтов, которые на частоте, в восемь раз более высокой, успевают последовательно пройти по малоразрядным шинам DQA/DQB. При другом направлении передач с помощью демультиплексоров из байтов собираются 72-разрядные данные, которыми на частоте работы ядра УЗС обмениваются с банками.

К архитектурным новшествам, использованным в микросхемах RDRAM, относится и *специальный канал Rambus Channel*, который объединяет до 32 микросхем памяти. Разработанный фирмой интерфейс RSL (Rambus Signaling Level) с амплитудой сигналов 0,8 В допускает передачи со скоростями 600 и 800 МГц в рамках освоенных технологий изготовления системных плат. Канал содержит 30 основных линий с интерфейсом RSL и 4 вспомогательных линии с КМОП-интерфейсом. Вспомогательные линии используются для инициализации микросхем.

В микросхемах RDRAM, как и в SDRAM, используется синхронизация ввода/вывода обоими фронтами тактовых импульсов (технология DDR), конвейеризация, многобанковость и другие методы повышения производительности. В сущности, в микросхемах RDRAM сочетаются "ядро", сходное со схемами других DRAM, и специфичный интерфейсный блок, позволяющий передавать в это ядро информацию с очень высокой производительностью.

RDRAM хорошо подходит для графических и мультимедийных приложений с типичным для них процессом — быстрой выдачей длинной последовательности слов для формирования изображения на экране или сходных с этим задач. Современные микросхемы RDRAM работают преимущественно на тактовой частоте 400 МГц, что при использовании DDR соответствует "эффективной частоте" 800 МГц. Для шины с 16 разрядами пропускная способность составляет 1600 Мбайт/с. Современные контроллеры RDRAM могут обслуживать до 4 шин. Таким образом, общая пропускная способность модуля может быть доведена до 6,4 Гбайт/с (сейчас в рабочих станциях используются варианты с пропускной способностью 3,2 Гбайт/с). Хотя в сравнении с первыми образцами современные RDRAM имеют сниженную латентность, их эффективность при произвольном доступе к малым объемам данных также значительно снижается.

В области высокопроизводительных динамических ЗУ в последние годы развернулась конкуренция между DDR SDRAM и DDR RDRAM. Пока по распространенности доминируют SDRAM. Для некомпьютерных применений, требующих больших емкостей памяти, эта ситуация может сохраниться на многие годы. В компьютерных схемах положение менее ясно и RDRAM представляется сильной альтернативой (на этот вид памяти ориентированы новые чипсеты фирмы Intel, хотя эта фирма не упускает из виду и разработки с применением SDRAM).

Структура типа CDRAM

В структурах CDRAM (Cached DRAM, кэшированная DRAM) на одном кристалле с DRAM размещена статическая кэш-память. При этом кэш обеспечивает быстрый обмен с процессором, если информация находится в кэше, а также быстрое обновление своего содержимого. Последняя возможность связана с тем, что размещение кэша на одном кристалле с DRAM делает связи между ними внутренними (реализуемыми внутри кристалла), а в этом случае разрядность шин может быть большой и обмен может производиться большими блоками данных. Например, в CDRAM фирмы Ramtron применена 2048-разрядная шина для обновления содержимого кэша.

Как синоним обозначения CDRAM иногда используется обозначение EDRAM (Enhanced DRAM). Кэширование, как и всегда, эффективно при выполнении программ, для которых промахи относительно кэша достаточно редки.

Динамические ЗУ с фрагментированной (блочной) базовой структурой

Динамические ЗУ с фрагментированной структурой (как и микросхемы RLDRAM и FCRAM, рассматриваемые далее) в классификации ЗУ отнесены к структурам, которые с точки зрения повышения быстродействия эффективны даже при произвольном доступе к ячейкам памяти. Добиться высокого быстродействия при произвольном доступе к данным труднее, чем обеспечить высокое быстродействие в страничном режиме (при кучности последовательно поступающих адресов).

Так как в компьютерных приложениях кучность адресов проявляется достаточно четко, для них целесообразно применять структуры SDRAM, DDR SDRAM, RDRAM. Для алгоритмов, реализуемых в сетевых приложениях, и многих других задачах мультимедийного типа важно быстродействие в условиях произвольного доступа. Динамические ЗУ с ускоренным произвольным доступом разработаны в самое последнее время и образуют как бы параллельную линию развития микросхем памяти — линию "некомпьютерных" ЗУ.

В последнее время с целью повышения быстродействия DRAM при произвольном доступе к данным единую структуру базового типа, рассмотренную в *§ 4.9*, стали *разделять на несколько частей* — фрагментировать. В DRAM с фрагментированной структурой (Embedded DRAM) длительность цикла произвольного доступа к памяти по сравнению с этим циклом в DRAM базовой структуры существенно снижается. Дело в том, что базовая структура ориентирована на минимизацию площади кристалла и стоимости микросхем памяти. Вся матрица запоминающих элементов делается плотно упакованной. Число ячеек, подключаемых к каждой разрядной линии в современных DRAM базовой структуры, составляет обычно 256 или 512. При

существующих отношениях емкостей запоминающего конденсатора и емкостей линий такое число еще приемлемо и может обеспечить амплитуды сигналов чтения, достаточные для надежного восприятия. Число словарных линий в обычных DRAM составляет 2048—4096, что еще приемлемо с точки зрения ограничения нагрузки на источники сигналов управления словарными линиями (WL-драйверы).

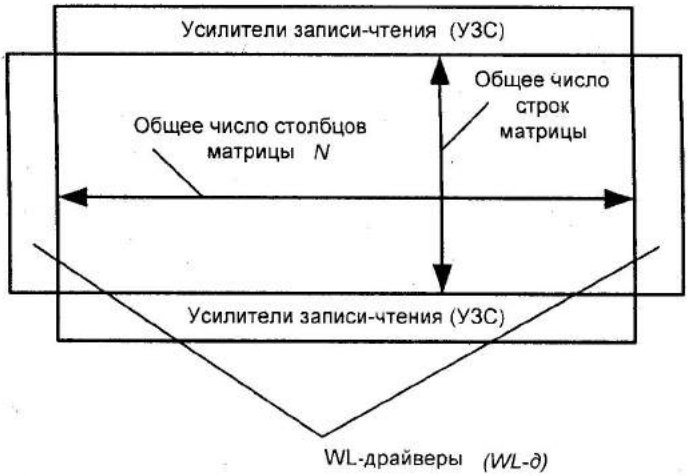
В схемах DRAM фрагментированной структуры *словарные и разрядные линии укорачиваются*, при этом матрица разбивается на части, значительно уменьшаются паразитные емкости и сопротивления укороченных линий, а, следовательно, и задержки переключений, происходящих в работающей схеме. Таким способом удается существенно повысить быстродействие DRAM, не требуя при этом какой-либо определенной последовательности адресов обращения к ЗУ. Платой за это является усложнение схемы ЗУ и соответствующее ему увеличение стоимости.

На рис. 4.49, *а* схематично показана структура DRAM, оптимизированная по критериям плотности упаковки и стоимости, т. е. обычная. На рис. 4.49, *б* изображена фрагментированная структура, обладающая повышенным быстродействием. В схеме фрагментированной DRAM сигналы управления словарными линиями WL вырабатываются несколькими дешифраторами-драйверами, нагрузками для которых служат укороченные линии, а введение дополнительных усилителей записи-чтения приводит к укорачиванию разрядных линий. Эти факторы определяют ускорение работы DRAM. Усложнение схемы выражается в увеличении числа усилителей записи-чтения и декодеров-драйверов для управления "отрезками" словарных линий. Если в базовой структуре DRAM непосредственно под запоминающие элементы удается отвести около 50% площади кристалла, то для фрагментированных DRAM этот процент снижается приблизительно до 35.

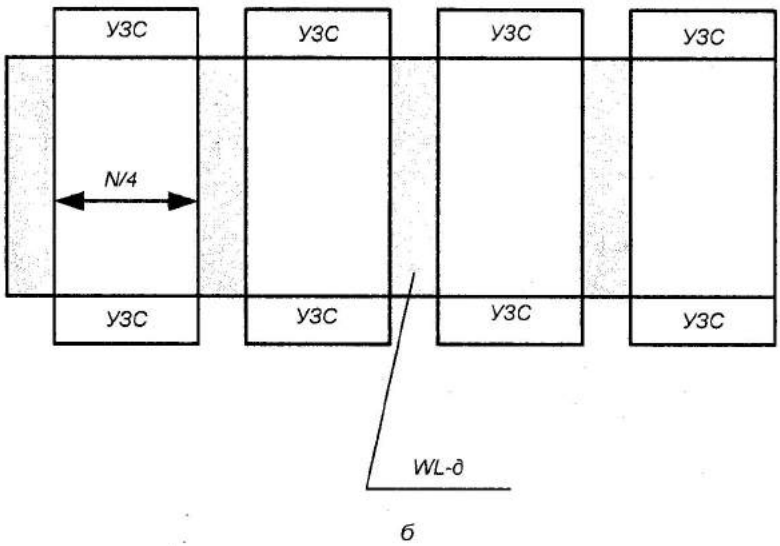
Уменьшение длины линии передачи сигнала вдвое соответствует снижению числа RC-звеньев в ее схеме замещения (модели) также в два раза. При этом время распространения сигнала по линии уменьшается в 4 раза, т. к. зависит от числа звеньев схемы по квадратичному закону. Укорачивание разрядных линий ведет также к увеличению амплитуды считываемых сигналов, которая линейно зависит от отношения емкости запоминающего конденсатора к емкости разрядной линии. Рост амплитуды считываемых сигналов в свою очередь способствует ускорению работы усилителей считывания.

Для дополнительного увеличения быстродействия фрагментированных DRAM используются, впрочем, как и в других структурах, конвейерная организация обработки сигналов.

В настоящее время реализованы фрагментированные конвейеризованные DRAM с произвольным доступом, уступающие по быстродействию лишь лучшим статическим ЗУ. В то же время уровень интеграции фрагментированных DRAM более чем в 5 раз превышает возможности статических ЗУ.



а



б

Рис. 4.49. Схематичные изображения базовой (*а*) и фрагментированной (*б*) структур динамического ЗУ

Структуры типа RLDRAM

Аббревиатура RLDRAM расшифровывается как Reduced Latency DRAM, что означает DRAM с уменьшенной длительностью полного цикла обращения к

памяти, т. е. с ускорением работы при произвольном доступе. Быстрый произвольный доступ в микросхемах RLDRAM во многом достигнут благодаря отказу от последовательных передач в память полуадресов по стробам RAS и CAS и *передаче сразу всего адреса одним тактирующим фронтом*. Этого ранее избегали для упрощения и удешевления ЗУ, но в структуре RLDRAM реализован другой выбор — усложнение ради ускорения. Получение памятью всего адреса сразу позволяет организовать процессы чтения/записи с более выраженным параллелизмом действий, чем при последовательном поступлении полуадресов, снизив тем самым время доступа T_{RAC}.

Разработчик RLDRAM, фирма Infinion (бывшая ранее подразделением фирмы Siemens) получила для них время произвольного доступа при чтении 25 нс, что приблизительно вдвое меньше, чем у быстродействующей компьютерной памяти типов DDR SDRAM и RDRAM. Информационная емкость кристаллов памяти составляла 256 Мбит, разрядности — 16 и 32. В структуре применена многобанковость (8 или 16 банков). Напряжение питания 1,8 В. Число выводов корпуса 144 (шариковые выводы тонкого корпуса BGA, Ball Grid Array).

Наличие многих банков дает возможность ускорять темп передачи данных в пакетных режимах. Таким образом, в ЗУ типа RLDRAM не только уменьшена латентность, но и сохранено повышение быстродействия ЗУ при кучности адресов последовательных обращений. По-видимому, этот вариант ЗУ будет успешно конкурировать с представителями как "компьютерных", так и "некомпьютерных" микросхем памяти.

Структуры типа FCRAM

Под этим названием, сокращенно обозначающим Fast Cycle RAM, т. е. ОЗУ с быстрым циклом, выпускается один из новых вариантов динамических ЗУ повышенного быстродействия (несмотря на отсутствие в обозначении буквы D, обычной для динамической памяти). Фирмы Fujitsu и Toshiba, разработавшие микросхемы FCRAM, благодаря ряду приемов получили ЗУ с информационными емкостями, характерными для DRAM и временами произвольного доступа, конкурирующими с возможностями быстродействующих SRAM. Микросхемы FCRAM имеют много общего с микросхемами SDRAM. Сравнивая варианты DDR SDRAM и DDR FCRAM, можно отметить такие отличия этих вариантов, как *сегментация ядра микросхемы*, разделенного на подматрицы, и введение во внешние относительно ядра схемы дополнительной логики для реализации конвейерной структуры, позволяющей обрабатывать одновременно до трех адресов. Поскольку основной целью создания FCRAM было ускорение произвольного доступа, режим страничного доступа этими микросхемами не поддерживается. После обращения строка автоматически закрывается и выполняется предзаряд в данном банке. Поддерживается многобанковая работа (Bank-Interleave Mode). Коэффици-

ент использования шин оценивается как 80%, что превышает соответствующую цифру для DDR SDRAM. В результате была получена длительность полного цикла доступа около 25 нс, что, как и у микросхем типа RLDRAM, приблизительно вдвое меньше, чем у микросхем DDR SDRAM и DRDRAM. Повышение быстродействия микросхем FCRAM сопровождается их экономичностью по потребляемой мощности (благодаря активизации только выбранного слова, а не всей строки, так называемому автопредзаряду и т. п.). Из четырех кристаллов FCRAM строится модуль памяти с 64-разрядной шиной, который при тактовой частоте 200 МГц имеет полосу пропускания 3,2 Гбайт/с. Таким образом, структуры FCRAM выступают в качестве еще одного претендента на роль быстродействующих ЗУ, пригодных и для некомпьютерных применений, в частности для портативной аппаратуры с автономным питанием (мобильных телефонов, PDA, плееров и др.).

Параметры динамических ЗУ

Основные параметры современных динамических ЗУ приведены в табл. 4.2.

Таблица 4.2

Тип ЗУ	Тактовая частота, МГц	Емкость ЗУ, варианты его разрядности	Максимальная пропускная способность, Гбайт/с	Напряжение питания ядра, В	Задержка T_{RAS}, нс
SDRAM	100, 133, 167	64 Мбит, 8/16/32; 128 Мбит, 4/8/16; 256 Мбит, 4/8/16	1,064 (133 МГц)	3,3	43,5
DDR SDRAM	100, 133, 167, 183, 200 2X	64 Мбит, 16/32; 128 Мбит, 4/8/16/32; 256 Мбит, 4/8/16	2,12	2,5	22,5
RDRAM	400, 533 2X	128/144 Мбит, 18; 256/288 Мбит, 18	1,6 3,2 (для двух каналов и 32-разрядных шин)	2,5	45 (400 МГц)
RLDRAM	300 2X	256 Мбит, 16/32	4,6	1,8	22,9 (300 МГц)

Таблица 4.2 (окончание)

Тип ЗУ	Тактовая частота, МГц	Емкость ЗУ, варианты его разрядности	Максимальная пропускная способность, Гбайт/с	Напряжение питания ядра, В	Задержка T_{RAS}, нс
FCRAM	154, 200 2X	256 Мбит, 8/16	2,46	2,5	22,5

В таблице пропускная способность указана для 64-разрядных шин; задержка T_{RAS} — время доступа от появления команды чтения до появления данных; в величине задержки для микросхем RDRAM не учтено время чтения команды (около 10 нс).

В ближайшее время ожидается вхождение в состав массовой продукции микросхем SDRAM емкостью 1 Гбит, в том числе с интерфейсом DDR-2 со скоростями 600—800М передач в секунду (фирмы Samsung, Infinion и др.).

О конструктивных модулях памяти

Для создания ЗУ нужной емкости микросхемы памяти объединяются в конструктивные модули: SIMM — Single In Line Memory Module или DIMM — Dual In Line Memory Module. Имеются уже и более новые типы модулей. Модули представляют собою небольшие текстолитовые платы с печатным монтажом, имеющие для подключения к разъемам системной (материнской) платы "ножевой" печатный разъем. Число контактов обычно 72 и 168, при разрядностях шин 32 и 64 соответственно. Платы модулей многослойные, чаще всего четырехслойные.

Модули SIMM с однорядными контактами уже устарели. У модулей DIMM в отличие от модулей SIMM противолежащие контакты на разных сторонах платы электрически не связаны друг с другом, что удваивает число выводов модуля. Микросхемы памяти у этих модулей также размещаются с обеих сторон платы. Модули подразделяются по напряжению питания и нагрузочной способности (буферизованные и не буферизованные). Варианты с большой нагрузочной способностью свойственны модулям памяти большого объема, т. к. в этом случае велики емкостные нагрузки линий передачи сигналов и, соответственно, требуются большие токи для быстрого изменения потенциалов шин при изменении сигналов.

В маркировках известных модулей синхронной памяти PC100 и PC133 цифры соответствуют частотам системной шины, выраженным в мегагерцах. В дальнейшем цифрами стали указывать пропускную способность канала памяти, и те же модули получили маркировки PC800 и PC1064, где цифры соответствуют размерностям мегабайт в секунду, поскольку шины у данных

модулей 64-разрядные (8-байтные). При этом имеется в виду максимальная пропускная способность (внутри пакета).

Модули типа RIMM для микросхем RDRAM имеют свои особенности. В частности, для обеспечения работы на особо высоких частотах в них применяются дорогостоящие материалы. Высокая стоимость модулей RIMM является одним из факторов, сдерживающих распространение памяти RDRAM.

§ 4.12. Перспективные запоминающие устройства (FRAM, PFRAM, MRAM, OUM)

Успехи создания ЗУ на основе полупроводниковой технологии не снимают проблемы дальнейшего совершенствования микросхем памяти. Чтобы приблизиться к идеалу, желательно к таким свойствам ЗУ, как высокая емкость, быстродействие и малая потребляемая мощность, добавить и энергонезависимость, которой современные ОЗУ не обладают. Если к такому комплексу качеств прибавить и низкую стоимость, то получатся ЗУ, близкие к идеалу. Пути приближения к идеалу включают в себя попытки использования нескольких новых для технологии ЗУ физических явлений — ферроэлектрических, магниторезистивных, связанных с изменением фазовых состояний материалов и др.

ЗУ типа FRAM (ферроэлектрические)

В ферроэлектрических FRAM (Ferroelectric RAM) основой запоминающего элемента служит материал, в кристаллической структуре которого имеется *бистабильный атом*. Занимая одно из двух возможных пространственных положений ("верхнее" или "нижнее"), этот атом создает в ферроэлектрическом материале внутренние диполи того или иного знака (спонтанная поляризация).

С помощью электрического поля можно придать внутреннему диполю тот или иной знак. Под воздействием внешнего электрического поля и при температуре не выше определенной (связанной с точкой Кюри) *материал поляризуется*, диполи выстраиваются упорядоченно и состояние материала может отображать двоичные данные 0 и 1. Зависимость поляризации P от напряжения U имеет петлю гистерезиса, показанную на рис. 4.50, *а*. Через U_C на рисунке обозначены коэрцитивные напряжения, через P_R — остаточные поляризации, которые сохраняются после снятия электрических полей.

Если бы вместо ферроэлектрического конденсатора был включен обычный, соединенный не с Plate-линией, а с общей точкой схемы (схемной землей), то получился бы запоминающий элемент обычного динамического ЗУ, и подключение конденсатора через ключевой транзистор к линии записи/считывания ЛЗС позволяло бы считывать хранимую элементом инфор-

мацию (ЛВ — линия выборки), т. к. в зависимости от заряженности или разряженности конденсатора при его подключении по-разному изменялось бы напряжение на линии ЛЗС. Здесь же нужно выявить не наличие или отсутствие заряда конденсатора, а знак поляризации запоминающего элемента. Простым подключением ферроэлектрического конденсатора к линии ЛЗС этого не определить. Поэтому после отпирания транзистора выборки на Plate-линию подается импульс длительностью около 10 нс. Если этот импульс вызовет переполяризацию элемента, то через него пройдет большой ток, который сможет ощутимо изменить напряжение на линии ЛЗС (на ее емкости, изображенной на рис. 4.50, *б* штриховыми линиями). Если же знак поляризации был иным и переполяризации элемента не будет, то ток через него будет малым и не сможет заметно повлиять на потенциал линии ЛЗС.

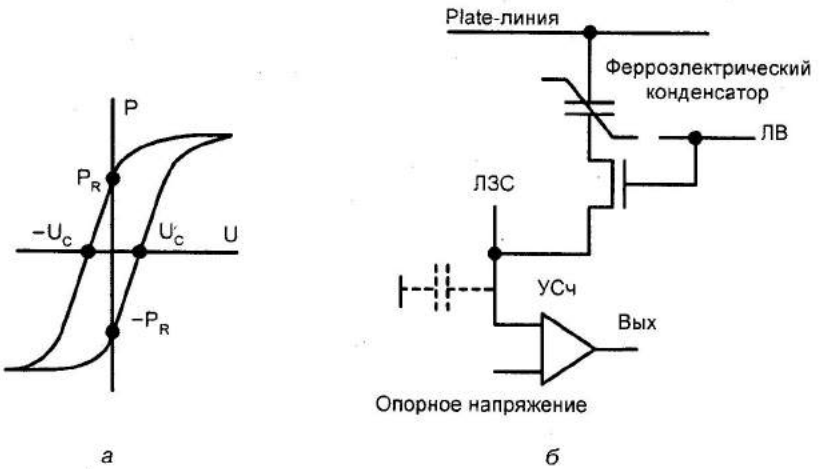


Рис. 4.50. Петля гистерезиса ферроэлектрического материала (*а*) и схема запоминающего элемента FRAM (*б*)

После пропускания импульса от Plate-линии подается питание на усилитель считывания УСч, логическое состояние которого определится тем, смогла ли линия ЛЗС зарядиться выше или ниже опорного напряжения, т. е. в конечном счете знаком поляризации запоминающего элемента. При этом выходной сигнал усилителя фиксируется (защелкивается) для обратной подачи на разрядную линию и возвращения ферроэлектрического конденсатора в его первоначальное состояние после проведенной разрушающей операции чтения. Такая обратная перезапись информации занимает время около 10—20 нс и сохраняет считанные данные. Процессы в ЗУ синхронизированы с фронтами управляющих импульсов.

Рассмотренный запоминающий элемент называют элементом типа 1T/1C, т. к. в его схеме используются один транзистор и один ферроэлектрический конденсатор. Существуют также элементы типа 2T/2C, похожие на сдвоенный элемент 1T/1C. В таких элементах две ячейки 1T/1C программируются в противоположных направлениях и в элементе имеются две разрядные линии с взаимноинверсными сигналами. Используется дифференциальный канал для восприятия сигналов, а это повышает помехоустойчивость ЗУ и улучшает также некоторые другие параметры.

Достоинства FRAM: *быстрые запись и чтение, практически неограниченное число циклов чтения/записи, малые напряжения питания и потребляемая мощность, компактность запоминающего элемента (площадь его соизмерима с площадью обычного запоминающего элемента DRAM), высокая радиационная стойкость, энергонезависимость.*

Такой набор достоинств позволяет FRAM выступить в роли конкурента как по отношению к динамическим ОЗУ, не обладающим энергонезависимостью, так и по отношению к EEPROM и Flash, не обеспечивающим быструю запись данных. В настоящее время ЗУ типа FRAM уже выпускаются рядом фирм (Ramtron International, Samsung, NEC и др.).

ЗУ типа PFRAM (полимерно-ферроэлектрические)

ЗУ типа PFRAM (Polimeric Ferroelectric RAM) — разновидность ферроэлектрических ЗУ. Они построены на основе ферроэлектрических материалов — пленок с двумя стабильными состояниями поляризации, полученных около 10 лет назад шведской фирмой Opticom. Над применением таких пленок в схемах ЗУ работает фирма Intel совместно с дочерней компанией указанной шведской фирмы.

В пленке, толщина которой меньше 0,1 мкм, образуются ориентированные диполи, которые служат запоминающими элементами, хранящими различные двоичные данные при изменении знака поляризации. Расположенные в полимерной пленке запоминающие элементы размещаются между двумя взаимно перпендикулярными металлическими дорожками, на которые подаются определенные напряжения (рис. 4.51). Индивидуальные биты активизируются возбуждением словарной и разрядной линий, на пересечении которых они находятся. Наличие созданных диполей себя проявляет, и набор чувствительных усилителей в разрядных линиях воспринимает значения битов данных.

ЗУ типа PFRAM отличаются крайне простыми запоминающими элементами — в них вообще нет транзисторов, не говоря уже о более сложных элементах. Для изготовления матрицы запоминающих элементов нужны лишь три шаблона (современные ЗУ требуют разработки и использования 20—30 шаб-

лонов). Запоминающие элементы очень компактны, их площадь приблизительно в 20 раз меньше, чем у флэш-памяти. Показанный на рисунке "сэндвич" может служить частью многослойной конструкции, в которой подобные "сэндвичи" собираются в "этажерку" (стек). В таких стеках можно хранить очень большой объем данных. Технология изготовления ЗУ проста и хорошо сочетается со стандартными процессами изготовления интегральных схем. Обеспечивается очень малая стоимость/бит. При хранении данных не требуются какие-либо затраты мощности. Считывание может быть как разрушающим, так и неразрушающим.

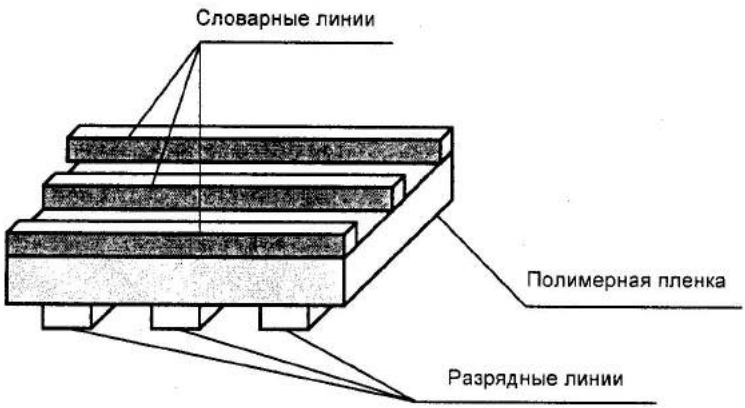


Рис. 4.51. Схематическая конструкция полимерно-ферроэлектрического ЗУ

Процессы записи и чтения идентичны по быстродействию — и тот, и другой занимают приблизительно по 50 мкс. Эта цифра исключает какой-либо разговор о быстродействии, она на три порядка превышает времена доступа обычных DRAM. Поэтому PFRAM перспективны не в качестве ОЗУ, а для замены дисковой памяти. Подсчитано, что плата PFRAM-памяти размером с кредитную карту по информационной емкости будет эквивалентна 400 тысячам CD.

Имеются сообщения о возможности существенного повышения быстродействия PFRAM при новых методах обработки полимерной пленки. Предполагается, что массовое производство PFRAM начнется приблизительно через 5 лет.

ЗУ типа MRAM (магниторезистивные)

В ЗУ типа MRAM (Magnetoresistive RAM) битам двоичных данных соответствуют участки намагниченности, создаваемые в материалах, обладающих остаточной намагниченностью. Участки с остаточной намагниченностью

образуют "микромагнитики", положение полюсов которых задается при записи информации. Магнитные ЗУ обладают естественной энергонезависимостью. Магнитные поля отдельных магнитиков обнаруживаются расположенными у их краев элементами с магниторезистивными свойствами, электрическое сопротивление которых зависит от магнитного поля, окружающего эти элементы. Чтение при этом не является разрушающим.

Для создания MRAM можно использовать два типа эффектов — так называемый *гигантский магниторезистивный эффект* (Giant Magnetic-resistive Effect) или *туннелирование носителей заряда* через тонкий слой, управляемое магнитным полем (в ЗУ типа MTJ, Magnetic Tunnel Junction). В последнее время почти все разработчики предпочли второе направление, на котором мы и остановимся.

Конструкция запоминающего элемента типа MTJ включает в себя два ферромагнитных слоя, разделенных тонким слоем диэлектрика, действующим как туннельный барьер (рис. 4.52). Электрическое сопротивление такого элемента зависит от создаваемого в тонком слое магнитного поля. Поле зависит от окружающих диэлектрик двух ферромагнитных слоев — если их магнитные моменты параллельны, то сопротивление элемента MTJ минимально, если антипараллельны, то максимально. Чтение осуществляется измерением туннельного тока между магнитными слоями. В этой конструкции разница между сопротивлениями элементов, находящихся в состояниях 0 и 1, достигает 50%.

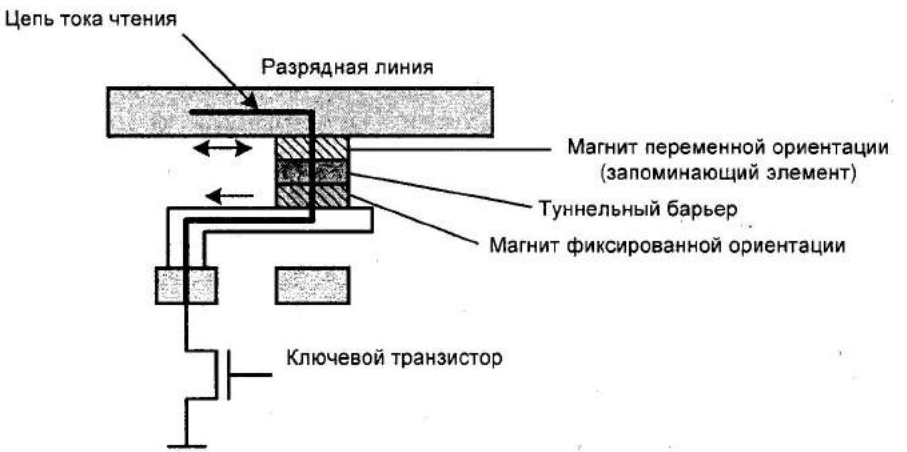


Рис. 4.52. Схематическая конструкция запоминающего элемента типа MTJ

Современные разработки MRAM характеризуются параметрами, приведенными в конце этого параграфа. Они еще далеки от теоретически достижимых,

уровень которых очень высок: время записи 2,3 нс, т. е. на три порядка меньше, чем у флэш-памяти, время считывания 3 нс, т. е. приблизительно в 20 раз меньше, чем у современных DRAM, число циклов практически неограничено (превышает 10^{15}), потребляемая мощность на порядки меньше, чем у DRAM. Микросхемы обладают повышенной радиационной стойкостью.

Фирма Motorola в 2002 г. продемонстрировала MRAM емкостью 1 Мбит (при топологической норме 0,6 мкм). К 2004 г. этой фирмой ожидаются MRAM емкостью 32 Мбит или более. К этому же году намерена выпустить прототипные кристаллы MRAM и фирма Intel.

ЗУ типа OUM (с использованием фазовых переходов вещества)

ЗУ типа OUM (Ovonyx Unified Memory, по названию фирмы Ovonyx) построены на основе физических эффектов, которые уже использовались в памяти на компакт-дисках. В OUM эти же эффекты применены для реализаций *памяти по интегральной технологии. Как и в дисках CD и DVD с* перезаписью данных, в памяти OUM применены *халкогенидные сплавы.* Халкогенид — сплав GeSbTe, который может иметь кристаллическое проводящее или аморфное непроводящее состояния. Эти состояния материал может сохранять, а выявлять их можно измерением сопротивления запоминающего элемента. Состояния "кристаллическое—аморфное" взаимно обратимы, их изменения происходят быстро. В конструкции запоминающего элемента (рис. 4.53) небольшой объем халкогенида играет роль резистора с программируемым сопротивлением при динамическом диапазоне между значениями низкого и высокого сопротивлений около 100.

Фазовое состояние халкогенида программируется пропусканием через элемент импульсов тока, имеющих разные параметры. Управление током производится с помощью МОП-транзистора. Чтение бита осуществляется путем измерения сопротивления элемента. При записи программируемый материал нагревается до температуры, превышающей точку плавления, и затем быстро охлаждается, что вызывает его переход в аморфное состояние. В кристаллическое состояние элемент переводится нагреванием до температуры ниже точки плавления с последующей выдержкой в ней в течение 50 нс.

В элементе памяти с халкогенидом можно программировать сопротивление не только для двух его значений (максимального и минимального), но и для промежуточных, а это означает принципиальную возможность использовать в памяти многоуровневые сигналы и, следовательно, хранить в одном элементе более одного бита данных.

Запоминающие элементы OUM просты по конструкции, потребляют малую мощность, энергонезависимы, имеют неразрушающее чтение, допускают до 10^{12} циклов записи/стирания. Особо можно отметить предполагаемую высокую надежность памяти OUM, что существенно для военной и аэрокосми-

ческой аппаратуры. Фирма Ovonics совместно с фирмой Intel разработала тестовый кристалл памяти OUM с топологической нормой 0,18 мкм.

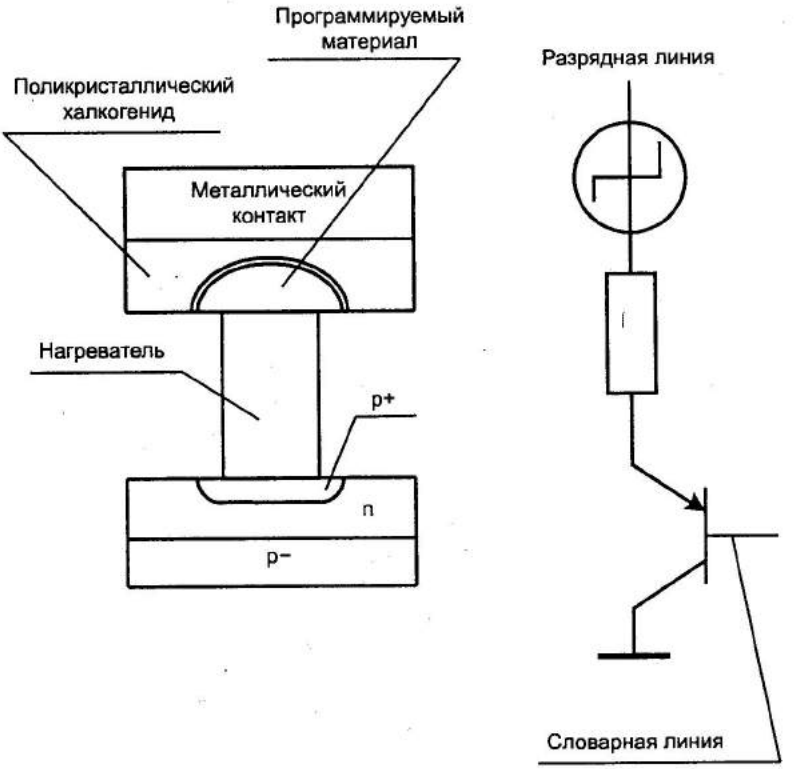


Рис. 4.53. Конструкция и схема запоминающего элемента памяти OUM

Параметры перспективных микросхем памяти

Параметры перспективных микросхем памяти, имеющих наибольшую степень практического освоения, приведены в табл. 4.3.

Таблица 4.3

Параметр	FRAM	MRAM	OUM
Информационная емкость, Мбит	64	1	4
Площадь ячейки, условных единиц*	18	10–20	5–8
Допустимое число циклов	10^{16}	10^{14}	10^{12}
Время чтения/записи, нс	40/40	50/50	50/50

Таблица 4.3 (оконча

Параметр	FRAM	MRAM	OUM
Дополнительных технологических операций	2	4	3–4
Производство	Освоено	2004	–

* Условной единицей площади является квадрат ширины межсоединения, выполненно го в нижнем слое металлизации.

§ 4.13. Заключительные замечания

Производство современных ИС ЗУ требует больших инвестиций для создания новых заводов (миллиардов USD). Тем не менее архитектура, схемотехника и технология ЗУ быстро совершенствуются. Поколения динамических ЗУ сменяются приблизительно через пять лет. Цены на новые микросхемы памяти обычно достаточно высоки, но со временем быстро снижаются. Уровень интеграции и быстродействие памяти растут с уменьшением топологических норм проектирования, которые в настоящее время достигли 0,13 мкм, а по прогнозу на ближайшие годы должны снизиться до 0,05—0,07 мкм. Это создает хорошие перспективы развития как новых, так и уже сложившихся разновидностей ЗУ, что и подтверждается регулярными сообщениями фирм-разработчиков о новых успехах. Так, например, недавно фирмой Intel опубликованы данные о микросхемах, изготовленных по технологическим процессам с нормой 90 нм (0,09 мкм). Применительно к статическим синхронным ЗУ в этих данных говорится о рекордно миниатюрном запоминающем элементе размером 1 мкм^2, что в 100 раз меньше размеров эритроцита крови. В то же время на горизонте появляются новые смелые идеи создания устройств памяти на новых материалах и физических явлениях, обещающие трудно вообразимые сегодня параметры емкости, быстродействия, энергетической экономичности, надежности ЗУ будущего. В конечном счете речь идет об отображении двоичных данных с помощью определенного состояния всего одного электрона.

Переживающая трудности отечественная электронная промышленность выпускает микросхемы памяти, параметры которых за последние 10—20 лет практически не изменились и являются весьма скромными по сравнению с уровнем мировой техники. Эту продукцию можно критиковать, но полезно и помнить, что покупая отечественную продукцию, мы поддерживаем возможности развития нашей электроники.

Параметры старших представителей наиболее известных отечественных серий микросхем памяти приведены на рис. 4.54.

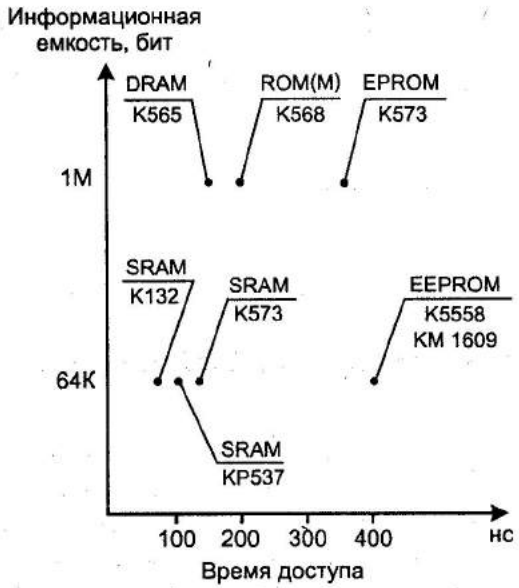
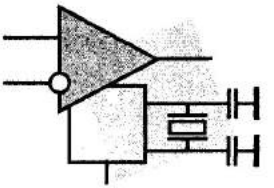


Рис. 4.54. Параметры емкости и быстродействия отечественных запоминающих устройств

Параметры зарубежных микросхем памяти высшего уровня интеграции характеризуются следующими цифрами. Динамические ЗУ массового производства достигли информационной емкости 2 Гбит, кристаллы флэш-памяти — 4 Гбит. Латентность динамических ЗУ с фрагментированными матрицами снижена до 7—15 нс. Быстродействие статических ОЗУ характеризуется тактовыми частотами в 250 МГц, а в ближайшие годы планируется достичь 600 МГц и далее 1 ГГц (в следующие 5—10 лет). Максимальная информационная емкость современных статических ОЗУ составляет 8—16 Мбит.

Литература к главе: [18], [24], [25], [29], [32], [35], [63], [67], [IV], [VI], [X], [XI], [XII], [XIV], [XIX], [XXIII], [XXV], [XXVI], [XXX].

Глава 5



Микропроцессорные БИС/СБИС и их применение в микропроцессорных системах

§ 5.1. Общие сведения. Структура и функционирование микропроцессорной системы

Микропроцессор — термин, по-видимому, самый модный в современной микроэлектронике и вычислительной технике. Необычайная популярность микропроцессоров объясняется тем, что их появление привело к внедрению вычислительной техники в самые разнообразные сферы жизни. Универсальность микропроцессоров ведет к большой тиражности их производства и, следовательно, к снижению их стоимости, а это, в свою очередь, расширяет круг потребителей и способствует дальнейшему удешевлению микропроцессоров.

Микропроцессором (МП) называют построенное на одной (реже на нескольких) БИС/СБИС программно-управляемое устройство, осуществляющее процесс обработки информации и управление им. Заметим, кстати, что в современных условиях уточнение "микро" применительно к термину "процессор" чаще всего излишне, в частности, это так для рассматриваемых в данной книге устройств, поскольку об иных вариантах реализации процессоров говорить не приходится.

Микропроцессоры появились, когда уровень интеграции ИС достиг значений, при которых блоки, необходимые для программной реализации алгоритмов, удалось разместить на одном кристалле. МП — центральный процессорный элемент микропроцессорной системы (микро-ЭВМ).

Выполнение заданной программы реализуется в *микропроцессорной системе (МПС)*, основными частями которой являются *микропроцессор, память, устройства ввода/вывода (внешние устройства) и интерфейсные схемы.* Решаемая задача определяет реализуемую программу, структура микропроцессорной

системы при решении различных задач остается неизменной, что и определяет ее универсальность.

Первый микропроцессор Intel 4004 появился в 1971 г. В то время в области вычислительной техники самой массовой продукцией были калькуляторы. Одна из японских фирм, работая в этом направлении, поставила задачу сужения номенклатуры микросхем, используемых в калькуляторах. По ее заказу работы велись, в частности, в одном из американских коллективов, где и была предложена Хоффом программно-управляемая микросхема. Так как калькуляторы работают в двоично-десятичной системе счисления, оперирующей с тетрадами (четырехразрядными словами), первый микропроцессор был четырехразрядным. Появление первого микропроцессора дало толчок бурному развитию микропроцессорной техники.

Первый микропроцессор содержал 2300 транзисторов и имел рабочую частоту 108 кГц при среднем времени выполнения команды приблизительно десять тактов. Современный микропроцессор содержит более 200 млн транзисторов, работает на тактовых частотах до 5 ГГц, выполняет команды за один такт или даже за долю такта. Движущими силами столь впечатляющего развития служат все возрастающие потребности практики.

В процессе развития архитектура МПС претерпела существенные изменения. Первые МПС строились по так называемой *Принстонской архитектуре* (архитектуре фон Неймана), в которой память для команд и данных является общей. Эта архитектура имеет свои достоинства — простоту, возможность оперативного перераспределения памяти между областями хранения команд и данных и др. Недостаток — последовательная во времени выборка из памяти команд и данных, передаваемых по одной и той же системной шине, что ограничивает производительность МПС. Тем не менее в силу своих достоинств Принстонская архитектура не только длительное время доминировала в микропроцессорной технике, но сохранила свое место и до настоящего времени.

В *Гарвардской архитектуре* память разделена на память команд и память данных, причем каждая из них имеет собственную шину для общения с процессором. При этом во время передач данных для выполнения текущей команды можно производить выборку и расшифровку следующей, что повышает производительность МПС. Реализация системы по сравнению с Принстонской архитектурой усложняется (в системе больше шин), ниже коэффициент использования памяти. Но в МПС высокой производительности и внутренних структурах высокопроизводительных МП Гарвардская архитектура находит широкое применение.

По другому архитектурному признаку, связанному с характером системы команд, микропроцессоры делятся на:

- CISC-процессоры;
- RISC-процессоры;
- VLIW-процессоры.

Процессоры CISC имеют так называемую сложную систему команд (CISC — Complex Instruction Set Computer), т. е. большой набор разноформатных команд при использовании многих способов адресации. Архитектура CISC присуща классическим процессорам, она в силу многообразия команд позволяет применять эффективные алгоритмы решения задач, но, в то же время, усложняет схему процессора и его стоимость и в общем случае не обеспечивает его максимального быстродействия.

RISC-процессоры имеют сокращенную систему команд (RISC — Reduced Instruction Set Computer), из которой исключены редко применяемые команды. Форматы команд, по крайней мере, подавляющее их большинство, идентичны (например, все команды содержат по 4 байта), резко уменьшено число используемых способов адресации. Данные, как правило, обрабатываются только с регистровой или непосредственной адресацией. Значительно увеличено число регистров процессора, т. е. его емкая внутренняя память, что позволяет редко обращаться к внешней памяти (модулю памяти МПС), а это повышает быстродействие системы. Идентичность временных циклов выполнения команд отвечает потребностям конвейерных схем обработки информации. В результате может быть достигнуто упрощение схемы процессора при увеличении его быстродействия.

Последними по времени появления (менее 10 лет назад) стали VLIW-процессоры (VLIW — Very Long Instruction Word), особенность которых состоит в использовании очень длинных команд (16 и более байт). Отдельные поля длинной команды определяют несколько подлежащих реализации операций, которые могут выполняться параллельно во времени в нескольких операционных устройствах процессора. Таким образом, одна длинная команда определяет сразу группу операций. VLIW-процессоры считаются перспективными для высокопроизводительных процессоров.

Эволюционное развитие микропроцессоров, направленное главным образом на повышение их производительности, обеспечивалось и обеспечивается несколькими факторами, в первую очередь следующими:

- применением конвейеров, т. е. разбиением на отдельные этапы выполняемых операций, которое позволяет переходить к выполнению этапов над следующими операндами сразу же после освобождения ступени конвейера от обработки предыдущих, т. е. не дожидаясь конца всей операции над предыдущими операндами;
- установлением кэш-памяти первого уровня на одном кристалле с процессором, что позволяет получить высокую скорость обмена процессора с кэш-памятью, а также применением ее как для команд, так и для данных;
- введением в структуру процессора предсказателей ветвления программы, повышающих эффективность кэш-памяти путем снижения процента промахов кэша;

- реализацией на одном кристалле с основным процессором вспомогательных специализированных процессоров (сопроцессоров), прежде всего для обработки данных с плавающей запятой, введением в структуру процессора дополнительных аппаратных средств реализации важных для решаемых задач специальных функций;
- выполнением команд не в порядке их следования в программе, а по мере готовности данных и ресурсов для данной команды с последующим учетом фактической последовательности выполнения команд;
- делением исполнительного ядра процессора на области (кластеры) с концентрацией по возможности операций внутри одного и того же кластера и минимизацией распространения сигналов на большие расстояния по кристаллу, поскольку времена распространения сигналов в пространстве уже становятся причиной ограничения быстродействия современных микропроцессоров;
- применением для внутренних исполнительных устройств процессора повышенной частоты тактирования с целью исключить возможные интервалы ожидания со стороны других устройств.

Перечисленные и некоторые новые намечающиеся способы повышения быстродействия микропроцессоров ведут к усложнению их структур. Изучение микропроцессоров в более или менее полном объеме является предметом обширного курса (достаточно упомянуть, что учебное пособие "Микропроцессорные системы" издательства "Политехника", 2002 г., имеет объем почти в тысячу страниц). В данном учебном пособии рассматриваются лишь основы микропроцессорной техники. Полезность такого рассмотрения обусловлена, по крайней мере, двумя соображениями: во-первых, всякое обучение должно идти от простого к сложному; во-вторых, в курсе по общим вопросам цифровой схемотехники изучение микропроцессора необходимо, поскольку он управляет работой других устройств, и без понимания взаимодействия процессора с этими устройствами непонятными становятся и принципы их функционирования. Принципы же управления различными устройствами со стороны микропроцессора во многом остаются одинаковыми для процессоров разных классов.

Кроме понятий "микропроцессор" и "микропроцессорная система" существует также понятие "микропроцессорный комплект БИС", которое сейчас употребляется не слишком часто, но ранее имело широкое распространение. *Микропроцессорным комплектом* (МПК) называют совокупность БИС/СБИС, пригодных для совместного применения в составе МПС. Понятие МПК задает номенклатуру микросхем с точки зрения возможностей их совместного применения (совместимость по архитектуре, электрическим параметрам, конструктивным признакам и др.). В состав МПК могут входить микросхемы различных серий и схемотехнологических типов при условии их совместимости.

Структура микропроцессорной системы

Структура МПС является *магистрально-модульной*. В такой структуре имеется группа магистралей (шин), к которым подключаются различные модули (блоки), обменивающиеся между собой информацией по одним и тем же шинам поочередно, в режиме разделения времени.

Термин "шина" относится к совокупности цепей (линий), число которых определяет разрядность шины. Термин "шина" применяется также в более широком смысле как синоним термина "интерфейс" *(см. главу 6)*.

Типична *трехшинная структура* МПС с шинами адресов ША, данных ШД и управления ШУ. Наряду с русскими терминами применяются английские AB (Address Bus), DB (Data Bus) и CB (Control Bus) или просто шины A, D, C. Трехшинная структура в чистом виде характерна для простых МПС, в более производительных системах шинная структура образует более сложную иерархию, но в своей основе разделение на перечисленные виды шин остается справедливым, хотя и может относиться не к отдельным шинам, а к их группам.

На рис. 5.1 показана типичная структура микропроцессорной системы. По адресной шине А в систему передаются адреса модулей, к которым обращается микропроцессор МП. В эту шину включен шинный формирователь ШФ, обеспечивающий работу МП на нагрузку, образуемую внешними цепями. Собственной нагрузочной способности у выводов МП, как правило, не хватает.

Адреса используются блоками постоянной и оперативной памяти ПЗУ и ОЗУ (ROM и RAM), а также адаптерами и контроллерами, посредством которых микропроцессор общается с различными внешними устройствами (ВУ). При адресации памяти старшие и младшие разряды адресов используются по-разному. Старшие разряды поступают на селектор адреса $СА_{ЗУ}$. Соответственно старшим разрядам селектор разрешает работу тех или иных блоков с помощью сигналов выбора кристаллов CS (Chip Select). Младшие разряды адресов используются для адресации слов внутри выбранного модуля ("декодируются на кристалле"). При работе оперативной памяти она получает от процессора управляющий сигнал записи/чтения R/W (Read/Write), указывающий на требуемое направление передачи данных между памятью и процессором. Постоянная память в подобном сигнале не нуждается, т. к. используется в рабочем режиме только для чтения. Селектор адресов внешних устройств $СА_{ВУ}$ декодирует адреса, присвоенные адаптерам и контроллерам, и вырабатывает сигналы CS разрешения работы выбранным модулям.

С внешними устройствами, работающими с параллельными кодами, процессор общается с помощью программируемого параллельного адаптера ППА. Показанный на рисунке адаптер ППА имеет три канала связи с внешними

устройствами (т. е. может обслуживать до трех полноразрядных ВУ). Для каналов ППА могут быть выбраны различные режимы ввода, вывода и двунаправленных передач при разных способах обмена с процессором. Общение процессора с ВУ, работающими с последовательными кодами, производится с помощью программируемого связного адаптера ПСА, который преобразует параллельные коды, получаемые от процессора, в последовательные для ВУ или для последовательной линии связи, либо наоборот, получая последовательные коды, преобразует их в параллельные для передачи в процессор. Программирование адаптера позволяет настраивать его на разные протоколы и режимы обмена. Последовательные выходные сигналы адаптера ПСА поступают на линию передатчика TxD (Tranceiver Data), последовательные входные — на линию приемника RxD (Receiver Data).

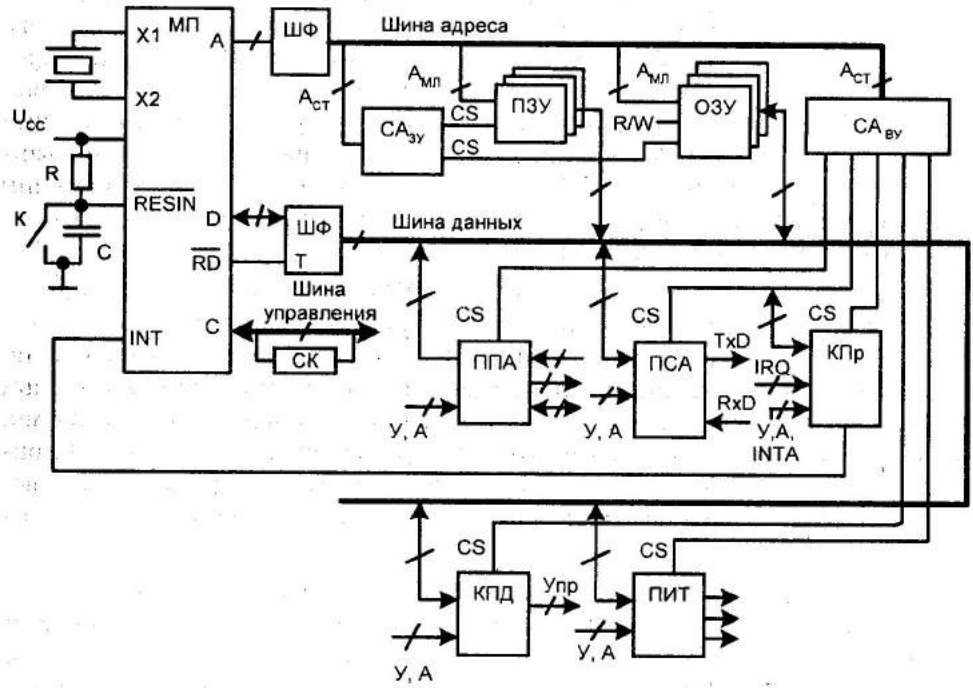


Рис. 5.1. Структура микропроцессорной системы

Контроллеры прерываний КПр обеспечивают обмен с внешними устройствами в режиме прерывания (временной остановки) выполняемой программы для обслуживания запроса от внешнего устройства. Контроллер прерываний принимает запросы прерываний IRQ (Interrupt Requests) от нескольких внешних источников и обрабатывает их с учетом приоритетов и маскирова-

ний. Определив подлежащий обслуживанию запрос, контроллер запрашивает прерывание у процессора, выставляя для него сигнал INT (Interrupt). Если процессор признает запрос, он производит необходимую подготовку для перехода от основной программы к подпрограмме обслуживания прерывания, после чего отвечает контроллеру сигналом подтверждения прерывания INTA (Interrupt Acknowledge). Сигнал INTA служит стробом чтения сформированного контроллером начального адреса подпрограммы *(вектора)*. Вектор поступает в микропроцессор, затем выполняется подпрограмма, после чего возобновляется выполнение основной программы. Так реализуются *векторные прерывания*.

Контроллер прямого доступа к памяти КПД (DMA, Direct Memory Access) обслуживает процесс прямого обмена данными между внешними устройствами и памятью. При отсутствии такого режима, называемого *прямым доступом к памяти*, передача данных между указанными модулями (например, между винчестером и ОЗУ) для каждого слова протекает так: сначала слово читается процессором из источника данных, потом оно записывается процессором в их приемник, что требует двух машинных циклов и замедляет передачу. При прямом доступе к памяти процессор отключается от шин системы и передает управление шинами предварительно запрограммированному КПД, который реализует более быструю непосредственную (за один цикл) передачу данных между источником и приемником. Особенно эффективен ПДП при блочных передачах.

Программируемые интервальные таймеры ПИТ (PIT, Programmable Interval Timer) выполняют над временными интервалами операции, играющие большую роль в ряде разнообразных ситуаций (часы реального времени, генерация звуковых сигналов, сторожевые таймеры, генерация временных меток в многозадачных процессах с разделением времени между задачами, выработка широтно-модулированных импульсных сигналов и т. д.). На рисунке показан таймер с тремя выходными каналами, каждый из которых может выполнять свою функцию по формированию временных последовательностей.

Кроме сигналов, индивидуально обозначенных на рис. 5.1, адаптеры и контроллеры имеют и многие другие управляющие сигналы, объединенные на рисунке в группы У, А (управление, адресация). В число таких сигналов входят стробы записи (для загрузки в программируемые модули управляющих слов), чтения (для чтения программой слов состояния модулей), адресные коды для адресации внутренних регистров модулей, сигналы сброса, тактирования и др.

Микропроцессор МП также вырабатывает множество сигналов управления системой, не показанных на упрощенной схеме (см. рис. 5.1).

Передачи данных в МПС осуществляются по системной ШД, разрядность которой определяет понятие *"разрядность процессора"*. Эти передачи двуна-

правлены, направление задается шинным формирователем ШФ (BD, Bus Driver) в зависимости от сигнала T (Transmit). При активном состоянии формируемого процессором сигнала чтения RD (Read) данные передаются через ШФ справа налево, при пассивном — в обратном направлении. К шине данных подключены информационные выводы всех модулей МПС.

Выводы X1 и X2 служат для подключения кварцевого резонатора или иных контуров, задающих частоту тактовому генератору, расположенному в МП.

Вход $\overline{\text{RESIN}}$ является входом асинхронного сброса, приводящим МП в исходное состояния. Сигнал сброса L-активный. Сброс может быть осуществлен замыканием ключа K и автоматически происходит при включении питания U_{CC}. В этом случае благодаря цепочке RC напряжение на входе $\overline{\text{RESIN}}$ после включения питания нарастает постепенно, и в течение некоторого времени остается низким (ниже порогового), что равноценно подаче на этом интервале времени L-активного сигнала $\overline{\text{RESIN}}$.

Выполняя программу, МП обрабатывает команду за командой. Команда задает выполняемую операцию и содержит сведения об участвующих в ней операндах. После приема команды происходит ее расшифровка и выполнение, в ходе которого МП получает необходимые данные из памяти или внешних устройств. Ячейки памяти и внешние устройства (порты) имеют номера, называемые адресами, которыми они обозначаются в программе.

Таким образом, по однонаправленной адресной шине МП посылает адреса, определяя объект, с которым будет обмен, по шине данных (двунаправленной) обменивается данными с модулями (блоками) системы, по шине управления в разных направлениях передаются управляющие сигналы.

ПЗУ (ROM) хранит фиксированные программы и данные, оно является энергонезависимым и при выключении питания информацию не теряет.

ОЗУ (RAM) хранит оперативные данные (изменяемые программы, промежуточные результаты вычислений и др.), является энергозависимым и теряет информацию при выключении питания (если не применяются специальные методы, например, автоматическое подключение автономного резервного питания на время отсутствия основного). Для приведения системы в работоспособное состояние после включения питания ОЗУ следует загрузить необходимой информацией.

Устройства ввода/вывода (УВВ) или внешние устройства (ВУ) — технические средства для передачи данных извне в МП или память, либо из МП или памяти во внешнюю среду. Для подключения ВУ необходимо привести их сигналы, форматы слов, скорость передачи и т. п. к стандартному виду, воспринимаемому данным МП. Это и выполняется адаптерами и другими интерфейсными блоками.

Многие микропроцессоры имеют *мультиплексируемую шину* адресов/данных (рис. 5.2). Применение мультиплексируемых шин позволяет уменьшить число внешних выводов кристалла. В этом случае разрядность адресной шины сокращается вдвое, и по ней в систему передается только старший полуадрес (в частности, в микропроцессорах типа Intel 8085, разрядности шин которых приведены на рис. 5.2, это старший байт 16-разрядного адреса). Младший полуадрес в начале машинного цикла выдается процессором по мультиплексируемой шине адресов/данных AD_{7-0} и управляющим сигналом ALE (Address Latch Enable) загружается в специальный внешний регистр, где и сохраняется на все время машинного цикла, так что выход регистра совместно с шиной A_{15-8} образуют полный адрес A_{15-0}. После передачи младшего полуадреса шина AD_{7-0} отдается для передачи данных. Эти передачи двунаправлены, и направление передач задается шинному формирователю сигналом по его входу T.

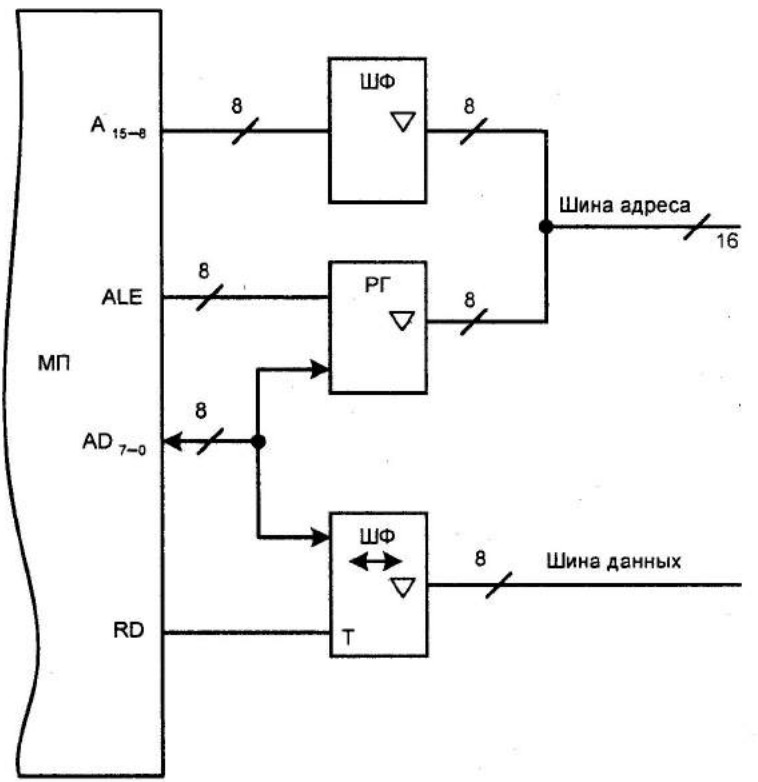


Рис. 5.2. Мультиплексирование шины адресов/данных

§ 5.2. Управление памятью и внешними устройствами. Построение модуля памяти

Память состоит из ячеек, каждой из которых присваивается свой адрес. Совокупность адресов, которые могут быть сформированы процессором, образует *адресное пространство* МПС. Адреса памяти могут занимать все адресное пространство (АП) или его часть, а линейно организованная память независимо от ее технической реализации может быть условно представлена набором регистров (ячеек), число которых M, а разрядность — N (рис. 5.3).

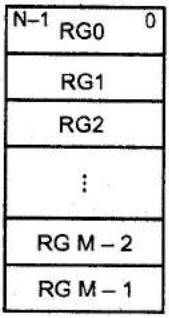


Рис. 5.3. Условное представление линейной организации памяти

Свои адреса имеют и внешние устройства (ВУ). Процессор при обмене данными всегда должен выбрать только одну из ячеек памяти или одно ВУ. Такой выбор осуществляется схемами декодирования адреса. При управлении памятью и ВУ процессор должен вначале сформировать нужный адрес, который затем декодируется.

В МПС применяют несколько способов формирования адресов.

При *прямой адресации* код адреса содержится в команде, подлежащей выполнению. Прямая адресация удобна, но удлиняет команды (увеличивает их разрядности), т. к. при значительных емкостях памяти разрядности адресов достаточно велики. В случае прямой регистровой адресации, когда операнд находится в одном из внутренних регистров процессора, адрес является малоразрядным, поскольку число таких регистров мало. В этом случае прямая адресация проявляет все свои достоинства.

При *косвенной адресации* в команде явно или неявно указывается регистр процессора, содержащий сведения об адресе операнда. Команда сохраняет компактность, но для ее выполнения требуется предварительная настройка — загрузка сведений в регистр косвенного адреса (индексный регистр). Косвенная адресация удобна при обработке списков, когда настройка производится од-

нократно, а очередной адрес получается модификацией предыдущего (изменением его на единицу). В зависимости от действий, которые производятся над содержимым индексного регистра при формировании адреса, различают несколько видов косвенной адресации. При *простой косвенной адресации* обращение к ячейке памяти производится по адресу, находящемуся в индексном регистре, т. е. никаких действий над его содержимым не производится. При *относительной косвенной адресации* адрес ячейки памяти получается суммированием содержимого индексного регистра и числа, задаваемого в команде. Существуют и другие разновидности косвенной адресации.

При *непосредственной адресации* в команде содержится сам операнд.

Возможность использования различных видов адресации сокращает объем и время выполнения программ.

С помощью того или иного способа адресации формируется физический адресный код, поступающий на шину адреса для выбора ячейки памяти или ВУ, с которыми взаимодействует процессор.

Адресация может быть *абсолютной* или *неабсолютной*. При абсолютной адресации обратиться к ячейке памяти или ВУ можно только по одному-единственному адресу. При неабсолютной адресации для ячейки памяти или ВУ можно выделить некоторую зону адресов. Число таких зон, естественно, будет меньше, чем число отдельных адресов, поэтому для указания зоны потребуется меньшая разрядность адреса. Иными словами, абсолютная адресация требует полного декодирования адреса, а неабсолютная — частичного, что упрощает схемы декодирования. Возможность использования неабсолютной адресации связана с наличием в адресном пространстве (АП) "лишнего" пространства. Частным случаем неабсолютной адресации ВУ является так называемая *линейная селекция* (линейный выбор), подробнее рассмотренная далее.

В простых МПС адресный код часто рассматривается как состоящий из двух частей. Одна часть указывает на *страницу*, в которой расположен искомый объект адресации, другая является адресом этого объекта на данной странице. Страницей является та или иная часть АП (какая именно — зависит от организации микросхем, из которых строится модуль памяти).

С точки зрения использования АП памятью и ВУ различают концепции *интерфейса с общей шиной и раздельной шиной*.

В рамках первой концепции для адресов памяти и ВУ выделяются части общего АП. К внешним устройствам обращение происходит так же, как и к ячейкам памяти, т. е. с помощью тех же команд и той же шины. Недостатком этой концепции является сужение АП для памяти, поскольку часть АП занимается внешними устройствами. Достоинство состоит в том, что над данными, получаемыми от ВУ, можно производить все те операции, которые имеются в системе команд процессора для данных, находящихся в

ячейках памяти. Таких операций много и это способствует улучшению параметров программ и упрощению программирования. Концепцию "с общей шиной" называют также *вводом/выводом, отображенным на память.*

В концепции "с раздельной шиной" ячейки памяти и ВУ имеют свои АП. При этом требуется наличие управляющих сигналов, определяющих, с каким типом объектов ведется обмен. Например, вводится сигнал IO/M (Input-Output/Memory), указывающий, адресуется память или ВУ. При этом память может использовать все адресное пространство (АП). Для обмена с ВУ обычно имеются только операции ввода IN port и вывода OUT port, и теряется возможность применять к данным от ВУ широкий набор команд, имеющихся для работы с данными, хранимыми в памяти.

Диапазон адресов, к которым может обращаться процессор (т. е. емкость АП) связан с разрядностью шины адреса m соотношением АП = 2^m. Например, с помощью 16-разрядной шины адреса можно адресовать 2^{16} = 64K объектов, с помощью 20-разрядной 2^{20} = 1M объектов и т. д.

АП используется блоками ОЗУ, ПЗУ и ВУ, к которым обращается процессор. Распределение АП между указанными претендентами производится проектировщиком системы, имеющим известную свободу действий, хотя у конкретных процессоров могут быть особенности, заставляющие отдавать определенную область АП для адресации соответствующих объектов.

Для краткости записей адреса в АП обычно выражают в шестнадцатиричной системе счисления, для оценки емкостей АП часто используется единица измерения K = 2^{10} = 1024 или M = 2^{20} = 1048576.

Модули памяти

Модуль памяти обычно состоит не из одной микросхемы, а из нескольких. Для микросхем памяти типична организация $2^k \times \ell$, где k — четное число; 2^k — число хранимых слов; ℓ — их разрядность. Если требуется модуль памяти с организацией $2^m \times n$, а имеются микросхемы с организацией $2^k \times \ell$, где $k < m$ и $\ell < n$, то при страничной организации модуля его состав и структура определяются следующими соображениями.

Для наращивания разрядности хранимых слов до требуемой включаются параллельно несколько микросхем (а именно, n / ℓ ИС). Это образует субмодуль (страницу), который хранит 2^k слов.

Для увеличения числа хранимых слов до 2^m требуется взять 2^{m-k} субмодулей. Адрес слова в пределах субмодуля указывается k младшими разрядами адреса, поступающими непосредственно на адресные входы микросхем, а старшие разряды адреса используются для формирования сигнала разрешения работы того или иного субмодуля (рис. 5.4).

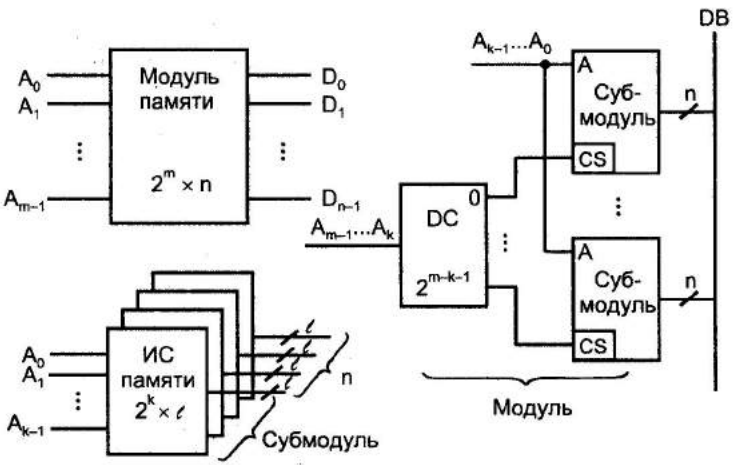


Рис. 5.4. Структуры модуля памяти

Сигналы управления

Адресация — только часть процесса управления памятью и ВУ. Кроме адресов требуются стробы чтения и записи ($\overline{RD}$ и $\overline{WR}$), задающие направление обмена, сигналы разрешения работы ($\overline{CS}$, EN), признак обращения к ВУ или памяти (IO/M). Процессор обычно вырабатывает минимальную группу сигналов, тогда как в системном интерфейсе может быть предусмотрена несколько иная группа. В частности, МП К1821ВМ85А формирует три сигнала: сигнал чтения ($\overline{RD}$), записи ($\overline{WR}$) и сигнал IO/M, т. е. сигнал обращения к ВУ при высоком уровне и к памяти — при низком. В системном же интерфейсе используется система из четырех сигналов: сигнала чтения из памяти $\overline{MEMR}$, записи в память $\overline{MEMW}$, чтения из ВУ $\overline{IOW}$ и записи в ВУ $\overline{IOR}$.

К четверке сигналов легко перейти по следующим соотношениям:

$$\overline{MEMR} = \overline{RD \cdot \overline{IO/M}} = \overline{RD} \vee IO/M;$$

$$\overline{MEMW} = \overline{WR \cdot \overline{IO/M}} = \overline{WR} \vee IO/M;$$

$$\overline{IOR} = \overline{RD \cdot IO/M} = \overline{RD} \vee \overline{IO/M};$$

$$\overline{IOW} = \overline{WR \cdot IO/M} = \overline{WR} \vee \overline{IO/M}.$$

Статические асинхронные ОЗУ могут быть нетактируемыми или тактируемыми. Для тактируемых ОЗУ нужен импульсный характер какого-либо сиг-

нала управления (обычно сигнала $\overline{CS}$). В этом случае для повторного разрешения работы памяти нужно предварительно вернуть сигнал в пассивное состояние. Для придания сигналу импульсного характера можно применить, в частности, соотношение CS = MEMR \/ MEMW. При этом обеспечивается пассивное состояние сигнала CS на интервалах, на которых не действуют ни сигнал чтения, ни сигнал записи.

Иногда условием обмена является *готовность* к нему памяти или ВУ. Для выявления готовности применяют такой метод: появление адреса медленного устройства ведет к запуску генератора одиночного импульса необходимой длительности, на время существования которого сигнал готовности RDY (Ready) снимается. Длительность интервала неготовности рассчитывается согласно требованиям медленного устройства. Процессор ждет появления сигнала готовности и только после его появления выполняет операцию обмена. Чтобы избежать потерь времени, желательно генерировать интервал неготовности с привязкой его к синхроимпульсам МПС.

Примеры схем управления памятью и ВУ со стороны процессора рассмотрены далее после изучения конкретного МП.

Вид обмена

Для передачи данных между какими-либо устройствами необходимо вырабатывать сигналы управления этим процессом. В МПС такие сигналы могут генерироваться либо процессором, либо контроллером прямого доступа к памяти. Таким образом, выполнение операций записи и чтения данных может проходить в МПС в режимах *программно-управляемого обмена или прямого доступа к памяти (ПДП).* В первом случае управление процессом ведет процессор, во втором — контроллер ПДП.

Различные виды обмена отличаются друг от друга не только источником управляющих сигналов, но и характером инициирования передач. С этой точки зрения существенно различными являются два случая. В первом случае *инициатором обмена является программа.* При этом возможно взаимодействие с устройством, всегда готовым к обмену или с ожиданием готовности устройства. В последнем случае вырабатываются сигналы, сообщающие о состоянии устройства. Процессор анализирует их и при готовности устройства реализует программу обслуживания данного устройства. Такой обмен может быть сопряжен с большими потерями времени. Быстродействие внешних устройств, с которыми идет обмен, зачастую очень мало в сравнении с быстродействием процессора. Ожидая готовности устройства, процессор не выполняет полезной работы, а занят в каждом цикле проверкой состояния внешнего устройства и простаивает в течение больших интервалов времени.

Во втором случае при обменах по прерываниям трата времени на ожидание исключается, т. к. *инициатива обмена исходит от внешнего устройства* (ВУ).

При своей готовности ВУ сигнализируют процессору, запрашивая у него прерывания основной программы и обслуживания обмена. Процессор завершает выполнение текущей команды и переходит к подпрограмме обслуживания прерывания. Отсутствие длительных интервалов ожидания существенно увеличивает производительность МПС.

Для обмена между памятью и ВУ без участия процессора используется режим ПДП. В обычном режиме пересылка данных между памятью и ВУ требует вначале приема данных в процессор, а затем выдачи их приемнику, что снижает темп передачи. В режиме ПДП процессор отключается от системных шин и передает управление обменом специальному контроллеру ПДП, что увеличивает темп передачи данных. Наличие ПДП также повышает эффективность МПС.

§ 5.3. Структура и функционирование микропроцессора

Во всем мире широко применяются микропроцессоры фирмы Intel и их аналоги. Эта фирма разработала первый МП, затем целый ряд их семейств и в настоящее время по разным оценкам производит 80—90% от общего мирового объема выпуска микропроцессоров. При изучении МП целесообразно ориентироваться на конкретные образцы. В данном случае выбран МП К1821ВМ85А — аналог микропроцессора Intel 8085A. Это простой для изучения объект, на котором легко проследить основные принципы работы МП. Несмотря на свой многолетний возраст, этот МП до сих пор встречается в каталогах фирм. Естественно, что областью применения подобных микропроцессоров не являются компьютеры, в которых сейчас применяют гораздо более мощные и производительные МП. Простые МП используются в тех системах управления различной аппаратурой, где их возможностей хватает для решения поставленных задач.

Структура микропроцессора К1821ВМ85А

Структура микропроцессора К1821ВМ85А показана на рис. 5.5.

Микропроцессор имеет восьмиразрядную шину данных (внутреннюю), через которую его блоки обмениваются информацией. На схеме приняты следующие обозначения:

- AC (Accumulator) — регистр-аккумулятор, выполненный на двухступенчатых триггерах и способный хранить одновременно два слова (один из операндов и результат операции);
- TR (Temporary Register) — регистр временного хранения одного из операндов;

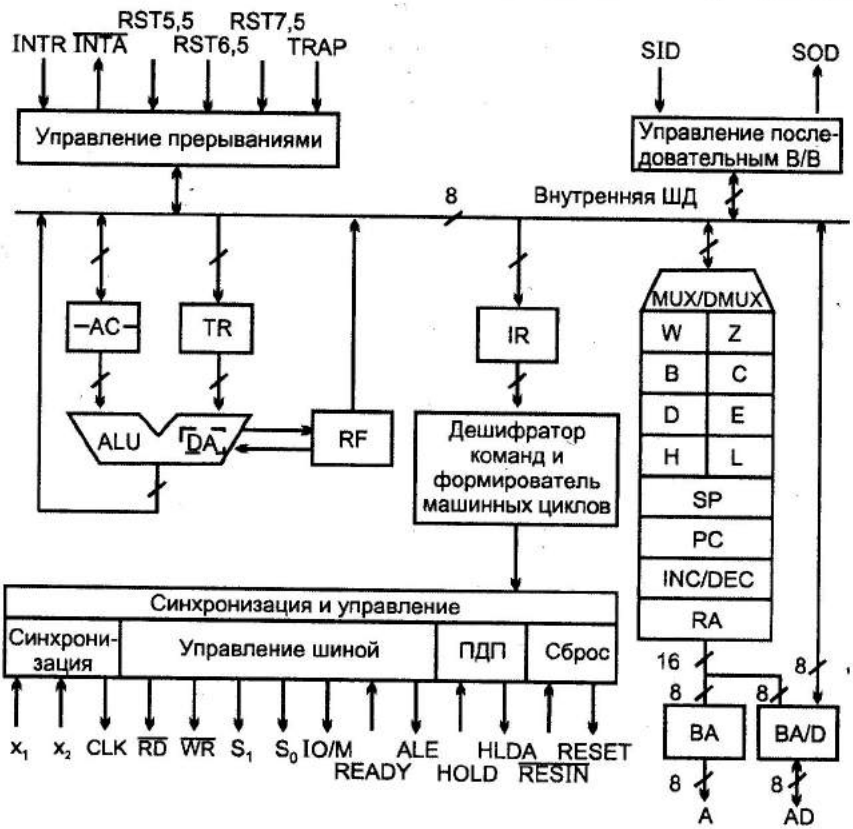


Рис. 5.5. Структура микропроцессора К1821ВМ85А

- ALU (Arithmetic-Logic Unit) — арифметико-логическое устройство (АЛУ), выполняющее действия над двумя словами-операндами, подаваемыми на его входы. Аккумулятор служит источником и приемником данных, TR — источником слова данных, хранимым на время выполнения операции. АЛУ функционирует согласно соотношению A:= A * B, где B хранится в TR, второй операнд поступает от аккумулятора, в него же поступает результат операции[1]. АЛУ непосредственно выполняет лишь операции сложения, вычитания, сдвига, сравнения слов, поразрядные логические операции (конъюнкцию, дизъюнкцию, сложение по модулю 2). Более сложные операции (умножение, деление и др.) выполняются по подпрограммам. В АЛУ имеется схема перевода двоичных чисел в двоично-десятичные (DA, Decimal Adjust);

[1] Звездочкой обозначен обобщенный символ операции.

- RF (Register Flags) — регистр флажков, т. е. битов, указывающих признаки результатов арифметических или логических операций, выполненных в АЛУ.

  Указываются пять признаков: Z (Zero) — нулевой результат, C (Carry) — перенос, AC (Auxiliary Carry) — вспомогательный перенос, S (Sign) — знак, P (Parity) — четность веса слова. Признак вспомогательного переноса (переноса между младшей и старшей тетрадами восьмиразрядного слова) нужен при выполнении операций в двоично-десятичном коде. Смысл остальных признаков ясен из их наименований. Признаки служат для управления ходом процесса обработки информации.

Блок регистров

С внутренней шиной данных через мультиплексор связан блок регистров, часть которых специализирована, другая часть (регистры общего назначения, РОН) программно доступна и может быть использована по усмотрению программиста. Регистры обозначены через W, Z, B, C, D, E, H, L, SP и PC. Регистры W и Z предназначены только для временного хранения данных при выборке команды из памяти и недоступны для программиста. Регистры B, C, D, E, H, L относятся к регистрам общего назначения, т. к. могут быть использованы по усмотрению программиста. Эти восьмиразрядные регистры могут применяться либо по отдельности, либо в виде пар B-C, D-E, H-L, играющих роль 16-разрядных регистров. Пары регистров именуются по первым регистрам пары как пары B, D, H. Пара H-L, как правило, используется для размещения в ней адресов при косвенной адресации. В блоке регистров имеются также 16-разрядные регистры SP и PC. Регистр SP (Stack Pointer) — указатель стека. Стек (магазинная память) удобен для запоминания массива слов, т. к. при этом не требуется адресовать каждое слово отдельно. Слова загружаются в стек в определенном порядке, при считывании также заранее известен порядок их следования. В частности, стек удобен при запоминании состояний регистров в момент прерывания программы. Порядок ввода слов в стек и их считывания предопределены его устройством. При организации типа LIFO (Last In — First Out) последнее записанное в стек слово при считывании появляется первым. Стек LIFO по порядку записи-считывания подобен стопке тарелок — для использования снимается верхняя, т. е. последняя положенная, затем вторая и т. д. Интересно отметить, что сам термин "стек" (stack) произошел именно от обозначения такой стопки.

Стек имеет дно и верхушку, направление возрастания номеров ячеек в нем может быть различным (обычный и перевернутый стеки). Операции со стеком — PUSH (запись слова) и POP (считывание слова).

Аппаратно стек реализуется в ОЗУ, где для него выделяется определенная область. Указатель стека SP содержит адрес последней занятой ячейки (рис. 5.6). При выполнении операций PUSH и POP значение SP уменьшается

или увеличивается. Задавая в SP начальное значение, можно размещать стек в той или иной области ОЗУ, следя при этом за тем, чтобы эта область не использовалась для других целей.

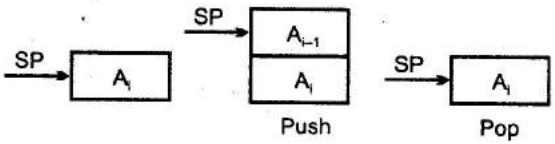


Рис. 5.6. Реализация стека в микропроцессорной системе

При байтовой организации памяти и занесении в стек содержимого регистровой пары старший байт запоминается по адресу SP − 1, а младший — по адресу SP − 2, содержимое SP уменьшается на 2. При выборке содержимое двух верхних ячеек стека помещается в соответствующие регистры, а содержимое SP увеличивается на 2.

Основное назначение стека — обслуживание прерываний программы и выполнения подпрограмм.

Программный счетчик PC (Program Counter) дает адрес команды и может обращаться в любую из 64K ячеек АП. При выполнении операции сброса микропроцессора PC принимает нулевое состояние, которое, таким образом, является адресом первой исполняемой команды, иначе говоря, выполнение программы начинается с нулевой ячейки. Длина команды составляет 1—3 байта. Содержимое программного счетчика после выборки очередного байта из памяти автоматически инкрементируется, так что в PC появляется адрес следующей команды, если текущая команда была однобайтовой, или следующего байта текущей команды в противном случае. Второй и третий байты команды поступают в регистры W и Z, которые не адресуются программой и используются только блоком внутреннего управления. Схема INC/DEC (Increment/Decrement) (см. рис. 5.5) изменяет передаваемые через нее слова на +1 или −1.

Регистр команд IR (Instruction Register) принимает из памяти первый байт команды, который после дешифрации порождает сигналы, необходимые для реализации машинных циклов, предписанных кодом операции.

Блок синхронизации и управления

Этот блок использует выход дешифратора команд и шифратора машинных циклов для синхронизации циклов, генерации сигналов состояния и управления шиной (внешними устройствами микропроцессорной системы).

При обмене между МП и памятью или ВУ адрес соответствующей ячейки памяти или ВУ от выбранной команды или одной из регистровых пар передается в регистр адреса RA (Register Address) (см. рис. 5.5). Буфер адреса BA с тремя состояниями выхода выдает сигналы старших разрядов адреса на линии адресной шины A_{15-8}. Буфер шины адресов/данных BA/D с тремя состояниями выхода передает с разделением во времени на шину AD младший байт адреса или байт данных.

Внутренняя восьмиразрядная шина данных передает байты между различными внутренними регистрами или обменивается с другими модулями МПС через мультиплексируемую шину адресов/данных.

Назначение блоков управления прерыванием и последовательным вводом/выводом ясно из их названий. Режимы прерывания и последовательного ввода/вывода подробнее рассмотрены далее.

При естественном следовании команд МП, начав работу, выбирает из памяти и выполняет одну команду за другой, пока не дойдет до команды "Останов" (HLT). Выборка и выполнение одной команды образуют *командный цикл*. Командный цикл состоит из одного или нескольких машинных циклов МЦ. Каждое обращение к памяти или ВУ требует машинного цикла, который связан с передачей байта в МП или из него. В свою очередь машинный цикл делится на то или иное число *тактов* Т, число которых зависит от типа машинного цикла.

Микропроцессор К1821ВМ85А имеет следующие типы машинных циклов:

- Выборки команды (OF, Opcode Fetch).
- Чтения из памяти (MR, Memory Read).
- Записи в память (MW, Memory Write).
- Чтения из ВУ (IOR, Input-Output Read).
- Записи в ВУ (IOW, Input-Output Write).
- Подтверждения прерывания (INA, Interrupt Acknowledge).
- Освобождения шин (BI, Bus Idle).
- Останов (HALT).

В начале каждого машинного цикла генерируются сигналы состояния, идентифицирующие тип цикла и действующие в течение всего цикла.

Функции выводов и сигналов микропроцессора

В этом разделе представлены функции выводов и сигналов микропроцессора Л1821ВМ85А.

- A_{15-8} — выходные линии с тремя состояниями для выдачи старшего байта адреса памяти или полного адреса ВУ. Переходят в третье состояние в режимах HOLD, HALT и RESET.

- AD_{7-0} — двунаправленные мультиплексированные линии с тремя состояниями для выдачи младшего байта адреса памяти или полного адреса ВУ в первом такте машинного цикла, после чего используются как шина данных. Как видно из сказанного, при адресации ВУ адресная информация обеих полушин (A_{15-8} и AD_{7-0}) дублируется.
- ALE (Address Latch Enable) — строб разрешения загрузки младшего байта адреса памяти во внешний регистр для его хранения в течение машинного цикла. Появляется в первом такте машинного цикла. Регистр загружается задним фронтом сигнала ALE.
- $\overline{RD}$, $\overline{WR}$ — микропроцессорные сигналы строба чтения и строба записи. Низкий уровень соответствующего сигнала свидетельствует о том, что адресованная ячейка памяти или внешнее устройство должны выполнить операцию чтения или записи. Выводы переходят в третье состояние в режимах HOLD, HALT и RESET.
- READY — входной сигнал, показывающий, что память или ВУ готовы к обмену с МП. Если готовности памяти или ВУ нет, МП входит в состояние ожидания, которое может длиться любое число тактов вплоть до появления единичного уровня сигнала READY.
- S_1, S_0 — сигналы состояния МП, сообщаемые внешней среде. Формируются в начале и сохраняются во время всего машинного цикла.
- IO/M — сигнал выбора памяти или внешнего устройства. При высоком уровне происходит обращение к ВУ, при низком — к памяти.

  Совместно с сигналами S_1S_0 сигнал IO/M идентифицирует тип машинного цикла. Сигналы состояния и управляющие сигналы $\overline{RD}$, $\overline{WR}$ и $\overline{INTA}$ для различных машинных циклов имеют значения, представленные в табл. 5.1.

Таблица 5.1

Тип МЦ	Сигналы состояния			Сигналы управления		
	IO/M	S_1	S_0	$\overline{RD}$	$\overline{WR}$	$\overline{INTA}$
OF	0	1	1	0	1	1
MR	0	1	0	0	1	1
MW	0	0	1	1	0	1
IOR	1	1	0	0	1	1
IOW	1	0	1	1	0	1
INA	1	1	1	1	1	0
BI	TC*	X	X	1	1	1
HALT	TC	0	0	TC	TC	1

* В приведенной таблице через TC обозначено третье состояние.

- x_1, x_2 — эти выводы присоединяются к кварцевому резонатору или другим частотно-задающим цепям для обеспечения работы внутреннего генератора синхроимпульсов МП. Частота на выводах x_1 и x_2 в 2 раза выше рабочей частоты.
- $\overline{\text{RESIN}}$ ($\overline{\text{RESET IN}}$) — вход сигнала сброса МП в начальное состояние. Сигнал может поступить в любое время по команде оператора. Автоматически формируется при включении питания. Под его воздействием сбрасываются регистры PC и IR, триггеры разрешения прерывания, подтверждения захвата и др.
- CLK (Clock) — выход синхроимпульсов для микропроцессорной системы. Частота этих импульсов в два раза ниже частоты на выводах x_1 и x_2.
- RESET — выходной сигнал сброса для внешних модулей системы, привязанный к тактовым импульсам CLK, т. е. отличающийся от сигнала $\overline{\text{RESIN}}$ по фазе.
- INTR (Interrupt Request) — вход запроса векторного прерывания, вызывающий генерацию строба $\overline{\text{INTA}}$, если прерывание разрешено программой. Адрес подпрограммы, вызываемой этим входом, выдается внешним устройством. При сбросе прием сигнала запрещается (прерывания запрещены).
- $\overline{\text{INTA}}$ (Interrupt Acknowledge) — микропроцессорный сигнал $\overline{\text{INTA}}$ выход строба подтверждения векторного прерывания после завершения текущего командного цикла. Используется для чтения вектора прерывания.
- RST 5,5. RST 6,5. RST 7,5 (Restart) — входы запросов радиального прерывания типа RSTn (n = 5,5. 6,5. 7,5). Начальные адреса подпрограмм обслуживания равны 8n. Приоритеты фиксированы, высший приоритет у входа RST 7,5. Приоритеты всей группы запросов выше приоритета запроса INTR. Запросы маскируемые, причем независимо друг от друга.
- TRAP — вход запроса немаскируемого прерывания, имеющий максимальный приоритет.
- SID, SOD (Serial Input Data, Serial Output Data) — вход и выход последовательной передачи данных. По команде RIM входной бит загружается в старший разряд аккумулятора, по команде SIM выводится из этого разряда.
- HOLD — сигнал запроса захвата шин. Формируется внешним устройством.
- HLDA (Hold Acknowledge) — сигнал подтверждения захвата. Является ответом на сигнал HOLD, формируемым в конце текущего машинного цикла. Свидетельствует об отключении МП от системных шин. При этом шины и линии управляющих сигналов $\overline{\text{RD}}$, $\overline{\text{WR}}$, IO/M и ALE переводятся в третье состояние.

Выводы x_1 и x_2, предназначенные для создания совместно с внутренними элементами МП генератора тактовых импульсов, могут быть использованы различными способами (рис. 5.7, *а*). Кварц может быть подключен непосредственно к выводам x_1 и x_2 как единственный частотно-задающий элемент. Если частота генератора составляет 4 МГц или более, могут понадобиться конденсаторы с рекомендованной емкостью 20 пФ для надежного запуска генератора. Параллельный LC-контур также может быть подключен непосредственно к выводам x_1 и x_2. При невысоких требованиях к стабильности частоты можно использовать частотно-задающую RC-цепочку. Возможна синхронизация от внешнего генератора ГТИ. При этом рекомендуется включать внешние логические элементы с открытым коллектором, причем при частоте генерации более 6 МГц включаются два логических элемента. Показанные на рис. 5.7, *а* параметры RC-цепи соответствуют частоте генерации 3 МГц.

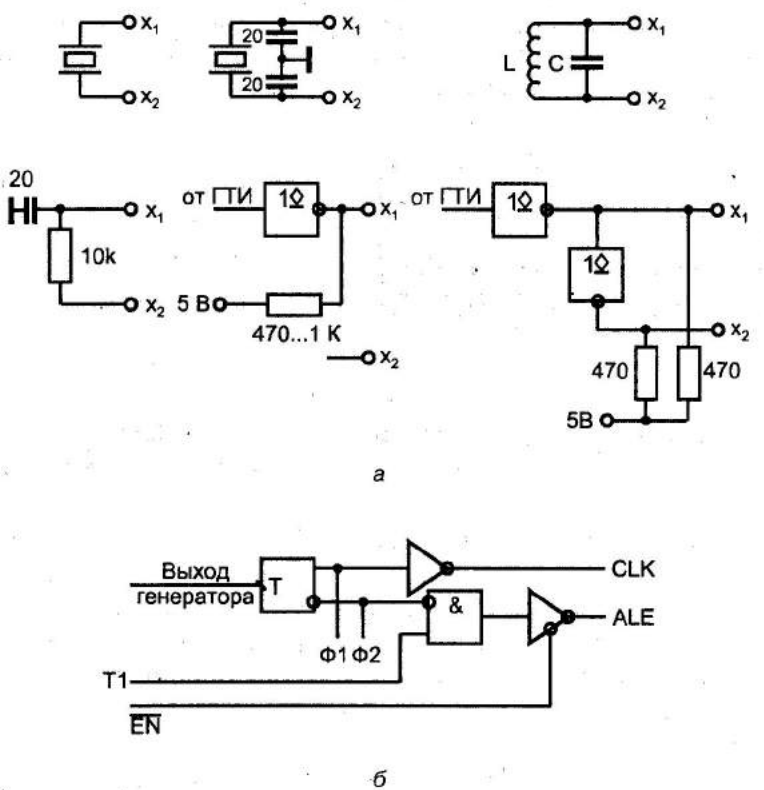


Рис. 5.7. Внешние элементы тактового генератора (*а*) и формирование синхросигналов (*б*) в микропроцессоре

Для образования сигналов синхронизации CLK выход генератора подается на вход счетного триггера (рис. 5.7, *б*). Триггер формирует две последовательности противофазных импульсов Ф1 и Ф2 для тактирования внутренних схем МП. Сигнал синхронизации системы CLK синфазен импульсам Ф2. Сигнал ALE формируется как один импульс последовательности Ф1, выделяемый из нее в первом такте (T1) каждого машинного цикла. Буфер выдачи сигнала ALE во внешние цепи имеет вход разрешения $\overline{EN}$. Частота синхросигналов МП в два раза ниже частоты генератора.

Синхронизация и последовательность действий МП

Командный цикл КЦ (рис. 5.8, *а*) начинается с выборки команды (Opcode Fetch, OF). Первый машинный цикл M1 всегда OF, в нем МП получает первый байт команды. После этого могут быть еще один или два машинных цикла типа MR (Memory Read), поскольку команда может быть однобайтной, двухбайтной или трехбайтной.

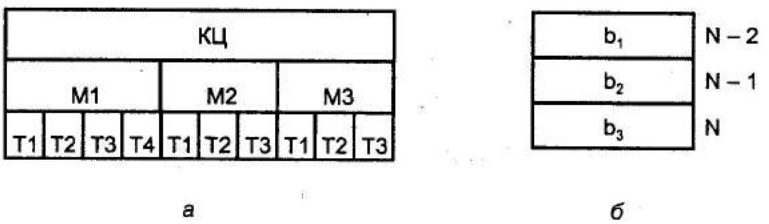


Рис. 5.8. Циклы и такты микропроцессора К1821ВМ85А (*а*) и пример размещения команды в памяти микропроцессорной системы (*б*)

Если команда трехбайтная, то она хранится в памяти так, как показано на рис. 5.8, *б*. Первый байт содержит код операции КОП, сведения о способе адресации, а если команда однобайтная, то и адрес операнда. Наличие адреса возможно для операций типа "регистр—регистр" с короткими адресами. Для адресации 8 регистров общего назначения достаточны трехразрядные адреса, а для адресации регистровых пар даже двухразрядные. Второй байт содержит младший полуадрес операнда, если команда трехбайтная, или непосредственный операнд либо адрес ВУ, если команда двухбайтная. Третий байт содержит старший полуадрес операнда или байт непосредственных данных при загрузке пары регистров. Адреса регистров и регистровых пар даны в табл. 5.2.

Таблица 5.2

Регистры							Пары регистров			
B	C	D	E	H	L	A	B	D	H	SP
000	001	010	011	100	101	111	00	01	10	11

После выборки и декодирования команды могут понадобиться дополнительные машинные циклы для ее выполнения. Всего в командном цикле может быть от одного до пяти машинных циклов.

Машинный цикл состоит из тактов, в которых выполняются типовые действия, рассмотренные далее. Число тактов в различных машинных циклах — 3...6. Большинство машинных циклов содержат три такта.

В командном цикле может содержаться от 4 до 18 тактов.

Сигналы, реализующие тот или иной МЦ, генерируются блоком управления МП на основании информации, содержащейся в первом байте команды.

Проиллюстрируем сказанное примером выполнения команды STA b_3b_2 (Store Accumulator Direct), передающей содержимое аккумулятора в ячейку памяти при прямой адресации, т. е. указании адреса ячейки в самой команде. Команда трехбайтная, для ее получения МП требуются три машинных цикла, в первом из которых байт b_1 передается в регистр команд IR, в последующих байты b_2 и b_3 передаются в регистры временного хранения W и Z. После получения всей команды МП выполняет ее, передавая байт из аккумулятора в ячейку памяти, адрес которой поступил в МП. Таким образом, цикл команды составится из четырех машинных циклов в следующем порядке OF—MR—MR—MW.

Каждый машинный цикл делится на *такты* (состояния) — интервалы между одноименными фронтами тактовых импульсов.

Типовые действия, выполняемые в тактах машинного цикла:

T1 Адрес памяти или ВУ выставляется на AD_{7-0} и A_{15-8}, генерируется сигнал ALE для фиксации битов AD_{7-0}. На линиях IO/M, S_1 и S_0 выставляется информация, определяющая тип цикла. Проверяется флаг HALT.

T2 Проверяются входы Ready и Hold. Программный счетчик инкрементируется, если данный машинный цикл есть часть выборки команды. Во всех машинных циклах кроме цикла BI (освобождения шин) один из управляющих стробов $\overline{RD}$, $\overline{WR}$ или $\overline{INTA}$ переходит из единичного состояния в активное нулевое.

T_W Появляется при неготовности памяти или ВУ к обмену (на линии READY низкий уровень напряжения). Состояния линий адресов, данных и управления остаются теми же, что и в конце такта. Сигнал READY проверяется в каждом такте ожидания.

T3 Байт команды или данных передается в микропроцессор или из него. Уровень активного управляющего строба изменяется с нулевого на единичный.

T4 Декодируется содержимое регистра команд.

T5, T6 Используются при необходимости для завершения некоторых команд. Системные шины не используются.

Машинный цикл всегда содержит такты T1—T3, иногда имеет большее число тактов, но для чтения или записи требуется только три такта. Временные диаграммы цикла чтения с тактом ожидания приведены на рис. 5.9.

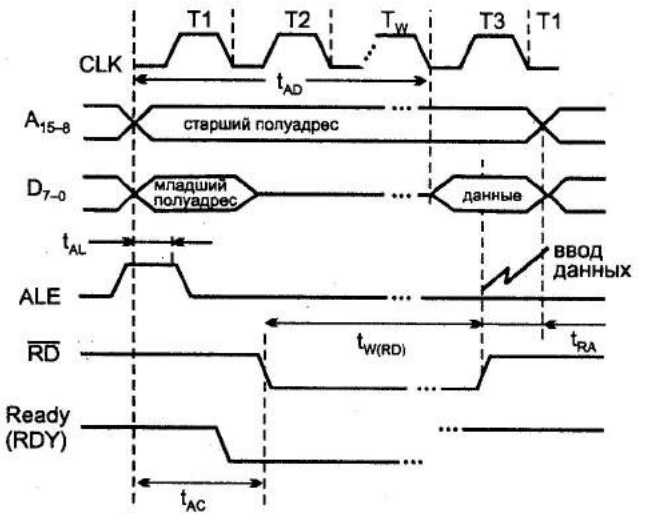


Рис. 5.9. Временные диаграммы цикла чтения микропроцессора

Система прерываний

При работе микропроцессорной системы в ней или вне ее могут произойти события, требующие немедленной реакции. Такая реакция обеспечивается прерыванием программы и переходом к обслуживанию источников запросов на прерывание. Внутри процессора запросы возникают при сбоях в работе, переполнении разрядной сетки, попытке деления на нуль и т. д. Ситуации подобного типа, связанные с ошибками в работе процессора, называются *исключениями*. Штатные ситуации, в которых запросы формируются внешними сигналами, называют *аппаратными прерываниями*. Если же запрос формируется командами программы, то говорят о *программных прерываниях*.

Аппаратные прерывания возникают, в частности, при требованиях обслуживания от внешних устройств. Извне могут поступать также сигналы аварийных ситуаций в управляемых объектах, неисправности источников питания и др.

Прерывания по запросам от медленно действующих внешних устройств увеличивают производительность системы, позволяя ВУ занимать время процессора только при их готовности к обмену. Когда ВУ нуждается в обслуживании, оно устанавливает триггер запроса прерывания, и сигнал запроса сохраняется, пока не будет воспринят и обработан процессором. В ответ на принятый запрос прерывания в микропроцессорной системе завершается выполнение текущей команды, запоминается состояние МП, выполняется подпрограмма обслуживания прерывания, восстанавливается состояние МП и затем возвращается управление очередной команде основной программы.

Микропроцессор К1821ВМ85А имеет пять входов прерывания и один выход управления им ($\overline{\text{INTA}}$). Прерывание должно ввести в действие команду CALL, согласно которой состояние программного счетчика PC передается в стек, а в PC загружается адрес подпрограммы, подлежащей выполнению. Инициатива ввода команды CALL принадлежит аппаратным средствам микропроцессорной системы. Если прерывания разрешены, то они осуществляются процессором в конце выполнения текущей команды.

Входы МП, связанные с прерываниями, называются TRAP; RST 5,5; RST 6,5; RST 7,5; INTR. При организации прерываний решаются задачи маскирования запросов и определяются их уровни приоритета при конфликтах из-за одновременного поступления нескольких запросов.

Маскирование состоит в запрещении действия соответствующего входа. Входы запросов прерывания могут быть маскируемыми или не маскируемыми, т. е. принимаемыми всегда.

Вход TRAP является немаскируемым и имеет наивысший приоритет. Он не может быть запрещен командами программы. К этому входу подключают сигналы, оповещающие о наиболее важных событиях в микропроцессорной системе, появление которых требует безусловной реакции (например, сигнал, оповещающий об аварии питания, требующей принятия немедленных мер).

Начальный адрес подпрограммы обслуживания прерывания TRAP размещен в фиксированной ячейке памяти с адресом 24H. Таким образом, появление запроса прерывания по входу TRAP независимо ни от чего вызовет соответствующее прерывание после завершения выполнения текущей команды.

Обозначение входов RSTn (n = 5,5; 6,5; 7,5) происходит от слова Restart. Прерывания по этим входам маскируемые, т. е. могут быть разрешены или запрещены командами EI (Enable Interrupt) и DI (Disable Interrupt), действующими на все три входа одновременно. Начальный сброс микропроцес-

сора запрещает обслуживание этих запросов, для их последующего разрешения следует подать команду EI. Имеется также возможность раздельного маскирования запросов RSTn с помощью специальной команды SIM (Set Interrupt Mask), по которой маски устанавливаются в соответствии со значениями битов A_0—A_2 содержимого аккумулятора. Загрузив предварительно 1 в соответствующий бит, можно запретить (замаскировать) тот или иной вход. Приоритеты входов RSTn фиксированы, они снижаются в порядке RST 7,5; RST 6,5; RST 5,5. Начальные адреса подпрограмм обслуживания прерываний типа RSTn известны. Команды RSTn заканчиваются загрузкой в программный счетчик числа 8n. Цифры 5,5; 6,5 и 7,5 определяют начальные адреса 002CH, 0034H и 003CH. Иными словами, для входов этого типа векторы прерывания определяются автоматически и их не требуется передавать в МП из внешних устройств.

Напомним, что вектором прерываний называют информацию, необходимую для перехода к соответствующей подпрограмме обслуживания, в простейшем случае это просто начальный адрес прерывающей подпрограммы.

Вход RST 7,5 является динамическим, реагирует на положительный фронт сигнала, а входы RST 6,5 и RST 5,5 — статические, реагируют на уровень сигнала и, следовательно, автоматически снимаются при исчезновении запросов по этим входам. Запрос RST 7,5, принимаемый триггером с динамическим входом, после снятия сигнала запроса не снимается и сохраняется, пока не будет обработано прерывание или до появления команд SIM или RESET.

При поступлении запроса по входу INTR (Interrupt) вектор прерывания должен быть передан в МП извне. К этому входу, в частности, подключают контроллер прерываний — блок, который воспринимает несколько запросов от внешних устройств, решает задачу приоритетности и маскирований и вырабатывает для МП единственный сигнал INTR, с пересылкой в МП соответствующего вектора прерывания. В данном случае также выполняется команда RSTn, но n зависит от источника прерываний. Аппаратный ввод байта в ответ на запрос INTR может быть реализован, например, согласно рис. 5.10. Появление запроса INTR при разрешенных прерываниях ведет к ответу микропроцессора сигналом $\overline{\text{INTA}}$, во время действия которого на шине AD появляется вводимый байт.

Сигнал $\overline{\text{INTA}}$ при этом поступает на входы разрешения выхода буферных усилителей $\overline{\text{OE}}$.

Во время обработки прерываний, пока не выполнится команда EI, запрещаются другие прерывания кроме TRAP. Немаскируемое же прерывание TRAP блокирует другие прерывания, но сохраняет состояние разрешения поступившего уже сигнала прерывания.

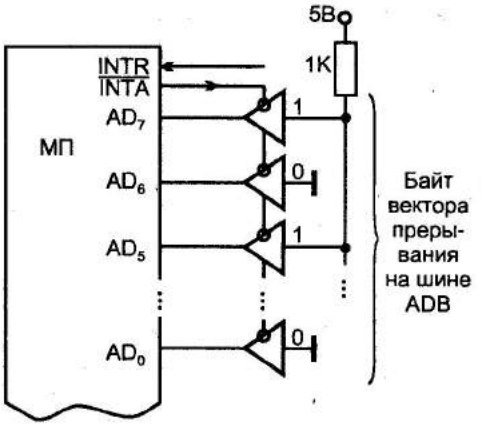


Рис. 5.10. Аппаратная реализация пересылки байта при выполнении операции рестарта

Последовательный ввод/вывод

Микропроцессор имеет два вывода для передач последовательных данных: SOD и SID (Serial Output Data и Serial Input Data).

Вывод SOD управляется командой SIM, а сигнал с вывода SID считывается командой RIM. Эти команды упоминались ранее как команды установки и сброса масок для входов прерываний RSTn, они же используются и для управления последовательным вводом/выводом.

До выполнения команды SIM в аккумуляторе формируется слово, биты которого интерпретируются следующим образом:

7	6	5	4	3	2	1	0
SOD	SOE	X	R 7,5	MSE	M 7,5	M 6,5	M 5,5

где SOD — последовательный выход данных, SOE (Serial Input Enable) — сигнал, единичное значение которого передает последовательные данные SOD на соответствующий выход микропроцессора, бит 5 не используется, R 7,5 сбрасывает вход RST 7,5 (напомним, что сигнал по этому входу принимается триггером с динамическим управлением), MSE (Mask Set Enable) — сигнал, активное состояние которого разрешает действие битов 2—0, биты M 7,5—M 5,5 маскируют запросы RST 7,5—RST 5,5, если соответствующий бит имеет единичное значение.

Например, установка SOD = 1, разрешение RST 6,5, сброс триггера RST 7,5 и маскирование RST 7,5 и RST 5,5 будут выполнены двумя командами по программе:

```
MVI A, b2      ; установка битов аккумулятора
SIM            ; изменение масок и бита SOD
```

Команда MVI A, b_2 передает в аккумулятор байт b_2, т. е. выполняет действие (A) ← (b_2) пересылки в аккумулятор данных при непосредственной адресации. Байт b_2 в данном случае имеет вид: 1 1 X 1 1 1 0 1.

Для ввода последовательных данных через контакт SID используется команда RIM, обеспечивающая ввод последовательных данных и чтение масок прерывания. После выполнения команды RIM в аккумуляторе фиксируется слово со следующим значением битов:

7	6	5	4	3	2	1	0
SID	I 7,5	I 6,5	I 5,5	IE	M 7,5	M 6,5	M 5,5

где SID — последовательные данные ввода через контакт SID; I 7,5; I 6,5; I 5,5 — логические уровни на выводах RST 7,5; RST 6,5 и RST 5,5 соответственно, IE — сигнал разрешения прерывания, M 7,5...M 5,5 — логические уровни масок.

Биты I 7,5...I 5,5 индицируют уровни во время команды RIM. Бит IE показывает, какая из команд EI и DI выполнялась последней, на него влияет также наличие в данное время режима прерывания, поскольку он сопровождается сбросом триггера IE, запрещая другие прерывания. Биты M 7,5...M 5,5 индицируют текущие состояния масок прерывания.

Система команд МП

Команды МП приведены в табл. 5.3. В первой графе таблицы даны мнемокоды команд с обозначениями регистров через r, пар регистров через r_p, ячеек памяти через M, третьего и второго байтов команды через b_3b_2, адресов ВУ через port. Ссылки на ячейки памяти M подразумевают косвенную адресацию — адреса этих ячеек берутся из регистровой пары H (регистров H и L) и, следовательно, не нуждаются в указании в самой команде. При описании команд, кроме приведенных в табл. 5.3 данных, обычно указываются также машинные коды команд, длины команд в байтах, число тактов, необходимых для выполнения команды, и число составляющих команду машинных циклов. Эти данные здесь не даются (в первом издании этой книги такие данные имеются).

Таблица 5.3

Мнемокод	Операция
Команды пересылки	
MOV r_1, r_2	Пересылка из регистра r_2 в регистр r_1
MOV M, r	Пересылка из регистра в память
MOV r, M	Пересылка из памяти в регистр
MVI r, b_2	Пересылка непосредственных данных в регистр
MVI M, b_2	Пересылка непосредственных данных в память

Таблица 5.3 *(продолжение)*

Мнемокод	Операция
Команды пересылки	
LXI $r_p b_3 b_2$	Загрузка непосредственных данных в пару регистров
LDA $b_3 b_2$	Прямая загрузка аккумулятора
STA $b_3 b_2$	Прямая запись аккумулятора в память
LHLD $b_3 b_2$	Прямая загрузка пары регистров H
SHLD $b_3 b_2$	Прямая запись пары регистров H в память
LDAX r_p	Косвенная загрузка аккумулятора посредством пары регистров B или D
STAX r_p	Косвенная запись аккумулятора в память посредством пары регистров B или D
XCHG	Обмен между парами регистров H и D
Команды арифметических и логических операций	
ADD r	Сложение регистра и аккумулятора
ADD M	Сложение памяти и аккумулятора
ADI b_2	Сложение непосредственных данных и аккумулятора
ADC r	Сложение регистра и аккумулятора с переносом
ADC M	Сложение памяти и аккумулятора с переносом
ACI b_2	Сложение непосредственных данных и аккумулятора с переносом
SUB r	Вычитание регистра из аккумулятора
SUB M	Вычитание памяти из аккумулятора
SUI b_2	Вычитание непосредственных данных из аккумулятора
SBB r	Вычитание регистра из аккумулятора с заемом
SBB M	Вычитание памяти из аккумулятора с заемом
SBI b_2	Вычитание непосредственных данных из аккумулятора с заемом
INR r	Инкремент регистра
INR M	Инкремент памяти
DCR r	Декремент регистра
DCR M	Декремент памяти
INX r_p	Инкремент пары регистров
DCX r_p	Декремент пары регистров
DAD r_p	Сложение регистровой пары H с регистровой парой
DAA	Преобразование аккумулятора в двоично-десятичный код
ANA r	Логическое И регистра и аккумулятора
ANA M	Логическое И памяти и аккумулятора

Таблица 5.3 *(продолжение)*

Мнемокод	Операция
Команды арифметических и логических операций	
ANI b_2	Логическое И непосредственных данных и аккумулятора
XRA r	Исключающее ИЛИ регистра и аккумулятора
XRA M	Исключающее ИЛИ памяти и аккумулятора
XRI b_2	Исключающее ИЛИ непосредственных данных и аккумулятора
ORA r	Логическое ИЛИ регистра и аккумулятора
ORA M	Логическое ИЛИ памяти и аккумулятора
ORI b_2	Логическое ИЛИ непосредственных данных и аккумулятора
CMP r	Сравнение регистра и аккумулятора
CMP M	Сравнение памяти и аккумулятора
CPI b_2	Сравнение непосредственных данных и аккумулятора
CMA	Инвертирование аккумулятора
STC	Установка флажка переноса
CMC	Инвертирование флажка переноса
RLC	Циклический сдвиг аккумулятора влево
RRC	Циклический сдвиг аккумулятора вправо
RAL	Циклический сдвиг аккумулятора влево через разряд переноса
RAR	Циклический сдвиг аккумулятора вправо через разряд переноса
Команды управления	
JMP b_3b_2	Безусловный переход
$J_{усл}$ b_3b_2	Условный переход
CALL b_3b_2	Безусловный вызов подпрограммы
$C_{усл}$ b_3b_2	Условный вызов подпрограммы
RET	Возврат
$R_{усл}$	Возврат при условии
RST n	Повторный запуск
SPHL	Пересылка пары регистров H в SP
Специальные команды	
PUSH r_p	Пересылка пары регистров в стек
PUSH PSW	Пересылка аккумулятора и регистра флажков в стек
POP r_p	Загрузка регистровой пары из стека
POP PSW	Загрузка аккумулятора и регистра флажков из стека
XTHL	Обмен между регистровой парой H и стеком

Таблица 5.3 (окончание)

Мнемокод	Операция
Специальные команды	
PCHL	Пересылка регистровой пары H в PC
IN port	Ввод
OUT port	Вывод
EI	Разрешение прерывания
DI	Запрещение прерывания
HLT	Останов
NOP	Нет операции
RIM	Чтение маски прерывания
SIM	Запись маски прерывания

Представленные в таблице команды с буквенными обозначениями источников и приемников данных являются обобщенными. Подставляя вместо буквенных символов определенные адреса, получим конкретные варианты команды (например, из обобщенной формы "пересылка из регистра в регистр" конкретный вариант "пересылка из регистра B в регистр D"). В некоторых командах управления содержатся индексы "усл". В этих случаях переход от обобщенной команды к конкретному варианту осуществляется введением в двоичный машинный код команды соответствующих условию значений трех специально выделенных разрядов. Например, команда условного перехода из обобщенной формы $J_{усл}\, b_3b_2$ введением в указанные разряды комбинации 000 переводится в вариант JNZ b_3b_2, определяющий переход к команде с адресом b_3b_2, если признак результата говорит о том, что результат не равен нулю. Введение в те же разряды кода 001 определит команду JZ b_3b_2 условного перехода при нулевом результате и т. д.

Признаки результатов операции формируются в регистре флажков RF, формат которого представляется в виде:

7	6	5	4	3	2	1	0
S	Z	0	AC	0	P	1	C

причем S = 0 означает "плюс", S = 1 — "минус", Z = 0 — неравенство нулю, Z = 1 — равенство нулю, C или AC = 1 — наличие переноса, C или AC = 0 — его отсутствие, P = 0 — нечетность, P = 1 — четность. Разряды 5, 3, 1 содержат константы и для признаков не используются.

В коде команды рестарта RST три разряда, отмеченные буквами nnn, формируются системой прерываний или указываются программистом. При вы-

полнении команды текущее содержимое программного счетчика PC загружается в стек, а в PC формируется код с нулевым старшим байтом и младшим байтом вида 00nnn000.

Операция сравнения производится вычитанием операндов с установкой признака результата (Z = 1 — равные операнды, S = 0 — содержимое аккумулятора больше второго операнда, S = 1 — меньше).

Признаки результатов фиксируются в регистре флажков при выполнении арифметических и логических операций, причем в виде полного либо частичного набора (например, без флага переноса либо только для флага переноса). При выполнении команд пересылки, команд управления и специальных команд флажки в регистре RF не устанавливаются.

Команды RLC, RRC, RAL и RAR реализуют циклические (кольцевые) сдвиги содержимого аккумулятора на один разряд в ту или иную сторону без включения (RLC и RRC) или с включением (RAL и RAR) в кольцо разряда C регистра флажков (рис. 5.11).

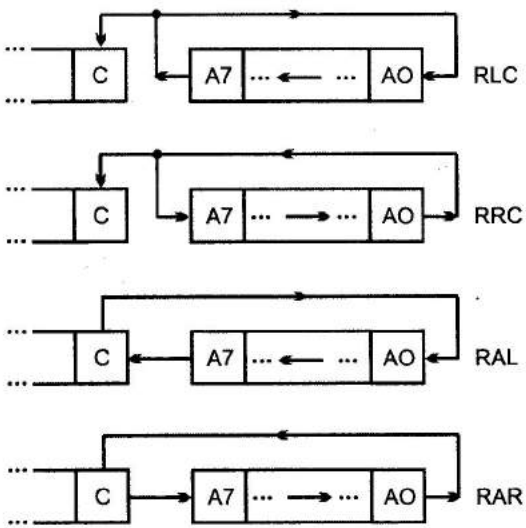


Рис. 5.11. Схемы, поясняющие выполнение сдвигов микропроцессором

Команды RIM и SIM подробно рассмотрены ранее. Два возможных значения чисел тактов и циклов приведены для команд, выполнение которых зависит от состояния признаков — флажков.

В табл. 5.4 на примере микропроцессоров фирмы Intel приведены сравнительные параметры двух микропроцессоров, "возраст" одного (8085) составляет около 25 лет, а второй (Pentium 4) выпущен сравнительно недавно. Для

микропроцессора семейства 8085 указан год выпуска базовой модели, параметры же указаны для модифицированного варианта 8085АН согласно каталогу продукции фирмы 2002 года.

Таблица 5.4

Параметр	8085АН	Pentium 4
Год выпуска	1977	2000
Разрядность	8	32
Адресное пространство	64К	64Г
Частота тактовых импульсов, МГц	6	1500
Среднее число тактов на операцию	≈10	0,5
Число выводов корпуса	40	420
Число транзисторов, млн	0,0062	42

Обращаясь к перспективным планам ведущих фирм по выпуску новых микропроцессоров, можно считать приведенные для процессора Pentium 4 цифры достаточно скромными. В частности, фирма Intel заявляет: "На наших заводах уже *возможно производство кристаллов с 1 млрд транзисторов*".

По мере развития микропроцессорной техники происходит естественный процесс специализации МП соответственно областям их применения. Наряду с микропроцессорами общего назначения стал интенсивно развиваться класс проблемно ориентированных МП — *процессоры цифровой обработки сигналов*, которые находят применение в современных системах связи, обработки графических изображений, медицине и многих других областях. Особое место среди микропроцессоров занимают *микроконтроллеры*, рассматриваемые в *главе 7*. Микроконтроллеры также специализируются, так, например, среди них выделился класс *коммуникационных* контроллеров. Сведения о различных микропроцессорах и микроконтроллерах, в частности, можно почерпнуть в работах [11], [18], [27], [34].

Пример выполнения команды

Управление модулями реализуется в МПС посредством ее шин. Процессор расшифровывает команду и задает на шинах такие последовательности потенциалов, которые заставляют управляемые им блоки выполнять требуемые действия. Для иллюстрации рассмотрим выполнение короткого фрагмента программы передачи байта из одной ячейки памяти в другую. Пусть численное значение байта будет 10Н, а его передача производится из ячейки 0100Н в ячейку 0101Н. Пусть также фрагмент программы размещается в памяти, начиная с ячейки 2000Н.

Для выполнения фрагмента сначала нужно переслать байт в аккумулятор, а затем из аккумулятора в память. Так как обращение к памяти подразумевает косвенную адресацию, вначале требуется загрузка пары регистров H адресом ячейки, к которой идет обращение. С учетом сказанного фрагмент программы в мнемокодах (на ассемблере МП) примет вид, показанный в левом столбце.

LXI H, 0100H	2000 21 00 01
MOV A, M	2003 7E
INX H	2004 23
MOV M, A	2005 77

Команда загрузки непосредственных данных в пару регистров LXI $r_p b_3 b_2$ имеет код 00ПР0001, где ПР — адрес пары регистров. Пара регистров H имеет адрес ПР = 10. Подставив это значение в код команды, получаем код 21. В правом столбце записана команда в кодах. Она имеет вид: 21 00 01, т. к. после кода операции из памяти извлекаются сначала младший (00), а затем старший (01) байты. Команда трехбайтная и занимает 2000—2002 ячейки памяти.

Однобайтная команда MOV A, M пересылки из памяти в аккумулятор является вариантом команды MOV r, M с кодом 01ППП110, где ППП — адрес аккумулятора. Подставив в этот код адрес регистра A = 111, получаем код команды 7E.

Команда INX H прибавляет единицу к содержимому регистровой пары и является вариантом, код которого получается из кода 00ПР0011 при подстановке адреса пары регистров 10, что дает код 23.

Последняя команда фрагмента программы (пересылка из аккумулятора в память) MOV M, A, имеющая код 77, передает в ячейку памяти, адрес который находится в регистровой паре H, содержимое аккумулятора. Эта команда завершает выполнение фрагмента программы.

Последовательность четырех рассмотренных команд сгенерирует временные диаграммы (рис. 5.12), посредством которых программа будет выполнена.

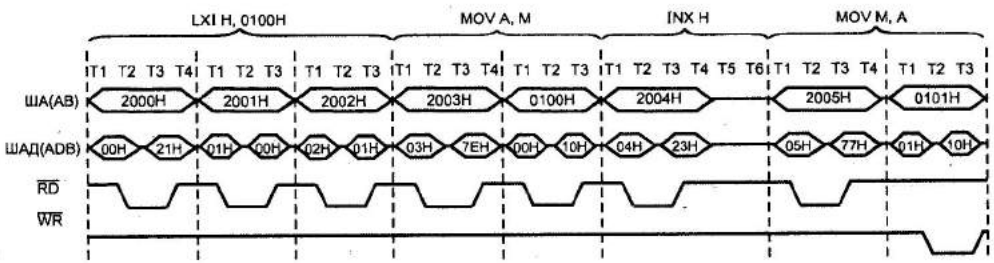


Рис. 5.12. Реализация программы сигналами шин микропроцессорной системы

§ 5.4. Схемы подключения памяти и внешних устройств к шинам микропроцессорной системы

Управление памятью и внешними устройствами рассмотрено с общих позиций в § 5.2. Проиллюстрируем теперь общие принципы примерами конкретных решений. Приводимые далее примеры ориентированы на работу процессора с асинхронными микросхемами памяти, которые были в течение длительного времени доминирующими в самых разнообразных применениях, чего сейчас уже нет. В современных высокопроизводительных системах, таких как компьютеры, работающие на частотах в сотни мегагерц, находят применение синхронные микросхемы памяти с более сложными механизмами управления процессами чтения и записи. Тем не менее, рассмотрение принципов работы процессора с асинхронными модулями памяти представляется целесообразным, т. к. на простых решениях показывает основные проблемы обеспечения правильного интерфейса между процессором и памятью.

При проектировании схем подключения микросхем памяти к микропроцессору решаются следующие задачи:

- разработка схем адресации памяти и формирования управляющих сигналов на функционально-логическом уровне;
- анализ нагрузочных условий в полученной схеме, обеспечение рабочих режимов для выходов с открытым коллектором (стоком) и введение при необходимости буферных элементов для устранения перегрузок источников сигналов;
- согласование временных диаграмм микропроцессора и микросхем памяти.

При адресации памяти размещают адреса постоянных и оперативных ЗУ в заданных областях адресного пространства.

Примечание

Для краткости адреса в АП обычно выражают в шестнадцатиричной системе счисления, информационные емкости памяти, как правило, оцениваются величинами K = 2^{10} или M = 2^{20}. Наиболее привычна для человека десятичная система счисления, а в цифровой технике используется двоичная. Поэтому при работе приходится переходить от одних единиц к другим. В задачах адресации памяти чаще всего встречаются преобразования значений в числах K или M в шестнадцатиричные и обратно. Для облегчения таких переходов удобно пользоваться табл. 5.5 (числа в первой строке измеряются в единицах K).

Таблица 5.5

N_k	1/1024	1/512	1/256	1/128	1/64	1/32	1/16
N_{16}	1	2	4	8	10	20	40
N_k	1/8	1/4	1/2	1	2	4	8
N_{16}	80	100	200	400	800	1000	2000
N_k	16	32	64	128	256	512	1024
N_{16}	4000	8000	10000	20000	40000	80000	100000

Пример 1

Типовым элементом схем адресации является дешифратор, в котором используются как информационные, так и разрешающие входы. На рис. 5.13 приведена схема адресации ПЗУ, составленного из трех субмодулей с организацией 4K×8. Адреса занимают 12K в верхней части АП, т. е. зону от 0000H до 2FFFH (последний адрес легко вычислить, пользуясь таблицей, приведенной выше, как сумму шестнадцатиричных значений, соответствующих выражению (8K + 4K − 1)).

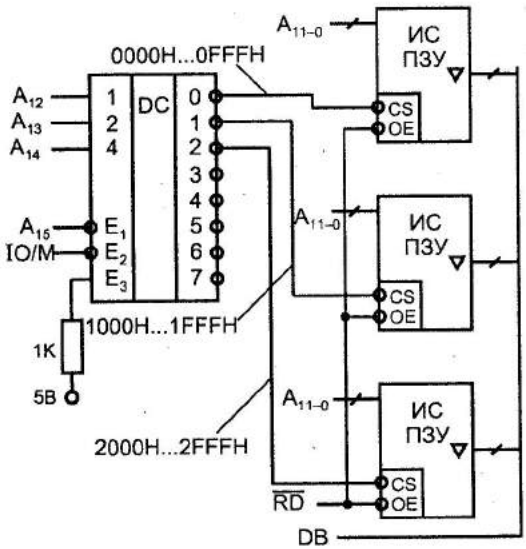


Рис. 5.13. Пример адресации модуля памяти

Сигнал разрешения работы дешифратора $E = \overline{E}_1\overline{E}_2E_3$. Двенадцать младших разрядов адреса выбирают ячейку в субмодуле. Старшие разряды адреса декодируются для формирования сигналов выбора кристалла $\overline{CS}$. Стробирующим сигналом $\overline{RD}$ определяется интервал выполнения операции чтения.

Одним из условий разрешения работы дешифратора является низкий уровень сигнала IO/M.

Пример 2

Адресация модуля памяти, составленного из субмодулей с организацией 2K×8 при размещении адресов в зоне АП, начинающейся с адреса 8000H, может использовать дешифратор, включенный, как показано на рис. 5.14. Если адрес находится в пределах 8000H...BFFFH, то работа дешифратора разрешена, т. к. этим пределам отвечают условия $A_{15} = 1$ и $A_{14} = 0$. Область АП, лежащая в указанных пределах, в зависимости от значений битов A_{13}—A_{11} делится на части по 0800H адресов в каждой (0800H = 2K). Каждый из выходов дешифратора сигналом $\overline{CS}$ может выбирать ЗУ с числом хранимых слов 2K. Линии адреса A_{10-0} адресуют ячейки на кристалле.

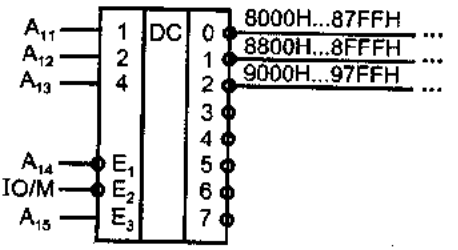
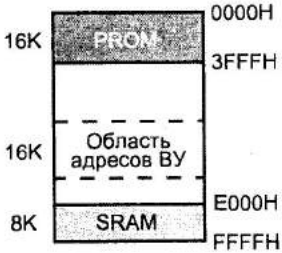


Рис. 5.14. Вариант адресации модуля памяти

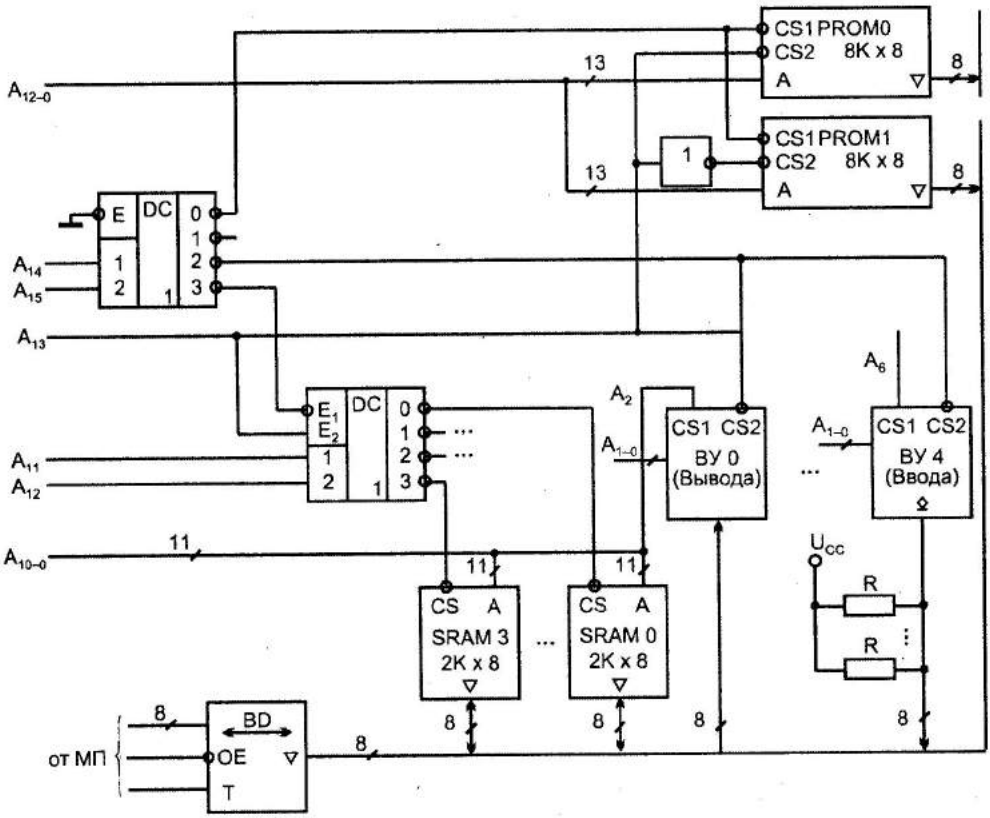
Пример 3

Рассмотрим пример размещения в АП адресов ПЗУ, ОЗУ и ВУ, т. е. совмещенный ввод/вывод. Для памяти используем абсолютную адресацию, а для ВУ — линейную селекцию. Пусть для ПЗУ отведено 16K адресов в начале АП, адреса ВУ занимают третью четверть АП, а адреса ОЗУ занимают последние 8K адресного пространства. Примем, что в системе имеется 5 ВУ, каждое из которых имеет 4 внутренних регистра со своими адресами, а в качестве ОЗУ используется триггерное тактируемое ЗУ. Распределение АП показано на рис. 5.15, *а*.

Пусть ПЗУ строится на микросхемах с организацией 8K×8, а ОЗУ на микросхемах 2K×8. Имея в виду байтовую организацию модуля памяти, видим, что каждая микросхема играет роль субмодуля (не нуждается в наращивании разрядности хранимых слов). Для адресации ВУ используем младшие разряды шины адреса, число которых определяется как N + 2, где N — число ВУ, а две линии нужны для адресации их внутренних регистров.



а



б

Рис. 5.15. Пример распределения адресного пространства между модулями памяти и внешними устройствами (*а*) и схема адресации модулей памяти и внешних устройств (*б*)

Таблица 5.6

Вид объекта	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
PROM0	0	0	0	d	d	d	d	d	d	d	d	d	d	d	d	d
PROM1	0	0	1	d	d	d	d	d	d	d	d	d	d	d	d	d
BY0	1	0	X	X	X	X	X	X	X	0	0	0	0	1	d	d
BY1	1	0	X	X	X	X	X	X	X	0	0	0	1	0	d	d
BY2	1	0	X	X	X	X	X	X	X	0	0	1	0	0	d	d
BY3	1	0	X	X	X	X	X	X	X	0	1	0	0	0	d	d
BY4	1	0	X	X	X	X	X	X	X	1	0	0	0	0	d	d
SRAM0	1	1	1	0	0	d	d	d	d	d	d	d	d	d	d	d
SRAM1	1	1	1	0	1	d	d	d	d	d	d	d	d	d	d	d
SRAM2	1	1	1	1	0	d	d	d	d	d	d	d	d	d	d	d
SRAM3	1	1	1	1	1	d	d	d	d	d	d	d	d	d	d	d

Схема адресации, соответствующая таблице адресов (табл. 5.6), приведена на рис. 5.15, *б*. Дешифратор DC1 делит АП на четыре части, его выходы разрешают работу тем объектам адресации, которые расположены в соответствующей четверти АП. Линия A_{13} разрешает работу микросхемы PROM0 в первой половине первой четверти АП при нулевом состоянии и работу микросхемы PROM1 — при единичном. Линии A_2—A_6 использованы для линейной селекции внешних устройств, а линии A_{12} и A_{11} декодируются дешифратором DC2 для разрешения работы микросхемам SRAM3—SRAM0 в их зонах адресов.

Символом X обозначены безразличные состояния адресных разрядов, а буквой d — разряды, "декодируемые на кристалле", т. е. входящие в состав адресных входов самих микросхем памяти или адресных линий внутренних регистров ВУ.

Пример 4

С целью упрощения схем декодирования и при наличии "лишнего" адресного пространства можно применить неабсолютную адресацию, при которой каждому объекту адресации присваивается не один-единственный адрес, а группа адресов (некоторая зона АП).

Пусть, например, в АП емкостью 64К требуется разместить всего два субмодуля памяти по 8К адресов в каждом. При абсолютной адресации (рис. 5.16, *а*) на адресные входы самих ИС памяти поступают 13 младших

разрядов адреса для адресации 8К ячеек субмодуля. Оставшиеся разряды A_{15-13} поступают на дешифратор, нулевой и единичный выходы которого разрешают работу субмодулей СМ1 и СМ2. Остальные выходы дешифратора могут быть использованы для подключения других объектов адресации в зоне АП, оставшейся свободной (48К).

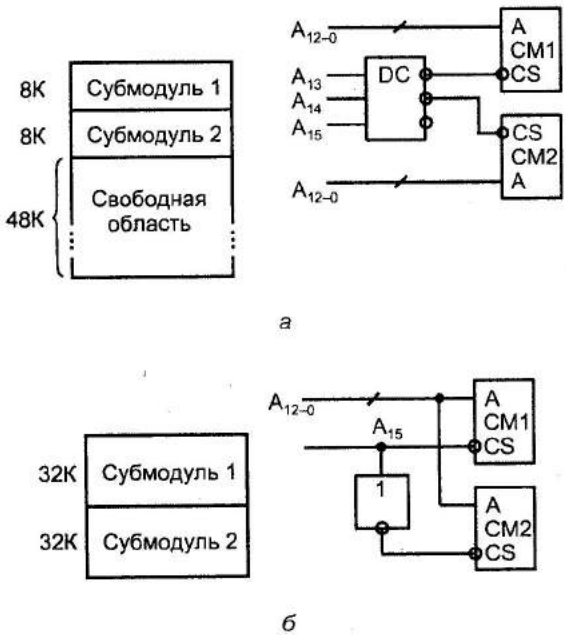


Рис. 5.16. Примеры реализации абсолютной (*а*) и неабсолютной (*б*) адресации субмодулей памяти

При неабсолютной адресации линии адреса A_{12-0} по-прежнему подаются на адресные входы ИС, а для выбора одной из них используется линия A_{15}. Линии A_{14} и A_{13} не используются вообще, их состояния безразличны. Схема адресации (рис. 5.16, *б*) упрощается, вместо дешифратора "3 на 8" нужен только инвертор. Платой за это является занятие двумя субмодулями по 8К всего АП. Действительно, все адреса вида 0XXdd...d принадлежат субмодулю СМ1, а это соответствует верхним 32К АП. Все адреса вида 1XXdd...d принадлежат субмодулю СМ2 и занимают 32К в нижней части АП.

Неабсолютная адресация — достаточно гибкий подход к построению схем декодирования адреса. Для адресации объекта можно использовать более или менее широкую зону АП, выбирая при необходимости компромисс между крайними решениями, показанными на рис. 5.16.

Так, в частности, если для адресации субмодулей СМ1 и СМ2 использовать не 14 разрядов адреса, как в последнем примере, а 15 при назначении адресов 00Xdd...d для первого субмодуля и 01Xdd...d для второго, то первая четверть АП будет занята адресами СМ1, вторая — адресами СМ2, а третья и четвертая свободны. Иначе говоря, "расходование" областей АП окажется вдвое меньшим, чем для схемы (рис. 5.16, *б*), но, в то же время, в схеме декодирования адреса вместо инвертора потребуется дешифратор "2 на 4". В итоге получается вариант, промежуточный между схемами абсолютной адресации и неабсолютной с минимальным числом адресных разрядов.

Пример 5

Для работы процессора с ИС памяти наряду с сигналами адресации нужны и другие управляющие сигналы. На рис. 5.17 дан пример выработки сигналов управления для тактируемого SRAM, требующего импульсных сигналов $\overline{CS}$. Для возвращения $\overline{CS}$ к пассивному (высокому) уровню между циклами обращения к ЗУ, что и придает импульсный характер этому сигналу, используется конъюнкция сигналов $\overline{MEMR}$ и $\overline{MEMW}$ как сигнал разрешения работы дешифратора. В этом случае между циклами обращения к ЗУ и сигнал чтения из памяти $\overline{MEMR}$, и сигнал записи в память $\overline{MEMW}$ пассивны, т. е. имеют единичные уровни. При этом на выходе конъюнктора вырабатывается единичный сигнал, запрещающий работу дешифратора, на выходе которого формируются пассивные (единичные) уровни, т. е. снимаются сигналы $\overline{CS}$. При любом обращении к памяти активизируются (становятся нулевыми) сигнал $\overline{MEMR}$ или $\overline{MEMW}$, что создает нулевой сигнал на выходе конъюнктора и разрешает работу дешифратора.

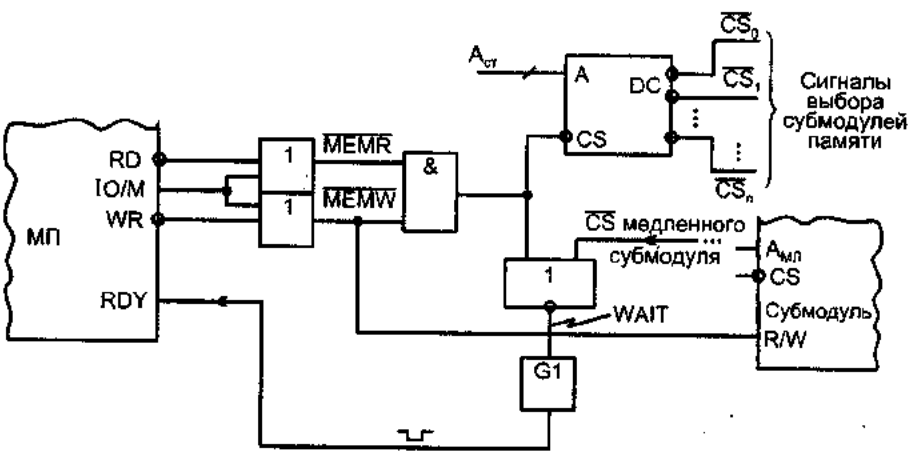


Рис. 5.17. Пример схемы выработки сигналов управления для асинхронного тактируемого статического ЗУ

На рис. 5.17 показан также возможный способ выработки сигнала готовности, подаваемого на вход RDY микропроцессора. Когда микропроцессор обращается к медленному субмодулю, соответствующий сигнал $\overline{CS}$ становится нулевым, и совпадение двух нулей на входах элемента ИЛИ-НЕ дает на его выходе единицу, запускающую генератор одиночных импульсов G1. На время существования этого импульса сигнал готовности снимается, и МП вводит в цикл обращения к памяти такты ожидания. По окончании импульса появляется сигнал RDY, МП выходит из состояния ожидания и реализует операцию обмена. Длительность импульса подбирается соответственно требованиям медленного субмодуля.

Показанный на рис. 5.18 генератор вырабатывает одиночный импульс, синхронизированный с тактовыми импульсами системы. При отсутствии сигнала WAIT в каждом машинном цикле сигнал $\overline{ALE}$ сбрасывает триггеры, на вход элемента И-НЕ действует нулевой сигнал с выхода триггера T1 и, следовательно, сигнал RDY = 1. Появление сигнала WAIT приводит к установке первым же тактовым импульсом триггера T1, на входах элемента И-НЕ оба сигнала становятся единичными и выход RDY принимает нулевое значение. Это состояние продлится до тех пор, пока единичное состояние не продвинется по цепочке триггеров до конца. Установка триггера Tn создает на входе элемента И-НЕ нулевой сигнал, и RDY станет единичным, что позволит МП перейти к операции обмена. Вводимое число тактов ожидания здесь соответствует числу триггеров в цепочке, начиная с T2.

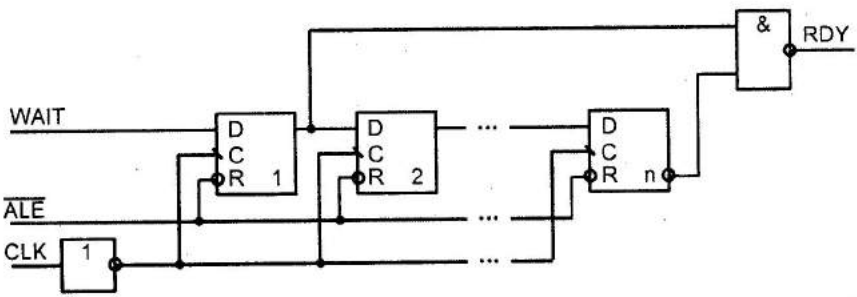


Рис. 5.18. Пример схемы генерации сигнала неготовности при работе процессора с памятью малого быстродействия

Анализ нагрузочных условий

После разработки функционально-логической схемы управления памятью следует провести анализ нагруженности всех ее элементов. Такой анализ требует учета как токовой нагрузки в обоих статических состояниях, так и

емкостной нагрузки. Для обоих состояний выхода элемента суммарный ток нагрузки не должен превышать указанного в технических условиях (ТУ) допустимого выходного тока. Суммарная емкость входов, подключенных к выходу данного элемента в сумме с емкостью монтажа, также не должна превышать допустимой для данного элемента.

Для выходов типа ОК (с открытым коллектором) определяются сопротивления резисторов внешней цепи U_{CC} – R *(см. § 1.2)*.

Перегрузки элементов должны быть устранены введением буферных каскадов или другими способами *(см. § 1.7)*.

Согласование временных диаграмм МП и ЗУ

Имея уточненную по результатам анализа нагрузок схему, можно рассмотреть задачу согласования временных диаграмм МП (точнее, системной шины) и асинхронных ЗУ. Такое согласование — необходимое условие работоспособности системы.

Исходными данными для анализа временных соотношений сигналов являются:

- временные диаграммы машинных циклов МП;
- временные диаграммы циклов работы ЗУ;
- схема адресации и формирования управляющих сигналов ЗУ;
- сведения о задержках сигналов в элементах схемы и цепях связи между ними.

При определении задержек элементов следует иметь в виду их зависимость от емкостных нагрузок на выходе ИС *(см. § 1.1)*.

Перечень режимных параметров ЗУ (необходимых длительностей сигналов, их предустановок, времен удержания и сохранения) достаточно велик. Методика их обеспечения будет показана на примере некоторых параметров. Анализ для других выполняется аналогичным способом.

Рассмотрим *процесс чтения для микросхемы SRAM* тактируемого типа. Выясним вопросы, связанные со временем доступа по адресу t_A, по сигналу выбора t_{CS} и предустановкой адреса относительно сигнала $\overline{CS}$. Для этого воспользуемся схемой, приведенной на рис. 5.19, на которой показаны тракты прохождения интересующих нас сигналов.

Началом отсчета считаем момент выставления адреса на выходах МП. После этого происходят следующие процессы:

1. Часть старшего полуадреса поступает на ЗУ через время t_{BUF} задержки буфера (другая часть старшего полуадреса поступает на дешифратор выработки сигналов $\overline{CS}_A$).

2. Младший полуадрес появляется на входах ЗУ позднее, чем старший, т. к. только через время t_{AL} сигнал ALE задним фронтом загружает регистр, после чего через время задержки регистра t_{RG} сформируется адрес на входах ЗУ.

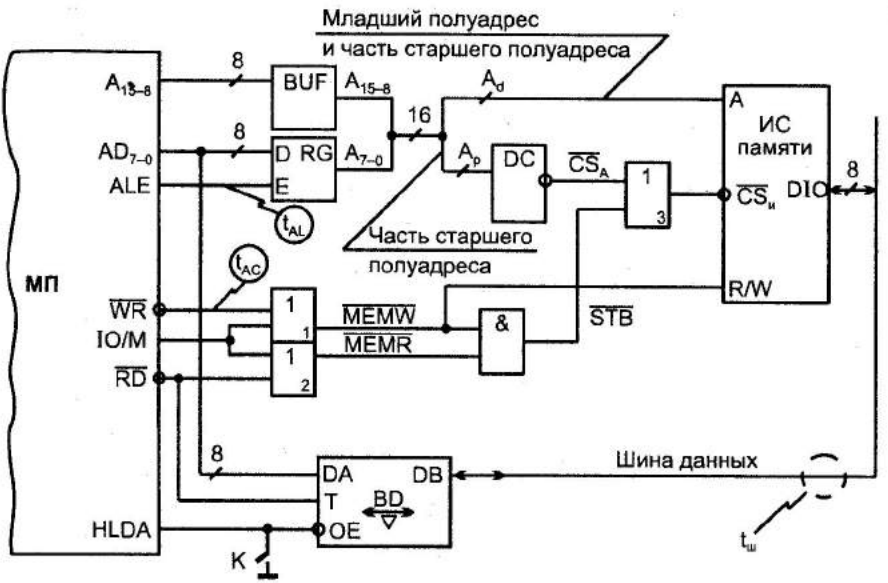


Рис. 5.19. Схема трактов передачи сигналов при управлении памятью

3. Через время t_A после поступления адреса ЗУ вырабатывает выходные данные (t_A — характеристика ЗУ по ТУ).
4. По истечении времени задержек шины и буфера данных ($t_Ш$ и t_{BD}) данные появятся на линиях $AD_{7–0}$ микропроцессора, и это должно произойти не позднее, чем в момент t_{AD}, определяемый временной диаграммой МП.

Среди перечисленных задержек пояснений требует лишь параметр $t_Ш$ — задержка шины. Такая задержка появляется, если ЗУ имеет выходы с открытым коллектором, и при обращении к памяти происходят переключения из нуля в единицу в линиях шины данных (каскады с открытым коллектором медленно формируют положительные фронты). Как правило, подобных ситуаций избегают (например, удерживают линии шины в единичных состояниях при отсутствии на них сигналов, так что при появлении данных происходят только переключения в нуль в соответствующих разрядах). Поэтому в дальнейшем задержку $t_Ш$ учитывать не будем.

Таким образом, на основании сказанного должно соблюдаться соотношение[2]:

$$t_{AL} + t_{RG} + t_A + t_{BD} \le t_{AD}.$$

Интервал t_{AD} согласно ТУ на МП выражается соотношением

$$t_{AD} = (5/2 + N)T - 225 \text{ нс},$$

где T — длительность такта и N — число тактов ожидания в цикле чтения. Если неравенство удовлетворяется при N = 0, то возможна работа без тактов ожидания. Иначе требуется ввести столько тактов ожидания, сколько потребуется для удовлетворения неравенства.

Из полученного неравенства следует условие, предъявляемое к параметру t_A:

$$t_A \le t_{AD}(N) - (t_{AL} + t_{RG} + t_{BD}).$$

Рассмотрим теперь требования к параметру памяти t_{CS}. Из отмеченного выше следует, что сигнал $\overline{CS}_A$ (сигнал выбора субмодуля, получаемый декодированием нескольких старших разрядов адреса) появляется на входе элемента ИЛИ с номером три через время $t_{BUF} + t_{DC}$. Нулевой сигнал на нижнем входе этого элемента ИЛИ появится позднее и определит тем самым момент поступления сигнала $\overline{CS}_И$ на вход ИС памяти. Этот сигнал, обозначенный как $\overline{STB}$, появится в момент времени $t_{AC} + t_{ИЛИ} + t_И$, где t_{AC} — параметр временной диаграммы МП (интервал между моментами выставления адреса и строба чтения). По истечении времени задержки элемента ИЛИ на входе $\overline{CS}_И$ сформируется сигнал выбора субмодуля. После этого памяти потребуется время t_{CS} для подготовки выходных данных, которые после задержки в буфере данных появятся на линиях AD_{7-0} микропроцессора, что должно произойти не позднее, чем в момент времени t_{AD}.

Из сказанного следует условие:

$$t_{AC} + 2t_{ИЛИ} + t_И + t_{CS} + t_{BD} \le t_{AD},$$

на основании которого предъявляется требование к величине t_{CS}:

$$t_{CS} \le t_{AD} - (t_{AC} + 2t_{ИЛИ} + t_И + t_{BD}).$$

Разность времен появления сигналов $\overline{CS}_И$ и адреса на входах ИС памяти определит их предустановку в схеме:

$$t_{SU(A-CS).CX} = t_{AC} + 2t_{ИЛИ} + t_И - (t_{AL} + t_{RG}).$$

Требуется соблюдение условия:

$$t_{SU(A-CS).CX} \ge t_{SU(A-CS).ТУ},$$

где $t_{SU(A-CS).ТУ}$ — параметр памяти.

---

[2] В этом соотношении и далее задержки элементов указываются без учета их разброса. При оценке ситуаций по методу наихудшего случая учитываются предельные значения задержек, максимальные или минимальные.

К процессам завершения цикла чтения тоже предъявляются определенные требования. Необходимо, чтобы "старые" данные были сняты с шины данных AD_{7-0} раньше, чем появится новое значение адреса (в следующем цикле). Временные диаграммы МП определяют интервал от конца строба чтения до появления нового адреса t_{RA} как величину T/2 − 10 нс. В паспортных данных ИС памяти имеется параметр $t_{DIS(CS)}$ — время запрещения данных после снятия сигнала CS. Временные соотношения сигналов для процесса завершения чтения (рис. 5.20) учитывают, что сигнал $\overline{CS}_И$ будет снят после окончания сигнала $\overline{RD}$ через время, равное суммарной задержке элементов ИЛИ с номером два, И и ИЛИ с номером три. На основании рисунка можно записать соотношение:

$$t_{DIS(CS)} \le t_{RA} - (2t_{ИЛИ} + t_{И}).$$

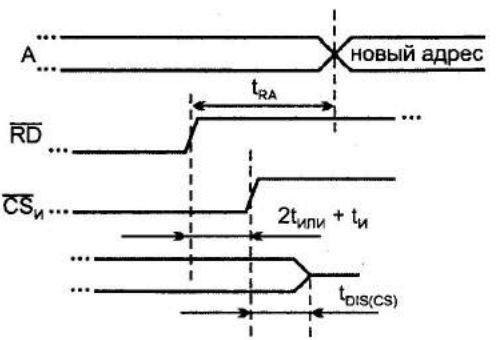


Рис. 5.20. Временные диаграммы сигналов для завершения цикла чтения из памяти

В цикле записи следует обеспечить выбор ячейки только после ее четкой адресации и предотвратить обращение к иным, кроме выбранной, ячейкам. Первое условие требует определенной предустановки адреса относительно строба записи на входах ИС памяти, второе — полного отключения от трактов записи предыдущей ячейки до начала нового цикла.

Согласование временных диаграмм памяти и МП для быстродействующих систем оказывается сложной задачей. В таких системах сами временные интервалы диаграмм малы, и их сдвиги из-за паразитных задержек сильно усложняют построение работоспособных схем. В последнее время *решение этой проблемы находят в разработках синхронных ЗУ.* Синхронные динамические ЗУ с конвейерной организацией тракта передачи данных рассмотрены в *§ 4.11.* Такие же по архитектуре ЗУ появляются и в микросхемах статической памяти.

Схемы реализации безусловного программного ввода/вывода

Для схем подключения внешних устройств к шинам МПС ранее был рассмотрен пример с линейной селекцией ВУ, удобной при малом их числе в системе. Возможностями подключения большого числа ВУ (до 256 ВУ ввода и 256 ВУ вывода при восьмиразрядных адресах) обладает вариант адресуемых портов.

Программный ввод/вывод, осуществляемый по инициативе программы, может быть безусловным или условным. Первый способ возможен при обмене с всегда готовым ВУ, второй требует учета готовности ВУ к операциям ввода/вывода. При условном обмене могут возникать потери времени на ожидание готовности ВУ. Алгоритм обмена с ожиданием готовности (рис. 5.21, *а*) таков, что МП может зависать в цикле ожидания готовности ВУ, причем при работе с ВУ малого быстродействия время ожидания может оказаться большим. Возможен также другой алгоритм условного ввода/вывода, когда при неготовности ВУ МП отказывается от операции обмена и продолжает основную программу. В последующем может быть выполнена новая попытка обмена (рис. 5.21, *б*).

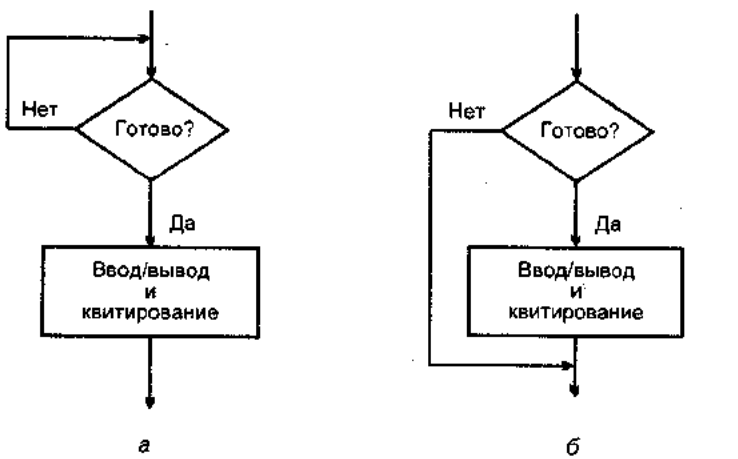


Рис. 5.21. Алгоритмы условного программного обмена с занятием цикла (*а*) и совмещенного (*б*)

Потери времени на ожидание готовности исключаются при обмене по прерываниям, когда инициатором обмена является само ВУ при его готовности к обмену. *Общие вопросы обмена по прерываниям уже отмечались в § 5.3*, дополнительные сведения будут даны в дальнейшем при рассмотрении контроллеров прерывания.

Несмотря на отмеченные недостатки, программный обмен широко применяется вследствие его эффективности с точки зрения аппаратных затрат.

При вводе данных микропроцессор вырабатывает адрес ВУ A_{7-0}, сигнал IO/M = 1 и строб чтения $\overline{RD}$. В МПС с четверкой сигналов управления чтением/записью и вводом/выводом при вводе формируются адрес ВУ и строб чтения $\overline{IOR}$.

При использовании первого из указанных наборов управляющих сигналов схема адресного порта безусловного ввода имеет вид рис. 5.22, *а*, где адрес ВУ принят равным 20H = 00100000. В этом случае сигнал выбора ВУ должен появиться при подаче с адресных шин на входы конъюнктора набора $\overline{A}_7\overline{A}_6A_5\overline{A}_4\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$ или, что то же самое, при подаче на входы элемента ИЛИ-НЕ набора $A_7A_6\overline{A}_5A_4A_3A_2A_1A_0$. Последний вариант лучше, т. к. требует инвертировать всего один бит адресного кода, тогда как при реализации первого потребуются семь инверторов.

Совпадение условий ввода дает нулевой сигнал на выходе элемента И-НЕ, открывающий линейку буферов с тремя состояниями выхода и передающий таким образом байт данных от ВУ на шину данных, откуда они и считываются микропроцессором.

На рис. 5.22, *б* показана реализация команды OUT 2BH. В этом случае безусловный вывод осуществляется при поступлении от микропроцессора адреса A_{7-0} = 00101011, сигналов IO/M = 1 и $\overline{WR}$ = 0. Поступая на конъюнктор с инверторами в соответствующих входных линиях, адрес ВУ формирует единичный сигнал выбора ВУ SEL, что в совокупности с единичными сигналами на двух других входах элемента И-НЕ дает отрицательный перепад напряжения на его выходе.

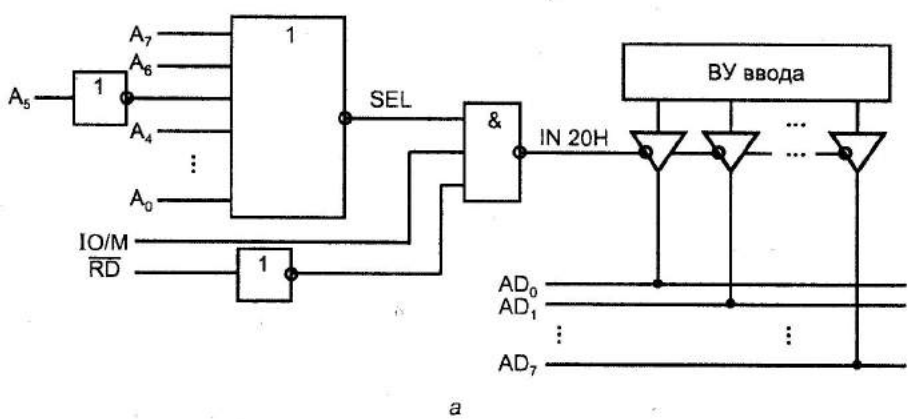


Рис. 5.22. Схема реализации команд ввода (*а*) при безусловном программном обмене

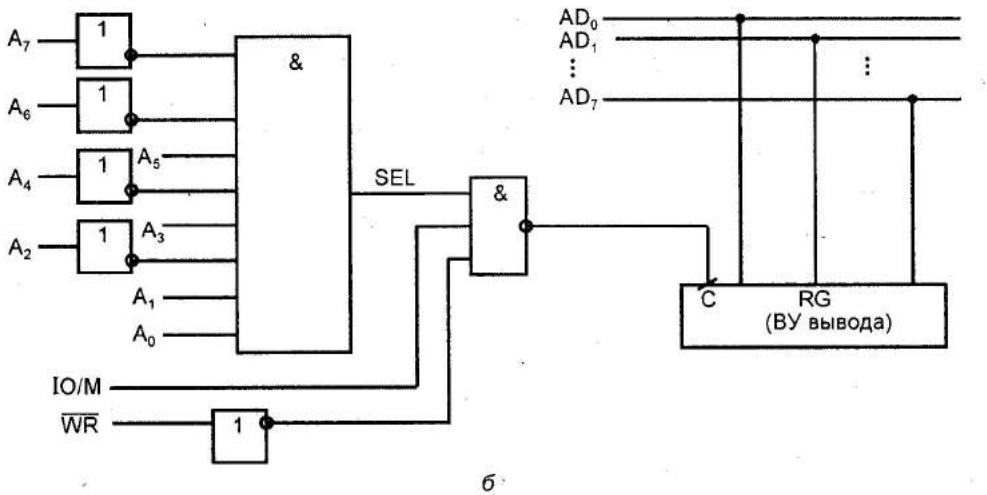


Рис. 5.22. Схема реализации команд вывода (*б*) при безусловном программном обмене

Задний фронт строба $\overline{WR}$ дает положительный перепад выходного напряжения элемента И-НЕ, загружающий данные с шины данных AD_{7-0} в регистр порта вывода.

Схемы реализации условного программного ввода/вывода

Внешние устройства чаще всего не имеют постоянной готовности к обмену и скоростному вводу/выводу в темпе процессора. Поэтому необходимо удостовериться в готовности ВУ, прежде чем начать обмен, т. е. операции ввода/вывода сопровождаются специальными сигналами готовности, генерируемыми ВУ и вводимыми в МП.

После операции ввода/вывода сигнал готовности должен быть снят и выставлен снова при новой готовности к обмену. Такой протокол называют *обменом с квитированием*. Обмен происходит со скоростью, определяемой внешним устройством.

Вариант построения порта ввода с квитированием и "четверкой" управляющих сигналов в интерфейсе (рис. 5.23, *а*) предусматривает наличие регистра-защелки RG. Под блоком DC понимается схема декодирования адреса (не обязательно состоящая из одного дешифратора). ВУ готовит данные на линиях D_{7-0} и оповещает об их наличии сигналом STB, загружающим регистр, и своим задним фронтом, устанавливающим триггер, формируя этим сигнал IBF = 1 (IBF, Input Buffer Full). Этим фиксируется готовность к вводу со стороны порта.

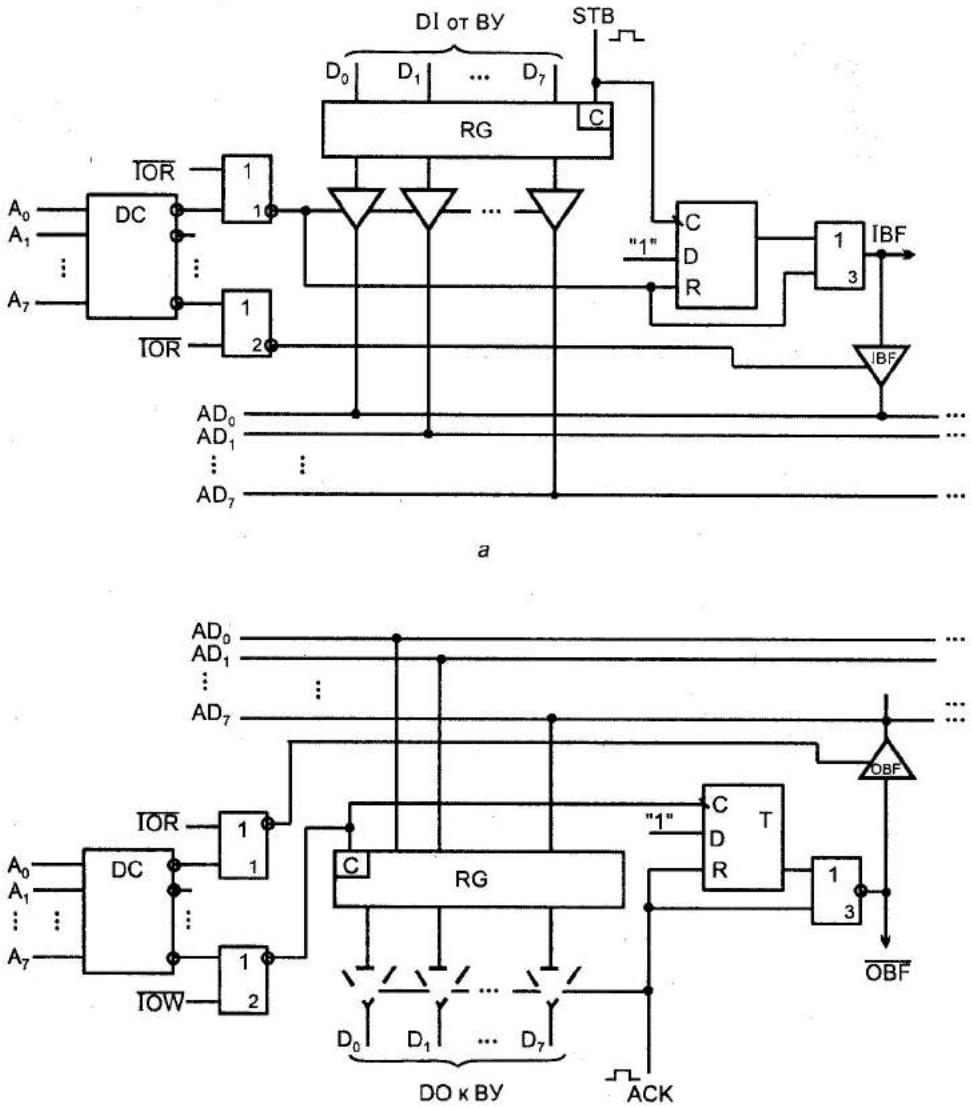


Рис. 5.23. Схемы реализации команд ввода (*а*) и вывода (*б*) при условном программном обмене

Микропроцессор начинает обращение к порту чтением сигнала IBF по адресу, присвоенному буферному каскаду IBF (для определенности принято, что этот адрес соответствует возбуждению нижней выходной линии DC). По

стробу $\overline{IOR}$ на выходе элемента $ИЛИ_2$ возникает единичный сигнал на линии, открывающий буферный каскад, через который сигнал IBF поступает на линию AD_0. Этот сигнал есть бит слова состояния, которое считывается процессором. Если значение этого бита 1, то далее осуществляется ввод по адресу порта (здесь этот адрес принят нулевым, и ему соответствует возбуждение верхней выходной линии схемы DC). Строб $\overline{IOR}$ устанавливает логическую 1 на выходе элемента $ИЛИ_1$, т. е. открывает линейку буферных каскадов, через которые байт D_{7-0} поступает на линии AD_{7-0} и далее вводится в процессор. По окончании строба $\overline{IOR}$ сигнал IBF становится нулевым. Это снимает готовность к обмену до новой загрузки входного буферного регистра RG от ВУ с установкой после этого триггера и приведения IBF в состояние логической 1, т. е. в состояние новой готовности к обмену.

В программе описанный процесс ввода отображается следующей процедурой:

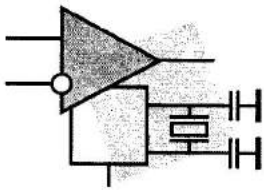
```
IA:     IN OFFH         ; Ввод слова состояния
        ANI 01H         ; Выделение бита
        JZ IA           ; Если IBF = 0, то ждать
        IN 00H          ; Иначе ввод данных
```

Условный вывод данных иллюстрируется схемой (рис. 5.23, *б*). Сначала вводится слово состояния с линий AD_{7-0}, на одной из которых действует сигнал готовности $\overline{OBF}$ (Output Buffer Full). Данные будут выводиться процессором в RG, готовность к выводу выражается в том, что данные регистра уже приняты ВУ. Об этом сигнализирует подтверждение ACK, сбрасывающее триггер и дающее после своего окончания $\overline{OBF} = 1$ (буфер пуст). Кстати, подача на входы вентиля 3 самого сигнала сброса и выхода сбрасываемого триггера не дает признаку $\overline{OBF}$ измениться до окончания импульса ACK. То же применено в схеме (см. рис. 5.23, *а*) для сигнала IBF. Появление готовности именно после завершения указанных действий требуется для надежной работы схем, приведенных на рис. 5.23.

Из введенного слова состояния выделяется бит $\overline{OBF}$ (как и ранее командой ANI, т. е. выполнением поразрядной конъюнкции со словом, содержащим единицу только в разряде, принадлежащем $\overline{OBF}$). При сигнале $\overline{OBF} = 1$ идет обмен, иначе — переход к циклам ожидания. Появление готовности вызывает запись в RG данных и установку триггера, дающего $\overline{OBF} = 0$, как сигнал для ВУ к использованию выводимых данных, после чего ВУ выдает сигнал ACK, сбрасывающий триггер и выставляющий сигнал готовности порта к новому выводу.

Литература к главе: [3], [13], [15], [18], [21], [27], [28], [37], [50], [53], [IV], [XIV], [XXIII], [XXX], [XXXV].

Глава 6



Интерфейсные и периферийные микросхемы

§ 6.1. Интерфейсы микропроцессорных систем

С задачами обмена информацией между цифровыми устройствами связано понятие стандартного интерфейса. *Интерфейс* — совокупность аппаратных и программных средств, обеспечивающих совместимость устройств, обменивающихся информацией. Унификация средств общения между устройствами необходима. Попытку построить систему как некоторое целое, состоящее из нескольких частей, при отсутствии стандартного интерфейса можно уподобить попытке (бесполезной) организовать беседу людей, каждый из которых говорит на своем языке.

Аспектами стандартизации интерфейса являются функциональная, электрическая и механическая совместимости.

Функциональная совместимость устройств требует смысловой общности (единства) управляющих сигналов, генерируемых обменивающимися модулями. Управляющие сигналы должны иметь заданное смысловое значение и определенные временные параметры.

Электрическая совместимость модулей обеспечивается заданными уровнями вырабатываемых ими сигналов, их нагрузочными способностями, мощностями и т. п.

Механическая совместимость предполагает применение определенных типов и размеров конструкций, соединителей и т. д.

Соответственно сказанному, к основным элементам интерфейса относят *протокол обмена* (совокупность правил, регламентирующих способ выполнения заданных функций), аппаратную часть (физическую реализацию устройств) и программное обеспечение.

Интерфейсы имеют *развитую классификацию* по признакам конфигурации цепей связи между объектами (магистральные, радиальные и др.), характеру передаваемых данных (параллельные, последовательные и др.), режиму передачи данных (дуплексный, полудуплексный, симплексный), способу обмена (асинхронные, синхронные).

На характер интерфейса существенно влияет область его применения, согласно областям применения выделяют несколько классов интерфейсов. Интерфейс межмодульного обмена в микропроцессорных системах, с которым связаны рассматриваемые в этой главе схемы, называют *системным (внутренним)*.

К интерфейсным схемам ниже отнесены шинные формирователи, буферные регистры, параллельные и последовательные порты и адаптеры и специальные интерфейсные средства JTAG, а к периферийным — контроллеры прерываний, контроллеры прямого доступа к памяти, интервальные таймеры.

Стандарт на интерфейс устанавливает набор применяемых сигналов, определяет их логические функции и электрические параметры, конструкции разъемов и т. д. Устройства, рассчитанные на совместную работу, должны подчиняться всем требованиям интерфейса. Таким образом, принятие стандартного интерфейса сопровождается разработкой целого спектра *устройств, обеспечивающих его использование*.

Одновременно с появлением популярных микропроцессоров были разработаны и первые интерфейсы микропроцессорных систем. Фирма Intel для простых систем, в которые входят не более десяти модулей, предложила интерфейс Microbus. При этом для сопряжения модулей были разработаны интерфейсные схемы семейства Intel 82XX. В обозначении 82XX цифры 82 отображают принадлежность ИС к этому семейству, а цифры на позициях, отмеченных крестиками, определяют конкретный тип схемы. Интерфейсные БИС семейства 82XX получили широкое распространение и оказались весьма долгоживущими. В настоящее время их реализация имеет несколько разновидностей:

- в виде отдельных микросхем;
- в виде частей сложных БИС/СБИС, в которых на одном кристалле размещены несколько интерфейсных схем разного функционального назначения;
- в виде так называемых *мегафункций* (макрофункций), т. е. данных для настройки определенной области кристалла программируемой логики на выполнение заданных функций. Такие мегафункции получили название *soft-ядер* или *IP* (Intellectual Properties, что по-русски можно назвать *"единицами интеллектуальной собственности"*);
- в виде специализированных областей с фиксированными функциями на кристаллах программируемой логики *(hard-ядер, аппаратных ядер)*, которые не подлежат изменениям при репрограммировании кристалла.

Интерфейсы начального периода развития МПС

Интерфейс *Microbus* был разработан в конце 1970-х гг. для построения систем на основе 8-разрядных микропроцессоров Intel 8080, Motorola 6800 и др. Интерфейс является системным, однопроцессорным, магистральным, параллельным, асинхронным с полудуплексной (двусторонней поочередной) передачей данных. Интерфейс получил широкое распространение при объединении в систему не более 10 подключаемых к магистрали устройств, расположенных в непосредственной близости друг от друга. Для него был разработан ряд интерфейсных ИС (в том числе отечественных).

В функциональном аспекте интерфейс задается набором сигналов и их временными параметрами (длительностями и расположением во времени). В интерфейсе Microbus 36 сигнальных линий, в числе которых 16-разрядная шина адреса, 8-разрядная шина данных и линии $\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, RDY, INT, $\overline{\text{INTA}}$, HOLD, $\overline{\text{HLDA}}$, CLK, RESET, $\overline{\text{BUSEN}}$ для управляющих сигналов, которые рассматривались ранее при описании микропроцессора Intel 8085A. В интерфейсе адресные пространства памяти и ВУ разделены, выполняются протоколы программного обмена, в том числе обмена по прерываниям, и прямого доступа к памяти.

Позднее фирмой Intel были разработаны несколько вариантов интерфейса *Multibus* (его отечественный аналог — И-41), в котором допускается использование 8- и 16-разрядных модулей, один из которых (активный) играет роль *задатчика*, другой (пассивный) — *исполнителя*. При запросах управления магистралью одновременно от нескольких задатчиков решается задача арбитража. В состав линий входят 25-разрядная шина адреса, 16-разрядная шина данных, 9-разрядная шина прерываний и ряд линий шины управления. На интерфейсе заданы протоколы: программного обмена (с возможным запретом обращения); арбитража запросов задатчиков на управление магистралью и смены задатчика; обработки прерываний и аварии в системе электропитания.

Отечественный *интерфейс МПИ* (на основе зарубежной шины Q-bus) — асинхронный при передаче данных и синхронный при передаче адреса. Адрес и данные передаются по одной и той же шине с разделением во времени (мультиплексируемой шине адресов-данных). Основное назначение интерфейса — построение однопроцессорных систем, точнее, систем с одним ведущим процессором. Выполняются адресный обмен (в том числе и блочный), захват магистрали и прерывания. Адресное пространство памяти и ВУ — общее и может составлять 64K (16-разрядный адрес) или 16M (24-разрядный адрес).

Интерфейсы последующих поколений МПС, персональных компьютеров и других современных систем

С ростом разрядности и быстродействия процессоров изменялись и соответствующие характеристики интерфейсов, применяемых как в персональных компьютерах (ПК), так и в других МПС.

Появление ПК IBM PC/AT ассоциируется с применением интерфейса (шины) *ISA* (Industrial Standard Architecture). Для систем с процессорами 80386 были разработаны шины *EISA* (Extended ISA) и *MCA* (Micro Channel Architecture). Сейчас применительно к требованиям современных компьютеров (но не ко всем областям применения) обе эти шины устарели. Важным стандартом на магистральные шины современных систем является шина *VME* (Versabus Module Europe), используемая в устройствах разного класса, в том числе в системах управления с большим числом передач не только цифровых, но и аналоговых данных. Шина ориентирована на применение со встроенными средствами МПС реального времени, сбор информации от датчиков и ее последующую обработку.

На уровне локальных шин компьютеров широко применяется шина *PCI* (Peripheral Component Interconnect) фирмы Intel, известна шина VL-Bus (VESA Local Bus) и др. Популярность шины PCI объясняется тем, что это интерфейс с новыми качествами, не зависящий от платформы (т. е. способный работать с разными процессорами), имеющий высокую производительность и недорогой в производстве. Концептуальными достоинствами шины являются:

- способность выполнять некоторые действия без обращения к процессору, тем самым уменьшая его загрузку;
- применение для связи с другими компонентами системы так называемых *мостов*, что оказалось удачным решением, свойственным современным *чипсетам* (т. е. конструктивно объединенным наборам микросхем, которые наряду с процессором и памятью составляют аппаратуру компьютера);
- реализация принципов ведущей и ведомой шин и др.

Шиной выполняется синхронный обмен данными, пакетные их передачи, обмен с квитированием. Шина приспособлена к распознаванию аппаратных средств системы и анализу ее конфигурации согласно стандарту Plug & Play. Малопроизводительный вариант шины имеет частоту тактирования 33 МГц и разрядность 32, более производительный — 66 МГц и 64 разряда. В последних вариантах частота работы повышена до 100 МГц.

Отметим, кстати, что тактовая частота современных системных шин сейчас обычно составляет 66—133 МГц, появились и шины с тактовой частотой 166 МГц.

Обмен *последовательными данными* осуществляется в микропроцессорных системах с помощью интерфейсов *SPI* (Serial Peripheral Interface), *I²C* (Inter Integrated Circuits), в последнее время резко возрастает популярность шины *USB* (Universal Serial Bus), отличающейся удобством подключения к ней дополнительных периферийных устройств. Для более дальних связей (связей системы с другими системами и т. п.) традиционны последовательные интерфейсы стандартов RS-232C, RS-485, а для более сложных задач шина *CAN* (Control Aria Network).

§ 6.2. Шинные формирователи и буферные регистры

Шинные формирователи

Шинные формирователи (ШФ), называемые также *приемопередатчиками*, *шинными драйверами* или *магистральными вентиль-буферами*, включаются между источником информации и системной шиной. Они усиливают сигналы по мощности, отключают источник информации от шины, когда он не участвует в обмене, формируют требуемые уровни сигналов логической 1 или 0. Двунаправленные ШФ позволяют в зависимости от сигнала управления передавать сигналы в шину или, напротив, принимать их с шины и передавать приемнику данных. В английской терминологии ШФ обозначается как BD (Bus Driver).

Различные шинные формирователи отличаются не только разрядностью, но и передачей сигналов в прямом (для ШФ) или инвертированном (для ШФИ) виде, а также прямыми или инверсными сигналами разрешения работы. Отличаются они и электрическими характеристиками. Схема ШФ Intel 8286, воспроизведенная во многих других сериях элементов, показана на рис. 6.1, *а*.

Шина A (линии A_{0_7}) принимает данные от МП или передает их ему, шина B (линии B_{0_7}) связана с магистралью, на которую передает информацию или с которой принимает ее. Сигнал $\overline{OE}$ переводит выходы усилителей в третье состояние (при его высоком уровне), либо разрешает их работу (при низком уровне). При разрешении работы направление передачи зависит от сигнала T (Transmit). Функционирование ШФ подчиняется условиям, указанным в табл. 6.1.

Таблица 6.1

$\overline{OE}$	T	Режим
1	0	Нет передачи
1	1	Нет передачи

Таблица 6.1 (окончание)

$\overline{OE}$	T	Режим
0	1	Передача от A к B
0	0	Передача от B к A

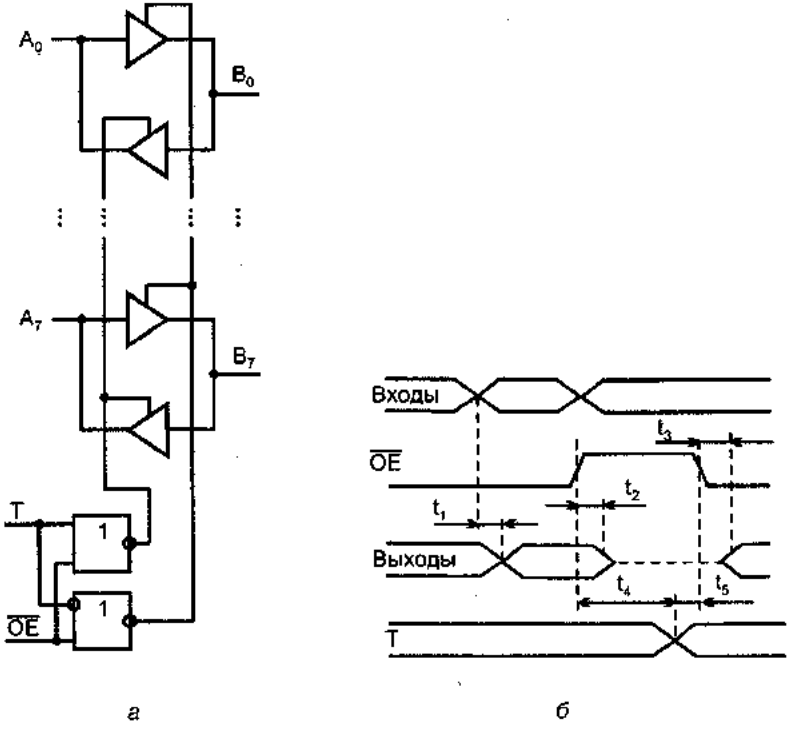


Рис. 6.1. Схема шинного формирователя (*а*) и временные диаграммы его работы (*б*)

Так как шина A связана с МП, а шина B — с магистралью, для них предусмотрена разная нагрузочная способность: выходы B обеспечивают токи большей величины, чем выходы A.

На временных диаграммах (см. рис. 6.1, *б*) показаны задержки сигналов при их распространении через открытые ШФ (задержка t_1) и задержки относительно изменений управляющих сигналов (задержка t_2 перехода выходов в состояние "отключено", задержка t_3 переходов от состояния "отключено" к активным состояниям). Интервалы t_4 и t_5 — времена выдержки и предустановки сигнала относительно моментов изменения сигнала OE. Временные

параметры ШФ даются для определенных нагрузочных токов (обычно максимальных) и емкостей.

Восьмиразрядный ШФ серии КР1533 (технология ТТЛШ) характеризуется следующими параметрами:

- выходной ток 30—112 мА;
- задержка распространения сигнала ≤ 10 нс;
- время выхода из ТС в активное состояние ≤ 20 нс;
- время перехода из активного состояния в ТС 25—40 нс.

Для ШФ серии КР1554 (технология КМОП) параметры таковы:

- выходные токи 86 мА и 75 мА для низкого и высокого уровней выходного напряжения соответственно при условии протекания не дольше 20 мс и 24 мА без ограничения времени;
- при напряжении питания 4,5 В задержка распространения сигнала ≤ 6 нс, задержка выхода из ТС в активное состояние ≤ 6,5 нс, задержка перехода из активного состояния в ТС ≤ 8,5 нс.

Буферные регистры

Буферные регистры также работают на магистраль, но, в отличие от ШФ, способны хранить данные. Благодаря этому они могут выполнять *временную буферизацию* данных, что составляет важнейшую функцию портов. Буферные каскады с тремя состояниями на выходах регистра обеспечивают портам возможность отключения от магистрали под действием управляющих сигналов, а также необходимую нагрузочную способность.

Через порты ввода данные от ВУ поступают в магистраль, а через порты вывода данные с магистрали передаются тому или иному модулю. Порты ввода/вывода могут выполнять обе указанные операции.

В сериях стандартных элементов имеются восьмиразрядные буферные регистры ИР82 и ИР83 (инвертирующий) — аналоги зарубежных ИС Intel 8282 и 8283. Буферный регистр ИР82 (рис. 6.2, *а*) принимает данные по шине А (линии A_{0-7}) в регистр.

Сигнал $\overline{OE}$ низким уровнем разрешает работу вентиль-буферов и тем самым передает содержимое регистра на выходную шину, высоким уровнем переводит выходы вентиль-буферов в состояние "отключено". Прием данных в регистр разрешается сигналом строба STB.

Временные диаграммы работы буферного регистра (рис. 6.2, *б*) показывают задержку t_1 сигналов от входа к выходу при STB = 1, задержку t_2 от моментов изменения $\overline{OE}$ до перехода к режиму "отключено" и задержку t_3 до выхода из этого режима. Задержка t_4 — интервал от момента изменения строба

до изменения выхода схемы. Интервал t_5 — время предустановки сигнала на входе относительно спада строба (часто не лимитировано), t_6 — время выдержки входного сигнала относительно спада строба ≥ 25 нс.

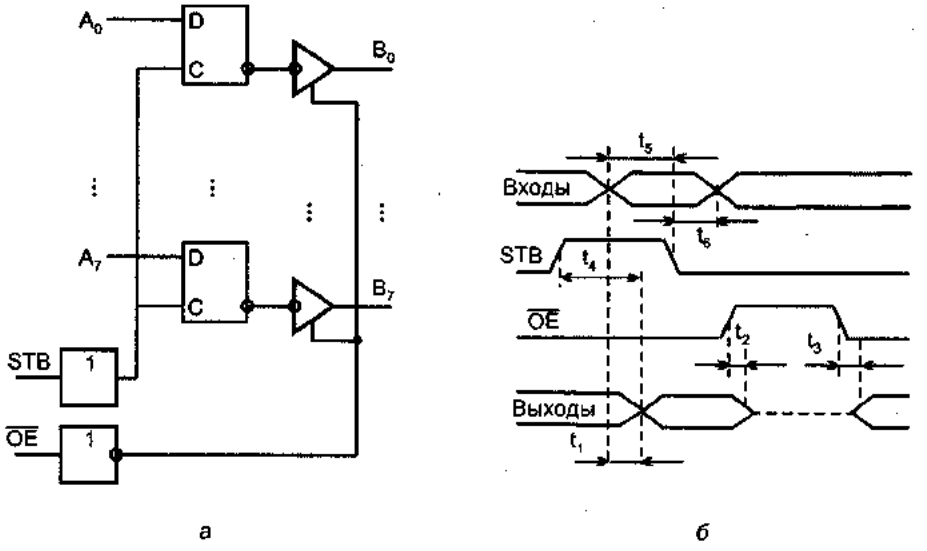


Рис. 6.2. Схема буферного регистра (а) и временные диаграммы его работы (б)

Буферный регистр ИР83 отличается от регистра ИР82 тем, что инвертирует передаваемые данные. Буферные регистры широко представлены в сериях ИС, в частности, тех, которые указаны ранее для ШФ.

В серии КР1533 буферные регистры обеспечивают выходные токи 15—70 мА при максимальных задержках от тактирующего входа около 15 нс, временах выхода из ТС около 20 нс и входа в ТС около 20—30 нс. В серии КР1554 выходные токи буферных регистров те же, что и для ШФ, задержки при U_{CC} = 4,5 В имеют порядок 10 нс.

§ 6.3. Параллельные порты и адаптеры

Схемотехника параллельных портов

Параллельные порты — основные средства обмена информацией между модулями МПС. Они могут быть организованы с помощью параллельных адаптеров или же представлять собою более простые схемы. Схемотехника параллельных портов различных устройств имеет как общие черты, так и свои особенности. В старой схемотехнике линии портов рассматривались

обычно как *группы с единым управлением*, образующие порт ввода либо порт вывода. Позднее схемотехника портов стала более гибкой, и во многих портах *каждая линия стала индивидуально конфигурироваться* как вход или выход, так что в составе одного и того же порта могут быть одновременно и линии ввода, и линии вывода с индивидуальным управлением. Кроме того, на линии портов могут быть возложены и дополнительные задачи индивидуального характера.

В схеме двунаправленной линии с индивидуальным конфигурированием на ввод или вывод (рис. 6.3) направленность линии задается битом, загружаемым в регистр (триггер) направления передачи, тактируемый синхросигналом "Запись направления передачи".

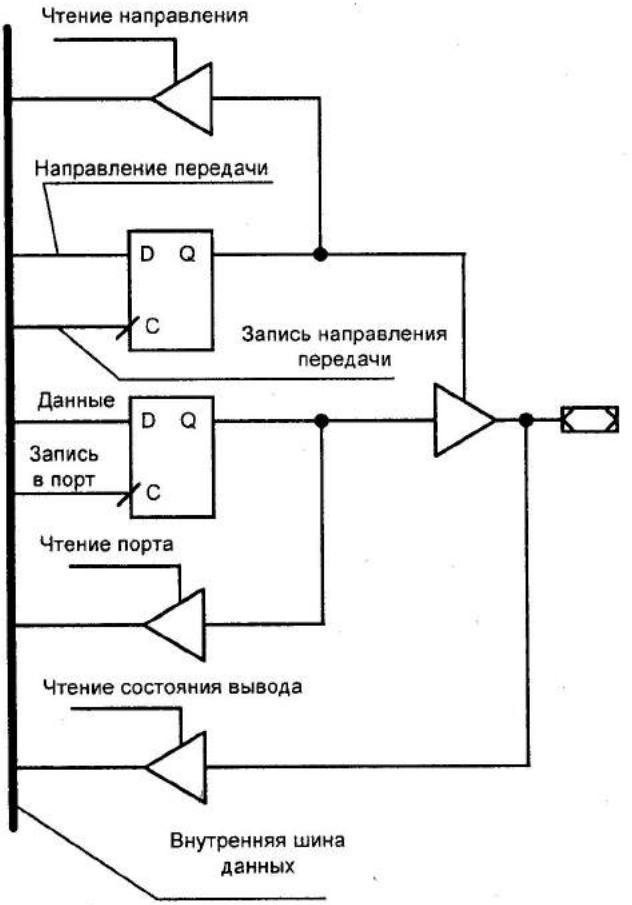


Рис. 6.3. Упрощенная схема программируемой линии порта ввода/вывода

При единичном значении этого бита работа выходного драйвера разрешена и контакт является выходом. Состояние регистра может быть прочитано по сигналу "Чтение направления", разрешающему работу драйвера направления передачи. Выводимый бит своим синхросигналом записывается в регистр данных, состояние которого может быть прочитано контроллером по сигналу "Чтение порта". Кроме того, можно читать непосредственно состояние контакта по сигналу "Чтение состояния вывода", разрешающему работу связанного с контактом входного драйвера.

Если бит направления передачи имеет нулевое значение, выходной драйвер находится в третьем состоянии, а вывод становится входом, передающим сигналы с контакта на шину данных под управлением сигнала "Чтение состояния вывода" входного драйвера.

При работе с портом используются три адреса: при обращении по одному из адресов возможны ввод или вывод сведений о *направлении* передач, при обращении по второму записываются или определяются *данные* порта, по третьему производится обращение для чтения фактического *состояния внешнего контакта*.

Наряду с работой в режиме общего назначения линии портов могут быть использованы и для выполнения специальных функций. На рис. 6.4 приведена схема линии порта, которая отличается от рассмотренной (см. рис. 6.3) наличием элементов "подтягивания" потенциала выходного контакта и возможностью использования линии в качестве выхода передатчика блока UART (Universal Asynchronous Receiver / Transmitter). Кроме того, для повышения помехоустойчивости ввода перед входным драйвером помещен триггер Шмитта, имеющий передаточную характеристику с петлей гистерезиса.

Для подтягивания потенциала выходного контакта к высокому уровню в схему введен ключевой транзистор, при отпирании которого напряжение питания через сопротивление проводящего транзистора подключается к внешнему контакту (при необходимости последовательно с транзистором включается также и резистор). В отличие от схем, в которых подтягивающий резистор подключен к контакту постоянно, здесь применено более гибкое решение — транзистор управляется от логического элемента и включается или запирается в зависимости от режима линии. Чтобы включить транзистор, на его затвор нужно подать низкое напряжение логического нуля. На выходе управляющего транзистором элемента логический нуль вырабатывается как функция $\overline{OE}$ & D, где D — состояние бита данных. Следовательно, цепь подтягивания потенциала выходного контакта вступает в действие только для режима ввода, когда это и имеет смысл, и при установке регистра (триггера) порта в единичное состояние.

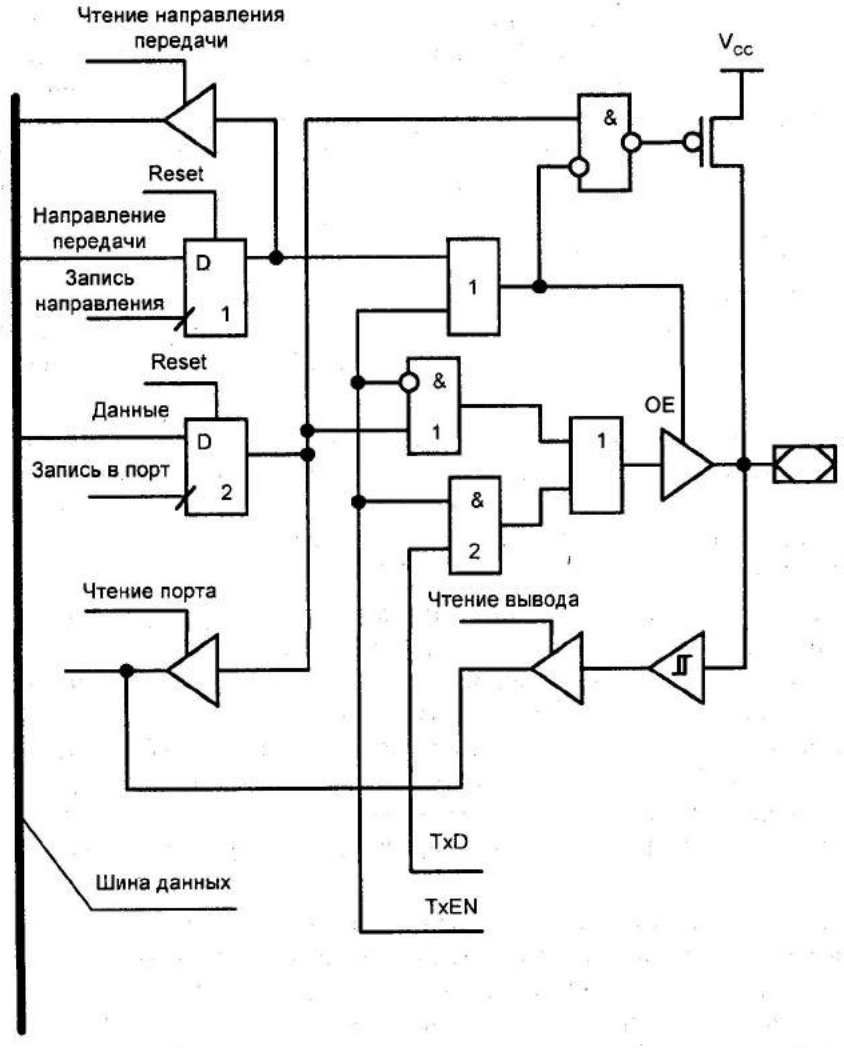


Рис. 6.4. Полная схема линии порта ввода/вывода

Примененный вариант подтягивающей цепочки, в частности, может быть использован при вводе в систему сведений о состоянии кнопки, подключенной к внешнему контакту (рис. 6.5). Поставив линию в режим ввода и записав 1 в регистр порта, можно воспринять состояние кнопки. Если она замкнута, введется нуль, а если разомкнута, то 1.

Другой вариант организации линий двунаправленного вывода иллюстрируется на рис. 6.6.

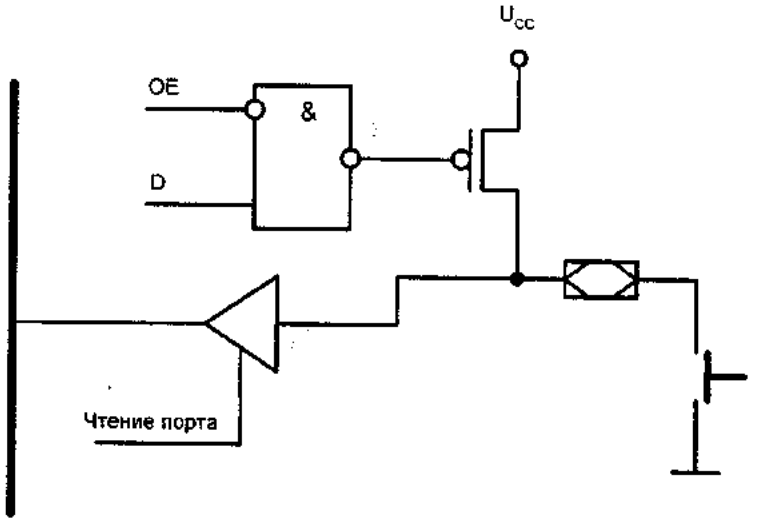


Рис. 6.5. Схема чтения состояния кнопки, подключенной к выводу порта

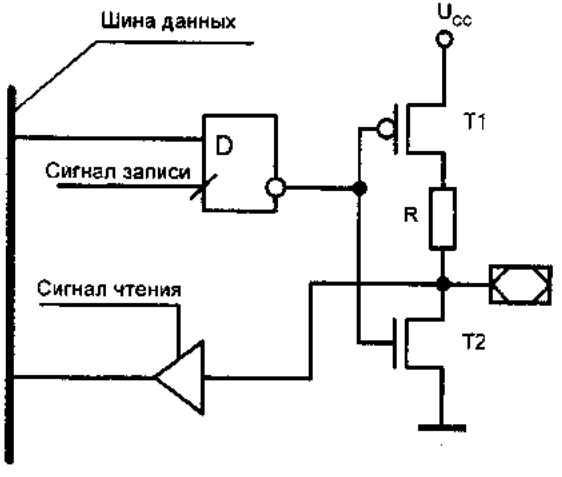


Рис. 6.6. Вариант организации двунаправленного вывода

Схемотехнически этот вариант проще предыдущего, но это достоинство сопровождается и некоторыми недостатками. В режиме вывода потенциал контактной площадки зависит от состояния триггера данных. При записи в триггер единичного сигнала низкий уровень напряжения на его инверсном выходе открывает транзистор T1 и запирает транзистор T2. Высокий уро-

вень напряжения U_{CC} через резистор R выводится на внешний контакт. Если же в триггере данных записан нуль, состояния транзисторов изменяются на противоположные, и внешний вывод через проводящий транзистор T2 подключается к потенциалу "земли". Третье состояние в схеме не реализовано, поэтому рассмотренный режим вывода при работе схемы всегда имеет место, в том числе и при вводе данных. Для обеспечения режима ввода в триггер данных записывается единица, при этом указанные выше состояния транзисторов выводят на внешний контакт сигнал логической единицы. Однако при больших значениях сопротивления R это будет "слабая" единица, и источник вводимого сигнала преодолевает ее, задавая внешнему контакту свой потенциал. Недостаток такого решения состоит в необходимости применения относительно высокоомных резисторов R, что замедляет формирование положительных фронтов при переключении выхода (нагрузочные емкости медленно перезаряжаются малым током).

Еще один вариант организации линии порта показан на рис. 6.7.

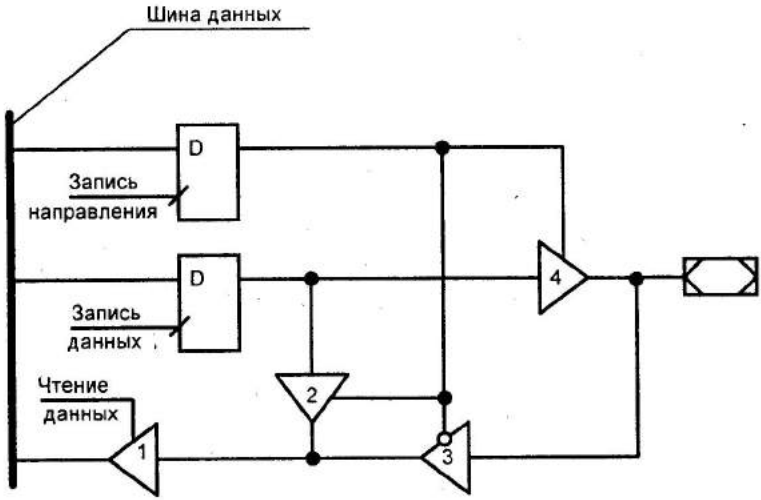


Рис. 6.7. Схема варианта двунаправленной линии порта

Запись нуля в триггер направления передачи запрещает работу буферов 4 и 2 и разрешает работу буфера 3, поэтому сигнал с внешнего вывода передается на шину данных по сигналу "Чтение данных". Если же в триггер направления записать единицу, то будет разрешена работа буферов 4 и 2 и запрещена работа буфера 3, т. е. сигнал от триггера данных будет передаваться на выход, а по сигналу "Чтение данных" на шину данных будет передаваться сигнал, записанный ранее в триггер данных, но не состояние внешнего контакта.

Параллельные периферийные адаптеры

Шинные формирователи осуществляют непосредственную, а порты буферизованную во времени передачу данных между МП и системной шиной. Более сложные операции выполняются периферийными адаптерами. *Программируемость адаптеров* обеспечивает им широкую область применения вследствие изменяемости процедур обмена с помощью команд программы, в том числе и во время работы микропроцессорной системы.

В схемах, обслуживающих обмен параллельными данными, как правило, используется базовая структура параллельного периферийного адаптера (ППА, PPI (Programmable Peripheral Interface)) Intel 8255A, имеющего отечественный аналог К580ВВ55А. Эти БИС представляют собою устройства параллельного ввода/вывода и обеспечивают двунаправленный обмен с квитированием или без него при программном обмене, инициатива которого исходит от программы или от запросов прерывания. С помощью ППА внешние устройства, работающие с параллельными кодами, связываются с магистралью системы.

Адаптер типа 55А (рис. 6.8) имеет три двунаправленных восьмиразрядных порта PA, PB и PC, причем порт PC разделен на два четырехразрядных канала: старший PC_H и младший PC_L. Обмен информацией между каналами A, B, C и шиной данных МПС производится через буфер данных BD в соответствии с сигналами управления.

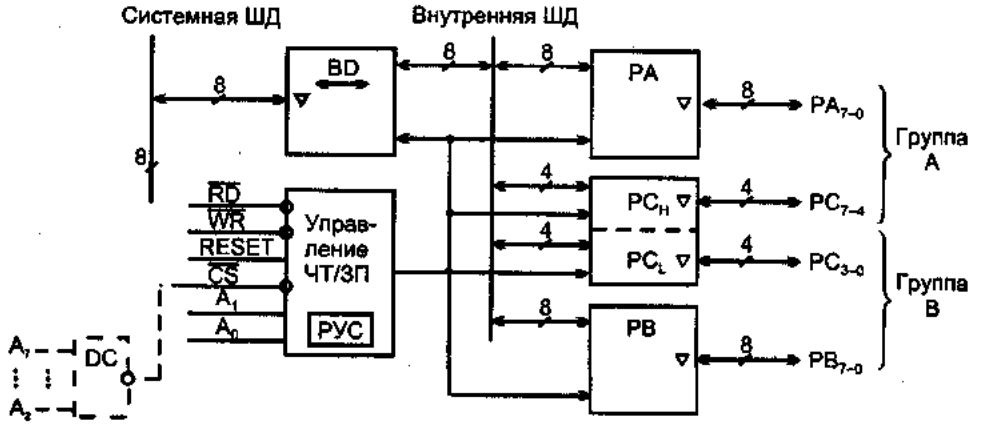


Рис. 6.8. Структура параллельного периферийного адаптера

Блок управления чтением/записью получает стробы чтения и записи $\overline{RD}$ и $\overline{WR}$ (это сигналы $\overline{IOR}$ и $\overline{IOW}$ стандартного интерфейса), сигнал сброса RESET, сигнал выбора адаптера $\overline{CS}$, получаемый декодированием старших

разрядов его адреса, и два младших разряда адреса A_1 и A_0 для адресации внутренних регистров. Внутренних адресуемых объектов 5: три порта (A, B и C), регистр управляющего слова РУС и команда установки/сброса битов порта C BSR (Bit Set/Reset). Адресация и направление передач информации определяются согласно табл. 6.2.

Таблица 6.2

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Операция
0	0	0	1	0	Порт A → ШД
0	1	0	1	0	Порт B → ШД
1	0	0	1	0	Порт C → ШД
1	1	0	1	0	Запрещенная комбинация
0	0	1	0	0	ШД → Порт A
0	1	1	0	0	ШД → Порт B
1	0	1	0	0	ШД → Порт C
1	1	1	0	0	ШД → РУС при D7 = 1
					ШД → BSR при D7 = 0
X	X	1	1	0	Шины отключены
X	X	X	X	1	Шины отключены

Как видно из таблицы, адрес $A_1A_0 = 11$ соответствует передаче управляющих слов РУС (УС1) или BSR (УС2), причем по этому адресу допускается только запись. Передача двух разных УС при одном и том же адресе возможна только потому, что признаком того или иного УС служит значение старшего бита передаваемого слова (D7). Таким образом, этот бит выполняет дополнительную адресацию управляющих слов.

Работа адаптера начинается после загрузки с ШД в РУС управляющего слова УС1, задающего портам адаптера один из трех возможных режимов и направленность порта (ввод или вывод).

Возможны *три режима работы портов*: 0, 1 и 2, причем порт A может работать в любом из трех режимов, порт B только в двух (0 и 1), а режим порта C зависит от режимов портов A и B.

Порт C имеет особенности: в отличие от портов A и B, которые оперируют со словами в целом, разряды порта C могут программироваться и использоваться поодиночке. В частности, любой из восьми разрядов порта C может быть установлен или сброшен программным способом. Это нужно для пере-

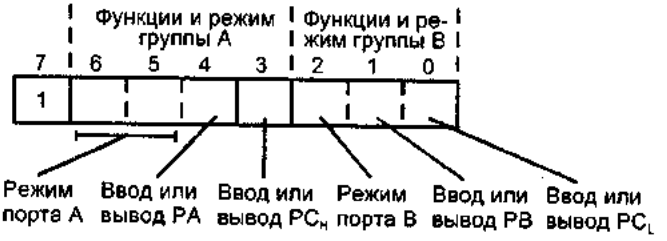
дач сигналов квитирования при обмене через порты A и B в режимах 1 и 2. При работе порта в режиме 1 для него под сигналы управления требуются три линии, в режиме 2 — пять.

Режимы работы портов:

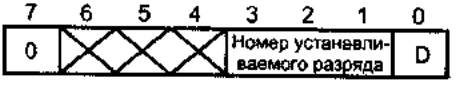
- режим 0 — однонаправленный ввод/вывод без квитирования, в этом режиме могут работать порты A и B, а также свободные (не занятые передачей служебных сигналов для портов A и B) линии порта C;
- режим 1 — однонаправленный ввод/вывод с квитированием;
- режим 2 — двунаправленный ввод/вывод с квитированием.

Квитирование, как известно, позволяет вести асинхронный обмен с учетом готовности абонента к передаче, т. е. иметь переменный темп обмена соответственно возможностям внешнего устройства.

Формат управляющего слова УС1 показан на рис. 6.9, *а*. Разряд 7 содержит единицу, что является признаком управляющего слова УС1. Разряды 6...3 определяют режим и вид портов A и свободных от служебных сигналов линий порта C_H (старшей половины порта), а разряды 2...0 — то же для порта B и младшей половины порта C (C_L).



а



б

Рис. 6.9. Форматы управляющих слов параллельного периферийного адаптера

Режим порта A выбирается по условиям: 00 — режим 0, 01 — режим 1, 1X — режим 2. Порт B имеет режим 0 или 1 при нулевом или единичном значении разряда 2 соответственно. Единичные значения разрядов 4, 3, 1 означают ввод, нулевые — вывод.

При записи нового УС1 все регистры портов сбрасываются. Управляющее слово УС2 задает значения 0 или 1 одному из разрядов порта С. Для приведения в определенное состояние нескольких выходов порта С нужно подать в адаптер соответствующее число слов УС2. В итоге словами УС2 на выходах порта РС задаются коды, определяющие режим работы ВУ и изменяемые программным способом.

Формат управляющего слова УС2 показан на рис. 6.9, *б*. Признаком этого слова служит нулевое значение разряда 7. Разряды 6...4 не используются. В разрядах 3...1 размещается двоичный код номера разряда, приводимого в то или иное состояние в порте С данным УС2. В нулевом разряде указывается состояние (0 или 1), которое следует придать данному разряду.

В режиме 0 осуществляется прямой однонаправленный ввод/вывод данных без сигналов их сопровождения. Каждый из четырех портов может быть использован для ввода или вывода независимо от других, так что возможны 16 вариантов режима 0. При вводе поступающая из ВУ информация адаптером не фиксируется и должна присутствовать на его входе во время действия сигнала чтения. При выводе информация от МП фиксируется в буферном регистре порта по заднему фронту сигнала записи и сохраняется до нового цикла вывода или смены режима работы порта.

При вводе информация выдается на ШД при выполнении микропроцессором команды IN port, при выводе — при выполнении команды OUT port.

Такой вариант соответствует работе "с раздельной шиной", при которой внешним устройствам принадлежит отдельное адресное пространство. Не исключается и организация обращения к портам, как к ячейкам памяти (интерфейс "с общей шиной").

В режиме 1 каждая из двух 12-разрядных групп (А и В) может быть запрограммирована на однонаправленный ввод или вывод с квитированием. При этом входные и выходные данные фиксируются адаптером. По линиям портов C_H и C_L передаются управляющие сигналы. Раздельная установка разрядов порта С позволяет ему играть роль схемы управления процедурами ввода/вывода, причем битам порта придается определенное функциональное назначение.

Режим 1 рассмотрим в полном объеме, т. к. он хорошо иллюстрирует принципы работы адаптера. При вводе используются следующие управляющие сигналы:

- $\overline{\text{STB}}$ — строб загрузки данных в регистр (по заднему фронту);
- IBF (Input Buffer Full) — входной буфер полон, сигнал подтверждения загрузки данных;
- INT — запрос прерывания.

Временные диаграммы процесса ввода в режиме 1 показаны на рис. 6.10, *а*.

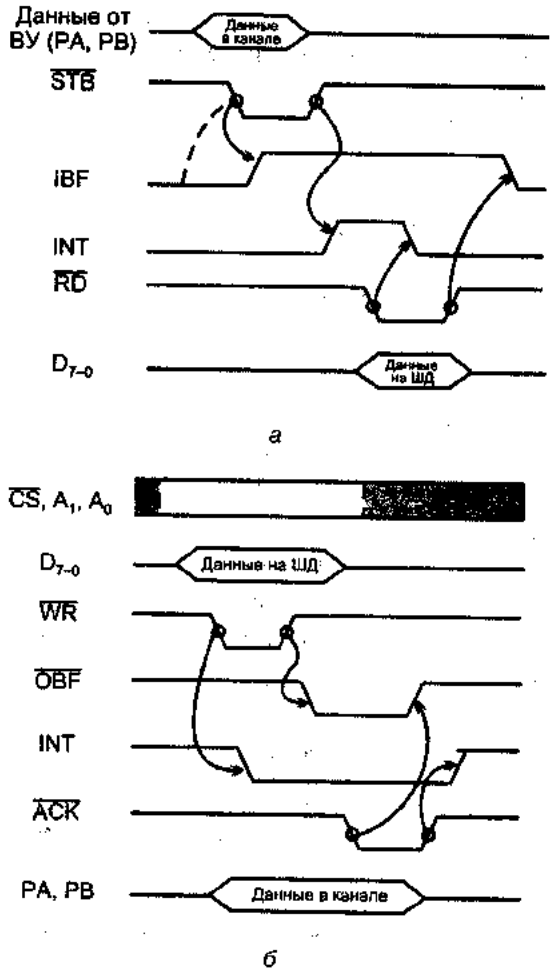


Рис. 6.10. Временные диаграммы процессов ввода (*а*) и вывода (*б*) в режиме 1 параллельного периферийного адаптера

Имея данные для ввода в порт, ВУ при условии IBF = 0 вырабатывает сигнал готовности информации $\overline{\text{STB}}$. Передний фронт этого сигнала устанавливает сигнал IBF, запрещающий внешнему устройству ввод следующего слова до освобождения порта. К моменту окончания $\overline{\text{STB}}$ данные введены в буфер порта, и если прерывания разрешены (внутренний триггер разрешения прерываний INTE установлен командой программы), то адаптер формирует запрос прерывания INT для МП, переходящего к подпрограмме обслуживания, содержащей команду IN port. При этом на адаптер поступают сигналы адресации и $\overline{\text{RD}}$. Передний фронт $\overline{\text{RD}}$ отмечает начало считыва-

ния слова микропроцессором и снимает запрос на прерывание INT. Пока прерывания не разрешены, осуществляется хранение данных в адаптере. Задний фронт $\overline{RD}$ отмечает завершение считывания слова микропроцессором и снимает сигнал IBF, допуская новую запись слова со стороны ВУ.

При выводе используются следующие управляющие сигналы:

- $\overline{OBF}$ (Output Buffer Full) — выходной буфер полон, строб вывода новых данных;
- $\overline{ACK}$ (Acknowledge) — подтверждение приема внешним устройством;
- INT (Interrupt) — запрос прерывания.

Временные диаграммы вывода в режиме 1 показаны на рис. 6.10, *б*. При выводе выполняется команда OUT port, и процессор устанавливает адрес порта и данные на ШД. При разрешенных прерываниях далее вырабатывается сигнал $\overline{WR}$, загружающий данные с ШД в буфер адаптера и сбрасывающий запрос прерывания INT. После окончания записи в адаптер формируется сигнал $\overline{OBF}$, указывающий на готовность данных для ВУ. Приняв данные, ВУ выдает сигнал подтверждения приема $\overline{ACK}$, снимающий $\overline{OBF}$, а по окончании сигнала $\overline{ACK}$ восстанавливается запрос прерывания (если триггер INTE установлен), что вызывает обслуживание следующего цикла вывода.

Сигналы управления при обменах с квитированием передаются по отдельным линиям порта С, специально для них предназначенным. Распределение управляющих сигналов по этим линиям, формирование в адаптере слова состояния и управляющее слово для ввода через порт А в режиме 1 показаны для примера на рис. 6.11.

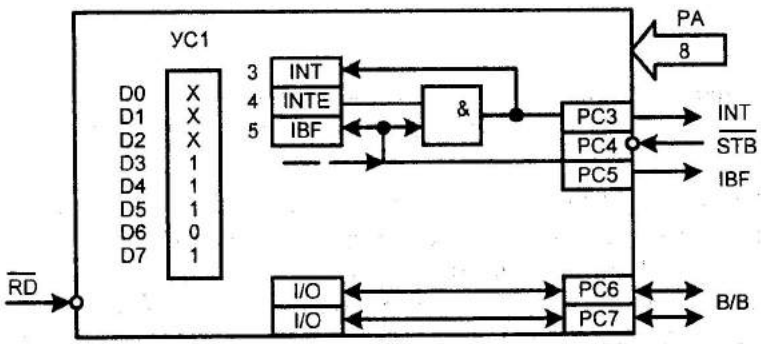


Рис. 6.11. Структура порта С, формирование слова состояния и использование разрядов порта С для ввода через порт А в режиме 1

Слева показаны разряды управляющего слова УС1, в следующем столбце — разряды слова состояния, относящиеся к порту А, правее обозначено использование разрядов порта С в рассматриваемом режиме. Сигнал IBF отображается в слове состояния и выдается во внешнюю среду для ВУ. Наличие этого сигнала и сигнала разрешения прерывания INTE ведет к выработке запроса на прерывание, отображаемого в слове состояния и поступающего во внешнюю среду через разряд PC3. Флажок INTE создает возможность маскирования запросов прерывания, позволяющего запрещать или разрешать работу ВУ. Сигнал STB принимается от внешнего устройства в разряд PC5. Свободные линии порта С могут быть использованы для простого ввода/вывода.

В слове состояния имеются независимые сигналы разрешения прерываний для ввода и вывода. Контроль текущего состояния портов в режимах 1 и 2 путем считывания порта С командой IN port позволяет анализировать процесс обмена, которым можно оперативно управлять.

В режиме 2 осуществляются двунаправленные передачи между ШД и ВУ. Особенности порта А допускают его применение для такого режима. При этом 5 линий порта С передают управляющие сигналы.

Двунаправленный асинхронный обмен через порт А выполняется как последовательность нескольких независимых этапов: записи с ШД в адаптер, ввода в адаптер из ВУ, чтения на ШД, вывода в ВУ, причем некоторые из них могут совмещаться во времени. Используются сигналы управления: $\overline{\text{STB}}$, IBF, $\overline{\text{OBF}}$, $\overline{\text{ACK}}$, INT, т. е. те же, что и для режима 1.

Ввод в адаптер управляющих слов УС1 и УС2 производится программным способом с помощью последовательности команд непосредственной загрузки аккумулятора и вывода данных в адресованный порт. На языке ассемблера фрагмент программы имеет вид:

```
MVI A, b2

OUT port,
```

где загружаемый в аккумулятор байт b_2 представляет собою вводимое в адаптер слово УС1 или УС2, а port — адрес регистров управления, шесть старших разрядов которого дают номер (адрес) адаптера, а два младших содержат единицы. Указанный фрагмент программы повторяется столько раз, сколько необходимо для задания адаптеру режима и функций, а выходам порта С нужных значений.

Улучшенный вариант адаптера ВВ55А отличается от предшественника ВВ55 работой с расширенным стробом записи, свойственным, в частности, и микропроцессору К1821ВМ85А.

Подробное описание рассмотренного ППА имеется в работах [28], [37], [50] и др.

Для связи с периферийными устройствами, удаленными от МПС (на расстояние не более 15 м), применяется интерфейс ИРПР (интерфейс радиаль-

ный параллельный), осуществляющий однонаправленные асинхронные передачи по 8- или 16-разрядной шине (в базовом варианте). Логические требования интерфейса ИРПР могут быть выполнены при использовании адаптера ВВ55/55А.

Пример использования ППА дан на рис. 6.12 для схемы подключения аналого-цифрового преобразователя (АЦП) и цифроаналогового преобразователя (ЦАП) к МПС, выполняющей задачу цифрового управления некоторым аналоговым объектом.

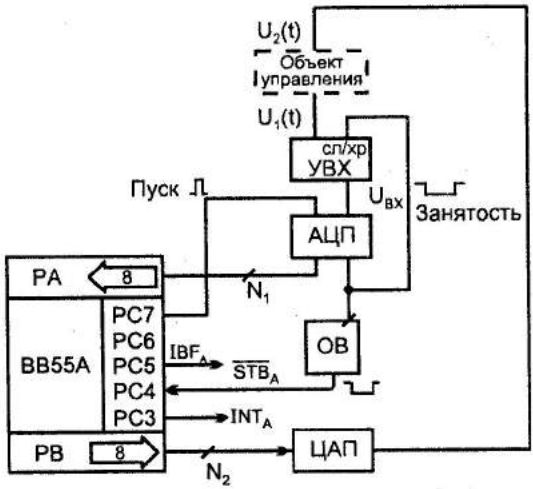


Рис. 6.12. Пример схемы использования параллельного периферийного адаптера для подключения АЦП и ЦАП к шинам микропроцессорной системы

Состояние объекта отображается сигналом напряжения постоянного тока $U_1(t)$, которое преобразуется в цифровой код N_1 и передается через адаптер процессору. Процессор согласно алгоритму управления объектом вырабатывает сигнал воздействия на него в виде кода N_2, который далее преобразуется в напряжение $U_2(t)$, воздействующее на объект. Для предотвращения ошибок в работе АЦП на время преобразования "напряжение-код", изменение входного напряжения АЦП должно быть исключено. Поэтому в схему введено устройство выборки/хранения (УВХ), имеющее два режима: слежения и хранения. В режиме слежения выходное напряжение УВХ повторяет входное, в этом режиме УВХ находится все время за исключением интервалов работы АЦП. Когда АЦП переходит в режим преобразования "напряжение-код", УВХ переводится в режим хранения, и изменение его выходного напряжения прекращается на время преобразования.

Работа схемы происходит следующим образом. Получив сигнал "Пуск" от адаптера, АЦП начинает преобразование "напряжение-код", причем для упрощения

схемы принято, что АЦП является восьмиразрядным. При этом АЦП сигналом "Занятость" переводит УВХ в режим хранения. Завершение процесса преобразования отмечается окончанием сигнала "Занятость", т. е. положительным перепадом напряжения на соответствующем выходе АЦП, запускающим одновибратор ОВ, который вырабатывает строб готовности данных $\overline{STB}_A$ для порта ввода РА. Строб загружает данные (код N_1) в порт А адаптера, при этом, как известно, формируются сигналы IBF_A и INT_A. Сигнал INT_A является запросом процессору на ввод байта из порта А.

Процессор выполняет подпрограмму обслуживания запроса и вводит код N_1 командой IN port.

Выработанный процессором код N_2 выводится через порт В адаптера на вход ЦАП. ЦАП представляет собою устройство, всегда готовое к работе (например, схему на основе сетки R-2R), поэтому для него приемлем прямой безусловный вывод. Выходное напряжение ЦАП воздействует на управляемый объект для обеспечения предписанного ему поведения.

Необходимый порядок работы блоков схемы обеспечивается при выполнении фрагмента программы, предусматривающего инициализацию адаптера и выработку сигнала "Пуск", после чего ввод кода в процессор будет обеспечен работой аппаратуры.

Формат управляющего слова УС1 определится соображениями: порт А работает как порт ввода с квитированием, а порт В как порт прямого вывода, что задает следующие значения битам УС1: $D_7 = 1$ (признак УС1), $D_6D_5 = 01$ (режим 1 порта А), $D_4 = 1$ (ввод для порта А), $D_3 = 0$ (эта линия выделена для вывода сигнала "Пуск"), $D_2 = 0$ (режим 0 для порта В), $D_1 = 0$ (вывод для порта В), $D_0 = 0$ (эта линия не используется, и ее состояние безразлично, для определенности принято состояние 0).

Таким образом, УС1 = 10110000 = B0H.

Формат управляющего слова УС2 должен обеспечить разрешение прерываний для порта А, чему соответствует условие $INTE_A = 1$. Так как в качестве флажка INTE в адаптере используется триггер разряда PC4, его нужно установить, т. е. принять $D_3D_2D_1 = 100$, $D_0 = 1$. Приняв состояния неиспользуемых разрядов $D_7D_6D_5D_4$ нулевыми, получим УС2 = 00001001 = 09H.

Пусть портам адаптера присвоены адреса: порту А адрес F0H, порту В — F1H, порту С — F2H и РУС — F3H. В этих адресах значения младших разрядов A_1A_0 соответствуют требованиям к адресации портов адаптера, а старшие разряды выбраны произвольно.

Программа инициализации будет такой:

```
MVI A, B0H      ; Загрузка в аккумулятор кода B0H при
                ; непосредственной адресации
OUT 0F3H        ; Загрузка УС1 в РУС адаптера
MVI A, 09H      ; Загрузка в аккумулятор команды
                ; устновки/сброса битов порта
OUT 0F3H        ; Установка бита 4 порта
```

После инициализации адаптер готов к работе и может начать процесс преобразования "напряжение-код" и ввода кода в процессор следующим образом:

```
MVI A, 0FH      ; Загрузка в аккумулятор команды
                ; установки бита P7
OUT 0F3H        ; Установка бита P7
MVI A, 0EH      ; Загрузка в аккумулятор команды
                ; сброса бита P7
OUT 0F3H        ; Сброс бита P7
```

Дальнейшая работа аппаратуры согласно описанному ранее порядку заканчивается вводом кода N1 в процессор.

Для вывода байта через адаптер на вход ЦАП достаточно выполнить команду OUT F2H.

§ 6.4. Последовательные интерфейсы и программируемые связные адаптеры

Общие сведения

При увеличении расстояний, на которые передаются данные, трудно обеспечить помехоустойчивую работу параллельных связей, кроме того, они становятся неприемлемо дорогими. В этом случае для передачи данных применяют их преобразование в последовательные. Кроме того, многие ВУ оперируют с последовательными кодами и для взаимодействия с процессором нуждаются в преобразовании данных из параллельной формы в последовательную и наоборот. Последовательные передачи используются также при связи удаленных объектов по обычным телефонным сетям. Задачи преобразования параллельных данных в последовательные и наоборот решаются в зависимости от их особенностей либо достаточно простыми модулями, либо более сложными БИС с широкими функциональными возможностями.

Тракт передачи последовательных данных в полном варианте (рис. 6.13, *а*) включает в себя источник и приемник данных (процессор и внешнее устройство), преобразователи параллельных данных в последовательные и наоборот (в этой роли выступают программируемые связные адаптеры ПСА) и модемы. Такой тракт соответствует взаимодействию процессора с ВУ, оперирующими параллельными кодами, но находящимися на большом расстоянии от процессора. Вместо ПСА возможно использование и других модулей аналогичного назначения.

Модемы (модуляторы-демодуляторы) преобразуют двоичные импульсные сигналы (последовательности нулей и единиц) в некоторый аналоговый мо-

дулированный сигнал, приспособленный к передаче по узкополосным телефонным линиям. Узкополосность телефонных линий (полоса пропускания около 3 кГц) ограничивает их *бодовую скорость*.

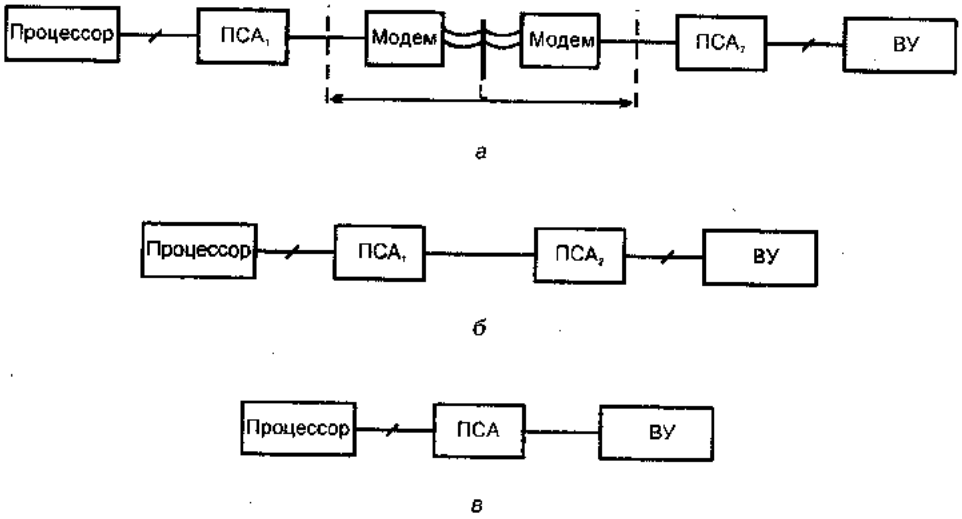


Рис. 6.13. Структура трактов передачи данных

В бодах измеряют число состояний канала в секунду. Количество изменений состояний канала в секунду из-за узкополосности линии невелико, и если состояния будут соответствовать просто двоичным цифрам 0 и 1, битовая скорость передачи, измеряемая в бит/с, будет мала. С помощью разных видов модуляции (фазовой, частотной, амплитудной) и их сочетаний получают сигнал, в котором один бодовый интервал содержит как бы несколько бит, так что битовая скорость в несколько раз выше бодовой. Например, если при фазовой модуляции синусоидального сигнала на бодовом интервале можно задавать четыре фазы сигнала (–90°, 0°, 90° и 180°), то это означает удвоение битовой скорости относительно бодовой. Современные модемы имеют битовые скорости передачи не менее 38,4 Кбит/с.

Если расстояние L между источником и приемником информации значительно, но не настолько велико, чтобы потребовались передачи по телефонным или подобным им сетям, то часть тракта с модемами не нужна и тракт передачи будет иметь вид (рис. 6.13, *б*), где $ПСА_1$ преобразует параллельные данные в последовательные, а $ПСА_2$ — последовательные в параллельные. Если требуется взаимодействие процессора с относительно недалеко расположенным ВУ, оперирующим с последовательными кодами, тракт передачи будет иметь вид (рис. 6.13, *в*).

При обмене последовательными данными передается, как правило, *символьная информация* (буквы, цифры и другие знаки). Символы кодируются группой битов, число которых обычно лежит в пределах от 5 до 8. Если разрядность группы 5, то непосредственно можно отображать до 32 различных символов. Такую разрядность имеет телеграфный код, в котором, однако, за счет дополнительных признаков принадлежности кода к той или иной регистровой группе число воспроизводимых символов расширено до 78.

Международное признание получил американский стандартный код обмена информацией ASCII (American Standard Code for Information Interchange), в котором символы кодируются семью двоичными разрядами. Этот код позволяет передавать цифры, прописные и строчные буквы латинского алфавита, целый ряд других символов (всего 96 символов, т. к. 32 кодовые комбинации выделены для представления команд обмена). На основе этого кода построен отечественный код КОИ-7 (код обмена информацией семиразрядный). Применяется также восьмиразрядный код ДКОИ-8.

Система передачи может быть *симплексной*, *полудуплексной* или *дуплексной*. В первом случае данные передаются только в одну сторону, во втором — в обе, но с разделением во времени, в третьем — в обоих направлениях одновременно.

Важнейшее требование правильного приема — определение приемником моментов времени, в которые следует воспринять очередной бит данных. Иными словами, речь идет о синхронизации процессов в передатчике и приемнике. Можно связать передатчик с приемником специальной линией для синхронизации. Можно обойтись и без такой линии, применив *синхронизацию приемника самим передаваемым сигналом*, для чего сигналу придается определенная структура.

Протоколы последовательного обмена задают два его вида: *асинхронный и синхронный*. При асинхронном обмене символы передаются по мере их готовности. Интервал между символами может быть различным, хотя интервалы между битами в одном символе фиксированы. При отсутствии готовых данных линия простаивает.

При синхронной передаче символы следуют один за другим слитно, поэтому можно говорить о передаче массива символов — текста. Если очередной символ не готов, передача не останавливается, передатчик посылает в линию синхросимволы до тех пор, пока не сможет передать следующий символ данных. Синхронный обмен повышает скорость передачи данных.

Скорость передачи оценивается числом передаваемых в секунду битов и соответствует типовому периферийному оборудованию. Принят ряд стандартных значений скорости, к которому принадлежат, в частности, 300, 1200, 2400, 4800, 9600, 14400, 19200, 33600, 56000 бит/с.

При асинхронных передачах *посылка (кадр)* , т. е. группа битов, отображающих символ, имеет следующий формат: начало посылки отмечается нулевым старт-битом, за ним следуют 5—8 информационных (младшим разрядом вперед), затем идет необязательный бит контроля по модулю 2 (бит четности/нечетности) и заканчивается посылка 1; 1,5 или 2 единичными стоп-битами (рис. 6.14, *а*).

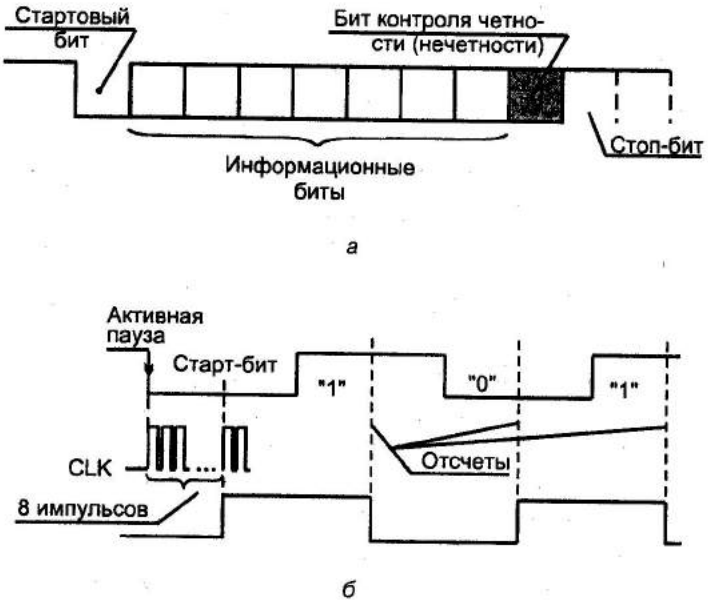


Рис. 6.14. Структура кадра для асинхронных передач (*а*) и временные диаграммы формирования временных меток в середине бита (*б*)

В отсутствие передачи линия находится под высоким потенциалом (активная пауза, *марка*), соответствующим логической единице. Появление низкого уровня означает поступление старт-бита, свидетельствующего о последующей передаче известного заранее числа информационных битов. Далее может идти контрольный бит четности (нечетности), назначение и способ выработки которого уже известны *(см. § 2.7)*. Стоп-бит также используется для проверки правильности передачи, но уже по другому критерию. Контролируется правильность формата посылки. Отсутствие на позиции стоп-бита высокого уровня напряжения свидетельствует об ошибке формата (кадра, обрамления). Длительность стоп-бита определяет минимальный промежуток между окончанием данного символа и началом следующего. Этот промежуток составляет 1—2 интервала, соответствующих биту.

Приемник синхронизируется самим сигналом и должен считывать значения битов в серединах их интервалов, где искажения импульсов меньше всего влияют на величину считываемого уровня. Это достигается следующим образом. *Передатчик и приемник имеют свои генераторы тактовых импульсов, работающие на одинаковой частоте.* При отсутствии передачи передатчик устанавливает в линии высокий уровень напряжения (марку). Появление нуля (старт-бита) отмечает начало передачи, которое, таким образом, фиксируется фронтом напряжения "1 — 0". От этого фронта начинает работать генератор приемника. Приемник выдерживает интервал в половину длительности бита, проверяет, есть ли еще нуль на входе (контролирует истинность старт-бита с целью исключить реакцию на кратковременную помеху), и затем начинает воспринимать данные с интервалом в длительность бита (если старт-бит не подтвердился, то приемник возвращается в исходное состояние).

Частоты генераторов передатчика и приемника реально отличаются, поэтому отсчеты постепенно "сползают" с середины битов и смещаются к тому или другому краю импульсов. Однако за время короткой посылки (не более 10—11 битов) смещение отсчетов с середины битов легко сделать пренебрежимо малым.

Выборка отсчетов в середине битов производится благодаря наличию в ПСА частоты, более высокой, чем частота следования битов (обычно в 16 раз). После пуска генератора CLK с помощью счетчика отсчитывается 8 импульсов, что и отмечает середину старт-бита. Затем отсчеты повторяются с интервалом τ, получаемым от деления частоты CLK на 16 (рис. 6.14, *б*).

В конце проверяется стоп-бит, отсутствие при этой проверке высокого уровня напряжения устанавливает триггер *ошибки формата*. Если это запрограммировано, то проверяется и четность веса посылки с учетом контрольного разряда. Для фиксации результата этой проверки также имеется специальный триггер *контроля четности*.

Принятый символ поступает в регистр хранения для последующей передачи в виде параллельного кода. После этого приемник ищет следующий символ и вдвигает его в регистр сдвига. В регистр хранения второе слово не идет, пока не считано первое. В это время может пойти третий символ, тогда символ будет потерян, поскольку хранить его негде. Это *ошибка пропуска* (переполнения), которая тоже фиксируется установкой соответствующего триггера-флажка в регистре состояния адаптера. Ошибка пропуска не возникает, если микропроцессор считывает слово за интервал, меньший, чем интервал ввода символа в сдвигающий регистр.

Различают *две разновидности синхронных передач* — с внутренней и внешней синхронизацией.

При внутренней синхронизации перед массивом данных передаются *слова-синхросимволы* (одно или два). При отсутствии передачи передатчик не перестает работать, а посылает в линию символы синхронизации, пока не во-

зобновится передача данных. Приемник при этом находится в режиме активного ожидания (в английской терминологии в режиме Hunt — охоты). Он сравнивает каждое принятое слово с символом синхронизации. Если результат сравнения отрицательный, то обращения к данному приемнику нет (по описанному протоколу к одному передатчику можно подключить несколько приемников, имеющих индивидуальные синхросимволы). Если же опознается синхросимвол данного приемника, то это означает, что передатчик обращается к нему и первое же слово, не являющееся синхросимволом, принимается как информационное, начинающее информационный массив. После начала массива приемник считает передаваемые символы или же сопоставляет их с символами синхронизации, определяя одним из этих способов конец передачи.

Символы данных не разделяются старт- и стоп-битами. После символа из 5—8 битов может идти контрольный бит, возможен и контроль по модулю 2 для всего массива, в этом случае контрольный бит появляется в конце передачи данных.

При внешней синхронизации в канал связи вводится дополнительная линия, по которой передается строб-сигнал, отмечающий интервал времени, соответствующий передаче данных. Фронты строба отмечают начало и конец передачи массива, в котором символы по-прежнему передаются слитно (без старт- и стоп-битов).

Программируемые связные адаптеры (ПСА)

На рис. 6.15 показана структура ПСА (PCI, Programmable Communication Interface) типа 8251A, аналогом которого является отечественный ПСА К580ВВ51А. Согласно типу реализуемых протоколов, этот ПСА называют универсальным синхронно-асинхронным приемопередатчиком (УСАПП), чему в английской терминологии соответствует USART — Universal Synchronous/Asynchronous Receiver/Transmitter.

Адаптеры, в которых реализуются только асинхронные протоколы, называются УАПП (UART — Universal Asynchronous Receiver/Transmitter).

В МПС адаптер трактуется как ВУ, программируется процессором для работы с различной аппаратурой, принимает от процессора символы в параллельной форме и преобразует их в последовательную для передачи или получает последовательные данные и преобразует их в параллельные символы для процессора. Кроме того, адаптер сигнализирует процессору о готовности принять новый символ для передачи или о том, что получил символ для процессора. В любое время процессор может читать слово состояния адаптера. Адаптер обеспечивает также обмен сигналами квитирования с терминалом (модемом).

Рис. 6.15. Структура программируемого связного адаптера

Буфер ШД — двунаправленный, восьмиразрядный, с тремя состояниями. Он связывает адаптер с системной шиной данных и принимает данные по командам OUT port, выдает — по командам IN port. Через буфер передаются также управляющие и командные слова и слово состояния адаптера. В буфере имеются регистры данных (входной и выходной), команд и состояния.

Блок управления чтением/записью принимает сигналы от системной шины данных и генерирует сигналы управления работой всех блоков адаптера.

Выводы и сигналы ПСА имеют следующее назначение:

- RESET — установка адаптера в исходное состояние, после которой адаптер находится в бездействии до записи управляющих слов, определяющих задаваемые ему функции. В состояние бездействия адаптер вводится также программой по команде сброса;
- CLK — вход процессорной тактовой частоты. Внешние входы и выходы адаптера не привязаны к тактам сигнала CLK, но частота этого сигнала должна быть выше частоты передачи данных не менее, чем в 30 раз;
- $\overline{RD}$, $\overline{WR}$ и $\overline{CS}$ — сигналы, смысл которых хорошо известен (стробы чтения и записи и сигнал выбора микросхемы);
- C/D (Control/Data) — указывает на тип передаваемой информации, при единичном значении этого сигнала вводятся управляющие слова или выводится слово состояния адаптера, при нулевом — передаются данные. Вместе с сигналами $\overline{RD}$ и $\overline{WR}$ определяет характер передачи. Обычно на этот

вход подключается младший разряд адреса A_0. Направления передачи и характер информации задаются для адаптера таблицей (табл. 6.3).

Таблица 6.3

C/D	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	Операция
0	0	1	0	ШД ← данные адаптера
0	1	0	0	Данные адаптера ← ШД
1	0	1	0	ШД ← слово состояния
1	1	0	0	Управляющее слово ← ШД
X	1	1	0	Отключено
X	X	X	1	Отключено

Адаптер имеет набор управляющих входных и выходных сигналов для управления модемом. Модем указан здесь как наиболее типичное устройство, работающее во взаимодействии с ПСА, хотя, в сущности, это сигналы общего назначения, которые могут быть использованы и для работы с другими устройствами. Для управления модемом (терминалом) имеются две пары сигналов квитирования;

- $\overline{DSR}$ (Data Set Ready) — сигнал готовности от передатчика терминала, сигнал связан с одноразрядным портом и может быть проверен процессором чтением слова состояния. Низкий уровень этого сигнала говорит о том, что модем (терминал) имеет информацию для передачи;
- $\overline{DTR}$ (Data Terminal Ready) — сигнал является реакцией на запрос $\overline{DSR}$. Активизируется соответствующим битом командного слова, если процессором разрешен обмен с модемом. Разрешает модему посылку данных на вход приемника адаптера;
- $\overline{RTS}$ (Request to Send) — сигнал связан с одноразрядным выходным портом. Является запросом от адаптера готовности приемника терминала принять данные. Задается программированием соответствующего бита в командном слове, когда процессором разрешен обмен с модемом;
- $\overline{CTS}$ (Clear to Send) — сигнал готовности приемника терминала принять данные. Низкий уровень этого сигнала разрешает адаптеру передачу последовательных данных, если установлен бит TxEN в командном слове. При снятии TxEN или $\overline{CTS}$ во время работы передатчика он будет передавать все данные, записанные до запрещения передачи, прежде чем остановится.

Передатчик ПСА

Буфер передатчика (буфер Tx) принимает параллельные данные от буфера ШД, преобразует их в поток последовательных битов, вводит в этот поток служебные символы или биты и выдает составленный им поток битов на вывод TxD по отрицательным фронтам импульсов TxC. Передача начинается после ее разрешения и при условии $\overline{CTS} = 0$. Вывод TxD принимает высокий уровень после сброса, запретов по условиям TxEN или $\overline{CTS}$ либо при условии "передатчик пуст", связанном с сигналом TxE (TxEmpty).

Схема управления передатчиком (управление Tx) вырабатывает следующие внутренние и внешние сигналы для процессов передачи последовательных данных.

- TxRDY — этот выходной сигнал указывает процессору на готовность передатчика адаптера принять символ данных. Сигнал может проверяться чтением слова состояния или использоваться как запрос прерывания (он может маскироваться битом TxEN командного слова). Автоматически сбрасывается передним фронтом строба записи $\overline{WR}$, когда символ данных загружается из процессора.
- TxE — сигнал устанавливается, когда адаптер не имеет символа для передачи (входной буфер в блоке "буфер ШД" пуст, и после выхода символа из сдвигающего регистра передатчика этот регистр будет нечем загрузить). Сбрасывается после получения символа от процессора, если передача разрешена, и остается высоким, если передача запрещена соответствующим битом командного слова. Сигнал может быть использован для индикации конца передачи и оповещения процессора о моменте переключения линии на другое направление в полудуплексном режиме работы. В синхронном режиме высокий уровень сигнала показывает, что символ не был загружен и в поток данных следует вводить синхросимволы. Пока передаются синхросимволы, высокий уровень сигнала сохраняется.
- TxC и RxC — сигналы синхронизации передатчика и приемника, задающие скорость следования последовательных битов. При синхронных передачах бодовая скорость равна частоте TxC (RxC), при асинхронных она является частью частоты TxC (RxC) (это 1, или 1/16 или 1/64 от TxC или RxC). Очень часто частоты TxC и RxC идентичны. Их синхронности с сигналом CLK не требуется.

Приемник ПСА

Буфер приемника принимает последовательные данные, преобразует их в параллельные, проверяет биты или символы, специфичные для посылок

данного типа и посылает принятый символ в процессор. Для приемника ПСА характерны следующие сигналы и выводы.

- RxD этот вывод служит входом последовательных данных. Блок управления приемником Rx обеспечивает управление всеми действиями, связанными с приемом информации. Схемы этого блока предотвращают восприятие неиспользуемой линии данных как L-активной в режиме паузы. Для начала приема требуется появление высокого уровня (марки) на входе RxD после сброса системы. Если это выполняется, то разрешается поиск отрицательного фронта входного сигнала (старт-бита). Истинность старт-бита устанавливается проверкой уровня сигнала в его середине. Ошибки работы адаптера устанавливают соответствующие биты в слове состояния (четности, формата или переполнения, если новая информация замещает старую раньше, чем она была использована).
- RxRDY — выходной сигнал, показывающий, что адаптер имеет символ, готовый к выводу в процессор. Может проверяться чтением слова состояния или использоваться как запрос прерывания для процессора. Если команда разрешения приема RxEN отсутствует, то сигнал RxRDY находится в состоянии сброса. Отсутствие чтения принятого символа из выходного регистра адаптера до появления следующего ведет к загрузке нового символа и потере старого. Устанавливается ошибка переполнения.
- SYNDET (SYNC Detect/Break Detect) этот вывод в синхронном режиме используется как SYNDET и может быть входом или выходом в зависимости от программирования адаптера. При внутренней синхронизации является выходом и устанавливается как признак выявления синхросимвола в режиме приема. Если запрограммированы два синхросимвола, SYNDET установится в середине последнего бита второго синхросимвола. Сигнал автоматически сбрасывается после операции чтения состояния. Когда используется как входной (режим внешней синхронизации), его появление заставляет адаптер начать прием данных. В асинхронном режиме вывод используется для сигнала Break Detect, который устанавливается при низком уровне на интервалах стоп-битов в двух последовательных посылках. Сигнал может быть выявлен чтением слова состояния. Сбрасывается при сбросе адаптера или возвращении входного сигнала к нормальному состоянию (появлению единиц на интервалах стоп-битов).

Программирование ПСА

Адаптер программируется загрузкой в него по команде OUT port *начального и текущего управляющих слов*. Начальное управляющее слово (инструкция режима MI — Mode Instruction) вводится после сброса и задает режим работы адаптера, формат передаваемых символов, скорость передачи/приема, характеристику контроля, тип синхронизации. Формат MI показан на рис. 6.16. Как видно, трактовка разрядов D_7 и D_6 слова MI зависит от режима.

При синхронном обмене и внутренней синхронизации после инструкции режима MI в адаптер вводятся один или два синхросимвола, для хранения которых в схеме управления приемником имеются два специальных регистра.

Рис. 6.16. Формат управляющего слова режима программируемого связного адаптера

После синхросимволов или непосредственно после MI, если задан режим асинхронного обмена или синхронного обмена с внешней синхронизацией, в адаптер загружается командное слово CI, Command Instruction, называемое также текущим управляющим словом. Новое командное слово может быть загружено в адаптер в любое время, что позволяет оперативно влиять на процесс обмена.

Формат командного слова показан на рис. 6.17, *а*.

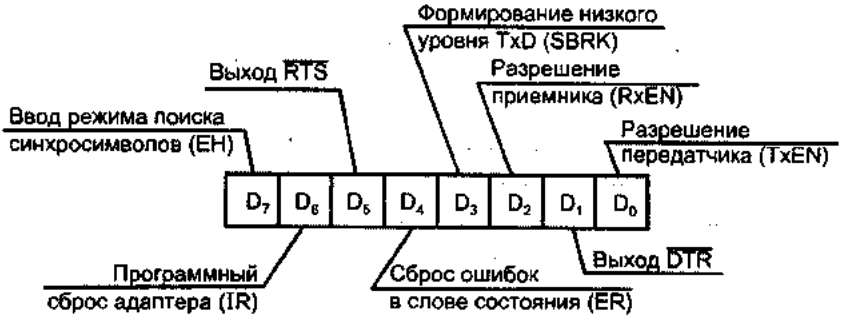
Правильная загрузка нескольких регистров без индивидуальных адресов у них обеспечивается жестким порядком записи управляющих слов в программируемый адаптер.

Формат слова состояния адаптера представлен на рис. 6.17, *б*. Структура программного блока, управляющего работой адаптера, приведена на рис. 6.18.

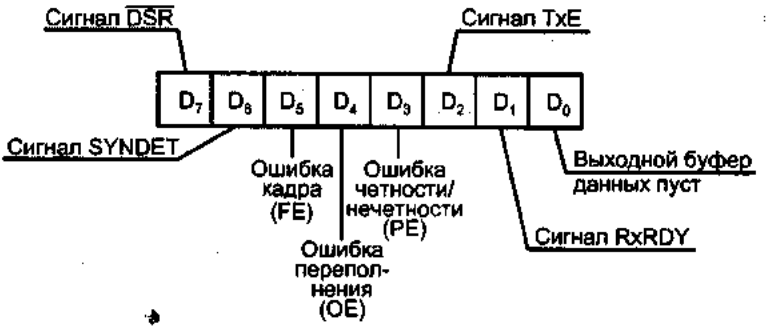
Адаптер может работать в одном режиме или комбинации совместимых режимов, осуществляя программный условный обмен процессора с ВУ или обмен по прерываниям. Первый вид обмена предусматривает программное чтение слова состояния адаптера и при его готовности выполнение подпрограммы обмена. При обмене по прерываниям сигналы готовности адаптера TxRDY и RxRDY используются как запросы прерывания для процессора.

Появление ошибок не останавливает работу адаптера. Ошибки выявляются установкой триггеров-флажков.

Рассмотрим для примера *временные диаграммы* процесса передачи в асинхронном старт-стопном режиме. В этом режиме после записи в адаптер параллельных данных они автоматически обрамляются старт- и стоп-битами, а при соответствующем программировании и битом контроля по модулю два. Если командным словом CI дано разрешение режима передач (D_0 = 1) и от терминала получено условие готовности $\overline{CTS}$, то на выход TxD начнет поступать поток битов с частотой, равной TxC или 1/16, или 1/64 этой частоты в зависимости от программирования адаптера. При отсутствии передачи на выходе TxD действует высокий уровень напряжения (марка). Если командным словом CI задана пауза, то уровень TxD становится низким.



а



б

Рис. 6.17. Формат командного слова (*а*) и формат слова состояния адаптера (*б*)

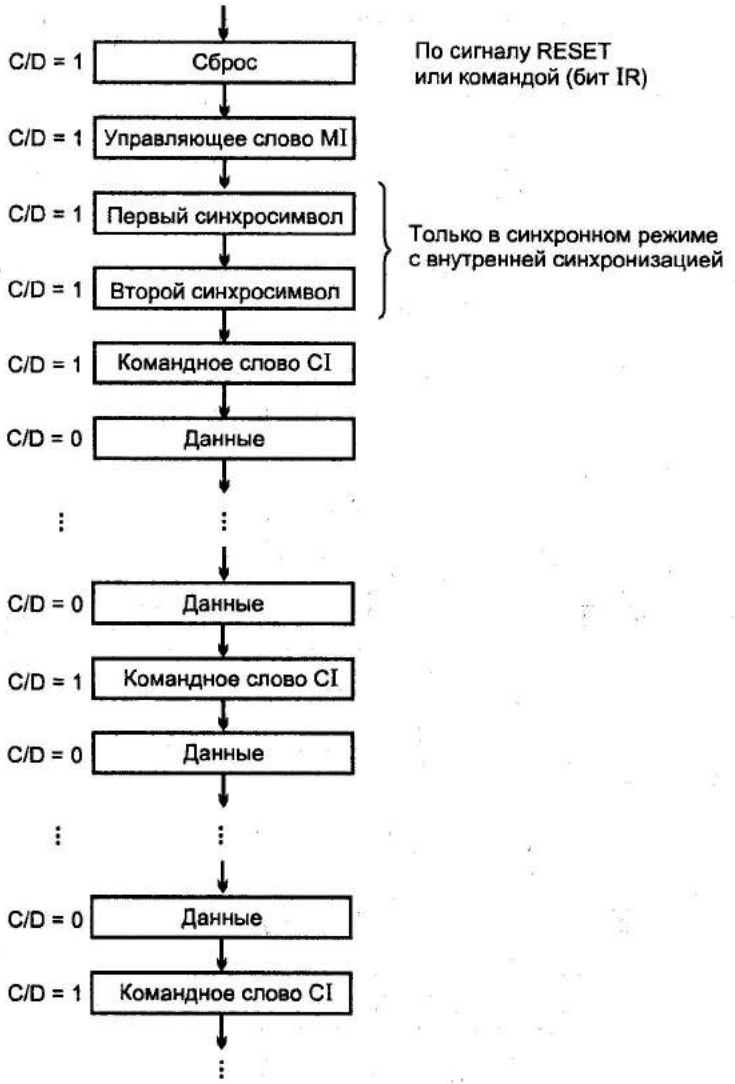


Рис. 6.18. Структура программного блока управления работой программируемого связного адаптера

При программном условном обмене (рис. 6.19) процессор осуществляет регулярный опрос состояния адаптера чтением слова состояния SW (Status Word). При готовности адаптера (TxRDY = 1, т. е. входной буфер пуст) выдается строб записи $\overline{WR}$, который передним фронтом снимает сигнал готовности (буфер уже занят), а задним, когда символ уже получен адаптером, начинает процесс передачи кадра (выталкивания символа из регистра сдвига). Начало

выдачи кадра говорит о том, что буфер шины данных освободился (символ уже в регистре передатчика) и нужно вернуть сигнал TxRDY в состояние 1. Вторая запись снимает готовность буфера, и его неготовность продлится до конца передачи первого кадра, за которой произойдет перегрузка символа из входного буфера адаптера в регистр передатчика, освобождение входного буфера и восстановление единичного уровня сигнала TxRDY. После чтения SW на интервале неготовности строб записи не вырабатывается. После появления готовности повторятся уже описанные действия.

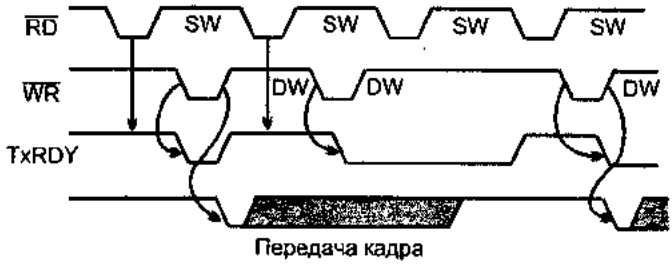


Рис. 6.19. Временные диаграммы программного условного обмена с помощью программируемого связного адаптера

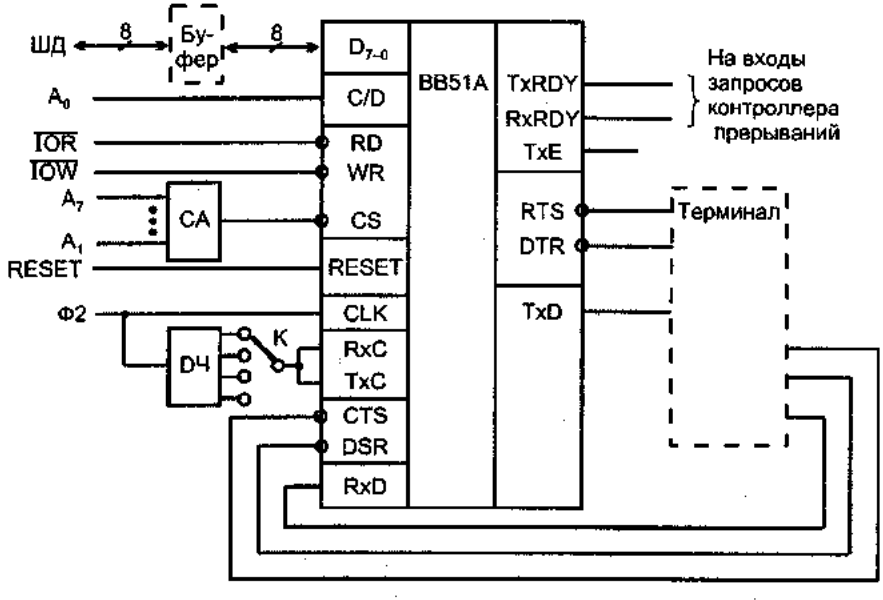


Рис. 6.20. Схема подключения программируемого связного адаптера к микропроцессору и терминалу

Пример подключения ПСА к МП и терминалу

Шина данных может подключаться к выводам адаптера через буфер или непосредственно в зависимости от нагрузочных условий. Селектор адреса CA (рис. 6.20) выдает на выходе низкий логический уровень, разрешающий работу адаптера, в ответ на одну-единственную комбинацию входных сигналов A_{7-1}. На вход CLK поданы синхроимпульсы Ф2 от МП, а частоты передачи и приема (в данном случае равные) получены из частоты Ф2 с помощью делителя частоты ДЧ. Как требуется условиями работоспособности адаптера, коэффициент деления должен быть ≥ 30. Делитель частоты имеет четыре выхода с разными частотами. С помощью ключа K можно изменять скорость передачи/приема данных. Остальные соединения понятны без дополнительных пояснений.

Последовательные порты строятся и на основе адаптеров UART, например, типа 16550 и 16550A фирмы Texas Instruments. Эти адаптеры во многом подобны адаптеру BB51A, но имеют 16-символьные буферы FIFO, предназначенные для приема и передачи данных. Конструктивно они могут входить в одну БИС с другими схемами.

Модули UART

В устройствах и системах, не требующих такой большой функциональной гибкости, которой обладает USART типа 51A или подобные ему, используют более простые модули последовательных интерфейсов. Самыми распространенными являются модули, называемые UART, SPI и I^2C.

Как следует из названия, модули UART работают в асинхронных режимах, но на самом деле многие из них способны реализовать и простые режимы синхронных передач. В режиме асинхронных передач модули UART поддерживают протоколы обмена RS-232C, RS-422, RS-485. Модули аналогичного типа, используемые фирмой Motorola, называются SCI (Serial Communication Interface).

Структура канала передатчика типичного UART приведена на рис. 6.21, *а*. Основу канала составляют два регистра — сдвигающий и буферный. Буферный регистр получает параллельные данные и загружает их в сдвигающий регистр для преобразования в последовательный код (с добавлением служебных битов обрамления информационных разрядов). Когда буферный регистр пуст, генерируется запрос прерывания для записи в буферный регистр нового слова (если это разрешено соответствующим битом управляющего слова), так что во время вывода слова сдвигающим регистром новое слово может уже ожидать своей очереди в буферном регистре. При отсутствии перерывов в загрузке буферного регистра асинхронные посылки передаются одна за другой, при этом старт-бит следующего кадра следует непосредственно за стоп-битом предыдущего.

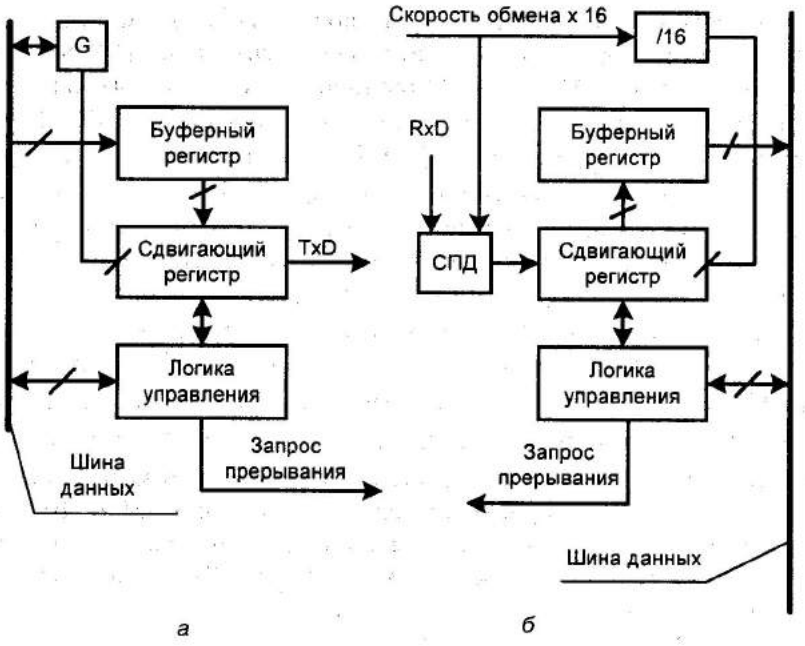


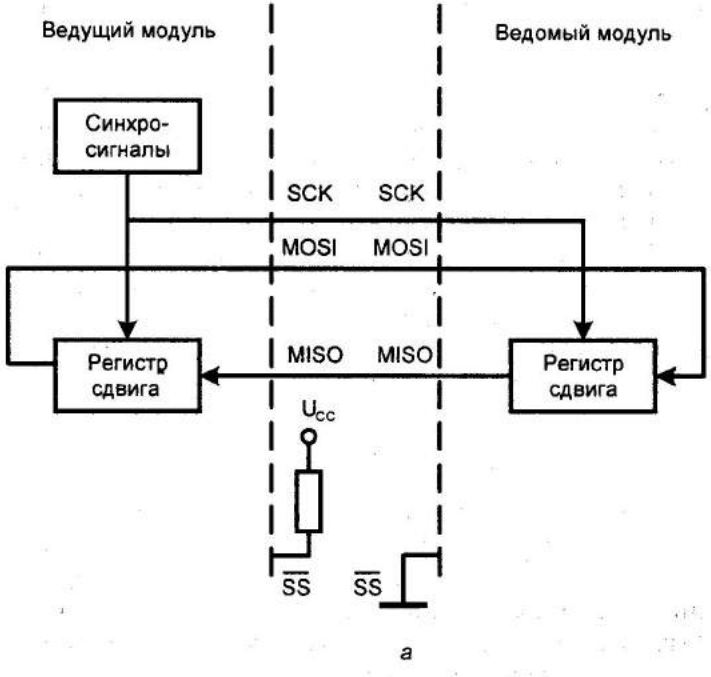
Рис. 6.21. Структуры передающего (*а*) и приемного (*б*) каналов блока UART

Схема приемного канала (рис. 6.21, *б*) несколько сложнее. В этом канале необходимо обеспечить считывание данных в середине битового интервала, причем часто в этом интервале берутся три отсчета и результат определяется мажоритарным голосованием. Для получения меток времени для отсчетов в приемном канале используются импульсы повышенной частоты, которая после деления на 16 должна соответствовать скорости приема данных в сдвигающий регистр. Кроме того, в канале имеется схема проверки правильности принятого слова по четности/нечетности его веса. После приема слова или при возникновении ошибки формируется запрос на прерывание для считывания слова из буферного регистра.

Интерфейс SPI

SPI (Serial Peripheral Interface) — синхронный интерфейс обмена последовательными данными. По протяженности связей его можно отнести к внутриплатным, обычно он служит для обмена данными между микропроцессором и периферийными устройствами или для связи между процессорами в многопроцессорных системах. Среди подключенных к линиям интерфейса модулей SPI выделяется ведущий, остальные (один или несколько) являются ведомыми. Такова основная конфигурация, принципиально возможно иметь

не только несколько ведомых, но и несколько ведущих модулей. В тракте обмена используются четыре линии: две сигнальные (MOSI — Master Output — Slave Input и MISO — Master Input — Slave Output), одна для синхросигналов (SCK) и одна для выбора активного ведомого модуля ($\overline{SS}$, Slave Select). Частота синхронизации до 5 МГц. Обязательными являются первые три линии, линия выбора активного ведомого модуля $\overline{SS}$ может отсутствовать, как, например, в приводимой далее схеме с одним ведущим и одним ведомым модулями. В схемах с несколькими ведомыми модулями ведомый может реагировать на сигналы интерфейса и генерировать данные на линии MISO только при низком уровне сигнала $\overline{SS}$. При высоком уровне этого сигнала выход MISO переходит в третье состояние. Характер использования сигнала $\overline{SS}$ ведомым модулем неоднозначен — возможно начинать операцию обмена (транзакцию) по отрицательному фронту $\overline{SS}$ и заканчивать по положительному, можно также задавать постоянный низкий уровень $\overline{SS}$, а транзакцию начинать по фронту сигнала синхронизации SCK. Второй вариант предпочтителен для простых схем с одним ведущим и одним ведомым модулями. Процесс обмена между двумя модулями SPI в такой схеме иллюстрируется на рис. 6.22. *а*.



а

Рис. 6.22. Схема обмена для интерфейса SPI (*а*)

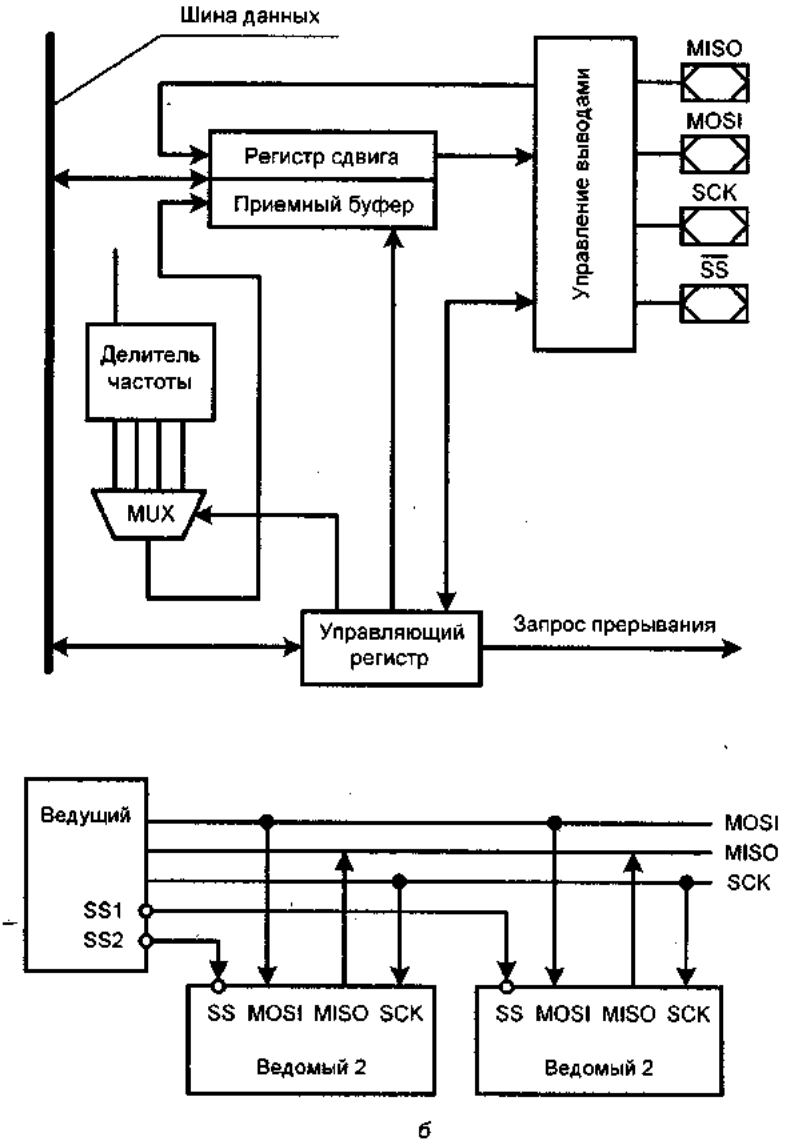


Рис. 6.22. Стандартная параллельная система для интерфейса SPI (*б*)

В схеме, изображенной на рис. 6.22, *а*, роли модулей (ведущий или ведомый) предполагаются неизменными, и линия $\overline{SS}$ заменена подачей на входы ведущего и ведомого модулей соответственно высокого и низкого уровней напряжения. В более общем случае роль каждого модуля задается

соответствующим битом регистра управления, а выводы $\overline{SS}$ ведомых модулей получают сигналы активизации от ведущего модуля.

Битами регистра управления ведущего модуля могут быть заданы скорость передачи (4 программируемых значения) и порядок передачи битов (старшим или младшим вперед). В регистре управления имеются также бит разрешения работы SPI (устанавливается до выполнения обмена), бит разрешения прерываний от SPI, биты, определяющие полярность синхросигналов и их активный фронт (передний или задний). В регистре состояния (на рисунке не показан) используются два бита: для установки флагов запроса прерывания и конфликта записи (при попытке записи во время передачи байта).

В ходе транзакции ведущий модуль на линии MOSI выставляет код операции и адресную информацию, затем идет передача данных. Запись байта в регистр данных инициирует передачу, запуская генератор тактовых импульсов и начиная поразрядную выдачу данных на вывод MOSI. При чтении данные идут от ведомого по линии MISO. Выдача байта останавливает генератор и устанавливает флаг запроса прерывания от SPI. Если прерывания разрешены, то генерируется запрос.

Соединенные в кольцо сдвигающие регистры обменивающихся модулей при поступлении серии сдвигающих импульсов обмениваются байтами — байт передатчика переходит в приемник и одновременно байт приемника переходит в передатчик, так что операции передачи и приема данных оказываются совмещенными.

В интерфейсах SPI с несколькими ведомыми модулями могут быть организованы разные системы обмена (стандартная последовательная, стандартная параллельная, со сравнением адресов). На рис. 6.22, *б* приведен пример стандартной параллельной системы, в которой для данных и синхросигнала сохраняется магистральная структура, а для сигналов выбора $\overline{SS}$ применены отдельные линии для каждого из ведомых модулей. В такой системе ведущий модуль должен раздельно формировать сигналы $\overline{SS}$ для каждого из ведомых. В другом варианте групповой схемы — схеме со сравнением адресов — активизация модулей сигналом $\overline{SS}$ не используется, ведомые модули все время прослушивают линию в поиске специальной адресной посылки. Опознав свой адрес, ведомое устройство воспринимает дальнейшую служебную информацию, после чего происходит обмен данными до завершения текущей транзакции. Этот вариант схемы требует от ведомых модулей более сложной логики поведения.

Интерфейс I^2C

Интерфейс I^2C (Inter Integrated Circuits) широко применяется в МПС для синхронной последовательной связи устройств. Его достоинство — возмож-

ность передачи данных с использованием всего двух линий, что обеспечивается передачей адреса устройства в составе посылки, т. е. аппаратное упрощение приобретается за счет снижения скорости обмена. По одной линии передается информация, по другой — синхросигналы. В отсутствие сигналов обе линии через резистор pull-up подключаются к высокому напряжению питания. Организация канала последовательной связи по интерфейсу I²C показана на рис. 6.23.

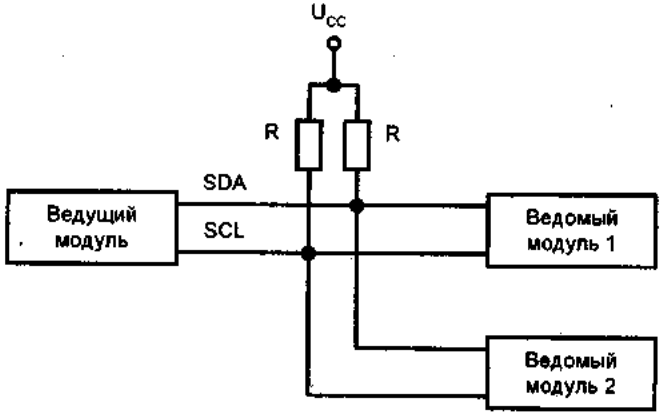


Рис. 6.23. Организация последовательного канала с интерфейсом I²C

SDA — двунаправленная линия передачи данных, SCL — линия, на которую ведущий модуль выдает синхросигналы, стробирующие передаваемые данные. Передача начинается с нулевого стартового бита, появляющегося при единичном значении синхроимпульса и создающего отрицательный перепад напряжения на линии SDA. После стартового бита ведущий модуль посылает байт, содержащий адрес ведомого модуля и разряд для сигнала управления R/W, задающего направление передачи (чтение/запись). Возможен и иной формат первоначального слова, в котором предусмотрена передача 10 бит. Все ведомые модули сравнивают посланный ведущим модулем адрес со своими адресами, и модуль, нашедший свой адрес, посылает в линию SDA сигнал подтверждения — нулевой уровень напряжения во время девятого импульса синхронизации. Наличие сигнала подтверждения является условием выполнения операций обмена, и такой сигнал посылается ведомым модулем в конце каждого пересылаемого байта. Отсутствие сигнала подтверждения заставляет ведущий модуль сформировать сигнал STOP в виде положительного перепада напряжения на линии SDA. Такой же сигнал формируется в конце обмена.

Реализуются такие типы обмена: от ведущего передатчика к ведомому приемнику и от ведомого передатчика к ведущему приемнику. В первом случае

после начального адресующе-управляющего байта передаются несколько байтов данных и ведомый модуль возвращает сигнал подтверждения после каждого байта. Во втором случае после передачи начального адресующе-управляющего байта ведомый возвращает бит подтверждения, после чего передает ведущему несколько байтов данных, а бит подтверждения посылается ведущим модулем в конце каждого байта (кроме последнего, в конце которого формируется сигнал конца передачи).

§ 6.5. Схемы обслуживания прерываний. Программируемые контроллеры прерываний

При работе микропроцессорной системы в ней или во внешней среде происходят события, требующие немедленной реакции, что обеспечивается прерыванием выполняемых программ и переходом к обслуживанию запросов прерывания *(см. § 5.3)*. Аппаратно прерывания обслуживаются специализированными ИС, простейшими из которых являются блоки приоритетного прерывания (Intel 8214, К589ИК14 и др.), решающие несложные задачи обработки нескольких векторных прерываний при фиксированных приоритетах запросов. Эффективную обработку запросов прерываний с широким набором режимов обеспечивают программируемые контроллеры прерываний. Для простых задач в микропроцессорной технике применяют *прерывания с программным или аппаратным опросом внешних устройств (поллингом)*.

Прерывания с опросом внешних устройств (поллингом) реализуются поочередной проверкой их на наличие запроса. Приоритеты ВУ фиксированы. Место ВУ в последовательности проверок определяет его приоритет. Опрос начинается с ВУ, имеющего старший приоритет.

При программном опросе линии запросов прерываний от всех источников объединяются по ИЛИ и наличие хотя бы одного запроса порождает общий запрос INT для процессора, инициируя тем самым последовательный опрос источников. Опрос проводится программой путем перебора адресов. При обнаружении ВУ, выставившего запрос, процессор читает его вектор прерывания, после чего выполняется подпрограмма обслуживания. Реализация программного опроса проста, но на него затрачивается много времени и приоритеты ВУ жестко фиксированы.

Аппаратный опрос источников прерываний

При аппаратном поллинге опрос последовательным перебором адресов заменяется процессом распространения сигнала опроса по так называемой дейзи-цепочке (рис. 6.24).

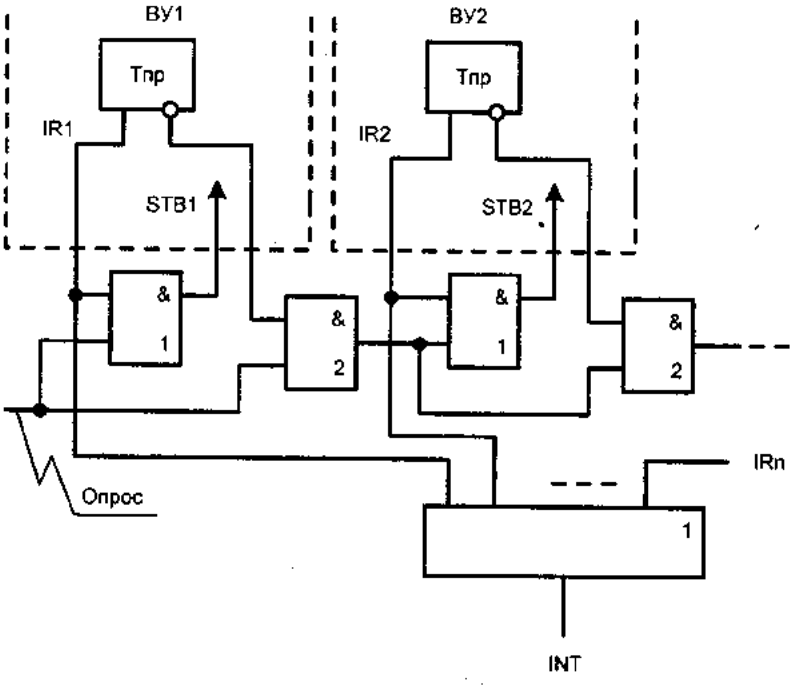


Рис. 6.24. Схема аппаратного опроса источников прерываний

В дейзи-цепочке каждому ВУ соответствуют два вентиля. Наличие или отсутствие запроса отображается состоянием триггера прерывания Тпр данного ВУ. Все линии запросов IR1—IRn объединяются по ИЛИ, так что наличие хотя бы одного запроса порождает для процессора сигнал INT. Если процессор подтверждает запрос, то он выдает сигнал на линию опроса. Сигнал опроса начинает свое распространение со старшего ВУ1. Если, например, это ВУ имеет запрос, то на линии строба STB1 появится сигнал выдачи процессору вектора прерывания от данного ВУ, а дальнейшее распространение сигнала опроса по цепочке будет заблокировано вентилем 2, на который поступает инверсное значение запроса, т. е. нулевой сигнал. Если же запроса у старшего ВУ нет, сигнал опроса пройдет на следующее ВУ и т. д., анализируя состояния ВУ. Первое же ВУ, имеющее запрос, получит сигнал строба STB и прекратит распространение сигнала опроса к следующим младшим устройствам.

Программируемые контроллеры прерываний

Более сложные задачи решаются программируемыми контроллерами прерываний (ПКП), в частности, ИС Intel 8259A, рассмотренной далее, и ее оте-

чественными аналогами К1810ВН59 и др. Эти контроллеры реализуют прерывания с обработкой до восьми запросов. С помощью нескольких ПКП легко организуется обработка до 64 запросов. Задача ПКП — выдать процессору команду перехода к подпрограмме обслуживания признанного запроса. При этом могут выполняться различные виды прерываний. Возможен и режим поллинга — обслуживания прерываний по результатам опроса источников запросов, начиная со старшего по приоритету. Обнаружение запроса ведет к его обслуживанию с переходом на соответствующую подпрограмму.

Вложение прерывания с фиксированными приоритетами входов

Из имеющихся у ПКП восьми входов запросов прерывания IR_7—IR_0 (IR от англ. *Interrupt Request*) высший приоритет имеет вход IR_0, низший — IR_7. *Вложенность* — возможность прерывания подпрограммы обслуживания запроса другой подпрограммой с более высоким приоритетом, которая, в свою очередь, также может быть прервана более приоритетной подпрограммой, и т. д. Вложенность прерываний обеспечивается введением команды EI (Enable Interrupt) в подпрограммы их обслуживания. Прерывания с фиксированными приоритетами реализуются просто, но *запросы неравноправны* и при интенсивном поступлении запросов с высокими приоритетами запросы с низкими приоритетами могут вообще не получить обслуживания, т. е. возможно их "грубое оттеснение" более приоритетными.

Прерывания с круговыми (циклическими) приоритетами

В этом случае у каждого входа тоже есть свой приоритет, но после обслуживания он изменяется так, что обслуженный вход получает низший приоритет, который по мере обслуживания других запросов начинает возрастать, вновь доходя до высшего. Такая дисциплина обслуживания характерна для ситуации с равноправными источниками, *не имеющими преимуществ друг перед другом*. Запрашивающее обслуживания устройство будет ждать в худшем случае до того, как семь других источников будут обслужены по одному разу. Работу с круговым приоритетом можно иллюстрировать примером (рис. 6.25) с регистром запросов, содержащим вначале шестой и четвертый запросы, где наивысший приоритет имеет четвертый запрос, который и будет обслужен. Строка "Уровни приоритета" рассматривается как кольцо, в котором позиции 7 и 0 являются соседними. После обслуживания приоритетность входов изменяется как бы вращением кольца, причем номер 7 с низшим приоритетом становится на четвертую позицию только что обслуженного запроса. Позицию низшего приоритета называют *дном приоритетного кольца*. В этих терминах работу с круговым (циклическим) приоритетом

можно выразить так: *после обслуживания дно приоритетного кольца устанавливается на позицию обслуженного запроса.*

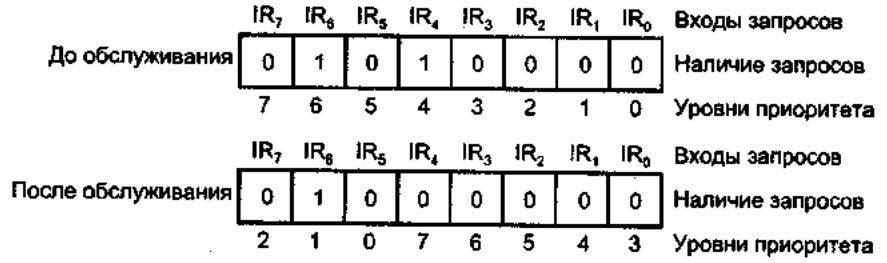


Рис. 6.25. Пример обслуживания запросов прерывания с круговым приоритетом

Кроме рассмотренного, имеется также режим адресуемого циклического приоритета, при котором дно кольца после обслуживания ставится в положение, определяемое программно, а не устанавливается автоматически на позицию обслуженного запроса.

Маскирование запросов

Контроллерами реализуется *маскирование запросов*, когда запрещается их восприятие с помощью соответствующих битов регистра маски. При этом могут быть реализованы разные ситуации. Обычная ситуация состоит в том, что маскирование какого-либо запроса ведет и к маскированию других запросов с меньшими приоритетами. *Специальное маскирование* заключается в блокировании восприятия только одного входа запросов при отсутствии маскирования младших по приоритету. После снятия маски обслуживание запросов становится возможным. В связи с разными дисциплинами приоритетности и видами маскирования процесс прерывания может завершаться в одном из нескольких вариантов.

Включение контроллера прерываний в систему

Включение контроллера прерываний в систему показано на рис. 6.26. Контроллер принимает запросы от внешних устройств, определяет, какой из незамаскированных запросов имеет наивысший приоритет, сравнивает его с приоритетом текущей программы и при соответствующих условиях выдает запрос прерывания INT для МП. После подтверждения запроса МП должен получить от контроллера информацию, которая укажет на подпрограмму, соответствующую данному ВУ, т. е. вектор прерывания.

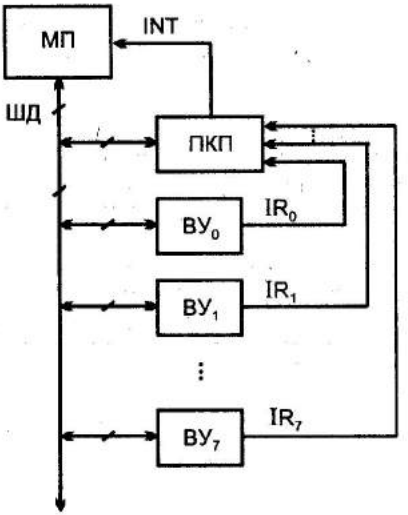


Рис. 6.26. Схема включения контроллера прерываний в микропроцессорную систему

Структура ПКП

Структура ПКП Intel 8259A представлена на рис. 6.27. В английской терминологии ПКП называют PIC, т. е. Programmable Interrupt Controller.

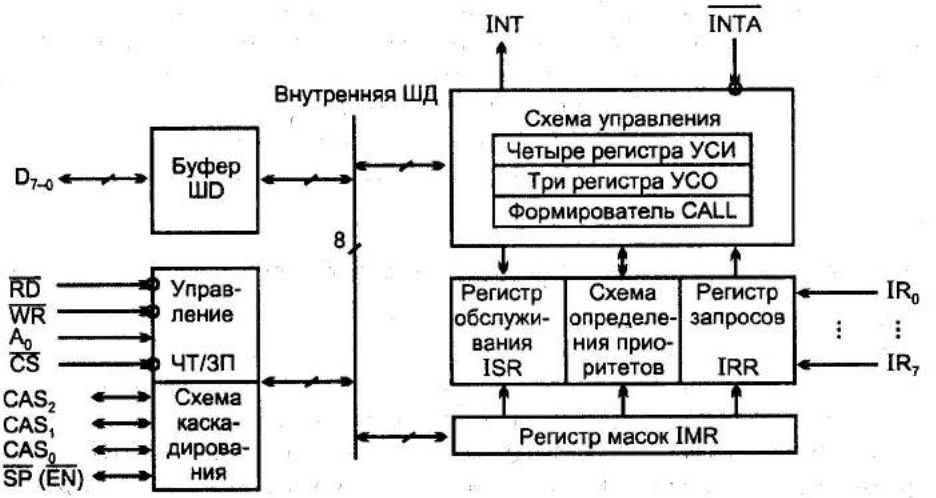


Рис. 6.27. Структура программируемого контроллера прерываний

Запросы прерываний от ВУ поступают на регистр запросов IRR (Interrupt Request Register), сохраняющий запросы до их принятия на обслуживание. Биты регистра IRR сопоставляются с битами регистра масок IMR (Interrupt Mask Register). Биты регистра масок действуют также на работу схемы определения приоритетов и регистр обслуживания ISR (Interrupt Servicing Register), так что маскирование осуществимо не только на стадии приема запросов, но и на более поздних стадиях их обработки.

Если приоритет запроса выше текущего приоритета, то при вложенных прерываниях формируется сигнал INT для процессора. При поступлении от процессора сигнала подтверждения прерывания $\overline{\text{INTA}}$ принятый запрос переходит в регистр обслуживания ISR и сбрасывается в регистре запросов IRR. Установка бита ISR запрещает прерывания от всех других запросов с меньшими приоритетами. Подпрограмма обслуживания прерывания завершается сбросом бита регистра ISR.

Буфер ШД восьмиразрядный, двунаправленный, с третьим состоянием. При программировании контроллера через буфер передаются управляющие слова и считывается состояние регистров. При обслуживании прерывания по сигналу $\overline{\text{INTA}}$ через буфер ШД в шину данных системы выдается трехбайтная команда вызова подпрограммы.

Смысл сигналов $\overline{\text{RD}}$, $\overline{\text{WR}}$ и $\overline{\text{CS}}$ ясен (совпадает со смыслом этих сигналов в описанных выше устройствах). Остальные сигналы имеют следующее назначение.

- Сигнал $\overline{\text{INTA}}$ поступает от процессора в виде трех последовательных импульсов для выдачи контроллером кода команды CALL, затем младшего и старшего байтов адреса начала подпрограммы. Первый импульс $\overline{\text{INTA}}$ сбрасывает запрос в соответствующем бите IRR.
- IR_0...IR_7 — входы запросов прерывания;
- A_0 — младший разряд адреса, вместе с другими признаками показывает, к какому регистру управляющих слов (УСИ или УСО) обращается процессор;
- CAS_{2-0} связаны с работой контроллера в групповой схеме, образуют выходную шину для ведущего контроллера и входную для ведомых;
- $\overline{\text{SP}}$ ($\overline{\text{EN}}$) — двухфункциональный сигнал, как $\overline{\text{SP}}$ он определяет, является ли контроллер ведущим или ведомым в групповой схеме, как $\overline{\text{EN}}$ используется в так называемом буферизованном режиме для разрешения/запрещения выходов различных модулей на шину системы, т. е. для управления выходными буферами участников обмена.

Программирование контроллера

Перед работой контроллер программируют загрузкой управляющих слов инициализации (УСИ) и операций (УСО).

УСИ, число которых в зависимости от назначаемого режима работы составляет 2—4, записываются стробами $\overline{WR}$ и приводят контроллер в исходное состояние. Три или четыре старших разряда УСИ1 задают часть младшего байта начального адреса той зоны адресного пространства, в которой нужно разместить векторы прерываний. Интервал между начальными адресами отдельных подпрограмм принимается равным 4 или 8, поэтому размер этой зоны составит 32 или 64 байта, а таких зон в адресном пространстве размером 64К будет 2^{11} или 2^{10}. Следовательно, задать одно из возможных положений зоны можно с помощью 11 разрядов адреса (при адресном интервале 4) или 10 (при адресном интервале 8), для чего контроллеру нужно передать разряды адреса A_{15}—A_5 в первом случае и A_{15}—A_6 во втором. Таким образом, младший байт адреса подпрограммы (вектора) будет иметь вид, показанный в табл. 6.4 для интервала 4 и в табл. 6.5 для интервала 8.

Таблица 6.4

IR	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
7	A_7	A_6	A_5	1	1	1	0	0
6	A_7	A_6	A_5	1	1	0	0	0
5	A_7	A_6	A_5	1	0	1	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	A_7	A_6	A_5	0	0	0	0	0

Таблица 6.5

IR	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
7	A_7	A_6	1	1	1	0	0	0
6	A_7	A_6	1	1	0	0	0	0
5	A_7	A_6	1	0	1	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	A_7	A_6	0	0	0	0	0	0

При интервале 4 три первых разряда программируются и участвуют в задании области размещения подпрограмм обслуживания в адресном пространстве системы, а пять младших (т. е. адреса восьми четырехбайтных зон внутри 32-байтной области с начальным адресом A_{15}—A_5) фиксированы и автоматически вводятся самим контроллером. "Скачки" через 4 или 8 байтов для получения начальных адресов отдельных подпрограмм обеспечиваются нулевыми значениями двух или трех младших разрядов адреса. При интервале 8 программируются всего два старших разряда младшего полуадреса.

Формат управляющего слова УСИ1 показан на рис. 6.28 (для примера взят интервал 4). Разряды 7...5 содержат программируемую часть младшего байта адреса подпрограммы, единичное значение четвертого разряда вместе с условием $A_0 = 0$ служит признаком слова УСИ1, разряд 3 задает способ восприятия входных запросов IR_7... IR_0 по фронтам или уровням (LTIM — Level Triggered/Edge Triggered Interrupt Mode), четвертый разряд AI задает адресный интервал (4 или 8), разряд SNGL определяет, является ли контроллер единственным или он работает в групповой схеме, а последний младший разряд отвечает на вопрос, понадобится или нет при программировании загрузка управляющего слова УСИ4, т. е. вводится или нет буферизованный режим.

Рис. 6.28. Форматы управляющих слов инициализации программируемого контроллера прерываний

УСИ2 содержит старший байт начального адреса зоны подпрограмм обслуживания прерываний. Если несколько контроллеров работают в групповой структуре, то загружается и УСИ3. Ведущему контроллеру это слово сообщает, какие его входы подключены к ведомым, а каждому из ведомых — к какому входу ведущего подключен его выход INT. Таким образом, УСИ3 отражает физическую схему соединения контроллеров. Для буферизованного режима (при единичном состоянии младшего разряда слова УСИ1, т. е. IC4 = 1) вводится УСИ4 (этот режим здесь рассматривать не будем).

Последовательность инициализации контроллера показана на рис. 6.29. В результате инициализации сбрасываются управляемые фронтами триггеры приема запросов и регистры контроллера. Входу IR_7 присваивается низший приоритет, снимается специальное маскирование, чтение состояния устанавливается на IRR, если IC4 = 0, то все функции, задаваемые УСИ4, обнуляются.

После инициализации контроллер может работать в базовом режиме. Для выбора других режимов загружаются управляющие слова операций УСО. Форматы УСО приведены на рис. 6.30.

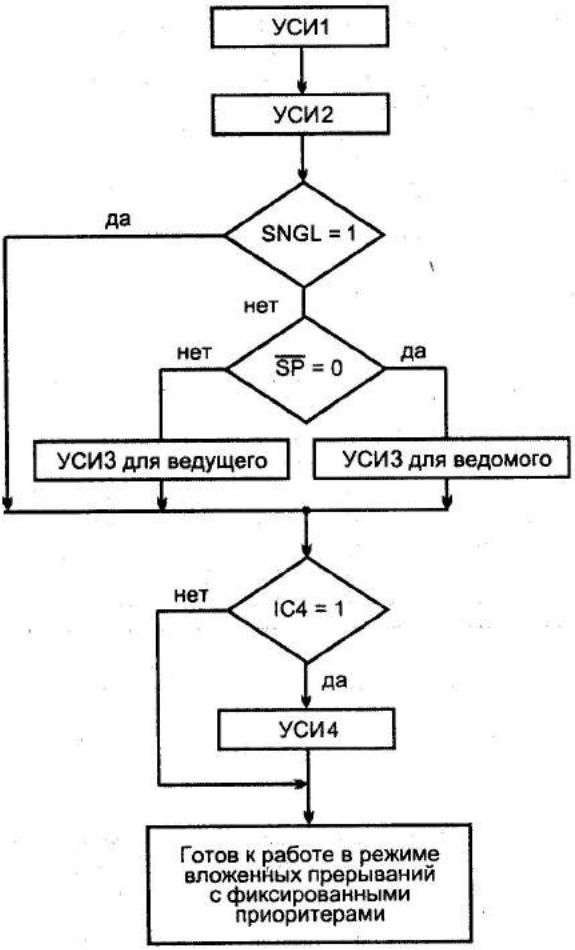


Рис. 6.29. Последовательность инициализации программируемого контроллера прерываний

С помощью УСО1 можно в любое время программно установить или сбросить отдельные биты регистра масок. Каждый вход может быть замаскирован словом, содержащим 1 в соответствующем разряде. Регистр масок воздействует и на IRR, и на ISR.

Слово УСО2 может задать пять вариантов завершения прерывания. Каждая подпрограмма обслуживания прерывания должна сообщать контроллеру о своем завершении, передавая ему одно из УСО2, в котором задан характер конца прерывания из числа следующих:

- КП — конец прерываний, т. е. неадресуемый конец прерываний, заключающийся в сбросе бита ISR с максимальным приоритетом, который был обслужен;

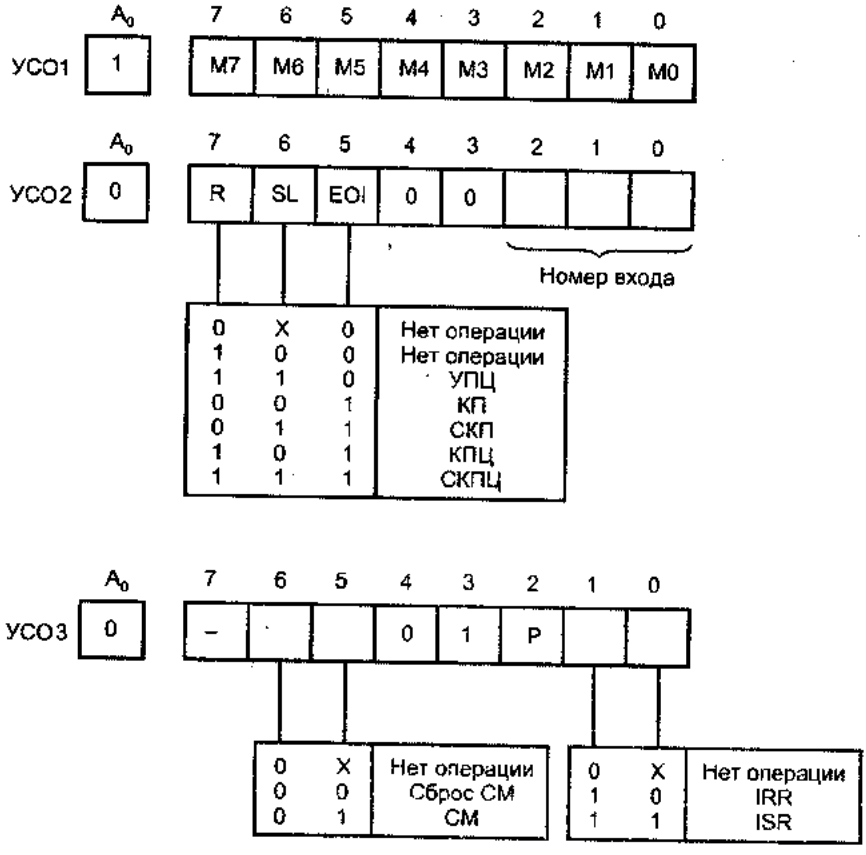


Рис. 6.30. Форматы управляющих слов операций программируемого контроллера прерываний

- СКП — специальный (адресуемый) конец прерываний, т. е. сброс бита, определяемого полем из трех разрядов слова УСО2;
- КПЦ — конец прерываний с циклическим сдвигом приоритета, т. е. сброс бита ISR, соответствующего последнему обслуженному запросу, и перевод дна приоритетного кольца на позицию этого бита;
- СКПЦ — специальный конец прерываний с циклическим сдвигом приоритетов, т. е. присвоение позиции дна тому входу, который указан полем слова УСО2 с одновременным выполнением обычного конца прерываний;
- УПЦ — установка приоритетов, т. е. присвоение позиции дна указанному в поле УС биту без выполнения операции обычного конца прерываний (без изменения регистра ISR).

Команды типа УСО3 (признак $A_0 = 0$, $D_3 = 1$, $D_4 = 0$) применяются в режиме чтения и при установке/снятии режима специального маскирования.

Чтение состояния заключается в чтении регистров контроллера (IMR, IRR и ISR) или кода старшего из поступивших запросов. Чтение регистра масок по условиям $\overline{RD} = 0$ и $A_0 = 1$ не требует предварительной загрузки УСО3. Остальные регистры считываются после загрузки соответствующего УСО3 по команде IN port или при подаче низкого уровня напряжения на вывод $\overline{RD}$.

В режиме опроса (поллинга) программа сама запрашивает информацию о запросах прерывания, обращаясь к источникам запросов поочередно в порядке понижения приоритета. Обнаружив запрос, программа осуществляет переход на подпрограмму обслуживания прерывания, которая и выполняет обмен данными. Режим опроса инициируется выдачей в контроллер УСО3 с единичным значением бита P (Polling).

Каскадное включение контроллеров

Каскадное включение контроллеров расширяет число обрабатываемых запросов. Возможно подключение к ведущему контроллеру (master) не более восьми ведомых (slave). На рис. 6.31 показано подключение одного ведомого контроллера, дающее схему с 15 входами (остальные семь контроллеров могут быть подключены аналогичным способом). По данной методике можно получить схему с 64 входами.

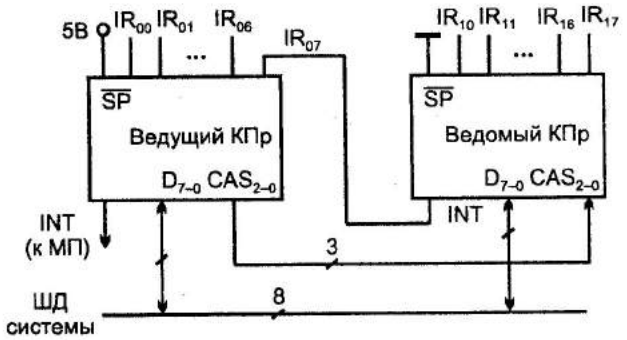


Рис. 6.31. Схема каскадного включения контроллеров прерываний

Функции ведущего и ведомого контроллеров определяются сигналами на входе $\overline{SP}$. Предварительно каждый ведомый контроллер получает номер, соответствующий номеру входа ведущего, к которому он подключен (это осуществляется загрузкой соответствующего УСИ3). На запросы по своим входам ведущий контроллер реагирует обычным способом, формируя команду CALL, как уже было описано. Для запросов от входов ведомого по

первому импульсу $\overline{INTA}$, поступающему от МП, ведущий выдает на ШД команду CALL, а на шине CAS номер ведомого. По сигналам $\overline{INTA2}$ и $\overline{INTA3}$ адресованный ведомый контроллер выдает на ШД код адреса подпрограммы обслуживания.

По сигналу $\overline{INTA}$ устанавливаются соответствующие разряды ISR обоих контроллеров. Поэтому подпрограмма обслуживания должна завершаться выдачей двух управляющих слов "Конец прерывания" для ведущего и ведомого контроллеров.

Подключение ПКП к шине микропроцессорной системы не требует специальных пояснений, поскольку выше были определены все цепи, с которыми связаны выводы контроллера.

§ 6.6. Контроллеры прямого доступа к памяти

Прямой доступ к памяти (ПДП) — создание прямого тракта передач данных от внешних устройств к памяти или от памяти к внешним устройствам.

В английской терминологии это DMA — Direct Memory Access. При обычном обмене (т. е. без ПДП) для передач между ВУ и памятью требуется вначале принять данные от источника в процессор, а затем выдать их из процессора приемнику, т. е. передачи реализуются за два цикла. При ПДП данные не проходят через процессор, и передача слова по тракту "Память — ВУ" производится за один цикл. ПДП особенно удобен при передачах блоков данных в высоком темпе, например при обмене данными между внешней памятью и ОЗУ. В режиме ПДП процессор отключается от системных шин и передает управление ими *контроллеру прямого доступа к памяти* (КПДП). Для ПДП выпускаются БИС, способные, благодаря программированию, обслуживать ПДП с учетом требований различных систем.

Взаимодействие блоков микропроцессорной системы при ПДП показано на рис. 6.32. Микропроцессор программирует КПДП, настраивая его на определенный режим работы, и может читать состояние контроллера. Соответствующие связи показаны штриховыми линиями. При ПДП микропроцессор отключен, а контроллер вырабатывает сигналы управления обменом для ВУ и ОЗУ. Тракт передачи данных непосредственно связывает ВУ с ОЗУ.

Возможны *два вида ПДП* — с *блочными* или *одиночными* передачами. В первом работа процессора останавливается на время передачи блока данных, во втором передачи слов в режиме ПДП перемежаются с выполнением программы, и для ПДП выделяются отдельные такты машинных циклов, в которых процессор не использует системные шины. Каждый командный цикл процессора начинается с машинного цикла M1 — выборки команды. В этом

машинном цикле есть такт декодирования принятой команды, в котором системные шины не используются. На это время их можно отдать для ПДП и передать одно слово. Производительность системы может возрасти благодаря тому, что ПДП будет для процессора "невидимым". Сам обмен с ПДП будет не быстрым, темп обмена нерегулярен, т. к. длительности циклов различных команд различны и, кроме того, ПДП может и замедлить выполнение программы, если цикл ПДП не уложится в интервал, соответствующий такту процессора.

Рис. 6.32. Схема взаимодействия блоков микропроцессорной системы при прямом доступе к памяти

При непрерывной передаче блока данных (в основном режиме работы ПДП) скорость обмена ограничивается длительностью циклов ЗУ, быстродействием самого контроллера и скоростью выдачи/приема данных внешним устройством.

В отличие от процессов прерывания при ПДП обмен выполняется без участия программы и не влияет на содержимое рабочих регистров МП. Поэтому при вхождении в режим ПДП не требуются затраты времени на передачу в стек содержимого рабочих регистров МП. ПДП предоставляется процессором по завершении текущего машинного цикла.

Структура и функции КПДП

Примером КПДП может служить БИС Intel 8237A и ее отечественный аналог К580ВТ57 (рис. 6.33).

Перечислим действия, выполняемые КПДП в режиме *блочных передач*.

1. Прием от процессора сведений об области памяти, отведенной для передаваемого блока (начальный адрес и размер блока).
2. Трансляция запроса ПДП, исходящего от ВУ, в запрос захвата шин для процессора с учетом маскирования и приоритетности запросов, поступающих на КПДП.

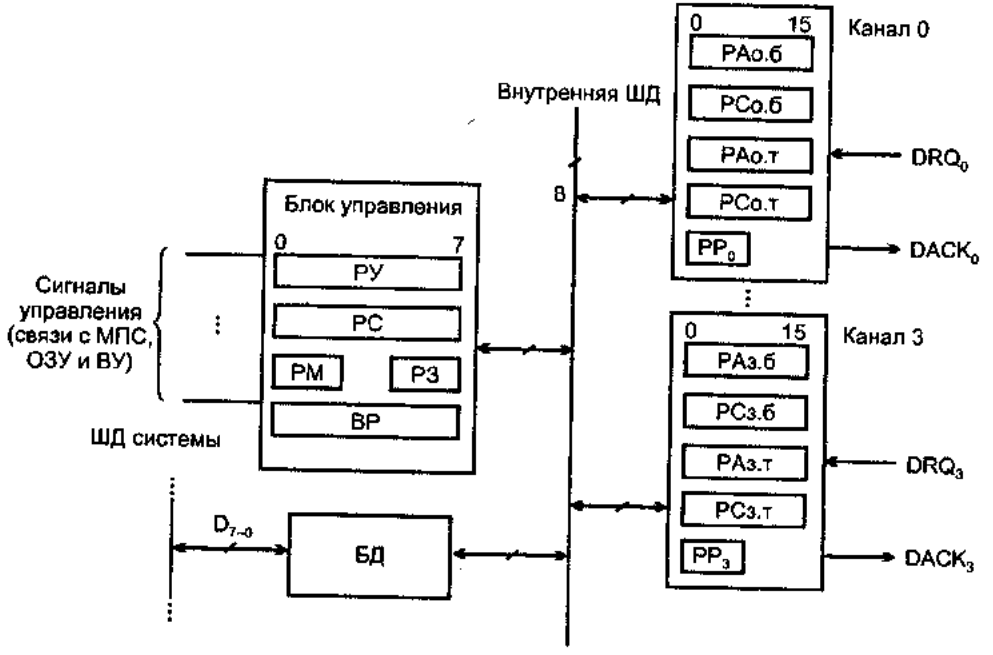


Рис. 6.33. Структура контроллера прямого доступа к памяти

3. Прием от процессора сигнала подтверждения ПДП, свидетельствующего о том, что процессор отключился от системных шин.
4. Генерация адресов и сигналов управления для ЗУ и ВУ.
5. Фиксация завершения ПДП.
6. Снятие запроса захвата шин с соответствующего входа процессора и возвращение управления основной программе.

Возможности КПДП позволяют организовать и обмен типа "память-память", т. е. решать задачу перемещения блока данных в адресном пространстве системы.

КПДП 8237А работает на частоте 3 МГц, его модификации 8237А-4 и 8237А-5 — на частотах 4 и 5 МГц соответственно. Контроллер имеет четыре независимых канала, возможно каскадирование схем до любого числа каналов.

В каждом из каналов контроллера размещено по пять регистров, а именно: два регистра адреса (базовый $РА_{i.б}$ и текущий $РА_{i.т}$, где i — номер канала), два регистра счета слов (базовый $РС_{i.б}$ и текущий $РС_{i.т}$) и регистр режима $РР_i$. Адресные регистры и регистры счета слов 16-разрядные, следовательно, начальный адрес блока данных может располагаться в любом месте адрес-

ного пространства емкостью 64K, а максимальный размер блока составляет 64 Кбайт.

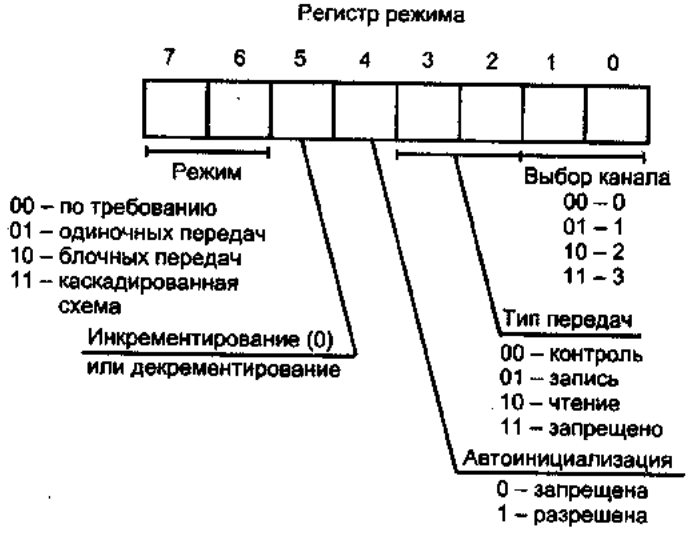
При программировании в оба адресных регистра загружается одно и то же значение адреса, а в оба регистра счета слов — одно и то же значение размера блока. При ПДП меняются состояния текущих регистров адреса и счета слов. Оба они *работают в режиме счетчиков* и при передаче очередного слова регистр адреса инкрементируется или декрементируется (в зависимости от программирования контроллера), а регистр счета слов декрементируется. Когда регистр-счетчик $PC_{i.T}$ дойдет до нулевого состояния (перейдет от состояния 0000H к состоянию FFFFH), выработается сигнал конца счета (т. е. в качестве начального значения в $PC_{i.T}$ следует загружать число, на единицу меньшее размера блока). Этим заканчивается режим блочного обмена.

Базовые регистры адреса и счета слов обслуживают *режим автоинициализации* канала. В них начальные адреса и размеры блоков сохраняются неизменными и, если в конце ПДП вновь загрузить текущие регистры теми же кодами, то можно вновь повторить вывод того же блока данных, что и в предыдущем ПДП. Такой режим нужен, например, при управлении дисплеем, который для поддержания на экране какого-либо изображения нуждается в повторении блока данных с частотой в несколько десятков герц.

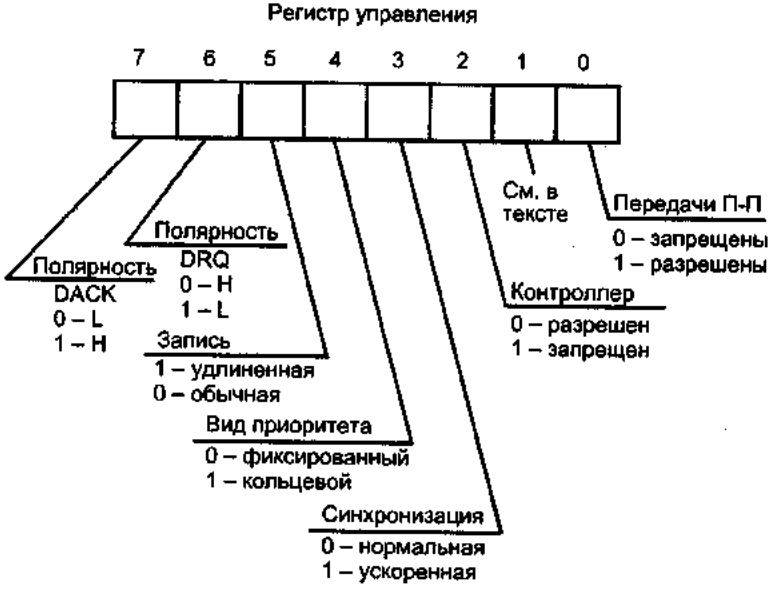
Регистр режима восьмиразрядный. Его формат показан на рис. 6.34, *а*. Два младших бита используются для выбора канала, т. е. загрузки слова в тот или иной канал, а шесть остальных определяют режимы, указанные на рисунке. В режиме "Контроль" нет передач, но сигналы доступа к данным формируются, что позволяет выполнять операции контроля информации. В режиме обмена по требованию передачи выполняются до выработки контроллером признака конца счета или поступления внешнего сигнала $\overline{EOP}$ (End of Process) или до перехода сигнала DRQ (DMA Request) в пассивное состояние, т. е. до "истощения" ВУ в смысле исчерпания его данных. При этом возобновление данных в ВУ позволяет продолжить ПДП с помощью изменения сигнала DRQ. В периоды между ПДП, когда позволено работать процессору, промежуточные значения адресов и счетчика слов хранятся в текущих регистрах.

Восьмиразрядный *регистр управления* РУ загружается процессором, сбрасывается сигналом RESET или командой Master Clear. Формат слова управления показан на рис. 6.34, *б*. Не расшифрованный на рисунке бит 1 имеет следующий смысл. При $D_0 = 0$, когда запрещены передачи "память-память" (П-П), состояние этого бита безразлично. При передачах П-П и $D_1 = 1$ запрещается, а при $D_1 = 0$ разрешается оставление адреса в канале 0.

Регистр состояния РС содержит информацию о текущем состоянии контроллера и может читаться процессором. Четыре младших бита этого регистра устанавливаются каждый раз при конце счета или появлении внешнего сигнала $\overline{EOP}$ в соответствующем канале и сбрасываются сигналом RESET при каждом чтении состояния. Четыре старших бита устанавливаются, если соответствующие каналы запрашивают обслуживание. Таким образом, РС позволяет определить, какие каналы закончили ПДП, а какие требуют его.



а



б

Рис. 6.34. Форматы регистров режима (*а*) и управления (*б*) контроллера прямого доступа к памяти

Четырехразрядный *регистр масок* РМ имеет биты, соответствующие четырем каналам. Установка бита запрещает действие входного запроса DRQ. Если канал не запрограммирован на автоинициализацию, то по окончании ПДП он вырабатывает сигнал $\overline{EOP}$, при этом устанавливается бит маски этого канала. Этот бит устанавливается или сбрасывается также программно. Весь регистр устанавливается сигналом RESET, что запрещает запросы до поступления команды сброса регистра Clear Mask Rg, разрешающей начать прием запросов.

Регистр запросов РЗ позволяет контроллеру реагировать на запросы ПДП, исходящие от программы. Каждый канал имеет свой бит в этом четырехразрядном регистре, биты немаскируемы, но подчиняются требованиям приоритетности. Биты устанавливаются и сбрасываются индивидуально программой или сбрасываются после генерации контроллером признака конца счета или внешним сигналом $\overline{EOP}$. Весь регистр одновременно сбрасывается сигналом RESET.

Временный регистр ВР используется при передачах типа "память-память" для временного хранения данных и всегда содержит последний байт, переданный в предыдущей операции, если не сброшен сигналом RESET.

Для выбора внутренних регистров контроллера используются четыре линии адреса A_{3-0}, но число регистров байтовой длины (эту единицу измерения нужно принять, т. к. 16-разрядные регистры могут загружаться только двумя передачами по 8 разрядов) превышает возможности адресации четырехразрядными кодами. Поэтому в каналах имеются триггеры счетного типа, переключающиеся при последовательных передачах байтов для различения их по признаку первый/последний и направления в соответствующую часть регистров. Эти триггеры должны быть сброшены до записи новых значений адреса и счета слов для правильного распознавания младших и старших байтов 16-разрядных слов. Для этого имеется команда Clear First/Last. Как и две упомянутые выше команды Master Clear и Clear Mask Rg, эта команда выполняется при программировании контроллера.

В контроллере применено мультиплексирование шин для передачи старшего байта адреса через шину данных во внешний регистр, где этот байт фиксируется стробом ADSTB (Address Strobe), после чего шина данных используется для других передач. В схеме (рис. 6.35) внешний регистр не показан (он не входит в состав ИС контроллера).

В работе контроллера можно выделить *две фазы* — простоя и активную. В фазе простоя контроллер находится, когда на его входах нет запросов ПДП. В этой фазе контроллер может быть запрограммирован процессором. Состояние программируемости продолжается и после начала действий ПДП, до момента, когда контроллер запросил захвата шин (сигнал HRQ — Hold Request), но еще не получил от процессора ответа (сигнал HLDA — Hold Acknowledge) о предоставлении ПДП.

При простое контроллер постоянно (в каждом такте) проверяет линии DRQ_i (не поступил ли запрос от ВУ) и $\overline{CS}$ (не обращается ли процессор к регистрам контроллера). В последнем случае линии адреса A_{3-0} выбирают регистры, а по стробам $\overline{IOR}$ и $\overline{IOW}$ производятся чтение или запись.

После поступления сигнала HLDA, когда процессор освободил шины системы, начинаются рабочие состояния контроллера.

Они проходят в активной фазе и, если требуется, вводятся такты ожидания с учетом сигнала готовности/неготовности READY участников обмена (входного сигнала контроллера). Для передач "ВУ → память" генерируются пары одновременных сигналов $\overline{IOR}$ и $\overline{MEMW}$ для передач "память → ВУ" — пары $\overline{IOW}$ и $\overline{MEMR}$.

Запрос незамаскированного канала порождает запрос захвата шин HRQ (Hold Request) для процессора, реализующего один из четырех вариантов ПДП, указанных седьмым и шестым битами регистра режима.

Подробная схема взаимодействия блоков в микропроцессорной системе с ПДП дана на рис. 6.35.

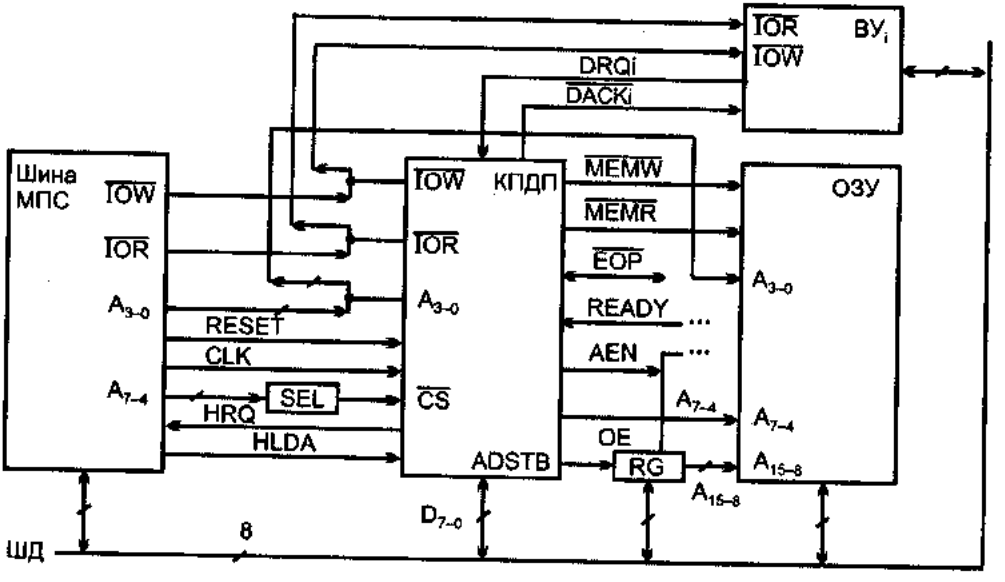


Рис. 6.35. Схема взаимодействия процессора, памяти и внешних устройств с контроллером прямого доступа к памяти

Выводы и сигналы контроллера

Выводы и сигналы контроллера имеют следующий смысл.

D_{7-0} — двунаправленные линии системной ШД с тремя состояниями.

RESET — сброс внутренних регистров контроллера и внешних выходных управляющих сигналов. Сброс регистра режима РР дезактивирует контроллер, он перестает реагировать на запросы ПДП до выполнения программы инициализации.

CLK — тактовый сигнал синхронизации (обычно сигнал Ф2 от микропроцессора).

$\overline{IOW}$ — в режиме программирования это входной сигнал, осуществляющий запись в тот или иной регистр контроллера, в режиме ПДП — выходной сигнал, стробирующий запись байта в ВУ.

$\overline{IOR}$ — в режиме программирования входной сигнал чтения того или иного регистра контроллера, в режиме ПДП — выходной, стробирующий введение данных от ВУ.

A_{3-0} — линии адреса, в режиме программирования входные, адресуют регистры контроллера, в режиме ПДП — выходные, дают 4 младших разряда адреса, формируемого контроллером.

$\overline{CS}$ — в режиме программирования разрешает работу контроллера, в режиме ПДП отключен. Вырабатывается дешифрацией адреса контроллера.

A_{7-4} линии адреса, формируемого контроллером в режиме ПДП, в других режимах переходят в третье состояние.

READY — готовность. Отсутствие этого сигнала переводит контроллер в состояние ожидания. Сигнал используется, если быстродействие основной памяти недостаточно для работы в синхронизме с ВУ, и тогда он удлиняет циклы обращения к памяти.

$\overline{MEMW}$ — сигналы записи и чтения памяти в режиме ПДП. В циклах чтения и записи.

$\overline{MEMR}$ — контроллер вырабатывает парные сигналы $\overline{IOW}$ и $\overline{MEMR}$ или $\overline{IOR}$ и $\overline{MEMW}$.

$\overline{HRQ}$ (Hold Request) — запрос захвата шин, выдается контроллером микропроцессору на его вход HOLD при наличии запроса от ВУ.

HLDA (Hold Acknowledge) — подтверждение захвата, поступает от МП и разрешает переход в режим ПДП.

AEN (Address Enable) — разрешение адреса, указывает на режим ПДП, может быть использован для блокировки шин адреса в других устройствах с целью запрета их ошибочной работы. Переводит адресные шины невыбранных устройств в третье состояние.

ADSTB (Address Strobe) — сигнал выдачи адреса, разрешающий запись старшего байта адреса, вырабатываемого контроллером, по ШД во внешний регистр-защелку, хранящий адрес до конца цикла.

DRQ_i ($i = 0...3$) (DMA Request) — запрос ПДП, формируемый ВУ. Для передачи одного байта должен быть снят внешним устройством по получении сигнала $DACK_i$ от контроллера. Для передачи массива данных должен удерживаться ВУ до получения от контроллера сигнала TC. Входы DRQ_i имеют фиксированные приоритеты (у DRQ_0 высший) или круговой приоритет.

$DACK_i$ (i = 0...3) — (DMA Acknowledge) — подтверждение (разрешение) ПДП. Информирует ВУ о предоставлении ему очередного цикла ПДП. Устанавливается и сбрасывается для передачи каждого байта данных. Вырабатывается согласно приоритетам входов DRQ_i.

$\overline{EOP}$ — (End of Process) — двунаправленный, информирует о завершении ПДП, генерируется в конце счета самим контроллером, может поступать извне. Появление внутреннего или внешнего $\overline{EOP}$ заставляет контроллер прекратить ПДП, сбросить запрос и, если разрешена автоинициализация, загрузить текущие регистры канала от базовых.

Возможность организации *передач типа "память-память"* позволяет простыми средствами перемещать блок данных из одной области адресного пространства в другую. Установка младшего бита регистра управления определяет использование каналов 0 и 1 для передач "память-память". Передачи начинаются программной установкой запроса DRQ в канале 0. После подтверждения ПДП сигналом HLDA контроллер читает данные из области памяти соответственно адресам в регистре текущего адреса канала 0, которые формируются обычным способом (инкрементированием или декрементированием регистра адреса в зависимости от программирования). Прочитанный байт помещается в регистр временного хранения ВР. Затем канал 1 выполняет операцию передачи из ВР в память соответственно текущим адресам в своем адресном регистре. Регистр счета слов канала 1 декрементируется. Когда в ходе передач текущий регистр счета слов канала 1 обнулится (точнее, перейдет в состояние FFFFH), генерируется признак конца счета, обуславливающий появление сигнала $\overline{EOP}$, прекращающего ПДП-обслуживание.

Наращивание числа каналов ПДП

Для увеличения числа каналов ПДП несколько контроллеров могут объединяться в *многоуровневую схему* с подключением линий HRQ и HLDA контроллеров следующего яруса к входам DRQ и DACK предыдущего.

На рис. 6.36 приведен пример двухуровневой схемы с двумя контроллерами во втором ярусе, что позволяет получить восемь каналов ПДП. Если использовать показанным образом все четыре канала контроллера первого яруса, получится схема с 16 каналами.

Запросы контроллеров второго яруса распространяются через схемы приоритета первого яруса. Контроллер первого яруса именно для этого и используется, он не оперирует адресами и другими управляющими сигналами. Его роль состоит в принятии сигналов DRQ и DACK, выработке HRQ и принятии HLDA. Другие функции не требуются.

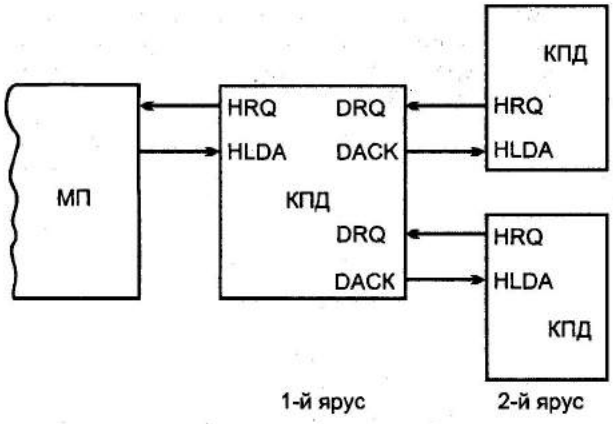


Рис. 6.36. Пример схемы наращивания числа каналов прямого доступа к памяти

§ 6.7. Таймеры

Таймеры выполняют операции, связанные с временами, частотами и интервалами. В микропроцессорных системах они используются для решения очень широкого круга задач. К таким задачам относятся, например, реализация часов реального времени, подсчет числа внешних событий, ввод/вывод широтно-модулированных сигналов, организация мультизадачных режимов для операционных систем реального времени, генерация звуковых сигналов и др.

Применительно к таймерам часто используется термин *"таймер-счетчик"*. Этот термин отражает существование двух типовых режимов работы таймеров. Если на вход таймера поданы тактирующие импульсы, т. е. сигналы постоянной частоты, то его роль состоит в формировании требуемых интервалов (режим таймера). Если же на его вход подаются внешние сигналы от каких-либо устройств, то ведется подсчет поступающих сигналов, и это является режимом счетчика.

Таймеры, входящие в состав микроконтроллеров

Далее приводятся сведения о таймерах, входящих в состав микроконтроллера AVR, которые являются достаточно типичными представителями этих устройств. В микроконтроллере AVR семейства 8515 три варианта таймеров: таймер 0, таймер 1 и сторожевой таймер.

Таймер-счетчик 0 (T0) выполняет лишь простые операции формирования временных интервалов и подсчета числа внешних событий. Структура таймера 0 показана на рис. 6.37. Основной блок таймера — восьмиразрядный счетчик, на вход которого поступают тактирующие сигналы с выхода мультиплексора MUX 8→1. Мультиплексор в зависимости от адресующего кода CS0CS2 выбирает для подачи на вход счетчика один из восьми сигналов. Выбор "заземленного" входа мультиплексора отключает счетчик, внешний сигнал может быть подан на счетчик в двух вариантах — как прямой или как инвертированный, а тактовые сигналы процессора — непосредственно или через делитель частоты с коэффициентами деления 8, 16, 256, 1024.

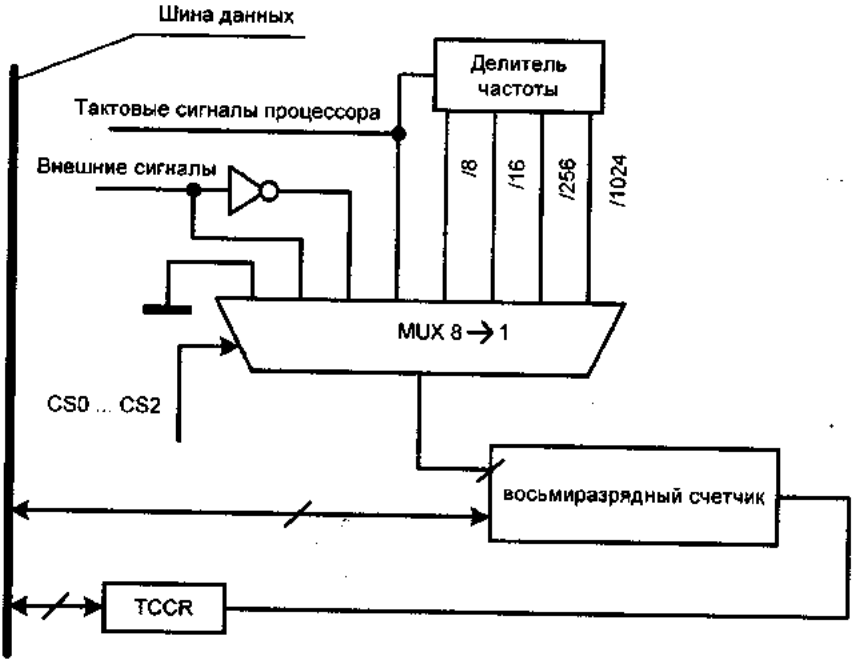


Рис. 6.37. Структура таймера 0 микросхемы AVR

Получая входные сигналы, счетчик ведет их подсчет и при переполнении (переходе от кода FFH к коду 00H) генерирует запрос на прерывание. Разрешение или запрещение прерываний от таймера регламентируются установкой или сбросом соответствующего бита в регистре масок прерываний. Таким образом, таймер T0 генерирует метки времени. Интервалы между этими метками зависят от выбора коэффициента деления делителя частоты. В режиме подсчета внешних событий содержимое счетчика инкрементиру-

ется от фронтов сигналов, выражающих факт наступления события. Счетчик доступен для записи и чтения в любое время.

Таймер 1 (T1) тоже называется таймером-счетчиком, но он сложнее, чем таймер T0, и способен выполнять и дополнительные функции. В его структуре повторяются схемы, входящие в состав таймера T0, и присутствуют новые блоки (рис. 6.38).

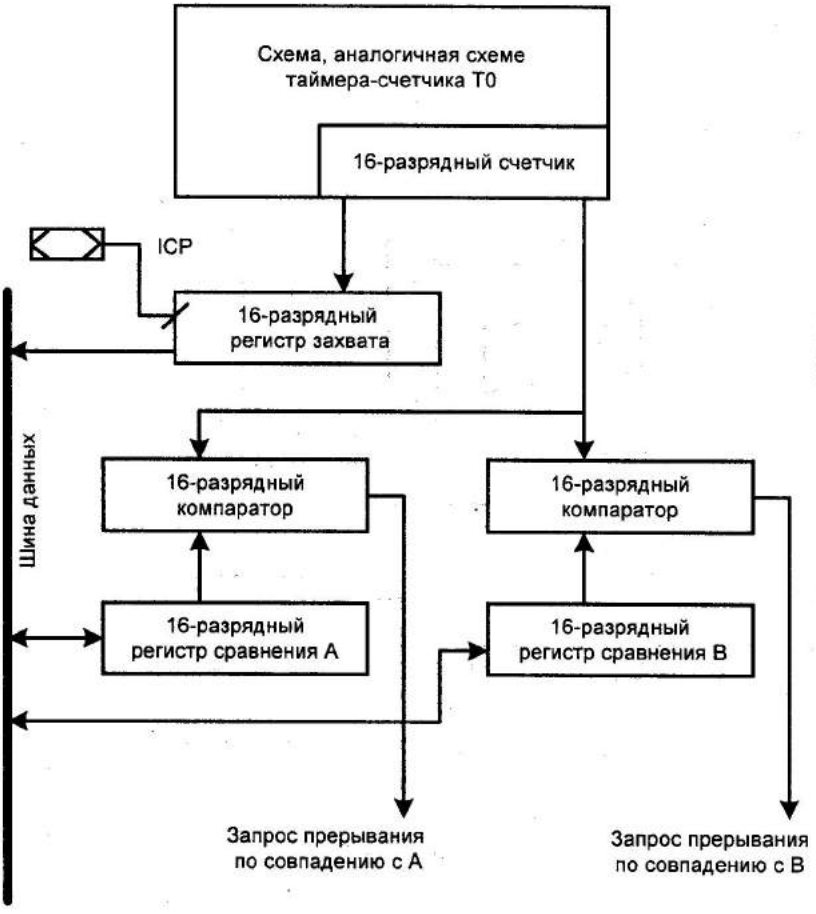


Рис. 6.38. Структура таймера-счетчика T1

В части схемы, подобной схеме таймера T0, отличие состоит лишь в разрядности счетчика (счетного регистра). В таймере T1 он не восьмиразрядный, а 16-разрядный. Новыми являются блоки 16-разрядного регистра захвата, двух 16-разрядных компараторов и двух 16-разрядных регистров совпадения (A и B).

К новым функциям относятся работа в режиме *таймера событий* или *генератора ШИМ-сигналов*. В первом из этих режимов используется регистр захвата, который в произвольный момент времени, задаваемый сигналом на внешнем входе или сигналом от аналогового компаратора, запоминает текущее состояние счетного регистра (захватывает его). При этом сигнал от внешнего входа защищен от действия помех специальными приемами подавления "дребезга контактов". Кроме того, компараторами фиксируется равенство содержимого счетного регистра и некоторых величин, заданных в регистрах совпадений, что может служить сигналом для определенных действий (сброса счетного регистра, изменения состояния какого-либо вывода МК).

Признаки состояний счетчика (переполнение, совпадение, захват) отображаются в регистре прерываний от таймера соответствующими флажками, а разрешение или запрещение прерываний задается битами регистра маскирования прерываний. Счетный регистр в любое время доступен для записи и чтения.

В режиме генерации ШИМ-сигнала (ШИМ-режиме) счетный регистр работает как реверсивный счетчик, а сама широтно-импульсная модуляция заключается в выработке импульсных сигналов с программируемыми частотой и скважностью. В ШИМ-режиме содержимое счетчика изменяется от нулевого до некоторого максимального (TOP-значения), затем направление счета изменяется и содержимое счетчика, уменьшаясь, возвращается к нулевому значению, после чего такой цикл вновь повторяется.

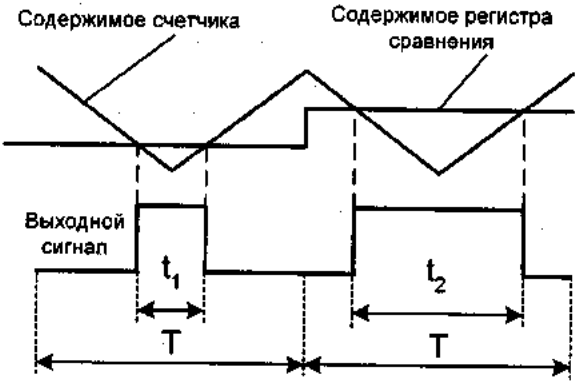


Рис. 6.39. К пояснению процесса генерации ШИМ-сигнала таймером T1

Пренебрегая ступенчатым характером изменения кода в счетчике, можно считать, что содержимое счетчика изменяется по пилообразному закону. Частота и амплитуда пилообразного сигнала зависят от входной частоты счетчика и его модуля и от TOP-значения. Код в регистре совпадения может

меняться в моменты достижения счетчиком TOP-значения. Компаратор сравнивает содержимое счетчика с величиной модулирующего воздействия, отображаемого кодом в регистре совпадения, выделяя во времени интервалы по признаку "больше" или "меньше" для указанных кодов. В результате генерируется ШИМ-сигнала, как показано на рис. 6.39.

Сторожевой таймер предназначен для отслеживания ситуаций, в которых МК неправильно выполняет программу, что может быть следствием воздействия помех, особенно опасных при работе МК в окружении мощного силового оборудования. Сторожевой таймер, переполняясь, вызывает сброс МК, если таймер не будет сброшен в течение определенного промежутка времени. Сброс МК означает его перезапуск, после чего начинается новое выполнение программы. Таким образом, сторожевой таймер защищает работу системы от сбоев. Если программа выполняется правильно, она должна периодически сбрасывать сторожевой таймер через промежуток времени, меньший его периода. Для управления работой сторожевого таймера (его включения/выключения и задания периода переполнения, называемого тайм-аутом) используются биты соответствующего управляющего регистра.

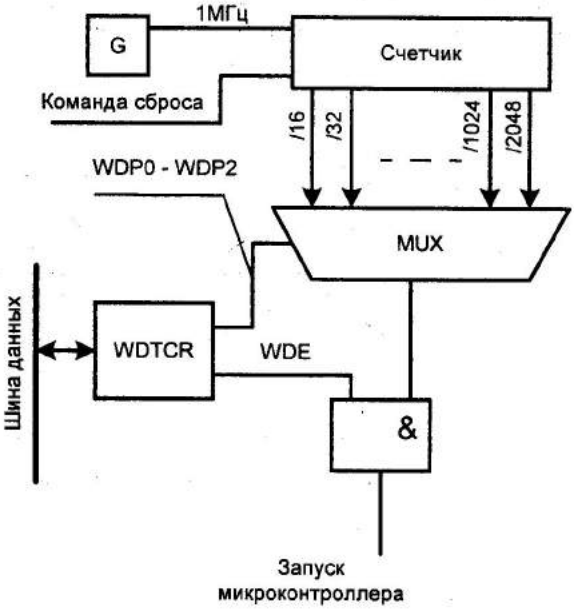


Рис. 6.40. Структура сторожевого таймера

Сторожевой таймер (рис. 6.40) имеет независимый генератор тактовых сигналов (в МК AVR частота независимого генератора при напряжении пита-

ния 5 В равна 1 МГц, а при напряжении питания 3 В это 350 кГц). Независимый генератор сохраняет рабочее состояние сторожевого таймера даже в режиме глубокого понижения мощности Power Down, так что этот таймер может быть использован для "пробуждения" микроконтроллера, т. е. вывода его из состояния Power Down. Сбросы таймера происходят под воздействием команды WDR (Watchdog Reset), а импульсы сброса/запуска контроллера выбираются мультиплексором с одного из восьми выходов счетчика. Коэффициенты деления входной частоты счетчика для этих выходов различны, а периоды тайм-аута в зависимости от выбора того или иного выхода составляют 16, 32, 64, 128, 256, 512, 1024 или 2048 мс. Величина тайм-аута выбирается трехразрядным кодом WDP0—WDP2, загруженным в регистр WDTCR. Включение сторожевого таймера производится установкой бита WDE в том же управляющем регистре.

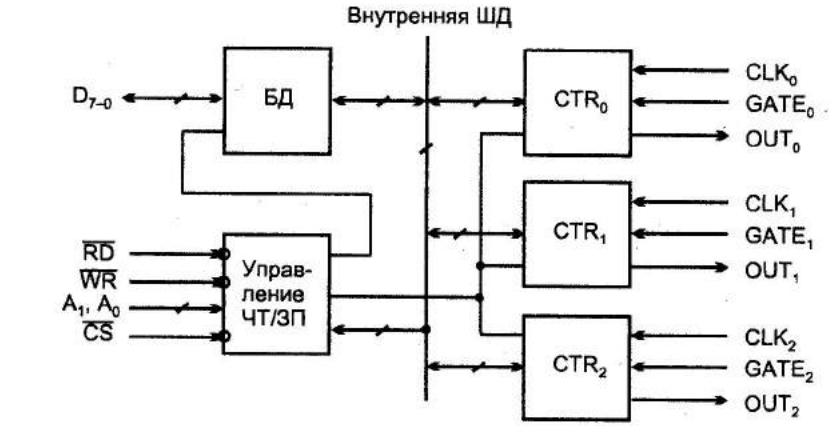
Программируемый интервальный таймер ВИ54

Программируемый интервальный таймер (ПИТ, PIT) ВИ54 серий К1821 и К1860 (аналог микросхемы Intel 8254), входящий также в состав интегрированных периферийных СБИС и библиотек для СБИС программируемой логики, — трехканальный, содержит три 16-разрядных счетчика с независимыми режимами работы при изменениях входной частоты от нулевой до 10 МГц (для разных модификаций максимальные частоты 5; 8 и 10 МГц). Таймер может работать в шести режимах в двоичной или двоично-десятичной системах счисления.

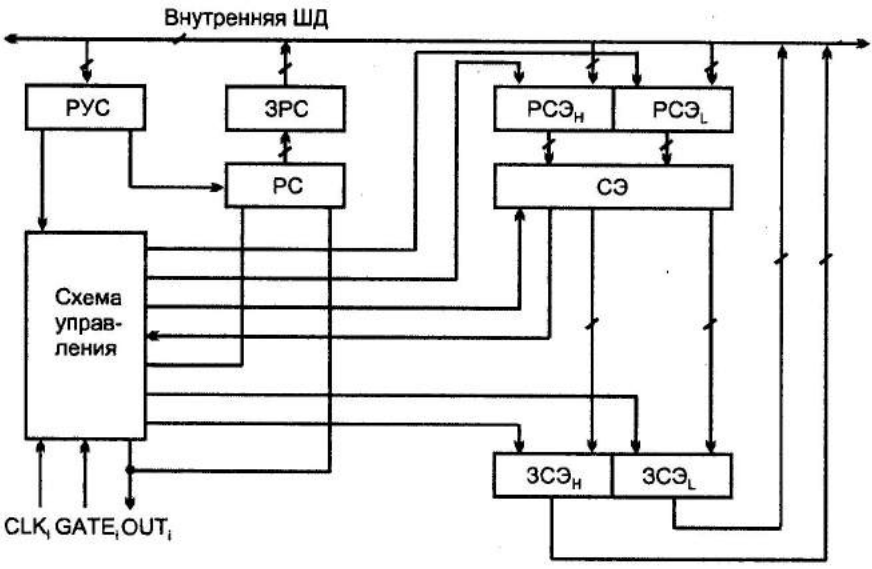
Структура ИС ВИ54 показана на рис. 6.41, *а*. Двунаправленный буфер данных БД с тремя состояниями выхода связывает ПИТ с шиной данных системы. Блок управления чтением-записью принимает от шин МПС сигналы $\overline{RD}$ ($\overline{IOR}$) или $\overline{WR}$ ($\overline{IOW}$), первый из которых передает содержимое адресуемого счетчика или регистра ПИТ на шину данных, а второй загружает байт с этой шины в адресуемый счетчик или регистр. Сигнал $\overline{CS}$ разрешает или запрещает работу ПИТ. Сигналы младших линий адреса A_1 и A_0 выбирают конкретный счетчик CTR_i комбинациями 00, 01 и 10, или регистр управляющего слова (комбинацией 11).

Внутренняя структура счетчика CTR_i приведена на рис. 6.41, *б*. Регистр управляющего слова РУС хранит загруженные в таймер сведения о назначенных режимах работы счетчиков. Регистр состояния РС вместе с защелкой ЗРС (защелкой регистра состояния) содержат текущее состояние РУС, состояние выхода OUT и флажок нуля счета. Собственно счетчик, обозначенный как СЭ (счетный элемент) — 16-разрядный синхронный вычитающий, со сбросом. Его состояние отслеживается двумя восьмиразрядными защелками счетного элемента ЗСЭ для старшего (H) и младшего (L) байтов числа, формируемого в счетчике. Имеется команда Counter Latch, по которой текущее число СЭ фиксируется в защелках до его считывания процессором, после чего защелки

вновь возвращаются в режим слежения за числом в СЭ. Управляющая логика обеспечивает поочередный вывод содержимого защелок на внутреннюю шину данных для вывода 16-разрядных слов по восьмиразрядным шинам. Состояние СЭ может читаться только через защелки.



а



б

Рис. 6.41. Структура интервального таймера (*а*) и одного из его каналов (*б*)

СЭ имеют на входах восьмиразрядные регистры счетного элемента РСЭ, хранящие старший и младший байты нового числа, подлежащего записи в СЭ. Управляющая логика обеспечивает загрузку регистров с внутренней шины данных для побайтной передачи 16-разрядных чисел. В СЭ оба байта загружаются одновременно. При программировании счетчика регистры РСЭ сбрасываются.

Работа ПИТ протекает следующим образом. После включения питания необходимо запрограммировать каждый счетчик, который будет использоваться, записывая управляющее слово, а затем начальное число.

Управляющее слово адресуется комбинацией $A_1A_0 = 11$ и само определяет, какой счетчик программируется. Формат управляющего слова показан на рис. 6.42, *а*. Управляющее слово определяет номер счетчика, для которого оно предназначено, чтение/запись для однобайтных или двухбайтных чисел, режим работы счетчика и систему счисления используемых чисел. Сведения, касающиеся команд Read Back и Counter Latch, будут пояснены далее при рассмотрении операции чтения.

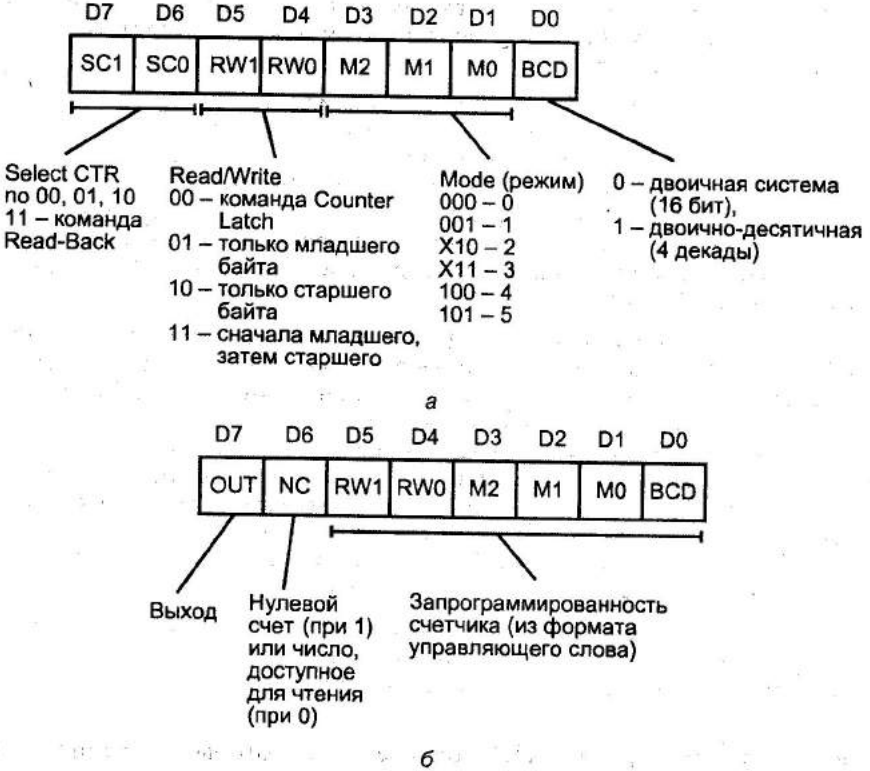


Рис. 6.42. Форматы управляющего слова (*а*) и слова состояния (*б*) интервального таймера

После управляющего слова в счетчик записывается начальное число в формате, определенном управляющим словом. Номер счетчика выбирается сигналами A_1A_0.

Процедуры инициализации таймера достаточно свободны в части последовательности подачи управляющих слов и начальных чисел для различных счетчиков. Следует соблюдать лишь два правила:

- для каждого счетчика управляющее слово должно быть записано до записи начального числа;
- начальное число должно подчиняться формату, заданному управляющим словом (соответственно битам D_5D_4 используются только младший байт, только старший байт или оба байта).

Новые начальные числа могут быть записаны в счетчик в любое время без воздействия на режим его работы.

Чтение содержимого счетчика без нарушения процесса счета организуется тремя способами: простым чтением, чтением по команде Counter Latch и чтением по команде Read Back. Первый вариант осуществляется запрещением входных тактовых сигналов выбранного счетчика по входу GATE (т. е. с помощью остановки счетчика).

При втором варианте в РУС записывается команда Counter Latch ($A_1A_0 = 11$ и D_7D_6 согласно номеру счетчика, но с битами $D_5D_4 = 00$, отличающими эту команду (биты D_3—D_0 безразличны)). Действие этой команды ведет к защелкиванию соответствующих ЗСЭ, из которых процессор читает число, после чего защелки возвращаются в режим слежения. Таким образом, содержимое счетчика читается "на лету", без влияния на его работу. Чтение идет согласно запрограммированному формату, причем для двухбайтных чисел не обязательно считывать оба байта подряд.

Третий вариант реализуется командой Read Back и позволяет проверить число в СЭ, запрограммированный режим работы и текущее состояние выхода OUT и флажок нулевого счета в выбранном счетчике. Команда записывается в РУС при $A_1A_0 = 11$, $D_7D_6 = 11$. Биты D_5D_4 определяют объект чтения (ЗСЭ или ЗРС) выбранного счетчика, биты D_3—D_1 выбирают счетчик ($D_3 = 1$ — счетчик 2, $D_2 = 1$ — счетчик 1, $D_1 = 1$ — счетчик 0). Бит D_0 не используется, должен иметь нулевое значение. Нулевое значение бита D_5 соответствует обращению к ЗСЭ выбранного счетчика, нулевое значение бита D_4 — к ЗРС. Последнее условие ($D_4 = 0$) может быть использовано для защелкивания информации о состоянии выбранного счетчика, которое доступно для операций чтения.

Формат слова состояния счетчика приведен на рис. 6.42, *б*. Он позволяет наблюдать уровень выходного сигнала, наличие в счетчике числа или нуля, запрограммированные режимы работы счетчиков.

До описания режимов работы ПИТ определим некоторые термины: CLK — тактовые импульсы, запуск — положительный фронт сигнала GATE, загрузка счетчика — передача числа из РСЭ в СЭ.

Режим 0 — прерывание по окончании счета. Здесь после записи управляющего слова выход OUT имеет низкий уровень L и остается таким до обнуления счетчика, приводящего выход к высокому уровню H до записи нового числа или управляющего слова режима 0.

Сигнал GATE разрешает (при единичном значении) или запрещает (при нулевом) процесс счета. При загрузке начального числа N переход сигнала OUT на высокий уровень происходит на N + 1 импульсе после записи начального числа. На рис. 6.43 показаны процессы в счетчике для режима 0 при постоянно разрешенном счете (*а*), наличии интервалов запрещения счета (*б*), когда сигнал GATE = var и при поступлении нового начального числа в процессе счета, т. е. перезаписи начального числа DW (Data Word) (*в*).

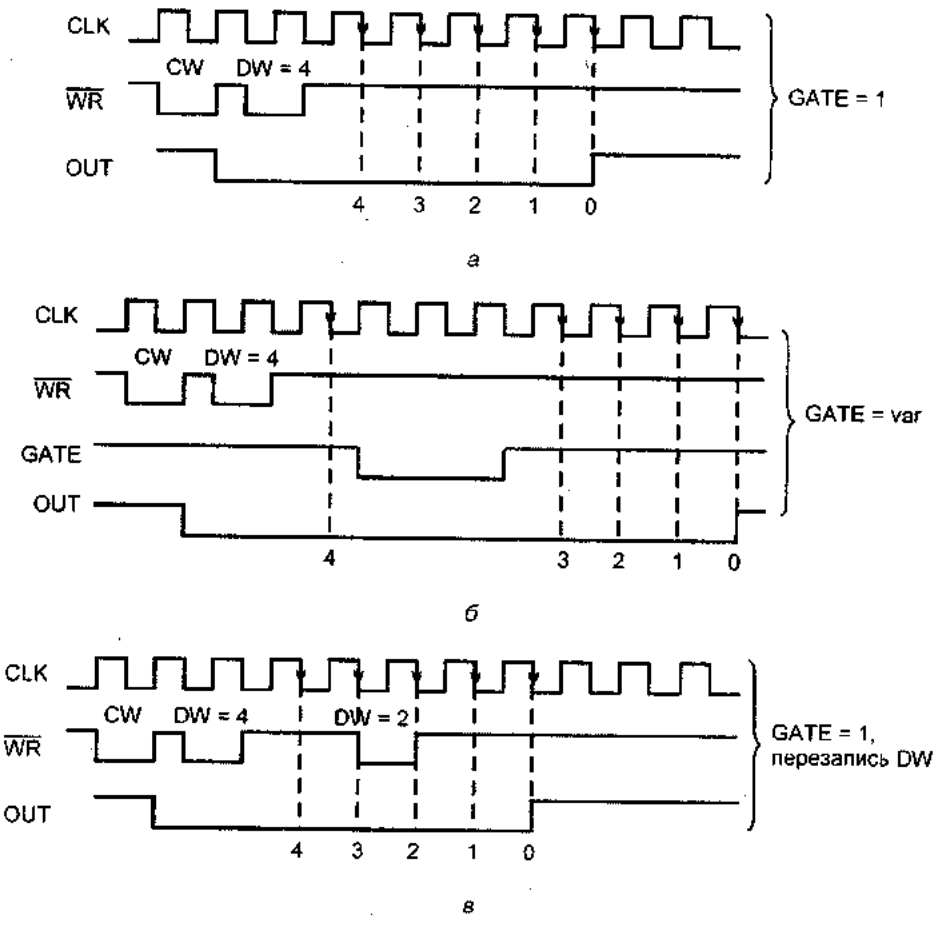


Рис. 6.43. Временные диаграммы режима 0 интервального таймера

Режим 1 — аппаратно-перезапускаемый одновибратор. В этом режиме выход OUT первоначально имеет высокий уровень, после сигнала запуска формируется его отрицательный фронт и начинается счет, а при достижении счетчиком нулевого состояния выход OUT возвращается в исходное состояние H до поступления нового сигнала запуска. Начальное число N дает импульс длительностью в N тактов.

Одновибратор назван перезапускаемым, т. к. выход OUT остается на низком уровне в течение N тактов после любого запуска, в том числе поступившего во время существования выходного импульса. Если новое число записывается в счетчик во время импульса, то текущий импульс не изменяется, если нет перезапуска. Перезапуск продлевает импульс на время, соответствующее новому загруженному числу. Временные диаграммы режима 1 показаны на рис. 6.44.

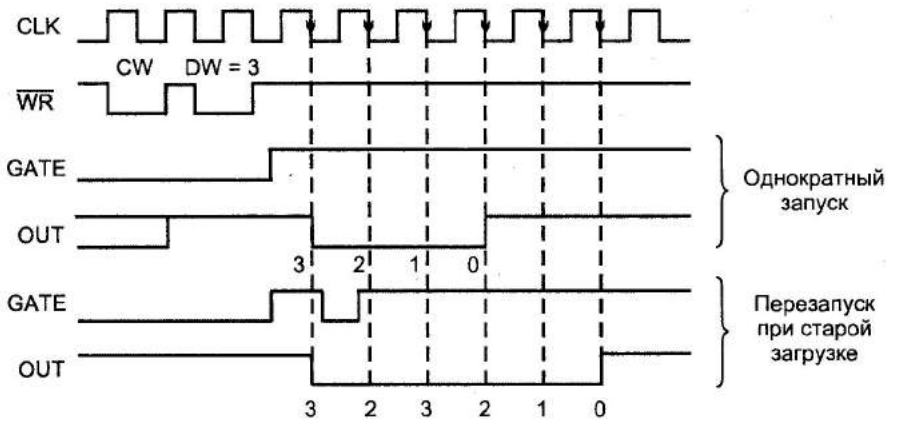


Рис. 6.44. Временные диаграммы режима 1 интервального таймера

Режим 2 — генератор частоты. В этом режиме счетчик делит частоту входных тактовых импульсов на N. Начальный уровень выхода OUT является высоким. Когда число в счетчике равно единице, выход снижается до низкого уровня L на время, равное одному периоду тактовых импульсов CLK, после чего вновь возвращается на высокий уровень H, счетчик перезагружается начальным числом и процесс повторяется. Режим периодический, начальное число N определяет период выходных импульсов (рис. 6.45).

Запрещение счета по входу GATE ведет к немедленному переходу выхода OUT на высокий уровень. Сигнал запуска перезагружает счетчик начальным числом. Запись нового числа во время счета не влияет на текущую последовательность счета.

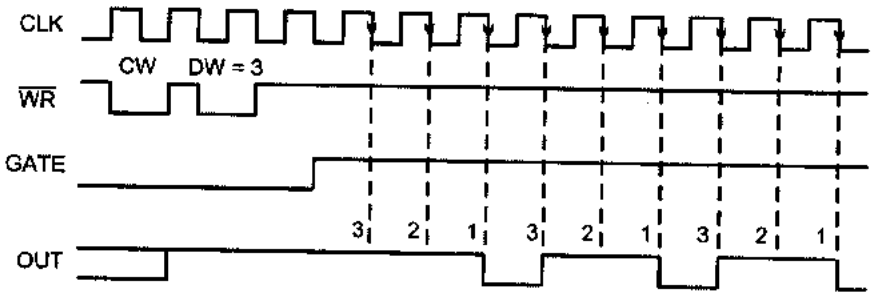


Рис. 6.45. Временные диаграммы режима 2 интервального таймера

Режим 3 — генерация *меандра*, т. е. последовательности импульсов с приблизительно одинаковыми длительностями обоих уровней H и L, т. е. с приблизительно одинаковыми длительностями импульса и паузы. Такие сигналы часто используются для тактирования бодовой скорости при последовательных передачах данных. Этот режим близок к режиму 2. Начальный уровень выхода OUT высокий. После уменьшения начального числа до его половины сигнал OUT переходит на низкий уровень на оставшееся время счета. Начальное число N определяет период выходных импульсов. Запрещающий уровень сигнала GATE немедленно переводит выход OUT на высокий уровень. Запуск перезагружает счетчик.

Запись нового числа во время счета не влияет на текущую последовательность. Если после записи нового слова происходит перезапуск до конца текущего полупериода, счетчик будет загружен новым числом в конце текущего полупериода и счет будет продолжен. При печетном начальном числе N длительности импульса и паузы составляют соответственно (N + 1)/2 и (N − 1)/2 периодов частоты CLK.

Режим 4 — генератор одиночного программно-запускаемого строба. Начальное состояние выходного сигнала OUT — высокий уровень. При полном списывании начального числа выход OUT переходит на низкий уровень на время одного тактового импульса частоты CLK и затем возвращается на высокий уровень. Счетная последовательность запускается самой записью начального числа. Перезагрузка счетчика во время счета дает следующее:

- загрузка младшего байта не влияет на счет;
- загрузка второго (старшего) байта позволяет новому числу записаться в счетчик по следующему импульсу частоты CLK (рис. 6.46).

Это позволяет последовательности запускаться от программного воздействия. Выход OUT перейдет на низкий уровень во время N + 1 импульса частоты CLK после записи нового числа N.

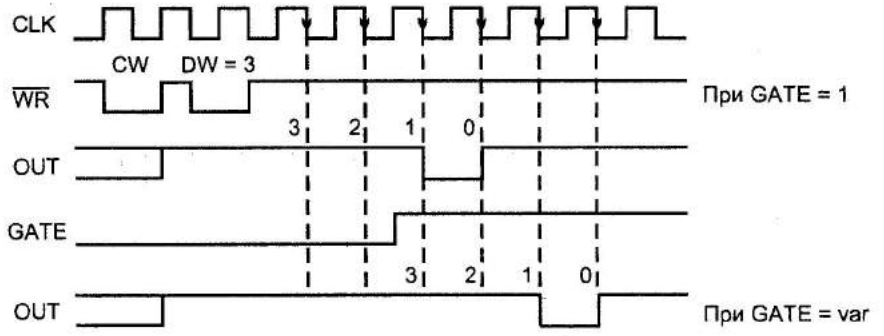


Рис. 6.46. Временные диаграммы режима 4 интервального таймера

Режим 5 — генератор одиночного аппаратно-запускаемого строба. Вначале выход имеет высокий уровень. Счет запускается фронтом сигнала GATE. Списывание начального числа ведет к переходу сигнала OUT на низкий уровень на время одного импульса частоты CLK, после чего OUT возвращается на высокий уровень (рис. 6.47).

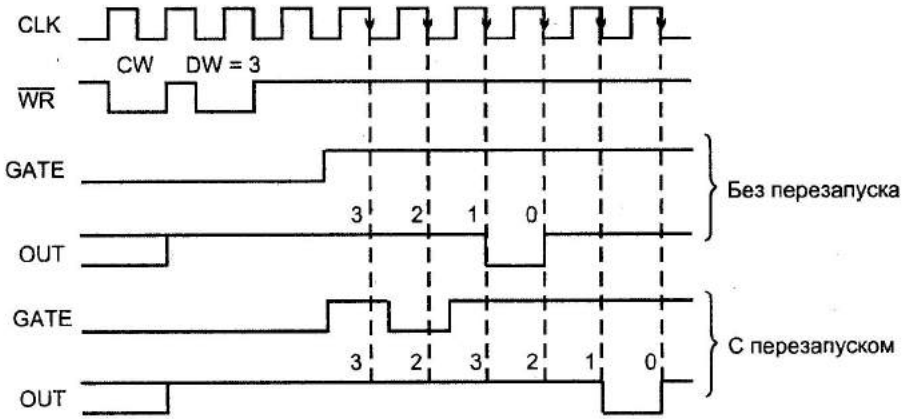


Рис. 6.47. Временные диаграммы режима 5 интервального таймера

§ 6.8. Схемотехника интерфейса JTAG

Интерфейс JTAG и граничное сканирование

Если рассмотренные выше интерфейсные схемы обслуживали операции обмена данными, то JTAG-интерфейс помимо обмена решает и иные, специ-

фичные задачи тестирования БИС и конфигурирования программируемых структур.

Термином JTAG-интерфейс была обозначена совокупность средств и операций, позволяющих проводить тестирование БИС/СБИС без физического доступа к каждому их выводу. Аббревиатура JTAG возникла по наименованию разработчика — объединенной группы по тестам Joint Test Action Group. Термином "граничное сканирование" (ГС) или, иногда, "периферийное сканирование" (по-английски Boundary Scan Testing или BST) назвали тестирование по JTAG-стандарту (IEEE Std 1149.1). Такое тестирование возможно только для микросхем, внутри которых имеется набор специальных элементов — ячеек граничного сканирования BSC (Boundary Scan Cells) и схем управления их работой. Позднее функции интерфейса JTAG были расширены и он нашел широкое применение для конфигурирования микросхем с программируемой структурой. В настоящее время все большее число БИС/СБИС поддерживает транспортный механизм интерфейса JTAG, а многие и весь стандарт 1149.1.

Основная концепция граничного сканирования иллюстрируется рис. 6.48, *а*. Ячейки BSC размещены между каждым внешним выводом микросхемы и схемами кристалла, образующими само проверяемое устройство, и могут работать в разных режимах. *В рабочем режиме* они просто пропускают сигналы через себя слева направо и не изменяют функционирования устройства. Входные сигналы проходят через ячейки BSC прямо к соответствующим точкам основных схем кристалла. При этом выводы обычного логического типа снабжаются одной ячейкой BSC, для выходов с третьим состоянием нужны две (вторая для выработки сигнала управления буфером), для двунаправленных выводов — три, что видно из рис. 6.48, *а*. Однако и здесь возможно так называемое пассивное тестирование. Поскольку ячейки BSC, соединяясь последовательно, образуют сдвигающий регистр, в котором могут быть зафиксированы значения пропускаемых сигналов, а они в последствии могут быть прочтены тестирующим устройством.

В режиме активного тестирования пропуск сигналов через ячейки прекращается, а в сдвигающий регистр из тестового прибора можно ввести последовательно тестовый код и переписать его в статический регистр-защелку для подачи на входные точки основной схемы кристалла. Оба регистра (сдвигающий и статический) создаются ресурсами самих ячеек. Результат, который выработает основная схема, можно передать в выходные ячейки BSC и затем вывести из БИС последовательно в режиме сдвигающего регистра для сравнения с ожидаемым правильным результатом (анализ результата, как и подготовку входных данных для тестирования, осуществляет внешний тестирующий прибор).

Схема ячейки BSC (рис. 6.48, *б*) содержит два мультиплексора и два D-триггера. В зависимости от адресного входа "Тестирование" выходного мультиплексора ячейка либо свободно пропускает сигнал со входа на выход, либо передает на выход состояние триггера Т2, образующего разряд статического регистра. Адресный сигнал входного мультиплексора "Сдвиг" управляет подачей на первый триггер входного сигнала (от логических входов микросхемы) или же сигнала от предыдущей ячейки.

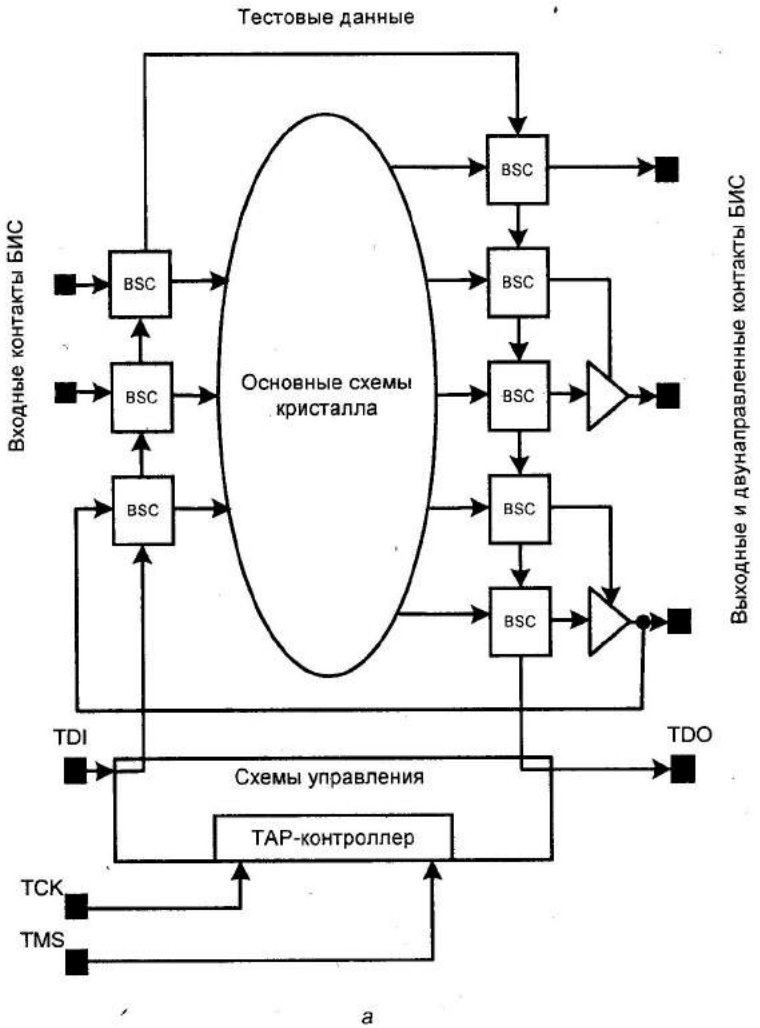
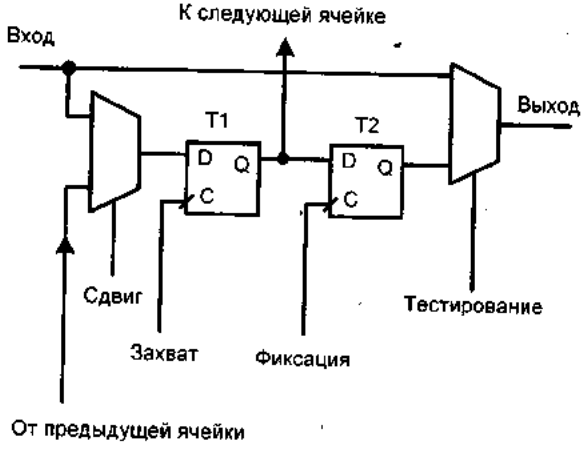


Рис. 6.48. Структура аппаратных средств интерфейса JTAG (*а*)



б

Рис. 6.48. Схема ячейки граничного сканирования (*б*)

Таким образом, совокупность триггеров T1 при подаче на них сигналов от нижнего входа мультиплексора образует сдвигающий регистр, а при подаче сигналов от верхнего входа мультиплексора эти триггеры загружаются параллельно данными от входных контактов БИС. При передаче через входной мультиплексор сигнала от предыдущей ячейки тактовый сигнал C1 ("Захват") производит сдвиг на один разряд в регистре, образованном последовательным соединением триггеров T1. Параллельная загрузка триггеров T1 также тактируется сигналом C1.

По синхросигналу "Фиксация" текущее содержимое регистра, составленного из цепочки триггеров T1, переписывается в статический регистр, составленный из триггеров T2. Сдвиги в регистре на триггерах T1 не будут влиять на содержимое регистра на триггерах T2.

Интерфейс JTAG. При разработке интерфейса JTAG основным требованием была минимизация числа контактов БИС, используемых для обмена информацией при выполнении операций тестирования. Таких контактов четыре (реже пять), их совокупность образует *порт доступа TAP* (Test Access Port). Назначение контактов:

- TCK — для сигнала синхронизации передач данных и команд;
- TMS — для выбора режима передач;
- TDI — для ввода данных и команд;
- TDO — для вывода данных, команд или состояния;
- TRST (если используется) — для сбрасывания в исходное состояние контроллера TAP.

Столь малое число контактов оказывается достаточным вследствие последовательного характера передач команд и данных.

Транспортный механизм. Если в тестируемом устройстве установлено несколько БИС с интерфейсом JTAG, то они могут быть объединены *в JTAG-цепочку* (рис. 6.49).

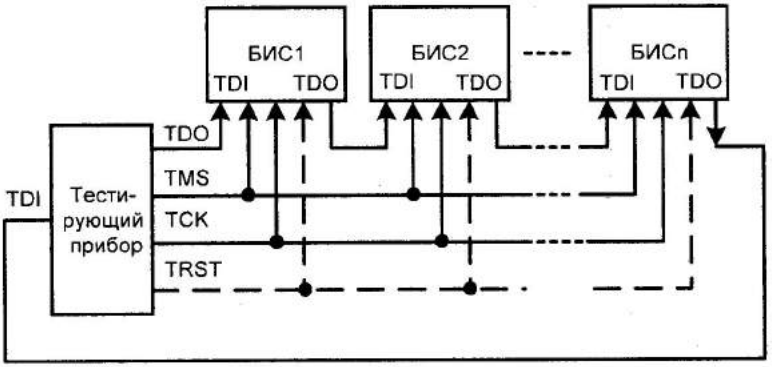


Рис. 6.49. Структура JTAG-цепочки

При выполнении любых действий команды и данные передаются в цепочке последовательно. Используя только контакты TMS и TCK, устройство управления JTAG-цепочкой, входящее в состав тестирующего прибора, может устанавливать автоматы TAP-контроллеров всех БИС цепочки в любое требуемое состояние (исходное, загрузки команд или данных в регистры, чтения из них). Совокупность регистров всех БИС цепочки является как бы одним регистром на пути от выходного контакта TDI тестирующего прибора до его же входа TDI. Поэтому при настройке одной из БИС цепочки необходимо составить и ввести в цепочку последовательность битов с длиной, соответствующей всей цепочке (это относится как к цепочке регистров команд, так и к цепочке регистров данных). В режимах передач данных и команд каждый импульс TCK сдвигает на один разряд код в цепочке регистров.

Устройство управления граничным сканированием. В это устройство (рис. 6.50) входят регистры команд (IR), данных (DR) и пропуска (BYPASS), а также выходной мультиплексор и контроллер управления (TAP Controller). Регистр данных (сканирующий регистр) образован последовательным соединением триггеров T1 ячеек BSC и принимает или выдает данные при выполнении в JTAG-цепочке любых команд.

Параллельно регистру данных включен регистр команд (параллельное включение означает общность последовательных входа и выхода TDI и TDO). Устройство управления принимает тактирующие сигналы TCK и интерпре-

тирует команды на входе TMS, выбирая тот или иной регистр для записи или чтения. Зафиксированные в регистрах-защелках команды дешифруются и формируют сигналы выбора режима граничного сканирования или регистров данных, подключаемых через мультиплексор к выходу TDO.

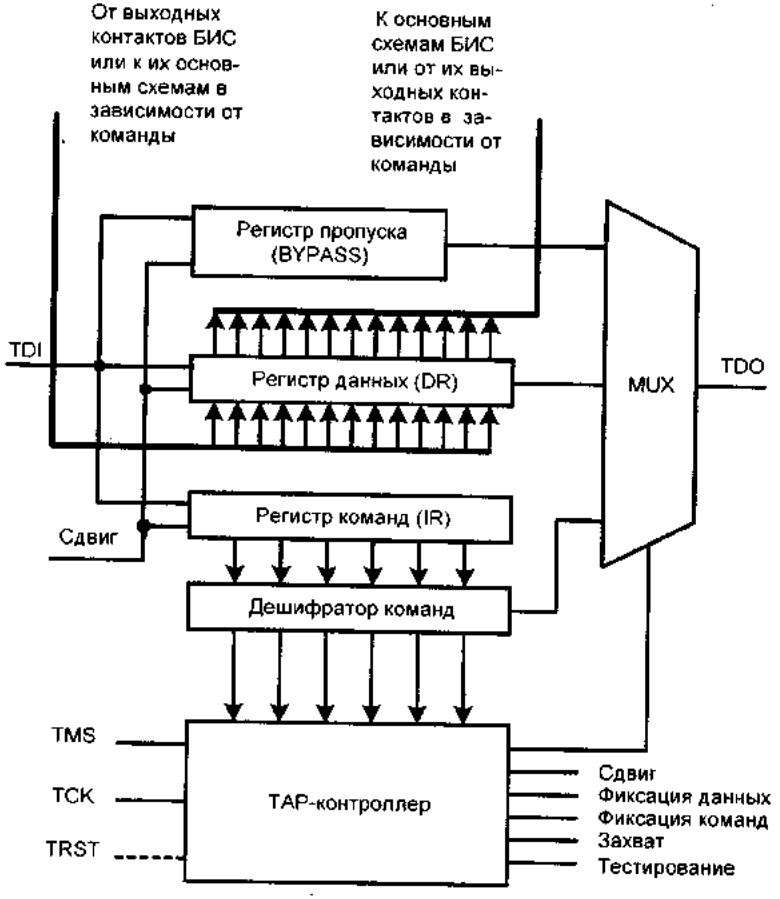


Рис. 6.50. Структура устройства управления граничным сканированием

Одноразрядный регистр пропуска (обхода) BYPASS используется в режимах загрузки/выгрузки данных как обходной путь для сдвигаемых многоразрядных данных, не относящихся к данной БИС. Прохождение в некоторых БИС JTAG-цепочки данных от входа TDI к выходу TDO через одноразрядный регистр BYPASS сокращает длину цепочки и ускоряет процессы тестирования.

Механизм граничного сканирования. Граничное сканирование выполняется в следующей последовательности: загрузка команды, загрузка данных, исполнение команды, чтение результата. Функциональные возможности ячеек BSC позволяют организовать различные режимы граничного сканирования.

- *Самотестирования БИС или записи/чтения внутрисхемных ЗУ.* В этом режиме выводы БИС изолируются от ее внутренних схем, на которые подается информация от триггеров T2 BSC-ячеек (JTAG-цепочки). Дальнейшие действия определяются поступившей командой. Результирующая информация (реакция на введенные входные данные) может быть зафиксирована в триггерах T1 ячеек BSC и передана для анализа в тестирующий прибор, что и требуется для самотестирования БИС или записи/чтения внутрисхемных ЗУ.
- *Тестирования межсоединений БИС.* В этом режиме внешние контакты БИС также отключаются от ее внутренних схем. В выходные ячейки JTAG-цепочки одной БИС загружаются данные, которые затем передаются во входные ячейки JTAG-цепочки второй БИС. При исправности межсоединений БИС данные должны совпадать, что и проверяется тестирующим прибором.
- *Тестирования штатной работы БИС.* Здесь внутренние схемы кристалла соединяются с внешними контактами БИС, что соответствует ее рабочим режимам. В ячейках BSC в заданный момент времени фиксируется состояние всех контактов. Передача полученных данных в тестирующий прибор позволяет оценить правильность работы БИС. При этом внутренние сигналы схемы становятся известными без физического доступа к ее контактам.
- *Смешанные режимы*, в которых часть БИС находится в штатном режиме, а другая часть — в тестовом.

Команды граничного сканирования. Выполняемые в ходе граничного сканирования тестовые процедуры определяются командами, поступающими из тестирующего прибора (обычно это персональный компьютер). Согласно стандарту минимальным является набор из четырех команд — INTEST, BYPASS, EXTEST, SAMPLE/PRELOAD. Можно использовать и дополнительные команды.

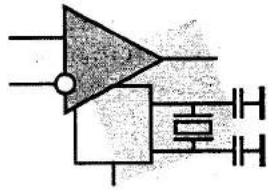
- Команда INTEST проверяет функционирование БИС при подаче данных от тестирующего прибора в сканирующие регистры DR. Запуск процедуры сканирования ведет к проверке основных схем кристалла. Затем в регистре DR фиксируются результаты тестирования.
- Команда RUNBIST тестирует кристалл без привлечения каких-либо внешних данных, активизируя встроенную в кристалл тестирующую схему. Результаты тестирования заносятся в регистр DR и позволяют оценить исправность БИС.

- Команда EXTEST используется для тестирования внешних соединений БИС в устройстве (на печатной плате) и для задания входных воздействий другим БИС. При ее выполнении данные, загруженные в BSC-ячейки выходов БИС, считываются ячейками входов другой БИС, отражая исправность их межсоединений. Внутренние схемы кристалла при этом отключаются от внешних выводов.
- Команда SAMPLE/PRELOAD в зависимости от состояния управляющего автомата позволяет либо "захватить" в сдвигающий регистр граничного сканирования данные с параллельных входов ячеек BSC либо загрузить данные из сдвигающего регистра в статический регистр-защелку (зафиксировать данные, т. е. обновить содержимое статического регистра). Эта команда используется и при проверке поведения БИС в рабочих режимах, фиксируя данные на ее внешних контактах. Внутренние схемы кристалла от внешних контактов при этом не отключаются. Наличие "снимка" режимов нескольких взаимосвязанных БИС и возможность задания различных конфигураций тестируемой системы (одна группа БИС формирует тестовые сигналы, а другая находится в рабочем режиме) позволяет организовать совместное тестирование межсоединений и функционирования внутренних схем БИС.
- Команды BYPASS, CLAMP и HIGHZ помещают между выводами TDI и TDO одноразрядный регистр BYPASS при различных состояниях выводов БИС (первая не влияет на работу схем на кристалле, вторая устанавливает на выводах данные от статического регистра, третья — Z-состояния).

С помощью интерфейса JTAG, как уже отмечалось, можно производить также реконфигурацию микросхем программируемой логики непосредственно в системе, без извлечения их из устройства.

Литература к главе: [14], [18], [27], [28], [34], [37], [50], [XIV], [XXIX].

Глава 7



Микроконтроллеры

§ 7.1. Основные сведения

Микроконтроллеры (МК) — разновидность микропроцессорных систем (микро-ЭВМ), ориентированных на реализацию алгоритмов управления техническими устройствами и технологическими процессами. Микроконтроллеры проще, чем универсальные микро-ЭВМ, и уже около 25 лет тому назад оказалось возможным размещать их на одном кристалле в виде "*однокристальных микро-ЭВМ*". Микроконтроллеры — БИС такой функциональной законченности, которая позволяет решать в полном объеме задачи определенного класса с помощью одного кристалла.

Что отличает МК от универсальной микро-ЭВМ? Прежде всего, это малый объем памяти и менее разнообразный состав внешних устройств. В состав универсальной микро-ЭВМ входят модули памяти большого объема и высокого быстродействия, имеется сложная иерархия ЗУ, поскольку многие задачи (автоматизированное проектирование, компьютерная графика, мультимедийные приложения и др.) без этого решить невозможно. Для МК ситуация иная, они реализуют несложные алгоритмы, и для размещения программ им требуются емкости памяти, на несколько порядков меньшие, чем у микро-ЭВМ широкого назначения. Для хранения промежуточных данных достаточна память небольшой емкости. Набор внешних устройств также существенно конкретизируется и сужается, а сами они значительно проще. В результате модули универсальной микро-ЭВМ (процессор, память, интерфейсные схемы) требовалось выполнять как конструктивно самостоятельные, тогда как МК размещается на одном кристалле, хотя и имеет модули того же функционального назначения.

Сопоставляя микропроцессор (т. е. центральный процессорный элемент достаточно сложной системы) и МК (т. е. микросхему простой системы в целом) с точки зрения коммерческих потребностей, можно видеть преобладание МК. Число пользователей МК в несколько раз превышает число пользователей микросхем МП. Применение МК поддерживается такими областями массового производства, как бытовая аппаратура, станкострое-

ние, автомобильная промышленность, военное оборудование и т. д. Годовой мировой выпуск микроконтроллеров сейчас оценивается цифрой 2 млрд, а их номенклатура насчитывает тысячи типов.

Первые МК были выпущены фирмой Intel в 1976 г. (восьмиразрядный МК 8048). В настоящее время многими поставщиками выпускаются восьми-, 16- и 32-разрядные МК с емкостью памяти программ до десятков килобайт, небольшими ОЗУ данных и набором таких интерфейсных и периферийных схем, как параллельные и последовательные порты ввода/вывода, таймеры, аналого-цифровые и цифроаналоговые преобразователи, широтно-импульсные модуляторы и др.

Среди выпускаемых МК широко известно семейство восьмиразрядных контроллеров MCS-51/151/251 и 16-разрядных MCS-96/196/296 (фирма Intel). Очень многие производители выпускают аналоги этих семейств или совместимые с ними МК. В отечественной номенклатуре это восьмиразрядные МК К1816ВЕ51, К1830ВЕ51. В последнее время фирма Intel сосредоточила усилия на разработке сложных микропроцессоров для компьютеров и уступила сектор рынка простых МК другим фирмам, в частности, фирме Atmel, которая выпускает несколько популярных семейств МК. Признанными авторитетами в области создания и производства МК являются такие фирмы, как Motorola, Microchip, Zilog и др. В настоящее время микроконтроллеры все чаще применяют в составе СБИС программируемой логики типа "система на кристалле".

Несмотря на появление новых 16- и 32-разрядных МК, наибольший успех на рынке остается за восьмиразрядными. Сейчас около половины рынка занято 8-разрядными МК, которые лидируют с большим отрывом относительно микроконтроллеров других разрядностей. На рынке восьмиразрядных микроконтроллеров доминирует следующая тройка: семейство 8051 фирмы Intel (аналоги микроконтроллеров этого семейства выпускаются несколькими фирмами), семейство AVR (фирмы Atmel) и микроконтроллеры семейства PIC (фирмы Microchip). В качестве примера современного микроконтроллера далее рассмотрена микросхема из семейства AVR. Сейчас, правда, самая большая доля рынка (около 25%) все еще принадлежит микроконтроллерам семейства 8051, за которыми следуют семейства AVR и PIC (приблизительно по 15%). Остальные микроконтроллеры по объему продаж значительно отстают от лидеров. Микроконтроллеру AVR предпочтение отдано как обладающему хорошо продуманной архитектурой и высоким быстродействием. Микросхемы AVR используют RISC-процессоры, которые в последнее время интенсивно внедряются в структуры микроконтроллеров. К тому же микроконтроллеры семейства 8051 многократно описаны в литературе, начиная с работ пятнадцатилетней давности, например, [50], и кончая современными, например, [34] и [37].

Микроконтроллеры марки AVR подразделяются на три семейства, среди которых базовым является семейство Classic. Ниже рассматривается представитель именно этого семейства, который для краткости называется просто микроконтроллером AVR.

Микроконтроллеры (далее иногда просто контроллеры) AVR имеют RISC-архитектуру и изготовляются по усовершенствованной КМОП-технологии.

> Напомним, что по одному из классификационных признаков микропроцессоры (и, в частности, микроконтроллеры) могут принадлежать к *CISC- или RISC-процессорам*. Процессоры CISC имеют сложную систему команд (CISC — Complex Instruction Set Computer), т. е. большой набор разноформатных команд, и используют многие способы адресации. Архитектура CISC присуща классическим (традиционным) процессорам, она в силу многообразия команд позволяет применять эффективные алгоритмы решения задач, но, в то же время, усложняет схему процессора и его стоимость и в общем случае не обеспечивает его максимального быстродействия.
>
> Процессоры типа RISC имеют сокращенную систему команд (RISC — Reduced Instruction Set Computer), из которой исключены редко применяемые команды. Форматы команд, по крайней мере, подавляющее их большинство, идентичны (например, все команды содержат по 4 байта), резко снижено число используемых способов адресации. Данные, как правило, обрабатываются только с регистровой или непосредственной адресацией. Значительно увеличенное число регистров процессора, т. е. его емкая внутренняя память, позволяет редко обращаться к внешнему модулю памяти микропроцессорной системы, а это повышает быстродействие контроллера. Идентичность временных циклов выполнения команд отвечает потребностям конвейерных схем обработки информации. В результате может быть достигнуто упрощение схемы процессора при увеличении его быстродействия.

Контроллеры семейства AVR имеют следующие параметры:

- почти все команды выполняются за один машинный такт, что при тактовой частоте 1 МГц дает производительность в 1 MIPS (Million Instructions Per Second);
- флэш-память программ емкостью 1—8 Кбайт имеет допустимое число репрограммирований 10^3;
- статическая память данных (SRAM) имеет емкость до 512 байт;
- память данных типа EEPROM с допустимым числом репрограммирований 10^5 имеет емкость 64—512 байт;
- многоуровневая система прерываний обслуживает от 3 до 16 источников запросов прерываний;
- имеется достаточно обширный набор периферийных устройств.

Базовая линия развития контроллеров AVR (линия Classic) насчитывает около двух десятков моделей. Далее рассматривается модель AVR 8515, об-

ладающая повышенной функциональной полнотой и поддерживающая большую часть возможностей, характерных для всего семейства в целом.

§ 7.2. Структура микроконтроллера

МК AVR — восьмиразрядный RISC-микроконтроллер с Гарвардской архитектурой и пониженным энергопотреблением. Набор команд, ограниченность которого свойственна RISC-архитектурам, в данном случае необычно широк (120 команд), однако при этом сохранено основное преимущество RISC-архитектур — повышенное быстродействие и сокращенное число операций обмена с памятью программ. Почти все команды размещаются в одной ячейке программной памяти и выполняются за один такт синхросигнала. Типичен режим с частотой синхронизации 1 МГц. Максимальная частота синхросигнала составляет 8 МГц. Доступ к памяти программ и памяти данных осуществляется через собственные шины этих модулей, поэтому можно не только сделать различными разрядности шин, но и реализовать параллелизм операций в процессах выполнения текущей команды и выборки и дешифрации следующей, т. е. ввести в работу МК элементы конвейеризации.

На рис. 7.1 показана схема МК AVR типа AT90S8515. Микроконтроллер имеет восьмиразрядную шину данных, посредством которой его модули обмениваются информацией.

Заметим, что с целью упрощения рисунка разрядности шин на нем не указаны, но они легко могут быть определены на основе достаточно простых соображений: все шины, выходящие на шину данных, имеют по 8 разрядов, такова же разрядность блока регистров и АЛУ, счетчик команд соответственно емкости памяти программ (2048 слов, т. е. 4096 байт) является 12-разрядным, флэш-память имеет 16-разрядную организацию, определяющую разрядность регистра команд IR. Линии шины управления RESET, ALE, ICP, входы и выходы последовательных каналов блоков SPI и UART, линия канала последовательного программирования флэш-памяти и EEPROM, линии передачи аналоговых сигналов и др. являются одноразрядными.

Многие блоки AVR по назначению аналогичны рассмотренным в *главе 5* блокам микропроцессора и имеют те же самые обозначения. Программный счетчик PC содержит адрес подлежащей выполнению команды и адресует флэш-память программ. Считанная из флэш-памяти команда поступает в регистр команд IR, ее КОП (код операции) декодируется дешифратором команд для выработки сигналов управления блоками микроконтроллера соответственно заданной операции, а КАД (адресная часть) адресует данные в блоке регистров или в памяти данных SRAM. В памяти типа EEPROM хранятся редко изменяемые данные (калибровочные константы и т. п.). Указатель стека SP используется для организации стека в некоторой области па-

мяти SRAM, глубина стека ограничивается только наличием свободной области в этой памяти. Регистры общего назначения (РОН) объединены в регистровый файл.

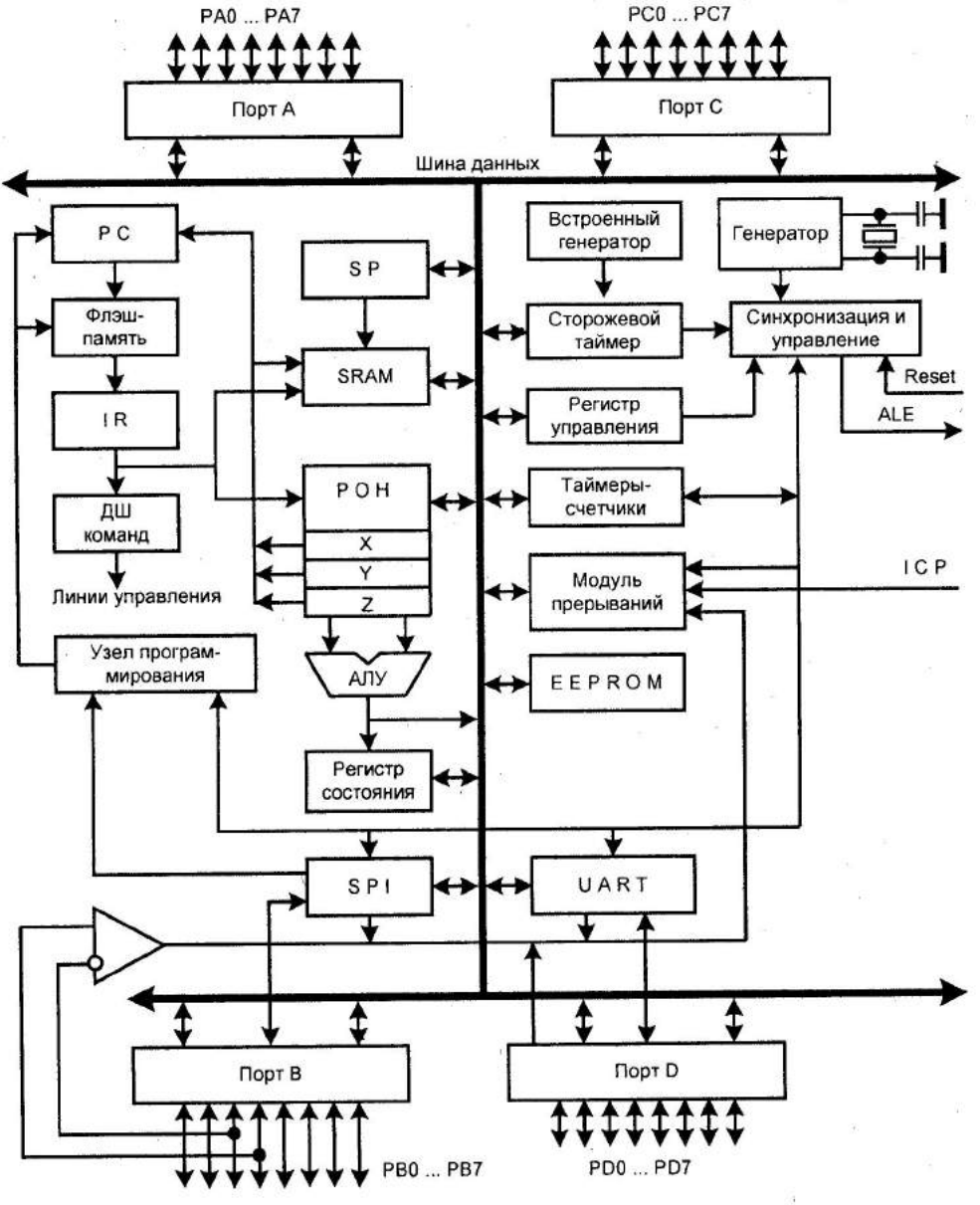


Рис. 7.1. Структура микроконтроллера AVR

Арифметико-логическое устройство (АЛУ) способно выполнять операции над содержимым любой пары регистров блока и направлять результат в любой регистр, т. е. *все регистры РОН непосредственно доступны для АЛУ.* Этим микроконтроллер AVR отличается как от рассмотренного выше микропроцессора, так и от микроконтроллеров других фирм, в которых рабочим регистром для АЛУ обычно служит только один регистр (аккумулятор). Наличие у АЛУ многих рабочих регистров позволяет выполнять операции за один такт. Три регистровые пары (X, Y, Z), получаемые объединением двух восьмиразрядных регистров в один 16-разрядный, используются для косвенной адресации. Регистр состояния по функциональному назначению аналогичен регистру флажков RF в структуре рассмотренного ранее микропроцессора, он содержит признаки результатов при выполнении некоторых команд (нуль, знак, перенос, половинный перенос и т. д).

Генератор синхросигналов имеет внешние выводы для подключения кварцевого или иного резонатора либо внешнего тактирующего сигнала. Кроме основного синхрогенератора микроконтроллер имеет и дополнительный встроенный RC-генератор с фиксированной частотой 1 МГц (при напряжении питания 5 В) для тактирования сторожевого таймера. Вход RESET (L-активный) служит для сброса микроконтроллера (приведения его в исходное состояние), а также перевода его в режим программирования при подаче на этот вход специального повышенного напряжения 12 В. Выход ALE имеет то же назначение, что и одноименный выход рассмотренного ранее микропроцессора и используется при подключении к микроконтроллеру внешнего ЗУ. В этом случае реализуется режим мультиплексируемой шины: старший полуадрес выводится в течение всего цикла через один восьмиразрядный порт, а младший — через второй восьмиразрядный порт загружается в начале цикла во внешний регистр-защелку, где сохраняется на все время цикла. После загрузки внешнего регистра-защелки этот же порт используется для передачи данных и сигналов управления.

Модуль прерываний служит для приема и обработки запросов прерывания основной программы, как внутренних, так и внешних. Предусмотрено наличие 10 внутренних и двух внешних запросов.

Блоки, относящиеся к центральному процессорному элементу, рассматривались при описании микропроцессора в *главе 5.* Порты ввода/вывода, таймеры, интерфейс SPI, блоки UART рассмотрены в *главе 6.* Описания не рассмотренных ранее блоков (таких, например, как аналоговый компаратор) даются далее.

Структура одного из четырех портов ввода/вывода МК AVR приведена на рис. 7.2. Порт представляет собою набор из восьми линий. Каждая из восьми линий любого порта конфигурируется как входная или выходная индивидуально с помощью управляющего слова, загружаемого в регистр направления передачи. Каждый бит этого слова задает конфигурацию своей линии.

Выводимые или вводимые данные поступают в регистр данных. Входные и выходные сигналы проходят через буферные каскады (драйверы).

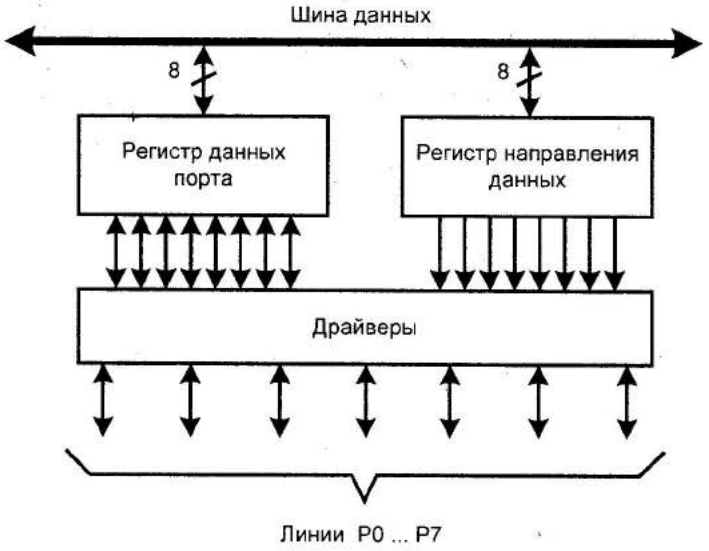


Рис. 7.2. Структура порта ввода/вывода микроконтроллера AVR

Для уяснения процессов работы основных цепей в линиях порта следует обратиться к рис. 6.3 и тексту к нему, а затем к более полному рис. 6.4. Возможности чтения состояния кнопки с помощью схем рис. 6.3 и 6.4 иллюстрируются на рис. 6.5. В состав микроконтроллера AVR входят таймеры (таймер 0, таймер 1 и сторожевой таймер), структуры которых показаны на рис. 6.37, 6.38, 6.40.

Последовательный периферийный интерфейс SPI применяется как для программирования микроконтроллера в последовательном режиме, так и для обмена данными с периферийными устройствами или между микроконтроллерами. Протокол обмена для SPI предусматривает работу МК в режиме либо ведущего (Master), либо ведомого (Slave). Скорости передачи задаются синхросигналами, получаемыми делением тактовой частоты МК, и имеют четыре программируемых значения: 1/4, 1/16, 1/64 и 1/128 от этой частоты. Интерфейс SPI рассмотрен в *главе 6* (см. рис. 6.22).

Асинхронный последовательный интерфейс между микроконтроллером и внешними устройствами обеспечивается блоком UART. Этот блок преобразует параллельные данные от микроконтроллера в последовательные для внешнего устройства и наоборот, работает по асинхронному протоколу, предусматривающему обмен с квитированием для учета готовности блоков к

пересылке очередного байта. Структура и функционирование блока UART рассмотрены в *главе 6* (см. рис. 6.21).

Аналоговый компаратор сравнивает напряжения, подаваемые на две специально выделенные линии порта В. Одно из этих напряжений сигнальное, другое опорное. Выходной сигнал компаратора имеет логический характер, показывает, какое из двух напряжений больше, доступен программе и поступает на блок прерываний. С помощью аналогового компаратора можно, например, измерять длительность входных импульсов, поскольку начало и конец импульса могут быть отмечены изменением состояния компаратора, а эти события могут служить источниками команды захвата для таймера, что позволит зафиксировать цифровые значения моментов начала и конца импульса. Можно возразить, что эта задача решается и самим таймером без аналогового компаратора, но это так только для импульсов, у которых уровни напряжения соответствуют уровням логических сигналов. При использовании аналогового компаратора таких ограничений нет и можно измерять длительности импульсов с произвольными значениями высокого и низкого уровней, установив величину опорного напряжения между этими уровнями.

§ 7.3. Организация памяти и функционирование микроконтроллера

Распределение памяти в МК AVR.

Адресные пространства микроконтроллера показаны на рис. 7.3. Согласно Гарвардской архитектуре, адресные пространства (АП) для памяти программ и памяти данных разделены. Память данных организована линейно и имеет два АП. В первом находятся адреса регистровой памяти и статического ЗУ (SRAM). Во втором размещены адреса энергонезависимой репрограммируемой памяти EEPROM. Кроме того, возможно подключение к микроконтроллеру внешнего ОЗУ, для которого шина адреса и мультиплексируемая шина адресов/данных организуются с помощью портов ввода/вывода PA и PC. Линии этих портов формируют 16-разрядные адреса для работы с внешней памятью большой емкости (до 64 Кбайт). При отсутствии внешней памяти достаточно использовать девятиразрядные адреса.

В регистровой области памяти данных размещены адреса регистров общего назначения РОН (32 адреса для адресации 32 байтов памяти) и регистров ввода/вывода РВВ (64 адреса для адресации 64 байтов памяти). Для косвенной адресации используются регистры X, Y, Z, представляющие собою 16-разрядные пары байтовых регистров. Общая емкость регистровой памяти составляет 96 байтов. Для адресов статической памяти SRAM отведены следующие 512 адресов. Подключение внешнего ОЗУ, как уже отмечалось, может довести емкость памяти до 64 Кбайт. Обращение к внешнему ОЗУ уве-

личивает время выполнения команды на 1—2 такта для каждого обрабатываемого байта.

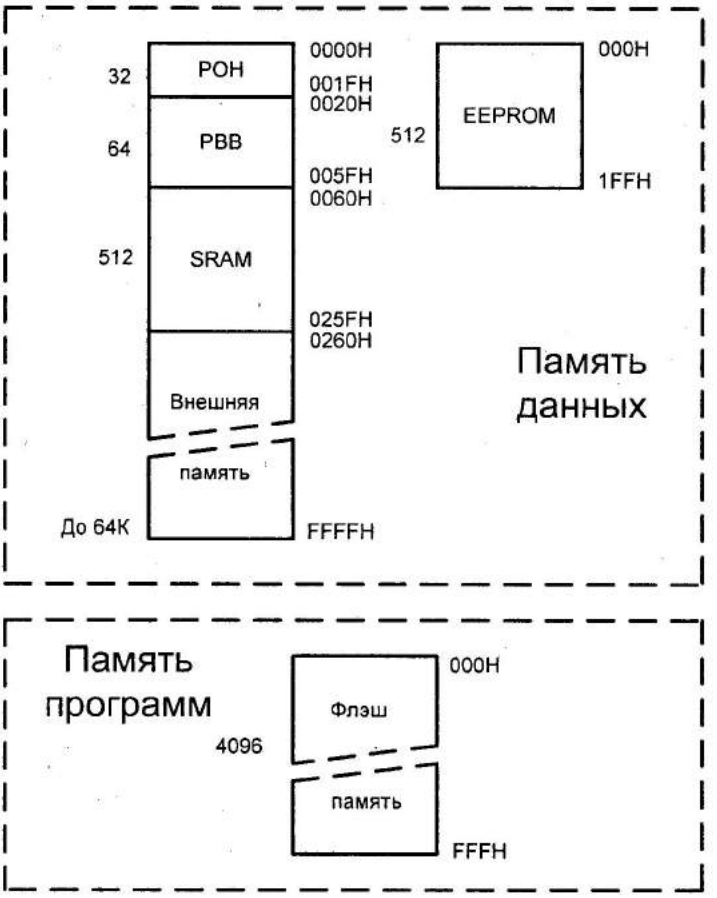


Рис. 7.3. Адресные пространства микроконтроллера AVR

При обращении к разным областям памяти данных используются команды с различными способами адресации. Адреса области РВВ являются операндами команды (прямая адресация).

В пространстве РВВ размещены служебные регистры микроконтроллера и регистры, относящиеся к внешним устройствам. В их числе 12 регистров для работы с портами ввода/вывода (таких портов 4, для каждого предусматриваются регистр данных, регистр направления данных и регистр выводов, функции которых рассмотрены в *главе 6*). Имеется регистр — указатель стека. Глу-

бина стека определяется наличием свободной области памяти программ. Многочисленные регистры обеспечивают работу модулей SPI, UART, таймеров, стека, имеются регистры состояния и управления микроконтроллером, регистр флагов и масок запросов прерывания, управления памятью EEPROM. Для управления микроконтроллером служит регистр с битами разрешения внешнего ОЗУ, разрешения перехода к режиму пониженного энергопотребления и выбора конкретного варианта из таких режимов, условий генерации запросов внешних прерываний (по фронту сигнала, по уровню и др.).

К РОН и РВВ возможны обращения как по командам ввода и вывода IN и OUT, так и как к ячейкам ЗУ. При этом для разных способов обращения диапазоны адресов РВВ несколько смещены (на 32 адреса).

Энергонезависимая память EEPROM, рассчитанная на хранение редко изменяющихся данных и имеющая длительные операции записи, отличается особой организацией. Доступ к ней происходит с использованием трех регистров области РВВ. Это регистры адреса, данных и управления. Так как емкости памяти в 512 байт соответствует девятиразрядный адрес, для адреса нужна пара регистров РВВ. Регистр данных служит для размещения данных, подлежащих записи в память или считываемых из нее. В регистре управления задействованы три бита: один определяет наличие или отсутствие разрешения записи, второй разрешает саму запись, третий разрешает чтение. Установка второго и третьего битов вызывает выполнение соответствующих процессов. Длительность цикла записи при напряжении питания 5 В составляет 2 мс, при 2,7 В — 4 мс. Чтение выполняется за один такт синхронизации.

Программы хранятся во флэш-памяти, разрядность которой соответствует формату команд и равна 16. По начальному адресу 000H расположен вектор сброса, конечный адрес равен FFFH, т. е. емкость памяти составляет 4096 слов или 8192 байта (организация флэш-памяти 4K×16).

Способы адресации, используемые в микроконтроллере

В микроконтроллере применяются прямая и косвенная адресации.

Прямая используется в следующих случаях:

- при адресации одного РОН, что занимает в слове команды 5 бит, или при адресации двух РОН с занятием 10 бит (по 5 бит для адресов источника и приемника данных);
- при пересылке между РВВ и РОН с занятием 11 бит в слове команды (6 на адрес РВВ и 5 на адрес РОН);
- при обращении ко всему адресному пространству ОЗУ с использованием двух слов команды, из которых второе целиком отдается под адрес ячейки памяти.

Косвенная с использованием индексных регистров X, Y, Z применяется в таких вариантах:

- простая (адрес находится в индексном регистре);
- относительная (адрес вычисляется как сумма содержимого индексного регистра и константы, содержащейся в команде);
- с преддекрементом (до обращения к ячейке памяти адрес в индексном регистре уменьшается на единицу);
- с постинкрементом (после обращения по адресу в индексном регистре его содержимое увеличивается на единицу).

Два последних варианта косвенной адресации позволяют эффективно обрабатывать массивы данных.

Выполнение команд при работе микроконтроллера

Повышению производительности МК способствует *выполнение почти всех команд за один такт машинного времени*. Такая возможность — следствие конвейерной обработки информации и непосредственного подключения АЛУ ко всем РОН. В нормальном режиме работы конвейера параллельно выполнению текущей команды производится выборка и декодирование следующей. При нарушении естественного следования команд (выполнении команд переходов) в работе конвейера возникает разрыв. В этом случае уже выбранная для очередной операции команда не может быть использована и приходится возвращаться к выбору новой, нарушая нормальный ритм работы конвейера. Время выполнения команды увеличивается на 2—4 такта.

Начало выполнения программы инициируется *сигналом сброса*, после которого МК обращается к адресу стартовой команды. Сброс вызывается несколькими причинами — включением питания, внешним сигналом, сигналом от сторожевого таймера, снижением питающего напряжения ниже допустимого уровня. После события, вызывающего сброс, запускается *таймер задержки сброса* и лишь после отработки таймером интервала задержки формируется внутренний сигнал сброса. Задержка позволяет микроконтроллеру принять определенное исходное состояние, необходимое для нормального начала работы. Таймер задержки сброса работает от внутреннего RC-генератора, основной генератор за время задержки выходит на стационарный режим. Относительно большое время выхода на стационарный режим характерно, в частности, для генератора с кварцевой стабилизацией частоты.

В ходе выполнения программы контроллер выполняет команду за командой, пока не дойдет до команды останова. Все *множество команд*, выполняемых контроллером, можно разбить на следующие группы:

- команды пересылки данных;
- команды арифметических операций и сдвигов;

- команды логических операций;
- команды операций с битами;
- команды передачи управления;
- команды управления системой.

Сопоставляя перечисленные группы команд с группами команд микропроцессора, рассмотренного в *главе 5*, можно видеть их большое сходство. Специфичная группа команд операций с битами содержит команды установки или сброса заданного разряда в регистре РОН или РВВ. Для удобства программирования задействованным разрядам регистров РВВ присваиваются символические имена, определение которых дается в специальном файле, подключаемом в начале программы. Детальное описание команд приводится в работах [20], [34] и в справочных данных фирмы Atmel.

Режимы различного потребления мощности

Микроконтроллер может находиться в нескольких *режимах потребления мощности*. В активном режиме потребляется полная мощность, кроме того, имеются режимы покоя (Idle) и глубокого понижения мощности (Power Down). В режимы пониженного потребления мощности ("спящие") МК переходит по команде SLEEP, которая устанавливает в регистре управления контроллером два флажка — флажок перехода к режиму пониженного потребления мощности и флажок, задающий конкретный тип такого режима.

В *режиме покоя* останавливается работа процессора, а периферийные устройства и система прерывания функционируют. Поэтому выход из режима покоя возможен по запросам прерывания от внутренних и внешних источников, в частности, от таймера. "Пробуждение" микроконтроллера из состояния покоя является быстрым.

В *режиме глубокого понижения мощности* прекращается работа всех блоков контроллера кроме сторожевого таймера и подсистемы обработки внешних прерываний. Выход из этого режима возможен по внешнему сигналу сброса, сигналу сброса от сторожевого таймера или внешним прерываниям. "Пробуждение" из режима глубокого понижения мощности является медленным, для него требуется интервал времени, соответствующий задержке таймера сброса.

"Спящие" режимы позволяют резко уменьшить потребляемые токи в периоды бездействия микроконтроллера. Это особенно полезно при использовании МК в устройствах с автономным питанием. Если выход из "спящего" режима происходит по событию прерывания, МК переходит в рабочий режим выполнения соответствующей подпрограммы и затем возобновляет выполнение основной программы с той команды, которая стояла после команды SLEEP. Если же "пробуждение" является результатом процесса сброса, происходит реинициализация контроллера.

Система прерываний

Микроконтроллер имеет *многоуровневую систему приоритетных прерываний.* Под таблицу векторов прерываний отводятся младшие адреса памяти программ, начиная с адреса 001H. Каждое прерывание имеет свой адрес в таблице, который загружается в программный счетчик для обслуживания прерывания. Приоритетность запросов прерываний непосредственно связана с их адресами, чем меньше адрес, тем выше приоритет запроса. Предусмотрены следующие прерывания: два внешних, четыре связанных с таймером 1 (при захвате, совпадениях A и B, переполнении), одно, связанное с таймером 0 (при переполнении), прерывание при завершении передачи по SPI, три связанных с UART (при завершениях приема и передачи и при пустом регистре данных), прерывания от аналогового компаратора.

Программирование микроконтроллера

Программирование микроконтроллера предусматривает запись машинных кодов в память программ и необходимых данных в энергонезависимую память EEPROM. Существуют два варианта программирования: параллельное при высоком напряжении и по последовательному каналу. Первый вариант требует применения специального программатора и характерен для условий массового производства. *Программирование по последовательному каналу* осуществляется через интерфейс SPI, не требует использования высокого напряжения, производится непосредственно в системе и в силу своих достоинств наиболее удобно для модернизации программ пользователями. Схема включения микроконтроллера в режим программирования по последовательному каналу показана на рис. 7.4.

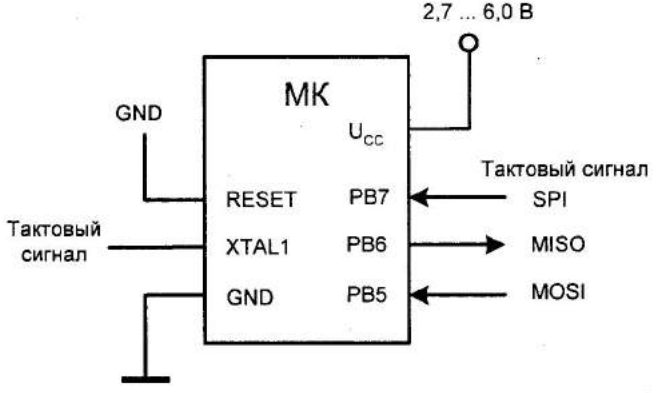
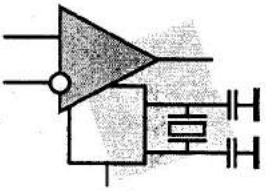


Рис. 7.4. Включение микропроцессора в режим программирования по последовательному каналу

Тактовый сигнал может подаваться от внешней схемы, создаваться подключением резонатора к выводам XTAL1 и XTAL2 микроконтроллера или восприниматься от внутреннего RC-генератора контроллера. Тактирующий сигнал SPI (сигнал SCK) должен иметь длительность обоих уровней более двух периодов тактового сигнала. Программирование состоит в посылке 4-байтовых команд на вывод MOSI. Результаты чтения снимаются с вывода MISO. Программирование памяти команд и памяти данных ведется побайтно посылкой команд "Запись флэш-памяти" и "Запись EEPROM". В этих командах содержатся адреса изменяемых ячеек и записываемое в них содержимое.

Литература к главе: [11], [14], [20], [27], [34], [VI], [XIV], [XXI], [XXXV].

Глава 8



Программируемые логические матрицы, программируемая матричная логика, базовые матричные кристаллы

§ 8.1. Вводные замечания

В цифровые системы входят как стандартные части, так и некоторые нестандартные, специфичные для данного проекта. К стандартным частям относятся, например, процессор и память, которые не изготовляются для конкретной системы по специальному заказу. Стандартный процессор решает требуемую конкретную задачу, исполняя программу, отображающую решение задачи как выполнение последовательности команд из присущего процессору фиксированного набора команд. Именно в программе, а не в структуре микропроцессора отражается конкретный характер решаемой задачи. Память также реализуется стандартными микросхемами — ее функции остаются принципиально теми же для разных систем.

Стандартные БИС/СБИС лидируют по уровню интеграции, т. к. высокая стоимость проектирования оптимизированных по плотности кристаллов, достигающая сотен миллионов долларов, оказывается в данном случае приемлемой, поскольку раскладывается на большое число производимых микросхем.

Нестандартные схемы используются для управления блоками системы, обеспечения их взаимодействия и т. д. Реализация нестандартной части системы исторически была связана с применением микросхем малого и среднего уровней интеграции (МИС и СИС). Использование МИС и СИС сопровождается резким ростом числа корпусов ИС, усложнением монтажа, снижением надежности системы и ее быстродействия, повышением стоимости. В то же время заказать для системы специализированные ИС высокого уровня интеграции затруднительно, т. к. это связано с очень большими затратами средств и времени на проектирование БИС/СБИС.

Возникшее противоречие нашло разрешение на путях разработки БИС/СБИС с программируемой и репрограммируемой структурой.

Первыми представителями указанного направления явились *программируемые логические матрицы* ПЛМ (PLA, Programmable Logic Array), *программируемая матричная логика* ПМЛ (PAL, Programmable Array Logic) и *базовые матричные кристаллы* БМК, называемые также *вентильными матрицами* ВМ (GA, Gate Array).

Микросхемы типов PLA и PAL в английской терминологии объединяются термином SPLD, Simple Programmable Logic Devices (простые программируемые логические устройства) или, короче, PLD (Programmable Logic Devices).

Появление ПЛМ, ПМЛ и БМК ознаменовало собою начало важнейшего направления развития цифровой элементной базы. Разработка БИС/СБИС с программируемой и репрограммируемой структурой оказалась чрезвычайно перспективным направлением и привела к созданию новых эффективных средств создания цифровых устройств и систем, таких как CPLD (Complex PLD), FPGA (Field Programmable GA), SOPC (System On Programmable Chip).

§ 8.2. Программируемые логические матрицы и программируемая матричная логика (ПЛМ и ПМЛ)

Структура ПЛМ

Программируемые логические матрицы появились в середине 1970-х гг. Основой их служат последовательно включенные программируемые матрицы элементов И и ИЛИ (рис. 8.1, *а*). В ПЛМ входят также блоки входных и выходных буферных каскадов (БВх и БВых).

> Заметим, что трактовка ПЛМ и ПМЛ как сочетания матриц И и ИЛИ обязана своим происхождением раннему этапу их развития. Позднее в структурах ПЛМ и ПМЛ стали встречаться матрицы других логических элементов, причем возникло несколько вариантов. Однако каноническое представление ПЛМ и ПМЛ в показанном на рис. 8.1 виде остается самым удобным, поскольку схемы, реализованные в булевском базисе, наиболее наглядны и легко понимаются. Общность изложения при этом не страдает, т. к. в конечном счете функциональные возможности разных схемных вариантов оказываются идентичными.

Входные буферы, если не выполняют более сложных действий, преобразуют однофазные входные сигналы в парафазные и формируют сигналы необходимой мощности для питания матрицы элементов И. Выходные буферы

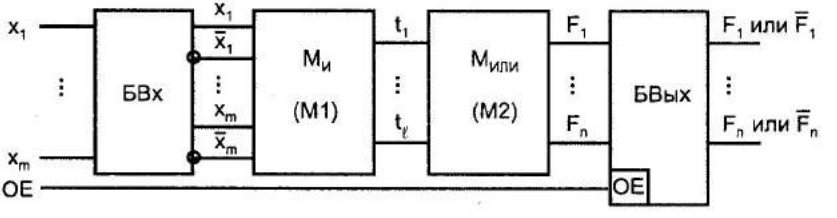
обеспечивают необходимую нагрузочную способность выходов, разрешают или запрещают выход ПЛМ на внешние шины с помощью сигнала OE, а нередко выполняют и более сложные действия.

Основными параметрами ПЛМ являются число входов m, число термов ℓ и число выходов n.

Схема ПЛМ на вентильном уровне показана на рис. 8.1, *б*. Крестики в пересечениях горизонтальных и вертикальных линий обозначают программируемые точки связей (ПТС). В зависимости от характера применяемых элементов связи возможны две ситуации с программированием ПТС. В первой незапрограммированная ПЛМ имеет соединения во всех пересечениях, а при ее программировании часть соединений удаляется. Как видно из схемы, в этом случае в исходном состоянии все термы и функции независимо от входных переменных имеют нулевые значения. Во второй ситуации все соединения отсутствуют, входные сигналы в схему не поступают, и значения термов и функций определяются внутренними цепями ПЛМ, как правило, они единичны. После программирования формируются необходимые термы, из которых и составляются требуемые функции.

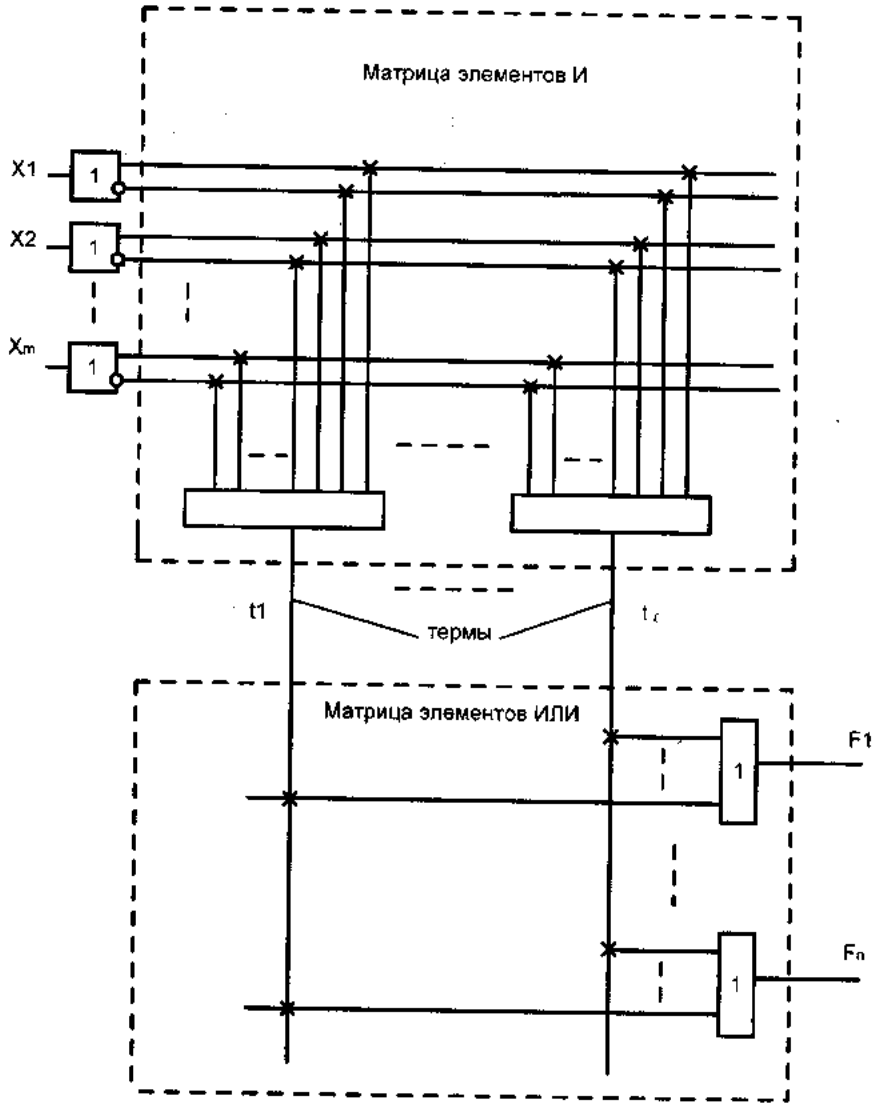
Переменные x_1—x_m подаются через БВх на входы элементов И (конъюнкторов), и в матрице И образуются ℓ термов. *Под термом здесь понимается конъюнкция, связывающая входные переменные, представленные в прямой или инверсной форме.* Число формируемых термов равно числу конъюнкторов или, что то же самое, числу выходов матрицы И. Термы подаются далее на входы матрицы ИЛИ, т. е. на входы дизъюнкторов, формирующих выходные функции. Число дизъюнкторов равно числу вырабатываемых функций n.

Таким образом, ПЛМ реализует *дизъюнктивную нормальную форму* (ДНФ) воспроизводимых функций, т. е. представляет их в виде логической суммы логических произведений (это один из вариантов двухуровневой логики). *ПЛМ способна реализовать систему n логических функций от m аргументов, содержащую не более ℓ термов.*



а

Рис. 8.1. Базовая структура ПЛМ (*а*)



б

Рис. 8.1. Схема ПЛМ на вентильном уровне (*б*)

Воспроизводимые функции — комбинации из любого числа термов, формируемых матрицей И. Какие именно термы будут выработаны и какие комбинации этих термов составят выходные функции, определяется программированием ПЛМ.

Схемотехника ПЛМ

Выпускаются ПЛМ на основе как биполярной, так и МОП-технологии. Во всех случаях в матрицах имеются системы горизонтальных и вертикальных линий, в узлах пересечения которых при программировании создаются или удаляются элементы связи.

На рис. 8.2, *а* в упрощенном виде (без буферных элементов) показана схемотехника *биполярной* ПЛМ (микросхемы К556РТ1) с программированием пережиганием перемычек. Показан фрагмент для воспроизведения системы функций

$$F_1 = \bar{x}_1\bar{x}_2x_3 \vee x_2\bar{x}_3 \vee x_1\bar{x}_4 = t_1 \vee t_2 \vee t_3;$$

$$F_2 = \bar{x}_1\bar{x}_2x_3 \vee x_1\bar{x}_2\bar{x}_3 \vee x_1x_2x_4 \vee x_2\bar{x}_3x_4 = t_1 \vee t_4 \vee t_5 \vee t_6;$$

$$F_3 = x_1\bar{x}_4 \vee x_1\bar{x}_2 = t_1 \vee t_7$$

размерностью 4, 7, 3 (параметрами микросхемы К556РТ1 в целом являются 16, 48, 8).

Элементами связей в матрице И служат диоды, соединяющие горизонтальные и вертикальные шины, как показано на рис. 8.2, *б*, изображающем цепи выработки терма t_1. Совместно с резистором и источником питания цепи выработки термов образуют обычные диодные схемы И. До программирования все перемычки целы, и диоды связи размещены во всех узлах координатной сетки. При любой комбинации аргументов на выходе будет ноль, т. к. на вход схемы И подаются одновременно прямые и инверсные значения аргументов, а $x\bar{x} = 0$. При программировании в схеме оставляются только необходимые элементы связи, а ненужные устраняются пережиганием перемычек. В данном случае на вход конъюнктора поданы $\bar{x}_1$, $\bar{x}_2$ и x_3. Высокий уровень выходного напряжения (логическая единица) появится только при наличии высоких напряжений на всех входах, низкое напряжение хотя бы на одном входе фиксирует выходное напряжение на низком уровне, т. к. открывается диод этого входа. Поскольку в данном случае выполняется операция И, вырабатывается терм $\bar{x}_1\bar{x}_2x_3$.

Элементами связи в матрице ИЛИ служат транзисторы (рис. 8.2, *в*), включенные по схеме эмиттерного повторителя относительно линий термов и образующие схему ИЛИ относительно выхода (горизонтальной линии). На рис. 8.2, *в* показана выработка функции F_1. Работа схемы ИЛИ, реализованной в виде параллельного соединения эмиттерных повторителей, была рассмотрена ранее в *§ 1.2.*

При изображении запрограммированных матриц наличие элементов связей (целые перемычки) отмечается точкой в соответствующем узле.

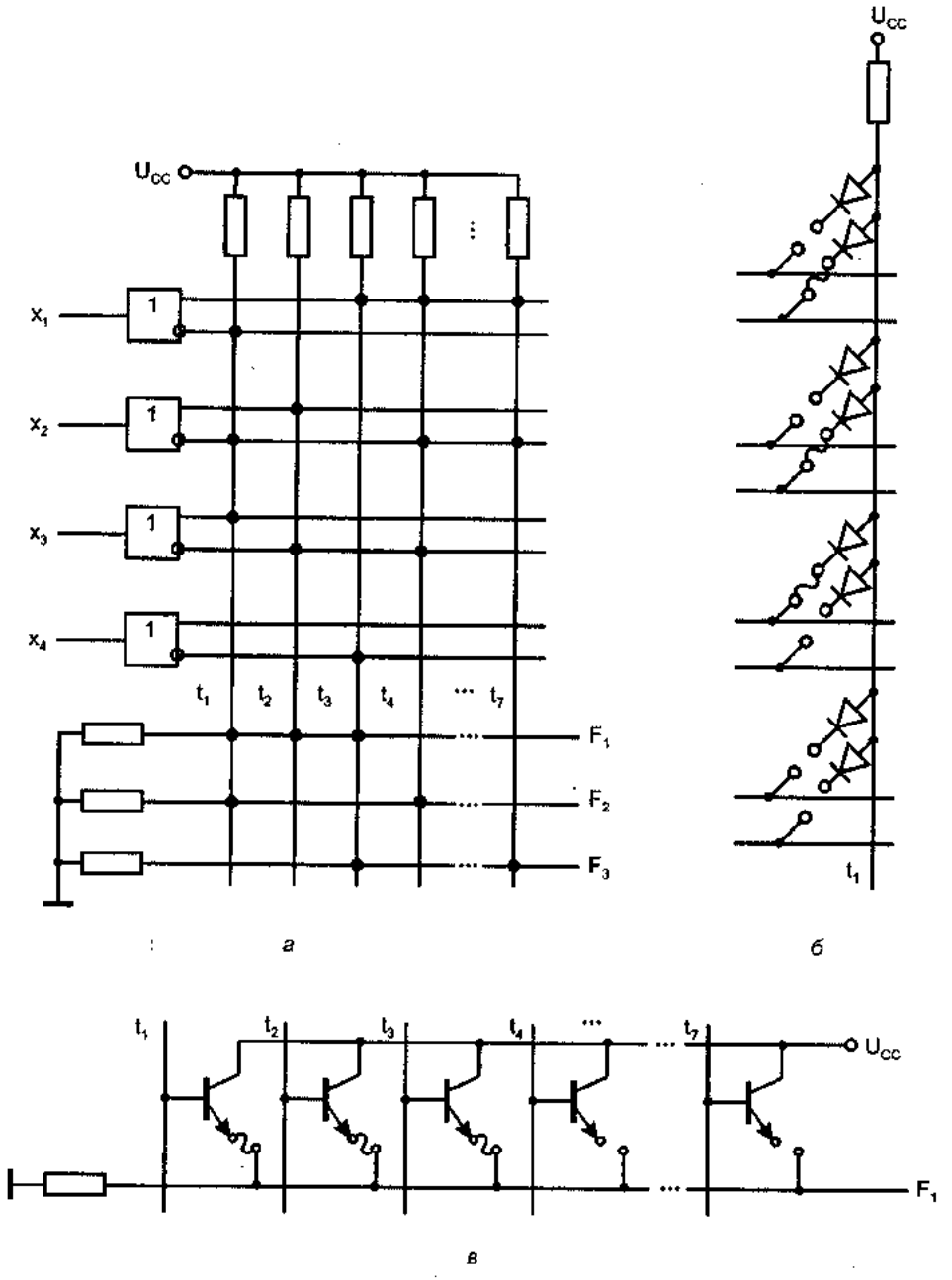


Рис. 8.2. Схемотехника ПЛМ, реализованной в биполярной технологии (*а*), и элементы связей в матрицах И (б) и ИЛИ (*в*)

В схемах на МОП-транзисторах в обеих матрицах используют один и тот же тип ячейки, наиболее удобный для реализации в принятой схемотехнологии. Чаще всего это ячейки ИЛИ-НЕ. Соответственно этому меняются и операции, реализуемые в первой и второй матрицах ПЛМ, а структура ПЛМ имеет вид (рис. 8.3, *а*). Такая ПЛМ является последовательностью двух матриц ИЛИ-НЕ, одна из которых служит для выработки термов, другая — для выработки выходных функций.

Терм t_1 в данном случае равен:

$$t_1 = \overline{x_1 \vee x_2 \vee \bar{x}_3} = \bar{x}_1 \bar{x}_2 x_3,$$

а функция:

$$F_1 = \overline{t_1 \vee t_2 \vee t_3}.$$

На основании этих выражений можно заключить, что известная связь между операциями, выражаемая правилами де Моргана, говорит о фактическом *совпадении функциональных характеристик биполярной ПЛМ и ПЛМ на МОП-транзисторах:* если на входы последней подавать аргументы, инвертированные относительно аргументов биполярной ПЛМ, то на выходе получим результат, отличающийся от выхода биполярной ПЛМ только инверсией.

На рис. 8.3, *б* показан фрагмент ПЛМ с программируемыми элементами типа EEPROM. Если в плавающий затвор транзисторного элемента связи ввести заряд электронов, то пороговое напряжение такого транзистора повысится, и он всегда будет запертым — соответствующий вход будет отключен. При отсутствии заряда в плавающем затворе транзистор функционирует как обычно. На рисунке показан один столбец первой матрицы и одна строка второй. Для входов, подключенных к управляющим затворам работающих транзисторов, образуется ячейка ИЛИ-НЕ. Действительно, для группы параллельно включенных ключевых транзисторов с общим сопротивлением нагрузки R достаточно хотя бы одного единичного входа, т. е. включенного транзистора, чтобы на выходе схемы напряжение снизилось до уровня логического нуля.

а

Рис. 8.3. Структура (*а*) ПЛМ, реализованная на МОП-транзисторах

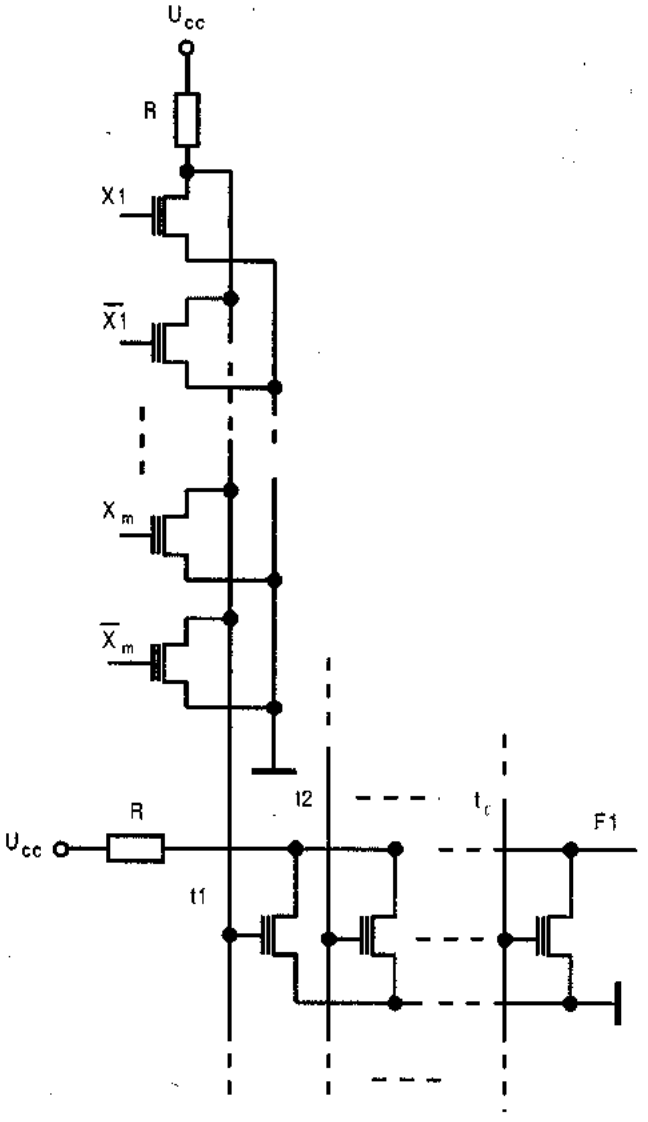


Рис. 8.3. Схемотехника (б) ПЛМ, реализованная на МОП-транзисторах

И только нулевые значения всех рабочих входов, т. е. запертые состояния всех транзисторов, позволят выходному напряжению повыситься до уровня логической единицы. Идентичные ячейки ИЛИ-НЕ используются и в первой, и во второй матрицах.

Подготовка задачи к решению с помощью ПЛМ

Для подлежащей воспроизведению системы функций с целью упрощения ПЛМ можно попытаться уменьшить число термов в данной системе. Содержанием минимизации функций будет поиск *кратчайших дизъюнктивных форм*. Имея в виду использование готовой (стандартной) ПЛМ, следует уменьшать по возможности число термов в данной системе функций задачи до уровня, когда число термов становится не превышающим ℓ — параметра имеющихся ПЛМ. Дальнейшая минимизация не требуется. Если размерность имеющихся ПЛМ обеспечивает решение задачи в ее исходной форме, то минимизация не требуется вообще, т. к. не ведет к сокращению оборудования.

Программирование ПЛМ

Программирование ПЛМ, выполняемое пользователем, проводится с помощью специальных программаторов и сведения для них должны иметь определенную форму. Имеются программаторы, которые принимают в качестве информации о ПЛМ таблицу функционирования (истинности), однако удобнее задавать сведения о самих перемычках. Символы, используемые при таком задании сведений для программирования ПЛМ:

- H — переменная входит в терм в прямом виде, т. е. нужно оставить целой перемычку прямого входа и пережечь перемычку инверсного входа;
- L — переменная входит в терм в инверсном виде, т. е. нужно сохранить перемычку у инверсного входа и пережечь у прямого;
- "–" — переменная не входит в терм и не должна влиять на него, т. е. нужно пережечь перемычки обоих входов.

  Оставление перемычек у обоих входов переменной как бы устраняет из матрицы соответствующую схему И, поскольку в силу равенства $x\bar{x} = 0$ выход этой схемы всегда нулевой и не влияет на работу матрицы ИЛИ, на вход которой подается;
- A — указывается в выходном столбце (столбце функции) и свидетельствует о связи данной схемы И с выходом ПЛМ через матрицу ИЛИ. Перемычка должна быть сохранена;
- "." — указывает на то, что данная схема И не подключается к выходу и должна иметь пережженную перемычку в матрице ИЛИ.

В принятой символике для программирования ПЛМ взятого ранее примера сведения будут заданы таблицей (табл. 8.1).

Таблица 8.1

x_1	x_2	x_3	x_4	F_1	F_2	F_3
L	L	H	—	A	A	.
—	H	L	—	A	.	.
H	—	—	L	A	.	A
H	L	L	—	.	A	.
H	H	—	H	.	A	.
—	H	L	H	.	A	.
H	L	—	—	.	.	A

Упрощенное изображение схем ПЛМ

Схемы ПЛМ достаточно громоздки, и поэтому изображать их желательно с максимально возможным упрощением. Используются изображения, в которых многовходовые элементы И, ИЛИ условно заменяются одновходовыми.

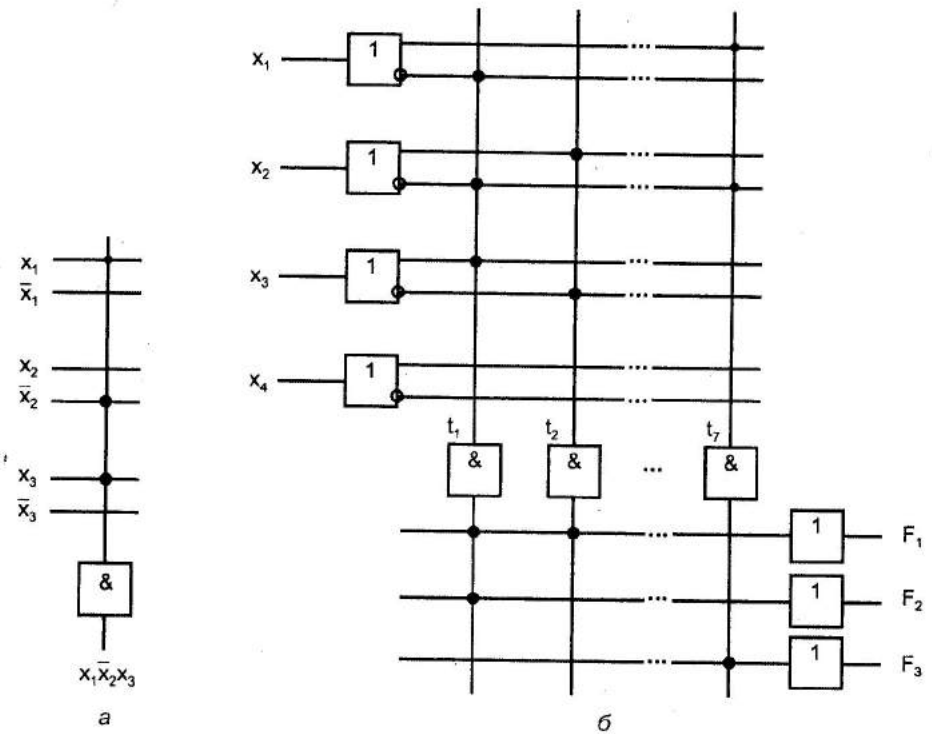


Рис. 8.4. Упрощенное изображение схемы многовходового логического элемента (*а*) и ПЛМ (*б*)

Единственная линия входа таких элементов пересекается с несколькими линиями входных переменных. Если пересечение отмечено точкой, данная переменная подается на вход изображаемого элемента, если точки нет, то переменная на элемент не подается. Пример многовходового конъюнктора с входами $x\bar{x}$ показан на рис. 8.4, *а*. Схема рис. 8.2, *а* в новом упрощенном изображении имеет вид, приведенный на рис. 8.4, *б*.

Воспроизведение скобочных форм переключательных функций

С помощью ПЛМ, как и с помощью рассматриваемых далее ПМЛ, можно воспроизводить не только дизъюнктивные нормальные формы переключательных функций, но и *скобочные формы*.

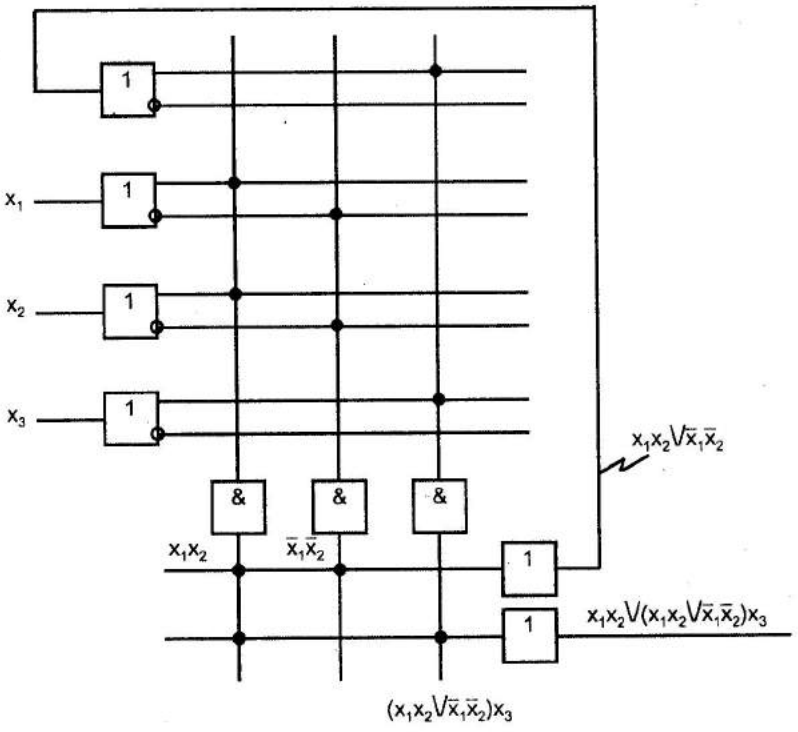


Рис. 8.5. Схема включения ПЛМ при воспроизведении скобочных форм переключательных функций

В этом случае сначала получают выражения в скобках, а затем они рассматриваются как аргументы для получения окончательного результата. В схеме

появляются обратные связи — промежуточные результаты с выхода вновь подаются на входы, логическая глубина схемы увеличивается, задержка выработки результата растет. Пусть, например, требуется получить функцию:

$$F = x_1x_2 \vee (x_1x_2 \vee \bar{x}_2\bar{x}_1)x_3.$$

Для этого следует применить включение ПЛМ по схеме (рис. 8.5).

Наращивание (расширение) ПЛМ

Если размерность задачи превосходит возможности имеющихся ПЛМ, приходится их наращивать. Когда число функций в системе N превосходит число n выходов ПЛМ, несколько ПЛМ включаются параллельно по входам (рис. 8.6). На выходах каждой из ПЛМ воспроизводится часть функций. Общее число ПЛМ определяется как $\lceil N/n \rceil$. Так как число термов предполагается достаточным ($\ell_{сист} < \ell$), все ПЛМ могут быть запрограммированы на одни и те же термы.

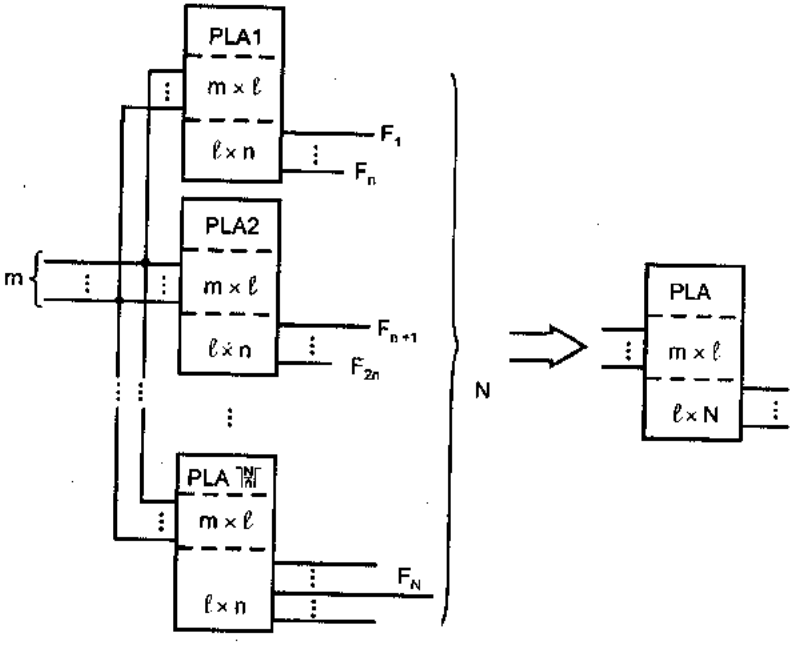


Рис. 8.6. Схема расширения ПЛМ по числу выходов

Если число термов системы $\ell_{сист}$ превышает число термов ПЛМ ($\ell_{сист} > \ell$), то к ПЛМ подключаются дополнительные с тем же числом входов и выходов. По входам ПЛМ включаются параллельно, а соответствующие выходы соединяются по ИЛИ или просто объединяются, если это выходы с третьим состоянием или возможностями монтажной логики. Каждая ПЛМ программируется на свои термы, затем из термов "собираются" на выходах нужные функции (рис. 8.7).

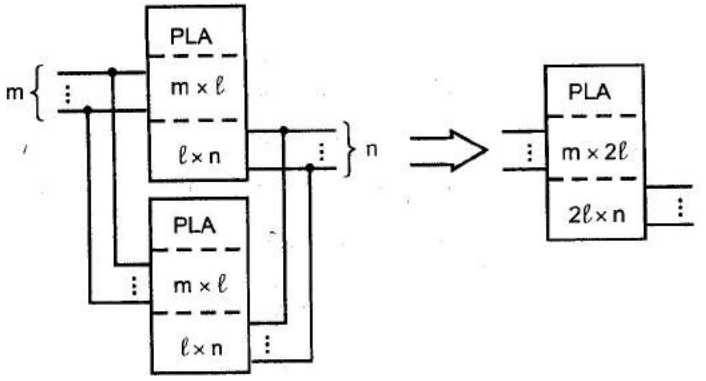


Рис. 8.7. Схема расширения ПЛМ по числу термов

Расширение числа входов — наиболее сложная задача, связанная с декомпозицией системы функций. В частном случае, если все термы содержат не более m переменных, множество термов можно разбить на подмножества, содержащие не более m одинаковых переменных. Для реализации потребуется число ПЛМ, равное числу подмножеств, а выходы ПЛМ будут соединены так же, как и при расширении числа термов. Входными переменными каждой ПЛМ будут только связанные с образованием термов данного подмножества.

Часто в числе входных переменных ПЛМ имеются тактирующие сигналы, взаимно исключающие друг друга в смысле одновременности вхождения в термы. Такие сигналы можно разделить на группы (подмножества), каждая из которых вместе с оставшимися переменными может обрабатываться отдельной ПЛМ (рис. 8.8).

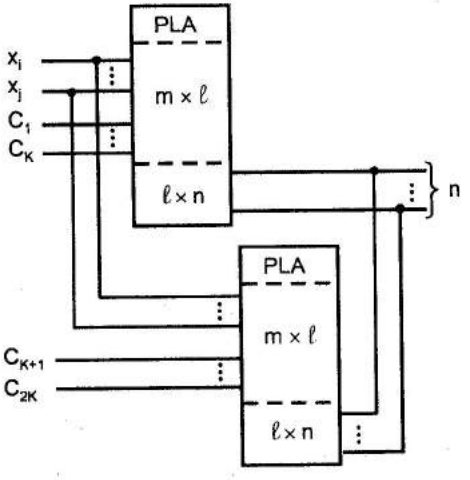


Рис. 8.8. Схема первого варианта расширения ПЛМ по числу входов

Стандартным приемом расширения ПЛМ по входам является перенос избыточного числа аргументов на предварительный дешифратор, выходы которого разрешают работу одной из ПЛМ, обрабатывающих оставшуюся часть аргументов. Этот прием рассматривался ранее применительно к наращиванию дешифраторов и других схем. Расширение числа входов ПЛМ на единицу, произведенное по такому методу, показано на рис. 8.9. Для значительного расширения числа входов этот прием мало пригоден, т. к. избыточные переменные образуют слова, подвергающиеся полной дешифрации, что резко увеличивает число ПЛМ в схеме (удваивает с добавлением каждого избыточного входа).

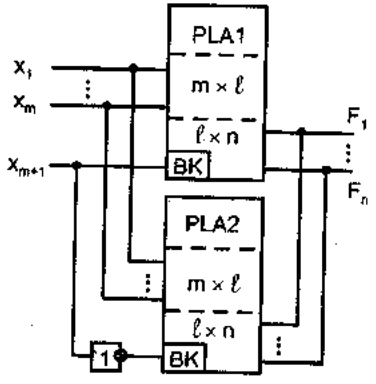


Рис. 8.9. Схема расширения числа входов ПЛМ на единицу

Простая эвристическая методика расширения ПЛМ по входам, не претендующая на оптимальность, предложена в работах Е. И. Пупырева и описана также в [45].

Первые отечественные ПЛМ были выпущены в составе серии К556 (микросхемы РТ1, РТ2 схемотехнологии ТТЛШ с программированием прожиганием перемычек). Их размерность 16 входов, 48 термов, 8 выходов, задержка около 50 нс. Микросхема РТ1 имеет выходы с открытым коллектором. Микросхема РТ2 имеет выходы с тремя состояниями.

Структура ПМЛ

Одно из важных применений БИС программируемой логики — замена ИС малого и среднего уровня интеграции при реализации так называемой произвольной логики. В этих применениях логическая мощность ПЛМ зачастую используется неполно. Это проявляется, в частности, при воспроизведении типичных для практики систем переключательных функций, не имеющих больших пересечений друг с другом по одинаковым термам. В таких случаях возможность использования выходов любых конъюнкторов любыми дизъюнкторами (как предусмотрено в ПЛМ) становится излишним

усложнением. Отказ от этой возможности означает отказ от программирования матрицы ИЛИ и приводит к структуре ПМЛ (PAL, GAL).

В ПМЛ (рис. 8.10) выходы элементов И (выходы первой матрицы) жестко распределены между элементами ИЛИ (входами матрицы ИЛИ). В показанной ПМЛ m входов, n выходов и 4n элементов И, поскольку каждому элементу ИЛИ придается по четыре конъюнктора ПМЛ, как и ПЛМ, воспроизводят дизъюнктивные нормальные формы логических функций, но с более жесткими ограничениями.

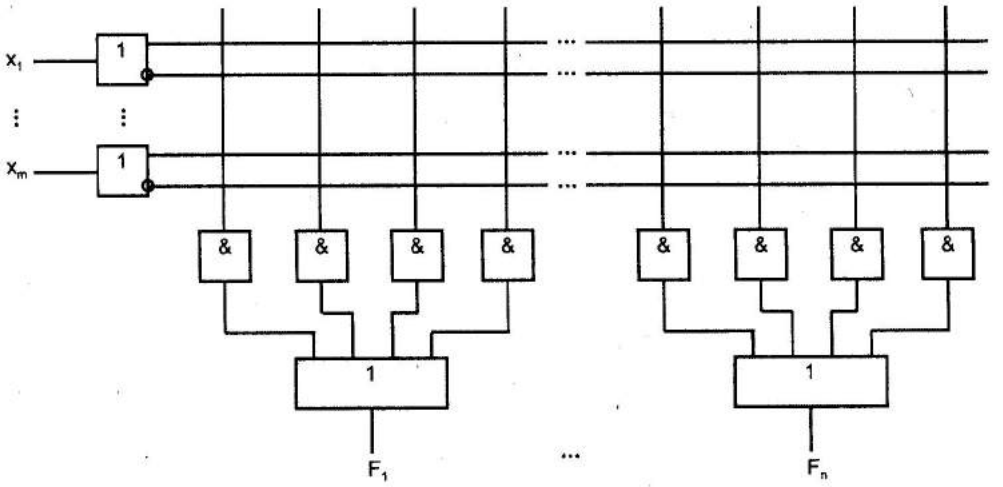


Рис. 8.10. Базовая структура ПМЛ

В сравнении с ПЛМ схемы ПМЛ имеют меньшую функциональную гибкость, т. к. в них матрица ИЛИ фиксирована, но их изготовление и использование проще. Преимущества ПМЛ особенно проявляются при проектировании несложных устройств.

Подготовка задач к решению на ПМЛ имеет много общего с подходом к решению задач на ПЛМ, но есть и различия. Для ПМЛ важно уменьшить число элементов И для каждого выхода, но если для ПЛМ стремятся искать представление функции с наибольшим числом общих термов, то для ПМЛ это не требуется, поскольку элементы И фиксированы по своим выходам и не могут быть использованы другими выходами (т. е. для других функций).

Обогащение функциональных возможностей ПЛМ и ПМЛ

Рассмотренные структуры ПЛМ и ПМЛ — базовые, с которых началось развитие этих направлений. В дальнейшем происходило обогащение функцио-

нальных возможностей ПЛМ и ПМЛ с помощью ряда приемов, в первую очередь следующих.

Программирование выходных буферов. В схемах с программируемым выходным буфером обеспечивается возможность выдачи выходных функций в прямом или инверсном виде. В такой схеме (рис. 8.11) выработанные матрицами функции F_1^*—F_n^* проходят через выходной буфер, разрядные схемы которого выполнены как сумматоры по модулю 2.

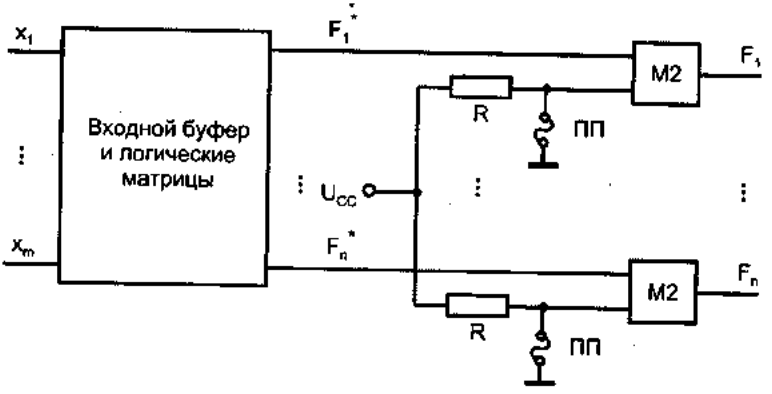


Рис. 8.11. Схема программируемого выходного буфера

В показанной на рисунке схеме вторые входы сумматоров получают нулевые сигналы от потенциала "земли" через плавкие перемычки ПП. При этом $F_i^* = F_i$ и функции с выхода матриц передаются через буфер без изменений. Если пережечь перемычку у нижнего входа сумматора, то он получит сигнал логической единицы от источника питания через резистор R. Складываясь по модулю 2 с единицей, функции F_i^* инвертируются. Следовательно, в линиях с целыми перемычками функции проходят через буфер неизменными, а в линиях с отсутствующими перемычками — инвертируются.

Программируемый буфер дает дополнительные возможности для минимизации числа термов в реализуемой системе. В исходной системе можно заменять функции их инверсиями, если это приводит к уменьшению числа термов. Никаких последствий в смысле введения дополнительных схем это не вызовет — возврат к исходной системе будет обеспечен просто программированием буфера.

Пример

Пусть нужно воспроизвести систему из двух функций:

$$F_1 = \bar{x}_3\bar{x}_2\bar{x}_0 \vee \bar{x}_3x_2x_0 \vee x_3\bar{x}_1x_0 \vee x_3x_1\bar{x}_0;$$
$$F_2 = \bar{x}_3\bar{x}_2x_0 \vee \bar{x}_3x_2\bar{x}_1 \vee x_3\bar{x}_1\bar{x}_0 \vee x_3x_1x_0.$$

Карты Карно для этих функций (рис. 8.12) показывают контуры, соответствующие восьми различным термам системы:

$$t_1 = \bar{x}_3\bar{x}_2\bar{x}_0,\ t_2 = \bar{x}_3x_2x_0,\ t_3 = x_3\bar{x}_1x_0,\ t_4 = x_3x_1\bar{x}_0;$$
$$t_5 = \bar{x}_3\bar{x}_2x_0,\ t_6 = \bar{x}_3x_2\bar{x}_1,\ t_7 = x_3\bar{x}_1\bar{x}_0,\ t_8 = x_3x_1x_0.$$

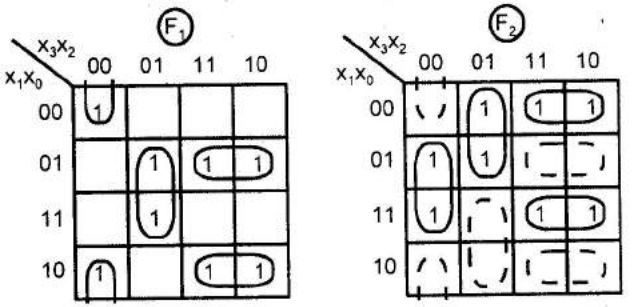


Рис. 8.12. Карты Карно для примера воспроизведения функций в ПЛМ с программируемым выходным буфером

При инвертировании функции единицы занимают в карте Карно те позиции, которые были нулями. Видно, что при инвертировании одной из функций получим карты Карно с меньшим количеством различных термов. При инвертировании, например, функции F_2 получим карту с контурами, показанными штриховыми линиями, и систему функций:

$$F_1 = \bar{x}_3\bar{x}_2\bar{x}_0 \vee \bar{x}_3x_2x_0 \vee x_3\bar{x}_1x_0 \vee x_3x_1\bar{x}_0 = t_1 \vee t_2 \vee t_3 \vee t_4,$$
$$\overline{F}_2 = \bar{x}_3\bar{x}_2\bar{x}_0 \vee \bar{x}_3x_2x_1 \vee x_3\bar{x}_1x_0 \vee x_3x_1\bar{x}_0 = t_1 \vee t_5 \vee t_3 \vee t_4,$$

в которой всего пять различных термов. Возврат от функции $\overline{F}_2$ к функции F_2 осуществляется пережиганием перемычки в линии выхода F_2.

Применение двунаправленных выводов. Используя элементы с тремя состояниями выхода, можно построить схему, в которой некоторые выводы можно приспосабливать для работы в качестве входов или выходов в зависимости от программирования перемычек. В такой схеме один из конъюнкторов предназначен для управления элементом с тремя состояниями выхода (рис. 8.13). Выход элемента одновременно связан с матрицей И как вход.

Возможны четыре режима вывода Вх/Вых в зависимости от того, как запрограммированы входы конъюнктора К:

1. Все перемычки нетронуты. В этом режиме на выходе конъюнктора К будет нуль, буфер имеет третье состояние выхода и вывод функционирует как вход.

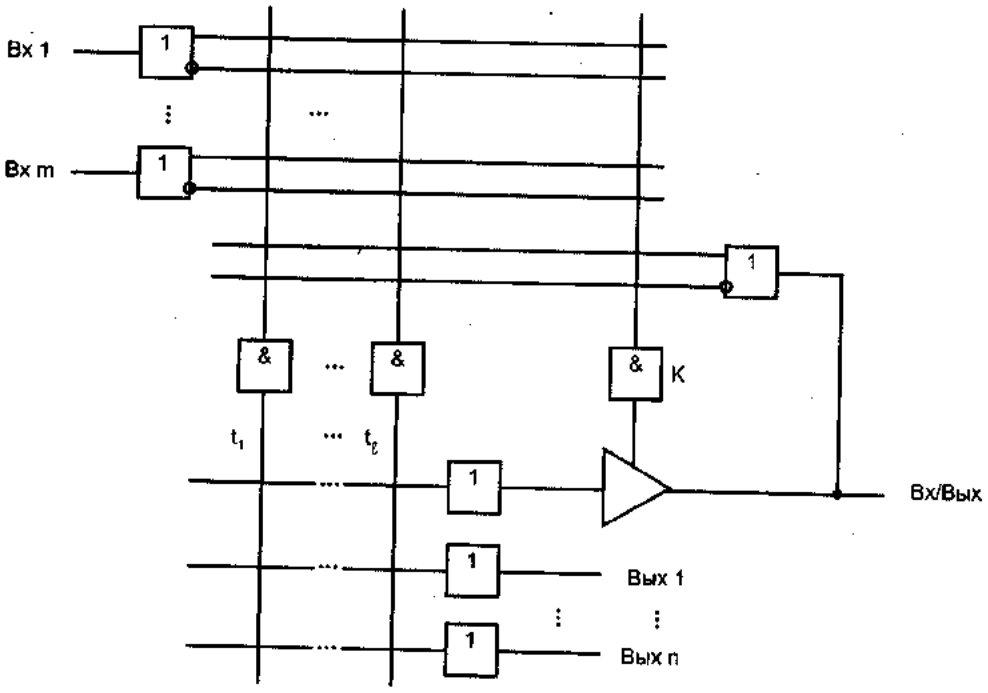


Рис. 8.13. Схема с двунаправленным буфером

2. Все перемычки пережжены, на выходе конъюнктора единица, буфер активен, вывод работает как выход (его сигналы не используются в матрице И).
3. Выход с обратной связью. Этот режим отличается от предыдущего только тем, что сигналы вывода используются в матрице И.
4. Управляемый выход. Здесь входы конъюнктора программируются. При заданной комбинации входных сигналов конъюнктор приобретает единичный выход, и вывод срабатывает как выход.

В схеме с некоторым числом двунаправленных выводов можно изменять соотношение числа входов/выходов. Если число входов равно m, число выходов n и число двусторонних выводов p, то можно иметь число входов от m до m + p и число выходов от n до n + p при условии, что сумма числа входов и выходов не превосходит m + n + p.

Введение элементов памяти. Схемы с элементами памяти позволяют строить автоматы наиболее удобным способом, т. к. помимо комбинационной части содержат на кристалле триггеры (регистры) (рис. 8.14) обычно типа D. ПЛМ с памятью характеризуется четырьмя параметрами. Кроме трех обычных параметров, она имеет и параметр r — число элементов памяти (разрядов регистра). Структура ПЛМ (см. рис. 8.14) совпадает с канониче-

ской схемой автомата. Результат данного шага обработки информации зависит в ней от результатов предыдущих шагов, что обеспечивается обратной связью с регистра на вход ПЛМ. Максимальное число внутренних состояний автомата 2^r. Автомат рассматривается как синхронный — петля обратной связи активизируется только по разрешению тактовых сигналов ТС.

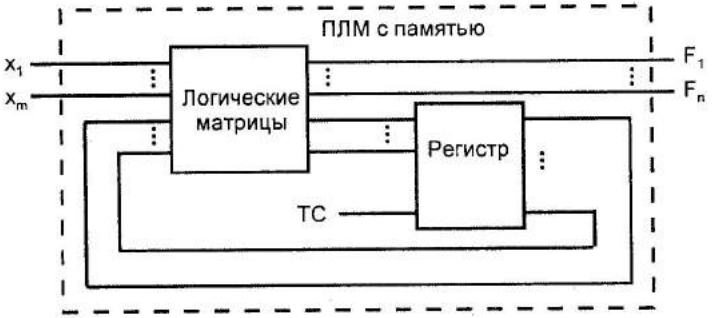


Рис. 8.14. Структура ПЛМ с памятью

Использование разделяемых конъюнкторов в схемах ПМЛ. Наряду с пригодными как для ПЛМ, так и для ПМЛ методами обогащения функциональных возможностей, рассмотренными ранее, существуют и специфические модификации, относящиеся только к ПМЛ. К ним относится вариант с так называемыми разделяемыми конъюнкторами. "Разделяемость" здесь означает "совместное использование", при котором одни и те же конъюнкторы могут быть отданы тому или иному выходу схемы. Прием "разделения конъюнкторов" в его простейшей форме состоит в следующем. Для двух смежных элементов ИЛИ отводится некоторое количество конъюнкторов (например, 16), которое может быть произвольно разделено между этими смежными элементами. Другие элементы ИЛИ использовать данный набор конъюнкторов не могут. Полного программирования матрицы ИЛИ здесь не возникает, но все же эта модификация является *шагом в направлении к ПЛМ.*

Вариант с разделяемыми конъюнкторами смягчает наиболее очевидное ограничение функциональных возможностей простых (жестких) ПМЛ — фиксированное число элементов И на входах элементов ИЛИ, которого может не хватить при воспроизведении сложных функций. Имея ПМЛ с разделяемыми конъюнкторами и размещая сложную функцию рядом с простой, можно позаимствовать часть общего набора конъюнкторов у простой функции в пользу сложной.

Вариант схемотехнической реализации разделяемости конъюнкторов, примененный в одной из промышленных ПМЛ, показан на рис. 8.15. В ПМЛ имеется дополнительный набор элементов ИЛИ и сложения по модулю 2

("исключающее ИЛИ"), с помощью которого можно комбинировать сигналы выходов обеих основных схем ИЛИ для образования окончательных значений функций F_1 и F_2. Выходы основных схем ИЛИ могут объединяться по операциям дизъюнкции или сложения по модулю 2 и распределяться по основным выходам F_1 и F_2. Операция сложения по модулю 2 дает дополнительные функциональные возможности. Характер получаемых функций зависит от того, какой из трех транзисторов в показанных двух группах будет проводящим.

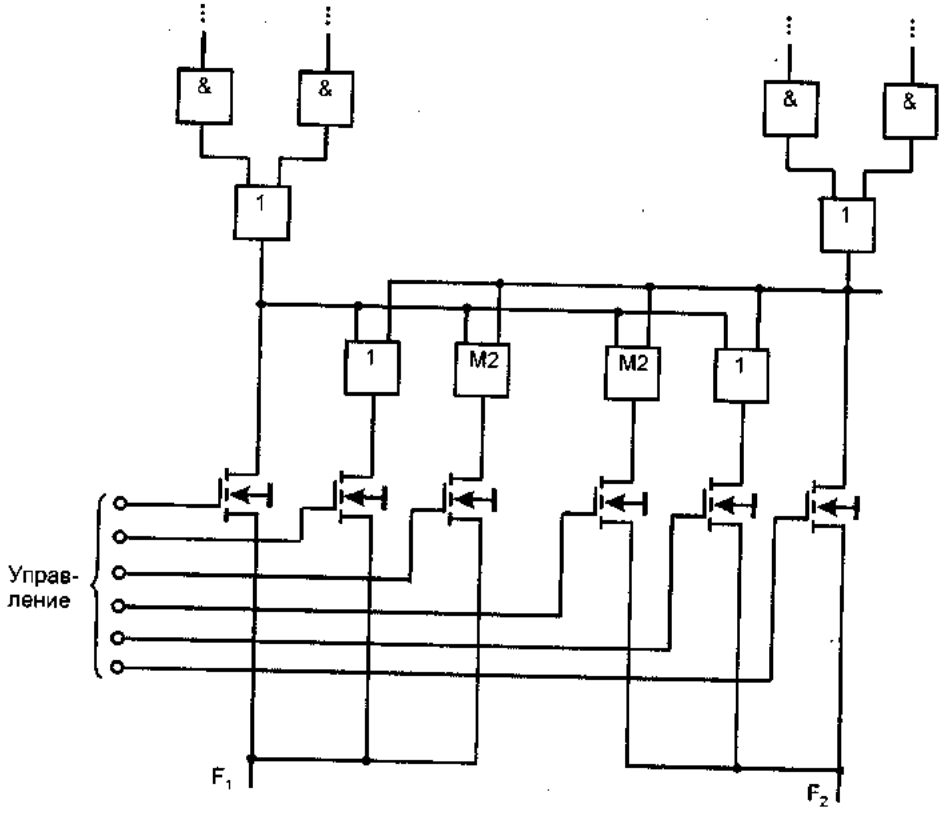


Рис. 8.15. Пример реализации разделения термов в ПМЛ

Примеры промышленных ПМЛ (серии К1556)

Первые отечественные ПМЛ появились в серии КР1556 (микросхема ХЛ8, за которой последовали ИС ХП4, ХП6, ХП8). Буквой Л в обозначении от-

мечаются ПМЛ чисто логического типа (без элементов памяти), а буквой П — ПМЛ с триггерами. Микросхема ХЛ8 (ПМЛ с двунаправленными выводами) показана на рис. 8.16. Число входов может изменяться от 10 (входы, показанные с левой стороны матрицы) до 16, если все двунаправленные выводы В2—В7 запрограммированы как входы. Число выходов изменяется от 2 до 8. Суммарное число входов и выходов не может превышать 18.

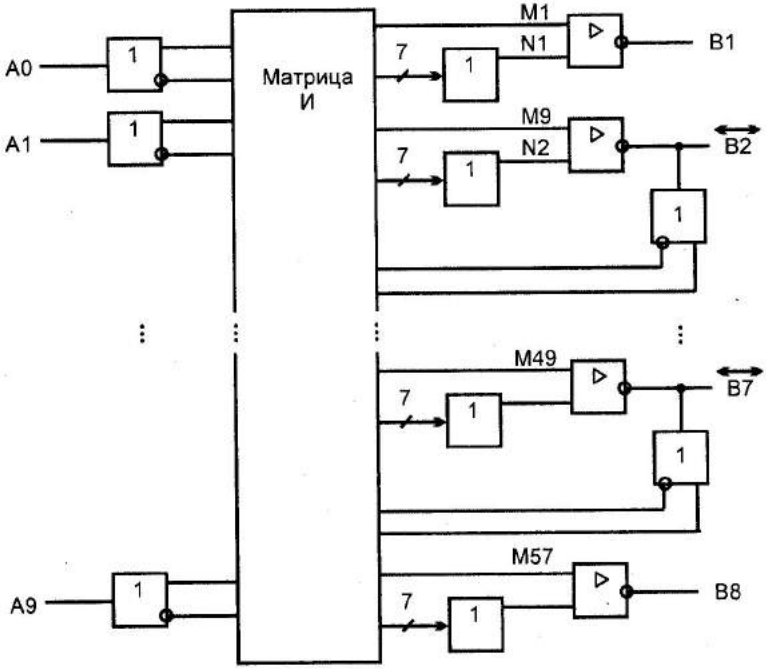


Рис. 8.16. Структура ПМЛ КР1556ХЛ8

Выходные буферы ПМЛ получают разрешение или запрещение работы от матрицы И, как было рассмотрено в предыдущем параграфе. Набор элементов ИЛИ состоит из восьми элементов с семью входами, т. е. на каждый элемент ИЛИ приходится по семь конъюнкторов с числом входов от 10 до 16. Исходя из сказанного, можно оценить и размерность матрицы И, содержащей 2048 узлов (64 × 32).

В микросхемах типа ХП имеются элементы памяти — триггеры типа D, число которых совпадает с цифрой в обозначении ИС (4, 6 или 8). Структура ИС ХП4 (рис. 8.17) имеет первый уровень логики, на котором образуются термы входных переменных, второй уровень — матрица ИЛИ, состоящая из восьми дизъюнкторов (четырех семивходовых и четырех восьмивходовых).

Выходные усилители выполнены по схеме с тремя состояниями. Четыре D-триггера имеют управление от положительного фронта внешнего синхросигнала C. Сигнал OE управляет буферами, подключенными к выходам триггеров.

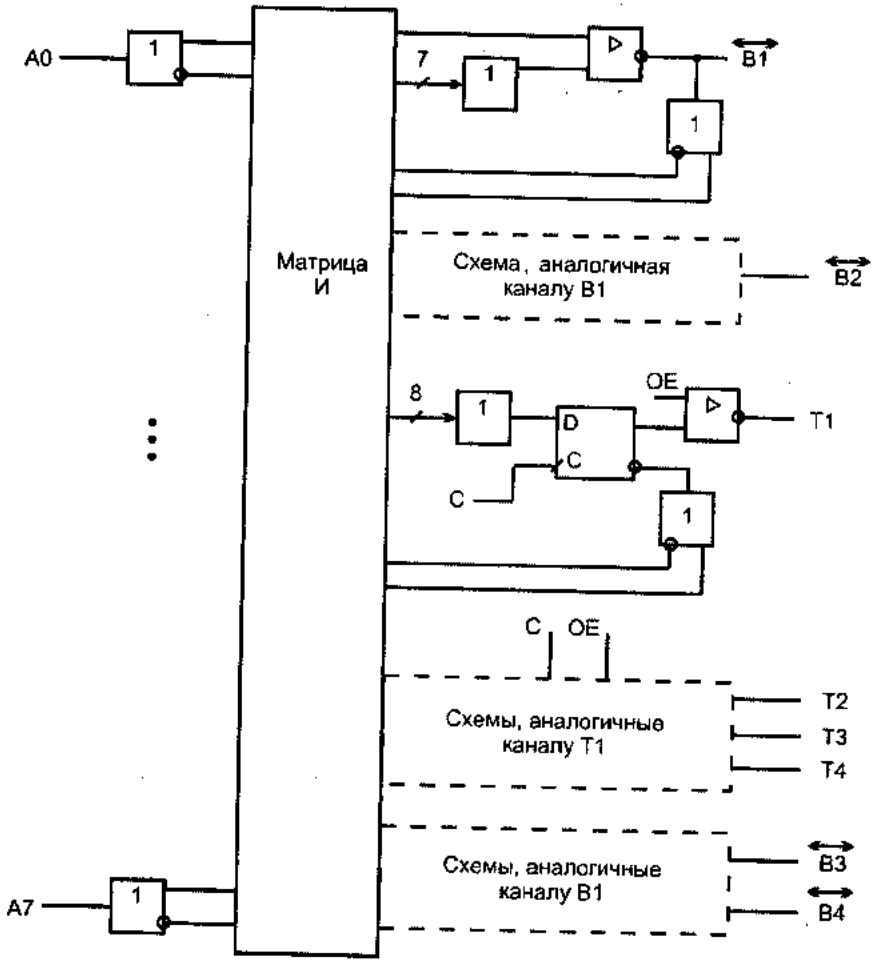


Рис. 8.17. Структура ПМЛ КР1556ХП4

Число входов у ПМЛ типа ХП — 8, число выходов 8(4), 8(2) и 8 для ХП4, ХП6 и ХП8 соответственно, задержка между выводами вход/выход не более 40 нс, а между тактовым сигналом и выходом не более 25 нс. Потребление тока — 180 мА.

Пример подготовки задачи к решению с помощью ПМЛ

Пусть на ПМЛ КР1556ХП4 требуется реализовать четырех разрядный синхронный счетчик, выполняющий помимо операции счета также операцию параллельной асинхронной загрузки. Для реализации устройства на основе ПЛМ или ПМЛ его функции нужно определить как систему переключательных функций. *В § 3.8 рассмотрен синхронный счетчик, построенный на триггерах типа JK*, здесь же в нашем распоряжении имеются триггеры типа D, соответственно видоизменяются и функции возбуждения триггеров.

Обозначим выходы разрядов счетчика, начиная с младшего, через Q_0, Q_1, Q_2 и Q_3. Сигнал асинхронной загрузки обозначим как LE (Load Enable). Загружаемое слово — $A_3A_2A_1A_0$.

Триггер младшего разряда счетчика переключается от каждого входного сигнала при отсутствии сигнала загрузки и принимает значение A_0 при загрузке. Следовательно, для его выхода в новом состоянии можно записать

$$Q_{0H} = \overline{LE}\,\overline{Q}_0 \vee LE\, A_0,$$

где первое слагаемое отображает процесс переключения триггера, а второе — параллельную загрузку.

Следующий разряд переключается только при условиях отсутствия сигнала загрузки и единичном состоянии триггера младшего разряда. При $Q_0 = 0$ этот триггер сохраняет свое состояние. Для его выхода можно записать:

$$Q_{1H} = \overline{LE}\,\overline{Q}_1Q_0 \vee \overline{LE}\,Q_1\overline{Q}_0 \vee LE\, A_1,$$

где первое слагаемое отображает переключение триггера, второе — сохранение его состояния при $Q_0 = 0$, третье — загрузку.

Продолжая аналогичные рассуждения, для последующих разрядов счетчика можно получить соотношения:

$$Q_{2H} = \overline{LE}\,\overline{Q}_2Q_1Q_0 \vee \overline{LE}\,Q_2\overline{Q}_1 \vee \overline{LE}\,Q_2\overline{Q}_0 \vee LE\, A_2;$$
$$Q_{3H} = \overline{LE}\,\overline{Q}_3Q_2Q_1Q_0 \vee \overline{LE}\,Q_3\overline{Q}_2 \vee \overline{LE}\,Q_3\overline{Q}_1 \vee \overline{LE}\,Q_3\overline{Q}_0 \vee LE\, A_3,$$

где первые слагаемые отображают процесс переключения разряда, последние — параллельную загрузку, а промежуточные — сохранение состояния при отсутствии условий переключения.

Поскольку искомые функции содержат не более пяти конъюнкций, возможна их непосредственная реализация на микросхеме ХП4 (в этой микросхеме число элементов И на входах элементов ИЛИ составляет 7 или 8 для разных выходов).

При проектировании устройств на основе ПЛМ и ПМЛ пользуются подсистемами автоматизации проектирования, т. к. ручная подготовка задачи, как правило, неприемлемо громоздка. Для подсистемы автоматизированного проектирования подготовка данных проводится с использованием входного языка таблиц или систем булевых уравнений, записанных в предусмотренной языком форме. Данные для программатора, пережигающего перемычки, получаются автоматически.

В подобных подсистемах имеются также режимы входного контроля ИС, проверяющего целостность перемычек ИС до программирования; ввода данных с эталона, т. е. уже запрограммированной ИС, установленной в специальную соединительную розетку; сравнения данных о программировании, находящихся в памяти подсистемы, с состоянием перемычек ИС и др.

Примеры зарубежных ПМЛ с усложненными макроячейками

PAL 22V10. В зарубежной схемотехнике ПМЛ получили широкое распространение. Примером может служить микросхема PAL 22V10 (буква V появилась от слова Versatile — гибкий, подвижный). У этой микросхемы 10 выходов, различающихся числом подключенных к ним конъюнкторов.

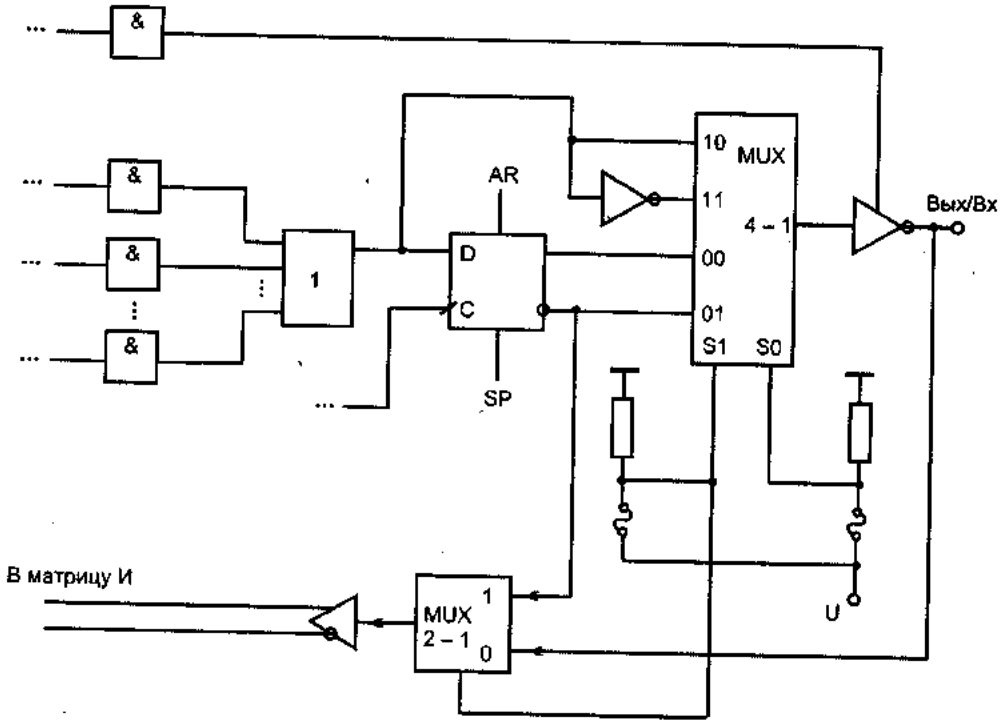


Рис. 8.18. Схема макроэлемента PAL 22V10

Разные выходы имеют от 8 до 16 конъюнкторов. Выходные величины вырабатываются не просто дизъюнкторами, а более сложными схемами, называемыми макроэлементами (макроячейками).

Схема макроэлемента PAL 22V10 содержит триггер типа D с цепями тактирования, асинхронного сброса AR и синхронной установки SP[1] (рис. 8.18). Мультиплексор MUX "4—1" работает на выходной буфер, мультиплексор MUX "2—1" передает сигналы обратной связи в матрицу И. Цепи с плавкими перемычками программируют мультиплексоры. На вход мультиплексора MUX "4—1" подаются прямой и инверсный сигналы от логической части ПМЛ, а также регистровый выход (с триггера) и его инверсия. Сигнал обратной связи можно взять с выхода ПМЛ или с выхода триггера. При установке выходного буфера в третье состояние внешний вывод может быть использован как вход. Таким образом, любой из 10 программируемых выходов может быть либо входом, либо комбинационным или регистровым выходом при H-активности или L-активности выходного сигнала.

Макроячейки, подобные макроэлементам микросхемы 22V10, имеются также в широко известных схемах типа GAL фирмы Lattice Semiconductor.

Пример более сложной структуры PLD

На рис. 8.19 показана структура БИС (матрица И и один из 12 макроэлементов, здесь термин "макроэлемент" равноценен термину "макроячейка"), интересная тем, что, будучи простой, сочетает в себе, тем не менее, несколько типичных для PLD приемов повышения функциональной гибкости: возможность разделения термов между соседними макроэлементами, программируемость полярности вырабатываемой логической функции (реализуемость F или $\overline{F}_1$), программируемость типа триггера (D или T), возможность выбора комбинационного или регистрового выхода, двунаправленность внешнего вывода.

Единая матрица И имеет 32 входа для подачи входных переменных и два входа для подачи сигналов обратной связи с выхода мультиплексора и использования внешнего вывода макроячейки в качестве входа при установке выходного буфера в третье состояние.

Конъюнкторы, имеющие по 68 входов, вырабатывают термы, которые подаются на элементы ИЛИ (по четыре терма на каждый из двух элементов ИЛИ). Столбец из четырех программируемых мультиплексоров реализует разделяемость термов, позволяя данному макроэлементу не только использовать термы от своих конъюнкторов, но и получать термы от соседних макроэлементов (при программировании мультиплексоров 1 и 4 на передачу данных от верхних входов) или отдавать свои термы соседям с выходов четырехвходовых дизъюнкторов.

---

[1] Сигналы AR и SP вырабатываются специальными термами матрицы И.

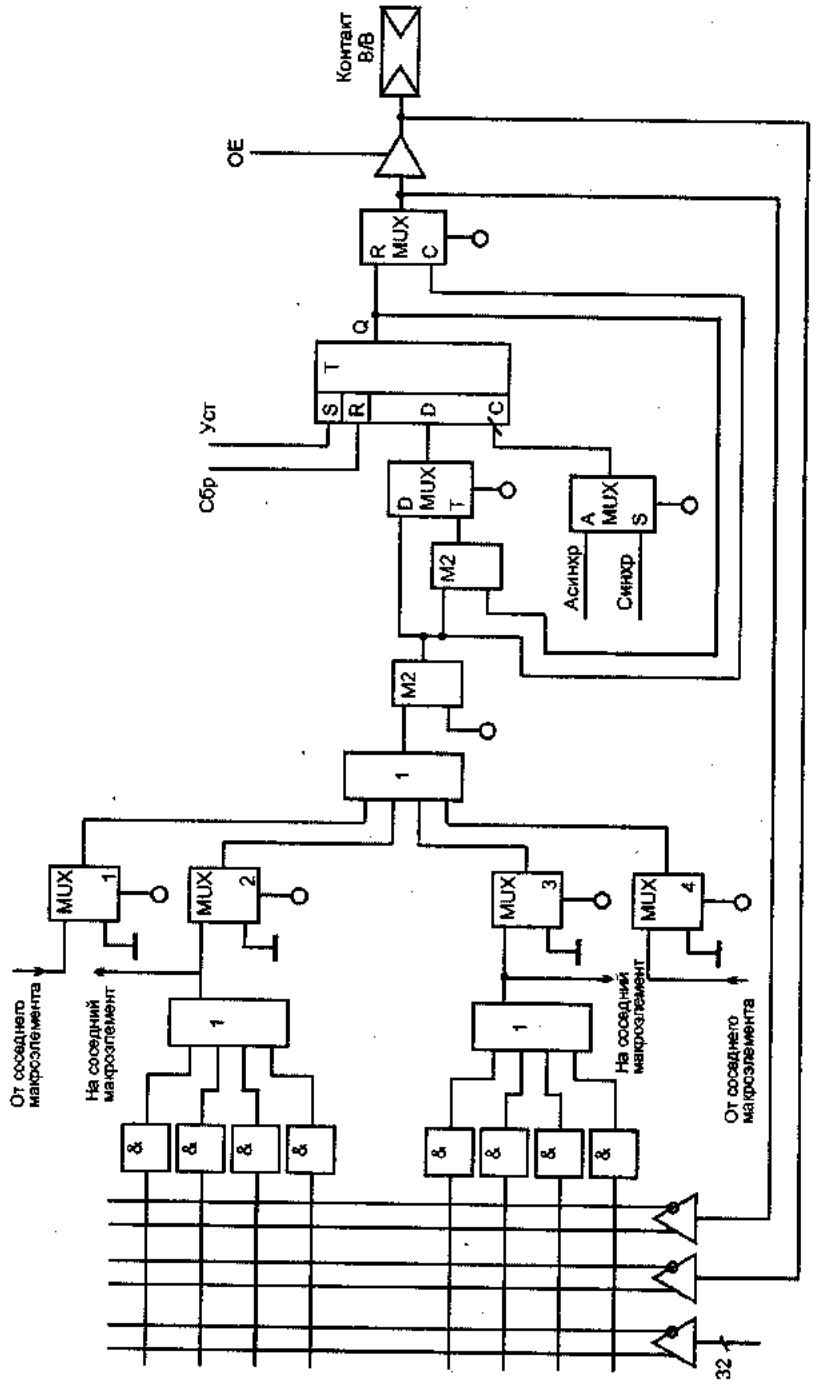


Рис. 8.19. Структура БИС типичной PLD

Окончательный набор термов формируется дизъюнктором, на входы которого поступают выходные сигналы мультиплексоров столбца. Программирование мультиплексора на передачу данных от нижнего входа исключает поступающие на него термы из формируемого набора.

Выработанная логическая функция передается в дальнейшие части макроэлемента через сумматор по модулю 2, на второй вход которого при программировании может быть подана логическая единица или логический нуль. В первом случае, проходя через элемент М2, функция инвертируется, во втором — не изменяется. Выход элемента М2 подключен к мультиплексору, информационные входы которого помечены буквами D и T. Если этот мультиплексор запрограммирован на передачу сигнала от входа D, то триггер просто получает сигнал и функционирует как триггер типа D. Если же мультиплексор запрограммирован на передачу сигнала от входа T, то триггер через обычный элемент сложения по модулю 2 замкнут в петлю обратной связи. При этом нулевое значение сигнала на верхнем входе обычного (не программируемого) элемента М2 обеспечивает передачу на вход триггера сигнала его текущего состояния Q, т. е. при поступлении тактового импульса состояние триггера сохранится. Единичное значение сигнала на верхнем входе элемента М2 приводит к сложению величины Q с единицей по модулю 2, т. е. к подаче на вход триггера через мультиплексор величины $\overline{Q}$, что ведет к переключению триггера. Видно, что в этом случае триггер работает как синхронный триггер типа T, причем роль сигнала T играет сигнал на выходе программируемого элемента М2.

Мультиплексор, информационные входы которого помечены буквами R (от *Registered*) и C (от *Combinatorial*), осуществляет в зависимости от программирования выбор типа выхода макроэлемента — в виде запоминаемого сигнала Q от триггера или непосредственно комбинационной функции по линии C, идущей в обход триггера. Через буфер с тремя состояниями выход макроэлемента связан с контактной площадкой (внешним выводом). Если буфер находится в третьем состоянии, контакт может использоваться как входной, с которого сигнал поступает в матрицу И.

Для управления триггером можно выбрать с помощью мультиплексора со входами A и S либо синхронный вариант (т. е. тактирование общим синхросигналом всей микросхемы), либо асинхронный (т. е. с выработкой сигнала тактирования от отдельного терма, иначе говоря, разрешением принятия информации при появлении определенной комбинации входных сигналов матрицы И).

Микросхема реализована по КМОП технологии, ее сложность оценивается числом 1800 эквивалентных вентилей. Двенадцать макроэлементов за счет комбинирования своих термов с термами соседних макроэлементов позволяют получать логические функции от 4, 8, 12 либо 16 термов.

§ 8.3. Базовые матричные кристаллы (вентильные матрицы с масочным программированием)

Основные сведения

Первые образцы базовых матричных кристаллов (БМК) появились в 1975 г. как средство реализации нестандартных схем высокопроизводительной ЭВМ без применения микросхем малого и среднего уровней интеграции. Разработка БМК позволила выполнить и нетиповые части машины на БИС. Формулировку "позволила выполнить" в данном случае следует понимать с позиций экономических факторов. Технологически можно было реализовать любую схему по индивидуальному заказу в виде БИС, однако стоимость таких БИС была бы неприемлемо высока.

Стоимость проектирования БИС/СБИС велика и достигает десятков или даже сотен миллионов долларов. Ясно, что производство БИС/СБИС становится рентабельным только при достаточно большом объеме их потребления, чего нет при разработке нестандартных частей конкретных систем.

Хорошее решение было найдено на путях разработки БИС, функционирование которых может быть приспособлено к решению той или иной задачи на заключительных этапах их производства. При этом полуфабрикаты производятся в массовом количестве без ориентации на конкретного заказчика. Придание полуфабрикатам индивидуального характера лишь на заключительных стадиях производства БИС/СБИС обходится значительно дешевле и требует значительно меньшего времени на проектирование. Такие БИС/СБИС называют *полузаказными* в отличие от полностью заказных.

Развитие полузаказных БИС/СБИС привело к появлению ряда их разновидностей. Применительно к БМК это *канальные, бесканальные и блочные архитектуры.*

> Прежде чем подробнее остановиться на рассмотрении перечисленных вариантов, уточним терминологию. Термин БМК характерен для литературы на русском языке и поэтому используется здесь наиболее часто. В английской терминологии принят термин GA (Gate Array), чему соответствует русский термин — вентильная матрица. В силу тенденции к единообразию терминов "вентильная матрица" предпочтительнее и, видимо, со временем станет основным обозначением данного типа БИС/СБИС.

Основа БМК первого поколения — совокупность регулярно расположенных на кристалле базовых ячеек (БЯ), между которыми имеются свободные зоны для создания соединений (каналы). Эта архитектура называется *канальной.* Базовые ячейки занимают внутреннюю область БМК, в которой

они расположены по строкам и столбцам, и содержат группы нескоммутированных схемных компонентов (транзисторов, резисторов и др.). В периферийной области кристалла размещены ячейки ввода/вывода, набор схемных компонентов которых ориентирован на реализацию связей БМК с внешними цепями.

Таким образом, БМК является заготовкой, которая преобразуется в требуемую схему выполнением необходимых соединений. Потребитель может реализовать на основе БМК некоторое множество устройств определенного класса, задав для кристалла тот или иной вариант рисунка межсоединений компонентов.

Первые БМК (фирмы Amdahl Corp., США) выполнялись по схемотехнике ЭСЛ, для которой полный процесс изготовления включал 13 операций с фотошаблонами. Для изготовления схемы на основе БМК (такие схемы называют МАБИС или БИСМ) требуются только три индивидуальных (переменных) шаблона для задания рисунка межсоединений. Соответственно этому сроки и стоимость проектирования МАБИС в 3—5 раз меньше, чем для полностью заказных БИС/СБИС.

Плата за сокращение сроков и стоимости проектирования — неоптимальность результата. МАБИС проигрывают по площади кристалла и быстродействию полностью заказным схемам, т. к. часть их элементов оказывается избыточной (не используется в данной схеме), взаимное расположение элементов и пути межсоединений не являются наилучшими и т. д.

Промышленное производство БМК широко развернулось с начала 1980-х гг. Применяются схемотехнологии КМОП, ТТЛШ, ЭСЛ и др. В настоящее время уровень интеграции БМК достиг десятков миллионов вентилей на кристалле.

При проектировании БМК стремятся наилучшим образом сбалансировать число базовых ячеек, трассировочные ресурсы кристалла и число контактных площадок для подключения внешних выводов. Неудачные соотношения между указанными параметрами могут существенно ограничивать полноту использования ресурсов кристалла при построении МАБИС.

Трассировочная способность БМК определяется прежде всего площадью, отводимой для межэлементных связей в ортогональных направлениях, и числом слоев межсоединений. Недостаточная трассировочная способность приводит к уменьшению числа задействованных при построении МАБИС базовых ячеек. Избыточная трассировочная способность ведет к нерациональному использованию площади кристалла, что понижает уровень интеграции БМК и повышает его стоимость. Примерно то же можно сказать и о числе внешних выводов БМК. Для современных БМК может потребоваться до 500—1000 внешних выводов. При проектировании БМК требуемые трассировочная способность и число внешних выводов рассчитываются по эмпирическим формулам, основанным на статистических данных, полученных

из опыта построения систем различного назначения. Эта работа выполняется до изготовления БМК и в этом смысле не входит в компетенцию системотехника. Системотехник (потребитель) должен иметь представление о существующих БМК, их разновидностях и особенностях, а также о средствах и методике разработки МАБИС.

До описания разновидностей БМК остановимся подробнее на основных понятиях и определениях.

Базовая ячейка (БЯ) уже определялась как некоторый набор схемных компонентов, регулярно повторяющийся на определенной площади кристалла. Этот набор может состоять из нескоммутированных, а также частично скоммутированных компонентов. Базовые ячейки внутренней области БМК именуются *матричными базовыми ячейками* (МБЯ), ячейки периферийной зоны — *периферийными базовыми ячейками* (ПБЯ). Применяются два способа организации ячеек БМК:

- из компонентов МБЯ может быть сформирован один логический элемент, а для реализации более сложных функций используются несколько ячеек;
- из компонентов МБЯ может быть сформирован любой функциональный узел, а состав компонентов ячейки определяется схемой самого сложного узла.

Функциональная ячейка (ФЯ) — функционально законченная схема, реализуемая путем соединения компонентов в пределах одной или нескольких БЯ.

Библиотека функциональных ячеек — совокупность ФЯ, используемых при проектировании МАБИС. Эта библиотека создается при разработке БМК и избавляет проектировщика МАБИС от работы по созданию на кристалле тех или иных типовых подсхем, т. к. предоставляет для их реализации готовые решения. Библиотека содержит большое число (сотни) функциональных элементов, узлов и их частей. Пользуясь библиотекой, проектировщик реализует схемы, работоспособность которых уже проверена, а параметры известны. Работая с библиотекой, он ведет проектирование на функционально-логическом уровне, поскольку проблемы схемотехнического уровня уже решены при создании библиотеки. Библиотечные элементы имеют различную сложность (логические элементы, триггеры, более сложные узлы или их фрагменты). В состав библиотечного элемента могут входить одна или несколько БЯ. Площадь библиотечного элемента кратна площади БЯ. При проектировании МАБИС функциональная схема изготовляемого устройства, как принято говорить, должна быть покрыта элементами библиотеки.

Функциональные элементы библиотеки одного из КМОП БМК показаны в Приложении 2. Таких элементов 24, их сочетания дают около 40 функциональных ячеек. В число функциональных элементов входят обычные инверторы, инверторы с повышенной нагрузочной способностью, элементы ИЛИ-НЕ

и И-НЕ, нескоммутированные сочетания этих элементов с инверторами, элементы двухступенчатой логики и их нескоммутированные сочетания с инверторами, четыре варианта триггеров, мультиплексор 2—1 на основе проходных ключей (ввиду отсутствия стандартного обозначения показан в виде схемы с транзисторами), цепочки ввода сигналов.

Эквивалентный вентиль (ЭВ) — группа элементов БМК, соответствующая возможности реализации логической функции вентиля (обычно это двухвходовой элемент И-НЕ либо ИЛИ-НЕ). Понятие "эквивалентный вентиль" предназначено для оценки логической сложности БМК.

Каналы трассировки — пути на БМК для возможного размещения межсоединений.

Классификация БМК

Классификация БМК показана на рис. 8.20. Первоначальной и, в известной мере, классической является структура канального БМК (рис. 8.21, *а*). Во внутренней (центральной) области такого БМК расположена матрица базовых ячеек 1 и каналы для трассировки 2.

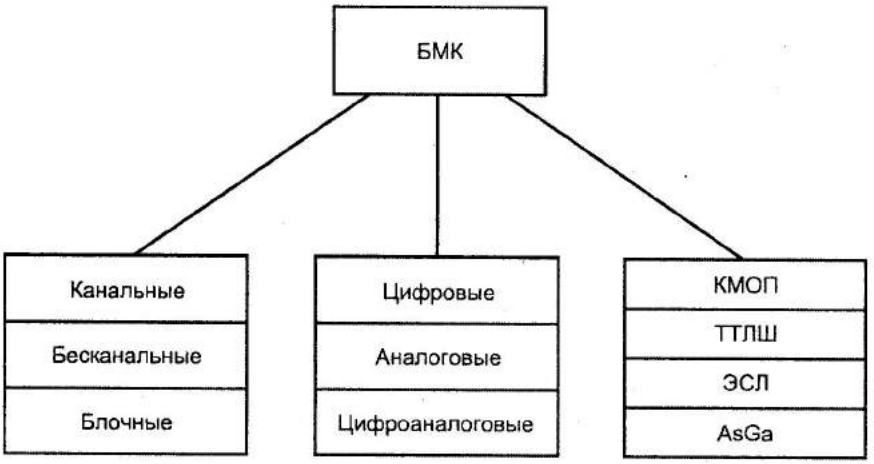


Рис. 8.20. Классификация базовых матричных кристаллов

Каналы могут быть вертикальными и горизонтальными как на рис. 8.21, *а*, либо только вертикальными (рис. 8.21, *б*). Канальные БМК могут иметь большие возможности по созданию связей, но имеют низкую плотность упаковки из-за значительных затрат площади кристалла на области межсоединений.

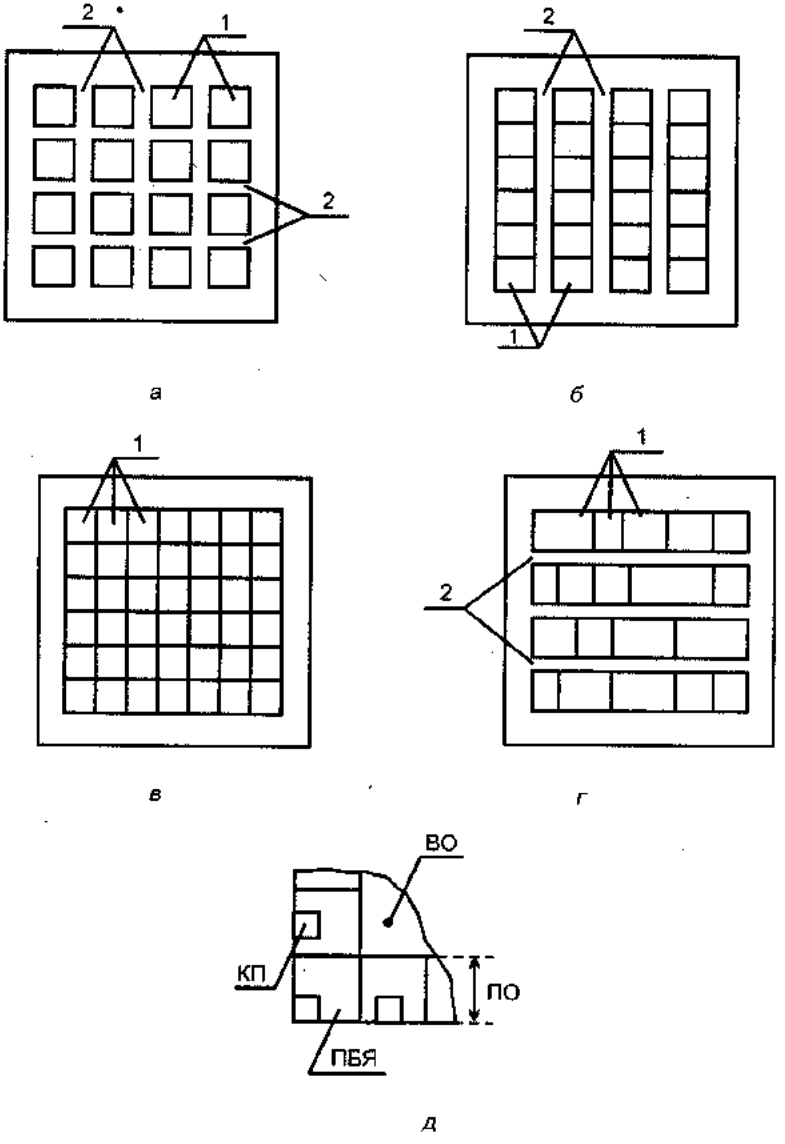


Рис. 8.21. Структуры БМК различных типов (*а*, *б*, *в*, *г*) и расположение областей БМК (*д*)

Канальная архитектура характерна для биполярных БМК, т. к. значительная мощность рассеивания биполярных БЯ сама по себе препятствует плотной их упаковке.

Повышение уровня интеграции БМК ведет к быстрому росту числа необходимых межсоединений между базовыми ячейками, а значит, и площади, отводимой для них. Поиск путей создания БМК высокого уровня интеграции с минимизацией площади, отводимой под межсоединения, привел к *бесканальной архитектуре* БМК (бесканальные архитектуры имеют также названия *"море вентилей" или "море транзисторов"*). Внутренняя область такого БМК содержит плотно упакованные ряды базовых ячеек и не имеет фиксированных каналов для трассировки межсоединений (рис. 8.21, *в*). В этом кристалле любая область, в которой расположены БЯ (строка, столбец либо их часть) может быть использована как для создания логической схемы, так и для создания межсоединений. Вследствие более рационального расположения связей в бесканальном БМК уменьшается и задержка передачи сигналов по связям, т. к. и длины, и паразитные емкости межсоединений уменьшаются.

Бесканальные БМК характерны для КМОП-схемотехники, в которой компактность схемных элементов и малая мощность рассеяния БЯ при их работе на не слишком высоких частотах способствуют возможностям плотной упаковки базовых ячеек.

Так как в бесканальных БМК, называемых иногда универсальными, положение трассировочных каналов и ячеек на рабочем поле не является жестким и при проектировании конкретной МАБИС площадь кристалла может перераспределяться между трассировочными каналами и функциональными ячейками, потери площади кристалла снижаются. Например, в БМК с плотным расположением на рабочем поле рядов транзисторов в некоторых рядах реализуются логические элементы, а другие ряды используются под трассировочные каналы, в них транзисторы остаются нескоммутированными и не используются (над ними проходят трассы). В зависимости от загруженности каналов, для них может быть отведено различное число рядов транзисторов.

В КМОП БМК используются также архитектуры с переменной длиной ячеек (рис. 8.21, *г*). Здесь каждая строка представляет собою последовательное соединение пар n- и p-канальных транзисторов. Если в такой длинной цепи разместить в заданных местах пары запертых транзисторов, то цепочка будет разделена на базовые ячейки произвольной длины. Возможность варьирования длиной БЯ ведет к более рациональному построению МАБИС и, следовательно, к повышению уровня интеграции реализуемых на БМК схем.

Внутренняя область кристалла (ВО) окружена периферийной областью (ПО) (рис. 8.21, *д*), расположенной по краям прямоугольной пластины БМК. В периферийной области расположены специальные ПБЯ, набор схемных элементов которых ориентирован на решение задач ввода/вывода сигналов, а также контактные площадки (КП). Рост уровня интеграции ведет к возможностям реализации на одном кристалле все более сложных устройств и систем. Это вызвало к жизни *блочные структуры* БМК, архитектура которых

упрощает построение комбинированных устройств, содержащих как блоки логической обработки данных, так и память или другие специализированные блоки. При этом в БМК реализуются несколько блоков-подматриц, каждый из которых имеет как бы структуру БМК меньшей размерности. Между блоками располагаются трассировочные каналы (рис. 8.22). На периферии блоков изготовляются внутренние буферные каскады для формирования достаточно мощных сигналов, обеспечивающих передачу сигналов по межблочным связям, имеющим относительно большую длину.

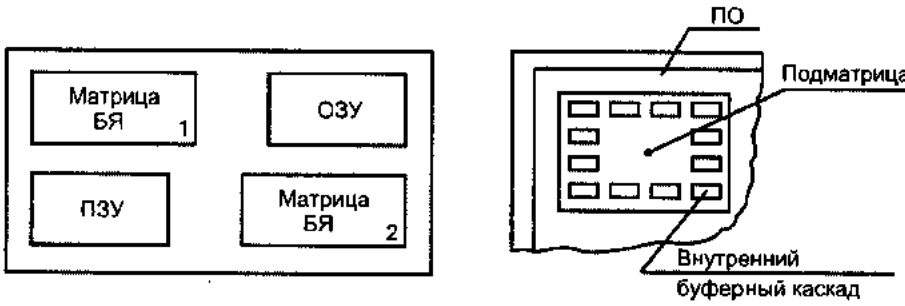


Рис. 8.22. Блочная структура БМК

Тип обрабатываемых сигналов (цифровые, аналоговые) влияет на качество и состав схемных элементов базовых ячеек. В связи с этим БМК подразделяются на *цифровые, аналоговые* и *цифроаналоговые*. Аналоговые и цифроаналоговые БМК, появившиеся позднее цифровых и менее распространенные, имеют состав базовых ячеек, позволяющий получать на их основе такие схемы, как операционные усилители, аналоговые ключи и компараторы и т. д.

Классификация по используемой схемотехнике отражает только основные варианты БМК. Варианты максимального быстродействия реализуются на схемах типа ЭСЛ или, что более экзотично, на арсениде галлия. Большое место занимает схемотехника КМОП, проявляющая свойственные ей известные достоинства.

Кроме перечисленных, известны и другие по схемотехнике БМК. Например, БМК на основе схемотехники БиКМОП, кремний на диэлектрике и др. Однако эти варианты не принадлежат, по крайней мере пока, к числу широко распространенных.

Важной характеристикой БМК является *число слоев межсоединений* (в настоящее время это 2—6). Многослойность облегчает трассировку и позволяет изготовлять БМК более высокого уровня интеграции. В простейшем случае двухслойной трассировки на первом (нижнем) уровне обычно выполняются переменные соединения внутри БЯ (часть соединений не зависит от реализуемой на БМК схемы и постоянна) и связи по вертикальным

каналам. Этот слой делается либо в виде диффузионной области самого кристалла, либо в виде поликремниевых или металлических дорожек. Второй слой металлизированных соединений дает разводку горизонтальных трасс и обслуживающих линий (питание, "земля", синхронизация и т. д.).

В четырехслойном кристалле в первом слое задаются связи внутри БЯ, во втором — вертикальные трассы, в третьем — горизонтальные, а в четвертом — обслуживающие цепи.

При увеличенном числе слоев можно исключить трассировочные каналы между ячейками, перейдя к бесканальным структурам.

На рис. 8.23 показан *компонентный состав БЯ БМК* типа ЭСЛ, рассчитанный на реализацию двухъярусных логических элементов. Не рассматривая функциональные возможности схем, получаемых на основе таких БЯ, укажем только, что резисторы R0, входящие в состав источников тока для вышележащих переключателей, могут включаться параллельно или последовательно. Это дает возможность получить несколько значений переключаемых токов, т. е. модификации схем, отличающиеся быстродействием и потребляемой мощностью.

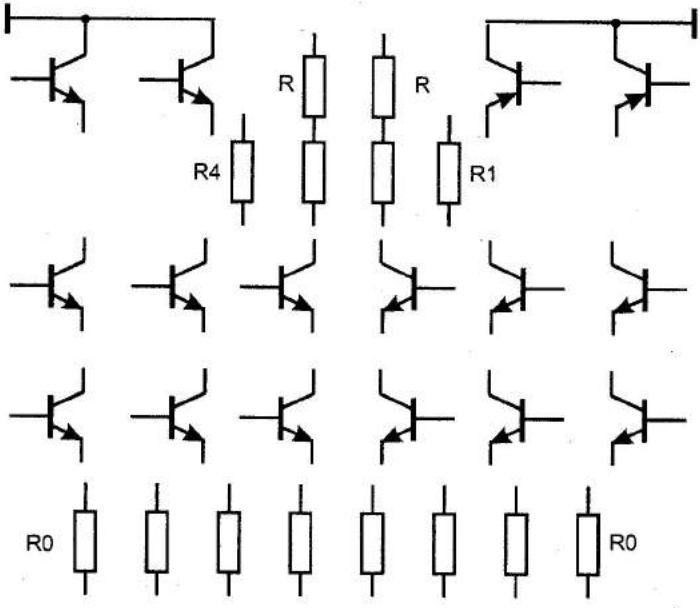


Рис. 8.23. Компонентный состав базовой ячейки БМК типа ЭСЛ

На рис. 8.24 представлен один из вариантов БЯ БМК типа КМОП. Схемными компонентами таких БЯ служат только транзисторы с p- и n-

каналами. Число транзисторов в ячейке выбирается по результатам анализа частоты использования различных логических элементов в устройствах заданного класса и преобладающих требований по нагрузочной способности, быстродействию и т. д. Высокий коэффициент использования транзисторов дают кристаллы с числом транзисторов в ячейке 4, 8 или 10. На рис. 8.24 показаны топология и электрическая схема ячейки с четырьмя транзисторами. Квадратные элементы топологического рисунка — контактные площадки к затворам и фиксированные контактные окна к элементам ячейки. Транзисторы можно соединять последовательно или параллельно, т. е. можно получать типовые подсхемы логических элементов И-НЕ и ИЛИ-НЕ. В схемотехнике КМОП транзисторы с противоположными по типу проводимости каналами всегда используются попарно, поэтому пары транзисторов могут иметь общий затвор.

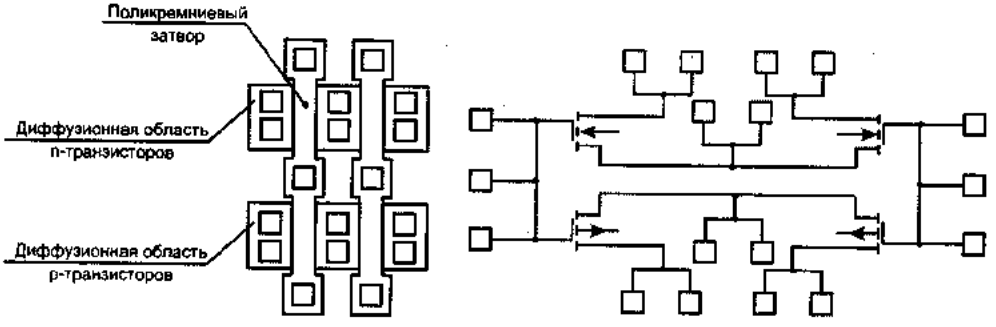


Рис. 8.24. Вариант базовой ячейки БМК типа КМОП

Усложнение ячейки достигается объединением простых ячеек в группу.

Параметры БМК

Параметры БМК можно разделить на четыре группы:

- функциональные возможности (число эквивалентных вентилей, тип БЯ, число МБЯ и ПБЯ, состав библиотеки функциональных ячеек и т. п.);
- электрические параметры (уровни напряжений, кодирующих логические сигналы, напряжения питания, потребляемые токи, задержки распространения сигналов, максимальные частоты переключений и т. п.);
- конструктивно-технологические (тип корпуса, число выводов, число уровней металлизации, площадь кристалла и т. п.);
- эксплуатационные характеристики (устойчивость к воздействию внешних факторов, надежность и т. п.).

В табл. 8.2 приведены основные параметры отечественных БМК производства ОАО "Ангстрем" (топологическая норма проектирования 0,54 мкм).

Таблица 8.2

Схема	Количество ячеек	Количество элементов библиотеки	Максимальная частота/задержка вентиля, МГц/нс
1806ХМ1	1500	125	6/8
1515ХМ1	3200	25	10/5
1593ХМ1	3200	70	35/1,5
1593ХМ2	6400	70	35/1,5
1537ХМ1	4500	51	30/2,2
1537ХМ2	17 600	51	30/2,2
1592ХМ1	100 000	230	50/1

На уровне мировой техники изготовляются БМК с десятками миллионов эквивалентных вентилей, обладающих задержками 0,1—0,2 нс.

Этапы проектирования МАБИС

Укрупненные этапы проектирования МАБИС показаны на рис. 8.25.

Исходное описание подлежащего реализации проекта может проводиться в разных вариантах. Ориентируясь на условия проектирования ОАО "Ангстрем", укажем следующие возможности представления проекта:

- в виде схемы, составленной из элементов библиотеки БМК;
- в виде описания на языках VHDL или Verilog (наиболее популярных языках из числа языков описания аппаратуры, называемых HDL, Hardware Description Languages);
- в виде схемы, составленной в элементном базисе микросхем программируемой логики, выпускаемых фирмами Xilinx, Altera, Actel;
- в виде электрической схемы, выполненной в любой библиотеке элементов;
- на основе технического задания.

Библиотека стандартных элементов БМК разрабатывается их изготовителем и является основой для проектирования МАБИС. Представление проекта в виде схемы, содержащей библиотечные элементы, выполненное заказчиком, упрощает задачи изготовителя. При использовании текстового описания на языках типа HDL "Ангстрем", получив такое описание от заказчика, перево-

дит его в описание на основе библиотечных элементов БМК. Точно так же, получив описание проекта в одном из перечисленных выше базисов микросхем программируемой логики или в виде электрической схемы в других базисах, изготовитель производит автоматизированный перевод описания проекта в базис библиотеки БМК. Сформулировав техническое задание на проект, заказчик совместно с изготовителем могут разработать требуемую схему в библиотечном базисе БМК.

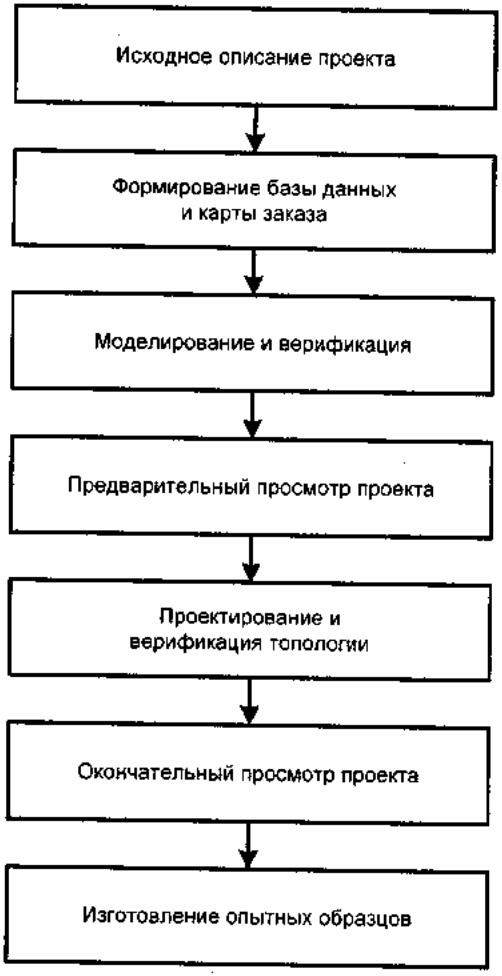


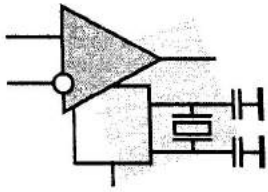
Рис. 8.25. Маршрут проектирования МАБИС

В решении задач, перечисленных для последующих этапов маршрута проектирования МАБИС, участвуют и изготовитель, и заказчик. В зависимости от конкретных условий участие заказчика может быть более или менее активным и он может взять на себя выполнение значительной доли общей работы. Как минимум требуется его участие в предварительном и окончательном просмотрах проекта.

Для разных по сложности БМК полный цикл проектирования МАБИС занимает от 1,5 до 3 месяцев.

Литература к главе: [7], [45], [47], [52], [VI], [XV], [XVI].

Глава 9



Архитектура и схемотехника БИС/СБИС с программируемыми структурами (CPLD, FPGA, смешанные структуры)

§ 9.1. Общие сведения

Микросхемы ПМЛ и БМК положили начало двум основным ветвям дальнейшего развития логических схем с программируемыми структурами. Продолжением линии ПМЛ стали БИС/СБИС сложных программируемых логических устройств CPLD (Complex Programmable Logic Devices), а продолжением линии БМК — программируемые пользователем вентильные матрицы FPGA (Field Programmable Gate Arrays). Стремление объединить достоинства обеих линий привело к созданию БИС/СБИС смешанной (комбинированной) архитектуры, для которых не выработано общепринятое название (фирма Altera, первой вступившая на путь создания таких схем, пользовалась названием FLEX (Flexible Logic Element MatriX), т. е. "матрицы гибких логических элементов"). Дальнейший рост уровня интеграции дал возможность размещать на кристалле схемы SOPC (Systems On Programmable Chip), сложность которых соответствует целым системам.

Связь поколений ИС с программируемой структурой иллюстрируется рис. 9.1, где под MPGA понимаются Mask Programmable GAs (вентильные матрицы с масочным программированием, т. е. БМК), а остальные термины уже объяснены.

Новизна темы, которой посвящена эта глава, сопровождается отсутствием установившейся терминологии, особенно в русской литературе, где иногда одни и те же термины обозначают разные вещи. Для некоторых терминов русские аналоги вообще еще не определились. Ввиду сказанного ниже используются преимущественно английские термины и аббревиатуры, что, кстати говоря, характерно также для справочной литературы и документации САПР.

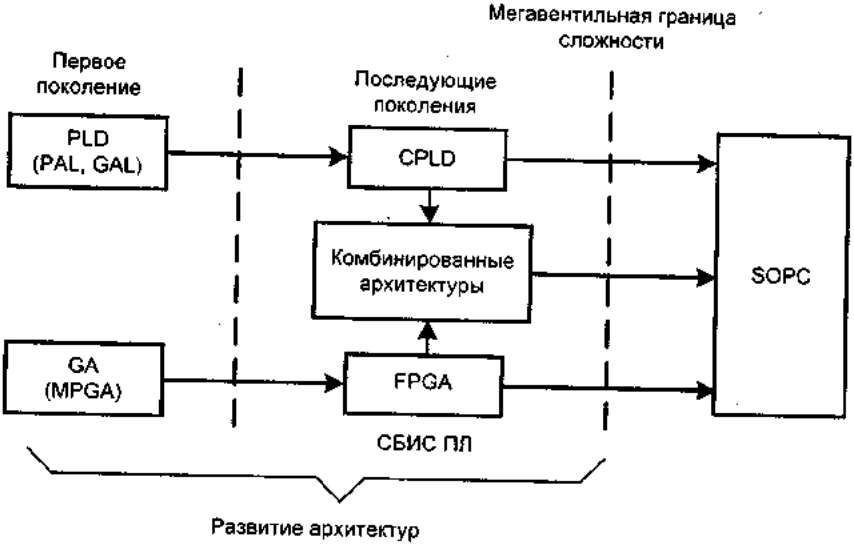


Рис. 9.1. Взаимосвязь поколений ИС программируемой логики

Об обозначениях на русском языке, принятых в этой книге — названием, объединяющим совокупность цифровых и аналоговых ИС, является ИСПС (Интегральные Схемы с Программируемой Структурой). Для цифровых (логических) схем будем пользоваться обозначением ПЛИС (программируемые логические интегральные схемы) или же (С)БИС ПЛ, т. е. (сверх)большие ИС программируемой логики. Для аналоговых микросхем с программируемыми структурами будем использовать наименование ПАИС (программируемые аналоговые интегральные схемы).

В разработке ИСПС участвуют десятки фирм, ведущими среди них являются Xilinx, Altera, Actel, Atmel, Lattice Semiconductor (все США) и некоторые другие. Перечисленные фирмы достаточно полно представляют спектр продукции в области ИСПС, хотя и не исчерпывают ее. Последующее изложение темы ориентировано в основном на разработки перечисленных выше фирм, хотя в необходимых случаях рассматриваются и микросхемы других производителей.

Сфера применения ПЛИС (СБИС ПЛ) (аналоговые программируемые ИС пока еще находятся в начале своего развития) чрезвычайно широка, на них могут строиться не только отдельные блоки систем, но и системы в целом, включая память и процессоры. Области применения ПЛИС (СБИС ПЛ) уточняются в дальнейшем; предварительно отметим важность двух применений:

- *отработка прототипов блоков и систем* при их проектировании, даже если их конечная реализация рассчитана на другие средства;

- *создание конечной продукции для относительно малотиражных изделий* быстрыми и эффективными способами.

ПЛИС классифицируются по нескольким признакам.

Классификация ПЛИС по конструктивно-технологическому типу программируемых элементов

Классификация ПЛИС (СБИС ПЛ) по конструктивно-технологическому типу программируемых элементов показана на рис. 9.2. Классификация сужена до класса цифровых микросхем, поскольку аналоговые программируемые кристаллы только появляются и классифицировать их еще рано. Программируемость, т. е. реализуемость конкретного проекта на стандартной микросхеме, обеспечивается наличием в ней множества двухполюсников, проводимость которых может быть задана пользователем либо очень малой (это соответствует разомкнутому ключу), либо достаточно большой (это соответствует замкнутому ключу). *Состояния ключей задают ту или иную конфигурацию схеме, формируемой на кристалле.* Число программируемых двухполюсников (программируемых точек связи ПТС) в ПЛИС зависит от ее сложности и может доходить до нескольких миллионов. Для современных ПЛИС (СБИС ПЛ) характерны в первую очередь следующие виды программируемых ключей:

- перемычки типа antifuse (общепринятый русский термин отсутствует);
- транзисторы с плавающим затвором (см. рис. 4.17 и текст к нему);
- ключевые МОП-транзисторы, управляемые триггерами памяти конфигурации ("теневым" ЗУ).

Программирование с помощью *перемычек типа antifuse* является однократным. В этом заключается основное ограничение на область применения схем с перемычками. Современные перемычки (фирмы QuickLogic и Actel) имеют высокое качество. Для примера на рис. 9.3 показаны перемычки первого поколения фирмы Actel. Эти перемычки компактны, имеют очень малые токи в непроводящем состоянии (единицы фемтоампер, 1 фА = 10^{-15} А). Емкости перемычек также очень малы, порядок их величин — фемтофарады. Перемычка образована трехслойным диэлектриком с чередованием слоев "оксид-нитрид-оксид", помещенным между проводящими поликремниевой и диффузионной шинами. Соответственно чередованию слоев Oxid-Nitrid-Oxid перемычки также называют перемычками типа ONO. Сложность изготовления кристалла с перемычками возрастает незначительно — по сравнению с изготовлением кристалла по базовой технологии добавляются 3 фотошаблона.

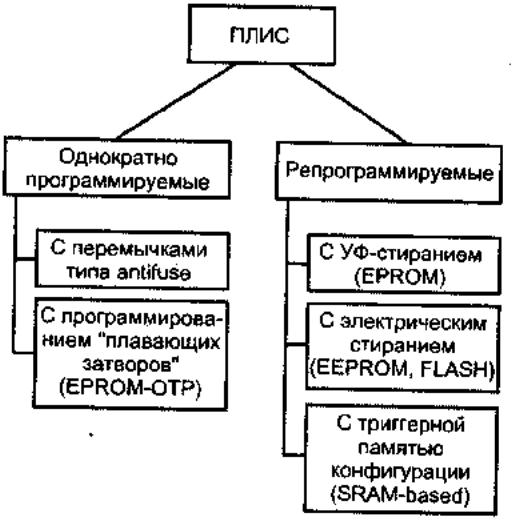


Рис. 9.2. Классификация ПЛИС по типу программируемых элементов

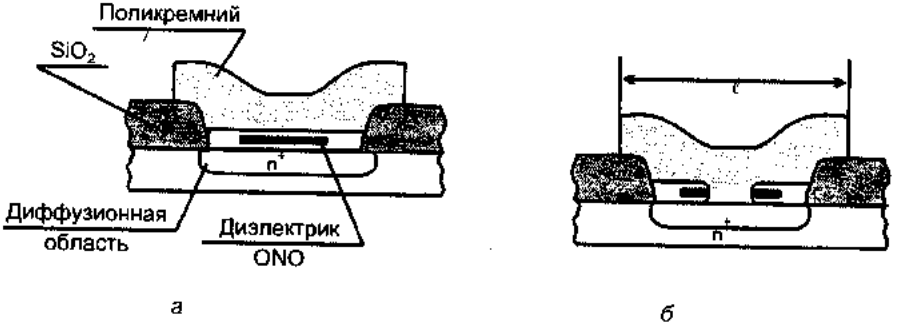


Рис. 9.3. Программируемые перемычки ONO первого поколения до (*а*) и после (*б*) программирования

Программирующий импульс напряжения пробивает перемычку и создает проводящий канал из поликремния между электродами (один электрод поликремниевый, другой — диффузионная область n^+). Величина тока, создаваемого импульсом программирования, влияет на диаметр проводящего канала, что позволяет управлять параметрами проводящей перемычки (ток 5 мА создаст перемычки со средним значением сопротивления 600 Ом, ток 15 мА — 100 Ом). Размер ℓ зависит от топологической нормы применяемой технологии (близок к ней). Параметры обоих состояний перемычки должны сохраняться около 40 лет. Малые сопротивления и малые паразитные емкости перемычек типа antifuse положительно влияют на скорости распространения сигналов в программируемых связях. Схемы с такими перемычками

обладают повышенной стойкостью к воздействию радиации, надежностью, высокой степенью засекреченности реализованного проекта и невысокой стоимостью.

В настоящее время фирмой Actel используются перемычки второго поколения, основное достоинство которых было заимствовано у фирмы QuickLogic и заключается в размещении области ONO между металлическими проводниками, что позволило реализовать перемычки не в одной плоскости с логическими схемами, а над ними, экономя таким образом площадь кристалла СБИС.

Элементы EPROM, EEPROM и *флэш-памяти* на транзисторах с плавающим затвором используются в схемах ППЗУ и рассмотрены в *главе 4.* Используются они и в ИСПС. Информацию, хранимую в памяти конфигурации с помощью программирования плавающих затворов, можно стирать с помощью ультрафиолетового облучения (УФ-стирание) или электрических сигналов. Запись новой информации в память осуществляется электрическими сигналами. В настоящее время элементы с УФ-стиранием в новой продукции встречаются редко. Но, в то же время, на основе EPROM реализуется популярный вариант без возможности стирания данных — *вариант EPROM-OTP* (OTP, One Time Programmable). Если в обычных EPROM стирание данных производится облучением кристалла через прозрачное окошко в корпусе, то в схемах OTP дорогостоящий корпус с окошком заменен на дешевый без окошка, при этом возможность стирания исключается и программирование является однократным.

Для репрограммируемых схем с зарядом/разрядом плавающих затворов сейчас применяются варианты памяти EEPROM и Flash. Не повторяя подробностей, напомним основные свойства этих вариантов. Запоминающим элементом для обоих вариантов служит транзистор с двойным затвором. Стирание старой информации и запись новой производятся электрическими сигналами и возможны в режиме ISP (In System Programming), т. е. без изъятия микросхем из смонтированных схем. Режимы стирания и записи информации отличаются от рабочих уровнями используемых напряжений (нужные для стирания и записи повышенные напряжения могут вырабатываться внутри микросхемы. Более того, имеются микросхемы со стиранием и записью напряжениями обычного уровня, но в этом случае длительности этих процессов возрастают). Число циклов репрограммирования хотя и ограничено, но достаточно велико (до 10^5—10^6).

Транзисторный *ключ, управляемый триггером памяти конфигурации*, показан на рис. 9.4. Ключевой транзистор T2 замыкает или размыкает участок *аб* в зависимости от состояния триггера, выход которого подключен к затвору транзистора T2. При программировании на линию выборки подается высокий потенциал, и транзистор T1 включается. С линии записи/чтения подается сигнал, устанавливающий триггер в состояние логической 1 или 0.

В рабочем режиме транзистор T1 заперт, триггер сохраняет неизменное состояние. Так как от триггера памяти конфигурации не требуется высокое быстродействие, он проектируется с оптимизацией по параметрам компактности и максимальной устойчивости стабильных состояний. Помехи в несколько вольт для такого триггера не влияют на его состояние. Схемы с триггерной памятью конфигурации *(SRAM-based)* впервые разработаны фирмой Xilinx.

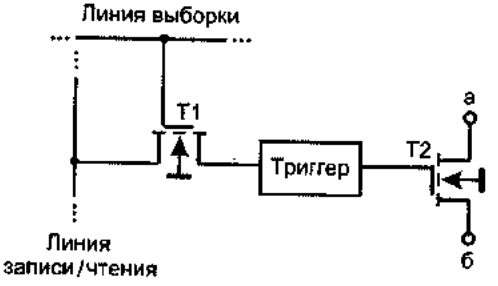


Рис. 9.4. Схема ключевого транзистора, управляемого триггером памяти конфигурации

Сравнивая элемент памяти типа "триггер плюс ключ" с элементами antifuse и транзисторами с плавающим затвором, можно видеть его более высокую схемную сложность. Тем не менее, именно такие элементы памяти конфигурации сейчас доминируют в ИСПС. Причина этого состоит не только в других достоинствах схем типа SRAM-based, но и в том, что схемная сложность триггерных элементов памяти компенсируется их *технологической однородностью* с другими схемами БИС/СБИС (логикой, регистрами), чего не имеют запоминающие элементы других типов (перемычки, транзисторы с плавающими затворами).

Загрузка соответствующих данных в память конфигурации программирует ИСПС. Процесс оперативного программирования осуществляется быстро и может производиться неограниченное число раз. В ИСПС с триггерной памятью конфигурация разрушается при каждом выключении питания. *При включении питания необходим процесс программирования (инициализации, конфигурирования) схемы* — загрузка данных конфигурации из какой-либо энергонезависимой памяти, что требует определенного времени.

Триггеры памяти конфигурации распределены по всему кристаллу вперемежку с элементами схемы, которые они конфигурируют. Ключевой транзистор T2 (в английской терминологии pass-transistor) можно назвать *программируемой точкой связи* ПТС. В английской терминологии используется название Programmable Interconnection Point, сокращенно PIP.

Режим программирования ИСПС с триггерной памятью конфигурации по своему характеру (по скорости, величине используемых напряжений питания) не отличается от рабочего режима, поскольку сводится к простой записи кодовой последовательности в цепочку триггеров. Стирание информации как специфический процесс воздействия на запоминающие элементы, требующий относительно длительных операций, вообще устранено.

О некоторых общих свойствах и возможностях применения ПЛИС

Говоря об общих свойствах ПЛИС (СБИС ПЛ), следует отметить, что благодаря регулярной структуре они *реализуются с уровнем интеграции, близким к максимальному*. Вместе с тем, поскольку для средств программирования межсоединений требуются затраты дополнительной площади кристалла, ПЛИС по количеству логических элементов, предоставляемых для реализации проекта, уступают БМК и, тем более, заказным схемам, но в последнее время разрыв между ними по указанному показателю сокращается.

В отличие от БМК, ИСПС выпускаются как полностью готовые, в них реализованы уже не только логические элементы, триггеры и т. п., но и межсоединения. Потребитель ИСПС не обращается к их изготовителю для выполнения каких-либо завершающих операций, т. к. программирование выполняет самостоятельно. *Это дает основания отнести ПЛИС (СБИС ПЛ) к стандартной продукции*, что сопровождается известными преимуществами — массовостью производства и снижением стоимости.

Как и при разработке других микросхем высшего уровня интеграции, в случае ПЛИС (СБИС ПЛ) большое внимание уделяется вопросам *понижения потребляемой мощности*. С ростом сложности СБИС потребляемая мощность становится наиболее критическим фактором. С уменьшением размеров схемных элементов снижаются и напряжения, создающие в областях транзисторов необходимые для их работы электрические поля. Снижение напряжений питания дает значительный полезный эффект как в части понижения потребляемой мощности, так и в быстродействии схем. Если длительное время типовым напряжением питания микросхем и в том числе БИС/СБИС ПЛ было 5 В, то сейчас это могут быть напряжения 3,3 В; 2,7 В; 1,8 В; 1,5 В, 1,2 В и даже менее.

Поскольку передачи по внешним связям по своим требованиям к уровням сигналов могут существенно отличаться от процессов обработки данных в ядре микросхемы, часто в СБИС используется несколько напряжений питания: U_{CC} для внутренних схем кристалла и напряжения U_{CCIO} для схем ввода/вывода данных. Современные ПЛИС (СБИС ПЛ) способны поддерживать большое число стандартов на сигналы ввода/вывода, отличающихся друг от друга уровнями напряжений и токов, числом линий передачи

(однополюсные и дифференциальные), вариантами схем согласования волновых сопротивлений в трактах передач. Чтобы реализовать набор таких стандартов, могут понадобиться *специальные опорные напряжения*, которые удобнее всего получать внутри кристалла с помощью встроенных схем. Соответственно этому для разных областей кристалла могут изготовляться транзисторы с разными пороговыми напряжениями.

В схемах особо низковольтной логики достигается очень малая мощность элементов, измеряемая единицами нВт/вентиль/МГц. При этом плотность упаковки такова, что достижима схемная сложность до нескольких десятков миллионов вентилей (сотен миллионов транзисторов).

В схемах ПЛИС (СБИС ПЛ) как правило используется иерархия режимов понижения мощности.

В активных режимах при программировании некоторых ПЛИС используется так называемый *турбо-бит*, с помощью которого выбирается один из двух режимов. Значение ON этого бита увеличивает скорость работы схемы при ограничении мощности допустимым значением. При состоянии OFF этот бит дает режим уменьшения мощности (со снижением скорости). Программируемый турбо-бит дает возможность пользователю предпочесть любой из вариантов работы схемы — более скоростной или более экономичный по потребляемой мощности. Турбо-бит включен в файл программирования СБИС.

В микросхемах программируемой логики используются и специальные схемы выявления фактов изменения входных сигналов. Каждый вход снабжается несложной схемой, содержащей элементы ИЛИ, М2 и элемент задержки. В таких схемах любое изменение входного сигнала выявляется и вызывает подачу на схему нормального питания, необходимого для быстрого протекания в ней рабочих процессов переключения. Затем автоматически питание схемы снижается, токи переходят на микроамперные уровни и потребляемая мощность падает до начала новых переходных процессов из-за новых изменений входных сигналов. В то же время схема выявления перепадов входных сигналов увеличивает задержку на пути "вход/выход" СБИС. Режимы пониженной мощности вводятся и программным путем.

Режим *Standby Power* (мощности в режиме покоя) используется, когда все входные переменные сохраняют неизменные значения. При этом схема сохраняет готовность к переходу в рабочий режим. Могут применяться и режимы *глубокого понижения мощности (Power Down)* с очень малым уровнем ее потребления. В этих режимах схема сохраняет свое информационное состояние, но для перехода в рабочий режим с обычными параметрами быстродействия требуется определенное время.

Для быстродействующих низковольтных схем разрабатывается глубоко субмикронная технология КМОП и возрождается интерес к схемам типа "кремний на диэлектрике" (SOI, Silicon On Insulator), в которых устраняются

многие паразитные схемные элементы. Кстати говоря, по технологии SOI изготовлен микропроцессор Opteron (2003 г.) известной фирмы AMD, содержащий на одном кристалле около 106 млн транзисторов, обладающий высокой производительностью и рядом других достоинств.

Эффективность ПЛИС (СБИС ПЛ) стимулирует быстрый рост соответствующей отрасли промышленности и объемов их производства, а также научных исследований по развитию их архитектур, схемотехники, алгоритмов решения практических задач.

Области применения ПЛИС

Вначале развитие PLA, PAL и первых CPLD было направлено на замену схем малого и среднего уровней интеграции более сложными, а развитие FPGA связывалось с переносом концепции БМК в область малотиражной аппаратуры, но в дальнейшем в связи с появлением репрограммируемости стало ясно, что это нечто значительно большее. Это, в частности, видно из приведенных ниже примеров.

Построение реконфигурируемых систем. В различной аппаратуре встречаются ситуации, в которых те или иные блоки работают поочередно. Например, использующие помехоустойчивые коды средства кодирования и декодирования при записи и чтении данных. Обе функции (кодирование и декодирование) никогда не выполняются одновременно. Поэтому не обязательно иметь два устройства (кодер и декодер), а можно иметь одну репрограммируемую схему с двумя разными конфигурациями, хранимыми в ПЗУ и поочередно загружаемыми в ПЛИС (СБИС ПЛ). То есть одна и та же аппаратная часть может выполнять различные преобразования после соответствующей перестройки.

Задачи логической эмуляции. При отладке устройств традиционно пользовались как изготовлением прототипа, так и программными моделями. Изготовление прототипа — сложная и дорогостоящая задача, но с его помощью можно вести тестирование с реальными сигналами и на высоких скоростях, наблюдая фактические возможности устройства. Программное моделирование лишено указанных достоинств, но проще и дешевле. Модели легко изменяются для удаления ошибок в проекте и в них просто обеспечивается хорошая наблюдаемость процессов в объекте исследования.

Применение ПЛИС (СБИС ПЛ) в задачах логической эмуляции дает сочетание достоинств обоих классических подходов. Система из микросхем программируемой логики легко создается и изменяется, но, с другой стороны, может работать с реальными сигналами и частотами их изменения. Однако по затратам труда и времени создание системы на ПЛИС сложнее, чем создание программной модели. Поэтому программные модели не зачеркиваются появлением репрограммируемых микросхем. Следует также помнить,

что полные свойства окончательно изготовленного многокристального устройства логическая эмуляция на микросхемах ПЛ отобразить не может, т. к. временные характеристики зависят от конкретной трассировки схемы, чего еще нет на этапе логической эмуляции. Таким образом, логическая эмуляция не отменяет прежние методы разработки и тестирования схем, а лишь хорошо дополняет их.

Кстати говоря, применение репрограммируемых микросхем может принести большую пользу при обучении студентов. Студенческие проекты требуют большой доработки исходных вариантов. На традиционных средствах (макетах) эту работу выполнить сложно из-за трудоемкости и дороговизны. Применение репрограммируемых ПЛИС может существенно упростить ситуацию.

Построение динамически реконфигурируемых систем. Динамическая реконфигурация (Run-Time Reconfiguration) применима в системах с выполнением действий по шагам, последовательным во времени, когда в данное время требуется только одна определенная настройка микросхемы. Вместо нескольких аппаратных блоков можно использовать один перестраиваемый, т. е. сэкономить аппаратные ресурсы за счет многократного использования одних и тех же средств в разных ролях.

Динамически реконфигурируемая микросхема может иметь практически любое число настроек, которое ограничивается лишь емкостью памяти для их хранения. Устройства с динамической реконфигурацией уже используются практически и дают ожидаемый положительный эффект. От указанных выше реконфигурируемых систем системы с динамической реконфигурацией отличаются тем, что в них требуется *быстрая смена настроек*. Обычная настройка с введением в микросхему последовательного потока битов или байт-последовательного потока занимает достаточно большое время. В динамически реконфигурируемых системах задача решается иначе. В самой системе уже имеется (хранится) набор предварительно загруженных настроек, быстро сменяющих друг друга соответственно требованиям реализуемого алгоритма. Проблемы построения систем на микросхемах ПЛ с динамической реконфигурацией в настоящее время активно исследуются.

В современной литературе ставится также вопрос о построении *FPGA-процессоров* с иными в сравнении с микропроцессорами свойствами. Алгоритмы работы процессора загружаются в FPGA принципиально подобно загрузке в память микропроцессорной системы выполняемой программы. Но при работе системы, в противоположность микропроцессорной системе, возникает сильно выраженный параллелизм на уровне мелкозернистых логических блоков с простейшими командами (типа воспроизведения функции от данного числа аргументов). Такие FPGA-процессоры могут давать хорошие результаты при параллельной обработке данных, где большое число переменных преобразуется сходным образом.

§ 9.2. Сложные программируемые логические устройства (CPLD)

Сложные программируемые логические устройства архитектурно произошли от PLD типа PAL (ПМЛ) и в английской терминологии называются CPLD (Complex Programmable Logic Devices). Переводу термина CPLD на русский язык соответствует название СПЛУ — сложные программируемые логические устройства, однако этот термин встречается редко.

Структура CPLD

CPLD (рис. 9.5) состоят из программируемой матрицы соединений ПМС (PIA, Programmable Interconnect Array), множества функциональных блоков ФБ, подобных ПМЛ, и блоков ввода/вывода БВВ.

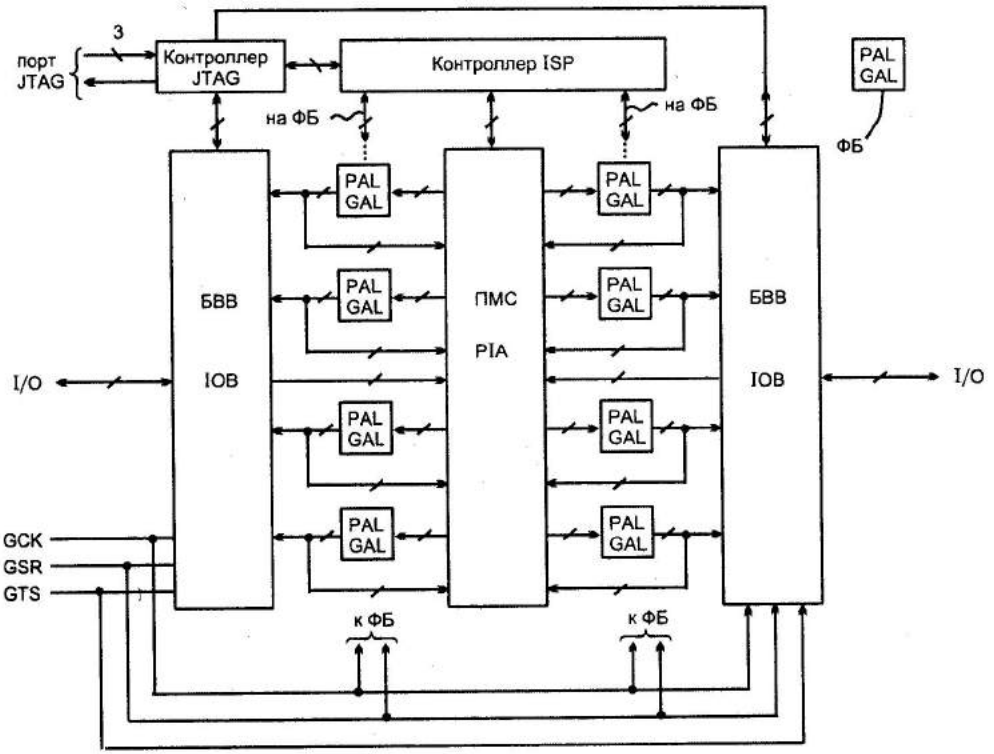


Рис. 9.5. Архитектура CPLD

В целом, CPLD представляет собою объединение нескольких PAL (ПМЛ) в единое устройство средствами программируемой коммутационной матрицы. Кроме основных блоков CPLD на схеме показаны контроллеры интерфейса JTAG и ISP, используемые для конфигурирования и тестирования создаваемых структур (подробнее эти вопросы рассмотрены в соответствующих главах книги).

Число ФБ, входящих в состав CPLD, изменяется в широких пределах в зависимости от сложности данной микросхемы. Каждый ФБ получает по m сигналов от ПМС, выходы ФБ, число которых n, подключены как к программируемой матрице соединений ПМС, так и к блокам ввода/вывода БВВ (Input/Output Block, IOB). Блоки ввода/вывода связаны с внешними двунаправленными выводами. Три вывода (на схеме слева внизу) специализированы и предназначены для глобальных, т. е. общих для всей схемы, сигналов тактирования GCK (Global Clock), сброса/установки GSR (Global Set/Reset), управления третьим состоянием GTS (Global Tri State). Возможно и иное использование специализированных выводов, если они не применяются по назначению. Число контактов ввода/вывода может быть меньше числа выводов всех ФБ. В этом случае часть макроячеек может быть использована только для выработки внутренних сигналов устройства (сигналов обратных связей), потребность в которых типична для многих видов устройств.

Функциональные блоки CPLD

Эти блоки (рис. 9.6) подобны PAL (ПМЛ) и содержат в своей основе многовходовую (wide) программируемую матрицу элементов И ($М_И$), вырабатывающую конъюнктивные термы из поступающих на ее входы переменных, группу элементов ИЛИ, между которыми распределяются выработанные термы, и некоторые другие элементы, обогащающие функциональные возможности ФБ и подобные рассмотренным в *§ 8.2*. Функциональные блоки реализуют двухуровневую логику типа ДНФ с вариантами формируемого результата (прямой или инверсный выходы, комбинационный или регистровый выходы и т. д.).

Основные части функциональных блоков CPLD:

- программируемая матрица элементов И ($М_И$);
- матрицы распределения термов МРТ;
- группа из нескольких (N) макроячеек.

Заметим, что в разных источниках структура функциональных блоков CPLD трактуется различно — иногда матрицы МРТ выделяются из состава макроячейки (как в показанной схеме), в других случаях МРТ рассматриваются как внутренние части макроячеек.

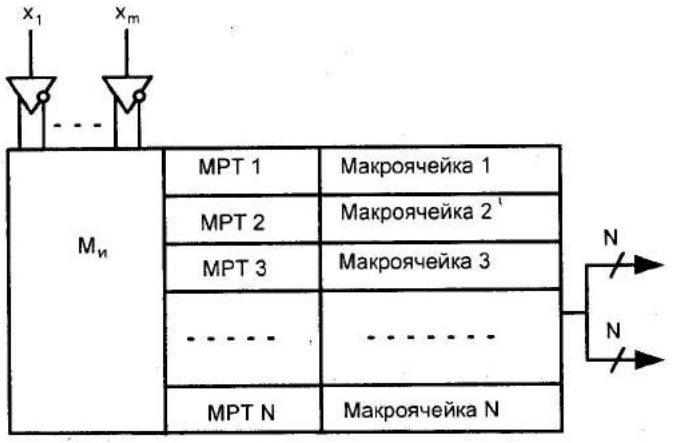


Рис. 9.6. Обобщенная структура функционального блока CPLD

Каждый ФБ представляет собою PAL-подобную структуру. Как и в классических PAL в блоке имеется матрица $M_и$, вырабатывающая конъюнктивные термы. В классических PAL термы жестко распределяются между дизъюнкторами, формирующими выходные функции в форме ДНФ. Совокупность дизъюнкторов образует фиксированную (не программируемую) матрицу элементов ИЛИ. В CPLD матрица элементов ИЛИ обычно не является полностью фиксированной, и благодаря введению в схему матрицы распределения термов МРТ возможно варьирование числа термов в вырабатываемой макроячейкой функции F_i. При этом термы заимствуются у других каналов выработки функций или отдаются им. Проще всего организовать коммутацию термов между соседними каналами. Через соседние каналы путем образования цепочечных связей можно собирать в одном канале много термов (в пределах одного функционального блока). Если термы используются не только дизъюнкторами формирования выходных функций, но и другими элементами данного ФБ (например, для управления триггерами, входящими в состав макроячейки), то и для них МРТ играет роль "раздатчика термов".

Целям повышения функциональной гибкости CPLD с помощью обогащения выходных функций макроячеек дополнительными термами служат так называемые *параллельные и последовательные логические расширители* (рис. 9.7).

Последовательные (разделяемые, общие) логические расширители создаются подачей инвертированного значения терма из МРТ данного канала обратно на один из входов матрицы $M_и$. Переданный в матрицу $M_и$ терм становится доступным для использования во всех каналах данного ФБ. Если, например, этот терм запрограммирован как

$$q = x_1\bar{x}_2x_5\bar{x}_{10},$$

т. е. инверсия терма есть $\overline{q} = \overline{x_1\overline{x}_2x_5\overline{x}_{10}} = \overline{x}_1 \vee x_2 \vee \overline{x}_5 \vee x_{10}$, то в том канале, где он будет использован вместе с входными термами канала, например, q_1, q_2, q_3, будет получена функция

$$F = q_1 \vee q_2 \vee q_3 \vee \overline{x}_1 \vee x_2 \vee \overline{x}_3 \vee x_{10} = q_1 \vee q_2 \vee q_3 \vee \overline{x_1\overline{x}_2x_5\overline{x}_{10}}\,.$$

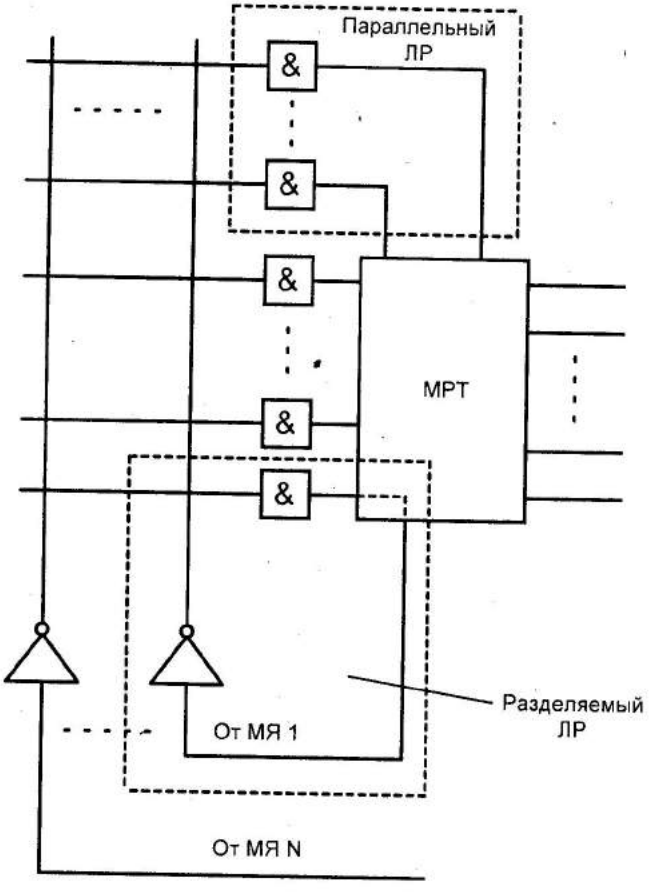


Рис. 9.7. Логические расширители параллельного и последовательного типов

Параллельный расширитель позволяет передавать термы одного канала другому. Возможность приема в свой канал термов от соседнего канала обычно означает и возможность приема через него термов и более далеких каналов с образованием цепочки для сбора термов от нескольких каналов (например, в пределах целого функционального блока). Можно, естественно, и отдавать

собственные термы или их часть другим каналам (в частности, соседним, а через них и более далеким).

Термы от МРТ поступают далее на макроячейку (МЯ). Макроячейка содержит в качестве основы элемент ИЛИ, программируемые мультиплексоры, триггер (или триггеры) и формирует выходные сигналы ФБ в нескольких вариантах.

На рис. 9.8 раскрыта схема одной из идентичных макроячеек ФБ. Прообразом показанной схемы является макроячейка CPLD XC9500 фирмы Xilinx, не имеющая разделяемого (общего) логического расширителя.

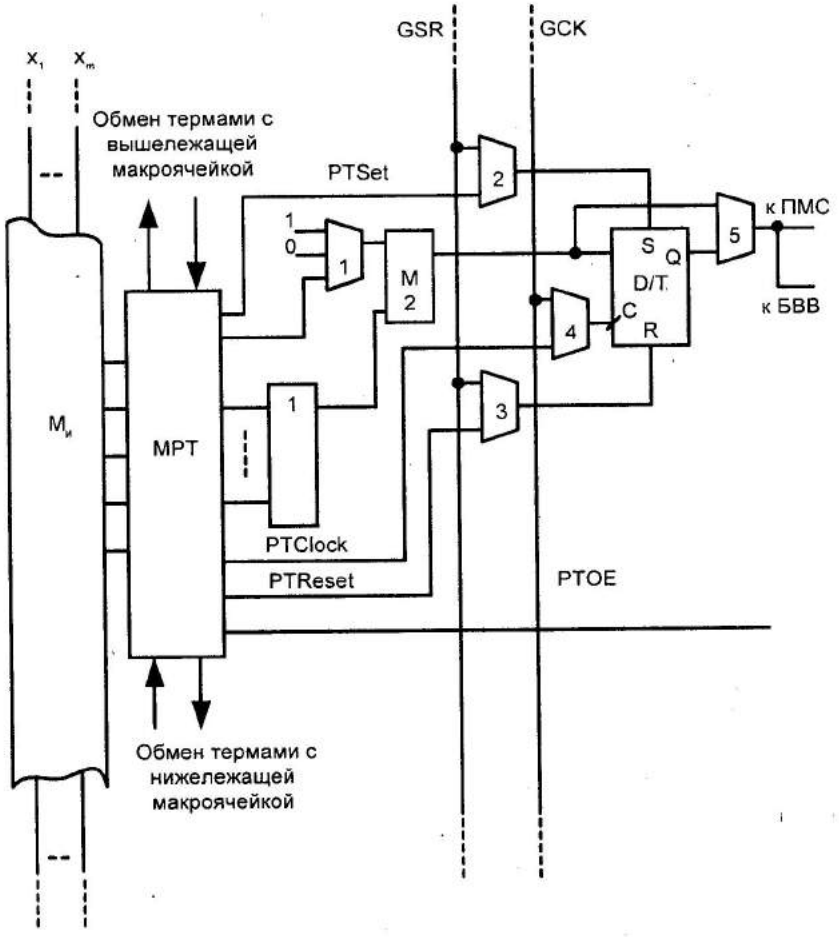


Рис. 9.8. Пример схемы макроячейки функционального блока CPLD

Программируемость мультиплексоров в этой схеме и всех дальнейших не отображается, т. к. присуща всем без исключения имеющимся в схеме мультиплексорам, если не оговорено противоположное.

В зависимости от программирования каждый мультиплексор передает на выход сигнал с того или иного входа. Триггер может программироваться на режимы работы триггера типа D или T. Заметим, кстати, что при описании микросхем программируемой логики триггеры в иностранной литературе чаще всего называют *регистрами*. Триггеры тактируются положительными фронтами синхросигналов и имеют входы установки S и сброса R. Выходные сигналы ФБ передаются в ПМС и в блоки ввода/вывода БВВ.

Аргументы x_1—x_m реализуемой макроячейкой функции поступают в матрицу И ($M_и$) из ПМС. Аргументами для МЯ могут быть как входные сигналы, поступающие извне через БВВ, так и сигналы обратных связей, подаваемые в матрицу $M_и$ с выходов макроячеек. На рис. 9.8 входные сигналы матрицы независимо от их характера обозначены через x_1—x_m. Входные буферы преобразуют сигналы в парафазные, представляя каждый сигнал его прямым и инверсным значениями, так что в матрице имеются 2m вертикальных линий и образующие ее конъюнкторы имеют по 2m входов. Пять термов из матрицы $M_и$ поступают на элемент ИЛИ для образования логической функции. Для управления триггером и буферами блока ввода/вывода вырабатываются также термы PTSet, PTClock, PTReset, которые могут быть использованы как сигналы установки, синхронизации и сброса триггера. Терм PTOE — программируемый терм управления третьим состоянием буфера БВВ (OE — Output Enable). Всего в матрице $M_и$ программируются 9N термов.

На выходе элемента ИЛИ вырабатывается логическая функция в форме ДНФ ранга не более m. Ее значение передается дальше на один из входов элемента сложения по модулю 2, на второй вход которого в зависимости от программирования мультиплексора 1 может быть подан логический нуль, логическая единица или терм PT1. В первом случае функция передается без изменений ($F = F^*$), во втором инвертируется ($F = \bar{F}^*$), в третьем передается в прямом виде во всех ситуациях за исключением такой, в которой PT1 = 1.

Мультиплексор MUX5 программируется для передачи на выход МЯ либо непосредственно значения функции F (комбинационный выход), либо состояния триггера (регистровый выход). Характер тактирования триггера определяется программированием мультиплексора MUX4, при этом возможно использование глобального синхросигнала (GCK, Global Clock) или сигнала, порождаемого термом PTClock. Асинхронные установка и сброс триггера производятся либо глобальным сигналом (GSR, Global Set/Reset), либо термами PTSet и PTReset, что определяется программированием мультиплексоров MUX2 и MUX3. Сам триггер программируется на режимы триггера задержки (типа D) или счетного (типа T).

Основной выходной сигнал макроячейки поступает как в ПМС, которая может направлять его по любому требуемому маршруту, так и в блоки ввода/вывода.

Системы коммутации CPLD (программируемые матрицы соединений)

В CPLD используется *непрерывная или одномерно непрерывная система связей*, реализуемая в виде программируемой матрицы соединений. В этом случае все связи идентичны, что дает хорошую предсказуемость задержек сигналов — важное достоинство, облегчающее проектирование и изготовление работоспособных схем высокого быстродействия.

В программируемой матрице соединений ПМС (рис. 9.9) выходы функциональных блоков ФБ подключаются к вертикальным непрерывным (не сегментированным) линиям, причем каждому выходу соответствует своя линия. Входы ФБ связаны с горизонтальными линиями, пересекающими все вертикальные линии. На пересечениях горизонтальных входных линий и вертикальных линий имеются программируемые точки связи. Замкнув одну из этих точек, можно подключить вход к соответствующему выходу. Таким образом любой вход ФБ может быть подключен к любому выходу, а каждый из выходов может быть подключен ко многим входам, чем обеспечивается так называемая *полная коммутируемость блоков*.

Достоинством ПМС рассмотренного типа является *малая и предсказуемая задержка* коммутируемых сигналов, т. к. для каждого соединения образуется идентичный всем другим канал связи с малым числом программируемых ключей (в линиях передачи сигналов, но не в матрице в целом). Замкнутые транзисторные ключи имеют в первом приближении схему замещения в виде инерционной RC-цепи и вносят основные задержки в процесс распространения сигнала. При этом задержка сигнала в цепочке пассивных RC-звеньев зависит от их числа по квадратичному закону.

Программируемые ключи в линиях передачи сигналов из ПМС в функциональные блоки могут даже отсутствовать, если эти передачи организованы так, как показано на рис. 9.10. В этом случае программируются только напряжения на нижних входах конъюнкторов, и ФБ получит сигнал от i-ой вертикальной линии ПМС (i = 1m), если транзистор T_i будет заперт, и на нижнем входе i-го конъюнктора будет действовать высокий потенциал логической единицы. Открытый транзистор T_i подключает нижний вход конъюнктора к нулевому потенциалу, создавая на нем и на выходе конъюнктора сигнал логического нуля. Таким образом, задавая триггеру T_i состояние логического нуля, а остальным триггерам — состояние логической единицы, можно обеспечить запертое состояние транзистора T_i и открытое

состояние всех других транзисторов, что означает подключение входа ФБ к i-ой вертикальной линии ПМС с образованием так называемого непрерывного соединения.

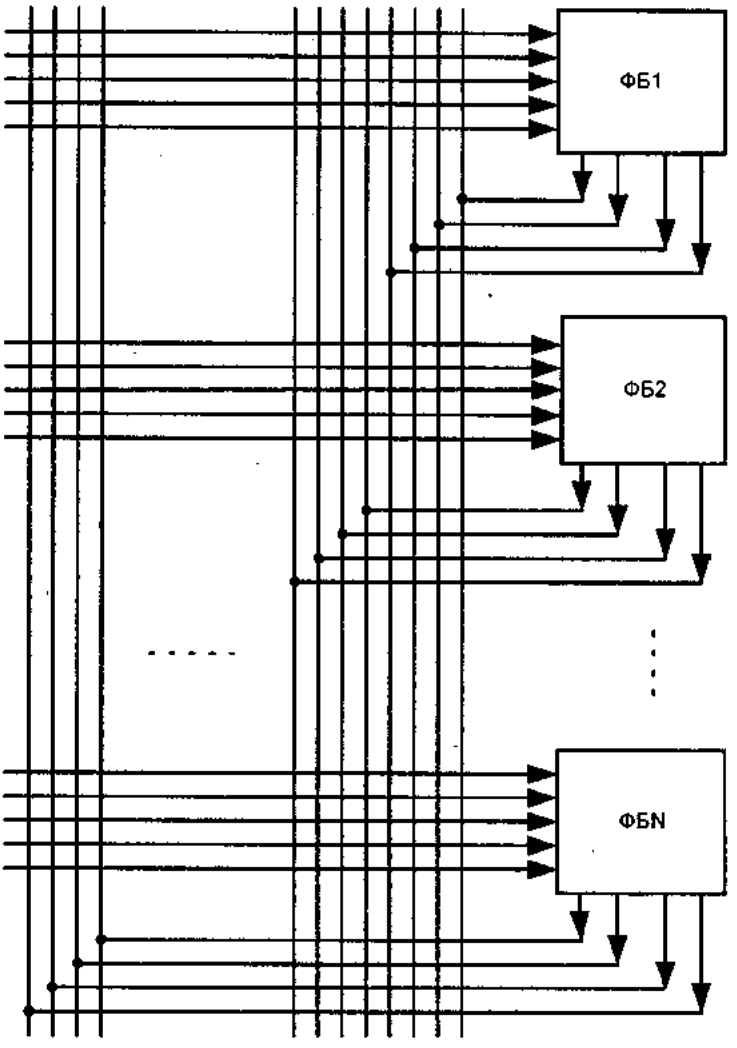


Рис. 9.9. Схема программируемой матрицы соединений CPLD

Разрешив прохождение нескольких сигналов, можно получить на выходе схемы их дизъюнкцию.

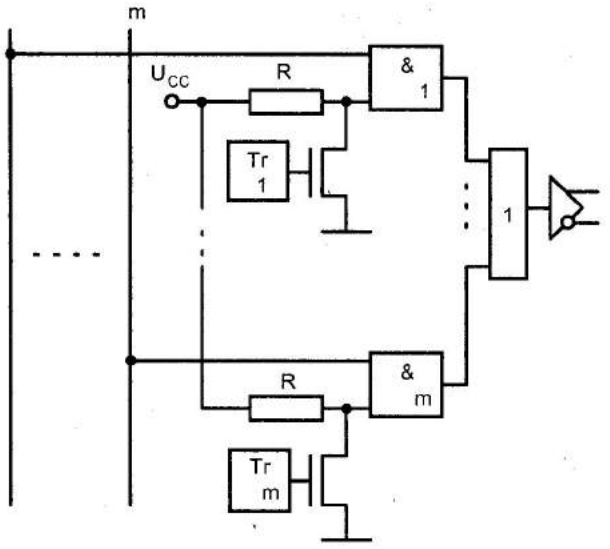


Рис. 9.10. Вариант схемы, передающей сигналы из матрицы соединений в функциональный блок

Программируемые матрицы соединений типа, показанных на рис. 9.9 эффективны в схемах с относительно небольшим числом коммутируемых блоков. При большом их числе, характерном, например, для FPGA, подобные ПМС были бы чрезмерно сложны, поскольку любое соединение образуется с помощью линий связи, проходящих по всей длине и ширине схемы, тогда как очень многие связи локальны и соединяют близлежащие блоки. Поэтому в FPGA системы коммутации строятся иначе — с помощью сегментированных линий связи.

Блоки ввода/вывода CPLD

Блоки ввода/вывода соединяют внешние контакты микросхемы с ее внутренними цепями. Характерным примером такого блока может служить БВВ CPLD типа XC9500 фирмы Xilinx, показанный на рис. 9.11.

Основой БВВ служат два буфера — входной (1) и выходной (2). Чтобы обеспечить постоянство уровней напряжения, поступающих на входной буфер, и их независимость от амплитуды входных сигналов, в схеме вырабатывается внутреннее напряжение питания V_{CCINT} и вводится цепь из двух фиксирующих диодов.

Схема программируемой общей точки ПрОТ позволяет пользователю при необходимости получать дополнительный "заземленный" вывод. Дополни-

тельные выводы для системы "заземления" повышают ее качество и тем самым снижают уровень помех в микросхеме.

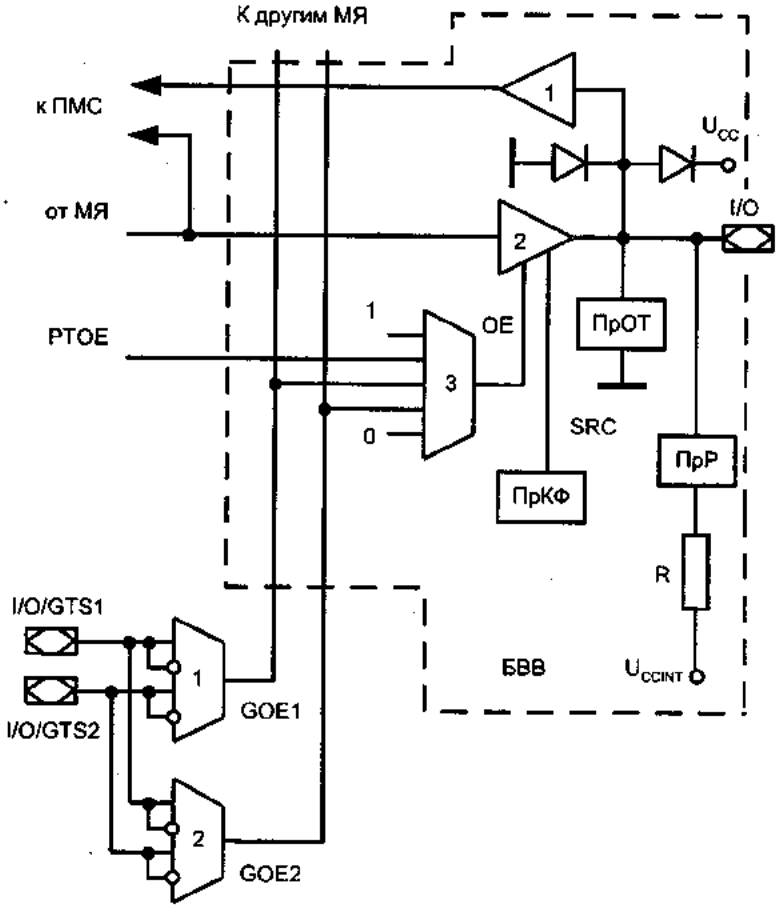


Рис. 9.11. Пример схемы блока ввода/вывода CPLD

Схема программирования подключаемого резистора ПрР введена для исключения плавающих потенциалов на контактах ввода, когда они не используются в рабочем режиме. В этом случае контакту задается высокий потенциал от цепи V_{CCINT} — R. Резистор R используется и в некоторых других режимах, а в рабочих режимах отключается.

Выходной буфер 2 получает сигналы разрешения работы OE и управления крутизной фронта выходного напряжения SRC (Slew Rate Control). Сигнал OE с помощью программируемого мультиплексора MUX3 вырабатывается в нескольких вариантах: от терма PTOE, получаемого от макроячейки, от лю-

бого из глобальных сигналов управления третьим состоянием (GOE1, GOE2), от константы 1 и от константы 0. Глобальные сигналы управления третьим состоянием образуются с возможностью выбора любой полярности исходных сигналов GTS1 и GTS2.

Выходные буферы конфигурируются для работы с напряжениями питания 5 В или 3,3 В при подключении внешнего источника питания с тем или иным уровнем напряжения (эти цифры относятся к рассматриваемому блоку типа XC9500, сейчас у блоков ввода/вывода нередко уровни выходных сигналов могут выбираться из многих возможностей, в том числе из таких низких напряжений, как 2,5 и 1,8 В).

Для класса CPLD типичны микросхемы с уровнем интеграции 600—20000 эквивалентных вентилей, числом макроячеек 32—512, числом функциональных блоков 2—16 и временем распространения сигнала от любого входа до любого выхода 5—20 нс. Эти CPLD представлены, в частности, такими популярными семействами микросхем, как MAX7000 и MAX3000A фирмы Altera, XC9500 и CoolRunner фирмы Xilinx, MACH1—MACH5 фирмы Lattice Semiconductor. Архитектура семейства MAX7000 использована несколькими фирмами как базовая для производства их собственных CPLD, обладающих той или иной спецификой.

В CPLD применяется энергонезависимая память конфигурации, причем *доминирует память типа EEPROM или Flash*. Энергонезависимость памяти конфигурации облегчает засекречивание проектов, т. к. не требует загрузки внешних данных конфигурации, доступных для чтения, при каждом включении питания. Содержимое памяти конфигурации на самом кристалле обычно защищается специальным битом секретности, сбросить который можно лишь при стирании всего содержимого памяти.

CPLD — более простые устройства в сравнении с FPGA и другими ПЛИС (СБИС ПЛ) высшего уровня сложности. На них удобно строить *относительно несложные устройства высокого быстродействия*, в которых не требуется реализация сложных вычислительных алгоритмов. Примерами таких устройств могут служить интерфейсные схемы или управляющие автоматы, хотя, конечно, область применения CPLD этими примерами не исчерпывается.

§ 9.3. Программируемые пользователем вентильные матрицы (FPGA)

Предварительные замечания

Программируемые пользователем вентильные матрицы (ППВМ или FPGA — Field Programmable Gate Arrays) топологически сходны с канальными БМК. В их внутренней области размещается множество регулярно расположенных идентичных конфигурируемых логических блоков (КЛБ),

между которыми проходят трассировочные каналы, а на периферии кристалла расположены блоки ввода/вывода БВВ (IOB, Input/Output Blocks). Перечисленные части FPGA составляют их основу, а для первого поколения по существу все, что размещалось на кристаллах. Позднее в состав FPGA стали вводить схемы памяти, схемы управления тактированием и др. Таким образом, архитектуру FPGA можно представить рисунком, подобным рис. 8.21, *а*, *д*, если вместо наименования "базовая ячейка" иметь в виду наименование КЛБ, вместо "периферийной ячейки" — БВВ, и учесть, что в трассировочном пространстве уже реализованы заготовки межсоединений, которые при проектировании нужной схемы остается лишь запрограммировать своими силами.

К широко известным классическим FPGA относятся БИС/СБИС фирмы Xilinx, которая в 1985 г. впервые выпустила FPGA с триггерной памятью конфигурации. Среди FPGA с перемычками типа antifuse следует отметить микросхемы фирм Actel и QuickLogic. Другие типы программируемых элементов для FPGA не характерны. Среди микросхем фирмы Xilinx классической архитектурой отличаются семейства XC4000, Spartan, Spartan II, Spartan IIE, Spartan-3.

Свойства и возможности FPGA зависят в первую очередь от характера их КЛБ и системы межсоединений.

Логические блоки FPGA

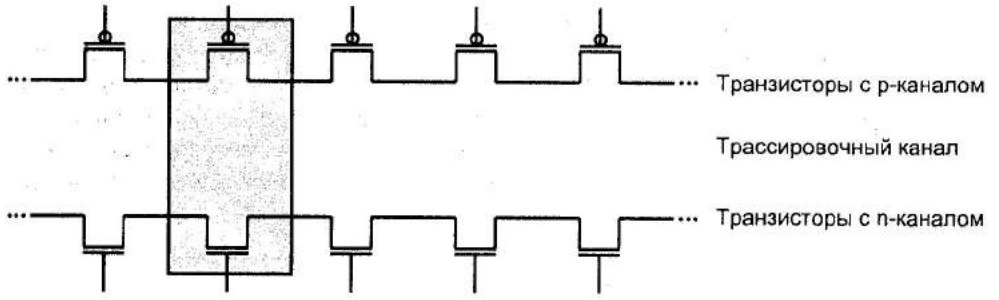
В качестве КЛБ (далее для краткости просто ЛБ — логические блоки) используются:

- транзисторные пары или простые логические вентили (И-НЕ, ИЛИ-НЕ и т. п.). Такие ЛБ называют SLC — Simple Logic Cells;
- логические модули на основе мультиплексоров;
- логические модули на основе программируемых ПЗУ, такие ЛБ называют LUTs — Look-Up Tables.

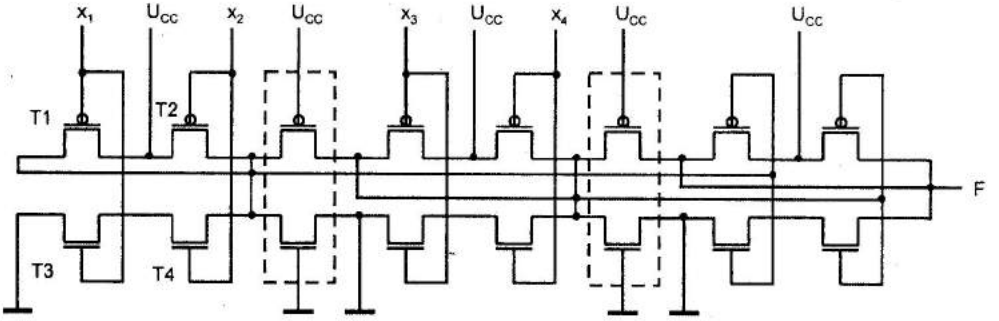
Важной характеристикой ЛБ является их *"зернистость"* (Granularity). Другой важной характеристикой считается *"функциональность"* (Functionality).

Первое свойство связано с тем, насколько "мелкими" будут те части, из которых можно "собирать" нужные схемы, второе — с тем, насколько велики логические возможности ЛБ.

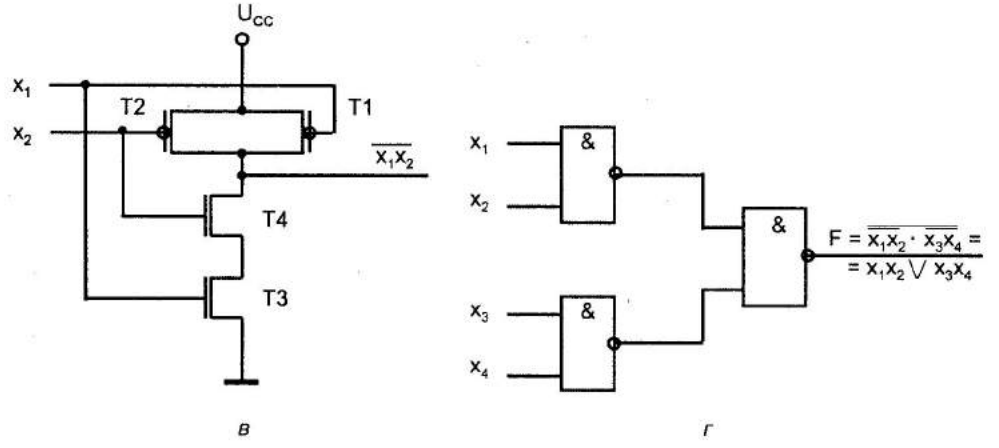
Примером наиболее *мелкозернистого* может служить ЛБ, показанный на рис. 9.12, *а*. Блок содержит цепочки транзисторов с p- и n-каналами (на рисунке использованы американские обозначения транзисторов, более простые, чем отечественные). Простейшим ЛБ может служить пара из транзисторов разного типа проводимости (выделенный прямоугольник). Между цепочками транзисторов имеются трассировочные каналы, в которых реализуются необходимые межсоединения элементов.



а



б



в *г*

Рис. 9.12. Схема мелкозернистых логических блоков (*а*), реализация межсоединений для воспроизведения функции $x_1x_2 \vee x_3x_4$ (*б*) и пояснения к этой реализации (*в*, *г*)

На рис. 9.12, *б* показан пример межсоединений, дающих реализацию функции $F = x_1x_2 \vee x_3x_4$. Пары транзисторов в прямоугольниках из штриховых линий имеют такие постоянные напряжения на затворах, что оказываются запертыми. Эти пары разделяют цепочки на части, изолированные друг от друга. В трех секциях собраны схемы типа рис. 9.12, *в*, т. е. ячейки И-НЕ обычного для схемотехники типа КМОП. Эти ячейки соединены между собою как показано на рис. 9.12, *г*, что и приводит к нужному результату.

Мелкозернистость ЛБ ведет к большей гибкости их использования, возможностям реализовать воспроизводимые функции разными способами, получая разные варианты в координатах "площадь кристалла — быстродействие". В то же время мелкозернистость ЛБ усложняет систему межсоединений FPGA в связи с большим числом программируемых точек связи.

Примерами *более крупнозернистых* ЛБ могут служить блоки микросхем фирмы Actel. В *главе 2* (на рис. 2.15, *а*) был показан ЛБ, состоящий из трех мультиплексоров "2–1" и элемента ИЛИ, для которого воспроизводимая функция

$$F = (\overline{S_0 \vee S_1})(\overline{S}_A A_0 \vee S_A A_1) \vee (S_0 \vee S_1)(\overline{S}_B B_0 \vee S_B B_1).$$

Подключая ко входам ЛБ переменные и константы, можно получить все комбинационные функции двух переменных, все функции трех переменных с, по меньшей мере, одним положительно юнатным входом, многие функции четырех переменных и некоторые функции большего числа переменных, вплоть до восьми. В целом получаются 702 различных варианта (макроса). Например, подключая к входам переменные и константы согласно рис. 9.13, где $S_0 = c$; $S_1 = S_A = B_0 = 0$, $A_0 = A_1 = 1$, $B_1 = a$, $S_B = b$, получим функцию $F = ab \vee \bar{c}$.

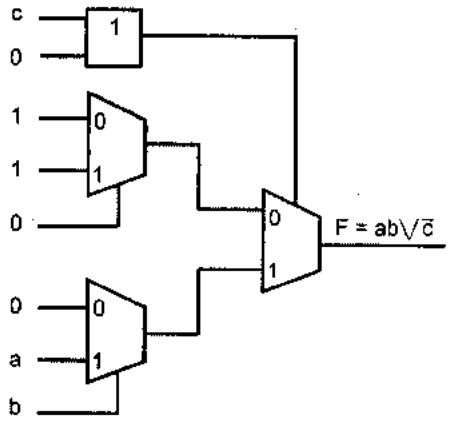


Рис. 9.13. Пример реализации функции $F = ab \vee \bar{c}$ с помощью мультиплексорного логического блока

В FPGA с триггерной памятью конфигурации обычно применяют крупнозернистые блоки, в которых реализуются более сложные функции, что ведет к упрощению программируемой части межсоединений. В то же время труднее полностью использовать логические элементы блоков, что может привести к излишним затратам площади кристалла и снижению быстродействия. Иными словами, меняя зернистость, можно выиграть в одном и проиграть в другом.

Крупнозернистые блоки, в частности, характерны для семейств FPGA фирмы Xilinx. Архитектурная основа этих блоков была заложена при создании семейства XC4000, КЛБ которого содержит три табличных функциональных преобразователя, два триггера и 16 программируемых мультиплексоров. В последующих FPGA высокой сложности (семействах Virtex) логические блоки были несколько упрощены. Такие обновленные КЛБ были затем использованы в микросхемах семейства Spartan, современного семейства FPGA средней сложности. Логический блок микросхем семейства Spartan в несколько упрощенном виде показан на рис. 9.14.

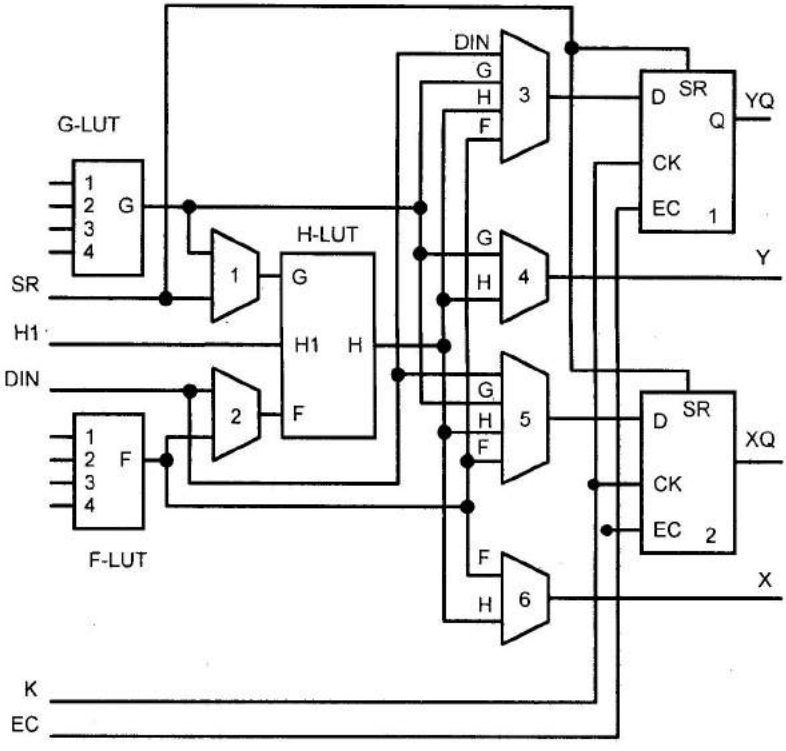


Рис. 9.14. Схема КЛБ семейства Spartan

В функциональных блоках этих микросхем логические преобразования выполняются тремя LUT-блоками (Look-Up Tables) G, F и H. Преобразователи G и F — программируемые запоминающие устройства (ЗУ) с организацией 16×1, способные воспроизводить любые функции четырех переменных, значения которых могут быть переданы на выходы Y и X через мультиплексоры 4 и 6 при соответствующем их программировании (через линии верхних входов мультиплексоров).

Заметим, что на рисунках, как и ранее, в обозначениях мультиплексоров не отражена их программируемость, поскольку все они без исключения обладают этим свойством.

Через верхний вход мультиплексора 1 и нижний вход мультиплексора 2 функции G и F могут быть поданы на ФП-H (H-LUT — ЗУ с организацией 8×1) для образования "функции от функций" с целью получения результирующей функции, зависящей от более чем четырех аргументов. К третьему входу ФП-H (H-LUT) подключен входной сигнал H1, так что H = f(G, F, H1). Аргументами для ФП-H (H-LUT), поступающими от мультиплексоров 1 и 2 в зависимости от их программирования может быть не только набор G, F, H1, но также наборы G, H1, DIN; SR, H1, DIN; SR, H1, F. Линии DIN (Data Input) и SR (Set/Reset) используются либо для передачи в триггер непосредственно входных данных и сигнала установки/сброса, либо как входы ФП-H (H-LUT).

Перечисленные ресурсы логической части ФБ позволяют воспроизводить следующие функции:

- любую функцию с числом аргументов до 4 включительно плюс вторую такую же функцию плюс любую функцию с числом аргументов до трех;
- любую функцию 5 аргументов (одну);
- любую функцию 4 аргументов и одновременно некоторые функции 6 аргументов, некоторые функции с числом аргументов до 9.

Сигналы H1, DIN, SR, EC являются для ФБ входными, они подаются на его внутренние схемы через группу из четырех мультиплексоров MUX размерностью "4—1" (на рисунке не показаны), к которым подключены 4 линии внешней шины управления. Это позволяет распределять сигналы H1, DIN, SR и EC по внешним линиям управления в любом желаемом варианте.

Мультиплексоры 3—6 направляют те или иные сигналы на триггеры 1 и 2. Триггеры могут использоваться для фиксации и хранения выходных сигналов функциональных преобразователей или же работать независимо от них. Входной сигнал ФБ DIN может быть прямым входом для любого триггера. Сигнал H1 тоже можно передавать любому триггеру, но через ФП-H (H-LUT), что вносит в цепь его передачи некоторую задержку.

Оба триггера имеют общие входы CK тактирования от сигнала K, разрешения тактирования EC и установки/сброса SR. Внутренние программируемые

цепи в схеме триггера (на рис. 9.14 не показаны) позволяют индивидуально программировать полярность тактирующего сигнала CK. Сигнал EC синхронизирован с сигналом CK, сигнал SR асинхронный и для каждого триггера с использованием внутренних цепей триггера программируется как сигнал установки или сброса. Этот сигнал определяет состояние, в котором окажется триггер после процесса конфигурации микросхемы. Конфигурация определяет и характер воздействия на триггеры импульсов GSR (Global SR) и SR при работе схемы.

Блоки ввода/вывода FPGA

Характерные черты блока ввода/вывода рассмотрим на примере семейства Spartan (рис. 9.15). Каждому выводу корпуса микросхемы придается блок ввода/вывода, который может конфигурироваться как вход, выход или двунаправленный вывод.

Режим выходного блока обслуживается следующими элементами: выходным буфером 1, триггером 1, мультиплексорами 1, 2, 5 и логической схемой ИЛИ. Выводимый сигнал O можно получать в прямой или инверсной форме в зависимости от программирования мультиплексора 2. Этот сигнал может передаваться на выходной буфер непосредственно или сниматься с триггера при соответствующем программировании мультиплексора 5. Сигналы T и GTS (Global Tri State) согласно логике ИЛИ управляют переводом буфера в третье состояние, причем активный уровень сигнала T программируется с помощью мультиплексора 1. Внутренние программируемые цепи триггера (на рисунке не показаны) позволяют изменять полярность тактирующего фронта. Сам буфер имеет программируемые крутизну фронта выходного сигнала и его уровни (КМОП/ТТЛ). Крутизна фронтов в некритичных к скорости передачи цепях снижается для уменьшения уровня помех на шинах питания и земли. Используется так называемый *мягкий старт* (Soft Start Up), снижающий помехи при конфигурировании схемы и ее переходе к рабочему режиму, когда одновременно активизируются многие буферы. Первая активизация автоматически происходит с пологими фронтами перепадов напряжения. Затем вступает в силу заданный выбор крутизны фронтов в зависимости от принятой конфигурации БВВ.

Тракт ввода сигналов содержит входной буфер 2, триггер 2, программируемые мультиплексоры 3, 4, 6, элемент задержки ЭЗ (DEL) и программируемые схемы задания определенных потенциалов выводу, к которому не подключен вводимый или выводимый сигнал (схемы Pull Up/Pull Down). Вводимый сигнал в зависимости от программирования мультиплексоров 3 и 4 или поступает непосредственно в систему коммутации FPGA по входным линиям I1 и I2 или же фиксируется триггером и с его выхода передается в эти линии. Триггеры могут конфигурироваться как тактируемые фронтом или как защелки (D-триггеры, управляемые уровнем). Выбор осуществляется

присвоением триггеру соответствующего библиотечного символа. В цепи передачи сигнала триггеру 2 могут быть включены элементы задержки (при передаче сигнала через нижний вход мультиплексора 6). Включение задержки гарантирует необходимые временные соотношения между входными сигналами триггера D и глобальным сигналом тактирования.

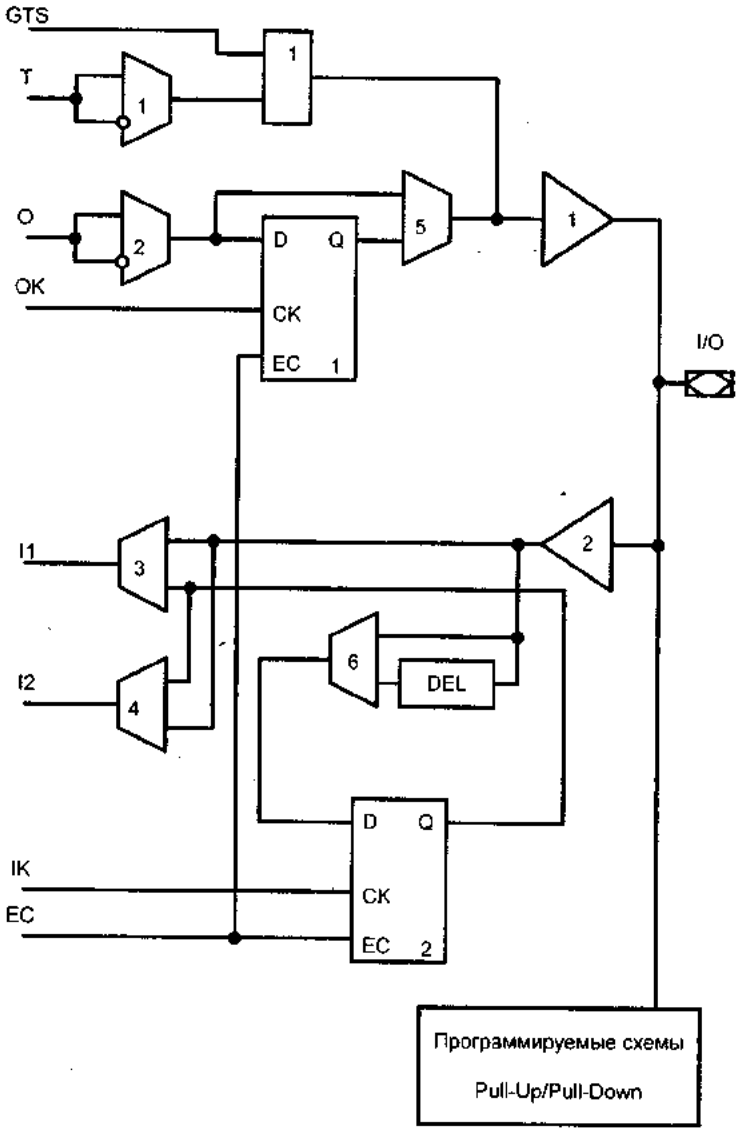


Рис. 9.15. Пример схемы блока ввода/вывода FPGA

Входной буфер может конфигурироваться для восприятия входных сигналов с пороговым значением ТТЛ (1,2 В) или КМОП (0,5 U_{CC}). Выходные уровни тоже конфигурируются, две глобальные регулировки входных порогов и выходных уровней независимы.

Системы межсоединений FPGA

Системы межсоединений (коммутации), как и логические блоки, реализуются в широком диапазоне архитектурных и технологических решений. Линии связей в FPGA *сегментированы*, т. е. составлены из отдельных проводящих сегментов (участков, не содержащих ключей). Сегменты соединяются друг с другом *программируемым элементом связи* (ключом). В системе межсоединений применяются сегменты различной длины. Малое количество сегментов (при их большой длине) ведет к недостаточно эффективному использованию логических блоков, слишком большое (при их малой длине) — к появлению большого числа программируемых ключей в линиях связи, что увеличивает затраты площади кристалла и вносит дополнительные задержки сигналов.

Короткие сегменты затрудняют реализацию длинных связей, длинные — коротких. Поэтому целесообразна *иерархическая система связей* с несколькими типами межсоединений для передач на разные расстояния. Целью построения системы связей является обеспечение максимальной коммутируемости блоков при минимальном количестве ключей и задержек сигналов, а также по возможности предсказуемость задержек, облегчающая проектирование устройств на основе FPGA. Наличие ключей и схем для их программирования усложняет межсоединения FPGA сравнительно с межсоединениями БМК.

Критерий *трассировочной способности* системы межсоединений отображает возможность создания в FPGA множества схем типового применения (только с помощью программируемых ключей, т. к. сегментная часть соединений стандартная).

Быстродействие FPGA существенно зависит от задержек сигналов в связях. Ключи в линиях связи имеют схему замещения в виде RC-звеньев и в основном определяют задержки сигналов. В последовательной цепочке RC-звеньев *задержка зависит от числа звеньев квадратично*, поэтому цепи с большим числом ключей особенно нежелательны. Может оказаться целесообразным разбиение длинной цепочки сегментов на несколько коротких с помощью промежуточных буферных каскадов.

Система межсоединений фирмы Actel. В разработке FPGA с однократным программированием перемычками antifuse важную роль играет фирма Actel. Логические блоки в FPGA этой фирмы располагаются в виде горизонтальных рядов, между которыми имеются трассировочные каналы. В каналах

горизонтально в четыре строки расположены сегменты различной длины и различного взаимного положения по горизонтали. Через ЛБ и трассировочные каналы проходят вертикальные сегменты. Каждый вход ЛБ соединен со своим вертикальным сегментом, пересекающим ближайший канал. Выход ЛБ имеет свой вертикальный сегмент, пересекающий несколько каналов (рис. 9.16).

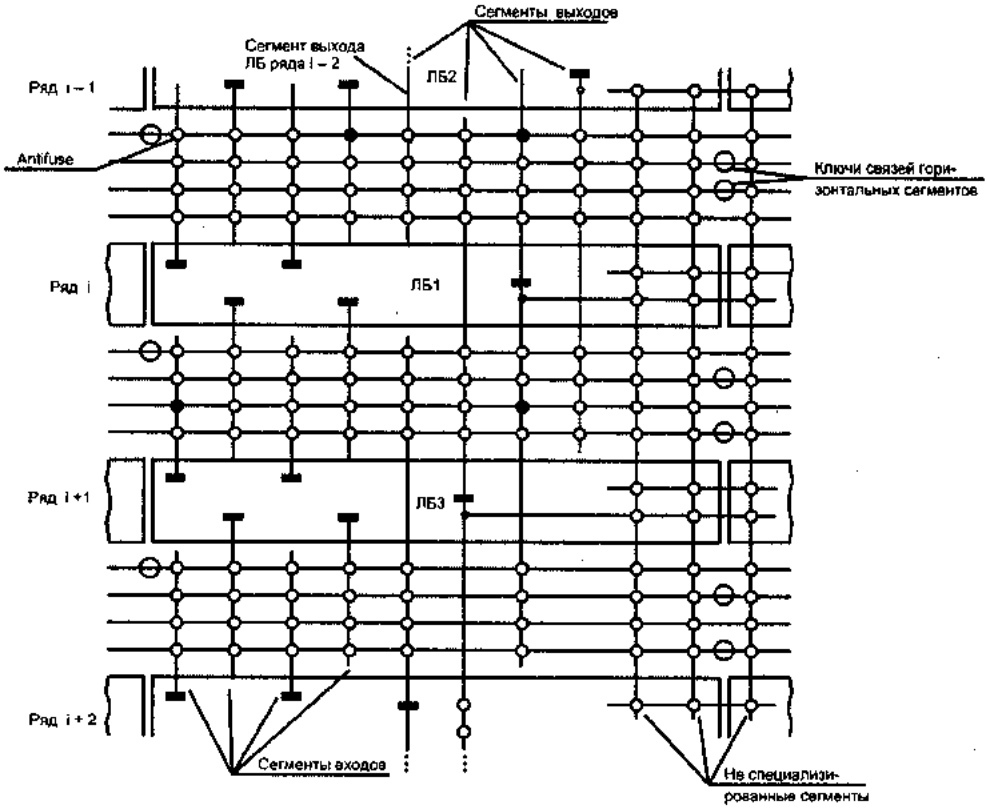


Рис. 9.16. Система коммутации FPGA фирмы Actel
(для упрощения рисунка пересечения горизонтальных и вертикальных линий без их соединения, т. е. незапрограммированные перемычки, показаны просто светлыми кружками)

В каждом пересечении сегментов предусмотрена программируемая перемычка, позволяющая соединять эти сегменты. Такая система коммутации обеспечивает разнообразие вариантов соединения ЛБ между собой. Выход какого-либо ЛБ соединяется с теми горизонтальными сегментами, которые связаны с входами других ЛБ, получающих сигнал от данного выхода.

На рис. 9.16 показаны ЛБ с четырьмя входами, имеющими выводы в ближайший трассировочный канал. Кружками обозначены программируемые перемычки, позволяющие создавать связи между линиями пересечения. Выходы ЛБ соединены с вертикальными сегментами, пересекающими два канала выше данного ряда и два канала ниже его. Имеются ключи, связывающие при необходимости концы горизонтальных сегментов друг с другом для удлинения линий связи.

В каналах имеются также непрерывные по всей длине сегменты, один из которых заземлен, а другой соединен с источником питания, что позволяет подавать на любой из входов ЛБ сигналы логического нуля или логической единицы. В вертикальных направлениях идут также неспециализированные сегменты, пересекающие несколько рядов ЛБ и трассировочных каналов. Каждый такой вертикальный сегмент может соединяться с горизонтальными, которые он пересекает. К таким сегментам есть связи от выходов соседних ЛБ. Наличие неспециализированных вертикальных сегментов увеличивает трассировочную способность системы коммутации.

На рис. 9.16 для примера показаны зачерненными кружками те перемычки, которые должны быть запрограммированы для подачи сигнала с выхода ЛБ1 на входы блоков ЛБ2 и ЛБ3.

В семействах FPGA фирмы Actel экономно реализуется *адресация программируемых перемычек*. Чтобы запрограммировать перемычку, т. е. замкнуть ее, к ней следует приложить повышенное напряжение $U_{пр}$. Это осуществляется следующим образом. Вначале выполняется предзаряд всех сегментов напряжением $U_{пр}/2$. Для этого в схеме имеются специальные транзисторные ключи, включенные параллельно перемычкам и используемые только при программировании и тестировании FPGA. В рабочих режимах ключи заперты и практически не влияют на работу схемы. При замыкании всех этих ключей сегменты соединяются в единые линии, которым и задают необходимые потенциалы. Для замыкания перемычки воздействуют на линии строки и столбца, в пересечении которых находится перемычка. Одна из этих линий заземляется, а другая подсоединяется к напряжению $U_{пр}$. Как видно, при этом только перемычка на пересечении адресующих линий попадает под напряжение $U_{пр}$. Все остальные попадают под напряжение $U_{пр}/2$, не пробивающее перемычку. Транзисторы логических блоков и блоков ввода/вывода, находящиеся в контакте с сегментами и при программировании перемычек попадающие под повышенное напряжение, специально проектируются с необходимой электрической прочностью. Тестирование FPGA производится многократно — до, во время и после программирования.

Система межсоединений FPGA фирмы Xilinx. Это иерархическая система, включающая в себя связи общего назначения (General-Purpose Interconnects), длинные линии (Long Lines), прямые связи (Direct Interconnects), линии тактирования (Clock Lines).

Не все перечисленные разновидности связей встречаются одновременно в одной FPGA. Связи общего назначения имеются у всех FPGA, а прямые связи, например, не у всех.

Связи общего назначения FPGA, типичные для микросхем фирмы Xilinx, показаны рис. 9.17, *а*. В этой системе коммутации переключательные блоки ПБ расположены на пересечении горизонтальных и вертикальных трассировочных каналов, каждый из которых имеет восемь линий.

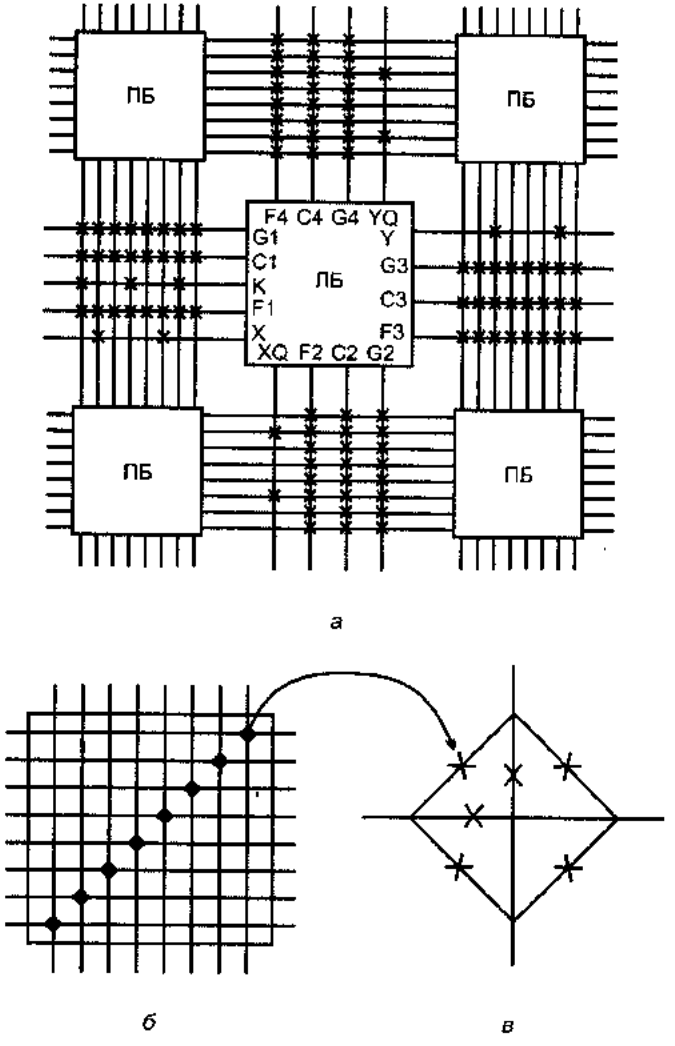
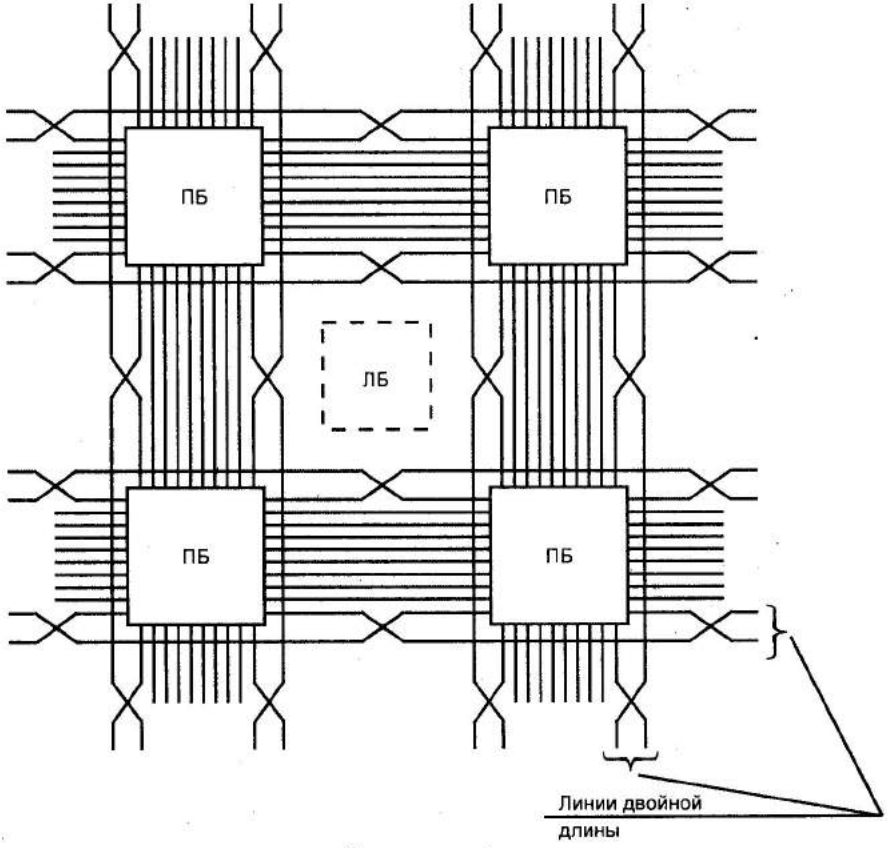


Рис. 9.17. Схема связей общего назначения с линиями одинарной длины (*а*) и схема переключательного блока (*б*, *в*) FPGA семейства XC4000E

Линии могут иметь одинарную длину (соединяя соседние переключательные блоки ПБ) или двойную (соединяя ПБ через один для сокращения числа ПБ в длинных путях). На рис. 9.17, *а* показана схема с одинарными линиями. Связи общего назначения позволяют подводить сигналы к разным сторонам логического блока ПБ. Крестиками отмечены программируемые точки связи.

Структура одного ПБ показана на рис. 9.17, *б*. Она позволяет передавать сигналы влево-вправо или вверх-вниз между смежными одинарными линиями, а также изменять направление передачи сигнала. Схема, соответствующая зачерненному квадрату (рис. 9.17, *в*), показана отдельно. Видно, что для обеспечения перечисленных передач в эту схему должны входить 6 ключевых транзисторов. Прохождение сигналов через ПБ вносит в процесс распространения сигнала задержку, зависящую от конкретного пути, что создает проблему возможных гонок сигналов и сбоев в работе схемы.



а

Рис. 9.18. Схема связей общего назначения с линиями двойной длины FPGA XC4000E (*а*)

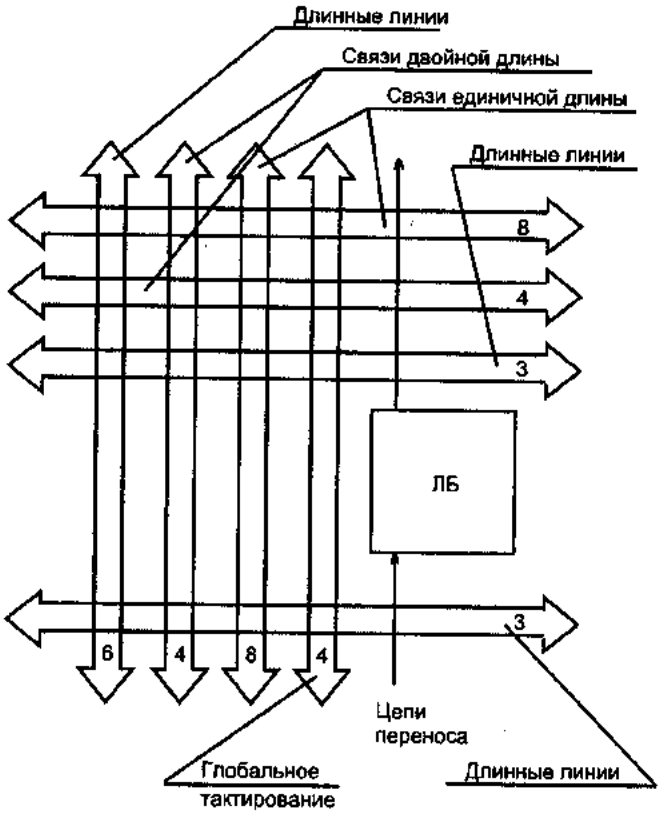


Рис. 9.18. Общие ресурсы связей этой микросхемы (б)

Для ускорения и упрощения дальних передач приняты специальные меры. Наряду со связями общего назначения, показанными на рис. 9.17, имеются связи с линиями двойной длины, в которых переключательные блоки соединены через один, что уменьшает их число при дальних передачах. Фрагмент таких связей представлен на рис. 9.18, *а*.

Для передач на большие расстояния с очень малой задержкой или для передач на разные приемники с малым расфазированием сигналов служат длинные линии (здесь термин "длинные линии" имеет прямой смысл, и его не следует путать с аналогичным термином, употребляемым при согласовании волновых сопротивлений в ином смысле). Длинные линии пересекают кристалл вдоль или поперек по всей его длине или ширине (на рисунках не показаны).

В микросхемах фирмы Xilinx применяются несколько типов длинных линий: горизонтальные и вертикальные линии (по несколько на каждую строку и столбец логических блоков), линии для тактирования блоков ввода/вывода (по две линии вдоль блоков ввода/вывода), так называемые глобальные линии с выходами на определенные БВВ, и линии для распределенных дешифраторов. При этом на каждый ЛБ приходится по 8 горизонтальных и вертикальных связей с линиями одинарной длины, по 4 с линиями двойной длины, по 6 горизонтальных и вертикальных длинных линий, 4 вертикальных глобальных длинных линии и 2 линии (вертикальных) для образования цепей переноса при построении сумматоров, счетчиков и т. д. Всего на каждый логический блок приходится 24 вертикальных линии и 18 горизонтальных (рис. 9.18, *б*).

Обобщенная структура FPGA

Ряд факторов, в первую очередь различия в зернистости логических блоков, обуславливают неоднозначность топологической структуры FPGA. Так, например, для FPGA с логическими блоками в виде транзисторных цепочек внутренняя область выполняется в виде совокупности строк, образуемых этими цепочками (строчные FPGA). Преобладающей в настоящее время стала структура, показанная в упрощенном виде на рис. 9.19, *а*.

Во внутренней области FPGA расположены по строкам и столбцам конфигурируемые логические блоки КЛБ (для упрощения рисунка показано малое число КЛБ, в реальных FPGA их число на порядки больше, это же относится и к числу других блоков, показанных на рис. 9.19, *а*). Кроме КЛБ FPGA содержит переключательные блоки ПБ (SB, Switch Blocks) и связные блоки СБ (CB, Connection Blocks), обеспечивающие коммутацию КЛБ.

Связные блоки реализуют соединения выводов КЛБ с линиями вертикальных и горизонтальных трассировочных каналов общего назначения. Схема СБ показана увеличенно (раскрыта) на рис. 9.19, *б* на примере связей двух соседних по горизонтали КЛБ с вертикальным трассировочным каналом. На пересечениях линий выводов КЛБ с линиями каналов размещаются программируемые точки связей, одна из которых показана кружком в укрупненном виде. Программируемые точки связей имеются во всех пересечениях, но во избежание загромождения рисунка они не изображены. С помощью СБ сигналы КЛБ выводятся в трассировочные каналы.

Переключательные блоки коммутируют линии трассировочных каналов. Их устройство рассмотрено ранее (см. рис. 9.17, *б*). Они позволяют переключать линии передачи сигналов, поступающих в блок, на любое направление (прямо, влево, вправо).

Средства коммутации, связанные с блоками ввода/вывода, сведены в так называемый *VersaRing*, окружающий внутреннюю область кристалла. Они связывают БВВ с каналами трассировки КЛБ. С помощью средств VersaRing

можно перераспределять внешние контакты микросхемы относительно выводов, сформированных во внутренней области кристалла. Это повышает функциональную гибкость и универсальность FPGA, при модификациях проекта позволяет сохранять неизменной распайку контактов микросхемы на плате.

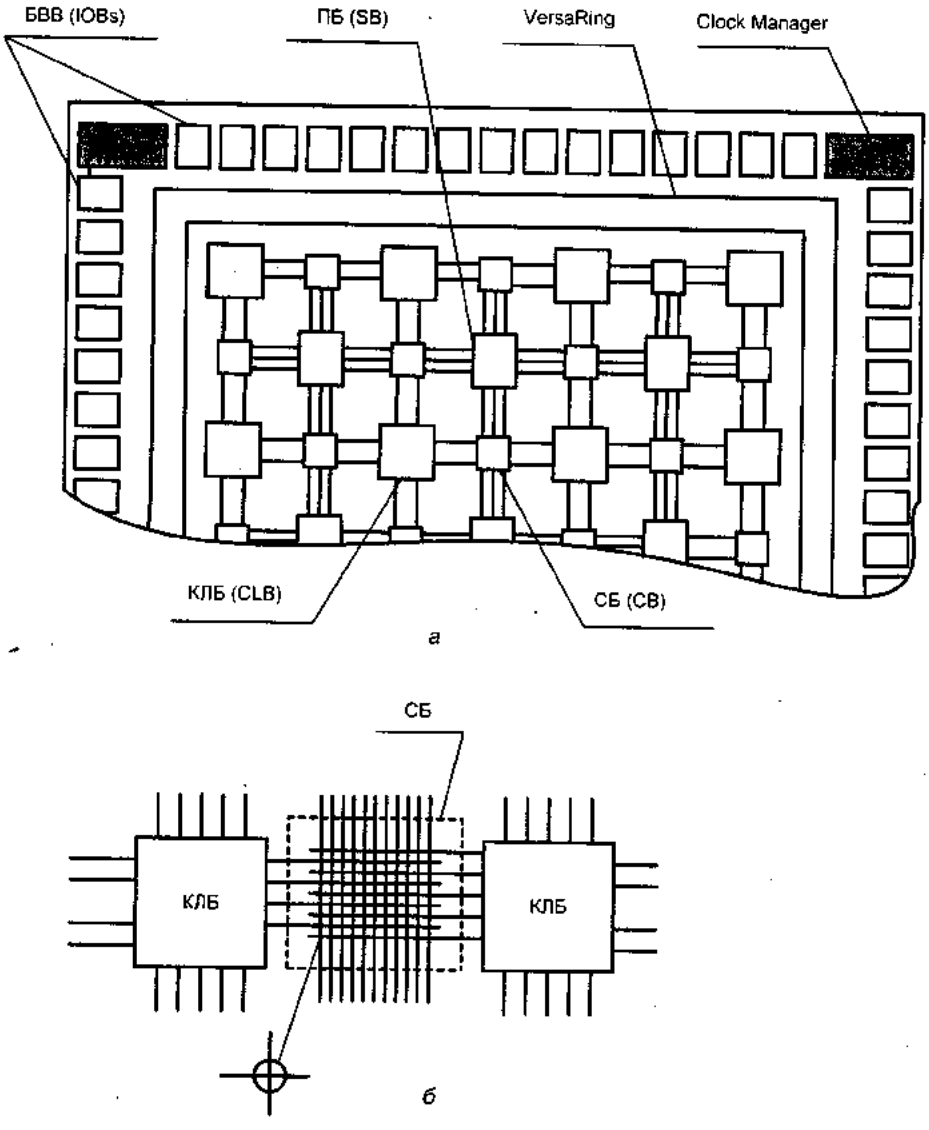


Рис. 9.19. Обобщенная структура FPGA (*а*) и схема связного блока (*б*)

В углах кристалла обычно размещают *специализированные устройства*, в быстродействующих FPGA прежде всего средства управления тактовыми сигналами, к которым относятся блоки DLL или PLL, рассмотренные в *главе 1*. На рис. 9.19 блок указанного назначения обозначен термином Clock Manager. В угловых областях кристалла нередко помещают и такие устройства, как генераторы тактовых сигналов, схемы обслуживания граничного сканирования кристалла по интерфейсу JTAG и др.

На упрощенной структуре рис. 9.19 не отображены такие средства коммутации, как ПБ со связями двойной длины для коммутации КЛБ, прямые связи между соседними КЛБ и длинные линии для передачи сигналов на большие расстояния. Эти средства коммутации рассмотрены ранее при описании FPGA фирмы Xilinx.

§ 9.4. СБИС программируемой логики с комбинированными архитектурами

Стремление объединить достоинства CPLD и FPGA привело к разработке микросхем комбинированной (смешанной) архитектуры, первыми представителями которых были семейства FLEX 8000 и FLEX 10K фирмы Altera (материал этого параграфа основан на рассмотрении микросхем этой фирмы). Линия развития СБИС ПЛ с комбинированной архитектурой нашла дальнейшее отражение в разработках следующих, более сложных, семейств фирмы Altera (APEX, Mercury, Stratix), созданных на основе новых технологических достижений. Схемотехнический и технологический опыт разработки новых семейств был затем применен и к "старым", более "скромным" архитектурам с целью улучшения их параметров (заметим, что описанная ситуация "римейков" достаточно типична и для других фирм). Так, в частности, после разработки семейства APEX фирмой Altera было выпущено семейство ACEX 1K, по архитектуре подобное FLEX 10K, но более современное по технологии. После семейства высшего уровня сложности Stratix — семейство Cyclone, более простое и дешевое. В первых комбинированных архитектурах черты CPLD были выражены достаточно сильно. По мере усложнения СБИС ПЛ их архитектуры сдвигаются в сторону FPGA.

Структура СБИС ПЛ с комбинированной архитектурой

Фрагмент структуры микросхем ACEX 1K приведен на рис. 9.20. По нему можно судить и о схеме в целом, поскольку она составлена повторением фрагментов, идентичных показанному.

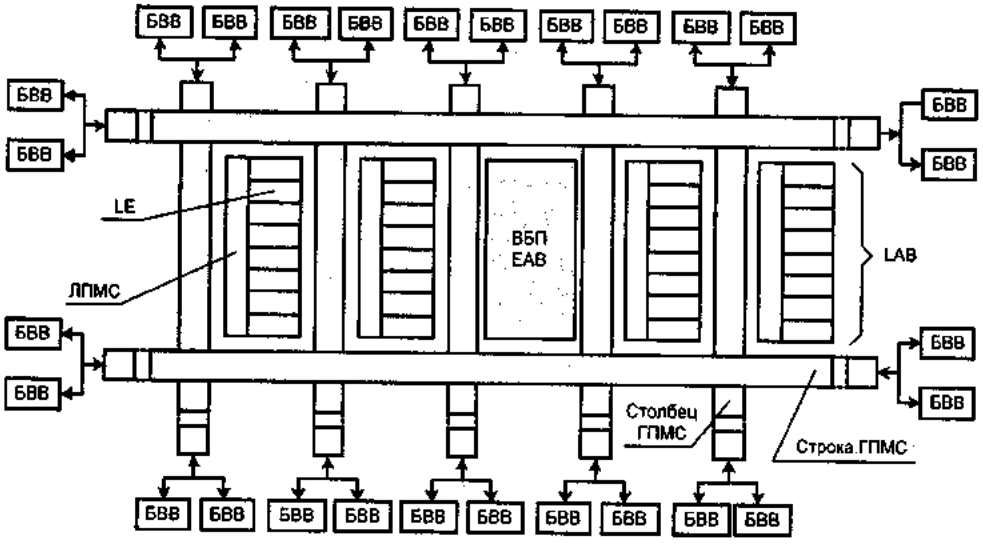


Рис. 9.20. Фрагмент структуры микросхем семейства ACEX 1K

В микросхему входят логические блоки LABs (Logic Array Blocks), содержащие логические элементы LEs (Logic Elements) и локальную программируемую матрицу соединений (ЛПМС) для внутриблочной коммутации логических элементов. Межблочная коммутация реализуется строками и столбцами глобальной программируемой матрицы соединений (ГПМС). При большом числе логических элементов применение единой одноуровневой системы коммутации было бы затруднительным, что и объясняет *двухуровневый характер принятой в микросхеме коммутации*. К концам строк и столбцов ГПМС подключаются блоки ввода/вывода (БВВ).

Начиная с семейства FLEX 10K, в составе СБИС ПЛ появились *встроенные блоки памяти ВБП*, имеющиеся и в микросхемах семейства ACEX 1K. Уточнение "встроенные" объясняется тем, что в микросхемах с LUT-блоками и ранее были ресурсы памяти, называемой *"распределенной"*. Действительно, LUT-блок для воспроизведения функций четырех аргументов (наиболее распространенный в схемах ПЛИС) есть не что иное, как память с организацией 16×1. Его можно использовать и для хранения данных, более того, из таких блоков можно создавать модули памяти большего объема, но за счет сокращения логических ресурсов FPGA и с более сложной реализацией, чем у специализированных блоков встроенной памяти.

Блоки ВБП при данной информационной емкости могут быть организованы в различных вариантах. Например, для блоков семейства ACEX 1K с емкостью 4K возможны организации 2048×2, 1024×4, 512×8 и 256×16. Эти блоки

могут быть ориентированы на реализацию буферов FIFO, иногда на построение двухпортовой памяти, ассоциативной памяти и др. Отдельные ВБП могут объединяться для создания более емкой памяти и применяться не только для хранения данных, но и как *табличные преобразователи для воспроизведения сложных функций* с числом аргументов 8—10 (в частности, на них строят арифметико-логические устройства АЛУ, перемножители 4×4, обладающие высоким быстродействием, и др.). В английской терминологии блоки ВБП семейств FLEX и ACEX называются EABs (Embedded Array Blocks), что точнее, т. к. функции встроенных блоков не ограничиваются только хранением данных.

Комбинирование признаков CPLD и FPGA в структуре микросхем ACEX 1K выражается в следующем:

- логические блоки расположены по строкам и столбцам и содержат логические элементы LUT-типа (табличные функциональные преобразователи), что характерно для FPGA;
- коммутация логических элементов в пределах логического блока осуществляется с помощью программируемой матрицы не сегментированных соединений, что характерно для CPLD, а коммутация самих логических блоков LAB производится с помощью ГПМС, относящейся к числу специальных трассировочных средств, называемых FTI (Fast Track Interconnect), которые хотя и имеют горизонтальные и вертикальные каналы (как и трассировочные ресурсы FPGA), но не используют в них сегментированных линий.

Логические элементы

Основа логических элементов микросхем ACEX 1K (рис. 9.21) — четырехвходовые функциональные преобразователи (ФП) табличного типа (LUT-блоки). ФП с четырьмя входами имеет 16 бит памяти и организуется в варианте 16×1. Те же самые 16 бит можно использовать в виде двух табличных ФП с организацией 8×1, реализующих две функции трех переменных, что отвечает потребностям построения схем сумматоров, счетчиков и т. д.

Особенность логических элементов — наличие специальных *трактов переноса и каскадирования* с прямыми и скоростными линиями связи логических элементов по этим трактам. Эти тракты могут распространяться и за пределы логического блока (на рис. 9.20 такие тракты не показаны).

Длинные цепочки переносов в пределах нескольких LAB формируются соединением четных либо нечетных LAB в строке, но не могут пересекать блоки EAB, расположенные в середине строк. Задержка цепи переноса мала (приблизительно 0,2 нс), что делает целесообразным применение простых структур с последовательными переносами даже для некоторых быстродействующих устройств. Необходимые для данного проекта цепи переноса соз-

даются компиляторами САПР или же могут быть заданы проектировщиком при вводе проекта в систему автоматизированного проектирования.

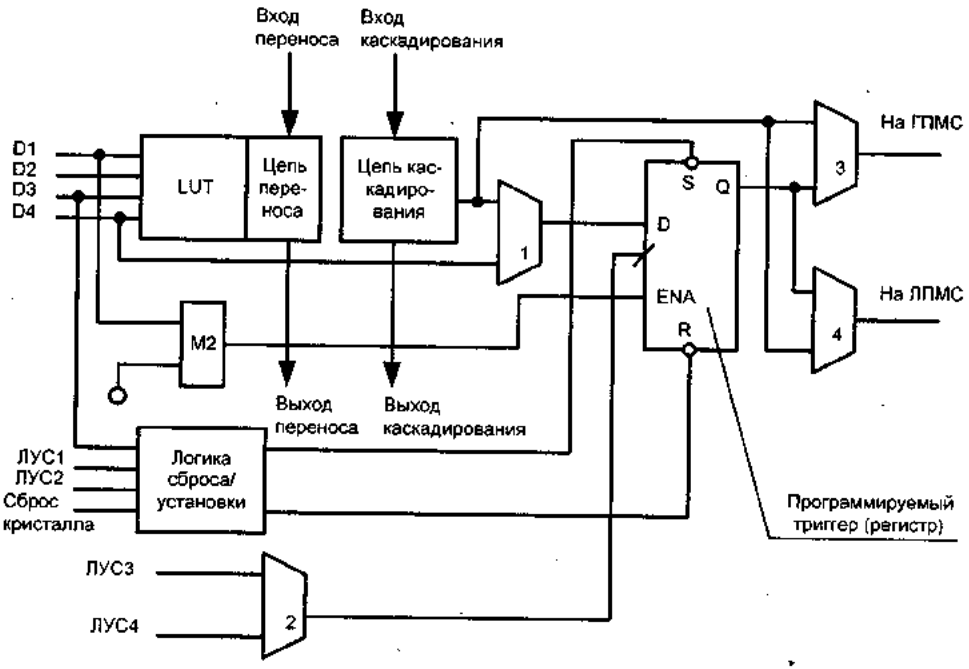
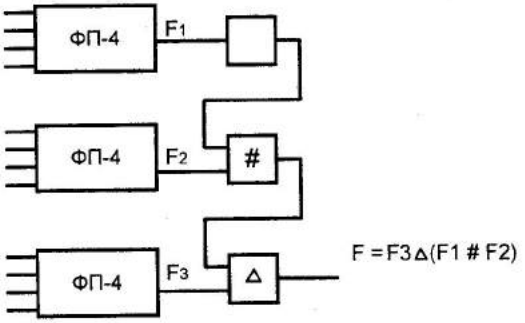


Рис. 9.21. Схема логического элемента микросхем семейства ACEX 1K

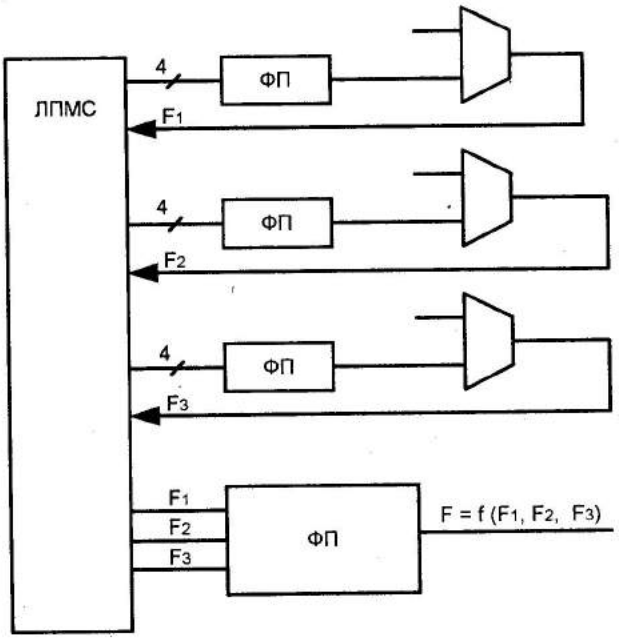
Цепочки каскадирования используются *для получения функций с числом аргументов более 4*. Эти цепочки связывают все логические элементы в пределах LAB и сами блоки LAB в одной и той же строке. При каскадировании можно реализовать функции с большим числом аргументов, используя их декомпозицию. При этом LUT-блоки соседних элементов используются для параллельного во времени вычисления частичных функций, а цепи каскадирования образуют последовательную цепочку для объединения частичных функций (рис. 9.22, *а*). В цепочке каскадирования объединяющими могут быть элементы И либо ИЛИ. Каждый элемент цепочки добавляет для воспроизводимой функции четыре аргумента и вносит дополнительную задержку в 0,6 нс. Цепочки каскадирования, как и цепочки переносов, могут формироваться автоматически компилятором САПР или вручную самим проектировщиком при вводе проекта в САПР.

Функции многих переменных можно получить и другим способом, используя *обратные связи*. При этом сначала вырабатывается некоторая функция четы-

рех переменных, затем она вводится в качестве одного из входов в другой ЛЭ и т. д. В результате вычисляется "функция от функций" с числом аргументов, превышающим 4 (рис. 9.22, *б*).



а



б

Рис. 9.22. Способы воспроизведения функций многих переменных методами каскадирования (*а*) и обратных связей (*б*)

Синхронный триггер в схеме рис. 9.21 может быть сконфигурирован как триггер типа D, T, JK, RS. Сигналы управления тактированием, сбросом и установкой могут поступать от глобальных линий, контактов ввода/вывода общего назначения или от внутренних логических схем. Таким образом, с помощью программирования можно задавать несколько режимов воздействия на триггер по входам тактирования, S и R. Аббревиатура ЛУС означает "локальный управляющий сигнал". В зависимости от программирования мультиплексора MUX1 в линии входа триггер может быть использован для фиксации значений функций, выработанных в логическом элементе, или как отдельный элемент с входом от линии D4. При выработке чисто комбинационных функций триггер в цепь передачи сигнала не включается, и сигнал по обходной линии и через выходные мультиплексоры поступает на выходы непосредственно от LUT-блока. Один из выходных сигналов логического элемента через программируемые мультиплексоры подается в локальную схему межсоединений в вариантах комбинационного (с обходом триггера) или регистрового (с триггера) выходов, а другой в тех же вариантах — в строку или столбец межсоединений типа FTI (Fast Track Interconnect). Два выхода логического элемента могут управляться независимо. Например, выход логического элемента может питать один из выходов, тогда как другой выход можно получать с триггера. Эта возможность функционально обогащает логический элемент, поскольку *триггер и LUT-блок могут формировать две независимые функции.*

Логические элементы имеют 4 базовых режима работы:

- нормальный;
- арифметический;
- реверсивного счетчика;
- сбрасываемого счетчика.

Каждый режим характеризуется своим вариантом использования схемных ресурсов логического элемента. Для реализации тех или иных устройств компилятор САПР автоматически выбирает подходящий режим. При необходимости проектировщик может самостоятельно организовывать специальные режимы, оптимально используя ресурсы логического элемента для данного проекта. Основной задачей нормального режима служит воспроизведение функций, причем число аргументов может доходить до четырех. Назначения остальных режимов понятны из их названий.

На рис. 9.23 показана реализация *n*-разрядного сумматора, иллюстрирующая сущность арифметического режима. Сумматор построен на n + 1 логических элементах, работающих в арифметическом режиме. В схеме сумматора организуется цепь переноса.

LUT-блок, имеющий исходную организацию памяти 16×1, разбивается на две части с организациями 8×1, способные воспроизводить функции трех

переменных. Одна часть вырабатывает сумму в данном разряде, причем результат, который может быть как комбинационным, так и регистровым, направляется на выход логического элемента.

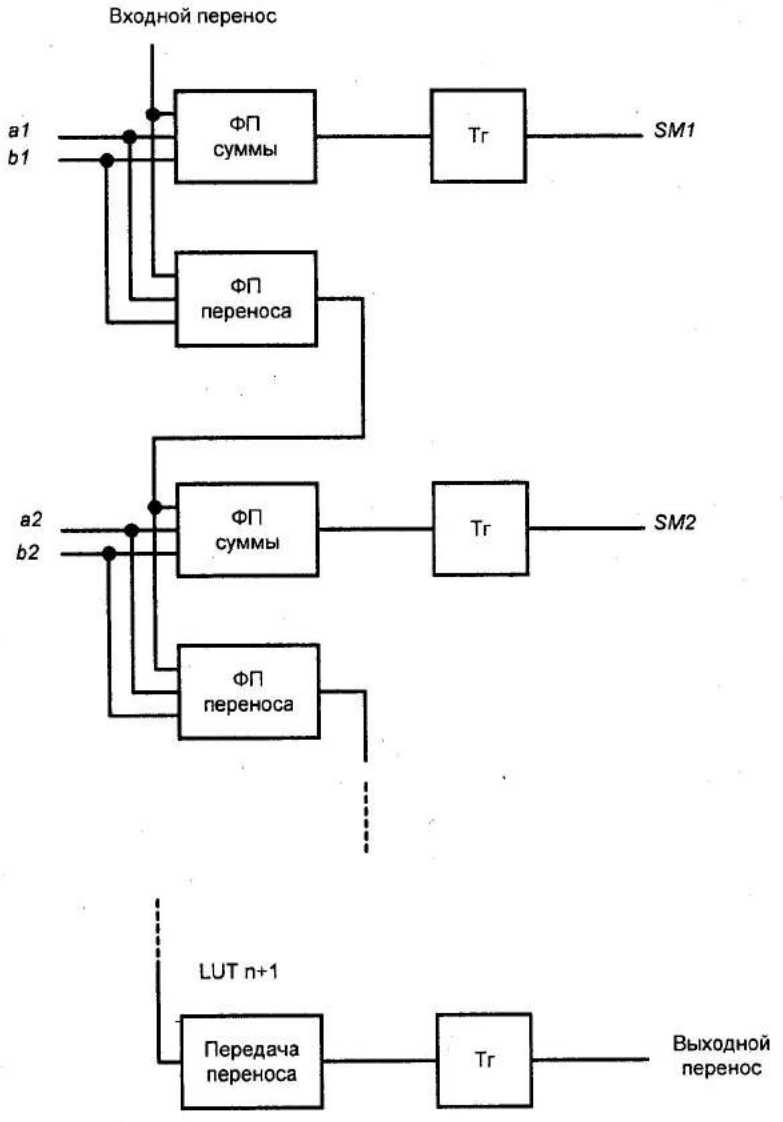


Рис. 9.23. Реализация сумматора на основе арифметического режима работы логических элементов

Другая часть вырабатывает выходной перенос, зависящий от тех же аргументов. Сигнал переноса направляется на вход переноса следующего разряда. Окончательный перенос выводится через логический элемент и может быть использован как сигнал общего назначения.

Логические блоки и их коммутация

В схему логического блока (рис. 9.24) входят 8 логических элементов LE 1—LE 8 с цепями переноса и каскадирования, схемы передачи сигналов управления логического блока и схемы, образующие систему локальных межсоединений.

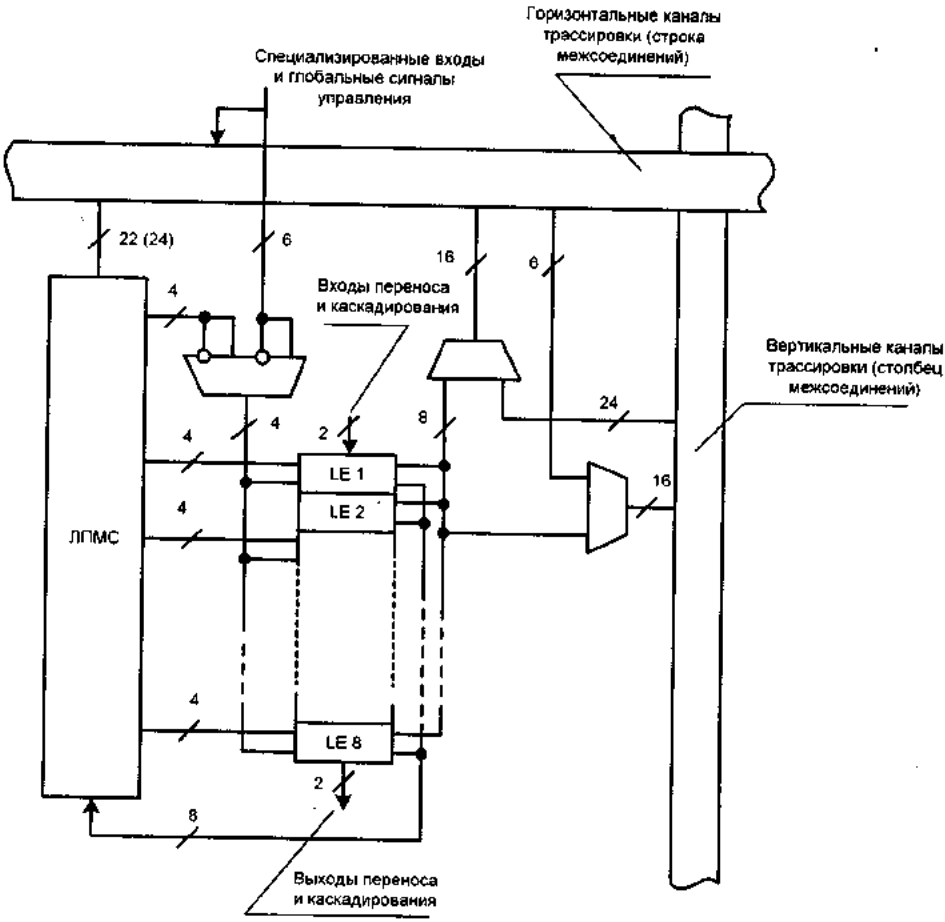


Рис. 9.24. Логический блок микросхем семейства ACEX 1K

На рис. 9.24 схема изображена в сжатом виде. В ней показаны несколько мультиплексоров с многоразрядными выходами. Реально каждый такой мультиплексор представляет собой группу мультиплексоров. Например, под мультиплексором с выходом, подключенным к строке межсоединений и имеющим 16 выходов и 32 = 8 + 24 входов, следует понимать группу из восьми пар мультиплексоров размерности "4–1", в которой каждая пара соответствует одному логическому элементу. Оба мультиплексора пары получают на один из входов выходной сигнал логического элемента, а на три других — сигналы от различных линий столбца межсоединений, причем каждая пара мультиплексоров принимает сигналы от шести линий столбца. Таким образом, этот мультиплексор условно отображает возможности вывода сигналов от логических элементов в строку межсоединений FTI и передачи сигналов из столбца FTI в строку. Второй многоразрядный мультиплексор, связанный по входам с выходами логических элементов и линиями строки межсоединений, передает сигналы логических элементов в столбец FTI или передает сигналы из строки межсоединений в столбец.

Каждая строка логических блоков обслуживается выделенными линиями строки межсоединений. Эти линии могут питать контакты ввода/вывода и передавать сигналы между логическими блоками строки. Линии столбцов передают сигналы между строками и могут передавать их на контакты ввода/вывода. Имеется возможность переключать цепи доступа к линиям строки или столбца от данного логического элемента к смежному, что повышает гибкость системы коммутации и позволяет более эффективно использовать ее ресурсы.

В каждом блоке вырабатываются четыре локальных управляющих сигнала (ЛУС) с программируемой полярностью, которые могут быть использованы во всех восьми логических элементах LE 1—LE 8. Два из них могут служить тактирующими, другие два — сигналами для сброса и установки триггера. Возможности тактирования велики, а именно: может быть использован сигнал от специального входного контакта, сигнал глобального тактирования, сигналы от контактов ввода/вывода или внутренние сигналы (через локальную систему межсоединений).

Линии горизонтального и вертикального трактов FTI (второго уровня коммутации) непрерывны и пересекают весь кристалл. Ресурсы этих средств коммутации характеризуются следующими цифрами:

- число горизонтальных трасс (строк) 3—12 (здесь и далее диапазоны значений параметров ограничены цифрами для младшего и старшего представителей семейства);
- число каналов (линий) в строке 144—312;
- число вертикальных трасс (столбцов) 24—52;
- число каналов (линий) в столбце 24.

Встроенные блоки памяти

В микросхемах ACEX 1K встроенные блоки памяти ВБП (EABs) размещены в середине каждой строки логических блоков. Их можно использовать как по прямому назначению, т. е. как статическое ЗУ, так и для реализации ПЗУ и логических преобразователей (табличных ФП повышенной размерности путем эмуляции ПЗУ с помощью загрузки таблицы в ОЗУ). Такие ФП дают более эффективные решения в сравнении с реализациями сложных функций средствами логических блоков.

> Например, один ВБП при организации 256×8 реализует перемножитель 4×4, способный работать на более высоких частотах и экономичный по затратам площади кристалла в сравнении с реализацией на логических блоках. Размерности ВБП различных микросхем (2—4 Кбит) позволяют использовать их в качестве логических преобразователей для воспроизведения таких функций, как перемножители, векторные преобразователи, схемы коррекции ошибок и т. п. Подобные функции могут быть скомбинированы для реализации алгоритмов цифровой обработки сигналов (например, цифровых фильтров). Настройка блоков на требуемое функционирование проводится при конфигурировании микросхемы, используются готовые файлы загрузки, соответствующие требуемым функциям.

Блоки встроенной памяти ориентированы также на организацию буферов FIFO, двухпортовой памяти и др. Несколько блоков можно объединять для создания более емкой памяти. Так как блоки памяти расположены на том же кристалле, что и логическая часть схемы, *работа с памятью отличается высоким быстродействием.*

Встроенный блок памяти для режима однопортовой памяти показан на рис. 9.25.

В структуре ВБП кроме модуля памяти RAM/ROM имеется несколько синхронных D-триггеров (регистров) и программируемых мультиплексоров. Локальная программируемая матрица соединений ЛПМС получает 2226 сигналов от строки глобальной матрицы ГПМС. Регистры 1 и 2 программируются для передачи в модуль данных и адресов разной разрядности в зависимости от заданной конфигурации памяти. В блоке с емкостью 2 Кбит разрядность данных может изменяться от 1 до 8, а разрядность адреса от 11 до 8. Запись в память в зависимости от программирования мультиплексоров 46 может быть синхронной (от регистров по сигналам тактирования) или асинхронной (непосредственно от ЛПМС).

Сигналы управления регистрами поступают от глобальной шины управляющих сигналов с возможностью выбора их полярности (мультиплексоры 1—3). Выходные сигналы блока с помощью двух программируемых мультиплексоров (на рисунке не показаны) могут передаваться как на линии строки, так и на линии столбца ГПМС в тактируемом или асинхронном вариантах.

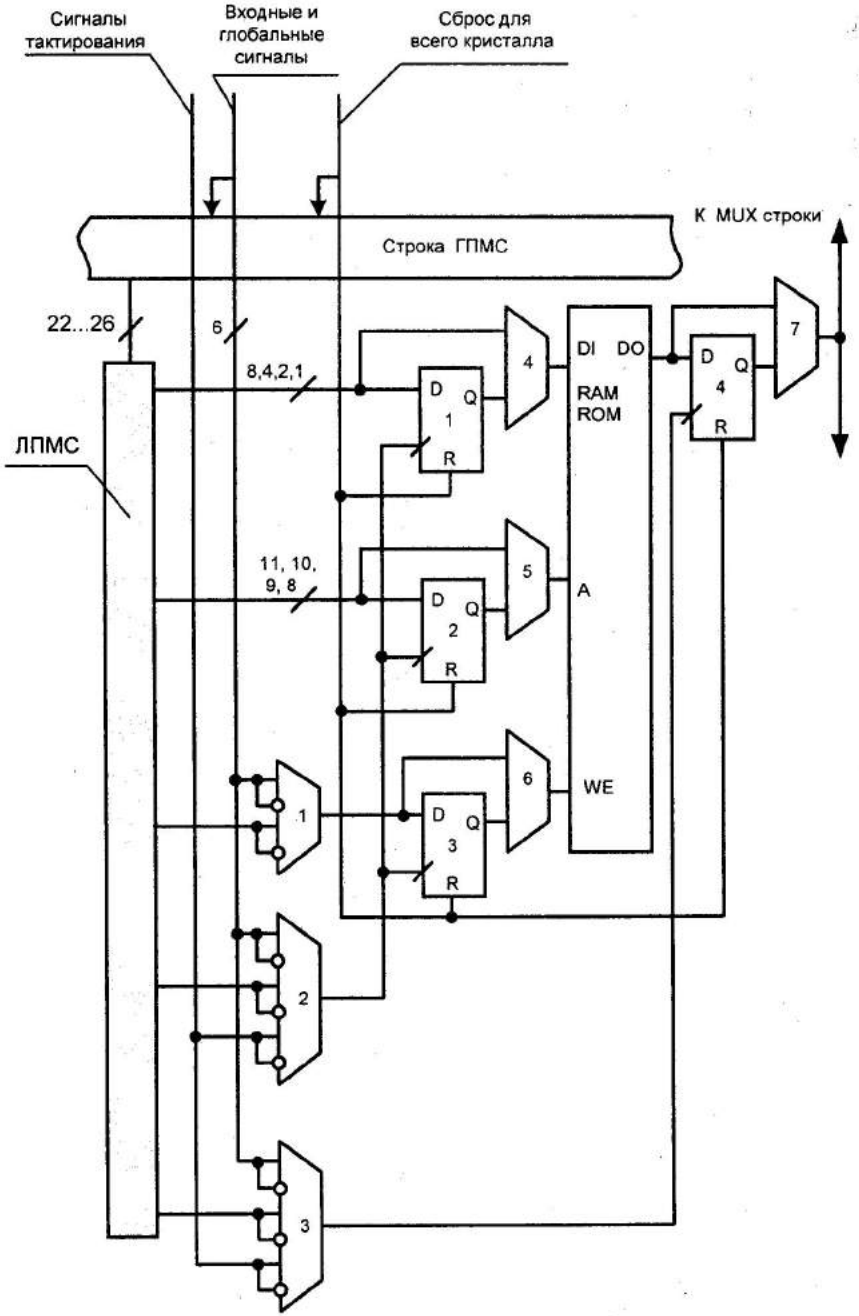


Рис. 9.25. Структура встроенных блоков памяти в микросхемах семейства ACEX 1K (для однопортового режима)

Блоки встроенной памяти семейства ACEX 1K могут быть организованы для работы в однопортовом или двухпортовом режимах. Во втором случае возможно раздельное тактирование для операций чтения и записи, а также раздельные сигналы разрешения тактирования. С помощью двух блоков EAB строятся двунаправленные двухпортовые ОЗУ, в которых реализуются одновременно два порта чтения или записи. Для управления входными и выходными регистрами блока могут быть использованы различные сигналы тактирования и его разрешения.

Блоки ввода/вывода

Блоки ввода/вывода (БВВ) микросхем ACEX 1K относительно сложны. В каждый блок входят программируемый буфер, три триггера типа D с управляющими входами тактирования и его разрешения и 12 мультиплексоров, подавляющее большинство которых являются многовходовыми.

БВВ связывают внешние контакты микросхемы с линиями горизонтальных и вертикальных трассировочных контактов, четырьмя специализированными входными линиями, двумя специализированными линиями тактирования и так называемой периферийной управляющей шиной, имеющей 12 линий и снабжающей БВВ управляющими сигналами.

Сложность БВВ объясняется реализованными в нем возможностями широкого выбора вариантов тактирования триггеров и его разрешения, а также сигналов разрешения работы буфера. С помощью периферийной шины управляющих сигналов возможны до восьми вариантов выбора сигнала разрешения работы буфера OE (Output Enable), до шести вариантов сигнала разрешения тактирования CE (Clock Enable), по два варианта выбора сигналов тактирования и сброса триггеров.

Как и в других современных микросхемах программируемой логики, в семействе ACEX 1K реализованы функции управления параметрами тактирующих сигналов (ClockLock и ClockBoost), обеспечения множества стандартов сигналов ввода/вывода (Multivolt Input/Output), фиксации потенциалов выводов при конфигурировании схемы, а также эмуляции внутренних шин с тремя состояниями.

Более подробные сведения о микросхемах семейства ACEX 1K, как и сведения о других микросхемах фирмы Altera приведены в *Приложении 3*.

§ 9.5. Программируемые аналоговые и аналого-цифровые схемы

Общие сведения

Многие управляющие системы оперируют с информацией, представленной не только в цифровой, но и в аналоговой форме. Аналоговые сигналы вырабатываются, например, датчиками физических величин. Они же использу-

ются в системах управления как воздействия на исполнительные механизмы. При вводе этих величин в цифровые устройства и при их выводе из них обычно необходимы определенные операции, часто обозначаемые термином End-Front Design (нормирование, фильтрация, аналого-цифровое преобразование, цифроаналоговое преобразование и др.). Таким образом, *задача интеграции всей системы на одном кристалле нередко приводит одновременно и к задаче реализации на этом кристалле программируемых средств обработки аналоговых сигналов.* В последние годы наблюдается устойчивая тенденция к увеличению доли аналоговой и цифроаналоговой частей в общем объеме СБИС ПС.

Аналоговые и аналого-цифровые фрагменты уже давно встраиваются в микропроцессорные системы в виде отдельных микросхем малого и среднего уровней интеграции, использующих дискретные (навесные) операционные элементы. Технология БМК также применяется в области аналого-цифровой техники, но программирование самим пользователем аналоговых и аналого-цифровых схем, целиком реализованных на кристаллах, т. е. создание ПАИС (программируемых аналоговых интегральных схем) до последнего времени практически не было освоено. Трудности освоения аналоговых интегральных схем в значительной мере объяснялись их пониженными точностными возможностями в сравнении со схемами на дискретных компонентах. Блоки аналого-цифровых и цифроаналоговых преобразователей (АЦП и ЦАП) с относительно давних пор встраивались в БИС (фирмы Analog Devices, Intel), но с очень нестабильными результатами. Появление "систем на кристалле" сделало проблему интеграции программируемых аналоговых схем и аналого-цифровых схем и цифровой части системы особенно актуальной. На основе микросхем с программируемыми структурами возможно быстрое проектирование подсистем аналоговой и аналого-цифровой обработки сигналов, их отладка, создание промышленных образцов и быстрый выход на рынок.

Несколько крупных фирм (Lattice Semiconductor, Cypress Semiconductor, Anadigm и др.) уже отреагировали на требования времени и уделили внимание разработкам программируемых аналоговых и аналого-цифровых структур, выполненных как на отдельном кристалле, так и совместно с цифровой частью системы.

Два варианта аналоговой схемотехники

Цифровые сигналы принимают лишь два значения, одно из которых соответствует логической единице, а другое — логическому нулю. Проблема точного задания этих сигналов отсутствует — требуется лишь надежно отличать один из этих сигналов от другого. Совершенно иным является положение в аналоговой технике, где *сигнал должен передавать точное значение величины с погрешностью в десятые или сотые доли процента*, т. е. требуется "дозирование" сигналов с разрешающей способностью в тысячи или даже

более уровней. Традиционно (до конца 70-х — начала 80-х гг. XX в.) роль дозирующих параметров в схемах нормирующих усилителей, сумматоров и т. п. играли *отношения сопротивлений точных резисторов*. Постоянные времени RC (масштабирующие параметры в схемах интеграторов, фильтров и др.) задавались совместно значениями сопротивления точного резистора и емкости операционного конденсатора. Так, например, в известной схеме масштабирующего усилителя, т. е. устройства умножения сигнала, заданного напряжением постоянного тока, на константу используются два точных резистора R1 и R2, от соотношения сопротивлений которых зависит функциональная характеристика схемы, в идеализированном виде имеющая вид

$$U2 = (-R2/R1)U1,$$

где U1 и U2 — входное и выходное напряжения соответственно. Интегратор имеет идеализированную функциональную характеристику вида

$$U2 = (-1/RC)\int U1(t)\,dt,$$

в которой роль масштабирующего коэффициента играет произведение сопротивления резистора входной цепи R на емкость конденсатора цепи обратной связи C.

В схемотехнике с дискретными (навесными) схемными элементами проблема реализации точных резисторов имеет удовлетворительное решение. Для технологии интегральных схем эта проблема намного сложнее, но существует *альтернативное схемное решение*, благодаря которому резисторы имитируются цепями, содержащими коммутируемые (переключаемые) конденсаторы (рис. 9.26).

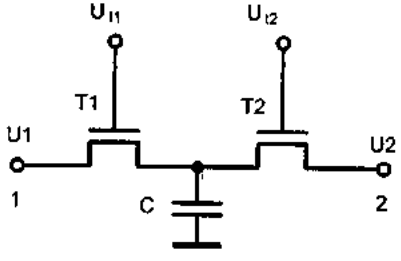


Рис. 9.26. Цепь с переключаемыми конденсаторами, моделирующая резистор

В такие цепи входят конденсатор C и ключевые транзисторы T1 и T2, управляемые тактирующими напряжениями U_{t1} и U_{t2}. Транзисторы T1 и T2 под воздействием тактирующих напряжений замыкаются поочередно, и конденсатор C попеременно заряжается через замкнутый ключевой транзистор до напряжения U1 или U2. В момент замыкания ключевого транзисто-

ра заряд конденсатора изменяется на величину $q = q_1 = q_2 = C(U1 - U2)$. Изменение заряда осуществляется короткими импульсами тока, протекающими через конденсатор при замыкании соответствующего ключевого транзистора.

Среднее значение тока в цепи между точками 1 и 2 составляет величину

$$i = q/T = (U1 - U2)C/T,$$

где T — период тактирующих импульсов.

Из полученного выражения видно, что для средних значений сигналов цепь ведет себя как резистор с сопротивлением $R = T/C$, которое пропорционально периоду тактирующих импульсов (т. е. обратно пропорционально их частоте) и обратно пропорционально емкости. Разница состоит в том, что резистор регулирует поток передаваемого через него заряда плавно, а цепь с переключаемыми конденсаторами — импульсно.

На основе схем с переключаемыми конденсаторами *можно строить разнообразные операционные звенья, аналогичные известным из традиционной аналоговой схемотехники*, заменяя резисторы эквивалентными им цепями. Сопротивления эквивалентных цепочек управляются значениями тактовой частоты $f = 1/T$, которые можно изменять и дистанционно. А это ставит аналоговые блоки в один ряд с цифровыми с точки зрения возможностей программирования в системе на расстоянии (например, через Интернет).

В схемотехнике с переключаемыми конденсаторами строятся *схемы с зависимостью функциональных характеристик только от отношения емкостей, которое может задаваться с высокой точностью*. Именно схемотехника на основе переключаемых конденсаторов дает возможность реализовать АЦП и ЦАП с высоким уровнем параметров. Параметры емкостей мало критичны к изменению температуры и старению. Резко (в сотни раз) снижается площадь, занимаемая цепями с переключаемыми конденсаторами в сравнении с цепями, содержащими точные резисторы. Таковы технологические достоинства схемотехники переключаемых конденсаторов. В то же время применение цепочек с переключаемыми конденсаторами имеет и свои недостатки. В цепях с непрерывными сигналами (без переключаемых конденсаторов) отсутствует проблема отделения полезной информации (среднего значения пульсирующей величины) от сопровождающих ее паразитных высокочастотных составляющих, что благоприятно влияет на динамические характеристики устройств. Кроме того, схемы с непрерывными сигналами имеют лучшие шумовые характеристики.

Практические разработки

Впервые о создании БИС с массивом программируемых пользователем аналоговых элементов объявила фирма Motorola (1997 г.). Эти БИС были анонсированы, но не доведены до промышленного выпуска. В 1999 г. фирма

Lattice Semiconductor выпустила семейство внутрисхемно программируемых (In-System Programmable) аналоговых схем типов ispPAC10 и ispPAC20, а затем и ispPAC80, ispPAC81, ispPAC30. Фирма Cypress Semiconductor выпустила микросхемы класса "система на программируемом кристалле" (PSoC) с реализацией цифровой и аналоговой части в пределах одного кристалла.

В микросхемах фирмы Lattice Semiconductor используются схемы с масштабирующими резисторами, в PSoC фирмы Cypress Semiconductor — оба вида схемотехнических решений, т. е. и схемы с точными резисторами, и схемы с переключаемыми конденсаторами.

Микросхемы семейства ispPAC архитектурно просты (имеют немного конфигурируемых ресурсов и контактов ввода/вывода). Память конфигурации реализована по технологии EEPROM и может загружаться через специально выделенные контакты интерфейса JTAG. Допустимое число циклов перепрограммирования не менее 10000. Конфигурация может быть закрыта от несанкционированного доступа битом секретности. В состав конфигурируемых ресурсов включены не только аналоговые, но и цифро-аналоговые средства (восьмиразрядные цифро-аналоговые преобразователи).

Микросхемы семейств ispPAC содержат входные (инструментальные) усилители IA (Input Amplifiers), выходные усилители OA (Output Amplifiers) и другие блоки. Состав микросхемы ispPAC30 показан на рис. 9.27.

Микросхема имеет единственное напряжение питания 5 В, режимы понижения мощности (в режиме Power Down потребляются микроваттные мощности), автокалибровку внутренних смещений и точное программирование коэффициентов усиления. Важная особенность — *возможность динамической реконфигурации* "на лету", причем не только быстрой, но и без ограничения числа циклов. Для этой цели микросхема снабжена триггерной памятью, допускающей любое число циклов стирания/записи. Наряду с триггерной памятью конфигурации имеется и энергонезависимая память типа EEPROM, постоянно хранящая настройки, из которых пользователь может выбирать требуемую и вводить ее в триггерную память. Среди хранимых в EEPROM настроек есть такая (Preset), которая вводится при включении питания (или при необходимости вернуться к ней в процессе работы). Естественно, что память конфигурации EEPROM можно обновлять в любое время.

Четыре входных усилителя имеют входные импедансы, которые настолько велики (лежат в области гигаомных значений), что практически снимают вопрос нагрузки на источники сигналов, в том числе на датчики исходных физических величин. Шкала входных напряжений у этих усилителей составляет от 0 до 2,8 В, коэффициенты усиления программируются в диапазоне от ±1 до ±10. Два усилителя имеют на своих входах мультиплексоры размерности 2—1. Управляя мультиплексорами, можно ввести в схему 4 различных сигнала, или же подать на мультиплексоры один и тот же сигнал, для которого будут предусмотрены различные тракты обработки.

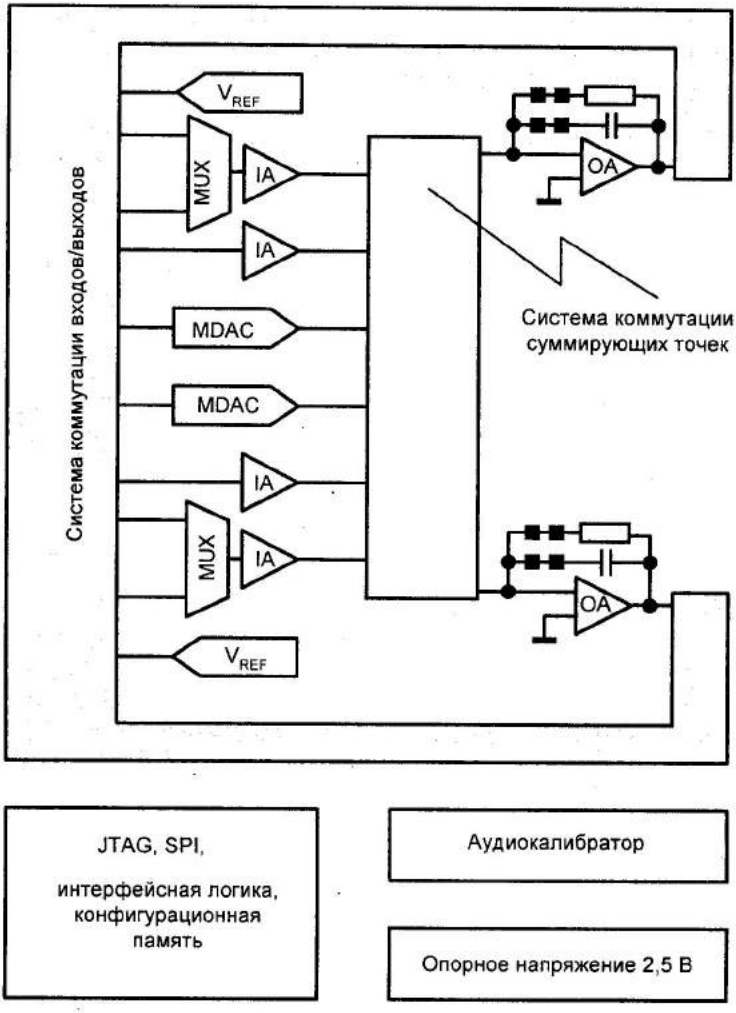


Рис. 9.27. Структура микросхем ispPAC30

Два выходных усилителя OA (Output Amplifiers) с диапазоном выходных напряжений от 0 до 5 В (в однополюсном режиме) имеют произведение коэффициента усиления на полосу пропускания (Gain-Bandwidth Product) 15 МГц. В зависимости от программирования перемычек в цепях обратных связей выходные усилители могут работать в режимах:

- *усилителя* (отключается емкость, точнее, оставляется минимальная емкость для сохранения устойчивого режима работы);
- *интегратора* (отключается резистор);

- *фильтра низкой частоты первого порядка* (включены оба элемента обратных связей). Для проектирования фильтров с различной полосой пропускания можно использовать 7 различных значений емкости С;
- *компаратора* (отключены оба элемента обратных связей). Отсутствие у ОА двух сигнальных входов заставляет выполнять компаратор в виде сумматора входного напряжения и некоторого опорного (эталонного) напряжения другого знака. При этом знак суммы определяет логический выходной сигнал ОА в соответствии с отношением "больше—меньше" между входным и опорным напряжениями. Для компараторов возможно программирование работы с гистерезисом или без него.

Любые из блоков IA (Input Amplifiers) или MDAC (Multiplying Digital-Analog Converters) могут согласно назначениям проектировщика подключаться к суммирующим точкам усилителей ОА, придавая микросхеме максимальную гибкость конфигурирования.

Кроме усилителей в число блоков входят умножающие цифроаналоговые преобразователи MDAC, выходные сигналы которых с высокой точностью пропорциональны как входному цифровому коду, так и входному напряжению. Входное напряжение умножается при этом на коэффициент, не превышающий 1, так что на выход MDAC в зависимости от поданного кода передается часть входного сигнала (от 100% до такой его части, которая соответствует единице младшего разряда кода). Являясь регулируемым аттенюатором входного напряжения, MDAC обеспечивает необходимые коэффициенты передачи в соответствующих цепях схем. Сочетание MDAC, входных усилителей и источников опорных напряжений V_{REF} еще более расширяет эти возможности (иллюстрируется далее).

Источники образцового напряжения $V_{REF} = 2{,}5$ В позволяют выводить рабочую точку усилителей в середину питающего напряжения для восприятия и выработки знакопеременных сигналов. Напряжения V_{REF} применяются и для более общих целей формирования необходимых значений напряжений. От каждого источника V_{REF} можно получить семь уровней напряжения (0,064 В; 0,128 В...2,048 В; 2,500 В). Напряжения могут суммироваться с другими сигналами или вычитаться из них, масштабироваться с коэффициентами от 1 до 10 с помощью усилителей IA и ослабляться до одного из 128 уровней с помощью преобразователей MDAC, имеющих семь информационных разрядов и один знаковый.

Комбинации перечисленных методов формирования точных уровней напряжения дают очень большое число вариантов. Если, например, подать один и тот же сигнал U на входной усилитель и преобразователь MDAC, а оба выхода этих блоков подключить к суммирующей точке усилителя ОА, то от усилителя IA получим целую часть результата (от 1U до 10U), а от MDAC — дробную. В итоге можно сформировать любой уровень напряже-

ния в пределах от −11U до +11U с разрешением, лучшим, чем 0,01U, что дает приблизительно 2500 вариантов.

Выходные функции реализуются усилителями ОА с элементами R и C в цепях обратных связей. Блоки IA, MDAC, V_{REF} предназначены для включения во входные цепи ОА. Во входных цепях ОА для режимов нормирующих усилителей, интеграторов, фильтров должны быть включены сопротивления определенных номиналов. Поэтому возникает вопрос о трактовке цепей с блоками IA и MDAC как некоторых эквивалентов сопротивлений. При этом последовательное включение усилителя IA и резистора R (для микросхем ispPAC30 R = 50 кОм) при единичном коэффициенте усиления трактуется как резистор с сопротивлением R, а при коэффициенте усиления 10 как резистор с сопротивлением 0,1R, т. е. для получения значения эквивалентного сопротивления фактическое его значение нужно разделить на коэффициент усиления усилителя IA. При последовательном включении резистора и преобразователя MDAC для получения эквивалентного значения сопротивления нужно разделить R на коэффициент передачи MDAC. Например, если преобразователь MDAC передает на выход 50% входного напряжения, то эквивалентное сопротивление составит 2R.

Схемы коммутации сигналов ввода/вывода и схемы коммутации суммирующих точек позволяют точно передавать аналоговые значения напряжений и токов, передавать входные сигналы микросхемы к любому усилителю IA или преобразователю MDAC (внешние контакты микросхемы на рис. 9.27 не показаны) и подключать любой IA или MDAC к суммирующей точке любого выходного усилителя. Распределение внешних контактов относительно точек внутренней схемы является гибким. В микросхемах семейства ispPAC чувствительные к помехам цепи выполнены по дифференциальной схеме, защищенной от воздействий других сигналов, что делает качество передачи сигнала независимым от положения линии связи на кристалле и улучшает другие характеристики схем. Из нескольких схем семейства ispPAC можно строить перестраиваемые активные фильтры различных порядков. Микросхемы ispPAC80/81 специально предназначены для создания активных фильтров пятого порядка с программируемыми характеристиками. Могут строиться фильтры с аппроксимацией частотных характеристик по Баттерворту, Бесселю, Чебышеву, фильтры с аппроксимацией эллиптического типа.

Аналоговые блоки следует характеризовать совокупностью их функциональных и точностных характеристик с учетом статических и динамических погрешностей. Укажем основные параметры микросхемы ispPAC30:

- Диапазон входных напряжений усилителей IA от 0 до 2,8 В, напряжение смещения, приведенное к дифференциальному входу при коэффициенте усиле-

ния 10, 60(200)[1] мкВ, его дрейф 50 мкВ/°С. Входное сопротивление 10^9 Ом, входная емкость 2 пФ, ток смещения 1 пА при 25 °С и 200 пА при 85 °С. Приведенная ко входу при коэффициенте усиления 10 плотность напряжения входного шума 70 $\frac{нВ}{\sqrt{Гц}}$ (на частоте 10 кГц).

- Перепад выходного напряжения усилителей ОА 5 В, максимальный выходной ток ± 30 мА.
- Диапазон программируемых коэффициентов усиления от 0 до 20 дБ, их погрешность 1(3)%, дрейф 35 ppm/°С. Отношение сигнал/шум в полосе от 0,1 Гц до 114 кГц 83 дБ, коэффициент гармонических искажений на частоте 10 кГц составляет –85 дБ, на частоте 100 кГц –75 дБ. Малосигнальная полоса пропускания 1.57 МГц, крутизна фронтов 15 В/мкс. Время установления положительного перепада с погрешностью 0,1% составляет 2—4 мкс.
- Разрешающая способность преобразователей MDAC — 128 уровней, интегральная нелинейность 0,25(0,5) цены младшего разряда LSB, дифференциальная нелинейность 1 LSB, напряжение смещения 3 мВ, погрешность коэффициента передачи 1(3)%, полоса пропускания (3 дБ) 1,6 МГц. Знаковый разряд определяет полярность выходного напряжения.
- Опорное напряжение V_{REF} = 2,500 В ± 0,2%, его дрейф 100 ppm/°С, напряжение шумов при полосе 100 кГц составляет 40 мкВ.
- Времена переключения компаратора — 4 мкс при зоне установления 10 мВ и 2,5 мкс при зоне установления 100 мВ.
- Параметры фильтра — диапазон угловых частот 49—619 кГц, погрешности угловых частот 3—5%, дрейф 0,05%/°С.

Относительно скромные точностные возможности микросхем ispPAC, тем не менее, приемлемы для построения на них ряда работоспособных функциональных узлов.

На рис. 9.28 приведена схема фильтра первого порядка, реализованная на кристалле ispPAC30.

Самый распространенный вариант реализации фильтра первого порядка — схема операционного усилителя с параллельной RC-цепочкой в обратной связи. Выходное напряжение такого фильтра с использованием преобразования Лапласа записывается в следующем виде

$$-U_{OUT}(p) = U_{IN}(p)/(1 + pT),$$

где p — комплексная переменная, используемая в преобразовании Лапласа, T — масштабный коэффициент, имеющий размерность времени, а $U_{OUT}(p)$ и $U_{IN}(p)$ — изображения выходного и входного напряжений.

[1] Если параметр указан двумя цифрами, то первая цифра относится к его типовому значению, а вторая (в скобках) — к предельному.

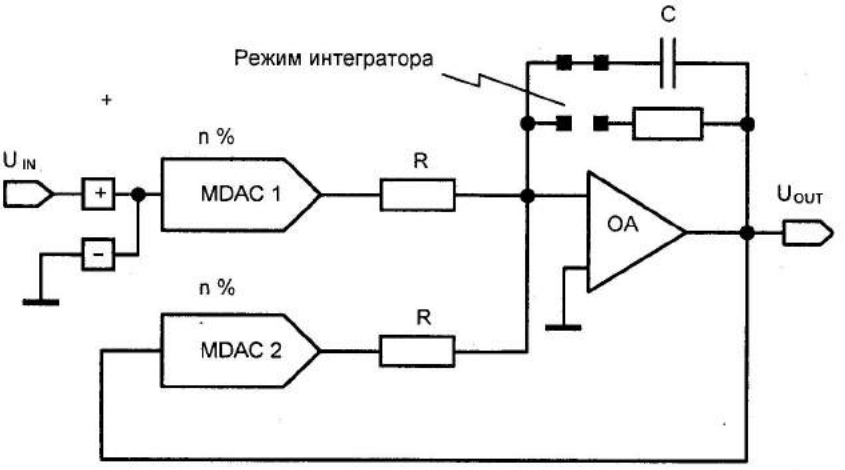


Рис. 9.28. Схема фильтра, реализованного на блоках микросхемы ispPAC30

Это выражение можно преобразовать в следующее

$$(U_{OUT}(p) + U_{IN}(p))/pT = -U_{OUT}(p),$$

где pT — оператор интегрирования.

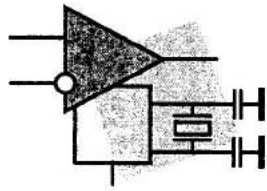
Именно по такому соотношению и реализован фильтр, показанный на рис. 9.28, в котором сумма входного и выходного напряжений интегрируется выходным усилителем (необходимые знаки напряжений и масштабные коэффициенты обеспечиваются программированием блоков MDAC).

Как уже отмечалось, программируемые аналоговые схемы (ПАИС) производятся уже несколькими фирмами. Микросхемы фирмы Cypress Semiconductor будут рассмотрены в *главе 10*. Фирма Anadigm Inc. выпустила микросхемы FPAA (Field Programmable Analog Arrays) для ввода, преобразования и вывода аналоговых сигналов. Схемы имеют небольшие матрицы программируемых аналоговых блоков CAB (Configurable Analog Block) с программированием операционных усилителей и схем их межсоединений. Блоки, выполняют преобразование сигналов в диапазоне частот от нулевой до 2 МГц. Отношение сигнал/шум для широкополосного диапазона составляет 80 дБ, для аудиосигналов 100 дБ. Напряжение смещения на входах операционных усилителей не более 100 мкВ. Напряжение питания 5 В. На процесс реконфигурации схемы затрачивается 100 мкс.

Программируемые аналоговые микросхемы FPAD (Field Programmable Analog Devices) имеются также в серии TRAC фирмы Fast Analog Solutions, входящей в группу ZETEX.

Литература к главе: [5], [16], [23], [36], [41], [42], [54], [58], [I], [III], [VI], [XIV], [XXIV], [XXXIV].

Глава 10



СБИС программируемой логики типа "система на кристалле"

§ 10.1. Общие сведения

Уровень интеграции современных микросхем СБИС ПЛ достиг величин в десятки миллионов эквивалентных вентилей, а их тактовые частоты составляют 500—600 МГц и даже более. На таких кристаллах можно разместить целую сложную систему высокого быстродействия (процессорную часть, память, интерфейсные схемы и др.), их появление стирает грань между элементной базой и аппаратурой.

Стратегическая значимость создания систем на программируемых кристаллах очевидна и может быть проиллюстрирована следующими соображениями:

- микропроцессорные системы — важнейшее универсальное средство решения самых разнообразных задач, они ежегодно реализуются тиражами в десятки миллиардов экземпляров;
- системы, созданные на одном кристалле, имеют наиболее высокие технические и экономические качественные показатели, кроме того, для наиболее сложных современных систем высшей производительности реализация на дискретных компонентах становится вообще нереальной;
- из-за высокой стоимости проектирования сложные системы на одном кристалле реализуемы только при массовом производстве;
- *массовость производства* обеспечивается универсальностью системы, а универсальность *может быть обеспечена только с помощью программируемости*, позволяющей приспособить систему к нуждам конкретных потребителей по целому ряду требований (в частности, по составу и параметрам периферийных устройств).

Для обозначения микросхем типа программируемая система на кристалле или "конфигурируемая система на кристалле" наряду с аббревиатурой SOPC (Systems On Programmable Chip) используется и ряд других, в частности, PSOC

(Programmable Systems On Chip), CSOC (Configurable Systems On Chip), FPSLIC (Field-Programmable System-Level Integrated Circuits) и т. д. Термин SOPC ниже принят как общее обозначение программируемых "систем на кристалле", а при описании конкретных микросхем будут сохраняться наименования фирм-изготовителей.

Интеграция все большего числа функциональных блоков в одном кристалле сопровождается сокращением площади печатной платы, повышением быстродействия устройств и систем и их надежности, уменьшением потребляемой мощности и стоимости. Возможность быстрой реконфигурации схем непосредственно в работающем устройстве открывает перспективы не только эффективной отработки прототипа проекта, но и *создания принципиально новых структур с многофункциональным использованием и динамическим реконфигурированием аппаратных средств.*

Большой значимости создания "систем на кристалле" отнюдь не препятствует то, что с точки зрения архитектуры (структуры) они не представляют собою чего-то принципиально нового. Если понятия CPLD или FPGA обозначают определенные архитектуры микросхем с их специфическими свойствами, то четкого понятия "система на кристалле" в этом аспекте не существует, такие системы архитектурно разнообразны, а термин "системы на кристалле" получил широкое распространение, поскольку имеет существенное значение с точки зрения практических и рекламных факторов.

IP-ядра и типы программируемых "систем на кристалле"

Микросхемы уровня "системы на кристалле" применяются для решения сложных задач. Их проектирование требует больших затрат труда и времени. Подсчитано, что в рамках традиционного проектирования разработка сложной СБИС требует до 500 человеко-лет. Необходимость *снижения трудоемкости и стоимости проектирования* — все более острая проблема микроэлектроники, одним из основных путей решения которой стало создание библиотек схемных решений, предлагаемых проектировщику. Такие библиотеки получили широкое распространение в практике проектирования матричных БИС (МАБИС) на основе базовых матричных кристаллов (БМК). Производители БМК сами разрабатывали библиотеки заранее проверенных и документированных схемных решений, чтобы облегчить использование своих кристаллов системотехниками. Усложнение БМК и СБИС ПЛ сделало проблему создания заранее отработанных схемных решений еще более важной. Созданием таких готовых решений стали заниматься не только изготовители кристаллов, но и специализированные фирмы, разработки которых пригодны для кристаллов разных производителей.

Изготовителями кристаллов и специализированными фирмами были синтезированы многочисленные библиотечные блоки, диапазон сложностей которых очень широк — от уровня функциональных узлов (счетчиков, регистров) до уровня таких устройств, как микропроцессоры, микроконтроллеры или ЦОС-ядра. Библиотечные блоки для СБИС ПЛ могут быть представлены в разной форме. Различают следующие варианты:

- *soft-ядра* или *виртуальные компоненты*. Это файл, который можно интегрировать в описание проектируемого устройства на языках HDL;
- *firm-ядра*, это вариант, близкий к первому, его применяют для блоков, быстродействие которых особенно важно. В этих ядрах задание схемы является более жестким, чем в файле поведенческого описания, свойственного soft-ядрам, поскольку в описании firm-блока предопределены некоторые схемные межсоединения;
- *hard-ядра*, представляющие собою реализованные на кристалле области с фиксированными функциями, что позволяет оптимизировать соответствующие схемы, спроектированные методами заказного проектирования, но не позволяет как-либо репрограммировать эти ядра.

Между soft- и firm-ядрами, с одной стороны, и hard-ядрами, с другой, имеется существенная разница. Hard-ядра приобретаются вместе с микросхемой как ее части и не являются в этом смысле самостоятельным товаром. Soft- и firm-ядра, получившие название *блоков IP* (Intellectual Properties), т. е. *единиц интеллектуальной собственности*, приобретаются как самостоятельные продукты, причем высокой стоимости. Наряду с международным обозначением IP в работах отечественного НИИ "Прогресс" используется русский термин СФ-блоки (сложные функциональные блоки).

В зависимости от характера применяемых ядер можно разделить SOPC на два типа.

- *Однородные* (в англоязычной литературе они иногда обозначаются термином *generic*). В этих SOPC используются soft-ядра или firm-ядра и разные функциональные блоки реализуются идентичными схемотехническими средствами благодаря их программируемости. Однородность здесь характеризует только незапрограммированный кристалл, т. е. понимается как признак конфигурируемости всех его областей, поскольку эти области содержат однотипные программируемые схемные ресурсы. Заметим, что принятое определение "однородные" следует относить лишь к основной части СБИС, в отдельных ее областях вспомогательного характера могут быть "неоднородности", в частности, наличие в СБИС ПЛ встроенной памяти, блоков управления параметрами синхросигналов и некоторых других устройств терминологически не выводит СБИС ПЛ из класса однородных SOPC. Данные для загрузки памяти конфигурации, приводящей к реализации нужного блока (soft-ядра, firm-ядра), поставляются многими фирмами, но имеют достаточно высокую стоимость. В одно-

родных SOPC *реализуемые блоки могут размещаться в разных областях кристалла с помощью их программирования (конфигурирования).*

- *Блочные.* Это определение возникло из-за того, что "системы на кристалле" стали приобретать характерные специализированные области (hard-ядра), жестко выделенные для определенных функций — *аппаратные ядра.* Так как системы разного назначения содержат, тем не менее, типовые части, возникает вопрос о целесообразности введения в СБИС ПС наряду с программируемой логикой специализированных областей с заранее определенными функциями. Таким образом, SOPC блочного типа включают в себя как программируемые, так и *фиксированные области*, в которых реализованы блоки с предопределенными функциями. В других областях кристалла *размещается программируемая пользователем часть*, чаще всего типа FPGA. Характер и сложность аппаратных ядер блочных SOPC со временем изменяются. Вначале аппаратные ядра были довольно простыми, сейчас основным ядром блочных SOPC нередко служит *микропроцессор или микроконтроллер.* Аппаратные ядра обеспечивают работу на максимальных частотах. Например, для технологии с топологической нормой 0,13 мкм аппаратное ядро 32-разрядного процессора работает на частотах до 300 МГц, а блоки памяти дают времена доступа 3 нс.

Сопоставление и возможности двух типов SOPC

В чем состоят преимущества и недостатки СБИС двух указанных типов? В однородных SOPC уровень интеграции уже позволяет сконфигурировать на кристалле области одного или нескольких процессоров, памяти и многочисленных периферийных схем. Современные средства САПР с их приспособленностью к взаимозаменяемым и стандартным решениям позволяют объединять на одном кристалле виртуальные компоненты разных разработчиков. Правда, при использовании soft-ядер возникает проблема их приобретения, поскольку стоят они достаточно дорого. В однородных SOPC *не достигаются предельные быстродействия ядер.*

Hard-ядра реализуют блоки, полученные методами проектирования заказных схем. Такие блоки в сравнении с их soft-аналогами занимают на кристалле значительно меньшую площадь (в несколько раз), поскольку они не содержат средств конфигурирования и оптимизированы для выполнения заданной конкретной функции. Если принять площадь, занимаемую цифровым устройством, реализованным по методу "на стандартных ячейках", за единицу, то при реализации того же устройства в технологии LPGA (на БМК с лазерным программированием) площадь составит в среднем 3,3 единицы, для технологии MPGA — 1,6 единиц, а для СБИС ПЛ — 1020 единиц. По этим же причинам существенно (на 20—50%) возрастает быстродействие hard-ядер в сравнении с soft-ядрами.

В то же время предопределенность функций hard-ядер снижает универсальность микросхемы (как говорят, уменьшает ее функциональную гибкость) и может сузить круг ее потребителей, что с точки зрения экономики является негативным фактором. Hard-ядра фиксированы на площади кристалла, что может затруднять решение задач размещения и трассировки для конфигурируемых областей микросхемы, препятствуя тем самым реализации максимальных показателей логической емкости и быстродействия для программируемых ресурсов схемы.

Во избежание больших потерь универсальности блочных SOPC для их hard-ядер отбираются только такие функциональные блоки, которые занимают значительную долю рынка.

Заметим, что современные SOPC с hard-ядрами имеют архитектурную преемственность с прошлыми разработками. Например, уже несколько лет в состав микропроцессорных систем вводят FPGA, создавая тем самым как бы "островок программируемости" среди блоков жесткой структуры. С другой стороны, в составе универсальных СБИС ПЛ тоже несколько лет назад стали появляться несложные аппаратные ядра, т. е. участки фиксированной структуры в окружении массива синтезируемой логики. Так что говорить о том, что во что встроено порою можно по-разному. *Чем выше процент синтезируемой части микросхемы, тем больший контроль над реализацией получает разработчик проекта, но тем больше блоков при этом теряют оптимальность своих параметров.*

По поводу перспектив применения двух разновидностей SOPC высказываются разные мнения. Например, имеется мнение, согласно которому структуры однородного типа с их высокой степенью регулярности легче переводятся на новый технологический уровень, поэтому освоение новых технологий для блочных структур должно всегда несколько запаздывать, а замедленный выход продукции на рынок наносит изготовителю экономический ущерб. С другой стороны, отмечается, что аппаратные ядра не только гарантируют улучшенные параметры блоков, но и упрощают деловую часть разработки проекта, упрощая для проектировщика взаимоотношения с третьими лицами (поставщиками IP). При этом вся техническая поддержка сосредотачивается в руках одного партнера — поставщика микросхем, а все затраты концентрируются в одном показателе (цене микросхемы). Таким образом, введение специализированных аппаратных ядер в схемы SOPC — процесс противоречивый по результатам. Он сокращает площадь кристалла и ведет к достижению максимального быстродействия, но и таит в себе возможность нежелательных последствий для изготовителя, т. к. может сузить рынок сбыта микросхем, а это ведет к росту цен и потере в какой-то мере конкурентоспособности продукции.

Что же будет преобладать? Для успеха разработок блочных SOPC ключевым вопросом был такой — какие именно аппаратные ядра выбираются для реа-

лизации? Самый очевидный выбор — блоки ОЗУ. Эти блоки в той или иной мере нужны почти для всех систем, причем некоторые из них требуют очень больших объемов памяти. Выяснились и условия эффективного использования ядер памяти — не слишком крупные блоки, возможность изменять организацию памяти, возможность иметь асинхронный и синхронный режимы работы, организовывать буферы FIFO и двухпортовую память. *Области ОЗУ — первые и, безусловно, главные специализированные аппаратные ядра*, которые вводятся в СБИС ПЛ уже немало лет. Далее можно назвать аппаратные умножители, используемые в некоторых микросхемах, а также схемы интерфейса JTAG. Ядра интерфейса JTAG успешно внедрились во многие СБИС ПЛ, поскольку они выполняют важные функции, нужные очень многим, и занимают небольшую площадь на кристалле. Свое место среди аппаратных ядер заняли и контроллеры шины PCI, также необходимые в очень многих приложениях и требующие максимального быстродействия. И, наконец, в составе аппаратных ядер появились такие сложные устройства, как микропроцессоры и микроконтроллеры, что ознаменовало собою "полную комплектацию" блочных SOPC.

Ввиду больших возможностей и специфических особенностей обоих типов SOPC при решении вопроса о выборе того или иного типа приходится учитывать целый комплекс показателей. Наиболее бесспорной областью использования блочных SOPC являются системы высшего быстродействия, поскольку, как бы ни выглядели рекламные преувеличения, процессоры с рабочими частотами в несколько сотен мегагерц, осуществимые для hard-ядер, находятся за пределами возможностей soft-ядер. В других, менее очевидных ситуациях, оценивают комплекс характеристик SOPC. В настоящее время *развиваются обе разновидности SOPC* и успехи видны в обоих направлениях.

Задача создания высококачественных законченных систем на одном кристалле решается более чем десятком крупных фирм в обстановке активной состязательности. Состязательность выражается как в конкурентном развитии двух направлений, отраженных в принятой нами классификации SOPC, так и в разработке многих разновидностей СБИС в рамках каждого из этих направлений.

Процессорные ядра SOPC

Технологические достижения привели к появлению в 1990-х гг. мегавентильных СБИС ПЛ. Это естественным образом создало SOPC с однородными структурами, пионерами в их разработке стали фирмы Altera и Xilinx. Позднее были выпущены SOPC с hard-ядрами процессоров (фирмы Triscend, Atmel, Altera, Xilinx, Cypress Semiconductor и др.).

Soft-ядра процессоров. Для своих однородных SOPC фирма Altera разработала *процессорное soft-ядро Nios*. Nios — RISC-процессор с изменяемой архи-

тектурой, конфигурируемым файлом регистров, 16-разрядными командами и шиной данных на 16 или 32 разряда по выбору проектировщика. Производительность процессора может достигать 50 MIPS (Million Instructions Per Second). Для ядра Nios применима разработанная ранее довольно обширная периферия: таймеры-счетчики, UART, широтно-импульсные модуляторы, контроллеры дисковой памяти, динамических ОЗУ и др. Применимы и использовавшиеся ранее в микросхемах фирмы библиотечные параметризируемые модули LPM.

Фирма Xilinx для своих однородных SOPC разработала *soft-ядра процессоров Microblase и Picoblase.* Microblaze — 32-разрядный процессор с Гарвардской архитектурой, работающий на тактовой частоте 125 МГц. Можно размещать на кристаллах и soft-ядра известных процессоров Power PC фирмы IBM. Процессоры сочетают высокое быстродействие с энергетической экономичностью. Для совместной работы с процессорами созданы soft-ядра периферийных устройств (арбитров, UART, контроллеров прерываний и др.). В последнем варианте семейств (Virtex II Pro) процессоры типа Power PC выполнены в виде hard-ядер.

Hard-ядра процессоров. В микросхемах блочных SOPC получили преимущественное применение восьми- и 32-разрядные процессорные ядра. В первых SOPC с аппаратными процессорными ядрами были использованы хорошо зарекомендовавшие себя восьмиразрядные микросхемы семейства 8051 фирмы Intel и AVR фирмы Atmel. Первый вариант представляет направление CISC-, а второй — RISC-процессоров. Подробнее о них говорится далее, т. к. эти процессорные ядра входят в состав рассматриваемых ниже SOPC.

Позднее проектировщики получили в свое распоряжение и варианты 32-разрядных ядер — приспособленные к базовому технологическому процессу производства FPGA схемы RISC-процессоров ARM, MIPS и PowerPC. Примечательно, что эти ядра могут быть так компактны, что занимают порою 1,5—2,0 мм^2 площади кристалла (без учета кэш-памяти). Рабочие частоты перечисленных ядер составляли 200 МГц или более, ядра имели в своей структуре пятиступенчатые конвейеры, 2—3 исполнительных устройства, были ориентированы на малое потребление мощности. Большинство команд выполнялось процессорами за один такт. Процессоры ориентированы на малое потребление мощности. Стандартность архитектур таких процессоров позволяет пользоваться при их применении имеющимися обширными инструментами и средствами проектирования, что снижает трудоемкость разработок и уменьшает время выхода продукции на рынок.

Для повышения быстродействия ядра проектировались с учетом кэширования памяти (кэш L1), но некоторые при пониженном быстродействии могли работать и без кэширования.

Для эффективного взаимодействия процессорных ядер с главной памятью, периферией и устройствами, реализованными в FPGA, нужны *шинные системы высокой производительности.* Этому вопросу при разработке ядер было

уделено большое внимание. Была принята ориентация в первую очередь на две шинные системы: AMBA от фирмы ARM и CoreConnect от фирмы IBM.

В шинной системе AMBA определены три шины: две системные шины (AHB, AMBA High-Speed Bus и ASB, AMBA System Bus) и периферийная шина APB (AMBA Peripheral Bus). Системные шины являются высокоскоростными, не мультиплексируются (имеют отдельные линии для адресов и данных). Более поздняя шина AHB имеет более изощренную архитектуру и наибольшее быстродействие, поддерживает пакетные передачи.

Шинная система CoreConnect состоит из локальной процессорной шины PLB (Processor Local Bus) и периферийной шины OPB (On-Chip Peripheral Bus). Шины не мультиплексируются, поддерживают одновременность чтения и записи (имеют раздельные шины данных для этих режимов), настраиваются на версии разных разрядностей (от 16 до 128). Частота тактирования шины PLB для разных разрядностей составляет 66, 133 и 183 МГц.

§ 10.2. СБИС ПЛ типа "система на кристалле" с синтезируемыми ядрами (с однородной структурой)

Однородные SOPC появились несколько лет назад в составе продукции фирм Altera (семейство APEX 20K), Xilinx (семейство Virtex), Actel (семейство proASIC), Lucent Technologies (семейство ORCA 4) и др. Появившиеся семейства как правило подвергаются периодическим модификациям с существенным улучшением параметров по мере освоения новых технологических процессов, так что возникают целые линии развития SOPC в рамках того или иного семейства. Например, по линии Virtex были с интервалом в год-два выпущены микросхемы Virtex E/EM, Virtex II, Virtex II Pro, а по линии APEX микросхемы APEX 20K/KE/KC, APEX II. Такова же ситуация и с разработками других фирм.

Примечание

Для СБИС ПЛ типа *generic* граница, за которой начинается классификационная область "системы на кристалле", не имеет четких признаков и *является условной*. В этой книге будет принято относить к классу SOPC микросхемы, сложность которых близка к миллиону эквивалентных вентилей или превышает этот уровень.

Семейства СБИС типа APEX 20K/KE/KC, APEX II

Эти микросхемы с триггерной памятью конфигурации обладают программируемостью всех основных областей кристалла. Вариант 20K имеет напряжение питания ядра 2,5 В, а вариант 20KE — 1,8 В. Вариант 20KC с напряже-

нием питания 1,8 В отличается от предшественников главным образом переходом от алюминиевых межсоединений к медным (буква С в обозначении происходит от слова Copper, т. е. медь), улучшившим параметры микросхем, в частности, их быстродействие. Семейство APEX II выполнено на основе технологии с топологическими нормами 0,15 мкм при напряжении питания ядра 1,5 В и многослойной системе медных межсоединений. Уровень интеграции — до 3 млн типичных эквивалентных вентилей, встроенная память в диапазоне приблизительно от 0,5 до 1,2 млн бит. Большое число каналов связи, в том числе дифференциальных, обладающих высокой пропускной способностью (1,6 Гбит/с и 624 Мбит/с), позволяет получать на кристалле быстродействующую систему из множества блоков, либо включать кристалл в более сложную систему в качестве быстродействующего блока. Основные параметры семейства приведены в *Приложении 3.*

Микросхемы построены по архитектуре, названной Multicore. В них комбинируются табличные методы реализации функций и реализации их в дизъюнктивных нормальных формах, т. е. сочетаются характерные признаки FPGA и CPLD. Имеется встроенная память и гибкая система интерфейсов. Фрагмент общего плана микросхем семейств APEX показан на рис. 10.1.

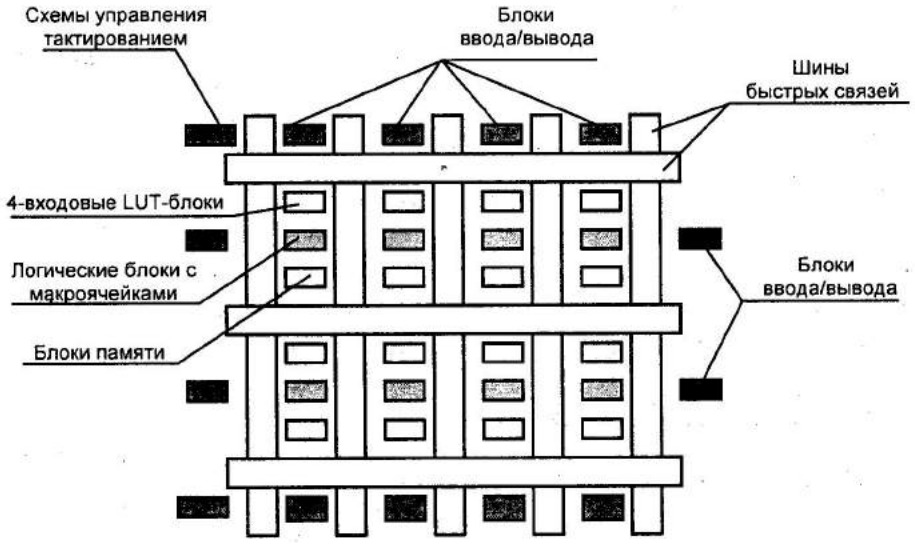


Рис. 10.1. Фрагмент общего плана микросхем семейства APEX.

В каждой "клетке", образованной шинами быстрых связей, размещены функциональные ресурсы трех типов:

- логические блоки с преобразователями типа LUT, подобные описанным *в § 9.4* логическим блокам семейства ACEX 1K;

- логические блоки типа SOP (Sum-of-Products) с макроячейками, заимствованные от выпускавшихся ранее CPLD типа MAX 7000 и принципиально подобные описанным ранее *в § 9.2*;
- встроенные блоки памяти, близкие к блокам EAB микросхем семейства ACEX 1K, но обладающие расширенными возможностями и называемые ESB (Embedded System Blocks).

Табличные LUT-блоки хорошо приспособлены к реализации сложных алгоритмов обработки сигналов и к построению узлов и устройств, содержащих большие количества триггеров. Блоки с макроячейками, реализующие ДНФ, удобны для получения функций быстродействующей управляющей логики (в частности, адресных декодеров) и управляющих автоматов (State machins). Совмещение обоих типов блоков в одном кристалле должно обеспечивать эффективность схемных решений, характерных для проектов разных типов. Этому же способствует наличие на кристалле встроенных блоков памяти высокого быстродействия, организуемых в виде RAM, двухпортовой RAM, FIFO с широким диапазоном программируемых длины и ширины, а также блоков ассоциативной памяти (у некоторых микросхем). Блоки ввода/вывода поддерживают интерфейсы PCI, GTL+, SSTL-3, LVDS и др.

Сложность кристаллов семейства APEX обусловила увеличение в них числа иерархических структурных уровней. Наименьшей структурной единицей логического типа является элемент, подобный логическому элементу семейства ACEX 1K (см. рис. 9.21), допускающий в зависимости от решаемых задач несколько стандартных конфигураций (работу в нормальном, арифметическом или счетном режиме) и создание цепей переноса и каскадирования. В нормальном режиме воспроизводится большинство логических функций и при каскадировании логических элементов дополнительные функции. В этом режиме LUT-блоки работают как функциональные преобразователи с четырьмя входами. В арифметическом режиме LUT-блок конфигурируется как два трехвходовых функциональных преобразователя, один из которых служит для выработки функций переносов, а другой для функций трех аргументов, реализуемых в разрядных схемах устройства. В счетном режиме к схеме, соответствующей арифметическому режиму, добавляются элементы, обеспечивающие функции разрешения счета, реверса, сброса и загрузки счетчика.

Десять логических элементов объединяются в логический блок ЛБ (LAB, Logic Array Block), имеющий свой уровень межсоединений — локальную матрицу соединений ЛМС (LI, Local Interconnect). Шестнадцать логических блоков и блок ESB объединяются в *мегаблок* (MegaLAB). Мегаблоки, представляющие собою самостоятельный функциональный уровень, также имеют свой уровень межсоединений, что не только увеличивает обшие ресурсы трассировки кристалла, но и позволяет получать в пределах мегаблока некоторые функционально законченные части системы. Функциональная автономность мегаблоков, когда она возможна, упрощает модификации системы

и закладывает возможности локальной оптимизации схемных решений. Мегаблоки коммутируются между собой по системе глобальной матрицы соединений ГМС (Fast Track Interconnect), линии которой непрерывны и проходят по всей длине (ширине) кристалла. К концам линий ГМС подключаются элементы ввода/вывода ЭВВ (IOE, Input/Output Elements).

Схемы ввода/вывода отличаются высоким быстродействием и поддерживают более 10 стандартов входных и выходных сигналов. Контакты ввода/вывода разделены на несколько банков, имеющих свои напряжения питания и способных независимо от других банков поддерживать те или иные стандарты ввода/вывода. Элементы ввода/вывода обеспечивают совместимость с шиной PCI (64 разряда, 66 МГц).

Большое внимание уделено схемам тактирования кристалла, реализована схема передачи синхросигналов с малыми фазовыми сдвигами, блоки микросхем могут использовать до восьми глобальных синхросигналов. Схемы управления параметрами синхросигналов выполняют функции минимизации их задержек и фазовых рассогласований (ClockLock), умножения частоты в некоторых блоках схемы, что позволяет разводить по кристаллу сигналы тактирования меньшей частоты (функция ClockBoost), функции ClockShift (введения специальных программируемых фазовых сдвигов в линии передачи сигналов).

Микросхемы семейства имеют программируемое управление по координатам скорость/мощность. Когда максимальное быстродействие не требуется, можно снижать мощность потребления с помощью соответствующей опции турбо-бита. Неиспользуемые блоки ставятся в режим глубокого снижения мощности.

Микросхемы семейства имеют настолько высокий уровень интеграции, что процессор Nios занимает небольшую долю их логической емкости. Например, для микросхемы EP20K200E ядро процессора Nios занимает 12% логических ресурсов кристалла. При стоимости кристалла около $ 80 (в единичных поставках) на ядро процессора приходится приблизительно $ 10, а при массовых поставках стоимость снижается приблизительно в два раза. Для микросхемы EP20K1500E с 1,5 миллионами вентилей доля расхода ресурсов на процессор Nios снижается до 1,5%. Изменяемая архитектура ядра Nios, как и других разработанных soft-ядер, придает SOPC высокую степень гибкости.

Семейство Stratix

Семейство Stratix (2002 год) — микросхемы класса SOPC высшей сложности и быстродействия (топологическая норма 0,13 мкм, все слои межсоединений медные, напряжение питания ядра 1,5 В). Старший представитель семейства — микросхема с 114 140 логическими элементами (сложность логического элемента оценивается 26 эквивалентными вентилями, так что в целом

схема насчитывает около 3 млн эквивалентных вентилей) и приблизительно 10 Мбит RAM при частотах работы отдельных устройств до 420 МГц.

Основные блоки микросхем показаны на рис. 10.2. Функциональные преобразования выполняются расположенными по строкам и столбцам логическими блоками LAB. Блоки коммутируются межсоединениями различной длины и различного быстродействия, которые связывают друг с другом блоки разного типа, входящие в состав микросхемы (логические блоки, блоки памяти, блоки цифровой обработки сигналов и др.). В каждом логическом блоке имеется 10 логических элементов LE, подобных рассмотренным ранее элементам микросхем семейства ACEX 1K и отличающихся от них несколько расширенными возможностями.

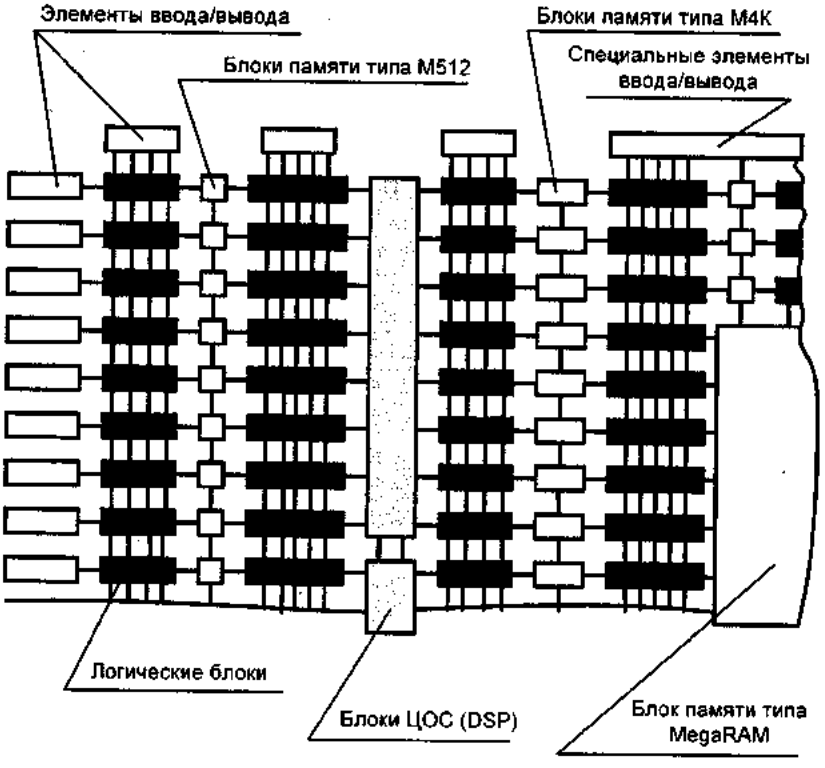


Рис. 10.2. Основные блоки микросхем семейства Stratix

Память микросхем семейства Stratix организована по варианту, названному TriMatrix. В этом варианте используются блоки памяти трех разновидностей. Блоки типа M512 RAM — простые однопортовые или двухпортовые ЗУ с разрядностью до 8 и емкостью 512 бит (с учетом контроля по четности ем-

кость равна 576 битам). На них могут быть реализованы также сдвигающие регистры, буферы FIFO. Блоки типа M4K RAM — двунаправленные двухпортовые, которые могут быть организованы и как простые двухпортовые или однопортовые с разрядностями до 36 и емкостью 4 Кбит (с учетом контроля по четности емкость равна 4608 бит). Оба эти типа блоков памяти могут работать на частотах до 312 МГц и расположены по столбцам, идущим по всему кристаллу между столбцами логических блоков. Блоки типа MegaRAM — двунаправленная двухпортовая память с емкостью 512 Кбит (с учетом контроля по четности емкость блока равна 589824 бита), которая может быть организована и как простая двухпортовая или как однопортовая память с разрядностью до 144, работающая на частотах до 300 МГц. Эти блоки размещаются поодиночке или попарно в пределах матрицы логических блоков.

Блоки цифровой обработки сигналов DSP, сгруппированные на кристалле в два столбца, могут выполнять операции умножения с возможностью накопления (суммирования) результатов или их вычитания при разрядностях операндов 9, 18 или 36. Вырабатывается произведение полной точности.

> При размерности 9×9 можно реализовать в блоке до восьми умножителей, при размерности 18×18 — четыре и при размерности 36×36 — один. Суммарно в микросхеме может быть создано до 224 умножителей размерности 9×9. Блоки ЦОС дают по 2×10^9 операций умножения-накопления в секунду, и если использовать все 28 блоков старшего представителя семейства, то будет достигнута вычислительная мощность в 56×10^9 операций MAC в секунду.

Блоки ЦОС содержат также 18-разрядные сдвигающие регистры. Имеющийся в блоках набор устройств позволяет строить типичные для ЦОС структуры, в частности, цифровые фильтры с конечной или бесконечной импульсной характеристикой. Максимальная частота работы этих блоков составляет 250 МГц.

Каждый вывод микросхемы снабжен элементом ввода/вывода IOE. Элементы IOE расположены по периферии кристалла в концах строк и столбцов логических блоков. Они поддерживают многочисленные стандарты ввода/вывода, имеют двунаправленные буферы и по 6 триггеров для фиксации входных и выходных сигналов и сигналов разрешения работы. При использовании специального тактирования эти триггеры позволяют реализовать интерфейс кристалла с быстродействующей внешней памятью, такой как DDR SDRAM, FCRAM, QDR SRAM, ZBT SRAM.

Микросхемы имеют до 116 быстродействующих дифференциальных каналов ввода/вывода, у 80 из которых скорость передачи до 840 Мбит/с. Каналы LVDS содержат блоки SERDES. Поддерживаются и несколько стандартов коммуникационных шин. Введены внутрисхемные элементы согласования

волновых сопротивлений в однополюсных линиях передачи для уменьшения числа внешних компонентов и упрощения проектирования систем.

Большое внимание уделено управлению тактированием, что особенно существенно для столь быстродействующих микросхем. Старшие представители семейства имеют до 12 блоков PLL. Для создания soft-ядер разработано множество IP (мегафункций).

В целом микросхемы семейства Stratix имеют большие функциональные ресурсы высокого быстродействия при их ориентации на решение задач цифровой обработки сигналов. Основные параметры этих микросхем приведены в *Приложении 3.*

Семейства СБИС типов Virtex E/EM, Virtex II, Virtex II Pro

Микросхемы Virtex класса SOPC с *архитектурой FPGA* (фирма Xilinx) — кристаллы с мегавентильными уровнями интеграции и большими емкостями встроенной памяти. В течение нескольких последних лет выпущены семейства Virtex E/EM, Virtex II и Virtex II Pro с напряжениями питания ядра 2,5; 1,8 и 1,5 В соответственно.

Семейство Virtex E/EM. Общий план микросхем семейства Virtex E/EM показан на рис. 10.3.

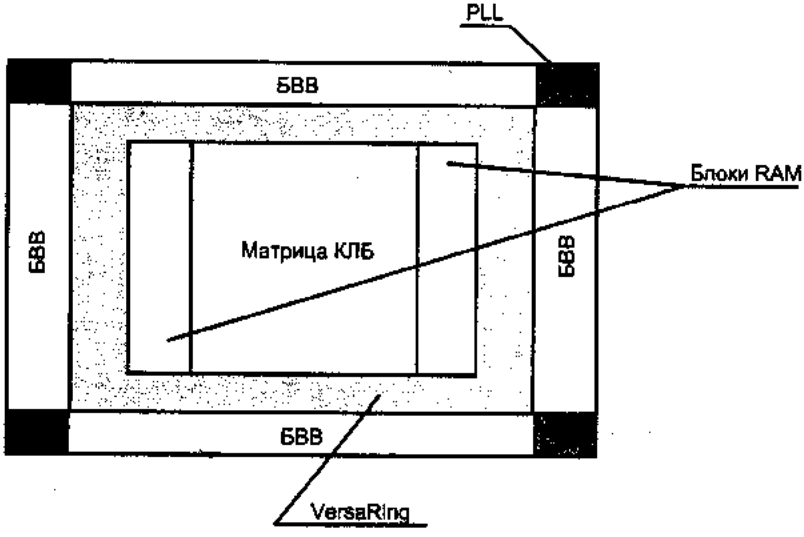


Рис. 10.3. Общий план микросхем семейства Virtex E/EM

Внутренняя область кристалла обозначена как матрица конфигурируемых логических блоков КЛБ. Фактически это матрица Versa-блоков (Versa Blocks), содержащих КЛБ и переключательные блоки ПБ, которые совместно с трассами связей между ними образуют глобальную матрицу соединений (рис. 10.4).

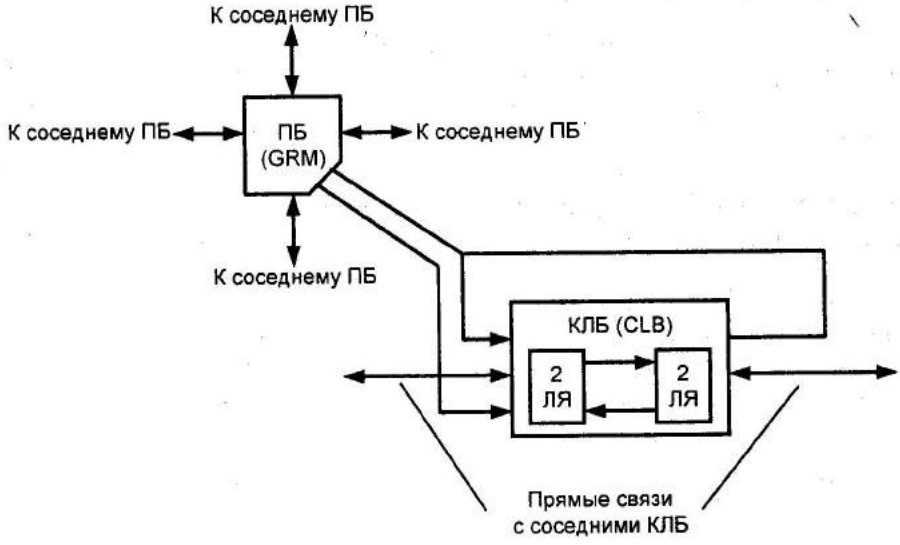


Рис. 10.4. Схема Versa-блока СБИС ПЛ семейства Virtex E/EM

В Versa-блоке имеются локальные ресурсы коммутации для связи КЛБ с глобальной матрицей соединений GRM (General Routing Matrix), линии обратных связей для самих КЛБ и прямые соединения между горизонтально-смежными КЛБ. Каждый КЛБ состоит из двух секций с двумя логическими ячейками (элементами) в секции. Versa-блоки имеют ресурсы, на которых можно реализовать небольшие части проекта, объединяемые далее предоставляемыми средствами коммутации.

Логические ячейки ЛЯ содержат четырехвходовые LUT-блоки с задержкой выработки функций 1,2 нс. Память LUT-блоков может использоваться и как ЗУ на 16 бит с возможностью объединения в более крупные блоки. Можно объединять два элемента LUT для воспроизведения функций пяти переменных. В логических ячейках имеются также высокоскоростные схемы переноса для построения каскадных структур, схемы управления и D-триггеры с общим (глобальным) тактированием и входами сброса/установки. Для упрощения программного обеспечения средств проектирования ЛЯ соединяются парами в секции (Slice).

Переключательный блок (ПБ (GRM)) коммутирует горизонтальные и вертикальные связи. В системе связей есть не только линии одинарной длины, передающие сигналы от одного ПБ к другому (смежному), но и буферированные линии передач на расстояние в 6 ПБ. По всей длине и ширине кристалла проходят длинные линии.

В области памяти содержится набор двухпортовых SRAM по 4 Кбит каждый с несколькими вариантами организации (4K × 1, 2K × 2, 1K × 4, 512 × 8, 256 × 16). Память синхронная, время цикла 10 нс. Предусмотрены конфигурации памяти типа видеобуфера, буферов FIFO.

На периферии кристалла расположены блоки ввода/вывода с буферированием входных и выходных сигналов и управлением третьим состоянием. Программируются такие возможности, как регулировка крутизны фронта, "подтягивание" линии к высокому потенциалу, регулировка задержки и другие аспекты операции буферирования. Возможен выбор того или иного стандарта сигналов интерфейса, включая GTL+, SSTL, LVTTL.

Многообразные ресурсы трассировки дополнены средствами, названными VersaRing. Это интерфейс между логикой внутренней области и блоками ввода/вывода, позволяющий при необходимости перекоммутировать связи внутренней области с внешними выводами, так что при модификации схемы во внутренней области связи с монтажом печатной платы могут не изменяться.

На кристалле имеются схемы PLL, обеспечивающие коррекцию временных соотношений для тактовых импульсов. Эти схемы играют известную роль, уже не раз упоминавшуюся ранее. Блоки ввода/вывода в регистровом режиме имеют задержки по пути "такт—выход" 6 нс без PLL и 3,5 нс с PLL. Максимальный темп передач ввода/вывода 110 МГц без PLL и 160 МГц с PLL.

Представители семейства имеют от 38 до 851 Кбит встроенной памяти, к которой могут добавляться от 24 до 1038 Кбит памяти от схем конфигурации логических блоков типа LUT. Схемы работают на системной частоте 180—200 МГц. Достижимый процент использования вентилей оценивается как 90%. Число пользовательских выводов лежит в диапазоне 180—804. Линии ввода/вывода программируются на ряд стандартов интерфейсных сигналов (GTL+, LVTTL, SSTL3-I и др.).

Файл конфигурации (для кристаллов с 100 тыс. эквивалентных вентилей) имеет объем 2 Мбит и загружается за менее чем 3 мс, что является малым временем, способствующим применению микросхем в системах с реконфигурацией аппаратных средств.

Семейство Virtex II. Фрагмент общего плана микросхем семейства Virtex II показан на рис. 10.5.

Возможности SOPC, создаваемых на микросхемах семейства Virtex II, значительно расширились в сравнении с возможностями для семейств-пред-

шественников. Топологические нормы в 0,12 и 0,15 мкм и восемь слоев медных межсоединений позволили довести уровень интеграции микросхем до 104 882 логических элементов. Достоинством семейства можно считать широкий диапазон сложности у разных его представителей (имеется 11 типов микросхем).

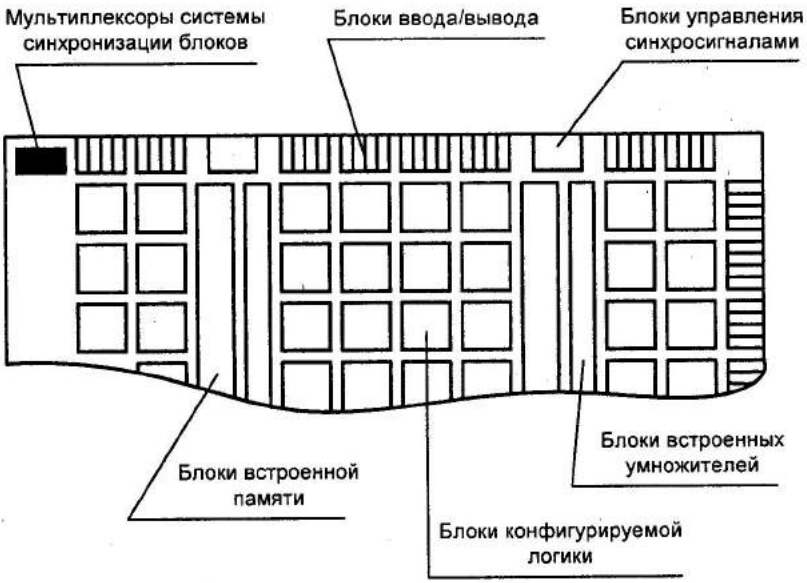


Рис. 10.5. Фрагмент общего плана микросхем Virtex II

По области применения семейство ориентировано на высокопроизводительные системы обработки данных и цифровой обработки сигналов с использованием в областях телекоммуникаций, беспроводной связи, построения сетей и т. п. Для размещения на кристаллах семейства фирма Xilinx разработала soft-ядра процессора Microblaze. Внутренняя тактовая частота микросхем достигает 420 МГц, последовательные каналы ввода/вывода имеют пропускную способность до 840 Мбит/с на контакт. Существует возможность выбора внутрикристальных резисторов-терминаторов с автоматическим согласованием их номиналов с внешними образцами.

Память микросхем иерархична. До 1,5 Мбит памяти распределены в LUT-блоках, до 3 Мбит реализуется в виде встроенных блоков двухпортовой памяти с емкостью до 18 Кбит в каждом. Блоки памяти программируются в вариантах 16K×1, 8K×2, 4K×4, 2K×9, 1K×18, 512×36. Обеспечиваются интер-

фейсы с внешней памятью высокой производительности, такой как DDR SDRAM, FCRAM и др.

Логические элементы имеют цепи переноса и каскадирования, возможно создание внутренних шин с тремя состояниями. Специализированными встроенными арифметическими модулями являются умножители 18×18.

Система межсоединений сегментирована, но отличается новизной (она названа Active Interconnect Technology), обеспечивающей ей улучшенную прогнозируемость задержек и более высокую скорость передачи сигналов.

Для управления параметрами тактовых сигналов имеются до 12 специальных модулей, называемых DCM (Digital Clock Manager), способных, в частности, дискретно сдвигать фазу тактовых импульсов с шагом в 1/256 тактового периода.

Семейство Virtex II Pro. Микросхемы этого семейства имеют еще более высокую сложность и быстродействие. На их кристаллах размещены hard-ядра процессоров, что относит их к блочным SOPC, рассматриваемым в следующем параграфе.

Параметры микросхем семейства Virtex приводятся в *Приложении 3*. Сопоставляя параметры таких микросхем, как Stratix (фирма Altera) и Virtex II, Virtex II Pro (фирмаXilinx), можно видеть как разницу между ними, так и много общих тенденций развития при приблизительно равном уровне совершенства.

SOPC с энергонезависимой памятью конфигурации

Все рассмотренные выше SOPC имели триггерную память конфигурации. Однако среди выпускаемых имеется и мегавентильная (пока единственная) SOPC с *флэш-памятью* для хранения данных конфигурации. Это *микросхема ProASIC Plus* фирмы Actel. Сравнительные достоинства и недостатки ПЛИС с разными типами памяти конфигурации уже обсуждались ранее. Здесь отметим только, что энергонезависимость флэш-памяти позволяет исключить из системы внешнее запоминающее устройство для стартовой загрузки конфигурационной памяти, а это благоприятно для засекречивания проектов. Кроме того, отсутствие начальной загрузки SOPC делает ее готовой к работе сразу после включения питания.

В номенклатуре однородных СБИС ПЛ имеется ряд микросхем, не достигающих мегавентильного уровня сложности, но имеющих средства поддержки построения систем, что приближает их к схемам, рассмотренным в этом параграфе. В этой группе микросхем прежде всего следует выделить *семейство Eclipse* фирмы QuickLogic с программированием перемычек типа antifuse.

§ 10.3. СБИС ПЛ типа "система на кристалле" с аппаратными ядрами (блочные)

Блочные SOPC получили развитие в самые последние годы. Их освоение началось с микросхем, содержащих такие hard-ядра, как интерфейс PCI, умножители и другие специализированные схемы, более простые, чем процессоры. Такие "беспроцессорные" SOPC полезны для многих проектов в различных областях специализации и имеют свои области применения. В то же время именно появление в составе блочных SOPC процессорных hard-ядер придало им универсальный характер и сделало их эффективным средством реализации самых разнообразных систем обработки информации.

Блочные SOPC, не содержащие процессорных ядер

SOPC с относительно простыми ("допроцессорными") ядрами, ориентированные на специализированные приложения, имеют и сохраняют свои области применения. Существование "заполненной" зоны между SOPC с простейшими и SOPC с самыми сложными ядрами закономерно и обуславливается потребностями системотехников. Блочные SOPC со специализированными ядрами выпускаются несколькими фирмами. Приведем несколько примеров.

Продукция фирмы Lucent Technologies ориентирована на рынок интерфейсных схем для систем коммуникаций. Микросхемы этой фирмы отличаются высоким схемотехнологическим уровнем (выполнены по технологии с минимальными размерами 0,13 мкм при семи слоях металлизации).

Фирма QuickLogic разработала семейство блочных SOPC под названием ESP (Embedded Standard Products) с подсемействами QuickRAM, QuickPCI, QuickPC, QuickDSP, QuickSD. Использование программируемых перемычек ViaLink типа antifuse придает микросхемам известные свойства — однократность программирования, но при этом высокую компактность межсоединений, высокое быстродействие, повышенную стойкость к воздействиям температуры и радиации, защищенность от рассекречивания проекта.

Кристаллы QuickRAM (1998 г.) содержат FPGA и встроенные блоки памяти емкостью до 24 Кбит с программированием организации как RAM, ROM, FIFO при способности работать на частотах до 160 МГц.

Микросхемы QuickPCI имеют FPGA и фиксированное ядро PCI. Для ядра PCI возможен выбор вариантов на 32- и 64 разряда с частотами 33, 66 и 75 МГц. Для связи ядра PCI с другими частями схемы введен набор буферов FIFO.

Схемы QuickPC имеют аппаратно реализованный канал Fibre Channel со скоростью передачи данных до 2,5 Гбит/с и 32-разрядным синхронным системным интерфейсом с буферами FIFO.

Кристаллы QuickDSP базируются на специальных арифметических ячейках ECU (Embedded Computational Units), выполняющих однотактно операции умножения 8×8 с задержкой 4,53 нс, 16-разрядное сложение с задержкой 2,54 нс, умножение с накоплением с задержкой 7,07 нс. На основе таких ячеек строятся схемы реализации алгоритмов цифровой обработки сигналов.

В микросхемах QuickSD комбинируются быстродействующие FPGA и блоки SERDES (Serializer-Deserializer), выполняющие преобразования данных из параллельной формы в последовательную, и наоборот. Три представителя семейства имеют в своем составе 6 или 8 каналов SERDES. Каждый порт SERDES может передавать последовательные данные со скоростью до 1 Гбит/с. Две программируемых схемы синхронизации каналов SERDES могут тактировать передачи, если синхросигналы не содержатся в самих данных. Имеются два программируемых блока PLL и от 24 до 36 блоков двухпортовой SRAM, а также 12—18 блоков QMAC умножения-накопления, которые в большой степени ускоряют вычисления при цифровой обработке сигналов. Результирующая скорость передачи составляет 8 Гбит/с, чего достаточно для большого числа применений. Логика FPGA может выполнять операции кодирования и декодирования информации, балансировки дифференциальных сигналов по постоянному току, обрамления пакетов данных, управления памятью и др., в том числе, реализацию некоторых интерфейсов.

Каждый канал SERDES может быть запрограммирован на выработку последовательных кодов для 1, 4, 7, 8, 10 или 20 линий параллельной шины, что позволяет блоку принимать сигналы с многоразрядных внутренних шин и расщеплять их для максимального использования ресурсов кристалла.

В качестве примера применения микросхем QuickSD можно указать интерфейс с дисплеем высокого разрешения. Поток данных для такого дисплея обычно передается через многоразрядный кабель. При наличии скоростного последовательного канала кабель можно исключить и заменить более дешевым последовательным интерфейсом. При этом возможности микросхем QuickSD способны удовлетворить требования систем с самыми современными дисплеями.

Внутренние передачи "регистр-регистр" производятся на частотах до 600 МГц, частота передач "кристалл-кристалл" свыше 225 МГц.

Блочные SOPC с процессорными ядрами

В микросхемах блочных SOPC получили преимущественное применение 8- и 32-разрядные процессорные ядра. В первых SOPC с аппаратными процессорными ядрами были использованы хорошо зарекомендовавшие себя в

многолетней практике восьмиразрядные CISC-процессоры семейства 8051 фирмы Intel и RISC-процессоры AVR фирмы Atmel. Позднее проектировщики получили в свое распоряжение и 32-разрядные ядра — RISC-процессоры ARM, MIPS и PowerPC, также подтвердившие свою эффективность практическим использованием в ряде разработок. Подробнее об аппаратных ядрах процессоров сказано ранее в *разд. "Процессорные ядра SOPC" § 10.1.*

Для эффективного взаимодействия процессорных ядер с другими модулями системы нужны шины высокой производительности. Этому вопросу было уделено большое внимание. Была принята ориентация на две шинные системы: AMBA от фирмы ARM и CoreConnect от фирмы IBM. Некоторые сведения об этих шинах также приведены в указанном ранее разделе.

Пионеры разработки блочных SOPC с процессорными hard-ядрами — фирмы Atmel и Triscend. Фирма Atmel первой анонсировала свою SOPC типа FPSLIC (эта разработка была признана лучшим электронным проектом США в 1999 г.), но фирма Triscend опередила ее в промышленном выпуске своего семейства E5. Позднее блочные SOPC с процессорными hard-ядрами были выпущены и другими поставщиками, например, фирмами Altera (семейство Excalibur), Xilinx (семейство Virtex-II Pro).

Семейство FPSLIC фирмы Atmel

До выпуска микросхем FPSLIC (Field Programmable System-Level Integration Chip) фирмой Atmel выпускались микроконтроллеры AVR и FPGA семейства AT40K, которые и объединены в одном кристалле FPSLIC. Кроме них кристалл содержит статическую память SRAM. Процессор передает для FPGA команды и исходные данные и получает от нее выработанные результаты.

Основными частями микросхемы являются:

- процессорное ядро и его периферия;
- память программ и память данных;
- FPGA.

Гарвардская архитектура процессора с разделением памяти команд и данных способствует повышению его быстродействия вследствие совмещения во времени процессов выборки и выполнения команд. Архитектура кристалла FPSLIC показана на рис. 10.6.

Процессорное ядро AVR и его периферия (микроконтроллер), а также блоки памяти рассмотрены в *главе 7.* В блоке FPGA используются *стандартные структуры микросхем AT40K* сложностью от 10 до 40K используемых эквивалентных вентилей с емкостью встроенной памяти от 2 до 18 Кбит, работающие на системной частоте 100 МГц. В основных чертах эти структуры типичны для FPGA, хотя имеют и некоторые оригинальные особенности (рис. 10.7).

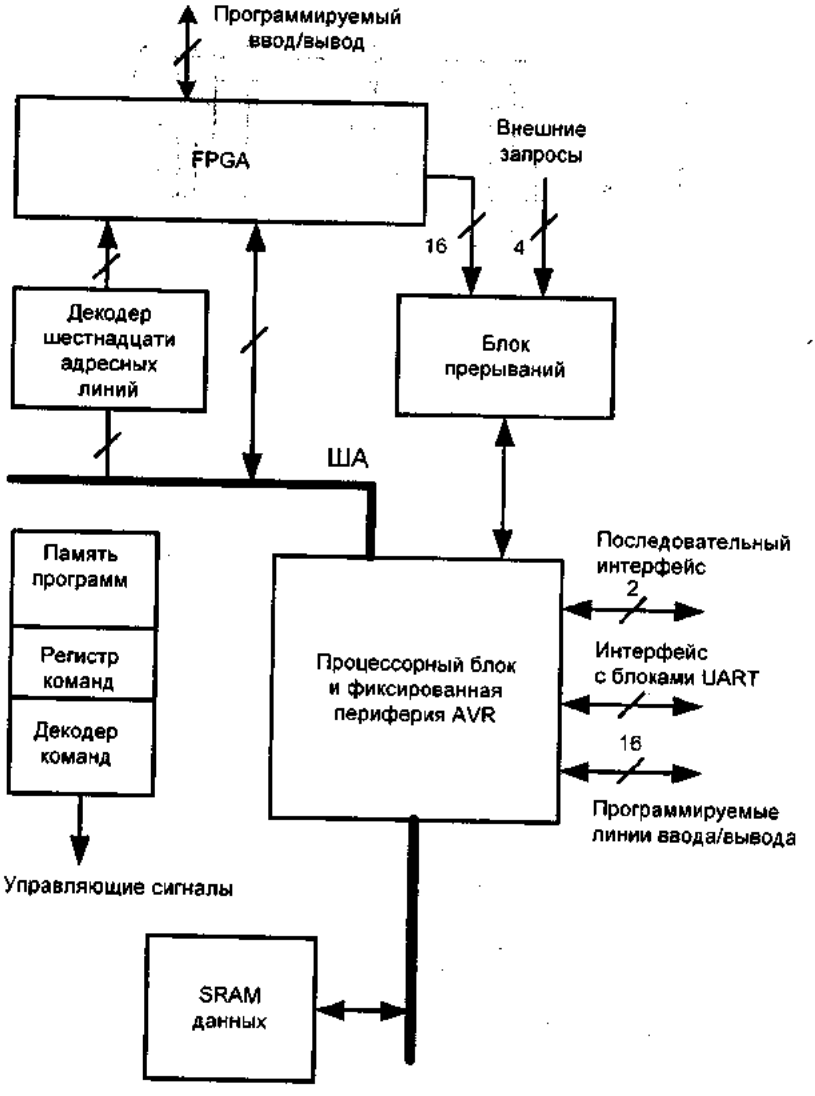


Рис. 10.6. Структура микросхемы FPSLIC фирмы Atmel

Основа FPGA (Field Programmable Gate Arrays) — симметричная матрица идентичных *логических ячеек*, в каждую из которых входят два табличных трехвходовых функциональных преобразователя (LUT-блоков), несколько программируемых мультиплексоров, триггер типа D, буфер с тремя состояниями. LUT-блоки воспроизводят функции трех аргументов (одних и тех же для обоих блоков) и могут быть объединены для выработки функций четырех переменных.

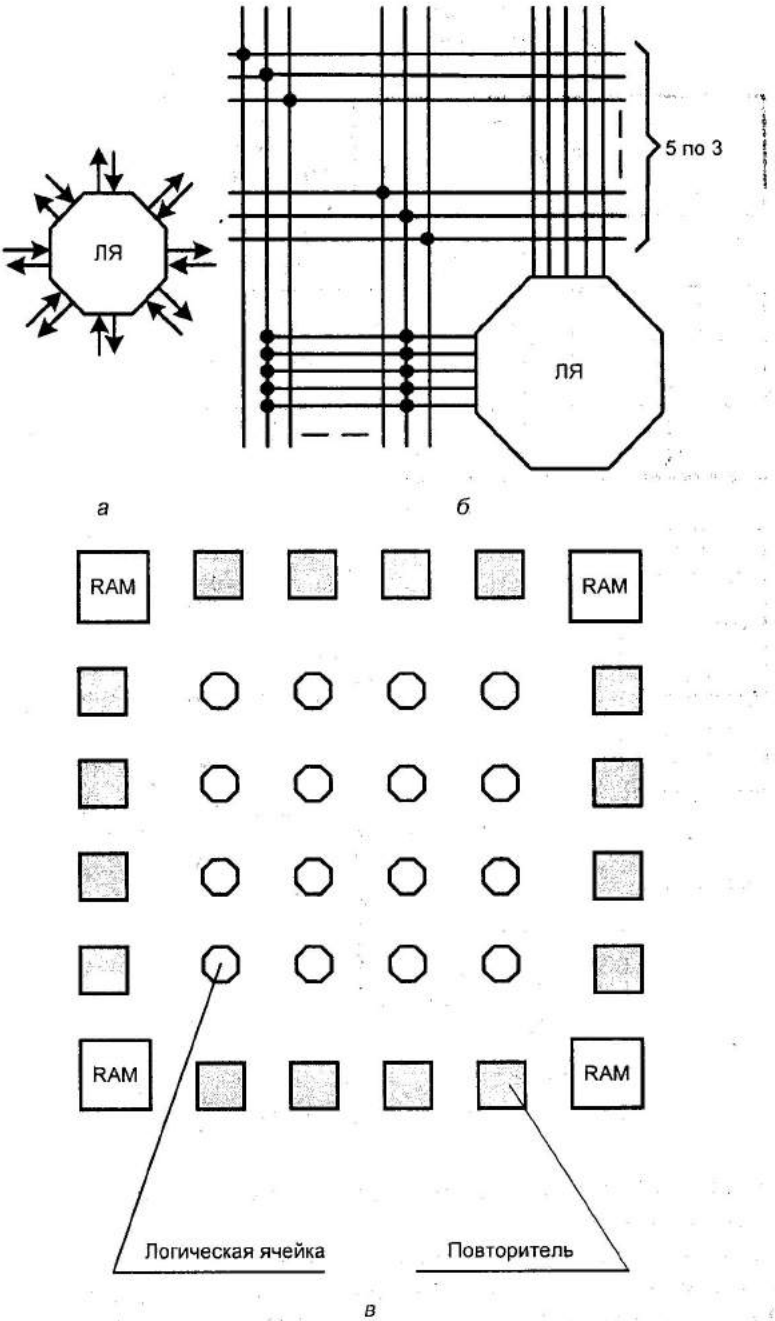


Рис. 10.7. Логическая ячейка (*а*), шинные связи (*б*) и фрагмент матрицы FPGA FPSLIC

Для логических ячеек предусмотрены следующие стандартные режимы (конфигурации):

- одноразрядного сумматора (арифметический режим);
- режим DSP/Multiplier;
- разряда счетчика;
- мультиплексоров с буферами на три состояния (режим Tristate/MUX).

Эти режимы ориентированы на проекты с интенсивными вычислениями, такие как реализация цифровых фильтров, быстрых преобразований Фурье, конвольверов, криптографических алгоритмов и многих других мультимедийных задач. На основе FPGA реализуются и общеупотребительные интерфейсные функции (UART, PCI и др.).

Примечание

Конвольвер — вычислительное устройство для определения свертки, от англ. *convolver*.

Топологически логическая ячейка — восьмиугольник и с восемью ближайшими соседями она имеет прямые связи (Direct Connects), проходящие по матрице ячеек в ортогональных и диагональных направлениях (такое решение оригинально, поскольку логические ячейки FPGA обычно имеют прямоугольную форму и, соответственно, прямые связи только с четырьмя соседними ячейками). Благодаря многочисленным прямым связям, в частности, строятся ультрабыстродействующие матричные множительные устройства.

Массив FPGA имеет 6 внешних и 2 внутренние *линии тактирования*. Для внутренних линий тактирования источником сигналов служит ядро AVR, причем одна из этих линий принадлежит системной линии тактирования микроконтроллера, а вторая может быть запрограммирована на соединение с одним из нескольких источников тактовых сигналов, генерируемых внутри AVR (таймеров и др.).

Ядро AT40K имеет свою *статическую память FreeRam* с временем доступа к данным 10 нс. Благодаря наличию FreeRAM функции памяти в создаваемых на основе FPGA устройствах реализуются без затрат ее логических ресурсов. Возможны различные варианты организации памяти FreeRAM — синхронный или асинхронный, одно- или двухпортовый для RAM, FIFO и др. Варианты организации создаются инструментальными средствами макрогенерации функций.

Система межсоединений элементов FPGA иерархична и включает в себя локальные шины и экспресс-шины. С восемью ближайшими соседями, как уже отмечалось, логические ячейки имеют прямые связи. Сегменты локальных шин покрывают расстояния в 4 ячейки, экспресс-шин — в 8. Шины соединяются через повторители (Repeaters), подключаемые к двум соседним

сегментам. Повторители регенерируют сигналы и выполняют также некоторые функции их коммутации. В системе межсоединений используются программируемые пасс-вентили (Pass gates), с помощью которых формируются шины с тремя состояниями.

Взаимодействие блоков SOPC. Взаимодействие блоков SOPC и их интерфейс с внешней средой специфичны для данного класса схем и их особенности заслуживают внимания.

Интерфейс FPGA с AVR (рис. 10.8) предусматривает для FPGA 16 входных линий декодированного адреса и 16 выходных линий для запросов прерываний с разными приоритетами. Декодированные адреса формируются дешифрацией четырехразрядных адресов, размещенных в специальных полях адресного пространства процессора. Таким образом, в FPGA могут быть созданы 16 адресуемых объектов, которым придается возможность иметь собственные запросы прерывания для передачи в процессор. Данные передаются по восьмиразрядной шине данных. Кроме того, процессор посылает в FPGA стробы чтения и записи, управляющие двунаправленной шиной данных. Таким образом, процессор взаимодействует с реализованными в FPGA устройствами как с внешними устройствами микропроцессорной системы.

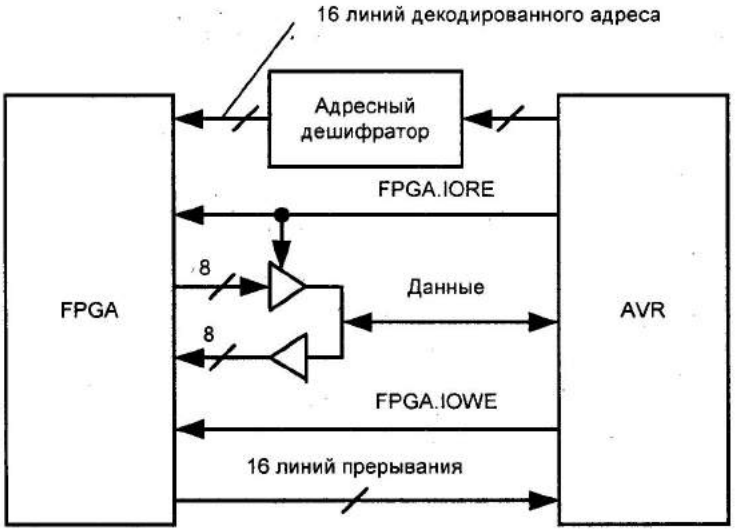


Рис. 10.8. Схема интерфейса между блоками FPGA и AVR в микросхеме FPSLIC

Другой аспект взаимодействия процессора с FPGA (рис. 10.9) реализуется их общим доступом к двухпортовой памяти SRAM с временем доступа 15 нс. Общая емкость двухпортовой памяти SRAM составляет 36 Кбайт.

20 Кбайт (10K×16) всегда используются как память программ, 4 Кбайт (4K×8) всегда используются как память данных. Остальные 12 Кбайт могут по желанию системотехника как целиком, так и частями, передаваться как памяти программ, так и памяти данных. В рабочем режиме устройства, реализованные в FPGA, имеют непосредственный доступ к памяти данных. Со стороны FPGA вырабатываются только сигналы разрешения записи WE, а чтение всегда разрешено. Операции обмена с памятью могут производиться со стороны FPGA и процессора одновременно, но при появлении одного и того же адреса необходим арбитраж. Обычно он заключается в ограничении доступа к памяти со стороны FPGA при возникновении конфликта.

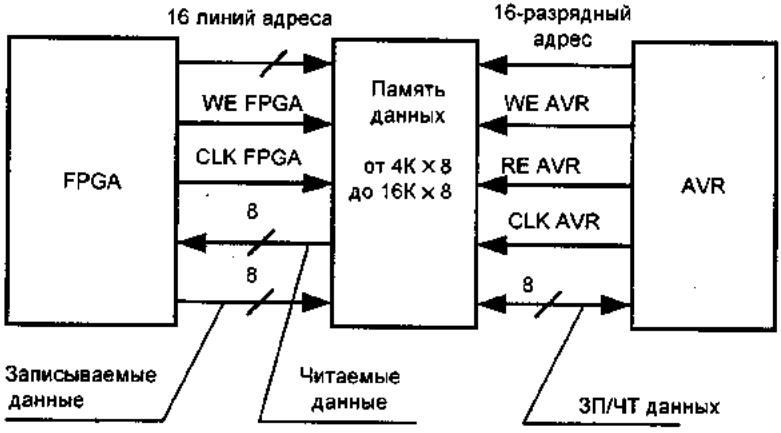


Рис. 10.9. Схема доступа блоков FPGA и AVR к общей двухпортовой памяти микросхемы FPSLIC

Концепция кэш-логики. Важная особенность микросхем FPSLIC — способность к реализации *концепции кэш-логики.* Кэш-логика позволяет конфигурировать систему "на лету", причем полностью или частично. Реконфигурация системы "на лету" производится без потери имевшихся данных и нарушения работы той части системы, которая остается неизменяемой. Данные, которые были выработаны к моменту перестройки той или иной части системы, сохраняются в недорогих устройствах памяти. Та часть аппаратуры, в которой в текущий момент идет обработка информации, представлена работающей схемой соответствующей конфигурации, созданной в блоке FPGA. Так можно выполнять требуемые преобразования при значительной экономии аппаратных средств, необходимых для решения задачи.

Перспективы применения кэш-логики расцениваются как достаточно обширные. Ее использование, снижая сложность аппаратуры, повышает тем

самым ее надежность, снижает потребляемую мощность и стоимость. Это важно, прежде всего, для портативной аппаратуры с батарейным питанием.

Такие SOPC, как FPSLIC обладают большими функциональными возможностями и эффект от их применения достигается там, где эти возможности используются достаточно полно. Для более простых задач можно найти и более простые "системы на кристалле".

Блочные SOPC фирмы Triscend

Исключительно "системами на кристалле" занимается корпорация Triscend ("Трайсенд"), выпустившая в 1999 г. первую промышленную микросхему блочной SOPC с процессорным ядром (по терминологии фирмы это CSOC, Configurable System On Crystal). Первым семейством CSOC этой фирмы были микросхемы E5, затем было выпущено семейство A7.

В микросхемах семейства E5 на одном кристалле объединены восьмиразрядное ядро микроконтроллера "турбо 8032", совместимое со ставшей в последнее время фактическим стандартом архитектурой 8051 фирмы Intel, FPGA, системная шина, память типа SRAM и несколько вспомогательных периферийных устройств. Для следующего семейства A7 фирмой Triscend использовано 32-разрядное процессорное ядро фирмы ARM.

В микросхемы семейства E5 входят следующие основные блоки (рис.10.10):

- микроконтроллерное ядро с длительностью командного цикла 4 такта и тактовыми частотами до 40 МГц, что обеспечивает производительность до 10 MIPS. В состав ядра входит ОЗУ (SRAM) с организацией 256×8, три таймера-счетчика, сторожевой таймер, блок прерываний на 12 запросов, два указателя данных и универсальный асинхронный приемопередатчик UART;
- FPGA с логическими ячейками типа "LUT + триггер" (от 256 до 3200 ячеек для различных микросхем семейства);
- высокопроизводительная конфигурируемая системная шина CSI (Configurable System Interconnect) с двумя восьмиразрядными шинами данных (одной для чтения, другой для записи), 32-разрядной шиной адреса и адресными селекторами. Шина связывает FPGA, периферийные узлы и ядро микроконтроллера при скорости передач до 40 Мбайт/с, поддерживает возможность обращения к внешним устройствам, режим циклического арбитража и реализацию циклов ожидания. Адресные селекторы обеспечивают доступ к устройствам, реализованным в блоке FPGA, и при этом на создание адресных дешифраторов не расходуются ресурсы программируемой логики;
- блок программируемых портов ввода/вывода ПВВ (PIO, Programmable Input-Output port) с программированием обычных или повышенных вы-

ходных токов. Схемы слежения за состояниями входов при отсутствии воздействия на них активного источника сигнала удерживают на входах последнее активное состояние. Возможно индивидуальное переключение линий ввода/вывода в энергосберегающий режим. Уровни сигналов совпадают со стандартом ТТЛ;

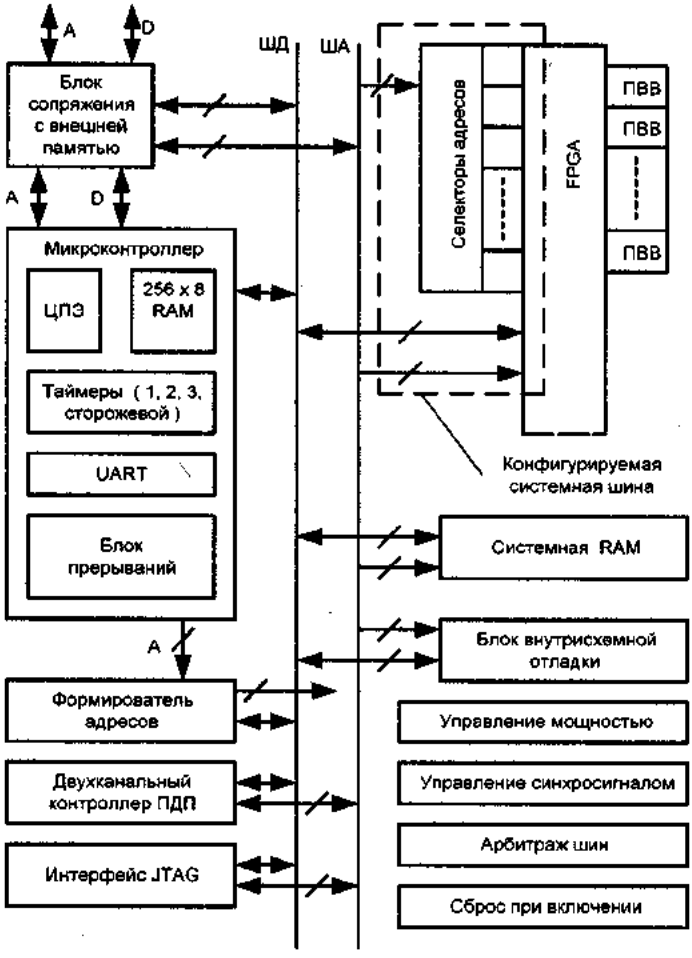


Рис. 10.10. Структура микросхем семейства E5

- блок сопряжения с внешней памятью для подключения к SOPC внешних запоминающих устройств с параллельным или последовательным интерфейсом и емкостью до 256K×8. Возможно дополнительное увеличение

числа адресных линий до 32 и доступ через блок сопряжения с внешней памятью к внешним устройствам с внутренней системной шины;

- контроллер прямого доступа к памяти ПДП или иначе DMA (Direct Memory Access) с двумя независимыми каналами и скоростью передач до 40 Мбайт/с. Передачи сопровождаются подсчетом циклической контрольной суммы. Возможен режим передач "память-память" для организации обмена между различными областями памяти без участия процессорного ядра;
- блок интерфейса JTAG для загрузки памяти конфигурации и исполняемой программы и для внутрисхемной отладки устройств, содержащих микросхемы семейства E5. Через этот блок возможен доступ к внутренней системной шине и адресуемым регистрам кристалла;
- специальный блок внутренней отладки (In-System Debugging Hardware Breakpoint Unit), позволяющий задавать различные условия останова при выполнении микроконтроллером программы. Одновременно можно задавать две точки останова. В состоянии останова разрешен доступ к адресуемым регистрам кристалла через JTAG-интерфейс;
- блок управления энергопотреблением для индивидуального задания энергосберегающих режимов устройствам системы и линиям портов ввода/вывода. В режиме снижения мощности для всех устройств ток, потребляемый кристаллом, не превышает 50 мкА;
- система тактирования с внешним или внутренним задающим генератором. При применении внутреннего генератора его частота задается внешним кварцевым резонатором.

К достоинствам CSOC фирмы Triscend можно отнести преемственность процессорного ядра с хорошо известными процессорами семейства Intel 8051, наличие специального отладочного блока и невысокую стоимость.

Блочные SOPC фирмы Altera

К числу фирм, выпускающих блочные SOPC, в 2000 г. присоединилась и фирма Altera. Производство SOPC этой фирмой началось с рассмотренного ранее семейства APEX20K типа generic (с soft-ядрами). Позднее появились блочные SOPC с hard-ядрами, которые базировались на разработках фирм ARM Limited и MIPS Technologies. Аппаратные ядра процессоров этих фирм вместе с soft-ядром Nios образовали семейство встроенных процессоров микросхем Excalibur.

Процессорные ядра ARM и MIPS оптимизированы для эффективной интеграции с программируемой логикой семейств линии APEX. Реализация в виде hard-ядер позволяет достичь производительности процессоров в сотни миллионов команд в секунду (MIPS) (наименование ядра MIPS совпадает с

наименованием единицы измерения производительности процессоров, хотя, естественно, это вещи разные).

Общий план микросхем Excalibur (рис. 10.11) показывает основные блоки кристалла, в котором роль процессора играют ядра ARM или MIPS. Кроме них на кристалле находятся оперативная память RAM, кэш-память, интерфейсные схемы для связи с внешними устройствами и последовательный порт UART.

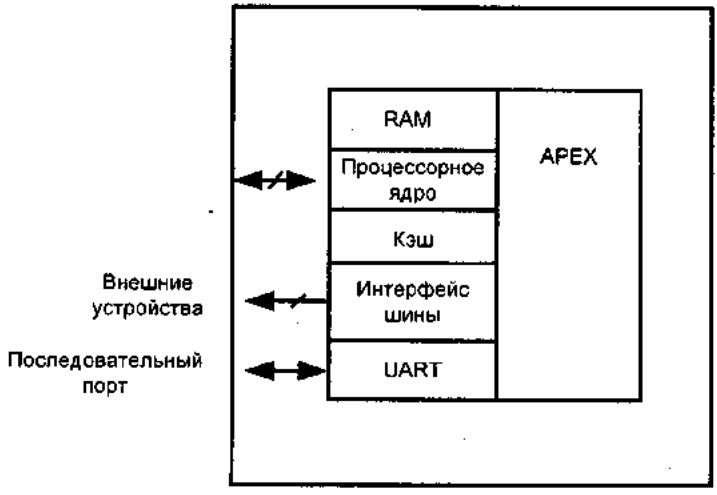


Рис. 10.11. Общий план кристалла микросхем семейства Excalibur

Микросхемы семейства Excalibur с аппаратными процессорными ядрами позволяют создавать системы с высокой производительностью и умеренной функциональной гибкостью.

Планируется развитие линии Excalibur, в частности, реализация 64-разрядных процессорных ядер и расширение набора устройств, реализуемых в области программируемой логики, а также и дополнительных средств для решения задач телекоммуникаций, цифровой обработки сигналов и поддержки обновленных интерфейсных стандартов. Ведется и разработка процессорных ядер для семейства ACEX.

Блочные SOPC с аналоговыми и цифровыми блоками

Блочные SOPC, в которых сочетаются цифровые и аналоговые программируемые блоки *(семейство CY8C25/26)* выпустила фирма Cypress Semiconductor. Микросхемы CY8C25/26xxx имеют восьмиразрядное процессорное ядро

M8C с умножителем/аккумулятором. Возможен выбор частот тактирования. Емкости памяти программ в зависимости от типа микросхемы варьируются (от 4 до 16 Кбайт флэш-памяти), память данных 256 байт SRAM.

В микросхемах CY8C25/26 вокруг быстродействующего процессорного ядра размещены и цифровые, и аналоговые программируемые блоки (SOC-блоки), функционирование которых задается при конфигурировании системы (рис 10.12).

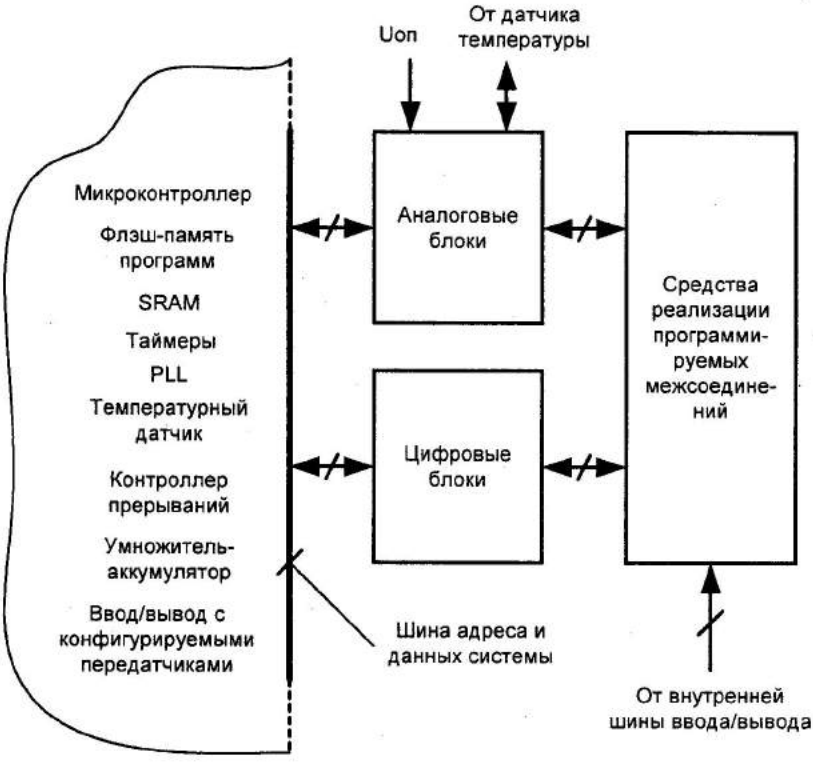


Рис. 10.12. Структура микросхем семейства CY8C25/26 с цифровыми и аналоговыми SOC-блоками

Выделение в отдельную часть цифровых блоков является вариантом организации интерфейса между процессором и FPGA. SOC-блоки могут соединяться друг с другом двумя основными способами: параллельным и последовательным. При параллельных соединениях достигается повышение точности блоков (увеличение разрядности аналого-цифровых преобразователей, повышение разрешающей способности таймеров и т. п.). При последовательных соединениях создаются тракты из нескольких операционных

звеньев, например, последовательное соединение предварительного усилителя, фильтра и аналого-цифрового преобразователя.

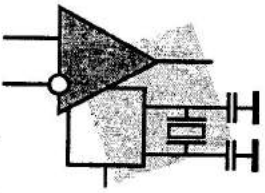
В состав SOC-блоков первого из выпущенных представителей семейства CY8C25/26 входят 12 аналоговых и 8 цифровых блоков. Из 12 аналоговых блоков 4 реализованы в схемотехнике с масштабирующими резисторами (блоки с непрерывными сигналами, CT — continuos time building blocks) и 8 построены на переключаемых конденсаторах (SC, switched-capacitor blocks). Блоки с непрерывными сигналами оптимизированы для построения усилителей с программируемыми коэффициентами усиления и мультиплексируемыми каналами входов и выходов, дифференциальных усилителей и быстродействующих компараторов. Блоки, использующие технику переключаемых конденсаторов, ориентированы на создание программируемых фильтров, регистров последовательного приближения, АЦП, ЦАП и т. п. Цифровые блоки реализуют таймеры, универсальные асинхронные приемопередатчики, широтно-импульсные модуляторы и др.

С помощью аналоговых и цифровых средств коммутации, содержащих мультиплексоры и ключи, SOC-блоки соединяются в так называемые *модули пользователя* (User modules), для формирования которых подготовлена специальная библиотека. Библиотека содержит несколько типов аналого-цифровых преобразователей (последовательного приближения, следящего типа, с дельта-сигма-модуляцией), цифроаналоговые преобразователи, компараторы, устройства выборки и хранения аналоговых данных, разнообразные виды фильтров (низкой частоты, высокой частоты, полосовых), генераторы и детекторы напряжений специальной формы, модуляторы и демодуляторы (всего около 40 аналоговых модулей пользователя).

Цифровая часть библиотеки (около 30 модулей пользователя) включает в себя таймеры, широтно-импульсные модуляторы, приемопередатчики и законченные структуры UART, интерфейс SPI, генераторы CRC, генераторы псевдослучайных последовательностей и схемы управления двигателями.

Литература к главе: [I], [III], [VI], [VIII], [XV], [XX], [XXXI], [XXXIV].

Глава 11



Некоторые аспекты применения микросхем с программируемой структурой

§ 11.1. Конвертация проектов

Одно и то же устройство или система (изделие) может быть реализовано на разных технических средствах — микросхемах с программируемой структурой, базовых матричных кристаллах, в виде заказной микросхемы, спроектированной методом стандартных ячеек и т. д. В течение жизненного цикла могут возникнуть условия, делающие целесообразным перевод изделия на другие средства реализации. Такой перевод и называют *конвертацией проекта.*

На стадии разработки изделия эффективны микросхемы, структура которых может быть репрограммирована пользователем. Репрограммируемость позволяет легко и быстро вносить в проект изменения, доводя изделие до полной работоспособности. Останется ли вариант с репрограммируемыми микросхемами конечной продукцией, зависит от объема его производства. *Если изделие имеет высокий спрос, то, возможно, целесообразно перевести его на реализацию в виде полузаказных или даже заказных схем*, т. к. это позволит устранить из микросхемы средства программирования ее структуры и снизит площадь и стоимость каждого кристалла (при необходимости позволит повысить его быстродействие). Затраты на проектирование фотошаблонов, необходимых для производства полузаказных или заказных схем, и другие однократные технологические затраты (NRE, Non Recurrent Expends) не зачеркнут полученный выигрыш, если тиражность изделия достаточно велика.

На последней стадии жизненного цикла изделия объем его производства снижается и становится неустойчивым. При этом может оказаться выгодным возврат от варианта с полузаказными и заказными микросхемами к БИС/СБИС с программируемыми структурами (FPGA, CPLD, SOPC), т. к. для них не существует проблемы заказа в виде только больших партий и можно избавиться от риска экономических потерь при уценках и распрода-

жах. Кроме того, легко быстро выпустить и дополнительную партию изделий любого объема, если конъюнктура этого потребует.

Как видно из сказанного, для разных ситуаций целесообразными могут быть как переходы "программируемые структуры — фиксированные структуры", так и наоборот. При этом в качестве устройств с фиксированной структурой вначале почти исключительно применялись БМК, позднее стали использоваться и схемы, спроектированные методом стандартных ячеек. Большинство конвертируемых проектов переводится в схемы на БМК. В схемы, реализованные по методу стандартных ячеек и, тем более, в полностью заказные схемы конвертация производится редко, т. к. это экономично только для очень сложных или высокотиражных проектов. Далее для определенности под микросхемами с фиксированной структурой будут иметься в виду БМК.

Для успешной конвертации проектов между разными средствами реализации одних и тех же изделий должна существовать взаимосвязь, поскольку иначе процесс перехода от одних средств к другим окажется затрудненным. Такая взаимосвязь оказывает влияние на архитектуры, электрические характеристики и конструктивные параметры как БИС/СБИС программируемой логики, так и БМК. Например, были выпущены специализированные репрограммируемые FPGA с триггерной памятью конфигурации, предназначенные для макетирования схем, реализуемых на определенных масочных БМК. Были созданы также масочные БМК, по архитектуре идентичные репрограммируемым FPGA. И в дальнейшем FPGA подстраивались под определенные БМК с целью их эффективного макетирования, а определенные БМК приспосабливались по логической структуре, библиотеке схемных решений и электрическим параметрам к конкретным FPGA.

Применительно к проблеме конвертации проектов можно говорить о существовании трех различных зон. В зоне проектов с небольшими логическими емкостями почти все проекты остаются реализованными на микросхемах программируемой логики и в качестве конечной продукции. Далее идет область проектов со средней логической емкостью, для которых могут приниматься различные решения на основе конкретной ситуации. В третьей зоне (для проектов с большой логической емкостью) конвертация как правило целесообразна. Границы зон подвижны и довольно быстро смещаются в сторону увеличения, и, соответственно этому, конвертации в направлении "ПЛИС-БМК" подвергается все меньший процент проектов. Согласно современным данным, средняя зона лежит в области от приблизительно 30 до 70 тыс. эквивалентных вентилей, хотя эти цифры отнюдь не бесспорны. Если исходный проект выполнен на нескольких FPGA, то можно удешевить его конвертацией лишь некоторых FPGA и оставлением других в обычном виде, что позволяет использовать их в традиционном варианте с возможностью реконфигурации.

Для конвертации своих проектов фирма Xilinx несколько лет назад создала микросхемы семейства HardWire FpgASIC, обеспечивающие 100% совместимость контактов ввода/вывода при замещении ими конвертируемых FPGA. Каждая микросхема семейства соответствует функционированию и особенностям конкретной FPGA. Применение этих микросхем в два раза ускоряло процесс конвертации в сравнении с традиционным процессом проектирования на основе БМК. Конвертация основана на заранее созданной и проверенной базе данных, тогда как обычный перевод схемы из реализации на FPGA в реализацию на БМК нуждается в повторной верификации, что сопряжено с большими затратами времени. Следует, однако, отметить, что реальная ситуация с продукцией HardWire не совпала с прогнозом. Ограниченное применение конвертации имеет несколько причин. Во-первых, часто не оправдываются ожидания большой тиражности выпуска изделия, во-вторых, нередко через полгода-год у производителя возникает желание внести в проект изменения, для чего хороши именно программируемые структуры, в-третьих, с усложнением схем и появлением в них разнородных областей все сложнее воспроизвести СБИС ПЛ в БМК или других ASIC с сохранением временных соотношений сигналов, работоспособности и быстродействия схем. Так или иначе, но фирма Xilinx прекратила усилия по внедрению микросхем HardWire и сосредоточилась на удешевлении своих FPGA. Тем не менее работа над проблемами конвертации не остановилась.

Почти одновременно со свертыванием фирмой Xilinx работ над микросхемами HardWire фирма Altera выступила с новым решением задачи перевода проектов, реализованных на ПЛИС, в проекты типа ASIC, объявив о выпуске микросхем HardCopy. Это решение дает удобства для тех, кто использовал ПЛИС фирмы на стадии проектирования и затем намеревается переключиться на реализацию по методу стандартных ячеек для тиражной продукции. Сообщается, что перевод проекта на микросхемы HardCopy дает до 70% экономии площади кристалла при снижении стоимости микросхем и pin-совместимости исходного и конечного вариантов. Переход от проекта, реализованного на ПЛИС, к эквивалентному проекту на ASIC производится приблизительно за 8 недель. С учетом времени, которое будет затрачено заказчиком на верификацию проекта, время до выпуска продукции оценивается приблизительно в 16 недель. Технология HardCopy, по мнению фирмы Altera, дает хорошие предпосылки для повышения эффективности процесса конвертации, причем она нацелена в основном на конвертацию проектов высокой сложности.

В рамках методики конвертации Altera создала версию HardCopy для своих высокосложных СБИС ПЛ Stratix. Перевод отработанного на кристалле Stratix проекта в версию HardCopy снижает стоимость кристалла на 30%, удваивает его быстродействие, снижает потребляемую мощность на 40%.

Фирма AMI (American Microsystems Inc.) извещает об эффективной конвертации в ASIC-реализацию таких сложных ПЛИС, как семейство Virtex фирмы Xilinx и семейство APEX фирмы Altera. Имея для такой конвертации отработанный набор схемных решений, фирма AMI обеспечивает быструю конвертацию проектов, содержащих разнообразные блоки, в том числе DLL, PLL, PCI, блоки памяти с разнообразными организациями, различные варианты схем ввода/вывода. Микросхемы семейства XL-3 фирмы AMI дешевле своих аналогов из семейств Virtex и APEX приблизительно в 4 раза.

§ 11.2. Конфигурирование микросхем

Способ *конфигурирования микросхем* с программируемыми структурами, т. е. настройки их на определенное функционирование, зависит от типа программируемых элементов (ПЭ). Для микросхем с необратимым изменением состояний ПЭ (типов fuse и antifuse) и с энергонезависимой памятью конфигурации (EPROM, EEPROM, Flash) для программирования используются специальные воздействия электрическими сигналами, которые существенно отличаются от рабочих (логических) сигналов. Конфигурирование таких микросхем может производиться вне создаваемой системы с помощью программаторов или же в ее составе (т. е. при сохранении монтажа микросхемы на плате). Запись информации лучше всего реализуется при повышенных напряжениях (например, 12 В при рабочем напряжении 5 В), для чего требуются несколько источников питания или внутренние преобразователи рабочего напряжения в повышенное напряжение программирования.

Загрузка статической памяти конфигурации не требует специальных электрических режимов, и процесс конфигурирования состоит в обычной передаче в микросхему информации по заданному протоколу и с фиксированными форматами данных. Переданная информация обеспечивает создание требуемых соединений в логических блоках, блоках ввода/вывода и подключение блоков к трассам межсоединений. Операция конфигурирования выполняется после каждого включения питания, причем, если установлены специальные загрузочные БИС, то сам факт очередного включения питания автоматически инициирует процесс конфигурирования, который может повторяться неограниченное число раз. Отсутствие специальных электрических режимов для записи информации в память конфигурации обеспечивает возможность ее проведения в работающей схеме, причем возможна и частичная реконфигурация, относящаяся лишь к части системы.

Конфигурирование БИС/СБИС ПЛ со статической (триггерной) памятью конфигурации представляет собою запись во внутренние регистры (триггеры) данных, задающих структуру блоков системы и их межсоединений. Каждый бит настроечных данных задает состояние соответствующему триггеру, управляющему программируемым ключом в настраиваемой схеме.

Микросхемы с программируемыми структурами обычно имеют несколько возможных режимов конфигурирования. Рассмотрим способы конфигурирования на примере SOPC семейства Virtex.

Для конфигурирования этой микросхемы используются как специализированные выводы, так и выводы, которые после завершения конфигурирования могут играть роль выводов общего назначения. К специализированным относятся выводы для задания кода того или иного режима, вывод для синхросигналов, выводы PROGRAM, DONE и выводы граничного сканирования *(см. § 6.8)*. Вывод синхросигнала может быть выходом, когда этот сигнал генерируется микросхемой, или входом, когда сигнал поступает извне.

Возможные способы конфигурирования:

- пассивный последовательный (Slave-serial mode);
- активный последовательный (Master-serial mode);
- байт-последовательный (SelectMAP mode);
- граничного сканирования.

В *пассивном последовательном режиме* микросхема получает данные конфигурирования в виде потока битов из последовательной памяти PROM или другого источника. Синхронизация осуществляется от внешнего источника, каждый положительный фронт синхросигнала вводит бит данных от входа DIN. Несколько микросхем могут быть соединены в цепочку для конфигурирования в едином процессе от общего потока битов. В этом случае после завершения конфигурирования очередной микросхемы данные конфигурации для следующих микросхем появляются на выводе DOUT микросхемы, завершившей конфигурирование.

В *активном последовательном режиме* выходной синхросигнал микросхемы подается на последовательное ЗУ, с которого на вход DIN микросхемы поступает последовательный поток битов конфигурации. Микросхема воспринимает каждый бит по положительному фронту синхросигнала. После загрузки очередной микросхемы, входящей в цепочку, данные для следующей снимаются с выхода DOUT той микросхемы, которая закончила конфигурирование. Для синхронизации процесса можно выбирать частоту из широкого диапазона значений (для рассматриваемой микросхемы в диапазоне от 2,5 до 60 МГц). По умолчанию используется наименьшая частота. Устанавливаемые частоты, естественно, должны соответствовать возможностям используемых PROM и включенных в цепочку микросхем.

В *байт-последовательном* режиме время конфигурирования минимально. Используется байт-последовательный поток данных, которые записываются в микросхему с учетом флажка ее готовности BUSY. Байтовый поток задается от внешнего источника, как и сигналы тактирования, разрешения работы (CS) и записи (WRITE). В этом режиме данные могут и читаться. Если сигнал WRITE пассивен, то данные конфигурации читаются из микросхемы.

Можно также конфигурировать одновременно несколько микросхем, но в этом случае они включаются параллельно по входам синхронизации, данных, WRITE и BUSY. Загружаются микросхемы поочередно путем соответствующего управления сигналами разрешения их работы CS.

В *режиме граничного сканирования* конфигурирование осуществляется исключительно через выводы порта тестирования TAP (Test Access Port) интерфейса JTAG. Используется специальная команда CFG_IN, позволяющая входным данным преобразовываться в пакеты данных для внутренней шины конфигурации микросхемы.

Процесс конфигурирования содержит три этапа: очистка памяти конфигурации, загрузка в нее данных и активизация логических схем, участвующих в процессе.

Конфигурирование начинается автоматически после включения питания, но может быть и задержано пользователем с помощью сигнала PROGRAM, снятие которого запрещает конфигурирование. Завершение очистки памяти выявляется с помощью сигнала INIT, а завершение всего процесса — с помощью сигнала DONE.

Данные для загрузки памяти конфигурации формируются системой автоматизированного проектирования.

Реконфигурация в системе (ISP, In-System Programmability) — одно из важнейших достоинств микросхем, позволяющее легко изменять логику их работы. Потребности в изменениях возникают для устранения не выявленных при первоначальном тестировании ошибок, для модернизации систем и в системах с многофункциональным использованием блоков. Наличие ISP облегчает работу с современными микросхемами, корпуса которых имеют большое число миниатюрных и легко повреждаемых выводов, что делает однократность установки микросхем на плату весьма желательной. Кроме того, реконфигурация микросхемы на расстоянии с использованием средств телекоммуникации или сети Интернет дает и экономическую выгоду, поскольку обходится дешевле, чем проведение этой операции у заказчика.

Возможности ISP растут, если при проектировании часть функциональных ресурсов микросхемы оставлять свободной, имея запас по скорости, функциональным возможностям и ресурсам межсоединений. При реконфигурации в системе должно сохраняться назначение внешних выводов, иначе потребуется изменить монтаж печатных плат.

Среди микросхем с программируемыми структурами имеются и такие, в которых *реализованы одновременно триггерная и энергонезависимая память конфигурации.* В этом случае конфигурирование микросхемы можно производить без внешних источников данных путем автоматической загрузки триггерной памяти из энергонезависимой.

§ 11.3. Защищенность проектов от рассекречивания

Проблема защиты интеллектуальной собственности при использовании микросхем программируемой логики приобретает *особую остроту*. Имея дело с программируемыми схемами, во многих случаях легко воспользоваться плодами чужого труда, т. к. при этом не потребуется что-либо разрабатывать и изготовлять, а нужно лишь получить сведения о содержимом памяти конфигурации и затем загрузить их в готовую микросхему, купленную у поставщика.

Взломщики могут преследовать цели *клонирования* (Cloning), т. е. простого неосознанного дублирования чужих проектов для получения работающих схем без знания их внутреннего устройства. Более сложна задача расшифровки чужих проектов с раскрытием их архитектуры и деталей реализации. Такую операцию можно назвать *реконструкцией* проектов (Reverse-engineering), для нее подходит и термин *"реинжениринг"*. Имея реконструированный проект, можно внести в него какие-либо несущественные изменения и попытаться обойти вопросы лицензирования.

Уязвимость микросхем программируемой логики по отношению к клонированию или реконструкции проектов зависит от характера проекта и схемотехнологии микросхемы.

В начале развития программируемых микросхем, когда они были представлены простыми комбинационными ПЛМ и ПМЛ, проекты можно было реконструировать, подавая на схему все возможные комбинации входных сигналов и фиксируя соответствующие им выходные комбинации сигналов. Из полученных сведений можно вывести булевы функции, воспроизводимые схемой, и, следовательно, реконструировать ее в соответствующем логическом базисе.

Более сложные схемы программируемой логики практически не поддаются такому методу реконструкции. Они имеют большое число вводов/выводов, назначение которых заранее не известно (например, у двунаправленных выводов). Уже одно это создает очень большие сложности для логического анализа проектов, т. к. неизвестно, подавать ли на вывод входной сигнал, снимать ли с него выходной сигнал, или же использовать его поочередно в обоих вариантах. Сложность внутренней структуры микросхем, наличие в них фрагментов, являющихся последовательностными схемами, и различных встроенных функций чрезвычайно затрудняют логический анализ проектов.

Для клонирования проектов нужно раздобыть сведения о содержимом памяти конфигурации микросхемы. По возможностям защиты этой информации от несанкционированного доступа *микросхемы разных схемотехнологий существенно* различаются.

Однократно программируемые микросхемы с пробиваемыми перемычками наиболее защищены от взлома. Для их эксплуатации данные конфигурирования не нужны, поскольку программирование перемычек завершается на стадии изготовления микросхемы. В распоряжении взломщика находится лишь сам кристалл. Для раскрытия проекта требуется определить состояние всех перемычек, получив для каждой ответ на вопрос "замкнута-разомкнута". Это практически невозможно сделать по следующим причинам. Число перемычек очень велико (сотни тысяч). Наблюдением поверхности кристалла нельзя выявить не только состояние перемычек, но и их местоположение, т. к. по виду они не отличаются от простого пересечения шин. Чтобы рассмотреть перемычку и определить ее состояние (наличие проводящей нити), нужно сделать в перемычке несколько поперечных срезов, а это вероятнее всего разрушит остальную часть кристалла. Таким образом, для выявления состояний всех перемычек потребуется испортить очень много кристаллов. Задаваясь вопросом, есть ли проекты, ценность раскрытия которых оправдает затраты на их взлом, можно ответить, что *возможность раскрытия проекта в данных ситуациях есть категория скорее теоретическая, чем практическая.*

Репрограммируемые микросхемы с энергонезависимой памятью конфигурации (EPROM, EEPROM, Flash) в рабочих режимах также не используют файлы конфигурирования. Взломщик, как и в предыдущем случае, имеет в своем распоряжении сам кристалл, в котором скрыта информация о проекте. Чтение памяти конфигурации может быть запрещено битом секретности, сбросить который можно только при операции стирания всего содержимого этой памяти (заметим, впрочем, что в литературе имеются сведения о случаях проникновения в запертую память конфигурации с помощью специальных электрических режимов). Исследовать сам кристалл проще, чем кристалл с перемычками, но также очень нелегко. Если предположить, что расположение транзисторов с плавающими затворами, состояния которых программируются, известно или может быть визуально определено, то задача раскрытия проекта сводится к исследованию состояний каждого из многих тысяч или даже миллионов транзисторов. Наличие или отсутствие заряда в плавающем затворе можно выявить без разрушения кристалла несколькими способами. Заряды создают электрические поля, которые можно обнаруживать специальными методами, можно использовать также электронный микроскоп или материалы, накладываемые на кристалл и изменяющие цвет под воздействием электрических полей. Трудоемкость и стоимость исследования состояний плавающих затворов остаются все же очень высокими, и подобные исследования имеют смысл только для чрезвычайно высокоценных проектов.

Далее, даже если станут известны местоположения и состояния всех транзисторов, то для простого клонирования проекта нужно транслировать эти сведения в битовый поток конфигурирования, а для реконструкции проекта

еще и в саму схему. Решение этих задач очень сложно, поэтому *схемы с энергонезависимой памятью конфигурации можно считать хорошо, хотя и не абсолютно, защищенными от взлома.*

Самыми уязвимыми для взломщиков являются микросхемы *с триггерной памятью конфигурации*, которую нужно загружать при каждом включении питания от внешнего источника данных. Для клонирования проекта достаточно прочитать содержимое этой внешней памяти и использовать его для конфигурирования клонов. Установление соответствия недокументированного битового потока конфигурирования и внутренней структуры схемы (реконструкция проекта) является более сложным, но не считается невозможным.

С целью повышения защищенности проектов (в первую очередь для микросхем с триггерной памятью конфигурации) принимается ряд мер: организационных, юридических, конструктивных и др.

К организационным мерам можно отнести договоренность с поставщиком о поставке немаркированных кристаллов, что эффективно затрудняет попытки взлома проектов.

К юридическим мерам можно отнести встраивание в проект некоторого недокументированного идентификатора (например, инициалов автора проекта).

К конструктивным мерам можно отнести покрытие кристалла и его связей с другими кристаллами непроницаемым слоем (например, эпоксидной смолой). Естественно, это имеет и отрицательные последствия, не позволяя в дальнейшем дорабатывать проект, и ухудшает тепловой режим кристалла. Сходен с этим прием размещения трассы передачи битового потока конфигурирования между энергонезависимой памятью и микросхемой в скрытых внутренних слоях печатной платы.

Можно вообще убрать с платы энергонезависимую память конфигурирования, если разместить на плате автономный источник питания (литиевую батарею), который будет сохранять запрограммированную конфигурацию в самой триггерной памяти микросхемы. Однако в этой ситуации потребуются специальные схемы для изоляции источника автономного питания от всех цепей, кроме памяти конфигурации, что увеличивает площадь кристалла и снижает его быстродействие. Кроме того, при истощении батареи схема все же разрушится. Разрушение может произойти и от мгновенной потери питания вследствие удара, действия помехи и т. д.

Можно использовать в микросхемах с энергонезависимой и триггерной памятью конфигурации методы шифрования информации с помощью сдвигающих регистров с линейными обратными связями и т. п. В этом направлении ведется много исследований, но их рассмотрение выходит за рамки задач этой книги.

§ 11.4. Оценка логической сложности и быстродействия ПЛИС

ПЛИС характеризуются многими параметрами и их подробная классификация по разнообразным признакам сложна и громоздка. К важнейшим параметрам относятся:

- *кратность программирования* (однократное, многократное с ограничением числа циклов, неограниченно многократное), определяемая типом программируемых элементов;
- *уровень интеграции* (максимальный уровень интеграции определяется возможностями технологических процессов);
- *быстродействие* (ограничивается возможностями технологических процессов);
- *структурная организация* (FPGA, CPLD, микросхемы комбинированной архитектуры).

Кроме перечисленных параметров, важную роль играет и ряд других: тип базового логического элемента, постоянство или непостоянство задержек сигналов в путях их передачи, наличие или отсутствие интерфейса JTAG и программирования в системе, совместимость со стандартными интерфейсами, уровни питающих напряжений и режимы пониженной мощности, наличие средств засекречивания реализованного проекта и др.

Способы оценки таких параметров, как уровень интеграции (Density) и быстродействие (Performance) требуют специальных пояснений.

Оценка логической сложности ПЛИС

Логическая сложность микросхемы (уровень интеграции, логическая емкость) оценивается максимальным числом эквивалентных вентилей (обычно это вентили 2И-НЕ), которые могут быть размещены на кристалле. Объективная оценка сложности не так проста, как может показаться на первый взгляд. Нельзя просто подсчитать число эквивалентных вентилей в микросхеме, поскольку их в ней может даже и не быть, а имеющиеся блоки могут не разбиваться на такие вентили.

Рекламируемые изготовителями оценки сложности кристаллов до сих пор получаются по разным методикам и поэтому не вполне сопоставимы. В свое время консорциум компаний под названием PREP (Programmable Electronics Performance Corporation) предложил *набор эталонных схем* для оценки возможностей микросхем программируемой логики. Согласно методике PREP, для измерения сложности кристалла каждая эталонная схема реализуется в нем в максимально возможном числе раз, что дает оценку сложности числом "помещающихся" на кристалле эталонных схем. В качестве эталонных

схем были выбраны более 10 типовых функциональных узлов (регистры, дешифраторы и т. п.). Для эталонных схем известно число эквивалентных вентилей, необходимых для их построения (из практики проектирования на основе БМК, логическими элементами которого как раз и были вентили, принятые в качестве эквивалентных). Таким путем можно выйти и на оценку уровня интеграции микросхемы числом эквивалентных вентилей. Иными словами, сложность ПЛИС определяется их *сопоставлением с некоторым базовым матричным кристаллом* с оценками на базе множества эталонных схем. При этом методика PREP подразумевает усреднение данных по разным эталонным узлам.

СБИС ПЛ первых поколений имели достаточно однородную структуру и для них указывались два числа эквивалентных вентилей — *полное* (total) и *пользовательское* (usable), т. к. не вся схема кристалла может быть использована системотехником для реализации его проекта.

В дальнейшем положение осложнилось. В ПЛИС стали широко применять LUT-блоки, которые можно использовать и в качестве функциональных блоков (табличных преобразователей) и как блоки памяти. При этом для режимов функционального блока и блока памяти одна и та же схема дает существенно разные оценки сложности, т. к. реализации этих режимов на вентилях БМК требуют разных аппаратных затрат. Кроме LUT-блоков в микросхемы стали вводить встроенную память, также способную работать в разных режимах, а также и другие блоки, как, например, блоки коррекции тактовых сигналов, которые вообще относятся к аналоговой схемотехнике. В результате для оценки логической сложности микросхем программируемой логики используются несколько параметров:

- сложность логического массива (произведение сложности логического элемента на число ЛЭ в данной схеме);
- логическая сложность массива памяти при реализации логических функций (произведение логической сложности модуля памяти, работающего в режиме функционального преобразователя, на число таких модулей в данной схеме);
- логическая сложность массива памяти при реализации запоминающих устройств (произведение логической сложности модуля памяти, работающего в режиме запоминающего устройства, на число таких модулей в данной схеме);
- диапазон изменения логической сложности;
- типичная логическая сложность.

Если воспользоваться конкретными цифрами из числа используемых фирмами-производителями СБИС ПЛ, то для схем с LUT-блоками, по данным фирмы Altera можно видеть следующую ситуацию. LUT-блок с четырьмя входами в ре-

жиме функционального преобразователя соответствует 12 эквивалентным вентилям, а в режиме блока памяти с организацией 16×1 тот же LUT-блок дает оценку сложности в 64 эквивалентных вентиля.

Таким образом, в зависимости от режима блоков можно получить разные оценки сложности кристалла, и границы ее значений определяют диапазон изменения логической сложности. Понятие типичной логической сложности определяется с привлечением сведений о пропорциях между частями схемы, используемыми как логические элементы и как блоки памяти (например, при указании, что предполагается использование в режиме запоминающих устройств 20—30% блоков памяти). Дополнительную трудность создают оценки сложности с учетом все более разнообразных специализированных областей кристалла, в том числе и таких, которые содержат аналоговые элементы (как, например, блоки PLL).

В конечном счете следует учитывать, *что оценка числом эквивалентных вентилей является лишь ориентировочной оценкой логической сложности кристалла.*

Наряду с числом эквивалентных вентилей в составе параметров микросхем программируемой логики указывается и *число макроячеек (для CPLD) или логических элементов (для FPGA и комбинированных архитектур).* Но и для этих параметров не все однозначно. Макроячейки имеют разные параметры, сложность воспроизводимых логическими элементами функций также может быть различной, в состав схем часто входят функциональные расширители, цепи переносов и каскадирования, которые не отображаются в виде самостоятельных логических элементов, но могут существенно влиять на функциональные возможности кристалла.

В силу указанных факторов при подборе микросхем ПЛ для реализации своего проекта системотехник должен хотя бы приблизительно "примерить" ресурсы кристалла к составу аппаратной части проекта.

Оценка быстродействия ПЛИС

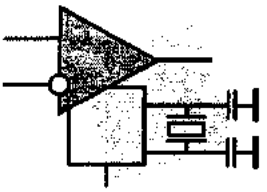
Быстродействие ПЛИС характеризуется задержкой распространения сигнала по указанным путям (pin-to-pin, corner-to-corner, clock-to-output) либо максимально возможной частотой работы схемы в целом *(системная частота)* или некоторого узла (обычно счетного триггера или счетчика указанной разрядности, *частота счетчика* F_{CNT}). Обычно системная частота приблизительно вдвое ниже, чем частота переключений счетного триггера. При наличии встроенного ОЗУ указывается и цикл доступа к памяти.

Задержками распространения сигналов характеризуется, как правило, быстродействие микросхем типа CPLD с их независимостью или малой зависимостью задержек сигналов от конкретного пути их распространения. Пути распространения сигналов в схемах FPGA более разнообразны и для этих

микросхем типичны оценки быстродействия в виде максимально допустимых частот тактирования схем в целом или их отдельных узлов (часто таким узлом является 16-разрядный счетчик). Максимальная частота тактирования сложной схемы (к сожалению, без уточнения ее конкретного характера) задается системной частотой.

Оценка быстродействия ПЛИС дается также с применением понятия *градация быстродействия*. Этот параметр входит в маркировку схемы в виде цифры, перед которой ставится дефис (-1, -2, -3, -4, -5 и т. д.). Для CPLD такая цифра приблизительно совпадает с задержкой распространения сигнала через данную микросхему, для других архитектур такая цифра имеет лишь относительное значение и ранжирует микросхемы семейства друг относительно друга.

Глава 12



Методика и средства автоматизированного проектирования цифровых устройств

В предыдущих главах вопросы проектирования освещались применительно к конкретным темам и отдельным проблемам создания цифровых устройств. Цель этой главы — дать целостную картину процесса проектирования ЦУ и рассмотреть его этапы, имея в виду работу с применением систем автоматизации проектирования (САПР).

§ 12.1. Общее описание процесса проектирования

Проектирование — разработка технической документации, позволяющей изготовить устройство с заданными функционированием и свойствами и в заданных условиях.

Сущность процесса проектирования удачно отражена в работе [17] следующим образом.

Стратегия проектирования — функциональная декомпозиция. Для системы в целом и ее блоков используется концепция "черного ящика". Для "черного ящика" разрабатывается функциональная спецификация, включающая внешнее описание блока (входы и выходы) и внутреннее описание — функцию или алгоритм работы: $F = \Phi(X, t)$, где X — вектор входных величин, F — вектор выходных величин, t — время. При декомпозиции функция Φ разбивается на более простые функции $\Phi_1 ... \Phi_K$, между которыми должны быть установлены определенные связи, соответствующие принятому алгоритму реализации функции Φ. В результате разбиения в конечном счете, получается структура. *Переход от функции к структуре — синтез.*

Проектирование в соответствии с [31] в зависимости от условий и целей рассмотрения целесообразно разделять на аспекты (функциональный, алгоритмический, структурный, конструкторский, технологический), на иерархические уровни детализации внутри каждого аспекта, на отдельные стадии (научно-исследовательская, опытно-конструкторская, технического и рабочего проектирования, испытаний опытного образца и т. д.) и на этапы. Под *этапами проектирования* при этом понимается условно выделенная часть проектирования, сводящаяся к выполнению одной или нескольких проектных процедур, объединенных по признаку принадлежности к одному иерархическому уровню и (или) аспекту рассмотрения.

Процесс проектирования можно рассматривать как создание последовательности моделей, проведение экспериментов над ними и анализ полученных результатов. Модели могут быть концептуальными или имитационными, в некоторых случаях даже физическими прототипами и отражать нужные свойства всей будущей системы или ее отдельных фрагментов. В соответствии с видом модели и эксперименты будут мысленными, машинными или физическими. В процессе анализа проверяется не только правильность работы, но и некоторые показатели, характеризующие будущее устройство. По результатам анализа производится корректировка модели или (если результаты очередного этапа разработки удовлетворяют исходным требованиям) переход к разработке следующих блоков или к следующему уровню иерархии проектирования.

Детализация функций фрагментов (декомпозиция) выполняется до тех пор, пока не получатся блоки, каждый из которых может быть реализован элементами выбранного уровня иерархии.

Все *этапы проектирования многовариантны*. Функциональная спецификация на разрабатываемое устройство может быть задана различными способами. Для конкретизации свойств проектируемой системы может привлекаться тот или иной формальный аппарат. Синтез также неоднозначен. Проект можно реализовать разными способами (в том числе и из компонент различных производителей). Более того, даже готовое устройство можно рассматривать и описывать с различных точек зрения. Устройство (и его модель) могут быть описаны в форме определения выходных откликов на входные воздействия — это *функциональное описание*, или в терминах соединения более примитивных элементов — это *структурное описание*. Как правило, используется смесь обоих способов описания устройств в иерархически или смешанно организованной форме. Физическая реализация проектируемого устройства находит отражение в *конструкторско-топологическом* представлении.

Иллюстрацией взаимосвязи различных уровней иерархической детализации проекта с различными аспектами процесса проектирования может служить обобщенная и модифицированная диаграмма Гайского—Кана [59], [60], приведенная на рис. 12.1. На рисунке процесс проектирования отображается

движением по спирали, начинающимся от функциональной спецификации и завершающимся заданием топологии отдельных технологических слоев обработки кремниевой подложки (полигонов). Естественно, в определенных случаях подобное движение может и начинаться, и заканчиваться на внутренних уровнях. Например, разработка СБИС или печатной платы со стандартными ИС начинается и заканчивается на разных уровнях иерархии. Из рисунка видно, что проектирование современной системы управления, содержащей, например, компьютер, исполнительные двигатели и антенную решетку, может последовательно проходить по всем уровням детализации. А проект, заключающийся в создании одиночной ИС, может начинаться с блочно-функционального представления.

Рис. 12.1. Многоаспектное и многоуровневое отображение процедуры проектирования

Процесс проектирования — многоуровневый, многошаговый и итерационный, с возвратами назад и пересмотром принятых ранее решений.

Последовательная декомпозиция проекта на отдельные фрагменты (с определением функций каждого фрагмента и его интерфейса) присуща всем иерархическим уровням и характерна для разработки широкого класса цифровых устройств, начиная от проектирования отдельных БИС/СБИС и кончая устройствами, содержащими в своем составе большое число микросхем. Такая методология проектирования отображает процесс *нисходящего проектирования* (проектирования "сверху-вниз"): от технического задания до электрических схем, файлов прошивки ПЗУ и конфигурирования программируемых приборов, а также конструкции устройства в целом.

В отличие от нисходящего *восходящее проектирования* (проектирование "снизу-вверх") предусматривает объединение простейших модулей в более сложную структуру до тех пор, пока, в конце концов, не будет достигнут желаемый результат. Исходные модули — это решения, созданные проектировщиком на более ранних этапах работы или в ходе работ над другими проектами, а также модули доступные проектировщику и входящие в состав имеющихся библиотек систем автоматизации проектирования — САПР.

Современным условиям, при которых сложные проекты выполняются с привлечением большого числа разработчиков, преимущественно соответствует смешанная стратегия, объединяющая достоинства обоих подходов.

В соответствии с [1] *декомпозиция заканчивается при получении типовых решений, соответствующих различным уровням иерархии.* При этом учитываются особенности элементной базы и взаимоотношения разработчика и изготовителя с точки зрения конструкторско-топологического аспекта. Так, на верхнем уровне (при многоплатной реализации) декомпозиция заканчивается при представлении проекта в виде отдельных плат, на следующем уровне в виде отдельной платы (типового элемента замены), еще ниже декомпозиция осуществляется до реализации функций с помощью той или иной стандартной микросхемы. А при ориентации на программируемые или разрабатываемые пользователем микросхемы процедура декомпозиции завершается на уровне детализации, определяемом составом технологических библиотек изготовителя или используемой САПР.

Техническая сторона проектирования определяется тремя основными составляющими:

- "строительными кирпичиками проекта" (используемой элементной базой);
- инструментарием проектировщика (при ориентации на разработку аппаратно-программных систем — это САПР);
- методикой использования инструмента и "кирпичей" (обычно для этого используется термин Design Flow — маршрут проектирования).

Три указанных составляющих настолько тесно связаны между собой, что конкретизация одной из них сразу резко сокращает возможные варианты выбора других. Практически невозможно говорить об одной составляющей, не упоминая об остальных. Именно поэтому в различных параграфах этой главы рассматривается один и тот же маршрут проектирования, но упор в каждом из них делается только на одну составляющую. Предыдущие главы дали достаточно информации о различных "строительных кирпичах" современной цифровой схемотехники. В данной главе основное внимание уделено двум оставшимся составляющим процесса проектирования.

Для определенности далее (кроме специально оговоренных случаев) будем считать, что объектом проектирования является печатная плата, содержащая

набор ИС различной степени сложности. Такой объект сохраняет общность в широком диапазоне исходных требований к проекту, поскольку необходимость разработки конструкции печатной подложки возникает даже в случае реализации всех функций проекта в форме одиночной ИС.

Один из основных факторов, влияющих на специфику проектирования, — *тип обрабатываемой информации* и связанные с ним способы ее обработки. Проект или его фрагменты могут включать аналоговые, аналого-цифровые и (или) цифроаналоговые элементы, может строиться на основе дискретных компонентов или микропроцессорных технологий. Отсюда возникает многообразие вариантов проектирования (проектирование для чисто цифровых, смешанных аналого-цифровых, смешанных аппаратных и программных объектов, проектирование с ориентацией на синтезируемые цифровые или аналоговые схемы и др.).

При традиционном разбиении процедуры на этапы системного, структурно-алгоритмического, функционально-логического и конструкторско-технологического проектирования *наиболее существенным представляется этап системного проектирования*. На этом этапе, исходя из требуемого функционирования устройства, проектировщик осуществляет разбиение проекта на отдельные фрагменты, определяет множества входных и выходных сигналов (как устройства в целом, так и его составных частей), их характер и взаимосвязь, а также выбирает способы реализации фрагментов.

Различие теоретической базы, понятийного аппарата и технических средств, привлекаемых на разных стадиях проектирования аппаратно-программных систем, приводит к тому, что процесс их проектирования необходимо разделять как по иерархическому принципу детализации, так и по функциональным отличиям в параллельных ветвях. Подобное разбиение процедуры проектирования после выполнения системного этапа показано на рис. 12.2. Составление маршрутов проектирования необходимо не только для проекта целиком, но и для его отдельных составляющих.

Порядок работы по параллельным ветвям процедуры проектирования произволен и может во времени выполняться как параллельно или последовательно, так и в произвольных комбинациях. Более того, даже этап конструкторско-технологического проектирования при ориентации на современные СБИС может начинаться раньше окончательного завершения работ над отдельными фрагментами проекта.

Общая методология процесса проектирования не зависит от варианта его разбиения на отдельные уровни и ветви, но содержание, методы и средства проектирования для этих уровней и ветвей специфичны и существенно зависят как от типа применяемой элементной базы, так и от способа изготовления конечного продукта.

Прежде всего рассмотрим влияние на процедуру проектирования выбора технологической реализации компонентов проекта.

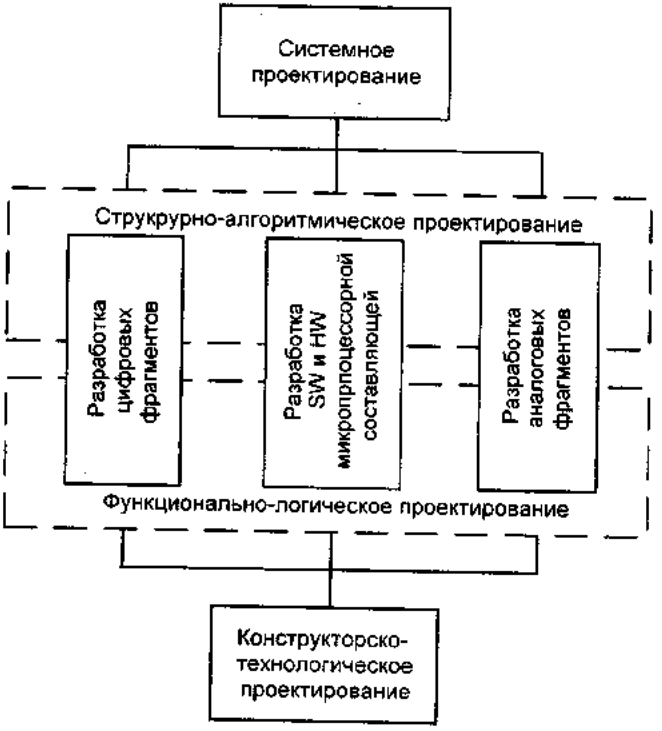


Рис. 12.2. Разбиение типовой процедуры проектирования

§ 12.2. Классификация интегральных схем по характеру их разработки, производства и применения

Выбор технической базы и технологического способа реализации проекта — одна из важнейших проблем, стоящих перед разработчиком. Как правило, одно и то же электронное изделие может быть реализовано различными способами. При выборе должен быть дан ответ на вопрос — будет ли проект построен на стандартных микросхемах или будут использоваться те или иные специализированные ИС и (или) комбинация различных типов ИС.

Классификация ИС по типу их проектирования и производства (массовое или по заказу) и зависящему от этих факторов характеру взаимодействия между заказчиком проекта, разработчиком и изготовителем продукции приведена на рис. 12.3.

К *стандартным* относятся микросхемы разных уровней интеграции (рис. 12.3, *б*). Микросхемы малой и средней степени интеграции МИС и СИС производятся массовыми тиражами в различных схемных и технологических вариантах и реализуют стандартные элементы и узлы, функционирование которых предопределено и никак не зависит от конкретных потребителей. К этой группе ИС принадлежат логические элементы вентильного уровня, буферные элементы, регистровые схемы, к которым относятся и триггерные элементы, а также комбинационные функциональные узлы (ФУ) типа дешифраторов, мультиплексоров и т. п.

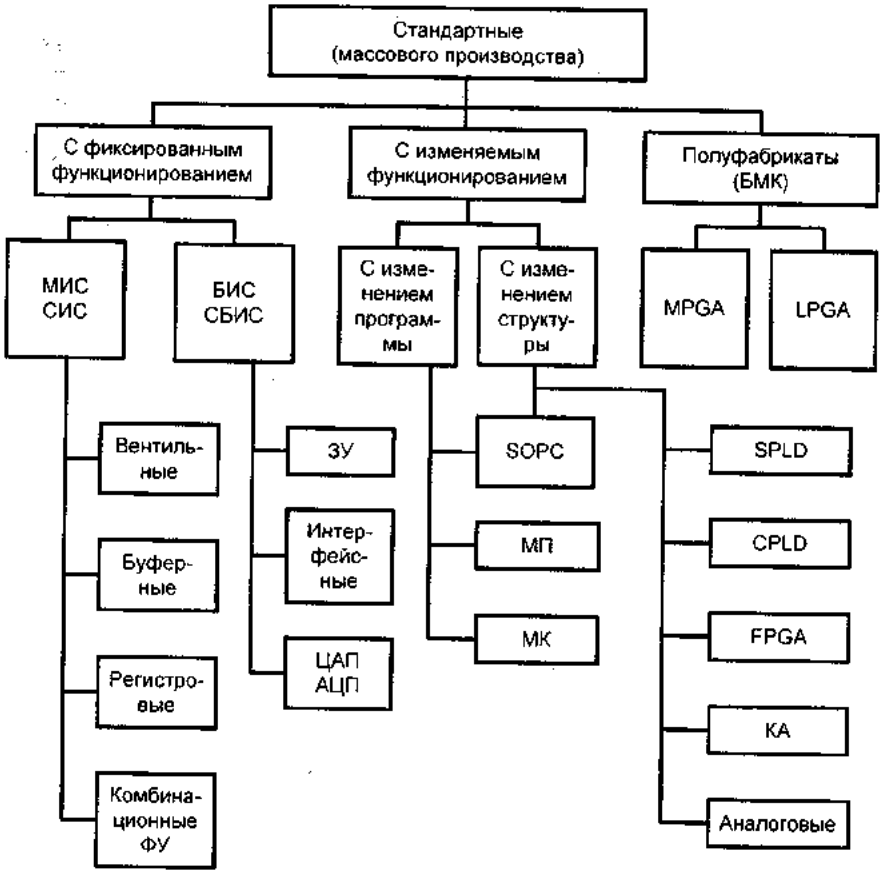
К стандартным схемам высокого уровня интеграции (БИС/СБИС) с фиксированным функционированием относятся микросхемы памяти (ЗУ), интерфейсные схемы микропроцессорных систем и аналого-цифровые схемы: цифроаналоговые преобразователи (ЦАП), аналого-цифровые преобразователи (АЦП).

Общим свойством обеих названных выше групп стандартных микросхем является неизменность их структуры и независимость функционирования от места применения, т. е. от устройств и систем, в которых они используются.

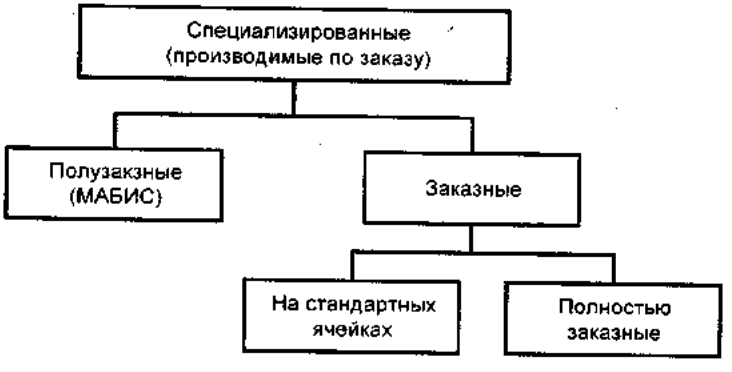
Две следующие группы микросхем являются *стандартными для производителя*, но способны изменять свое функционирование и (или) структуру соответственно нуждам разработчика проекта. При этом такие БИС/СБИС, как микропроцессоры (МП) или микроконтроллеры (МК) настраиваются на решение определенной задачи методом *изменения программы* при независимости структуры от решаемой задачи, а БИС/СБИС с программируемой структурой, напротив, приспосабливаются к решаемым задачам путем *изменения структуры*. Для этой группы микросхем в английской литературе выработан термин ASSP (Application Specific Standard Products).

а

Рис. 12.3. Классификация ИС по типу их производства (*а*)



б



в

Рис. 12.3. Классификация ИС по характеру разработки и применения (*б*, *в*)

Заметим, что с точки зрения терминологии целесообразным было бы различать два метода настройки микросхем на решение конкретных задач и называть МП и МК *программируемыми БИС/СБИС*, а ИС с изменяемой структурой — *конфигурируемыми*. Однако в практике уже закрепилось терминологическое смешение понятий "программирование" и "конфигурирование" (в частности, микросхемы конфигурируемой логики чаще всего называют ПЛИС — программируемые логические ИС).

К стандартной продукции относятся и базовые матричные кристаллы БМК, являющиеся полуфабрикатами, которые используются проектировщиками для заказа на их основе нужных им схем. Подвергая БМК индивидуальным операциям разводки межсоединений элементов, выполняемым на предприятиях электронной промышленности с помощью небольшого числа заказных шаблонов, получают так называемую матричную БИС (МАБИС).

Существуют *две разновидности БМК*. Описанная ранее разновидность с масочным программированием межсоединений относится к MPGA (Mask Programmable Gate Arrays). Другая разновидность — лазерно-программируемые кристаллы (LPGA — Laser-Programmable Gate Arrays). В этих кристаллах стандартный полуфабрикат (полупроводниковый кристалл) доводится до готового изделия путем удаления избыточных соединений лазерным лучом. Проигрывая в стоимости как самих кристаллов, так и процесса изготовления, изготовители обеспечивают очень высокие показатели по скорости создания первых образцов конечного продукта и меньшие затраты при необходимости внесения изменений в проект. Некоторые фирмы — производители MPGA предваряют их массовый выпуск передачей заказчику опытных образцов кристаллов, выполненных по технологии LPGA.

Достоинством стандартных ИС является то, что они по своему функционированию приспосабливаются к требованиям конкретного проекта, но *этот процесс не затрагивает изготовителя*, для которого схемы являются стандартным продуктом со всеми вытекающими из этого выгодами. Стандартные ИС имеют обширный рынок, что благоприятно для снижения их стоимости.

Специализированные ИС (СпИС) в отличие от стандартных ИС массового производства изготавливаются *по конкретному заказу* в соответствии с требованиями реализуемого проекта (рис. 12.3, *в*). В английской терминологии СпИС именуются ASICs (Application Specific Integrated Circuits). Реальные взаимоотношения между изготовителем СпИС и заказчиком зависят от их возможностей и варьируются в широких пределах — от проекта "под ключ" до проектов с передачей масок от разработчика к изготовителю.

Среди СпИС различают классы *полузаказных* и *заказных*, имеющих различную технологию проектирования. Заказные ИС в свою очередь разделяются на полностью заказные и схемы на основе стандартных ячеек.

Полностью заказные схемы целиком проектируются по требованиям конкретного заказчика. Проектировщик имеет полную свободу действий, опре-

деляя схему по своему усмотрению вплоть до уровня схемных компонентов (отдельных транзисторов и т. п.). Для изготовления схемы требуется разработка всего комплекта фотошаблонов, верификация и отладка всех схемных фрагментов. Такие схемы очень дороги и имеют длительные циклы проектирования. Таким способом, например, имеет смысл проектировать очередную модификацию популярного микропроцессора, который будет иметь заведомо широкий спрос и это оправдает высокие затраты на его разработку.

Схемы на стандартных ячейках отличаются от полностью заказных тем, что их фрагменты берутся из заранее разработанной библиотеки схемных решений. Такие фрагменты уже хорошо отработаны, стоимость и длительность проектирования при этом снижаются. Для производства схем тоже требуется изготовление полного комплекта фотошаблонов, но разработка их облегчена. Потери сравнительно с полностью заказными ИС состоят в том, что проектировщик имеет меньше свободы в построении схемы, т. е. результаты оптимизации ее по критериям площади кристалла, быстродействию и т. д. менее эффективны. Наивысших технических параметров добиваются от полностью заказных схем, однако метод стандартных ячеек популярен, т. к. при небольших потерях в технических характеристиках с его помощью можно заметно упростить проектирование схемы. Полностью заказные схемы разрабатываются за время, превышающее время разработки методом стандартных ячеек приблизительно в два раза.

К *полузаказным* схемам относятся МАБИС, о которых сказано ранее. Технология подготовки к изготовлению на основе БМК требуемых МАБИС проще и дешевле по сравнению с заказными ИС. В этом случае стандартный полуфабрикат доводится до готового изделия с помощью индивидуальных межсоединений. Реализация требует изготовления лишь малого числа фотошаблонов (только ответственных за межсоединения). Стоимость и длительность проектирования в сравнении с полностью заказными схемами сокращаются в 3—4 раза, но результат еще дальше от оптимального, поскольку в МАБИС менее рационально используется площадь кристалла (на кристалле остаются неиспользованные элементы и т. п.), длины связей не минимальны и быстродействие не максимально.

Сходство методов проектирования на БМК и стандартных ячейках состоит в использовании библиотек функциональных элементов. Различие в том, что для схем, проектируемых по методу стандартных ячеек, библиотечный набор элементов имеет более выраженную топологическую свободу. Например, стандартизируется только высота ячеек, а их длины могут быть различными. При проектировании из набора библиотечных элементов вначале подбираются необходимые функциональные блоки, а затем решаются задачи их размещения и трассировки.

Методика, а соответственно и САПР для проектирования по методу стандартных ячеек более сложны, чем для проектирования на основе БМК, ко-

торому свойственны более жесткие топологические ограничения. Ограничения вводятся и для метода стандартных ячеек (постоянство высоты ячеек, предопределенность геометрических размеров и положения шин питания, тактирования и др.), но по мере применения более мощных САПР ограничения ослабляются.

Длительность изготовления БИС/СБИС методом стандартных ячеек превышает этот же показатель для МАБИС на основе БМК в 1,3—1,8 раз.

Приведенное классификационное деление является в известной мере условным. Ряд фирм использует методологию проектирования смешанного типа БМК/СЯ (GA/SCI — Gate-Array/Standard Cell Intermix), размещая в одной СБИС области, выполненные по методу стандартных ячеек СЯ и по методу БМК. При этом более компактные и быстродействующие схемы типа СЯ используются в критических трактах обработки сигналов, остальная площадь занимается транзисторами БМК.

Другие фирмы, изготавливающие СБИС, реализованные по методу стандартных ячеек или базовых матричных кристаллов, включают стандартные фрагменты, соответствующие МП системам и ПЛИС, либо в состав кристалла, либо в состав своих библиотек. К какому классу СБИС при этом их относить? Чаще всего этот тип ИС относят к классу *"система на кристалле"*.

§ 12.3. Области применения ИС различных типов

Все типы ИС имеют свои области применения. Каждому типу свойственно определенное соотношение таких параметров, как сложность (достижимый уровень интеграции), быстродействие, стоимость. На выбор типа ИС для реализации проекта влияет совокупность свойств. Основные соображения можно пояснить с позиций экономики, обратившись к формуле стоимости ИС, изготовляемой уже освоенным технологическим процессом:

$$C_{ИС} = C_{изг} + C_{пр}/N,$$

где $C_{изг}$ — стоимость изготовления ИС (стоимость кристалла и других материалов, стоимость технологических операций по изготовлению ИС, контрольных испытаний). Затраты на изготовление относятся к каждой ИС, т. е. повторяются столько раз, сколько ИС будет произведено; $C_{пр}$ — стоимость проектирования ИС, т. е. однократные затраты для данного типа ИС; N — объем производства (тиражность), т. е. число ИС, которое будет произведено.

Стоимость проектирования БИС/СБИС велика и может достигать сотен миллионов долларов. Так, только изготовление набора фотошаблонов обходится в сумму порядка $500 тыс. При всей тщательности отработки проекта не более чем в 50% случаев удается получить положительный результат от

первого комплекта фотошаблонов. Для дорогостоящих вариантов проектирования БИС/СБИС производство становится рентабельным только при большом объеме их продаж.

Затраты $C_{пр}$ и $C_{изг}$ находятся во взаимосвязи. Рост затрат на проектирование, как правило, ведет к снижению $C_{изг}$, поскольку чем совершеннее проект, тем рациональнее используется площадь кристалла и другие его ресурсы. Отсюда видно, *что выигрыш по экономичности могут получать те или иные типы СпИС в зависимости от сложности их производства и тиражности (N).*

Диаграмма областей целесообразного применения разных типов специализированных ИС в зависимости от их сложности и тиражности приведена на рис. 12.4.

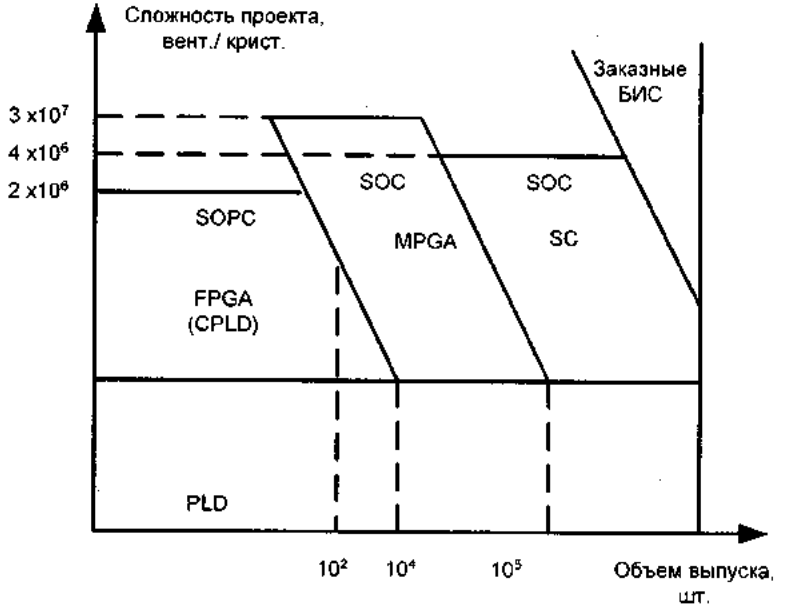


Рис. 12.4. Диаграмма областей целесообразного применения различных типов специализированных БИС/СБИС

Применительно к микросхемам программируемой логики справедливы следующие положения. Простые устройства со сложностью в сотни эквивалентных вентилей целесообразно реализовывать на PLD (PAL, GAL, PLA). При росте сложности проекта естественен переход к FPGA и CPLD, если тиражность ИС сравнительно невелика. Необходимость или целесообразность использования систем на кристалле определяется пока еще не столько

стоимостными (схемы SOC относительно дороги), сколько габаритными и конструктивно-технологическими соображениями, либо с целью повышения быстродействия.

Рост тиражности (приблизительно свыше десятков тысяч) ведет к преимуществам реализаций на БМК, т. к. стоимость изготовления небольшого числа шаблонов для создания межсоединений разложится на большое число микросхем, а стоимость изготовления каждой ИС уменьшится благодаря исключению из схемы схем программируемых связей и средств их программирования.

При еще большей тиражности выгодным оказывается метод стандартных ячеек, позволяющий дополнительно улучшить параметры схемы, плотнее разместить ее элементы на кристалле, т. е. уменьшить $C_{изг}$ и улучшить быстродействие. При этом слагаемое $C_{пр}/N$ в формуле стоимости ИС не окажется слишком большим благодаря большой величине N, хотя необходимость проектировать весь комплект шаблонов для технологических процессов приводит к большим затратам $C_{пр}$.

Полностью заказное проектирование для СпИС не характерно. Оно стоит настолько дорого, что применяется практически только для создания стандартных БИС/СБИС массового производства. Например, проектирование первого 32-разрядного микропроцессора обошлось в свое время в $140 млн, а проектирование современных СБИС требует затрат до $2 млрд.

§ 12.4. Место программируемой логики в процессе создания современной аппаратуры

Проектирование стандартных ИС массового производства, как и проектирование заказными методами вообще, — удел крупных специализированных фирм. На долю системотехников приходятся главным образом другие разработки: цифровых устройств малой сложности на МИС и СИС, микропроцессорных систем для целей управления техническими объектами и технологическими процессами, малотиражной аппаратуры либо прототипов систем на основе ИС программируемой логики.

Проектирование на основе МИС, СИС — наиболее традиционный процесс, в котором используются как эвристические подходы, так и формализованные методики. Проектировщик задает структуру устройства на базе своих знаний, идей и освоения опыта предшественников, а при определении функций отдельных блоков пользуется и формальными методами. Требуется знание типовых функциональных узлов, их свойств и параметров. Можно надеяться, что материал, приведенный в предыдущих главах книги, поможет

проектировщикам успешно разрешать проблемы, возникающие перед ними. Соответствующий материал изложен в [2], [9], [19], [30], [38], [46], [65].

Микропроцессорная система создается в результате разработки комплекса программно-аппаратных средств. Разработка аппаратной части сводится к компоновке системы из типовых модулей: центрального процессорного элемента, различных видов памяти, адаптеров, контроллеров и внешних устройств. Способы подключения модулей к шинам микропроцессорной системы, описания основных модулей, *сведения о методике их программирования и применения приведены в главах 5, 6, 7.* Дополнительно может использоваться литература [21], [27], [34], [37], [50] и др. Довольно много отечественных и зарубежных изданий посвящено *вопросам разработки программного обеспечения.* Достаточно обратиться к [15].

В свое время (около 30 лет назад) именно появление микропроцессоров предоставило широкому кругу разработчиков возможность не только самостоятельно разрабатывать, но и изготовлять устройства с достаточно сложным заданным поведением и высокими эксплуатационными характеристиками. Многообразие реализуемых систем при сохранении конструктивного исполнения достигалось при этом благодаря программной реализуемости требуемого функционирования.

В современных МПС реализуемость проектов с большим объемом программного обеспечения позволяет решать задачи практически любой алгоритмической сложности. Однако при этом выполняется последовательность достаточно примитивных команд, поэтому скорость получения результатов оказывается зависящей от сложности проблемы.

В современных условиях многообразия задач, решаемых на конструктивно совпадающих устройствах, можно достичь при ориентации на альтернативную элементную базу — *программируемую логику.* Структурная организация ПЛИС в наибольшей степени отвечает разбиению исходной задачи на параллельное исполнение отдельных ее составляющих. Поэтому целесообразно применение ПЛИС для решения задач, требующих повышенного быстродействия и допускающих распараллеливание обработки информации. Естественно, еще большие перспективы открываются при совмещении в одном кристалле обеих концепций. Изменение программ и перестройка архитектуры позволяют оптимизировать распределение между программной и архитектурной частями решения задачи в зависимости от требуемых сложности и быстродействия устройства.

Бóльшая часть разработчиков электронной аппаратуры в России ориентируется на применение стандартных интегральных схем. Это обусловлено, с одной стороны, относительно небольшой тиражностью выпускаемой продукции, а с другой стороны, сложностью и дороговизной изготовления заказных и полузаказных ИС у отечественных и зарубежных производителей. Представляет интерес сравнить области целесообразного использования

стандартных ИС различного типа. Для проектов с различной элементной базой на рис. 12.5 приведены зависимости цены комплектации от логической сложности реализации проекта.

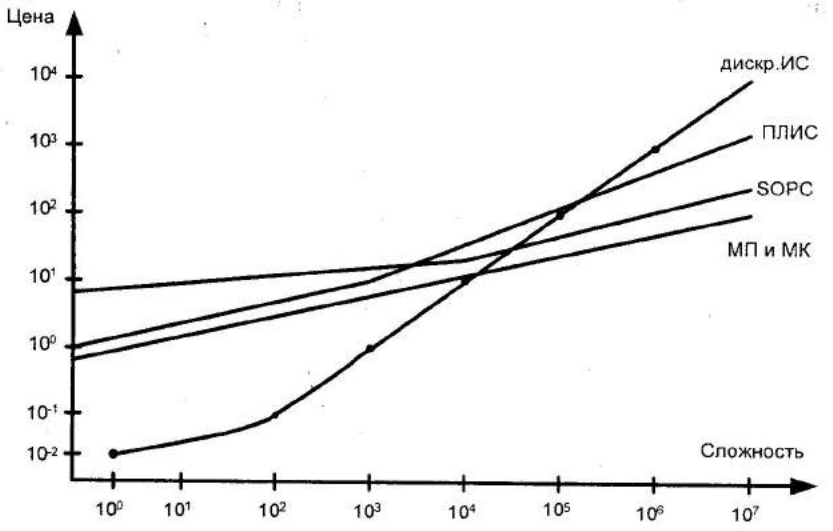


Рис. 12.5. Связь стоимости проектов с их сложностью

На основе дискретных ИС системы целесообразно строить при их относительно малой сложности. Усложнение функций системы ведет к резкому увеличению затрат не только на приобретение комплектующих изделий, но и на изготовление и отладку многокомпонентных печатных плат. При этом не учитывается уменьшение надежности таких плат, увеличение габаритов, потребляемой мощности и т. д. Привлекательность этой элементной базы для многих разработчиков состоит в отсутствии необходимости разрабатывать программное обеспечение.

Для других разработчиков, напротив, возможность реализовать новую продукцию, изменив только программное обеспечение, предопределяет стремление использовать микропроцессорную технику. В 1980-х гг. микроконтроллеры и микропроцессорные системы феноменально быстро завоевали обширный рынок, что не в последнюю очередь обусловлено малым влиянием количества и сложности решаемых задач на требуемые ресурсы.

В настоящий момент от повального перевода всех проектов на ПЛИС или SOPC удерживают две основные причины. Во-первых, относительно высокие цены на комплектацию, а во-вторых, неготовность многих разработчи-

ков переходить на новую элементную базу. В некоторых случаях свою роль играет и отсутствие отечественных ИС этого типа.

Львиной долей инженерных разработок аппаратуры в условиях современной России, по-видимому, как раз и явится *использование схем с программируемой структурой* для создания требуемых устройств и (или) их отладки. При этом программируемые ИС могут использоваться как в виде автономных устройств, так и в составе микропроцессорных систем.

Системы на кристалле и ПЛИС, благодаря определенной структурной избыточности и перестраиваемости, как организации их блоков, так и связей между ними, рационально применять для отработки прототипов будущего устройства или даже на ранних этапах его внедрения.

В SOC связь МП-ядра с периферийным оборудованием осуществляется без выхода за пределы кристалла, что позволяет существенно повысить производительность МП-системы (путем увеличения как тактовых частот, так и разрядностей обрабатываемых данных).

Совмещение в SOC аналоговых и цифровых блоков упрощает проектирование одного из самых сложных и трудоемких этапов — конструирования аналоговой части проекта. Профессионально решенные проблемы реализации аналоговых элементов и их стыковки, как между собой, так и с цифровыми элементами, существенно упрощают работу проектировщиков.

В этой главе основное внимание уделено рассмотрению процедуры проектирования схем с программируемой структурой. Причинами этого являются:

- перспективность реализации проектов на схемах ПЛИС;
- недостаточное освещение этих вопросов в отечественной литературе;
- в процедуре проектирования ПЛИС содержатся почти все проектные процедуры, характерные для других типов СпИС;
- специфические этапы проектирования других типов СпИС менее интересны большинству разработчиков, поскольку выполняются изготовителем ИС, а не ими.

§ 12.5. Инструментарий проектировщика

Современное проектирование (даже для проектов не очень высокой сложности) немыслимо без привлечения *инструментальных средств*. Среди них ведущее место занимают системы автоматизированного проектирования — САПР. Современная переориентация ведущих фирм разработчиков САПР с дорогих и поэтому имеющих ограниченное распространение рабочих станций на достаточно дешевые персональные компьютеры способствует расширению областей применения автоматизированного проектирования.

Чем сложнее (и дороже) проект, тем важнее автоматизация каждого этапа проектирования и тем более сложные средства могут и должны привлекаться для реализации каждого этапа. Для оценки современного состояния проблемы проектирования электронных систем приведем некоторые количественные характеристики больших проектов:

- для схем ASIC — более 20 млн вентилей в кристалле, для ПЛИС — более 5 млн вентилей;
- для описания поведения проекта на системном уровне требуется более 0,5 млн строк кода на языке C;
- при описании проекта на уровне регистровых передач используется более 5 млн строк кода RTL.

Сложность, широкая номенклатура доступных проектных средств, большая стоимость полного проектного комплекса привели к использованию фирмами — поставщиками САПР понятия *платформа проектирования*. Платформа — это уже не просто компьютер с набором всех программных пакетов фирмы, а комплекс только тех аппаратно-программных средств, который необходим для решения конкретной задачи разработчика. Понятие платформы включает не только набор скомпонованных и взаимно состыкованных средств, но и методику проектирования на его основе.

Средства системного этапа проектирования

Как уже отмечалось (см. рис. 12.2), проектирование начинается с *системного этапа*. На этом этапе формируются *основные идеи* проекта и анализируется его реализуемость. Определяются ориентировочные затраты и стоимость конечного продукта. Выбирается платформа проектирования. Именно здесь закладывается набор требуемых далее средств. Выполнение работ этого этапа очень плохо поддается автоматизации.

На системном этапе выбирается характер технологической реализации отдельных ИС (стандартные дискретные компоненты, ASIC, FPGA, CPLD и т. д.), определяются производители и выбираются конкретные семейства используемых ИС.

С другой стороны, на этом же этапе общесистемная задача разбивается на функционально обособленные фрагменты. Основа разбиения — ориентация на реализацию фрагментов на специфическом элементном базисе (аналоговом, цифро-аналоговом, микропроцессорном, базисе цифровых микросхем с фиксированными или программируемыми структурами и т. д.).

Для конкретизации данных об отдельных составляющих требуется определение общесистемных характеристик. Здесь и привлекаются ПК с соответствующими программными пакетами. Для дискретных фрагментов чаще всего используется математический аппарат теории массового обслуживания

и соответствующие языки (GPSS, Simula). При анализе частей проекта, описываемых в понятиях теории динамических систем, ориентируются на программные пакеты MathCAD, MATLAB и SIMULINK. Исследования аналоговых и цифроаналоговых фрагментов выполняются с привлечением программных пакетов, основой которых является модификация программы схемотехнического моделирования Spice (Simulation Program with Integrated Circuit Emphasis).

В результате совместной работы заказчика и проектировщика на основании оценки требуемого времени и ресурсов разработки создается функциональная спецификация на каждую обособленную ветвь проектирования, включая перечень средств, необходимых для последующих этапов проектирования.

Разработка специфических фрагментов проекта

Последующие этапы проектирования выполняются с привлечением средств автоматизации. Следует отметить общность процедур проектирования по всем параллельным ветвям (независимо от специфики элементной базы). Как разработка программного обеспечения для МП (МК) ядра, так и разработка дискретной и аналоговой частей проекта укрупненно могут рассматриваться как последовательность трех этапов работы с САПР, выполняемых после составления спецификации:

- ввод в САПР исходной для проектирования информации;
- обработка введенной в САПР информации;
- обработка и анализ полученных результатов.

Конкретное содержание этапов для аппаратной и программной части проекта (а тем более, цифровой и аналоговой частей), естественно, различное. Разработка аппаратной части проекта приводит к синтезу устройства (или устройств) в базисе заданных элементов, а компиляция программной части проекта приводит к синтезу кодового представления программ. Полученные результаты требуют тщательной проверки, поэтому за этапом синтеза следует этап анализа, проводимого моделированием и теоретической верификацией.

Моделирование, как правило, имеет несколько уровней с разной степенью отображения свойств реального объекта. На самом верхнем уровне проверяется правильность функциональной организации устройства или программы. На следующем уровне учитываются некоторые особенности структурной реализации объекта (элементная база, способ взаимодействия программных фрагментов и т. д.). На самом низшем уровне учитываются особенности физической реализации устройства (задержки сигналов в схемах устройства, связанные с конкретным размещением элементов, время исполнения отдельных программных фрагментов, влияние параметров трассировки на работу схем, учет паразитных эффектов, шумов, температуры и т. д.). В ре-

зультате моделирования могут выявиться ошибки, требующие исправления, что придает процессу проектирования итеративный характер с возвратами к прежним этапам и введением в проект нужных коррекций.

Более того, по мере отработки решений по отдельным ветвям проектирования все большее значение приобретает анализ их взаимодействия. Наиболее остро эта проблема встает при стыковке программной и аппаратных частей проекта. Традиционно они детально анализировались и отрабатывались только на этапе комплексной отладки проекта. Естественно, такое последовательное проектирование (тем более с учетом многократных итерационных возвратов к началу проектных процедур) существенно замедляет проведение работ. Поэтому понятно стремление разработчиков использовать унифицированные технические средства и приемы, позволяющие ускорить процесс проектирования.

Средства разработки процессорной части проекта

При разбиении проекта разработчик, прежде всего, оценивает целесообразность применения процессорного блока со стандартной архитектурой. Наличие процессора позволяет попытаться возложить именно на него решение большинства задач проекта. Практическая невозможность поручить процессору решение всех задач заставляет выделить специфические задачи, которые (как уже отмечалось ранее) могут быть отнесены к одной из двух групп со специфическим подходом к их решению. К первым относятся задачи, требующие аналоговой и аналого-цифровой обработки. Ко вторым — задачи, связанные с быстродействием, не поддерживаемым выбранным типом МП.

Разработка процессорного фрагмента предполагает, как показано на рис. 12.6, выполнение двух параллельно решаемых задач — проектирования аппаратной и программной (в английской терминологии HW (Hardware) и соответственно SW (Software)) составляющих процессорной системы. Наиболее ответственным при этом является оптимальность распределения задач между ними. Традиционная методика проектирования аппаратной части микропроцессорных систем на основе дискретных компонентов предполагает проектирование электрических схем и их моделирование. Электрические схемы лучше всего создавать с помощью программных пакетов, поддерживающих в дальнейшем автоматизацию разработки и изготовление печатных плат. К числу таких пакетов относятся P-CAD, OrCAD [39], Protel и некоторые другие. Наиболее мощные пакеты позволяют моделировать поведение аппаратуры не только на логическом уровне, но и с учетом паразитных эффектов в печатной подложке.

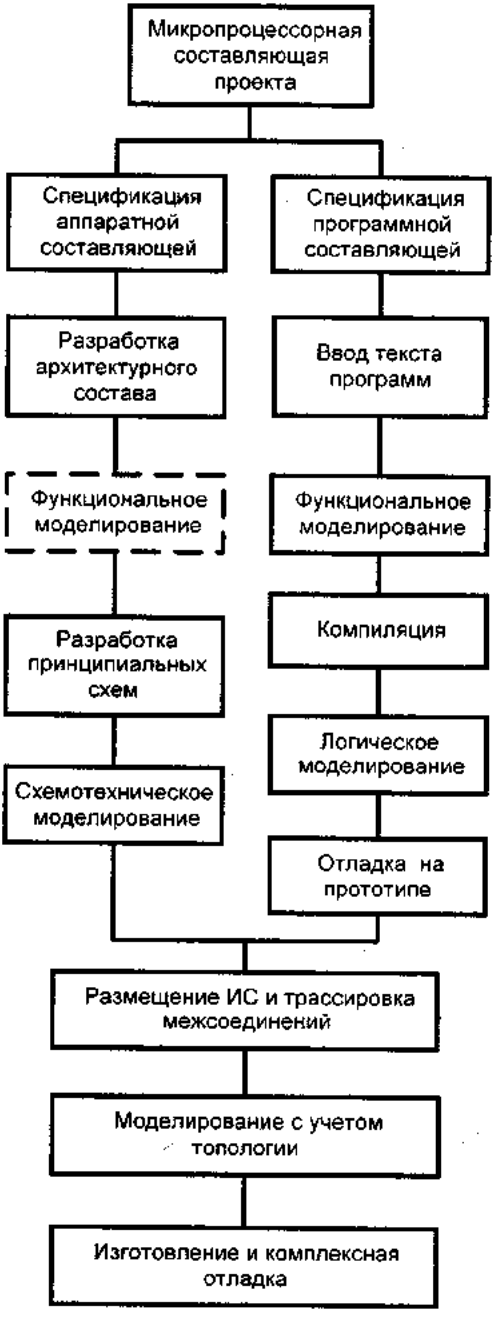


Рис. 12.6. Этапы разработки процессорной составляющей проекта

Для МПС, интегрированных в одном кристалле (МК или SOPC), необходимость отдельного моделирования аппаратной части, как правило, отсутствует. Необходимость определяется только степенью отработки архитектуры используемого МП ядра. Как правило, такая проверка может быть отодвинута на более отдаленные стадии проектирования и совмещена с комплексной отладкой.

Самая ответственная задача в разработке МПС — разработка программного обеспечения. Ключевым средством этого процесса является *компилятор* — компьютерная программа, упрощающая написание и отладку пользовательских программ. Обычно используемые языковые средства — либо язык ассемблера, либо C/C++. Сложность и длительность процедуры отладки заставляет привлекать разнообразные аппаратно-программные средства. Обычно весь комплекс таких отладочных средств, сконцентрированных вокруг отладочного (инструментального) компьютера, называют *средой или системой разработки*. Отладка может ориентироваться на работу с виртуальным прототипом (моделью), с физическим прототипом или с конечной продукцией.

Отладка на модели — способ, доступный с самых начальных этапов проектирования. Достоинство метода — наблюдаемость (видимость) и управляемость всеми объектами модели. Степень учета в модели поведения не только программной составляющей, но и аппаратуры проекта определяет свойства и название системы моделирования. Чем больше аппаратная часть, тем больше замедляется работы модели, но точнее отражаются свойства объекта. Программное моделирование работы только процессорного блока носит название *кросс-ассемблирования*. Работа с программной моделью процессора и моделями стандартной периферии системы — *симуляция*. Совместная работа программной модели одной части проектируемой системы с физически реализованной другой частью носит название *эмуляции*. Под эмуляцией понимают и работу с физической реализацией фрагмента проектируемой системы, отличающейся от реализации, планируемой в конечной продукции. Программной моделью может быть любая часть системы или процессор, или периферийные блоки. Адекватность задания асинхронного взаимодействия модели среды и модели системы — основная проблема, с которой сталкивается проектировщик при работе с программами-моделировщиками.

Отладка на прототипе — способ получения наиболее достоверных данных о работе разработанного программного обеспечения в его взаимодействии с аппаратурой системы и внешней средой.

Конечно, только отладка на конечном продукте при задании различных вариантов взаимодействия с окружающей средой может дать разработчику уверенность в правильности взаимного функционирования аппаратуры и программного обеспечения.

Средства разработки цифровой части проекта

Цифровая часть проекта может быть выполнена на дискретных компонентах с фиксированной или (и) программируемой логикой (ПЛИС). Целесообразность использования тех или иных компонент с точки зрения затрат и достигаемых характеристик проектируемого устройства рассмотрена ранее (см. рис. 12.5), в этом разделе акцент делается на различиях в маршруте и средствах проектирования. Последовательность разработки цифровой части проекта приведена на рис. 12.7.

Технология *проектирования на дискретных ИС* имеет длительную историю. Несмотря на постоянное увеличение логической мощности дискретно монтируемых ИС, многое в процедуре проектирования осталось прежним. Современный элементный базис не исключает использования дискретных ИС, т. к. выполнение ряда функций (гальванической развязки, формирования тактовой частоты, преобразования питающих напряжений и т. д.) не включается в современные ПЛИС и даже SOPC.

В проектировании на дискретных компонентах следует выделять *две составляющие*: конструкторскую и схемотехническую. Конструкторское проектирование, не меняясь в своей сути, медленно эволюционирует под влиянием технологических изменений в организации корпусов ИС, способов их монтажа и т. д. Изменения в схемотехнической разработке более динамичны. Реализация на СИС и МИС тесно связывает схемотехническое и конструкторское проектирование, в отличие от реализации на ПЛИС, когда взаимосвязь этих работ все более ослабляется.

Основа схемотехнического этапа работ — составленная спецификация проекта, содержащая функциональные и структурные схемы будущего устройства. Этот этап заканчивается составлением принципиальных электрических схем соединений ИС, выбранных для реализации проекта. В состав этих схем могут быть органически включены и аппаратные ресурсы микропроцессорной части. Разработка электрических схем хорошо поддерживается современными САПР с ориентацией на последующее использование электрических схем при конструировании печатных плат. Описания конструктивных параметров элементов дискретной логики включаются в технологические библиотеки фирм — изготовителей ИС и могут быть найдены разработчиками в Интернете. Если вся проектируемая система размещается в одной интегральной схеме, разработка конструкции печатной платы может быть начата задолго до завершения работ над внутренними схемами ИС.

Другая ситуация складывается с моделированием разрабатываемых схем. Программные пакеты ряда фирм (Micro-Cap, DesignLab, Aplac, Electronics Workbench и другие) были разработаны для моделирования на схемотехническом уровне. Однако уровень элементного базиса был очень низок и в лучшем случае соответствовал схемам малой степени интеграции. Это связано с тем, что эти пакеты были ориентированы на совместное моделирование цифровых и аналоговых фрагментов и базировались на языках и средст-

вах смешанного представления сигналов (типа Pspice, основанного на решении дифференциальных уравнений).

Рис. 12.7. Этапы разработки цифровой составляющей проекта

Основной проблемой для большинства современных САПР является интеграция в их систему моделирования моделей компонент, описанных на языках высокого уровня. Попытку включить в состав своего пакета конструкторского проектирования моделирующую программу одной из первых предприняла фирма Personal CAD Systems. Система моделирования PC-LOGS для дискретных компонент была основана на троичном асинхронном событийном моделировании и позволяла создавать собственные модели, дополняя ограниченную библиотеку встроенных моделей типовых компонентов. Пользовательские модели могли строиться на уровне структурного объединения элементов низшего уровня иерархии или на уровне поведенческого описания компонентов. Для создания моделей предлагался специальный язык описания поведения. Особенно большого распространения версия в DOS не получила, а в варианте Windows она даже не была реализована.

Наиболее известна мощная сквозная система проектирования DesignLab, ее возможности в версии 8.0 подробно изложены в книге [40]. Попытки решить в рамках одной системы все вопросы схемотехнического и конструкторского проектирования пока приводят к очень сложным, громоздким, неоперативным и, конечно, дорогим системам. В современных САПР влияние физической реализации проектов для простоты учитывается с функциональным выделением этой проблемы. Например, в большинстве САПР осуществляется обособленный анализ электрических характеристик цепей печатных плат для учета их паразитного влияния на передаваемые сигналы или внешние узлы.

Ввиду важности и специфичности разработка цифровой части проекта на ПЛИС будет детально рассмотрена в следующем параграфе.

Средства разработки аналоговых и аналого-цифровых фрагментов

Аналоговые и аналого-цифровые фрагменты проекта (если они не интегрированы в SOPC) могут быть (как и цифровые фрагменты) реализованы в форме совокупности отдельных ИС с фиксированной архитектурой и параметрами либо в форме программируемых аналоговых ИС (ПАИС). Поэтому процедура проектирования таких фрагментов внешне совпадает с последовательностью, приведенной на рис. 12.7 для цифровой части проекта. Вместе с тем следует отметить, что, хотя объемы аналоговой части, как правило, не превышают 15% от общего объема проекта, их отладка "съедает" до 80% общего времени отладки. Те же соотношения характерны и для разработки топологии печатных плат с аналоговыми элементами. При ориентации на дискретные компоненты этап разработки электрических схем отличается от соответствующего этапа для цифровых компонент только набором исходных ИС.

Сложность проектирования аналоговых и смешанных схем определяет важность этапа их моделирования. Возможности отечественных проектировщиков использовать программные моделировщики ведущих мировых фирм ограничиваются высокой (сотни тысяч долларов) стоимостью лицензий на эти пакеты. Сердцевиной большинства таких систем являются современные разновидности программы Pspice. Последние разработки ведущих фирм, таких как Cadence [VII], представляют собой сквозные системы проектирования печатных плат, что позволяет моделировать проект до и после автоматической трассировки соединений его компонент. Моделирование с учетом паразитного влияния разнообразных элементов топологии печатной платы существенно повышает эффективность разработки, увеличивает ее достоверность и сокращает общее время проектирования.

Работы и средства этапа комплексной отладки проекта

После завершения разработки принципиальных электрических схем фрагментов на цифровых, аналоговых элементах, создания файлов прошивки памяти команд МП и файлов конфигурирования программируемых БИС, а также проработки конструктивной реализации системы возможна реальная проверка работы всего устройства, либо его отдельных фрагментов — физическая реализация проекта или его частей. Однако для экспериментальной проверки принятых решений совсем не обязательно ждать реализации будущей системы целиком. Для отладки могут использоваться средства моделирования, эмуляции (программной и аппаратной) и средства физического прототипирования. Естественно, предпочтительной представляется практическая проверка проекта с использованием предлагаемых различными фирмами специальных отладочных средств. Названия таких средств отражают их целевую направленность. Спектр отладочных средств простирается от предназначенных для предварительного знакомства с БИС рассматриваемого класса (обычно называемых Starter Kit) и средств для отладки прикладных проектных решений (обычно называемыех Evaluation Board или Development Board) до средств, замещающих на начальных этапах выпуска готовой продукции оборудование, которое еще находится на этапах конструкторско-технологической разработки (обычно называемых Prototype Plate). Несмотря на некоторые отличия по имеющимся ресурсам и предлагаемым возможностям, любая плата позволяет производить конфигурирование системы под требования проектировщика и выполнять желаемые эксперименты.

Помимо перечисленных ранее средств применяются и традиционные средства отладки, такие как осциллографы, программаторы, эмуляторы, логические анализаторы и т. д.

Специфика конструирования и отладки проектов на ПЛИС и SOPC

Существуют специфические особенности процедуры создания и отладки проектов, конструктивно расположенных в одном кристалле — ПЛИС или SOPC. Хотя функциональное назначение элементов, входящих в ИС этих классов, остается таким же, как и при реализации на дискретных компонентах, резко возрастает как сложность самих проектов, так и интеллектуальная мощность привлекаемых отладочных средств. Возможность упреждающей конструкторской разработки приводит к тому, что самым "узким" местом становится комплексная проверка проекта по параллельным направлениям. Все усилия проектировщиков направлены на то, чтобы как можно раньше приступить к комплексным экспериментам.

Следует выделить *два основных момента*. Первый связан со сложностью отладки современных проектов традиционными методами и средствами. Второй связан с легкостью и оперативностью внесения изменений в проект на любом этапе (в том числе даже после передачи физических образцов заказчику).

Интеграция в одном кристалле огромной логической мощности сопровождается уменьшением количества внешних контрольных точек системы (вывод некоторых промежуточных точек проекта на выходные контакты бывает даже недопустим, т. к. может приводить к потере производительности проекта). Отсюда следует неэффективность применения традиционных методов отладки, базирующихся на подключении разнообразной контролирующей аппаратуры к тестовым точкам отлаживаемой системы. Поэтому резко возрастает интерес к системам моделирования, позволяющим при отладке одновременно наблюдать поведение совместной работы аппаратуры и программ.

Другой подход, открывающий проектировщику доступ к внутренним элементам проекта, состоит в интеграции фрагментов отладочных средств в состав целевой системы. Допустимыми считаются затраты на эти цели от единиц до десятков процентов общих ресурсов БИС. При использовании расширенного варианта интерфейса JTAG подобный подход даже не требует увеличения числа контактов БИС. Более того, репрограммируемость ПЛИС и SOPC позволяет загружать в них специальные тестовые конфигурации, которые ориентированы на контроль и (или) настройку оборудования, расположенного вокруг БИС.

Ввиду легкости репрограммирования аппаратной части проекта, для SOPC и программной части этап экспериментальных работ с БИС может быть отложен до завершения конструкторской разработки печатной платы. Даже значительные изменения схемы обычно не влекут за собой столь катастрофических последствий, как в проектах на основе жесткой стандартной логики. Наличие у фирм — разработчиков ПЛИС и SOPC широкой номенклатуры БИС раз-

личной логической мощности с совпадением общего числа выходных контактов, их функционального назначения и расположения делает модификацию проектов вопросом скорее экономическим (более мощные БИС стоят дороже), чем техническим. Поэтому экспериментальные работы целесообразно производить на различных отладочных прототипных системах исключительно для ускорения общего процесса проектирования, поскольку в этом случае удается совместить во времени этапы конструкторской разработки печатной платы и экспериментальных работ с разработанным проектом.

Итак, особенности БИС с программируемой структурой приводят к определенным изменениям не только в процедуре изготовления конечной продукции, но и в процедуре контроля и отбраковки готовой продукции.

§ 12.6. Системный этап проектирования цифровых устройств на базе ПЛИС

Все современные методики проектирования цифровых устройств (ЦУ) на базе сложных программируемых БИС/СБИС основаны на применении САПР. Однако прежде чем приступить к работе с ними проектировщик как минимум должен решить, на какой САПР он будет выполнять проектирование. Выбор САПР, на которой (или на которых) будут выполняться дальнейшие работы, не в последнюю очередь связан с выбором средств описания проекта и его частей и способах занесения информации об этих основных частях в САПР. Для этого требуется предварительно представить исходную проектную проблему в блочно-функциональном виде. Все эти подготовительные работы на рис. 12.2 были включены в этап под названием "Системное проектирование". Автоматизация всех работ этого этапа в настоящий момент не выполняется, а ведущие мировые фирмы — производители САПР стремятся автоматизировать лишь отдельные работы этого этапа.

Выбор САПР

Правильный выбор САПР — важнейшее условие эффективного проектирования и ускорения выпуска продукции, т. е. сокращения времени до продажи (по-английски Time-to-Market).

В общем случае при таком выборе приходится учитывать целый ряд соображений, а именно:

- распространенность САПР;
- цену САПР, ее сопровождения и модификаций;
- широту охвата задач проектирования и эффективность выполнения различных этапов и уровней проектирования;

- удобство работы с САПР и ее "дружественность";
- наличие широкой библиотечной поддержки стандартных решений;
- возможность и простоту стыковки с другими САПР;
- возможности корпоративной работы.

Очевидно, трудно рассчитывать на такой выбор САПР, который удовлетворял бы сразу всем зачастую взаимно противоречивым требованиям и свойствам.

Ситуация облегчается выбором маршрутов проектирования, основанных на *групповом использовании* САПР, когда каждая из них обеспечивает наилучшие решения для соответствующего этапа проектирования. Целесообразности такого подхода способствует и позиция фирм — производителей ПЛИС. Необходимость использования САПР этих фирм, как минимум на последних этапах компиляции проекта (связь смонтированного проекта с конфигурационными файлами), с одной стороны, и малая эффективность этих САПР на этапах синтеза (в меньшей степени и моделирования), с другой стороны, определяют целесообразность привлечения в маршрут проектирования САПР сторонних фирм. При экономической оценке подобных подходов следует учесть универсальность этих САПР и легкость их переориентации на элементную базу других фирм — производителей ПЛИС.

Представление проекта на блочно-функциональном уровне

Первая работа, выполняемая разработчиком над проектом, — переход от технического задания (ТЗ) к формализованному описанию проектируемого устройства. Как правило, ТЗ является смесью словесного и технического описания, его формализация приводит к вычленению основных блоков устройства (или алгоритма) и определению их связей и (или) взаимодействия. Эта процедура носит название — *спецификация проекта*. Для спецификации проектов, содержащих микропроцессорную составляющую, чаще всего используются специальные разновидности языка C++ [VII, XXVII].

В сущности именно в этот момент реализуются начальные действия первого этапа проектирования. Формально же первый этап — разбиение задачи на отдельные функционально обособленные подзадачи — этап декомпозиции. Разработчик составляет содержательные граф-схемы алгоритма и (или) функциональные блок-схемы устройства.

Способ и средства разбиения чаще всего и прежде всего определяются симпатиями проектировщика и лишь иногда являются предопределенными. Сама форма ТЗ может провоцировать проектировщика на использование тех или иных средств, хотя не исключено, что более эффективным мог бы быть другой метод описания проекта или его фрагментов. Для спецификации

микропроцессорной составляющей проекта проектировщики обычно стремятся к составлению схем алгоритмов, а для аппаратной части — к составлению функциональных схем. Первый вариант больше тяготеет к последующей программной реализации, а второй приближает к уровню регистровых передач.

Как уже указывалось, возможно как только временное (поведенческое), так и только пространственное (архитектурно-структурное) описание проекта. Однако обычно целесообразно совмещать обе возможности.

При разработке ЦУ бывает естественным разбиение его на два блока: операционный и управления. Операционный блок (ОБ) выполняет преобразование данных и строится из стандартных частей, а блок управления (устройство управления УУ) обеспечивает необходимую последовательность операций, выполняемых в ОБ (одном или нескольких). Для этого УУ передает на входы ОБ управляющие сигналы. Последовательность действий и, следовательно, управляющих сигналов зависит от результатов операций в ОБ и внешних воздействий. Отсюда видно, что *УУ удобно задавать в форме конечного автомата с памятью* (АП) того или иного типа. Разработаны и формальные методики преобразования алгоритмов, описывающих функционирование ЦУ, в описания конечных автоматов.

В сложных проектах возможно разделение ЦУ на несколько функционально слабо связанных пар ОБ-УУ на одном уровне иерархии или создание пары, иерархически погруженной в ОБ (реже в УУ).

Операционный блок обычно представлен набором регистров, логических схем (как правило, многофункциональных и управляемых), буферных схем и коммутируемых связей между ними. Важно лишь наличие на более низких иерархических уровнях описания проекта однозначной трактовки функционирования всех элементов ОБ.

Разработка общей структуры операционного блока. Основа этапа — выбор допустимых для данного уровня иерархии элементов, определение связей между ними и, если параметры элементов являются настраиваемыми, то и их настройка.

Описание работы управляющего автомата (УА). На этом этапе определяется функционирование УА, обеспечивающее требуемое взаимодействие элементов ОБ. Следует подчеркнуть, что два последних этапа сильно взаимосвязаны, и если не разрабатываются параллельно, то обычно выполняются итерационно.

Средства описания проекта

Применение САПР требует эффективных, наглядных, управляемых и контролируемых средств описания проекта. Описания проектируемого устрой-

ства и его частей на различных этапах проектирования имеют различные цели, поэтому современные САПР допускают описание объектов разными способами. Основная задача любого способа описания — передача разработчиком в САПР моделей, описывающих, что и как надо реализовать в процессе обработки и преобразований входного описания.

Ведутся интенсивные поиски *универсального способа* описания, пригодного для всех уровней проектирования и независимого от формы реализации проектов, однако пока рано говорить об успешности этих поисков. Для описания сложных систем, реализация которых явно потребует использования в том или ином виде процессорного ядра, наибольшее распространение получили методы задания спецификации проекта на языках, производных от C++. Фирмы, ведущие работы в этом направлении, включают в свои САПР средства моделирования, работающие на уровне *исполняемых спецификаций*. Подобный подход позволяет уже на самых ранних этапах проектирования оценить работоспособность принимаемых решений.

Основная проблема на этом пути — трудность создания программ, автоматизирующих перевод описания проекта с уровня исполняемых спецификаций на уровень, "понятный" синтезирующим компиляторам. Фирма Synopsys [XXVII] в 2003 г. анонсировала выпуск пакета CoCentric SystemC Compiler, помогающего и упрощающего перевод описаний с языка SystemC на синтезируемые языки уровня регистровых передач.

В настоящее время к наиболее распространенным универсальным способам описания, применимым для любого уровня иерархии проекта, относят *графический и текстовый*. Реже используются непосредственная разводка схем FPGA в редакторе топологии, описания в виде требуемых временных диаграмм и др. Каждый из способов описания проекта имеет свои достоинства и недостатки.

Графическое представление проекта

Графическое представление проекта создается в базисе допустимых для выбранного редактора САПР примитивов (для схемного редактора, например, в базисе элементов стандартной серии ТТЛ(Ш)). Главное достоинство графического способа — его традиционность и наглядность, связанные с привычностью разработчиков к восприятию изображений схем. Конечно, это преимущество проявляется только при правильном иерархическом и структурном разбиении проекта.

Варианты схемотехнического задания представляются интуитивно легко осваиваемыми и более наглядными. Смешанные (текстовые и графические) методы позволяют совместить достоинства обоих подходов. В последнее время схемные методы все чаще оставляются только для отчетной и обучающей документации и повсеместно заменяются текстовыми.

Графическое представление может использоваться как наглядное отражение текстового описания. Ряд САПР обеспечивает тесную связь текстового и графического описаний и дает возможность их взаимных трансформаций.

В отличие от текстовых, графические способы представления проекта обычно узко специализированы и требуют особых средств или приемов для переноса информации о проекте в другую среду. Создание единой базы данных для САПР различных фирм — вопрос отдаленного будущего. На сегодня возможные варианты решения этой проблемы состоят либо в применении специальных универсальных языков передачи информации о проекте (типа языка EDIF, Electronic Design Interchange Format), либо в задании проекта в текстовой форме.

Текстовое описание

Современные языки описания аппаратуры (HDL, Hardware Description Languages) допускают описание проектируемого устройства, как с точки зрения его *поведения*, так и с точки зрения его *структуры*. Эти возможности делают все более распространенным представление проекта в форме текстового описания алгоритмов функционирования его фрагментов в сочетании с текстовым же описанием межблочных соединений для сложных проектов. Достоинства текстового способа описания проекта заключаются в его компактности и относительной простоте автоматизации любых преобразований, включая начальную генерацию описания проекта. В проекты легко вносить изменения (добавлять или удалять объекты), объекты проектов легче модифицируются, можно создавать блоки с легко задаваемыми и изменяемыми параметрами. Очень важна возможность использования стандартных универсальных языков типа HDL, обеспечивающая простоту переноса проекта с одной аппаратной платформы на другую и переход от одной САПР к другой.

Языковое описание аппаратуры получает все большее распространение. Текстовые описания имеют две основные разновидности — *языки низкого уровня* (аналоги языков программирования типа ассемблера) и *высокого уровня*.

Языки низкого уровня

Языки низкого уровня ближе к аппаратным средствам, вследствие чего представляют для компиляторов потенциальные возможности создания проектов с более выигрышными параметрами. Платой за это является обычно жесткая ориентация на определенную аппаратуру и производящую ее фирму. Примерами таких языков могут служить языки PLDASM (фирма Intel), AHDL (фирма Altera) и ABEL (фирмы Xilinx). С помощью языков низкого уровня легче создавать проекты с наилучшими временными параметрами, т. к. в проектах будут учтены специфические особенности архитектуры CPLD или FPGA той или иной фирмы.

Сами эти языки развивались от описания комбинационных схем в форме логических выражений и их соединений с другими типовыми элементами программируемой логики (триггерами, элементами памяти и т. п.). Компиляторы с таких языков прежде всего должны были создать список соединений в базисе программируемой логики, для чего осуществляли перевод логических выражений, минимизацию и определение настраиваемых характеристик полученных схем и соединение компонент между собой. На заключительных этапах компиляции САПР решали задачу реального размещения и соединения компонент и формирование файла конфигурирования.

Языки высокого уровня

Языки высокого уровня менее связаны с аппаратными платформами и поэтому более универсальны. Среди них наиболее распространены языки VHDL и Verilog. Эти языки, как и другие алгоритмические языки высокого уровня, в принципе позволяют описать любой алгоритм в последовательной форме, т. е. через последовательность операторов присвоения и принятия решений. Основное их отличие в способности отражать также и параллельно исполняемые в аппаратуре действия, представляемые отдельными параллельно выполняемыми процессами с общим инициализирующим воздействием.

Средства описания автоматов

Автоматы играют важнейшую роль в проектировании современных систем. Формы и средства описания автомата разнообразны. *Современная тенденция состоит в переходе от прямой записи логических выражений, определяющих поведение автоматов, к предварительной графической форме задания.* Описание в виде *граф-схемы* переходов (диаграммы состояний) становится одним из самых распространенных вариантов исходного задания автоматов (в английской терминологии State Machins). Графические редакторы для создания автоматов включаются в состав средств ввода исходных проектов большинства современных САПР.

Хотя редакторы разных фирм имеют различный интерфейс, отличаются правилами записи условий и имеют другие особенности, для всех них характерны исключительная простота, естественность и "дружественность" интерфейса с пользователем, а также отсутствие жесткой необходимости знания выходного языка редактора. В большинстве случаев редакторы по выбору создают выходные тексты на языках VHDL или Verilog. Графические редакторы обладают полным набором средств для выполнения всей проектной процедуры разработки УА, позволяющих реализовать следующие операции:

- рисовать граф переходов, включая наименования состояний, направления, условия и приоритеты условий переходов, формируемые сигналы и способы их образования;

- проверять корректность составленного графа переходов (повторение имен, неоднозначность или некорректность перехода и т. д.);
- компилировать проект (формировать выходной текстовый файл) в выбранном языковом базисе;
- моделировать поведение автомата в интерактивном или компиляционном режиме (дополнительные возможности некоторых редакторов).

В последующих разделах этой главы приведены результаты работы двух пакетов: программы StateCAD Version 3.2 из пакета Workview Office фирмы Viewlogic (сейчас Innoveda [XIII]) и программы HDL Designer фирмы Mentor Graphics [XVII].

Достоинством программы StateCAD Version 3.2 является возможность широкого выбора форм представления результата (описания на языках высокого уровня VHDL и Verilog и на языках низкого уровня ABEL, AHDL).

Заметим, что *специфика продукции той или иной фирмы — производителя ПЛИС сказывается и на языках высокого уровня*, выражаясь прежде всего в отличиях в библиотеках, требуемых для работы, и в сложности и вариантности допустимых синтаксических конструкций для компиляторов. Конечные результаты компиляции одной и той же исходной граф-схемы автомата или последующей компиляции одной и той же программы с языка высокого уровня в загрузочный файл ПЛИС, полученные от компиляторов разных фирм, могут существенно различаться и иметь различную эффективность. Программа StateCAD Version 3.2 пакета Workview Office удобна тем, что перед трансляцией графа переходов нужно задать не только желательное языковое представление (VHDL, AHDL, Verilog, ABEL и т. д.), но и фирменные атрибуты, что позволяет оптимизировать запись автомата и избежать применения синтаксических конструкций, недопустимых для компиляторов соответствующих фирм.

Особенностью программы HDL Designer является разнообразие форм графического задания проектов, включающего структурное описание, потоковое описание, описание автоматов, таблиц истинности и в том числе редактора стандартных схемотехнических элементов. Выходной язык программы — VHDL или Verilog.

Как уже отмечалось, при использовании графических редакторов от пользователя не требуется обязательное владение выходным языком редактора. Однако в определенных случаях такое владение исключительно полезно. Знание языковых конструкций полезно, например, в ситуациях, когда автомат должен быть минимизирован по тем или иным параметрам, прежде всего по временным интервалам между формируемыми выходными сигналами, что может приводить к временным состязаниям сигналов. Именно в этих случаях владение языком и искусство проектировщика облегчают получение наилучших результатов.

§ 12.7. Маршрут проектирования ПЛИС и возможности типовых САПР

Этапы проектных процедур с использованием САПР

Разработка проекта обычно выполняется в следующем порядке.

1. **Ввод данных о проекте в САПР.** После составления проекта и его функциональной проверки информация о нем заносится в САПР. Занесение информации осуществляется с помощью редакторов, входящих в состав выбранной САПР.

   Для сложных иерархически организованных проектов помимо редакторов низшего уровня большую роль играют программы управляющих менеджеров проекта, редакторов проектной иерархии, блочные редакторы, символьные редакторы.

2. **Компиляция проекта.** После занесения информации о проекте или о его наиболее ответственных частях можно приступать к самому ответственному этапу проектирования — *компиляции*. Именно во время или сразу после компиляции выявляется большинство скрытых ошибок и нестыковок.

   Технически компиляция в САПР разбивается на ряд последовательных подэтапов: построение базы данных проекта, формирование списка соединений, проверка проектных правил и контроль соединений, логическая минимизация проекта, разбиение на блоки и их размещение, конкретизация физически реализуемых межсоединений, определение требуемых аппаратных ресурсов и в конце формирование загрузочного (конфигурационного) файла.

   На любом подэтапе могут возникать ошибки, требующие повторной компиляции после их коррекции. Результат компиляции — загрузочный файл, т. е. конфигурационная информация для выбранной микросхемы ПЛ. Помимо этого, обычно создается и файл отчета, содержащий всю информацию, как о процессе компиляции, так и о его результатах.

3. **Тестирование проекта.** Тестирование разработанного устройства, а в мало-мальски сложных проектах и отдельных его фрагментов — один из важнейших этапов проектирования, поскольку практически не бывает бездефектных проектов, созданных с чистого листа. Обнаружение дефектов проекта — сложнейшая задача. Скорость и тщательность тестирования во многом зависят от искусства разработчика.

   Для сложных проектов важным представляется функциональное моделирование наиболее существенных фрагментов или блоков проекта. *На этапе функционального моделирования* прежде всего осуществляется разра-

ботка требуемых тестовых примеров. Далее производится ввод этих примеров и выполняется собственно моделирование. Результаты моделирования позволяют решить: переходить ли к следующему этапу или вернуться к предыдущему.

Программы для тестирования (так называемые Test-Bench) могут быть построены на основе архитектурно-поведенческого тела, в котором проектируемый модуль представлен как структурный компонент, а генератор воздействия — в поведенческой форме.

В большинстве реальных ЦУ после подачи на них некоторых начальных данных выполняются несколько повторяющихся циклов. Необходима проверка работы устройства на нескольких наборах однотипных данных, поэтому можно рекомендовать следующую структуру программного модуля (процесса), представляющего тестовое воздействие: генерация сигналов начальной установки, затем реализация двух вложенных циклов, причем внутренний цикл последовательно формирует тестирующие сигналы для выполнения действий на одном наборе входных данных, а во внешнем производится их изменение.

В современных САПР широко распространено тестирование путем выполнения экспериментов над *программной моделью* спроектированной схемы. В отличие от функциональной модели, программная модель отражает все характерные свойства схемы. Моделирование с учетом конкретной реализации позволяет определить, достигнуты ли при проектировании требуемые или допустимые временные характеристики. Тестирование требует задания не только моделируемого устройства, но и моделей внешних сигналов. Задание этого взаимодействия выполняется с помощью редакторов временных диаграмм, которые делятся на *компилирующие и интерпретирующие.*

В многооконных САПР интерпретирующего типа результаты моделирования отображаются для текущего момента модельного времени во всех видах отображения проекта (сигналы в электрических схемах, в топологии). Поведение тестовых сигналов задается интерактивно, поэтому в этих редакторах легко изменить дальнейший ход эксперимента и состав отображаемых сигналов.

В компилирующих редакторах отображается ход всего эксперимента сразу. Любые изменения условий проведения теста требуют нового цикла компиляции эксперимента. Достоинством компилирующих систем моделирования является минимизация временных затрат.

4. **Определение временных характеристик разработанного устройства.** Современные САПР имеют внутри себя полную информацию о структуре проектируемого устройства и временных параметрах всех его компонентов, и это позволяет автоматизировать процесс вычисления разнообразных временных характеристик проекта.

В большинстве САПР предусмотрено автоматическое вычисление трех основных классов временных параметров:

- минимальных и максимальных задержек между источниками (входными сигналами) и приемниками (выходными сигналами), информация о которых выдается в виде матрицы задержек;
- максимально возможной производительности устройства (пропускной способности) в виде максимальной частоты тактирования элементов памяти, используемых в проекте;
- времен предустановки и выдержки сигналов, гарантирующих надежную работу схем при фиксации сигналов в синхронных элементах памяти.

Многие САПР позволяют также выделять критические пути передачи и преобразования информации для схемного или топологического представления проекта.

Хотя выполнение перечисленных вычислений не гарантирует обнаружения всех ошибок проектировщика, связанных с временными процессами в ЦУ, оно существенно уменьшает число таких ошибок или, как минимум, позволяет обнаружить в проекте места, опасные с точки зрения сбоев.

5. **Организация натурных экспериментов.** Последний этап проектирования — экспериментальная проверка спроектированного устройства. При всей тщательности выполнения предыдущих этапов всегда существует далеко не нулевая вероятность того, что в проекте имеются дефекты, которые могут проявиться на этапе внедрения или даже штатного использования устройства и повлечь за собою нежелательные последствия.

Выполнение натурных экспериментов существенно увеличивает вероятность выпуска бездефектной продукции. Средства ускорения работ на этом этапе и возможности их переноса на ранние этапы разработки, т. е. до завершения изготовления конечного продукта, были рассмотрены ранее — это прототипные системы и средства проведения экспериментов с ними. *Прототипные платы* широко использовались и ранее, в частности, при создании микропроцессорных систем. Аналогична и ситуация при разработке систем и устройств на основе программируемой логики. Широкий спектр прототипных плат, содержащих микросхемы программируемой логики и дополнительную аппаратуру (прежде всего микросхемы быстродействующих ОЗУ), выпускается различными зарубежными фирмами. Здесь можно указать средства фирм Altera (Demo Board); PLD Applications (платы PCI Bus Evaluation Board); Xilinx, Virtual Computer Corp., Video Software (платы HOT PCI Design Kit) и др.

Как отмечалось выше, определенная избыточность, заложенная разработчиком в БИС конечного продукта, позволяет использовать методы внутрикристальной отладки. В этих случаях в САПР вводятся такие функции,

выполнение которых способствует упрощению процедуры отладки готового проекта. Например, в САПР Quartus фирмы Altera предусматривается наличие всех трех необходимых составляющих такой процедуры:

- отладочных средств, помещаемых в отлаживаемую СБИС;
- информационно-транспортных средств, связывающих отлаживаемую СБИС с САПР инструментального компьютера;
- программных средств в составе САПР, которые управляют отладкой и отображают ее результаты.

Средства так называемого SignalTap Logic Analysis позволяют после загрузки модифицированной версии конфигурационного файла (проекта с отладочными фрагментами) в реальном масштабе времени регистрировать состояния не только на контактах ПЛИС, но и в ее произвольных внутренних точках. Занесение этой информации в память ПЛИС и передача сохраненной информации в компьютер с помощью интерфейса JTAG позволяют отображать эту информацию в стандартном редакторе временных диаграмм (Waveform Editor).

При успешном завершении экспериментальных работ конфигурационные файлы могут использоваться для изготовления требуемых программируемых БИС либо для записи в конфигурационные ПЗУ. Файлы отчетов о результатах компиляции обычно содержат информацию о конкретных данных по монтированию проекта в реальную БИС. Поэтому уже после этапа компиляции проектов возможен переход к разработке печатных плат, являющихся, как правило, конечной продукцией проектирования.

Итерационные возвраты к повторным процедурам компиляции в ходе конструкторско-технологического этапа проектирования возникают в том случае, если целесообразно изменить месторасположение входных и (или) выходных контактов БИС ПЛ, исходя из соображений как более эффективной разводки межсоединений БИС на печатной плате, так и их подключения к выходным разъемам платы. Подобная возможность следует из способности современных БИС ПЛ обеспечивать различные варианты монтирования одного и того же проекта в одну и ту же БИС.

§ 12.8. Основные сведения о языке VHDL

Остановимся на некоторых вопросах, относящихся к наиболее известному языку проектирования аппаратных средств — VHDL, который использован далее при рассмотрении примера проектирования цифрового устройства средствами САПР.

Язык VHDL появился в начале 80-х гг. по запросам организаций Министерства обороны США. Первая его версия, предназначенная в основном для унификации описаний проектов в различных ведомствах, была принята в 1985 г. В 1987 г. язык VHDL был принят Международным институтом IEEE (Institute of Electrical and Electronic Engineers) как стандарт VHDL-87. Он использовался, главным образом, для описания (спецификации) уже спроектированных систем. Использование для задач синтеза устройств (работа с компиляторами) началось с 1991 г. В 1993 г. IEEE принимает новый расширенный стандарт VHDL-93.

Язык может быть использован для проектирования ЦУ разных иерархических уровней — от вентильного до системы в целом. В 1999 г. был утвержден стандарт языка IEEE Std 1076.1-1999, известного под названием VHDL-AMS. Это расширение языка VHDL ориентировано на описание аналоговых и смешанных (аналого-цифровых) устройств. В настоящее время язык VHDL является, видимо, самым популярным среди проектировщиков цифровой аппаратуры. Сравнимым по популярности является язык Verilog, и практически любая современная САПР средств ВТ или цифровых устройств имеет в своем составе компиляторы с этими языками (как входными, так и выходными).

Язык VHDL является проблемно-ориентированным, его основные прикладные аспекты связаны с использованием в качестве рабочего инструмента для задач описания структуры и (или) поведения широкого класса цифровых устройств. Описания могут использоваться для синтеза и (или) моделирования таких систем. В соответствии с назначением, язык приспособлен для описания систем как с точки зрения их структурной организации (из модулей с известным поведением), так и с точки зрения поведения либо системы в целом, либо всех ее составных частей. Наибольшие ограничения на набор допустимых (относительно стандарта) операторов языка имеют компиляторы для синтеза устройств, значительно меньше ограничений существует у систем моделирования.

Синтаксические конструкции и основные понятия языка

Синтаксические конструкции языка содержат две составляющие — общеалгоритмическую (свойственную большинству обычных алгоритмических языков) и проблемно-ориентированную.

Общеалгоритмическая составляющая языка достаточно традиционна и содержит как традиционные операторы действия (присваивания (:=), условия (IF), выбора (CASE), цикла (LOOP), вызова процедуры), так и традиционные типы данных: числовые, логические, символьные, перечислительные и аг-

регатированные (массивы, записи и файлы). Не самым распространенным можно считать лишь набор ключевых слов и синтаксических правил составления предложений.

В программах на языке VHDL используются следующие термины и понятия. Все проекты выражаются в терминах объектов проекта — Entity. Каждый объект проекта имеет объявление интерфейса объекта — Entity Declaration и описание архитектурного тела объекта — Architecture Body. Entity содержит имя объекта и его интерфейс (входы и выходы). Architecture Body содержит описание структуры или поведения объекта. Верхний уровень проекта описывается через объекты верхнего уровня; если устройство иерархично, то описания объектов верхнего уровня содержат в себе обращения к компонентам более низкого уровня, которые описываются как самостоятельные объекты нижнего уровня. В свою очередь, объекты нижнего уровня могут связываться с объектами еще более низкого уровня. Для определенности функционирования системы независимо от числа уровней иерархии все объекты самых нижних уровней иерархии должны иметь описание, определяющее их функционирование.

Описание каждого VHDL-проекта содержит, по меньшей мере, одну пару Entity и Architecture Body. Один и тот же объект может иметь несколько архитектурных тел (естественно, что при моделировании поведения системы или при ее синтезе специальная проектная единица "объявление конфигурации" (Configuration Declaration) определяет единственный вариант описания). Еще одна пара проектных единиц — объявление пакета (Package) и тело пакета (Package Body) служит для хранения глобальных данных, функций, подпрограмм и т. д.

Описание проекта на языке VHDL

Описание проекта на языке VHDL имеет типовую структуру: в его начале указываются библиотеки функциональных элементов, которыми может пользоваться САПР (Library Declaration), далее (если необходимо) перечисляются подключаемые пакеты, и, наконец, следуют описания объектов (Entity), которые будут использованы как компоненты проектируемого устройства.

Многие другие термины и понятия здесь и далее не затрагиваются, поскольку цель более или менее серьезного изучения языка может ставиться лишь в работах достаточно большого объема. Из отечественной литературы можно рекомендовать [8, 16, 43], а из зарубежных изданий [51, 64]. Изложение приведенного далее материала продиктовано желанием обеспечить понимание читателями, не знакомыми со спецификой языков описания аппаратуры, примеров проектирования, рассмотренных в последующих параграфах.

Проблемно-ориентированными и поэтому *наиболее важными средствами и понятиями языка VHDL являются*:

- средства описания структуры отдельных объектов проекта;
- средства задания и описания параллелизма для выполняемых действий;
- средства реализации обмена данными между параллельно работающими фрагментами с сохранением причинно-следственных связей их временного взаимодействия.

Раздел архитектуры объекта (Architecture Body) может быть представлен в структурном, поведенческом или смешанном вариантах. Для описания структуры необходимо задать интерфейс компонентов используемого типа (уточнение их поведения или дальнейшая детализация их структуры даются в разделе архитектуры следующего иерархического уровня) и описание соединений конкретных компонент между собой. Интерфейс структурной компоненты объектов, включающий перечисление имен контактов с указанием допустимого для этого контакта типа информации, задается специальной синтаксической конструкцией (COMPONENT ... PORT), конкретизация компонент объекта и схема их взаимного соединения между собой обеспечиваются конструкцией (PORT MAP). В качестве имен цепей, подключенных к входам и выходам объекта, используются имена контактов, описанных в интерфейсной части объекта (ENTITY). Эти имена для поведенческих фрагментов архитектурного тела соответствуют именам сигналов, а для структурных компонент — именам соединительных цепей. Аналогично, для задания имен внутренних цепей в языке используются имена сигналов, объявленных в проекте. Указанная дуальность имен цепей и сигналов позволяет легко создавать в рамках одного архитектурного раздела смешанные структурно-поведенческие описания. Использование конструкций (GENERIC) и (GENERIC MAP) очень важно для создания универсальных и настраиваемых под конкретное применение компонент.

Создание компактного описания фрагментов структуры с однотипными компонентами и однотипной схемой подключения облегчается при употреблении конструкций (FOR ... GENERATE) и (IF ... GENERATE). Поскольку в одном проекте допустимо существование различных вариантов описания объектов с одним и тем же интерфейсом, синтаксическая конструкция (FOR ... USE) поможет конкретизировать конфигурацию.

Понятие сигнала (SIGNAL) является одним из ключевых в языке VHDL. Кроме уже упоминавшегося использования в качестве имени цепи, по которой сигнал может передаваться, или имени контакта, на который поступает одноименный сигнал, это понятие отражает основные свойства реальных входных и выходных данных проекта. Среди различных свойств сигналов важнейшим представляется их временное поведение (наличие прошлого, настоящего и будущего состояний). В языке это находит отражение, в част-

ности, в наличии у сигналов атрибутов (attribute), а в операторах присваивания им значения (<=) даже возможности задания временной диаграммы (waveform). Выполнение оператора присваивания для сигналов отличается от поведения простых переменных после выполнения похожего оператора *присваивания значения переменной* (:=). Для переменной ее значение устанавливается сразу же после выполнения такого оператора. А для сигналов (во избежание неправильной трактовки) в моделирующих программах на языке VHDL изменение значения сигнала, обусловленное выполнением текущего оператора программы, сначала должно быть запомнено (как следующее значение), а замена текущего значения должна быть выполнена в указанный момент модельного времени и одновременно для всех сигналов проекта.

Важнейшее свойство языка VHDL — отражение параллелизма выполнения действий в реальных схемах. Следует различать синтаксические конструкции, создающие механизмы, используемые в моделирующих программах, и конструкции, обеспечивающие при синтезе устройства сохранение необходимых причинно-следственных связей между взаимодействующими блоками. Параллелизм, прежде всего, потребовал введения понятия параллельных операторов. К важнейшим из них (кроме упоминавшегося выше параллельного оператора присвоения значения сигналу) относятся операторы процесса (**PROCESS**) и охраняемого блока (**BLOCK**). Важнейшим свойством параллельных операторов является то, что их инициализация осуществляется либо в результате достижения заданного модельного времени, либо вследствие выполнения заданного события. Взаимодействие параллельных операторов между собой осуществляется только при помощи сигналов. Помимо введения синтаксических конструкций, соответствующих параллельным операторам, моделирующие программы для обеспечения правильной работы таких операторов вынуждены заводить механизмы, планирующие и управляющие порядком выполнения операторов. Эти механизмы опираются в своей работе на список-календарь планируемых событий.

К параллельным относятся операторы селективного (**WITH...SELECT...<=...WHEN**) и условного присваивания (**...<=...WHEN...ELSE...WHEN**), которые обеспечивают компактную форму задания временного поведения сигнала. Для управления параллелизмом естественно введение операторов, задающих моменты запуска (абсолютных — **WAIT** ..., относительных — **AFTER**), и операторов, задерживающих момент запуска (**WAIT ... UNTIL, WAIT ... FOR**). Ряд традиционно последовательных операторов, таких, например, как вызов процедуры, имеет аналогичный вариант со свойствами параллельных операторов.

Возможность различными способами описать поведение одной и той же системы или объекта, оставаясь в рамках одного архитектурного тела, приводит к понятию *стиля описания* (программирования).

Можно выделить следующие типы стиля:

- *последовательный*, когда преобразование потока входных данных в поток выходных данных осуществляется с использованием только последовательных операторов;
- *параллельный*, когда описание поведения задано в виде параллельно выполняемых процессов;
- *потоковый*, когда описание задано в виде последовательности параллельных операторов языка.

К специфическому стилю можно отнести автоматный способ описания, когда функционирование задано в форме описания конечного автомата того или иного типа.

Примеры поведенческих описаний элементов на языке VHDL

Проиллюстрируем поведенческие варианты описаний на простейших примерах. Пусть требуется описать на языке VHDL логический элемент, реализующий функцию Z = (a \/ b)c под наименованием input3_orand1. Начиная с раздела Entity Declaration, описание может иметь следующий вид (листинг 12.1). Здесь и далее в листингах и тексте полужирным шрифтом выделены ключевые (зарезервированные) слова языков VHDL и AHDL.

```
ENTITY input3_orand1 IS -- entity declaration
 PORT (a,b,c: IN BIT; -- port statement
       Z: OUT BIT);
END input3_orand1;
ARCHITECTURE one OF input3_orand1 IS
-- architecture "one" of entity input3_orand1
 BEGIN
 orand3: PROCESS
       BEGIN
             IF (c ='1') THEN z<= a OR b
             ELSE z<='0';
             END IF;
             WAIT ON a,b,c;
       END PROCESS;
END;
```

Описание архитектуры восьмиразрядного счетчика с тактируемым входом сброса можно задать следующим образом:

```
Synch_count: PROCESS
BEGIN       WAIT UNTIL clock='1';
      IF (reset='1') THEN count<="00000000";
      ELSE count<=count+'1';
      END IF;
END PROCESS;
```

Язык VHDL для моделирования и синтеза

Помимо стандарта языка VHDL, определенного для моделирования, в 1999 г. был принят стандарт IEEE Std 1076.6, ориентированный на синтез — IEEE Standard for VHDL Register Transfer Level (RTL) Synthesis. Фирмы — разработчики компиляторов с языка VHDL (несмотря на близость решений) трактуют синтаксис языка со значительными расхождениями. Совершенно естественно желание проектировщиков использовать одно и то же языковое описание сначала для моделирования проектируемой системы, а затем и для ее синтеза. Однако подобный подход (при расхождениях в трактовке стандарта) должен использоваться с осторожностью.

В современных системах моделирования удается реализовать почти полный стандарт языка, значительно меньшие возможности допускают синтезирующие средства. Вопрос не только в допустимых для использования в том и другом вариантах синтаксических конструкциях, но и в совпадении свойств модели и синтезированного устройства. Существующие сегодня САПР позволяют автоматизировать преобразование проектов, описанных на уровне регистровых передач (RTL — Register Transfer Level), в описание на уровне вентильных межсоединений. Конкретные реализации ячеек вентильного уровня, включая их макроблоки, содержатся в технологических библиотеках целевой ИС проекта.

Многие синтаксические конструкции языка ориентированы только на моделирование и игнорируются при компиляции. К игнорируемым относится группа конструкций, связанная с понятием модельного времени, например, выражения для установления временных задержек (**AFTER**) в операторах сигнального присваивания и в операторах временного управления (**WAIT...FOR**). Другой игнорируемой группой являются операторы проверки (**ASSERT**), обеспечивающие вывод на консоль информации о специфической ситуации при моделировании.

Результаты моделирования могут отличаться от результатов, получаемых при работе синтезированного по этому же описанию устройства. Различие, пре-

жде всего, обусловлено замещением одиночной, но достаточно емкой синтаксической конструкции проекта на соединение ряда элементов используемой технологической библиотеки. Наличие у этих элементов определенных временных характеристик и многовариантность подобного замещения может приводить не только к расхождению по времени, но и появлению дополнительных (ложных) изменений выходных и промежуточных сигналов.

Различия поведения модели и синтезированного устройства могут быть связаны и с определенной трактовкой компилятором некоторых синтаксических конструкций языка. Например, при моделировании фрагмента схемы, описываемого процессом, его инициализация определяется составом списка чувствительности процесса, а при синтезе в качестве входных воздействий синтезируемого фрагмента выбираются все сигналы, определяющие поведение процесса. Все обращения к функциям и подпрограммам замещаются на встроенные экземпляры схемной реализации тел этих операторов.

Структурный и поведенческий варианты описания проекта

Когда следует использовать структурный, а когда поведенческий вариант описания проекта?

Структурный вариант предусматривает перечисление как типов компонент и их интерфейса (их выводов), так и связей всех компонент между собой, тем самым непосредственно отражая задаваемую для реализации схему, которая и будет создана в выбранной СБИС ПЛ.

Поведенческий вариант определяет функции, которые должны быть реализованы, но не говорит о том, каким именно способом это должно быть сделано. Так как схемотехнические реализации узлов и устройств практически всегда многовариантны, САПР получает для их реализации определенную свободу действий. Большое достоинство поведенческого варианта — его компактность и наглядное представление функционирования устройства, что хорошо видно хотя бы из приведенных примеров. Платой за это является ослабление контроля за способом реализации проектируемого устройства или его фрагмента, поскольку это отдается "на усмотрение" компилятора. При этом следует ожидать, что компилятор может принять реализацию схемы, которая не будет столь же эффективна по быстродействию и затратам ресурсов, как выполненная квалифицированным специалистом, хорошо знающим ресурсы и особенности структурной организации выбранной ПЛИС. Поэтому чаще всего комбинируют использование структурных и поведенческих описаний в рамках одного и того же проекта. Для критичных по скорости фрагментов целесообразно использовать структурные описания. Например, задавая функции счета или суммирования, реализуют структуры с параллельными переносами, обеспечивающими наибольшее быстродейст-

вие, тогда как компилятор (при неустановленных специальных опциях), возможно, создаст более простые структуры с последовательными переносами. В то же время остальные части проекта целесообразно описать в поведенческом варианте, что существенно упрощает задачу.

О возможностях и средствах описания типовых узлов цифровой техники

Весьма существенным вопросом, интересующим разработчика, является наличие в языке возможностей и (или) средств для описания типовых узлов и устройств цифровой техники. В соответствии с традиционным разбиением, необходимо оценить возможности описания следующих типов узлов: *комбинационных схем, регистровых схем и цифровых автоматов.*

Функционирование комбинационных схем удобно описывать достаточно широким классом средств: арифметическими и логическими выражениями, условным или селективным (по выбору) назначением сигналу. В разделе операторов процесса допустимо использование операторов условия (**IF**) и выбора (**CASE**). Входы и выходы схемы могут быть представлены в виде сигнала или переменной. Как правило, используются следующие типы данных: `bit`, `bit_vector`, `std_logic`, `std_logic_vector`. Использование других типов данных может потребовать определения функций взаимных преобразований. Пример образования схемы под именем input3_orand1, приведенный ранее (см. листинг 12.1), как раз и соответствует представлению комбинационной схемы в языке VHDL.

Поведение регистровых схем удобно описывать, используя либо процессное, либо блочное представление. При процессном представлении внутри оператора **PROCESS** обычно пользуются операторами условия (**IF**) и выбора (**CASE**). При блочном представлении возможно использование операторов условного назначения сигналу (**<= ... WHEN**). В качестве примера рассмотрим описание поведения триггера D типа "защелка", имеющего вход асинхронного сброса.

```
ENTITY d_ff IS -- entity declaration
PORT (d,c,r: IN BIT; -- port statement
      q:INOUT BIT);
END d_ff;

ARCHITECTURE one OF d_ff IS
-- architecture "one" of entity d_ff
 BEGIN
 beh_tr: BLOCK (c='1' OR r='1');
```

```
    BEGIN
        q<=GUARDED '0' WHEN r='1'
        ELSE d WHEN c='1'
        ELSE q;
    END BLOCK beh_tr;
END one;
```

При описании поведения регистровых схем необходимо уделять внимание наличию или отсутствию синхронизации входных сигналов и ее типу (асинхронное, потенциальное или динамическое управление) и способам задания входных, выходных сигналов и внутренних состояний при описании многоразрядных схем (счетчиков, регистров сдвига и т. д.). Для определения этих переменных или сигналов в многоразрядных схемах можно использовать понятие вектора или ограниченного целого (с диапазоном изменения, связанным с разрядностью схемы). При описании многоразрядных схем таких как счетчики, регистры сдвига и т. д. (последовательностных схем), обычно используется описание в форме двух процессов. Один определяет поведение регистровой части схемы — запись в регистр, сброс и (или) предустановку регистра. Другой процесс задает поведение комбинационной части схемы (сложение, вычитание, сдвиг и т. д.). Подобный подход к описанию последовательностных схем позволяет создавать конструкции, более эффективные с точки зрения синтеза средствами САПР.

Следующим типовым фрагментом, играющим очень важную роль в проектировании, являются цифровые автоматы. Выделение автоматов в самостоятельный класс типовых фрагментов связано не только с удобством разбиения (для разработчика) проектируемых устройств на операционные блоки и управляющий автомат (как описано ранее), но и со специфическим подходом компиляторов к оптимизации синтезируемой структуры. Из двух видов автоматов (асинхронных и синхронных) наибольшее распространение получили последние. Хотя этот тип автоматов проигрывает в быстродействии, его использование в проектировании значительно проще.

Каноническое разбиение поведения автоматов сводится к двум разновидностям (автоматам Мили и Мура). Внешне структурное отличие между ними (рис. 12.8, *а* и 12.8, *б*) сводится к отсутствию у автоматов Мили (рис. 12.8, *б*) подключения входных сигналов (X на рисунке) к блоку комбинационной логики, определяющей значения выходных сигналов (Y). Базовым элементом автоматов является *регистр состояния* (не путать с термином "регистр состояния" для периферийных БИС, микропроцессоров и т. п.). Выходные сигналы (S) триггеров этого регистра определяют как следующее состояние автомата (устанавливаемое по фронту тактового сигнала Clk), так и значения выходных сигналов.

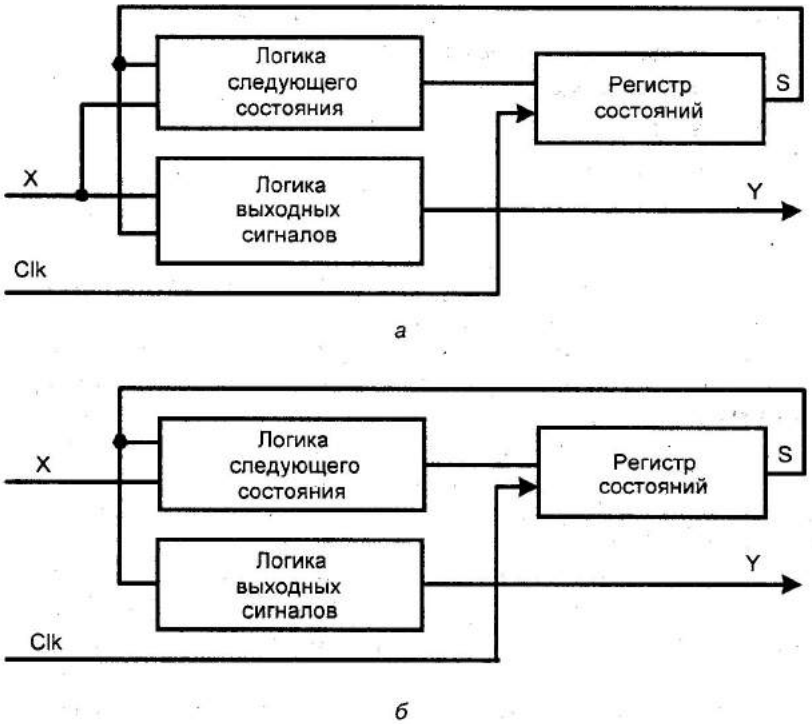


Рис. 12.8. Функциональная схема автоматов Мура (*а*) и Мили (*б*)

В условиях автоматического синтеза автомата на вентильном уровне достаточно часто могут появляться проблемы с использованием выходных сигналов. В моменты смены состояний автомата или изменения значений входных сигналов могут возникать паразитные импульсы, обусловленные рисками, появляющимися в комбинационной логике выходных сигналов. Поэтому выходные сигналы автомата опасно использовать в качестве тактовых сигналов или сигналов асинхронных сбросов и установок триггеров. Для устранения указанного эффекта используются автоматы с синхронными выходами (рис. 12.9). На рисунке Y_C — комбинационные сигналы, а Y_R — выходные сигналы, синхронизируемые тактовым сигналом. При подключении триггеров к выходам автоматов необходимо учитывать, что выходные сигналы триггеров начинают как бы "запаздывать" на один такт относительно сигналов входной комбинационной логики. Другими словами, логика управления выходными триггерами автомата должна формировать значения "следующего" (или будущего) состояния сигнала.

Другая проблема — возможность появления *временных перекосов* в блоках комбинационной логики автомата. Это может происходить вследствие асин-

хронности входных сигналов относительно тактовой частоты автомата. Когда изменение входного сигнала происходит в момент времени, близкий к фронту тактового сигнала, в логике, управляющей следующим состоянием, появляется вероятность перевода автомата либо в неправильное состояние, либо в неопределенное. В этой ситуации и в логике формирования выходных сигналов могут возникать ложные импульсы. И то, и другое чаще всего приводит к катастрофическим последствиям для работы автомата. Модификация схемы автомата, устраняющая указанный эффект, приведена на рис. 12.10 (на рисунке: X_C — синхронные входные сигналы, а X_A — асинхронные).

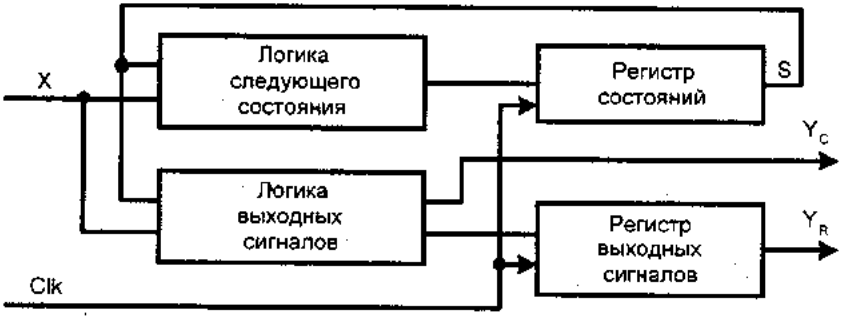


Рис. 12.9. Функциональная схема автомата с синхронными выходами

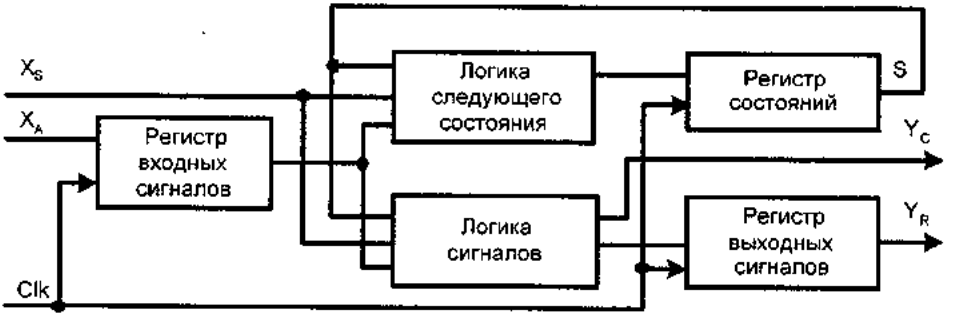


Рис. 12.10. Функциональная схема автомата с синхронизацией входов

Классическим описанием автоматов является табличное задание переходов и выходных сигналов. В клетках таблицы переходов записаны состояния, в которые переходит автомат из исходного состояния при соответствующих условиях, а в клетках таблицы выходов — выходные сигналы при тех или иных условиях. Достоинство такого способа определения автоматов — лег-

кость контроля над пропуском неопределенных переходов или условий формирования выходных сигналов.

Наглядно отражает алгоритмы управления и поэтому получает все большее распространение задание автомата *в форме графа переходов*. Для упрощения их создания в состав все большего числа САПР вводят графические редакторы диаграмм состояний. Несмотря на некоторые различия в формальном способе конкретизации, все такие редакторы позволяют задать желательные структуру автомата и способ формирования выходных сигналов.

В качестве примера описания рассмотрим автомат, диаграмма состояний которого приведена на рис. 12.11. В автомате под именем состояния перечисляются имена сигналов, активизируемых в этом (или этим) состоянии. Например, сигналы En и Ld в состоянии Write. Если из состояния возможен переход в несколько других состояний, то на дугах переходов указываются логические выражения, составленные из имен входных сигналов, определяющих условия переходов. Если из текущего состояния переходы являются условными, то обычно предполагается, что альтернативой является остановка в текущем состоянии. В примере это сигналы Ack, Rd и Wr. В кружочках на дугах (или рядом) указывается приоритет переходов при одновременности выполнения условий для нескольких переходов. Кроме указания выражения, определяющего переход, на дуге может содержаться информация о сигналах, возникновение которых обусловлено выполнением условия этого выражения. Такие сигналы в примере — Busy. В примере два выходных сигнала BusyC и BusyR. Первый из них соответствует комбинационному типу, а второй — синхронному. Поэтому комбинационный сигнал повторяет на выходе автомата во времени возникновение и исчезновение условия с задержкой, соответствующей задержке комбинационной схемы. А синхронизированный сигнал выставляется и снимается на выходе по условию его формирования в моменты тактирования. Поэтому длительность первого сигнала совпадает с длительностью входного сигнала, а длительность второго оказывается зависящей от условий его формирования и кратной периоду тактирования.

Для правильной совместной работы автоматов и триггерных схем, имеющих схемы пропускания входных сигналов (управляемые сигналом разрешения) и тактирующие входы (синхронные с управляющим автоматом) оказывается важным иметь ненулевым *время предустановки* (опережение сигналом разрешения сигнала тактирования). В примере такой парой являются сигналы En и Ld. Для обеспечения необходимого опережения сигнал En присутствует и на дуге перехода (комбинационный сигнал, возникающий почти сразу после появления Wr на входе автомата), и в списке сигналов, инициируемых состоянием, т. е. синхронизируемых тактовыми импульсами. Объединение сигналов по логике ИЛИ дает сигнал En, обладающий необходимыми свойствами предустановки.

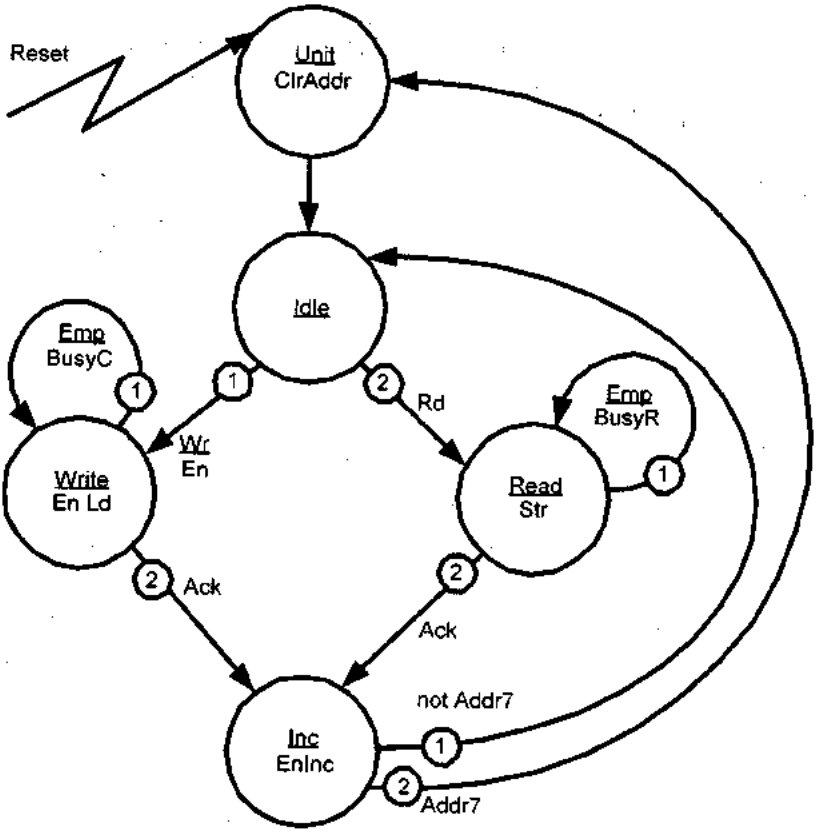


Рис. 12.11. Пример задания автомата в форме диаграммы переходов

Возможность использования в языке перечислительного типа данных позволяет ввести перечислительные типы данных "состояния", а при желании разработчика и типы данных "входы" и "выходы". Предпочтительна процессная форма описания поведения автомата. В зависимости от желания разработчика и типа автомата, его архитектура может включать от одного до пяти процессов. Оператор варианта целесообразно использовать в качестве основного каркаса процессов, для чего каждому состоянию автомата назначается вариант в операторе выбора. Для описания альтернативных вариантов формирования переходов и выходных сигналов (если это необходимо) обычно применяется условный оператор. Приоритетность переходов соответствует порядку записи условий. Отдельный процесс может вводиться для описания процедуры тактирования и начальной установки автомата.

Описывающая автомат программа на языке VHDL может иметь вид, приведенный в листинге 12.2. Для создания текста программы использовался гра-

фический редактор описания конечных автоматов (в САПР этот тип редакторов обычно носит название editor of FSM — Finite State Machine). В данном случае это была программа HDL Designer (версии 2002.1b) фирмы Mentor Graphics [XVII].

Листинг 12.2

```
-- Generated by Mentor Graphics' HDL Designer(TM) 2002.1b (Build 7)
--
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;

ENTITY example IS
   PORT(
      Ack      : IN     std_logic;
      Addr7    : IN     std_logic;
      Clk      : IN     std_logic;
      Emp      : IN     std_logic;
      Rd       : IN     std_logic;
      Reset    : IN     std_logic;
      WR       : IN     std_logic;
      BusyC    : OUT    std_logic;
      BusyR    : OUT    std_logic;
      ClrAddr  : OUT    std_logic;
      En       : OUT    std_logic;
      EnInc    : OUT    std_logic;
      Ld       : OUT    std_logic;
      Str      : OUT    std_logic;
      StrS     : OUT    std_logic                    -- вставка 1
   );
-- Declarations
END example ;
-- hds interface_end
--
-- VHDL Architecture exam.fsm
-- Generated by Mentor Graphics' HDL Designer(TM) 2002.1b (Build 7)
--
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;

ARCHITECTURE fsm OF example IS
   -- Architecture Declarations
   TYPE STATE_TYPE IS (
      Unit,
      Idle,
      Write,
      Read,
      Inc
   );
   -- State vector declaration
   ATTRIBUTE state_vector : string;
   ATTRIBUTE state_vector OF fsm : ARCHITECTURE IS "current_state" ;
   -- Declare current and next state signals
   SIGNAL current_state : STATE_TYPE ;
   SIGNAL next_state : STATE_TYPE ;
   -- Declare any pre-registered internal signals
   SIGNAL BusyR_cld : std_logic  ;
   SIGNAL StrS_int : std_logic  ;          -- вставка 2
BEGIN  ----------------------------------------------------------------------
-

   clocked : PROCESS(
      Clk,      Reset
   )
   ----------------------------------------------------------------------
   BEGIN
      IF (Reset = '1') THEN
         current_state <= Unit;
         -- Reset Values
         BusyR_cld <= '0';
         StrS <= '0';                -- вставка 3
      ELSIF (Clk'EVENT AND Clk = '1') THEN
         current_state <= next_state;
         -- Registered output assignments
         StrS <= StrS_int;                  -- вставка 4
```

```
      -- Default Assignment To Internals
      BusyR_cld <= '0';
      -- Combined Actions for internal signals only
      CASE current_state IS
      WHEN Read =>
         IF (Emp = '1') THEN
            BusyR_cld <= '1' ;
         END IF;
      WHEN OTHERS =>
         NULL;
      END CASE;
   END IF;
END PROCESS clocked;
-------------------------------------------------------------------
nextstate : PROCESS (
   Ack,      Addr7,      Emp,      Rd,      WR,      current_state
)
-------------------------------------------------------------------
BEGIN
   CASE current_state IS
   WHEN Unit =>
         next_state <= Idle;
   WHEN Idle =>
      IF (WR = '1') THEN
         next_state <= Write;
      ELSIF (Rd = '1') THEN
         next_state <= Read;
      ELSE
         next_state <= Idle;
      END IF;
   WHEN Write =>
      IF (Emp = '1') THEN
         next_state <= Write;
      ELSIF (Ack = '1') THEN
         next_state <= Inc;
      ELSE
         next_state <= Write;
      END IF;
```

```
    WHEN Read =>
       IF (Emp = '1') THEN
          next_state <= Read;
       ELSIF (Ack = '1') THEN
          next_state <= Inc;
       ELSE
          next_state <= Read;
       END IF;
    WHEN Inc =>
       IF (Addr7 /= '1') THEN
          next_state <= Idle;
       ELSIF (Addr7 = '1') THEN
          next_state <= Unit;
       ELSE
          next_state <= Inc;
       END IF;
    WHEN OTHERS =>
       next_state <= Unit;
    END CASE;
END PROCESS nextstate;
-------------------------------------------------------------------
output : PROCESS (
    Emp,        WR,        current_state
)
-------------------------------------------------------------------
BEGIN
    -- Default Assignment
    BusyC <= '0';
    ClrAddr <= '0';
    En <= '0';
    EnInc <= '0';
    Ld <= '0';
    Str <= '0';
    -- Default Assignment To Internals
    StrS_int <= '0';                    -- вставка 5
    -- Combined Actions
    CASE current_state IS
    WHEN Unit =>
       ClrAddr <= '1' ;
```

```
      WHEN Idle =>
         IF (WR = '1') THEN
            En <= '1' ;
         END IF;
         IF ((Rd = '1') AND (WR = '0')) THEN        -- вставка 6
            StrS_int <= '1' ;
         END IF;
      WHEN Write =>
         En <= '1' ;
         Ld <= '1' ;
         IF (Emp = '1') THEN
            BusyC <= '1' ;
         END IF;
      WHEN Read =>
         Str <= '1' ;
         IF (Ack = '0') THEN                    -- вставка 7
            StrS_int <= '1' ;
         END IF;
      WHEN Inc =>
         EnInc <= '1' ;
      WHEN OTHERS =>
         NULL;
      END CASE;
   END PROCESS output;
   -- Concurrent Statements
   -- Clocked output assignments
   BusyR <= BusyR_cld;
END fsm;
```

В основе работы автомата лежит использование двух сигналов — current_state и next_state. Первый соответствует выходным сигналам регистра состояния (см. рис. 12.9), а второй — выходным сигналам блока комбинационной логики следующего состояния. Сигналы относятся к объявленному перечислительному типу state_type и заданы списком имен состояний. Объявлен сигнал внутренних цепей, поступающих на вход регистра выходных сигналов. Это сигнал BusyR_cld. Архитектурное тело автомата содержит четыре параллельных оператора: три оператора процесса и один условного присваивания сигнала.

Запуск процесса "clocked" определяется изменением двух сигналов Reset и Clk. Сигнал сброса устанавливает в исходное состояние все триггеры автомата. Тактовый сигнал обеспечивает перепись в триггеры регистров состояний выходов блоков комбинационной логики (следующего состояния — current_state, и выходных сигналов — BusyR_cld). Важно обратить внимание на то, что изменения состояний происходят в момент появления нарастающего фронта сигнала Clk, т. к. запускающее событие определено как "появление единицы и наличие переходного процесса на входе Clk".

Синтаксическая конструкция Clk'event называется *атрибутом сигнала.* Атрибут сигнала может принимать значение "истинно" или "ложно" и характеризует некоторые свойства сигнала на момент моделирования (в данном контексте — переходный режим).

Использование выражения "Clk'event" соответствует реальной структуре устройства, реализующего автомат, в котором состояние отображается состоянием регистра. Так как этот регистр является одновременно датчиком информации о текущем состоянии и приемником нового значения, во избежание гонок необходимо использовать регистры с динамическим управлением, реагирующие на изменение сигнала, что и задается используемой конструкцией условия.

Следующий процесс, имеющий имя "nextstate", описывает требуемое поведение комбинационной логики следующего состояния для всех переходов автомата. Приоритет выполнения условных переходов задается последовательностью записи условий. Список чувствительности процесса содержит перечисление сигналов, определяющих работу логики, включая текущее состояние автомата — current_state.

Процесс с именем "output" описывает поведение блока комбинационной логики выходных сигналов в зависимости от текущего состояния автомата. Как и в предыдущем описании, список чувствительности содержит перечисление всех входных сигналов логики. В начале описания процесса имеется группа операторов присваивания значений сигнала, которая соответствует значениям сигналов по умолчанию. Тем самым обеспечивается более компактная форма записи формируемых сигналов.

Последняя группа параллельных операторов присваивания значений сигналам (в нашем примере требуется только один — BusyR) соответствует описанию формирования сигналов на выходе регистра выходных сигналов.

Полученная программа была проверена средствами моделирующего пакета ModelSim SE 5.7b фирмы Model Technology (подразделение фирмы Mentor Graphics). Поведение автомата соответствовало ожидаемому. Далее этот же текст был откомпилирован в САПР MAX+Plus II фирмы Altera [III]. При компиляции в качестве планируемого семейства ИС было выбрано семейство ACEX 1K. При синтезе автомата компилятор использовал 20 логических ячеек (из них в пяти использовались триггеры DFF). Результаты моделирования по-

ведения синтезированного автомата приведены на рис. 12.12. На диаграмме можно увидеть, что поведение выходных сигналов в целом совпало с запланированным (появилось запаздывание выходных сигналов, обусловленное задержками элементов, образующих схему). Однако видно и наличие коротких паразитных импульсов у ряда сигналов. Такое поведение сигналов может служить причиной некорректной работы управляемых ими схем.

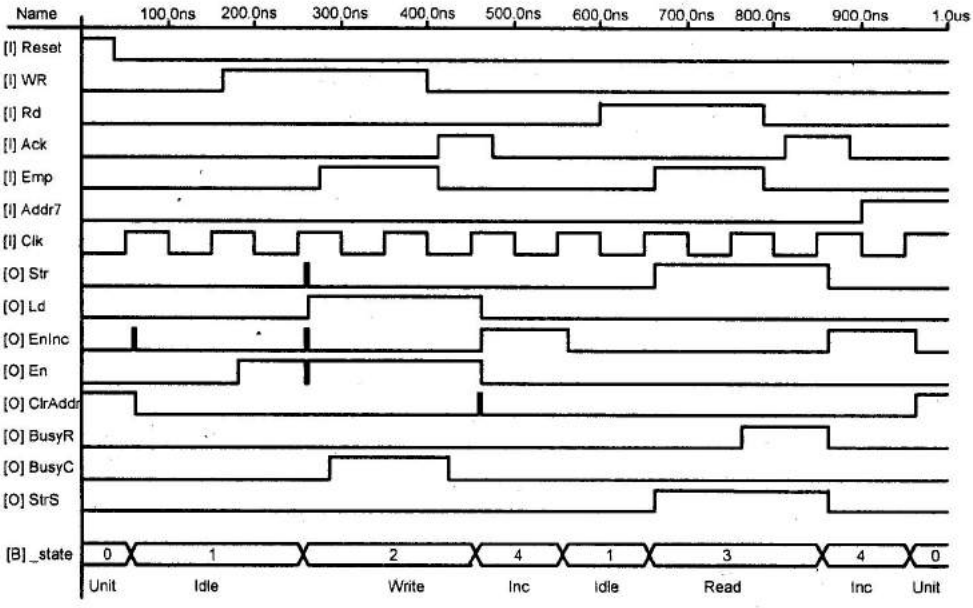


Рис. 12.12. Временная диаграмма поведения автомата

Если, например, сигнал ClrAddr планировался для использования в качестве асинхронного сигнала начальной установки счетчика, то паразитные импульсы могут сбрасывать счетчик в моменты времени, не предусмотренные разработчиком. Одной из причин возникшего расхождения между поведением автомата до и после синтеза является то, что при синтезе компилятор использовал (с целью экономии ресурсов) кодирование состояний автомата. Дешифрация состояний автомата привела к возникновению некоторых из наблюдаемых на диаграмме рисков. Переход к другим способам кодирования (например, к методу One Hot — один триггер соответствует одному состоянию) может снимать проблему. Общим принципом специальных кодировок является закрепление за интересующим проектировщика выходным сигналом одного из триггеров регистра состояния. В языке AHDL (см. [4]), например, в синтаксической конструкции, соответствующей заданию авто-

мата, могут перечисляться такие сигналы и их значения в зависимости от состояний автомата.

Другим широко распространенным и универсальным (не зависящим от причины возникновения) способом борьбы с рисками у выходных сигналов является ориентация на автоматы с синхронными выходами (см. рис. 12.9). Для демонстрации такого подхода в примере выбран сигнал Str. Убрать паразитный импульс (см. рис. 12.12) у исходного комбинационного сигнала можно при формировании синхронного сигнала StrS. Добавление сигнала StrS к проекту привело к необходимости ввода в листинг 12.2 семи вставок (отмечены пронумерованными комментариями):

- вставка 1 — добавляет в проект новый выходной сигнал — StrS;
- вставка 2 — объявляет внутренний сигнал проекта StrS_int, соответствующей комбинационной логике, которая подключена к входу триггера StrS;
- вставка 3 — обеспечивает формирование асинхронного сброса триггера StrS;
- вставка 4 — соответствует указанию на необходимость фиксации в триггере StrS состояния выхода комбинационной логики StrS_int по фронту тактирующего сигнала автомата Clk;
- вставка 5 — определяет нулевое значение сигнала StrS_int по умолчанию;
- вставка 6 — обеспечивает формирование единичного значения комбинационного сигнала StrS_int одновременно с формированием сигналов, обеспечивающих переход автомата из состояния Idle в состояние Read. Приоритетность перехода по сигналу Wr заставляет учесть это в записи условного выражения;
- вставка 7 — позволяет обеспечить нулевое значение сигнала StrS после перехода автомата в состояние Inc. Перезапись комбинационного сигнала StrS_int в триггер блокируется при наличии сигнала Ack.

Из временной диаграммы (см. рис. 12.12) видно, что поведение сигнала StrS повторяет поведение сигнала Str, но в отличие от последнего не имеет паразитных выбросов.

Специальные виды кодировки состояний автомата, как правило, должен предусмотреть разработчик либо путем составления соответствующего (более детального) описания, либо заданием при синтезе указаний на использование соответствующих кодировок (для этого необходимо, чтобы компилятор в исходном тексте распознал описание автомата). Существенную помощь проектировщикам оказывают специально включаемые в используемый маршрут проектирования дополнительные пакеты, которые выполняют предварительную компиляцию проекта. Среди отечественных разработчиков наибольшее распространение получили два подобных пакета.

Одним из них является синтезатор Synplify фирмы Synplicity [XXVIII], а другим LeonardoSpectrum фирмы Mentor Graphics [XVII]. Для использованного в примере программы HDL Designer в качестве такого промежуточного синтезирующего пакета логично ориентироваться на LeonardoSpectrum.

При компиляции текста листинга 12.2 пакет LeonardoSpectrum версии 2002e.16 создал структуру автомата в базисе ИС семейства ACEX 1K, требующую для своей реализации всего 12 логических ячеек (вместо 20). Синтезированная структура опирается на 7 триггеров (DFF), что позволило практически для всех выходных сигналов (исключение составляют сигналы EN и BusyC) использовать выходы триггеров. Созданный синтезатором выходной файл (в формате EDIF) был откомпилирован в САПР MAX+Plus II, а затем поведение результирующей конфигурации ИС было промоделировано. Как и следовало ожидать, на временной диаграмме у всех сигналов (выходов триггеров) отсутствовали ложные выбросы.

Рассмотренный пример показывает, что описание поведения устройства управления цифровыми блоками в форме автоматов открывает для проектировщика широкие возможности для гибкого управления работой используемых операционных блоков.

Введение в язык VHDL-AMS

Язык VHDL-AMS предназначен для моделирования смешанных (непрерывно-дискретных) систем. Стандарт этого языка IEEE Std 1076.1-1999 был утвержден в 1999 г. Язык ориентирован на описание систем достаточно широкого класса — электрических, механических, электромагнитных, тепловых и т. д. Для этого потребовалось не только добавить к возможности описания дискретных фрагментов средства описания поведения и структуры фрагментов, представленных обыкновенными нелинейными дифференциально-алгебраическими уравнениями, но и корректировать описание дискретных систем VHDL и добавить механизмы связи дискретных и непрерывных фрагментов. В настоящий момент язык может использоваться не только для описания непрерывно-дискретных систем, но и для моделирования поведения систем, описанных на этом языке. Примером пакета, поддерживающего решение таких задач и к тому же имеющего удобный графический интерфейс, является программа SystemVision фирмы Mentor Graphics. Стремительное развитие элементной базы реконфигурируемых пользователем аналоговых и цифроаналоговых интегральных схем позволяет надеяться, что появление программных пакетов, синтезирующих описание аналого-цифровых устройств в описание конфигурации соответствующих типов ИС, вопрос не столь уж отдаленного будущего. Ниже остановимся на вопросах моделирования электронных аналого-цифровых систем.

Прежде всего рассмотрим средства языка, требуемые для представления поведения аналоговых сигналов. Основной особенностью этих сигналов является их непрерывность, как в диапазоне значений, так и в процессе изменения во времени. Необходимость описания поведения аналоговых сигналов потребовала введения в язык новых классов объектов **TERMINAL** и **QUANTITY**. Введенные объекты отражают основные свойства аналоговых сигналов. Для контроля допустимости соединения блоков между собой уже в описании интерфейса объектов (**ENTITY**) язык требует сначала указания принадлежности интерфейсных контактов к передаче сигналов определенных классов дискретных (**SIGNAL**) или непрерывных (**TERMINAL** или **QUANTITY**) и лишь затем приводится характеристика (технологическая) типа соединения (например, OUT `bit` и `electrical` соответственно).

В архитектурных телах описываемых объектов могут использоваться два различных подхода к описанию поведения непрерывных сигналов. Первый подход, применяемый к консервативным системам (conserved systems), ориентирован на закон сохранения энергии при определении взаимодействия двух соединенных аналоговых элементов. В трактовке электрических цепей это соответствует поведению, подчиняющемуся законам Кирхгофа. Второй подход, характерный для неконсервативных систем (non-conserved systems), соответствует направленному воздействию одного аналогового объекта на другой. В этом случае описание аналогового объекта включает фрагменты, отвечающие за получение входных величин, а также фрагменты, соответствующие их последующему математическому преобразованию, и фрагменты, определяющие вывод преобразованных величин. Класс портов (контактов) **TERMINAL** служит для определения консервативных портов модели, предназначенных для подключения цепей с непрерывными сигналами. Класс портов (контактов) **QUANTITY** так же предназначен для подключения непрерывных сигналов, но служит для определения неконсервативных портов модели и в соответствии с принципом обработки данных таких портов требует указания направления передачи аналоговых величин IN или OUT.

В декларативной части архитектурных тел помимо определения дискретных компонентов потребовалось введение объявления объектов нового класса — **QUANTITY**. Основное свойство этих объектов — возможность принимать значения в непрерывном диапазоне и изменять их в промежутки времени между отдельными событиями. Существуют три основные разновидности **QUANTITY**-параметра:

- free **QUANTITY**;
- branch **QUANTITY**;
- source **QUANTITY**.

Первый тип параметров (free **QUANTITY**) может использоваться для представления аналоговых величин в неконсервативных фрагментах. Этот тип пара-

метров упрощает просмотр в моделирующих программах временного поведения сигналов и делает более простым и понятным описание поведения моделируемого фрагмента.

Второй тип параметров (branch **QUANTITY**) используется при определении консервативных систем. Описание электрических цепей (с точки зрения законов Кирхгофа) требует определения для **TERMINAL**-контакта или **QUANTITY**-параметра трех аспектов поведения. Один соответствует **ACROSS**-аспекту — потенциалу между **TERMINAL**-контактами, другой **THROUGH**-аспекту — току между контактами и, наконец, последний **TO**-аспект — идентификация связываемых контактов между собой.

Третий тип параметров (source **QUANTITY**) применяется при спектральном анализе сигналов или анализе шумов.

В основной части архитектурного тела поведение цифровых фрагментов описывалось в языке VHDL предложениями параллельных операторов (Concurrent Statement), а в языке VHDL-AMS поведение аналоговых фрагментов задается новым типом предложений (Simultaneous Statement — с непрерывными операторами). Оба типа предложений могут размещаться в архитектурном теле в произвольном порядке.

Хотя синтаксис непрерывных операторов естественно отличается от синтаксиса параллельных операторов, набор операторов традиционен и содержит как безусловные, так и условные операторы. Прежде всего требуется оператор, определяющий функционирование аналоговых сигналов. Оператор (= =), определенный в стандарте как элементарное непрерывное предложение (Simple_Simultaneous_Statement), для неконсервативных систем задает функциональную зависимость выходного параметра (free **QUANTITY**) от времени. Частным случаем зависимости может быть и постоянство значения. Для консервативных же систем оператор, кроме задания зависимости параметра **QUANTITY** от времени, определяет правила взаимодействия **ACROSS** и **THROUGH** аспектов параметра (branch **QUANTITY**) между собой.

Далее необходимы условные непрерывные операторы (**IF** и **CASE**). Внешне синтаксис этих операторов совпадает с синтаксисом аналогичных последовательных операторов и отличается от них заменой ключевого слова **THEN** на **USE**. В теле этих операторов естественно должны использоваться непрерывные операторы. Непрерывные предложения могут включаться и в тело параллельных операторов (**FOR**. . .**GENERATE**).

Задание функциональной зависимости между различными аспектами объектов типа **QUANTITY** (для электрических цепей — током и напряжением) потребовало не только введения нового оператора присваивания (= =), но и новых типов атрибутов для аспектов QUANTITY-параметров в том числе: `dot` — временного дифференцирования параметра; `int` — интеграла во времени и `delyaed(T)` — задержки на время T изменения параметра.

Пример описания идеального конденсатора соответствует последовательности следующих операторов:

```
ENTITY capacitor IS
  GENERIC (
    cap             : capacitance;          -- емкость в фарадах
    v_ic            : real := real'low);    -- начальное напряжение
   PORT (
    TERMINAL p1, p2 : electrical);
END ENTITY capacitor;
ARCHITECTURE ideal OF capacitor IS
  QUANTITY v ACROSS i THROUGT p1 TO p2;
BEGIN
  IF domain = quiescent_domain AND v_ic /= real'low USE
    v == v_ic;                          -- Учитываем начальное значение v_ic
  ELSE
    i   == cap * v'dot;                 -- Основное дифференциальное уравнение
  END USE;
END ARCHITECTURE ideal;
```

В приведенном примере непрерывное изменение значений напряжений на обкладках конденсатора задается дифференциальным уравнением зависимости тока и напряжения $I = C \cdot dV/dt$, при этом учитывается начальное значение напряжения.

Ряд специальных синтаксических конструкций языка позволяет задать в моделирующей программе требование спектрального анализа (**SPECTRUM**) или анализа шумов (**NOISE**) контролируемого параметра (модели электрического сигнала).

Смешанное моделирование требует введения механизмов и средств связи между событийно-управляемым моделированием цифровой части системы и решения задачи Коши для аналоговой части.

Чтобы обеспечить связь аналогового фрагмента с цифровым, аналоговый фрагмент должен сформировать событие. Инициализация события возникает в процессе решения дифференциальных уравнений, описывающих аналоговую часть системы при достижении выбранным непрерывным параметром определенной величины. Чаще всего формируется сигнал, запускающий соответствующее действие (например, процесс). Для этого могут использоваться условные операторы последовательные или параллельные. Для записи условия достижения используется атрибут `above`, определенный в языке

для QUANTITY. Примером фрагмента, описывающего поведение аналогового компаратора, который формирует единичное значение дискретного сигнала d при превышении разностью потенциалов V между входными контактами a и ref заданного порогового уровня threshold, может служить следующий приведенный текст.

```
ENTITY comparator IS
  GENERIC (threshold: real:=0.0);
  PORT (TERMINAL a, ref : electrical;
    SIGNAL d : OUT bit);
END ENTITY comparator;

ARCHITECTURE simple OF comparator IS
  QUANTITY v ACROSS i THROUGH a TO ref;
BEGIN
    v == 1.0E6*i;              -- входное сопротивление компаратора 1 Мом.
    d <= '1' WHEN v'above(threshold) ELSE '0';
END ARCHITECTURE simple;
```

Передача управления от дискретных к аналоговым фрагментам происходит при возникновении соответствующего события. В этом случае событие в дискретном фрагменте инициализирует запуск или изменения параметров того или иного непрерывного параметра в аналоговом фрагменте. Инициализация достигается исполнением оператора присваивания непрерывного сигнала либо автономного, либо входящего в условный оператор. В качестве примера рассмотрим описание идеализированного переключателя электрических сигналов. При изменении состояния цифрового сигнала sw_state происходит соответствующее изменение параметров соединения выходного контакта p_out с входными контактами p_in1 и p_in2. Атрибут ramp дискретного сигнала r_sig1 в примере (листинг 12.3) обеспечивает формирование соответствующего значения аналогового параметра r1 через заданный временной интервал с сохранением непрерывности значений параметра r1.

Листинг 12.3

```
ENTITY switch IS
  PORT ( sw_state : IN std_ulogic; -- Цифровой управлякщий вход
    TERMINAL p_in1, p_in2, p_out : electrical );
    -- Аналоговые контакты переключателя
END ENTITY switch;
```

```
ARCHITECTURE ideal OF switch IS
  CONSTANT r_open : resistance := 1.0e6;
  -- Сопротивление открытого состояния переключателя
  CONSTANT r_closed : resistance := 0.001;
  -- Сопротивление закрытого состояния переключателя
  CONSTANT trans_time : real := 0.00001;
  -- Время переключения в каждое состояние
  SIGNAL r_sig1 : resistance := r_closed;
  -- переменная сопротивление закрытого состояния ключа
  SIGNAL r_sig2 : resistance := r_open;
  -- переменная сопротивление открытого состояния ключа
  QUANTITY v1 ACROSS i1 THROUGH p_in1 TO p_out;  -- V и I от in1 до out
  QUANTITY v2 ACROSS i2 THROUGH p_in2 TO p_out;  -- V и I от  in2 до out
  QUANTITY r1 : resistance;
  -- время зависимое сопротивление от in1 до out
  QUANTITY r2 : resistance;
  -- время зависимое сопротивление от in2 до out
BEGIN
  PROCESS (sw_state) IS -- запуск от дискретного сигнала sw_state
  BEGIN
    IF sw_state = '0' OR sw_state = 'L' THEN
    -- закрыть sig1, открыть sig2
      r_sig1 <= r_closed;
      r_sig2 <= r_open;
    ELSIF sw_state = '1' OR sw_state = 'H' THEN
    -- открыть sig1, закрыть sig2
      r_sig1 <= r_open;
      r_sig2 <= r_closed;
    END IF;
  END PROCESS;
  r1 == r_sig1'ramp(trans_time, trans_time);
  -- Перевод сигнала в параметр с сохранением непрерывности сопротивления
  r2 == r_sig2'ramp(trans_time, trans_time);
  -- Перевод сигнала в параметр с сохранением непрерывности сопротивления
  v1 == r1 * i1; -- Применение закона Ома для контакта in1
  v2 == r2 * i2; -- Применение закона Ома для контакта in2
END ARCHITECTURE ideal;
```

Рассмотренные выше фрагменты используем при описании смешанной аналого-цифровой схемы, приведенной на рис. 12.13. Схема содержит осциллятор (генератор osc цифровых сигналов V_OSC = V_INP с периодом 80 мкс и длительностью импульса 50 мкс), управляющий через инвертор U1 цифро-аналоговым переключателем switch. Образуемые в результате переключения импульсы V_SW поступают на интегрирующую цепочку, состоящую из резистора R1 (10 ком) и конденсатора C1 (1 нф). Образуемое на конденсаторе напряжение V_INT сравнивается в аналого-цифровом компараторе comparator с пороговым уровнем LEVEL. Выходная цифровая схема 2И (блок U3) осуществляет временную селекцию цифрового сигнала V_MID, образуемого компаратором, с входным напряжением V_OSC = V_INP и формирует выходной сигнал V_OUT.

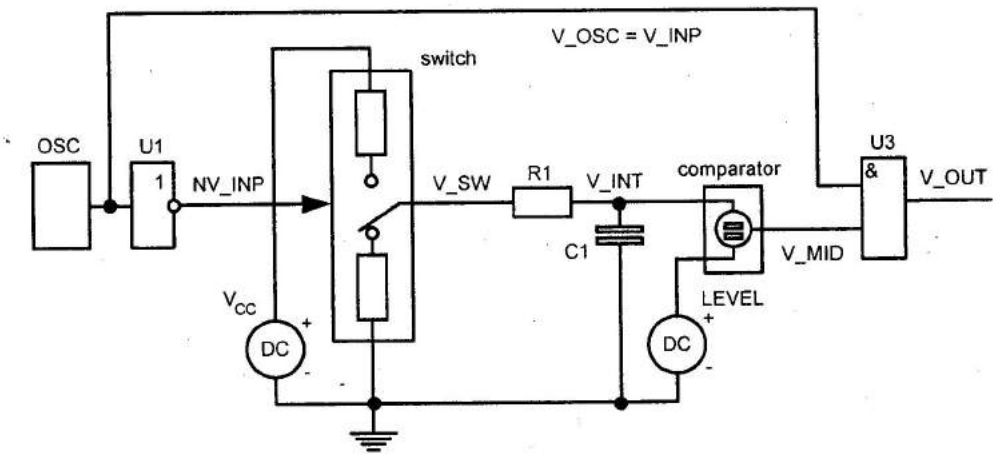


Рис. 12.13. Смешанная аналого-цифровая система

Языковое описание этой схемы создавалось с привлечением средств моделирующего пакета SystemVision (версия SV2002.0.16) фирмы Mentor Graphics. Для этого в состав проекта common сначала были включены текстовые описания, рассмотренные ранее примеров описаний поведения блоков switch и comparator, идеализированной модели конденсатора и резистора, а затем также описания традиционных дискретных блоков. Затем были образованы соответствующие этим компонентам графические блоки, из них и из стандартных элементов библиотеки SystemVision была создана схема системы, соответствующая рисунку. Результат компиляции верхнего уровня иерархии приведен в листинге 12.4.

```
--
-- File : d:\CADWork\AMS\Default\genhdl\vhdl\common.vhd
-- CDB  : d:\CADWork\AMS\default\default.cdb
-- By   : CDB2VHDL Netlister version 16.1.0.2

-- Entity/architecture declarations

LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;

LIBRARY IEEE_proposed;
USE IEEE_proposed.electrical_systems.ALL;

ENTITY common IS
END common;

ARCHITECTURE common OF common IS
    -- Component declarations
    -- Signal declarations
    TERMINAL LEVEL : electrical;
    SIGNAL nV_INP : std_logic;
    TERMINAL V_IN : ELECTRICAL;
    SIGNAL V_INP : std_logic;
    SIGNAL V_MID : std_logic;
    SIGNAL V_OUT : std_logic;
    TERMINAL V_SW : electrical;
    TERMINAL XSIG010010 : electrical;
BEGIN
    -- Signal assignments
    -- Component instances
    OSC1 : ENTITY work.OSC(ideal)
        PORT MAP(
            clock_out => V_INP
        );
```

```
U1 : ENTITY work.inverter(ideal)
    PORT MAP (
        input => V_INP,
        output => nV_INP
    );
switch1 : ENTITY work.switch(ideal)
    PORT MAP (
        sw_state => nV_INP,
        p_in1 => ELECTRICAL_REF,
        p_in2 => XSIG010010,
        p_out => V_SW
    );
R1 : ENTITY work.resistor(ideal)
    GENERIC MAP(
        res => 10000.0
    )
    PORT MAP (
        p1 => V_SW,
        p2 => V_IN
    );
C1 : ENTITY work.capacitor(ideal)
    GENERIC MAP (
        cap => 1.0e-9
    )
    PORT MAP (
        p1 => V_IN,
        p2 => ELECTRICAL_REF
    );
comparator1 : ENTITY work.comparator(simple)
    PORT MAP (
        a => V_IN,
        ref => LEVEL,
        d => V_MID
    );
v1 : ENTITY work.v_constant(ideal)
    GENERIC MAP (
```

```
            level => 5.0
        )
        PORT MAP (
            pos => XSIG010010,
            neg => ELECTRICAL_REF
        );
    v2 : ENTITY work.v_constant(ideal)
        GENERIC MAP (
            level => 2.5
        )
        PORT MAP (
            pos => LEVEL,
            neg => ELECTRICAL_REF
        );
    U3 : ENTITY work.and2(ideal)
        PORT MAP (
            in1 => V_INP,
            in2 => V_MID,
            output => V_OUT
        );
END common;
```

Как обычно, проект опирается на стандартную библиотеку (строка — `LIBRARY IEEE;`) и использует понятие стандартной логики (строка — `USE IEEE.std_logic_1164.ALL;`), кроме того, аналоговая часть проекта требует подключения новой библиотеки (строки — `LIBRARY IEEE_proposed;` и `USE IEEE_proposed.electrical_systems.ALL;`).

Пакет создал проект в форме структурного описания соединения компонентов структуры. Поэтому его вид характерен для традиционных VHDL-программ. Основное отличие состоит в том, что для соединений используются объявленные в архитектурном теле цифровые (`SIGNAL`) и аналоговые (`TERMINAL`) сигналы. Остальные отличия можно увидеть только в файле предыдущего уровня иерархии (в описаниях поведения аналоговых и смешанных компонентов).

Результаты компиляции использовались для моделирования поведения проекта во времени. Эти результаты приведены на рис. 12.14 и показывают изменение во времени как цифровых, так и аналоговых сигналов.

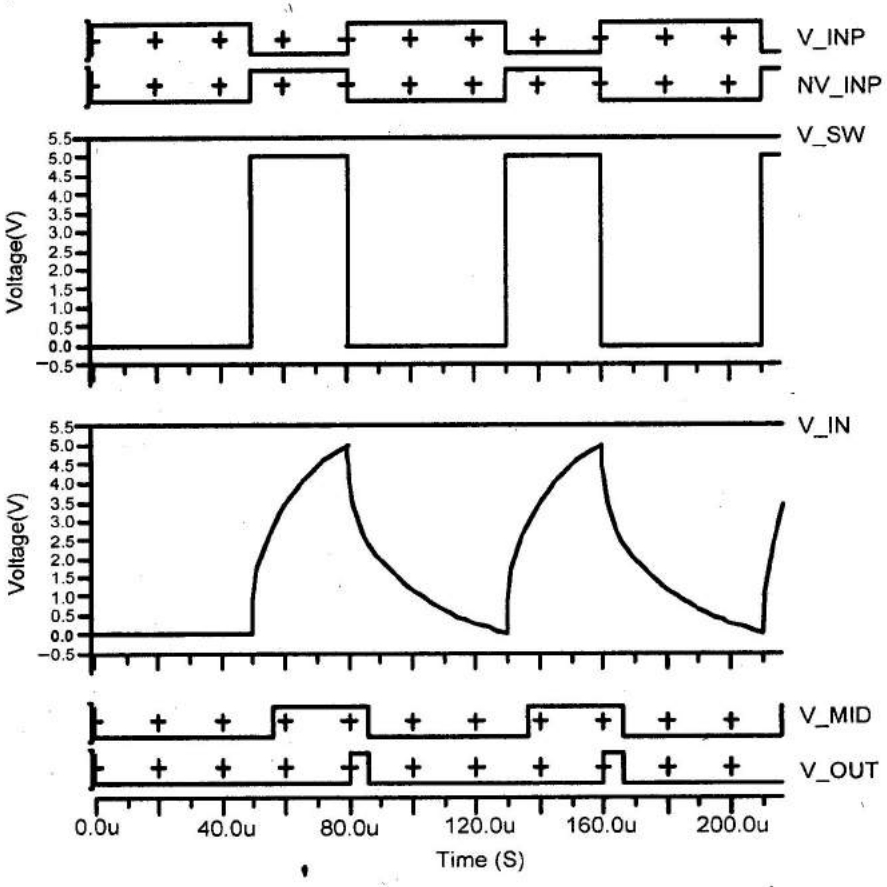


Рис. 12.14. Результаты моделирования аналого-цифровой схемы

§ 12.9. Пример автоматизированного проектирования цифрового устройства с использованием языков описания аппаратуры

Современные методы и средства проектирования рассмотрим на примере разработки устройства, либо записывающего по запросу параллельный восьмиразрядный код в буферное ОЗУ, либо выводящего байт из заданного адреса буферного ОЗУ в форме последовательного кода. Для определенности будем ориентироваться на микросхемы программируемой логики фирмы Altera, а вследствие этого и на САПР этой же фирмы MAX+Plus II.

Первый этап. Рассмотрение ТЗ на разрабатываемое устройство

Независимо от формы представления, ТЗ, очевидно, должно содержать следующие ключевые сведения:

- объем буферного ОЗУ 256 восьмиразрядных слов;
- запись в ОЗУ осуществляется сигналами внешнего управляющего устройства;
- внешнее чтение статуса устройства позволяет определить состояние его выходного регистра (пуст или полон);
- вывод последовательного кода осуществляется по запросу приемника последовательного кода и сопровождается стробирующими сигналами со стороны разрабатываемого устройства.

Перечисленные выше пункты ТЗ предопределяют основные блоки устройства и их взаимодействие. Блочная схема устройства приведена на рис. 12.15. Функциональное назначение блоков следует из их названий. Схема укрупненно отображает следующие процессы.

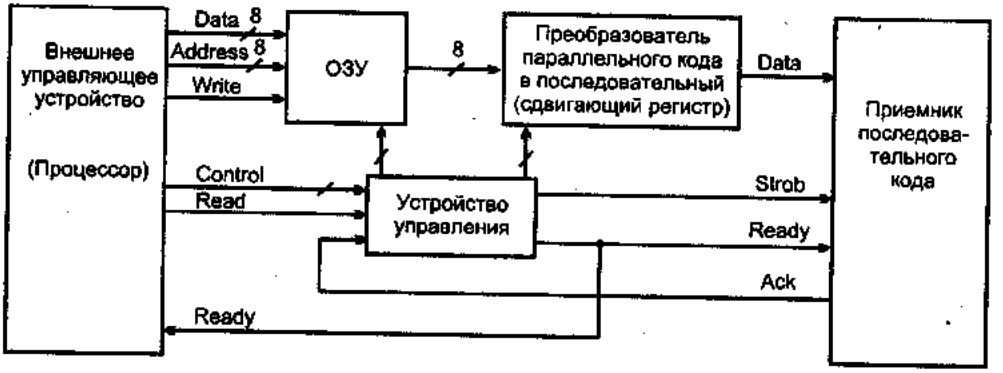


Рис. 12.15. Блок-схема устройства, принятого в качестве примера для проектирования средствами САПР

Внешнее управляющее устройство (процессор) обеспечивает запись байтов в ОЗУ, подавая на него помимо восьмиразрядных данных также восьмиразрядный адресный код и строб записи Write. Преобразователь параллельного кода в последовательный представляет в своей основе сдвигающий регистр, загружаемый параллельно байтом из ОЗУ, и затем по командам из устройства управления выводящий последовательные данные во внешнее устройст-

во — приемник последовательного кода. Появление разрядов последовательного кода отмечается во времени сигналами стробирования Strob. Загрузка данных в преобразователь параллельного кода инициируется процессором по сигналу Read, адрес загружаемых данных должен быть перед этим задан процессором.

Готовность преобразователя кода к передаче данных индицируется сигналом Ready, поступающим как на приемник последовательных данных, так и на соответствующий вход процессора. Асинхронный характер обмена с приемником данных обеспечивается сигналом разрешения передачи Ack со стороны приемника. В состав сигналов шины Control входят сигналы тактирования, сброса и др.

Второй этап. Разработка общей структуры операционного блока

Сопоставляя функциональный состав библиотеки САПР MAX+Plus II и блоков схемы (см. рис. 12.15), нетрудно увидеть, что для реализации рассматриваемого устройства из состава библиотеки выбранной САПР можно использовать следующий набор библиотечных параметризируемых модулей (LPM):

- блок ОЗУ (LPM_RAM_DQ) с организацией 256×8;
- восьмиразрядный сдвигающий регистр выходного кода (LPM_SHIFTREG);
- счетчик сдвигов регистра вывода на 8 состояний (LPM_COUNTER).

Понятие параметризуемых модулей соответствует возможности настроить выбранный библиотечный элемент на определенный режим функционирования, на определенную разрядность данных, их полярность и т. д.

Структурная схема устройства, включающая эти операционные блоки и автомат, управляющий считыванием кода из ОЗУ и его преобразованием в последовательную форму (AvtOutBt), может приобрести вид, приведенный на рис. 12.16. Кроме указанных ранее базовых блоков, в схеме присутствует ряд дополнительных элементов. Условные обозначения всех элементов схемы соответствуют стандарту, принятому в САПР MAX+Plus II. Необходимость введения дополнительных элементов (инвертора, четырех D-триггеров и двух схем 2ИЛИ-НЕ) диктуется требованиями временной или аппаратной совместимости отдельных блоков схемы. Более подробные пояснения будут приведены в следующем разделе, поскольку этапы разработки операционной части и устройства управления операционными элементами тесно связаны и обычно выполняются итерационно.

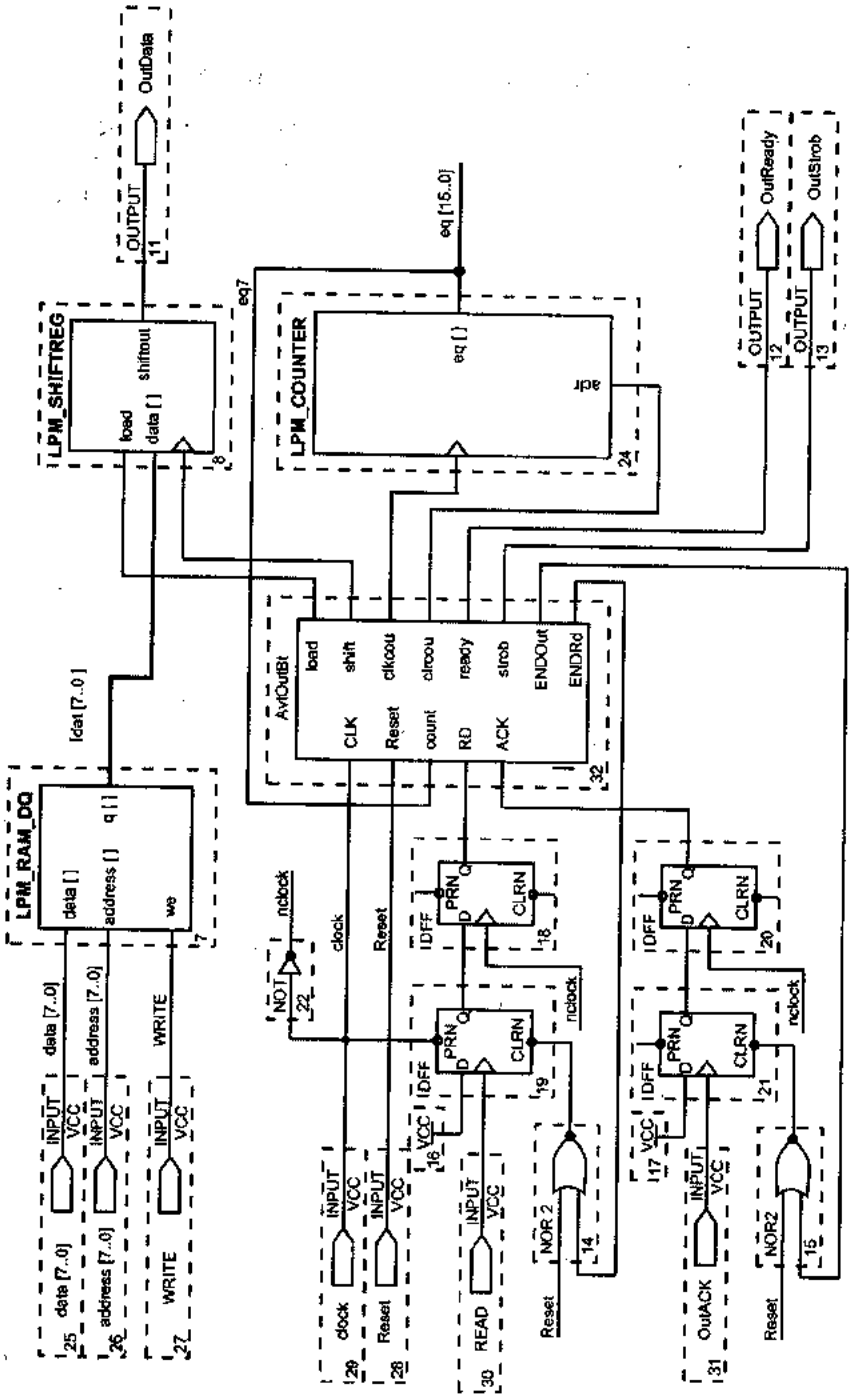


Рис. 12.16. Структурная схема устройства, проектируемого средствами САПР

Третий этап. Описание работы управляющего автомата

При разработке поведения управляющего автомата необходимо учесть, что функционирование устройства определяется сигналом clock и происходит асинхронно относительно внешнего устройства, управляющего чтением и записью в ОЗУ и относительно другого внешнего устройства, запрашивающего и принимающего информацию в последовательной форме.

При выборе из библиотеки САПР в качестве ОЗУ модуля типа LPM_RAM_DQ (т. е. с раздельными шинами чтения и записи данных) и при его настройке на асинхронный режим работы исчезает целый ряд проблем. Во-первых, нет необходимости введения элементов, разделяющих данные для записи и считывания. Во-вторых, полностью снимается с управляющего автомата проблема организации синхронизации режима записи данных в ОЗУ со стороны внешнего устройства. А вот операция чтения из ОЗУ должна быть синхронизирована с работой автомата. Поэтому внешнее устройство, управляющее памятью, прежде чем снять установленный адрес, должно убедиться, что предыдущая информация из сдвигового регистра забрана. Для этой цели оно может ориентироваться на сигнал OutReady. Сигнал устанавливается, когда информация уже находится в сдвигающем регистре, и сбрасывается, когда выдан последний бит данных.

Возможный алгоритм работы устройства управления разрабатываемого устройства, отвечающий сформулированным выше требованиям, может приобрести вид, соответствующий граф-схеме переходов автомата, приведенной на рис. 12.17. Граф-схема переходов при помощи графического редактора программы StateCAD Version 3.2 пакета Workview Office фирмы Viewlogic (сейчас Innoveda [XII]) была занесена в соответствующий диаграммный файл, что, как будет показано далее, существенно упрощает не только отладку и возможные корректировки алгоритма, но и создание соответствующих программных текстов.

Перейдем к анализу автомата AvtOutBt, управляющего выводом последовательного кода по запросу и сопровождающего такую выдачу сигналами стробирования.

Основу алгоритма составляют два последовательно выполняемых блока. Первый блок по сигналу Rd автомата, образуемому из сигнала READ внешнего управляющего устройства, считывает байт из памяти, а затем загружает его в сдвигающий регистр. Второй блок содержит последовательность действий, которая в ответ на запускающий внешний сигнал OutAck, преобразуемый в сигнал Ack автомата, осуществляет циклический вывод на выходной контакт OutData данных из сдвигающего регистра и при этом сопровождает каждый бит стробирующим сигналом OutStrob (контакт автомата Strob). Количество требуемых итераций цикла подсчитывает счетчик сдвигов. Следует

обратить внимание на петли ожидания в состояниях Idle, EndLdBt и WaitOut, наличие которых обеспечивает необходимую синхронизацию сигналов "запрос-ответ" от внешнего устройства (квитирование). Определенные проблемы могли бы создать сигналы READ и OutAck ввиду асинхронности изменения их значений относительно тактовой частоты автомата. Триггеры DFF 18 и 19 (см. рис. 12.16) обеспечивают требуемые времена предустановки и удержания входных сигналов и позволяют избежать некорректной работы автомата (подробнее об этом говорилось в *§ 12.8*). Триггеры DFF 19 и 21 введены в схему, чтобы свести к минимуму вероятность появления состояния *метастабильности (см. § 3.3)* у триггеров DFF 18 и 20.

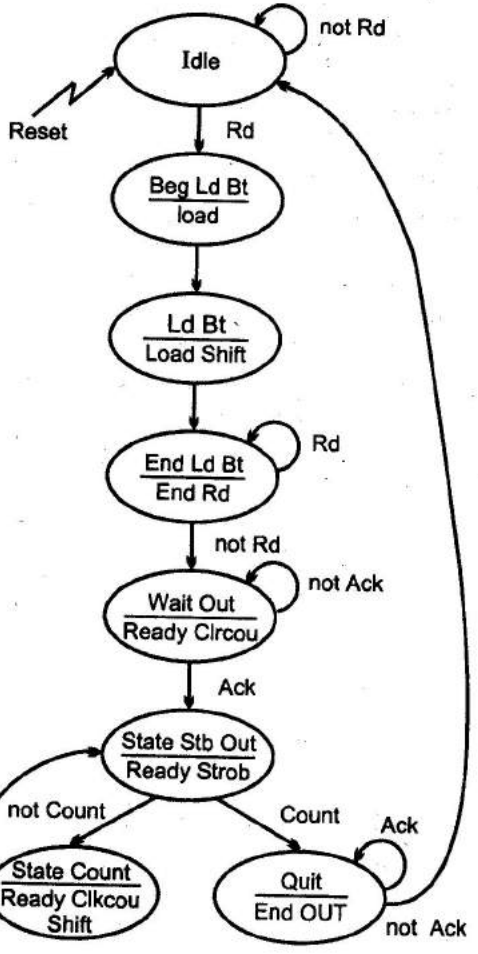


Рис. 12.17. Граф-схема автомата управления для проектируемого средствами САПР устройства

Пояснения к синтаксису AHDL и VHDL программ устройства управления

Для автомата нашего примера с помощью программы StateCAD Version 3.2 были выполнены две трансляции диаграммы (для разных вариантов языкового описания). Во-первых, был создан вариант, ориентированный на возможности языка описания аппаратуры низкого уровня AHDL (листинг 12.5). Во-вторых, было создано описание этого же автомата на языке высокого уровня VHDL (листинг 12.6).

Начнем с анализа *программы автомата AvtOutBt на языке AHDL*. Этот вариант описания автомата AvtOutBt, используемый в проекте, компилировался с помощью программы StateCAD с ориентацией на язык AHDL фирмы Altera. Как и в большинстве других языков, прежде всего в программу вводится фрагмент, отвечающий за интерфейс программы. После ключевого слова **SUBDESIGN** и имени проекта AutOutBt в круглых скобках перечислены все входные и выходные сигналы с отнесением их к соответствующим типам. После этого начинается собственно программа. Как и в других языках (если в этом есть необходимость), она начинается с раздела описания переменных (**VARIABLE**). В нашем примере в данном разделе определяется автомат (машина состояний) с именем `sreg` и перечисляются все его состояния. Кроме этого, в этом же разделе для устранения недопустимых пиков у сигнала ready вводится дополнительный триггер DFF, входному контакту (**NODE**) которого присваивается имя `TempReady`.

Листинг 12.5

```
% AHDL code created by Visual Software Solution's StateCAD Version 3.2 %
% This AHDL code was generated using: %
% enumerated state assignment with structured code format. %
% Minimization is enabled, implied else is enabled, %
% and outputs are manually optimized. %

SUBDESIGN AVTOUTBT(CLK, Reset, count, RD, ACK: INPUT;
  load, shift, clkcou, clrcou, ready,
  strob, ENDOut, ENDRd: OUTPUT;)
VARIABLE
  TempReady: NODE;
  sreg: MACHINE WITH STATES (Idle, BegLdBt, LdBt, EndLdBt, WaitOut,
StateStbOut, StateCount, Quit);
```

```
BEGIN
% Clock setup %
  sreg.clk=CLK;
  ready=DFF(TempReady, CLK, VCC, VCC);
  IF (Reset) THEN
   sreg=Idle;
   load=GND;   shift=GND;   clkcou=GND;   clrcou=GND;
   TempReady=GND;   strob=GND;   ENDOut=GND;   ENDRd=GND;
  ELSE
   CASE sreg IS
     WHEN Idle=>
       load=GND;   shift=GND;   clkcou=GND;       clrcou=GND;
       TempReady=GND;  strob=GND;   ENDOut=GND;   ENDRd=GND;
      IF (RD) THEN  sreg=BegLdBt;  END IF;
      IF (!RD) THEN  sreg=Idle;  END IF;
     WHEN BegLdBt=>
       load=VCC;   shift=GND;   clkcou=GND;       clrcou=GND;
       TempReady=GND;   strob=GND;  ENDOut=GND;   NDRd=GND;
      sreg=LdBt;
     WHEN LdBt=>
       load=VCC;   shift=VCC;   clkcou=GND;       clrcou=GND;
       TempReady=GND;   strob=GND;  ENDOut=GND;   ENDRd=GND;
      sreg=EndLdBt;
     WHEN EndLdBt=>
       load=GND;   shift=GND;   clkcou=GND;       clrcou=GND;
       TempReady=GND;  strob=GND;   ENDOut=GND;   ENDRd=VCC;
      IF (RD) THEN  sreg=EndLdBt;  END IF;
      IF (!RD) THEN  sreg=WaitOut;  END IF;
     WHEN WaitOut=>
       load=GND;   shift=GND;   clkcou=GND;    clrcou=VCC;
       TempReady=VCC;  strob=GND;   ENDOut=GND;   ENDRd=GND;
      IF (ACK) THEN  sreg=StateStbOut;  END IF;
      IF (!ACK)THEN  sreg=WaitOut;       END IF;
     WHEN StateStbOut=>
       load=GND;   shift=GND;   clkcou=GND;   clrcou=GND;
       TempReady=VCC;  strob=VCC;   ENDOut=GND;   ENDRd=GND;
      IF (count) THEN sreg=Quit;   END IF;
```

```
      IF (!count)THEN sreg=StateCount;  END IF;
     WHEN StateCount=>
       load=GND;    shift=VCC;    clkcou=VCC;        clrcou=GND;
       TempReady=VCC;  strob=GND;    ENDOut=GND;    ENDRd=GND;
      sreg=StateStbOut;
     WHEN Quit=>
       load=GND;    shift=GND;    clkcou=GND;        clrcou=GND;
       TempReady=GND;    strob=GND;    ENDOut=VCC;    ENDRd=GND;
      IF (ACK) THEN  sreg=Quit;  END IF;
      IF (!ACK)THEN  sreg=Idle;  END IF;
   END CASE;
  END IF;
END;
```

Собственно программа начинается с предложения `sreg.clk=CLK`, которое является предложением, обеспечивающим переходы автомата от состояния к состоянию. Следующим действием, требующим тактирования, является перепись внутреннего сигнала TempReady в выходной триггер ready (устранение пиков достигается за счет использования разных фронтов тактирующего сигнала у автомата и у триггера).

В помещенном далее теле описания автомата полностью повторяются (в семантическом плане) операторные конструкции, интерпретирующие в синтаксисе языка AHDL все основные блоки граф-схемы переходов автомата, приведенной на рис. 12.17. Для перебора состояний автомата программа ориентируется на возможности, предоставляемые оператором **CASE** — (**CASE IS**, **WHEN**, ... **END CASE**), а для организации необходимых разветвлений использует оператор **IF** — (**IF**, **THEN**, **[ELSE]**, **END IF**).

Первая конструкция **IF** обеспечивает сброс устройства в исходное состояние при наличии сигнала Reset. Основу альтернативной ситуации (отсутствие Reset) образует встроенная после ключевого слова **ELSE** конструкция **CASE**. Анализируя текущее состояние автомата `sreg`, эта конструкция для каждого возможного состояния автомата формирует после конструкции (**WHEN** <имя состояния> =>) требуемую последовательность выходных сигналов и подготавливает переход к следующему состоянию автомата. Входящие в состав некоторых вариантов **WHEN** условные операторы (**IF** — **THEN** — **END IF**) обеспечивают выполнение различных разветвлений хода работы автомата. Программа завершается ключевым словом **END**.

Следует особо отметить, что ориентация в тексте программы на переменные, относящиеся к данным перечисляемого типа **MACHINE WITH STATES**,

позволяет компилятору MAX+Plus II учитывать опционное задание способа кодирования синтезируемого автомата (сравните с примером *§ 12.8*).

Второй вариант описания автомата, который может управлять спроектированным устройством, компилировался с помощью программы StateCAD *с ориентацией на язык VHDL*. При компиляции из графической формы в текстовую программа StateCAD учитывает, для компилятора какой фирмы предполагается использовать описание автомата. Аналогичные соображения должны приниматься во внимание при ручном написании программ. Это ограничение возникает из-за того, что набор допустимых синтаксических конструкций языка для различных фирм существенно отличается от стандартного. Для примера выбрана ориентация на САПР Workview Office (как имеющую меньшие ограничения). В этой программе многое повторяет (в плане функционального назначения) предыдущую программу, и большинство отличий связано с синтаксисом операторов языка VHDL.

```
-- VHDL code created by Visual Software Solution's StateCAD Version 3.2
-- This VHDL code (for use with Workview Office) was generated using:
-- enumerated state assignment with structured code format.
-- Minimization is enabled, implied else is enabled,
-- and outputs are manually optimized.

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

LIBRARY synth;
USE synth.vhdlsynth.ALL;

ENTITY AvtOutBt IS
  PORT (CLK,Reset,count,rd,ack : IN std_logic;
   load,shift,clkcou,clrcou,ready,strob,endout,endRd: OUT std_logic);
END;

ARCHITECTURE BEHAVIOR OF AvtOutBt IS
  TYPE type_sreg IS (Idle,BegLdBt,LdBt,EndLdBt,WaitOut,StateStbOut,
   StateCount,Quit);
  SIGNAL sreg, next_sreg: type_sreg;
```

```
BEGIN
  PROCESS (CLK, next_sreg)
  BEGIN
   IF CLK='1' AND CLK'event THEN
     sreg<=next_sreg;
   END IF;
  END PROCESS;

  PROCESS (sreg,Reset,count,rd,ack)
  BEGIN
   load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
   ready<='0'; strob<='0'; endout<='0'; endRd<='0';
   next_sreg<=Idle;

   IF (Reset='1') THEN
     load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
     ready<='0'; strob<='0'; endout<='0'; endRd<='0';
     next_sreg<=Idle;
   ELSE
     CASE sreg IS
      WHEN Idle =>
        load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
        ready<='0'; strob<='0'; endout<='0'; endRd<='0';
        IF (rd='0') THEN
         next_sreg<= Idle;
        END IF;
        IF (rd='1') THEN
         next_sreg<=BegLdBt;
        END IF;
      WHEN BegLdBt =>
        load<='1'; shift<='0'; clkcou<='0'; clrcou<='0';
        ready<='0'; strob<='0'; endout<='0'; endRd<='0';
        next_sreg<=LdBt;
      WHEN LdBt =>
        load<='1'; shift<='1'; clkcou<='0'; clrcou<='0';
        ready<='0'; strob<='0'; endout<='0'; endRd<='0';
        next_sreg<=EndLdBt;
```

```
      WHEN EndLdBt =>
        load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
        ready<='0'; strob<='0'; endout<='0'; endRd<='1';
        next_sreg<=Idle;
      WHEN WaitOut =>
        load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
        ready<='0'; strob<='0'; endout<='0'; endRd<='0';
        IF (ack='0') THEN next_sreg<=WaitOut; END IF;
        IF (ack='1') THEN next_sreg<=StateStbOut; END IF;
      WHEN StateStbOut =>
        load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
        ready<='1'; strob<='1'; endout<='0'; endRd<='0';
        IF (count='0') THEN next_sreg<=StateCount; END IF;
        IF (count='1') THEN next_sreg<=Quit; END IF;
      WHEN StateCount =>
        load<='0'; shift<='1'; clkcou<='1'; clrcou<='0';
        ready<='1'; strob<='0'; endout<='0'; endRd<='0';
        next_sreg<=StateStbOut;
      WHEN Quit =>
        load<='0'; shift<='0'; clkcou<='0'; clrcou<='0';
        ready<='0'; strob<='0'; endout<='1'; endRd<='0';
        IF (ack='0') THEN next_sreg<=Idle; END IF;
        IF (ack='1') THEN next_sreg<=Quit; END IF;
      WHEN OTHERS =>
    END CASE;
  END IF;
 END PROCESS;
END BEHAVIOR;
```

Разделы проектного модуля типичны для языка VHDL. В заголовочном разделе **ENTITY** (аналог прототипа в алгоритмических языках) перечислены имена и типы всех сигналов: входных внешних управляющих сигналов — тактового сигнала (CLK), начального сброса (Reset), запросов на чтение и вывод (RD и ACK) соответственно, и, наконец, сигнала окончания счета (count) и выходных управляющих сигналов — управления сдвигающим регистром (load, shift), управления счетчиком сдвигов (clrcou, clkcou), внешними выходными сигналами (ready, strob) и сигналами окончания подрежимов (endout, endRd).

Следующий раздел — **ARCHITECTURE** — представляет собой описание архитектуры или поведения (в нашем случае поведения) блока, интерфейс которого был описан в **ENTITY**. Как и в обычных языках, в начале раздела дается описание типов и объявление переменных, используемых при описании действий, выполняемых в разделе **ARCHITECTURE**. В данном автомате определен перечислительный тип данных type_sreg со всем списком допустимых значений (они, естественно, совпадают с именами, введенными в графсхеме переходов). Далее в тексте объявлены две переменные (класса **SIGNAL**) sreg и next_sreg введенного типа type_sreg. Введение двух переменных связано с желанием избежать гонок в автомате при переходе от одного состояния к другому.

Главная часть архитектурного тела содержит два оператора параллельного типа (процесса). Первый процесс запускается на исполнение каждый раз, когда происходит изменение сигналов CLK или next_state. Однако его основное действие — назначение автомату нового состояния — происходит только по переднему фронту сигнала CLK. Использование для тактирования автомата переднего фронта синхронизирующего сигнала (предложение `IF CLK='1' AND CLK'event THEN sreg<=next_sreg; END IF;`) служит для синхронизации выбранных библиотечных операционных узлов и обеспечивает стабильность входных управляющих сигналов в моменты тактирования.

Поведение управляющего автомата в тексте программы задано вторым процессом. Второй процесс запускается каждый раз, когда изменяется состояние автомата или изменяется какой-либо входной сигнал, содержимое этого процесса, и определяет поведение управляющего автомата. При составлении программы автомата учитывалась необходимость его установки в исходное состояние при подаче сигнала сброса (выражение `IF (Reset='1') THEN next_sreg<=Idle;`).

Конечные автоматы в языке VHDL удобно описывать посредством оператора выбора **CASE**, используя в качестве ключа выбора варианта переменную состояния автомата в текущий момент времени. Внутри каждого варианта определяется состояние перехода и значения выходных сигналов, формируемых в соответствии с входными условиями. Состояние перехода из текущего состояния (назначение переменной next_sreg нового значения) задается условным оператором, логическое выражение которого совпадает с последовательностью условий, встречающихся на соответствующих путях переходов на схеме алгоритма. Аналогично определяются и выходные сигналы, вырабатываемые на переходах и задающие исполняемые в других блоках операции. Основное отличие программы на языке VHDL от аналогичной программы на языке AHDL наблюдается в правилах формирования будущих состояний.

Четвертый этап. Компиляция проекта и основные параметры устройства

После создания всех фрагментов проекта и схемы проекта в целом выполняется его компиляция. Требуемый объем ОЗУ сделал целесообразным выбор в качестве БИС PLD семейства FLEX 10K. После успешной компиляции был получен файл отчета (*.rpt), показавший, что данный проект далеко не полностью использует возможности, предлагаемые самым младшим представителем семейства EPF10K10. Общие затраты БИС на реализацию проекта компилятор определил как 7%. Правда, на реализацию модуля ОЗУ компилятор использовал 33% имеющихся у БИС ресурсов.

Пятый этап. Тестирование проекта

Тестирование проекта также выполнялось средствами САПР MAX+Plus II. Созданная тестовая последовательность должна была проверять лишь ключевые моменты работы разработанного устройства. Результаты моделирования приведены на рис. 12.18. Дадим пояснения основным фрагментам моделирования.

1. Системное время в интервале от 0 до 0,1 мкс. Режим начального сброса устройства. Проверка всех возможных исходных ситуаций перед сбросом весьма громоздка и в данном примере эти варианты из соображений большого объема не включены.
2. Системное время в интервале от 0,1 до 0,3 мкс. Режим записи в ОЗУ по адресу 04 (здесь, как и далее, значения всех адресов и данных будут даны в шестнадцатиричной системе счисления) содержимого, равного 45.
3. Системное время в интервале от 0,3 до 0,55 мкс. Режим записи в ОЗУ по адресу 05 данных, равных AA.
4. Когда системное время достигло значения 0,7 мкс, начался режим записи в сдвигающий регистр содержимого ОЗУ по адресу 04. Физически запись произошла при переходе автомата sreg в состояние LdBt, т. е. приблизительно в момент времени 1,1 мкс, однако поскольку внешним признаком завершения процесса перезаписи служит появление сигнала OutReady, то именно после появления этого сигнала внешнее устройство может изменять значения адреса ОЗУ от удерживаемого ранее значения 04. Поэтому окончанием четвертого этапа тестирования можно считать момент системного времени после 1,7 мкс.

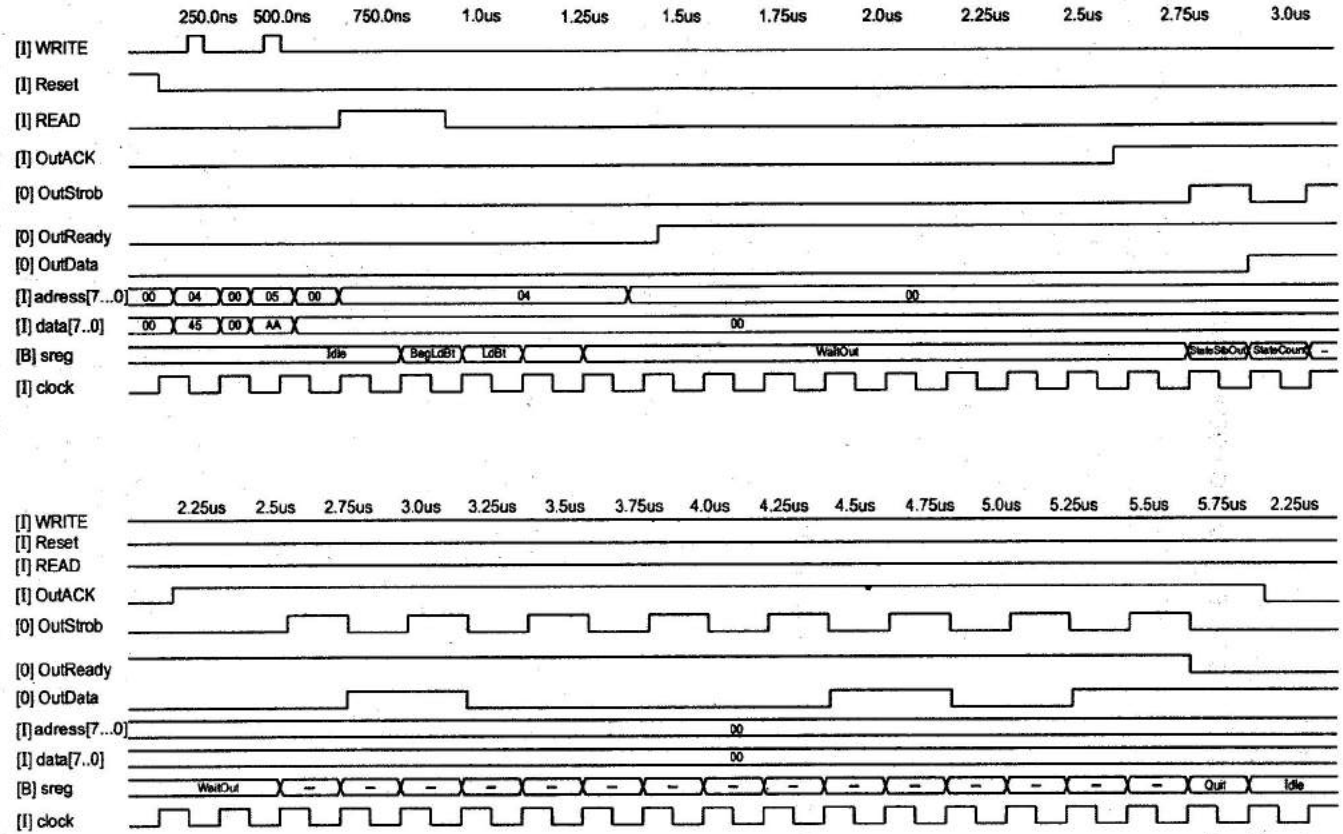


Рис. 12.18. Временные диаграммы, построенные средствами моделирования для устройства, спроектированного средствами САПР

5. В качестве следующего проверяемого фрагмента алгоритма работы устройства целесообразно выбрать вывод содержимого сдвигающего регистра в форме последовательного кода (со стороны старших разрядов) на выходной контакт OutData. Инициализирующим вывод сигналом является входной сигнал устройства OutAck, после его появления (в примере это момент времени 2,25 мкс) начинается цикл выдачи последовательного кода, каждый бит после его появления на выходном контакте через 0,2 мкс сопровождается стробирующим сигналом OutStrob. Признаком окончания цикла служит сброс сигнала OutReady (в примере это происходит после 5,7 мкс).

Анализ временных диаграмм позволяет не только проверить правильность функционирования устройства, но и исследовать временное поведение отдельных элементов проекта.

Шестой этап. Автоматическое определение временных характеристик устройства

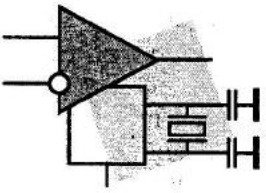
Возможности САПР вычислять временные соотношения между различными фрагментами проекта существенно облегчают проектировщику задачу проверки правильности работы проекта во временной области. Автоматизация этого этапа проектирования избавляет от необходимости ручного перебора исходных данных проекта с целью обнаружения отклонений от допустимых временных установок.

Седьмой этап. Практическое использование результатов проектирования

Основным результатом работы компилятора является файл конфигурации БИС, соответствующий техническому заданию. Средства САПР позволяют на заключительных этапах работы поместить содержимое этого файла в интересующую проектировщика БИС, и на этом процесс проектирования может быть переведен в плоскость натурных экспериментов. Эксперименты могут производиться как с конечной продукцией, так и с прототипными средствами различных типов.

Литература к главе: [4], [6], [8], [16], [21], [27], [37].

Приложение 1



Сведения о некоторых популярных восьмиразрядных микроконтроллерах

Ежегодно в мире появляется много новых микропроцессоров и микроконтроллеров. Это не означает быстрой смены их архитектур, разнообразие вариантов объясняется главным образом широкими диапазонами их параметров (быстродействия, потребляемой мощности и т. д.). Несмотря на относительную стабильность архитектур, справочные данные по микропроцессорам и микроконтроллерам весьма обширны и здесь не приходится ставить задачу их подробного освещения. Исключая high-end-варианты высшей производительности и среднюю зону, ориентированную на поддержку изощренных операционных систем, остановимся лишь на единичных представителях простых восьмиразрядных микроконтроллеров, все более снабжаемых периферийными устройствами, интегрированными на кристалле, что и ценно для применения микросхем в системах управления техническими объектами. Большая доля рынка занята именно этими вариантами. Взятые примеры трех восьмиразрядных микроконтроллеров (табл. П1) никоим образом не претендуют на полноту оценки продукции фирм, работающих в области создания микропроцессоров и микроконтроллеров.

Таблица П1

Параметр	Микросхема		
	MCS 518 (Intel)	AVR (Atmel)	PIC16C (Microchip)
Частота процессора, МГц	12, 16, 24 или 33	1–24	0–20
Разрядность шин, бит (адрес/данные)	8	16/8	14/8
Разрядность команды, бит	8	16	12/14

Таблица П1 (окончание)

Параметр	Микросхема		
	MCS 518 (Intel)	AVR (Atmel)	PIC16C (Microchip)
Рабочее напряжение, В	4,5–5,5 (допуск 10%)	1,8–5,5	2–6
Ток или мощность, потребляемые на максимальной частоте	24 мА	1–3 мА	50 мВт
Режимы понижения мощности	Покоя, глубокого понижения	Меньше 1 мкА	Спящий, контроль переключений на входах
Аппаратные умножители	8×8	8×8 или 16×16	–
Память	8, 16 или 32 Кбайт EPROM, OTP, ROM; 256–512 байт RAM	1–128 Кбайт флэш-памяти; 64 байта–4 Кбайт EEPROM; 128 байт–4 Кбайт SRAM	768 байт–14 Кбайт ROM/OTP/Flash; 68–368 байт SRAM; 64–256 байт EEPROM
Корпуса	44 PLCC/PDIP/MQFP	8/20/28/40 DIP; 32/44/64 TQFP; 20 SSOP; 8/20 SOIC	14–44 DIP/SOIC/SSOP/ PLCC/TQFP/MQFP/CE RDI/QFN
Таймеры	Три 16-разрядных, сторожевой	Два 8-разрядных, два 16-разрядных	Несколько 8/16-разрядных с режимами ШИМ и захвата/сравнения
Интерфейс	Дуплексный UART/последовательный I/O	SPI, UART, I^2C, 8-разрядный параллельный порт	USART, I^2C, SPI, LIN, MI^2C, 13–33 GPIOS, USB
Прерывания	8	27 внутренних, 8 внешних	4–12
АЦП, ЦАП	–	8-канальный 10-разрядный АЦП	4–8-канальный 8/12-разрядный АЦП, 8-разрядный ЦАП
Дополнительные возможности	–	JTAG, средства отладки	ISP, индикация низкого напряжения, внутренний генератор, аналоговый компаратор, источник опорного напряжения, операционный усилитель
Стоимость, USD (в лотах 10 000 шт.)	1,61–5,48	0,5–4	1,10–5,50

Приложение 2

Функциональные ячейки библиотеки БМК

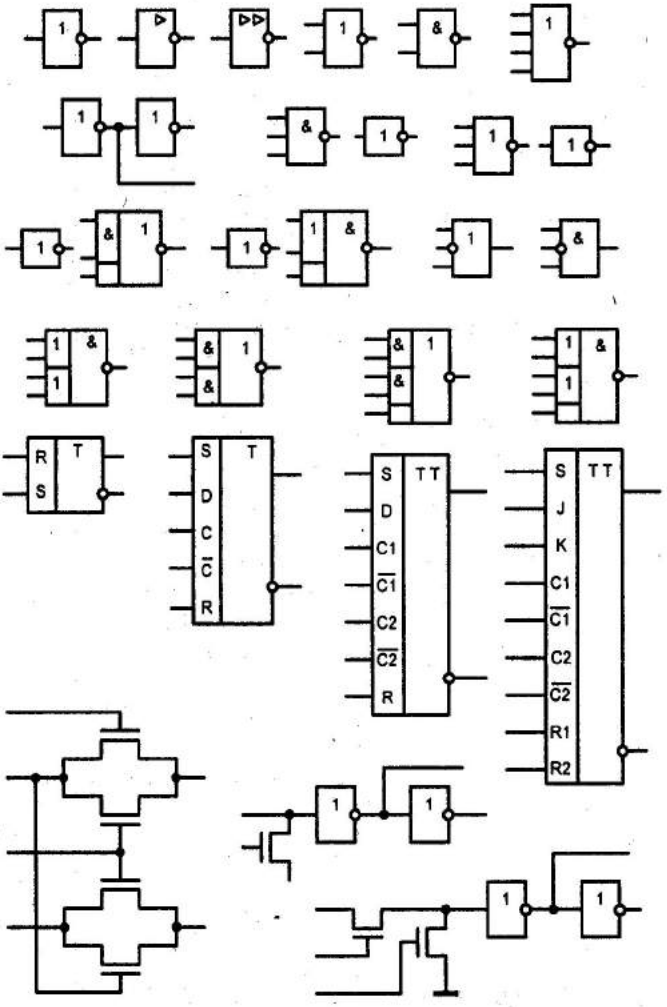
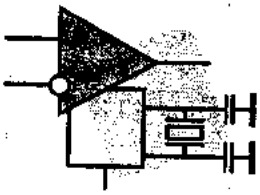


Рис. П2. Библиотека функциональных ячеек базового матричного кристалла 1815ХМ1

Приложение 3



Обзор продукции основных производителей микросхем программируемой логики

Производство микросхем с программируемой структурой растет быстрее, чем остальные секторы рынка логических схем. Это является устойчивой тенденцией. Микросхемы с программируемыми структурами выпускаются следующими фирмами: Actel, Altera, Anachip, Atmel, Cypress Semiconductor, Lattice Semiconductor, Leopard Logic, QuickLogic, STMicroelectronics, Triscend, Xilinx. Выбирая микросхемы, стоит обратить внимание на продукцию разных фирм, однако в кратком обзоре ограничимся ознакомлением с продукцией только основных производителей.

П.3.1. Однократно программируемые микросхемы

Однократно программируемые микросхемы представлены на рынке продукцией фирм Actel и QuickLogic. Эти микросхемы с программированием перемычек типа antifuse имеют невысокую стоимость, высокое быстродействие, повышенную стойкость к воздействию радиации и высшую степень засекречивания проекта. В то же время единожды сконфигурированную схему в дальнейшем изменить нельзя. По своей архитектуре микросхемы с перемычками типа antifuse относятся к классу FPGA.

Фирма Actel выпускает семейства eX, SX-A, MX и микросхемы с повышенной стойкостью к воздействию радиации RH (Radiation Hardened) и RT (Radiation Tolerant). Семейства eX и SX-A имеют напряжение питания 2,5 В, семейство MX — 5 В. Максимальная системная частота для представителей этих семейств составляет приблизительно 250 МГц. Особенностью микросхем семейства eX является весьма малая потребляемая мощность. Параметры микросхем приведены в табл. П3.1.

Таблица П3.1

Семейство	Системных вентилей, тыс.	Триггеров	Логических ячеек	Максимальное число выводов пользователя	Стоимость, USD*
eX	3–12	64–256	128–572	84–132	2,3–6,3
SX-A	12–108	256–2012	768–6036	130–360	4,4–23,1
MX	3–54	147–1822	295–2438	57–202	2,9–23

* Стоимость указывается для самых дешевых вариантов микросхем, поставляемых в партиях порядка 10 тыс. шт.

Недавно фирмой Actel выпущено семейство Axcelerator, имеющее напряжение питания 1,5 В, повышенную логическую сложность и рабочие частоты до 500 МГц при стоимости микросхем более $120.

Фирма Actel выпускает также семейства ProASIC и ProASIC Plus с логической сложностью уровня SOPC, флэш-памятью конфигурации и стоимостью микросхем $10—15.

Фирма QuickLogic выпускает семейства pASIC2, pASIC3 и Eclipse. Большинство изготовляемых фирмой микросхем имеют логическую сложность в пределах от 3 до приблизительно 100 тысяч эквивалентных вентилей, исключение составляет семейство Eclipse с логической сложностью до 580 тысяч эквивалентных вентилей. Это же семейство отличается широкими возможностями буферов ввода/вывода данных, поддерживающими много стандартов интерфейса, в том числе и для дифференциальных передач, а также большим объемом встроенных блоков двухпортовой памяти RAM. Кроме того, фирма выпускает блочные SOPC, не содержащие процессорных ядер *(см. главу 10)*.

П.3.2. Микросхемы с программированием состояний плавающих затворов

Микросхемы с программированием состояний плавающих затворов (памятью конфигурации EEPROM, FLASH) по архитектуре относятся как правило к CPLD. Они более всего приспособлены для реализации относительно несложных устройств высокого быстродействия (интерфейсных схем, автоматов управления и т. п.). Пионер в разработке CPLD — фирма Altera — представлена семейством MAX7000, его упрощенным вариантом MAX3000A и семейством MAX9000. Микросхемы MAX3000A отличаются низкой стоимостью (они проходят выходной контроль по нормам, менее жестким, чем микросхемы MAX7000). Семейство MAX9000 имеет повышенную логиче-

скую сложность (12 тысяч макроячеек против 5 тысяч у старшего представителя семейства MAX7000).

На основе архитектуры MAX7000 разработаны CPLD Ultra37000 (фирма Cypress Semiconductor) и ATF1500 (фирма Atmel). Фирма Xilinx также поучаствовала в выпуске CPLD, разработав, в частности, семейства CoolRunner и CoolRunner-II, отличающиеся сверхмалым потреблением мощности. Микросхемы CPLD производятся также фирмами Lattice Semiconductor (ispLSI5000VE, ispLSI8000V) и др.

Сведения о некоторых микросхемах с памятью конфигурации типов EEPROM и FLASH приведены в табл. П3.2.

Таблица П3.2

Семейство	Число эквивалентных вентилей	Число макроячеек	Число макроячеек в блоке/ число входов блока	Время распространения сигнала от вывода к выводу, нс	Число пользовательских выводов	Стоимость, USD
MAX 3000A	600–5000	32–256	16/36	4,5–6,0	34–158	1–8
MAX 7000B	600–10000	32–512	16/36	3,5–5,5	36–212	1,40–29
CoolRunner II	–	32–512	16/56	3,5–6,0	33–270	1,15–39,50

Представителем семейства MAX7000 в табл. П.3.2 выбрана микросхема с литерой "B", отличающаяся высоким быстродействием и высоким уровнем ряда других параметров. Микросхемы CoolRunner-II фирмы Xilinx с ультранизким потреблением мощности имеют напряжение питания 1,8 В и статический ток не более 100 мкА.

П.3.3. Микросхемы с триггерной памятью конфигурации

Оперативно репрограммируемые микросхемы с триггерной памятью конфигурации разнообразны по архитектуре и логической сложности. По архитектуре это, как правило, FPGA и комбинированные структуры, а по логической сложности — микросхемы, содержащие от нескольких тысяч до нескольких миллионов эквивалентных вентилей.

Архитектурные основы FPGA были заложены в структурах микросхем XC4000 фирмы Xilinx. За ними последовали семейства VirtexE/EM, VirtexII и VirtexIIPro типа "система на кристалле". Старшие представители семейства Virtex-II отличаются высокой логической сложностью, соответственно кото-

рой подобные микросхемы относят к схемам типа Platform IC, имея в виду, что они становятся основной частью всего проектируемого устройства. Микросхемы семейства Virtex-II могут работать на внутренних частотах до 420 МГц и обеспечивать скорость обмена данными до 840 Мбит/с на один контакт ввода/вывода. Фирмой Xilinx разрабатывается и линия микросхем Spartan. В ее составе семейства Spartan, Spartan-II меньшего уровня интеграции в сравнении с микросхемами линии Virtex и семейство Spartan-3, которое характеризуется как "FPGA уровня платформы", оно было первым, выпущенным по технологии 90 нм (в области программируемой логики), отличается высоким быстродействием (способно работать на частотах свыше 300 МГц) и малым соотношением стоимость/вентиль. Сведения о параметрах новых микросхем фирмы Xilinx приведены в табл. П3.3.

Таблица П3.3

Семейство	Логических элементов	Системных вентилей, тыс.	Конфигурируемых логических блоков	Выводов пользователя	Дифференциальных выводов	Емкость распределенной памяти, бит	Емкость встроенной памяти, бит	Стоимость, USD
Spartan-3	1728—74880	50—5000	190—8320	124—784	56—344	12K—520K	0—1872K	—
Virtex II	576—104882	40—8000	64—11648	88—1108	—	8K—1456K	72K—3024K	514—3900

Триггерная память конфигурации используется и в микросхемах с архитектурой смешанного типа, разрабатываемых в первую очередь фирмой Altera. Ее микросхемы ACEX 1K средней сложности имеют невысокую стоимость. Семейства APEX 20K/KE/KC и APEX II принадлежат к сложным СБИС ПЛ (наряду с семействами Virtex, VirtexII и VirtexIIPro фирмы Xilinx). Семейство Mercury ориентировано на применение в высокопроизводительных системах цифровой обработки сигналов и сложных коммуникационных системах. Для этого большое внимание уделено как полосе пропускания каналов ввода/вывода, так и быстродействию самого ядра СБИС. Cyclone — семейство, разработанное после сложных микросхем Mercury и Stratix с целью получения удешевленного и упрощенного (но все же в 4—5 раз более сложного, чем семейство ACEX 1K) варианта для реализации в первую очередь операционных устройств (устройств типа Datapath). Технологический уровень характеризуется проектной нормой 0,13 мкм, и медными соединениями во всех слоях.

Семейство Stratix высокой сложности и быстродействия имеет до 28 блоков цифровой обработки сигналов, систему быстрых передач, включающую в

себя, в частности, линии низковольтных дифференциальных связей LVDS и др. Микросхемы семейства Stratix перспективны для использования в составе сложных устройств.

Сведения о микросхемах комбинированной архитектуры и FPGA фирмы Altera приведены в табл. П3.4.

Таблица П3.4

Семейство	Системных вентилей, тыс.	Типичное число вентилей, тыс.	Логических элементов	Системных блоков (EAB, ESB)	Общая емкость памяти, бит	Число выводов пользователя	Стоимость, USD
ACEX 1K	56–257	10–100	576–4992	3–12	12228–49152	130–333	5–17
APEX 20K	113–526	30–200	1200 (192)*–8320 (832)*	12–52	24576–106496	192–832	27–185
APEX II	1900–7000	600–4000	16640–89280	104–372	425984–1523712	492–1140	160–1070
Mercury	–	120–350	4800–14400	12–28	49152–114688	303–486	100–230
Cyclone	–	–	2910–20060	–	59904–294912	104–301	–
Stratix	–	–	10570–114140	–	920448–10118016	–	100–1600

* Так как микросхемы семейства APEX 20K имеют функциональные ресурсы как в виде логических элементов, так и в виде макроячеек, в таблице без скобок приведены числа логических ячеек, а в скобках – числа макроячеек.

Для SOPC с hard-ядрами сведения табличного характера недостаточны, поскольку представление о них можно получить только при рассмотрении структуры в целом. Такие сведения даны в *главе 10*. В дополнение отметим, что нижний уровень стоимости несложных "систем на кристалле" (типа E5 фирмы Triscend или FPSLIC фирмы Atmel) составляет $5–7. Стоимость сложных SOPC (например, Excalibur) составляет от нескольких десятков до нескольких сотен долларов.

П.3.4. Пример более подробного описания микросхемы программируемой логики

Далее рассмотрено несколько подробнее семейство ACEX 1K, популярное у отечественных разработчиков аппаратуры широкого применения (для разработки спецаппаратуры больше подходят, в частности, микросхемы фирм Xilinx, Actel).

В семейство ACEX 1K входят четыре микросхемы. Напряжение питания микросхем 2,5 В. Микросхемы могут принимать сигналы от схем с напряжениями питания 2,5 В, 3,3 В и 5,0 В и быть источниками сигнала для схем с такими же напряжениями питания.

Стоимость микросхем (самых дешевых вариантов в лотах по 10 тыс. шт.) лежит в пределах от $5 до $17.

Параметры семейства:

- 10, 30, 50 и 100 тысяч типичных вентилей;
- диапазон числа логических элементов 576—4992. Основными частями логических элементов являются четырехвходовый LUT-блок, программируемый триггер и цепи переноса и каскадирования;
- микросхемы имеют от трех до двенадцати встроенных блоков памяти EABs с размером каждого блока 4 Кбит и общей емкостью от 12228 до 49152 бит;
- микросхемы имеют 1—2 встроенных блока PLL;
- задержки распространения сигналов в цепях "синхронизация—выход" таковы, что допускают тактирования ввода/вывода на частотах до 250 МГц, системная частота микросхем, несмотря на их невысокую стоимость, превышает 100 МГц.

Более определенная оценка функциональных возможностей и быстродействия микросхем семейства содержится в данных табл. П3.5 с указанием рабочих частот и потребляемых ресурсов для некоторых типовых функциональных устройств, полученных при компиляции средствами САПР без каких-либо дополнительных вмешательств с целью улучшения проекта.

Как и другие современные ПЛИС, микросхемы семейства ACEX 1K поддерживают технологию многовольтового ввода/вывода (Multivolt I/O), программирования в системе (ISP) по интерфейсу JTAG, граничного сканирования по методике JTAG, обеспечивается совместимость микросхем с шиной PCI (64 разряда, 66 или 33 МГц). Поддерживается возможность встраивания в схему soft-ядра процессора Nios.

Таблица П3.5

Устройство	Используемые ресурсы		Быстродействие для градации скорости			Единицы измерения
	LEs	EABs	-1	-2	-3	
16-разрядный счетчик с параллельной загрузкой или 16-разрядный аккумулятор	16	0	200	188	128	МГц
Мультиплексор "16 на 1"	10	0	3,2	4,3	5,5	нс
16-разрядный умножитель с трехступенчатым конвейером	544	0	93	86	64	МГц
RAM с организацией 256×16, операции чтения	0	1	212	181	131	МГц
RAM с организацией 256×16, операции записи	0	1	142	128	94	МГц

Потребляемая микросхемами мощность практически пропорциональна тактовой частоте. Конкретные цифры зависят от реализованного в микросхеме проекта, для ориентировочной оценки фирма приводит следующие данные о потребляемых микросхемами токах для частоты тактирования 100 МГц (микросхемы работают в типичных условиях при ненагруженных выходах):

- для микросхемы EP1K10 — 100 мА;
- для микросхемы EP1K50 — 160 мА;
- для микросхемы EP1K100 270 мА.

Для проектирования может быть использовано программное обеспечение САПР MAX+Plus II.

По оценке фирмы семейство ACEX 1K представляет собою эффективные по стоимости средства для реализации проектов достаточно большой тиражности. Имеющийся набор функциональных блоков позволяет воспроизводить сложные логические и регистровые функции, например, такие как ЦОС. Особо отмечается такая область применения микросхем, как быстродействующие коммуникационные системы.

Глоссарий

2D — структура ЗУ с однокоординатной выборкой слов путем возбуждения линии выборки от дешифратора адреса.

2DM — структура ЗУ (модификация структуры 2D), в которой слова выбираются поэтапно — вначале выбираются "длинные" слова с помощью дешифрации одной части адреса, а затем из них слова нужной разрядности с помощью дешифрации другой части адреса.

3D — структура ЗУ с двухкоординатной выборкой запоминающих элементов на пересечении двух линий выборки, возбуждаемых выходами двух дешифраторов адреса.

x86 — семейство процессоров фирмы Intel, совместимых по системе команд: 8086, 80186, 80286, 80386, 80486, Pentium—Pentium 4.

A

Автомат Мили — автомат с памятью, выходные сигналы которого зависят как от его состояния, так и от входных сигналов.

Автомат Мура — автомат с памятью, выходные сигналы которого зависят только от состояния автомата.

Адресация абсолютная — адресация, при которой ячейке памяти или внешнему устройству соответствует один-единственный адрес.

Адресация неабсолютная — адресация, при которой ячейке памяти или внешнему устройству соответствует некоторая зона адресов.

Адресное ЗУ — ЗУ, в котором доступ к единицам хранения информации осуществляется по их адресу (местоположению в памяти).

Адресное пространство — диапазон адресов, к которым может обращаться процессор.

Аналоговое моделирование — моделирование, опирающееся на действительные значения параметров и сигналов в устройствах и системах (токов и напряжений в электронных схемах) в отличие от моделирования с упрощенным дискретным представлением этих параметров и сигналов.

Асинхронность — поведение (свойство) устройств и процессов, заключающееся в обработке ими информации (сигналов) по мере их поступления без участия внешних тактирующих (исполнительных, командных) сигналов.

Асинхронный сигнал — сигнал, моменты появления и исчезновения которого не привязаны к синхросигналам устройства (системы).

Асинхронные установочные входы — входы сброса и установки триггеров, действие которых не зависит от тактирования и доминирует над воздействиями других входов.

Ассоциативное ЗУ (CAM, Content Addressable Memory) — ЗУ, в котором доступ к единицам хранения информации осуществляется не по их адресу, а по специальному признаку (ключу).

Б

Базовый матричный кристалл (БМК) — полузаказная БИС/СБИС, содержащая нескоммутированные схемные элементы, основа для создания требуемого устройства путем реализации межсоединений элементов методом масочного программирования металлизации.

Бесканальный БМК — базовый матричный кристалл, внутренняя область которого сплошь заполнена базовыми ячейками и не содержит свободных каналов, заранее отведенных для трассировки (этот тип БМК называют кристаллами типа "море вентилей" или "море транзисторов").

БМК блочной структуры — базовый матричный кристалл, содержащий специализированные области (логической обработки, памяти, реализации отдельных операций и т. п.).

Библиотека функциональных ячеек — совокупность функциональных ячеек, используемых при проектировании на основе БМК, создается при его разработке.

Быстрый страничный доступ (FPM, Fast Page Mode) — ускоренный доступ к данным в динамических ЗУ, возможный при условии "кучности" их адресов, когда запрашиваемые данные принадлежат одной и той же странице (строке матрицы запоминающих элементов).

В

Вектор прерывания — сведения о местоположении в памяти подпрограммы обслуживания данного прерывания, пересылаемые в процессор источником запроса прерывания или контроллером прерываний.

Векторное прерывание — прерывание, для обслуживания которого требуется передать в процессор вектор прерывания.

Вентильная матрица (ВМ) — синоним понятия БМК (см. ранее).

Верификация — проверка по формальным методикам совпадения проектной спецификации и результатов синтеза устройства.

Вес кодовой комбинации — число единиц в разрядах данной комбинации.

Видеопамять — ЗУ с последовательным цикличным доступом к словам и периодом цикла, соответствующим процессу сканирования монитора электронными лучами.

Виртуальные компоненты (Soft-ядра) — файлы конфигурирования областей кристаллов программируемой логики, обеспечивающие реализацию в этих областях заданных функций.

Витая пара — одна из распространенных конструкций линий передачи сигналов, представляющая собою два скрученных провода.

Внешняя синхронизация — процесс координации событий в схеме с состояниями тактирующей системы, не принадлежащей самой схеме.

Внутрикристальная отладка — методы отладки, базирующиеся на размещении внутри отлаживаемой ИС средств тестирования и их соединения с основным тестирующим оборудованием специальными протоколами.

Внутрисхемная эмуляция — объединение ресурсов отлаживаемой системы с ресурсами отладочной системы.

Волновое сопротивление — параметр линии передачи сигналов, трактуемой как "длинная линия".

Временная диаграмма — графическое представление сигналов электрических схем, показывающее, как изменяются сигналы во времени и как они взаимно соотносятся.

Временное моделирование — моделирование электрических схем с учетом действительных временных соотношений между входными и выходными сигналами всех элементов схемы.

Время выдержки (Hold Time) — (1) для триггера — интервал времени после поступления синхросигнала, в течение которого входные информационные сигналы должны оставаться неизменными; (2) в более общем смысле для двух сигналов A и B это интервал времени между началом сигнала A и окончанием сигнала B (это время называют также временем удержания).

Время предустановки (Set-Up Time) — (1) для триггера — интервал времени до поступления синхросигнала, в течение которого входные информационные сигналы должны оставаться неизменными; (2) в более общем смысле для двух сигналов A и B это интервал времени между началом сигнала A и началом сигнала B.

Г

Гарвардская архитектура процессора — архитектура с раздельной памятью программ и данных.

Градация быстродействия — часть маркировки микросхем, отображающая их принадлежность к той или иной группе по признаку быстродействия (обычно это цифра после дефиса).

Граничное сканирование (Boundary Scan Testing) — тестирование БИС/СБИС по интерфейсу JTAG.

Д

Двоичный дешифратор — устройство, преобразующее двоичный код в код "1 из N".

Двоичный счетчик — счетчик, модуль счета которого равен целой степени числа 2, а состояния кодируются двоичными числами.

Двунаправленный вывод — вывод, который в зависимости от программирования может быть использован как вход или выход микросхемы.

Двухпортовое ЗУ — ЗУ, в котором возможны одновременное чтение по одному адресу и запись по другому.

Декомпозиция — процедура сведения задачи верхнего уровня иерархии к группе более простых задач низшего уровня.

Демультиплексор — устройство, передающее входную величину в один из нескольких выходных каналов в зависимости от адресующего входного кода.

Динамическая реконфигурация (Run-Time Reconfiguration) — быстрая смена настроек в схемах программируемой логики, ориентированных на использование в аппаратуре с многофункциональным использованием одних и тех же ИС.

Длинная линия — (1) линия, время распространения сигнала в которой соизмеримо с длительностью фронтов передаваемых импульсов, что требует согласования волновых сопротивлений в тракте передачи сигналов; (2) непрерывная линия межсоединений, проходящая по всей длине или ширине кристалла БИС/СБИС программируемой логики для быстрой передачи сигналов на большие расстояния.

ДНФ — дизъюнктивная нормальная форма представления логической функции, дизъюнкция конъюнктивных термов.

ДОЗУ (DRAM) — динамическое оперативное ЗУ, запоминающими элементами которого являются конденсаторы.

Дребезг контактов — последствия упругих свойств механических контактов, приводящие к появлению серий переключений вместо одного при однократном изменении положения контакта.

З

Защелка (Latch) — термин, применяемый к триггерам и регистрам, имеющим режим прозрачности и пропускающим сигналы со входа на выход при разрешающем уровне тактирующего сигнала.

Зернистость (Granularity) — характеристика логических блоков БИС/СБИС программируемой логики, связанная со степенью их сложности.

И

Информационная емкость ЗУ — максимальный объем хранимой ЗУ информации.

Интерфейс — совокупность аппаратных и программных средств, унифицирующих процессы обмена между модулями системы.

Интерфейс с общей шиной — интерфейс, в котором адреса ячеек памяти и внешние устройства имеют общее адресное пространство.

Интерфейс с раздельной шиной — интерфейс, в котором для адресов внешних устройств имеется отдельное адресное пространство.

Интерфейс с двойной скоростью передач (DDR-технология) — организация передач по линиям связи, при которой операции приема (выдачи) информации осуществляются обоими фронтами синхросигналов.

Интерфейс с учетверенной скоростью передач (QDR-технология) — термин, применяемый для обозначения запоминающих устройств, в которых DDR-технология сочетается с двухпортовостью.

Исполняемая спецификация — описание проекта на высших уровнях абстракции, допускающее моделирование в САПР.

К

Канал трассировки — свободная зона на кристалле БМК, выделенная для реализации межсоединений ячеек.

Канальный БМК — базовый матричный кристалл, в конструкции которого предусмотрены определенные каналы трассировки.

Клонирование проектов — несанкционированное копирование проектов без осознанного раскрытия их внутренней структуры и принципов работы.

Код — совокупность кодовых комбинаций, используемых для представления информации. Этот же термин используется в качестве синонима понятия "кодовая комбинация" в тех случаях, когда это не может вызвать каких-либо недоразумений.

Код "1 из N" — код, в кодовых комбинациях которого один разряд активен, а все остальные пассивны. Кодирование этим способом в английской терминологии именуется ONE, One-Hot Encoding. Активным может считаться значение логической 1 или логического 0.

Код Грея — код, в котором соседние кодовые комбинации отличаются друг от друга только в одном разряде.

Код Хемминга — код, кодовые комбинации которого содержат несколько контрольных разрядов для проверки на четность/нечетность весов определенных групп разрядов. Обладает свойствами не только обнаружения, но и исправления ошибок единичной кратности.

Кодовая комбинация — набор из символов принятого алфавита.

Командный цикл — интервал времени, соответствующий выполнению одной команды программы.

Комбинационная цепь — схема, установившиеся значения выходных сигналов которой зависят только от текущих значений входных сигналов.

Компаратор (цифровой) — устройство, определяющее отношения между двумя словами.

Компиляция — этап автоматизированного проектирования электронных устройств, заключающийся в их синтезе.

Конвейеризация — способ повышения частоты тактирования в тракте обработки данных, для реализации которого комбинационные цепи тракта разбиваются на ступени.

Конвертация проектов — перевод проектов на реализацию на других средствах. Наиболее широко используются переходы от ПЛИС к БМК и от ПЛИС к реализациям на основе стандартных ячеек.

Контроллер ПДП — контроллер прямого доступа к памяти, устройство, управляющее обменом данными между памятью и внешними устройствами без участия процессора.

Контроль по четности/нечетности — контроль с проверкой четности/нечетности веса кодовых комбинаций. Обладает свойством обнаружения ошибок единичной кратности.

Контрольный разряд — дополнительный разряд, вводимый в информационное слово для обеспечения четности/нечетности его веса или веса отдельных групп разрядов при контроле по модулю два или с помощью кода Хемминга.

Конфигурируемый логический блок (Configurable Logic Block) — логический блок микросхем программируемой логики, настраиваемый (программируемый) на воспроизведение требуемых функций.

Конфигурирование ИС — настройка ИС с программируемой структурой на выполнение определенных функций с помощью задания замкнутых или разомкнутых состояний программируемым элементам связей.

Коэффициент отражения — отношение амплитуды отраженной волны к амплитуде падающей волны в концах длинной линии.

Коэффициент разветвления — число входов, которые могут быть подключены к одному выходу в схеме из однотипных элементов. Максимально допустимый коэффициент разветвления определяется паспортом элемента.

Кратность ошибки — число неверных разрядов в данной кодовой комбинации.

Кратчайшая ДНФ — дизъюнктивная нормальная форма представления переключательной функции, содержащая минимальное число конъюнктивных термов.

Критический путь — маршрут распространения сигнала в устройстве (схеме), определяющий его (ее) максимально возможное быстродействие.

Кэш-память — особо быстродействующая память, хранящая копии информации, используемой в текущих операциях обмена с процессором.

Кэш-память наборно-ассоциативного типа — вариант кэш-памяти, промежуточный относительно вариантов с полной ассоциацией и прямым размещением.

Кэш-память с полной ассоциацией — ассоциативная кэш-память с произвольной загрузкой данных.

Кэш-память с прямым размещением — кэш-память, в которой одна или несколько страниц основной памяти строго соответствуют одной строке кэш-памяти.

Кэш первого уровня (L1) — внутрипроцессорная кэш-память, размещенная на одном кристалле с процессором.

Кэш второго уровня (L2) — кэш-память, расположенная вне кристалла, на котором размещен процессор. Емкость кэш-памяти второго уровня, как правило, превышает емкость кэш-памяти первого уровня.

Л

ЛИЗМОП — МОП-транзистор с лавинной инжекцией заряда. Имеет "плавающий затвор", т. е. изолированную область над каналом, в которой можно создавать или не создавать электрический заряд, отображая тем самым логические состояния 1 и 0. Кроме того, может иметь или не иметь обычный управляющий затвор (варианты "с плавающим затвором" и "с двойным затвором").

Литерал — литерал логической переменной, т. е. либо сама переменная, либо ее инверсия.

Логический анализатор — устройство позволяющее наблюдать или фиксировать поведение группы цифровых сигналов в отлаживаемой системе.

М

Магистрально-модульная структура — структура микропроцессорной системы, в которой к одним и тем же шинам подключаются различные модули.

Мажоритарный элемент — логический элемент с нечетным числом входов, выходная величина которого определяется тем, какие сигналы (0 или 1) составляют большинство среди входных сигналов.

Маскирование запросов — воздействие на сигналы запросов прерывания, прямого доступа к памяти и др., запрещающее обслуживание этих запросов.

Масочное программирование — запись данных в ПЗУ или задание межсоединений в БМК, осуществляемые при производстве кристаллов методами интегральной технологии (с помощью шаблонов металлизации).

Матричная базовая ячейка — базовая ячейка внутренней области БМК, предназначенная для реализации на ее основе функциональных ячеек.

Машинный цикл — интервал времени, составляющий часть командного цикла, соответствующий в основном обращению процессора к памяти или внешнему устройству и передаче байта (слова) в процессор или из него.

Метастабильное состояние — аномальное состояние триггера, в котором он длительное время находится вблизи равновесного состояния. Вызывается нарушением условий предустановки и выдержки информационных сигналов относительно тактирующего или другими факторами, вводящими триггер в режим, близкий к равновесному (симметричному).

Микроконтроллер — однокристальная микроЭВМ, ориентированная на выполнение относительно простых алгоритмов управления техническими объектами и технологическими процессами.

Микропроцессор — реализованное на одном или нескольких кристаллах программно-управляемое устройство, осуществляющее процесс обработки информации и управление им.

Микропроцессорный комплект БИС — набор микросхем, пригодных для совместного применения при построении микропроцессорной системы.

Микропроцессорная система — система, в которой реализован законченный процесс выполнения заданной программы, содержащая в качестве основных блоков (модулей) процессор, память, внешние устройства и интерфейсные схемы.

Минимальное кодовое расстояние — минимальное кодовое расстояние между двумя любыми кодовыми комбинациями, принадлежащими данному коду.

Минимизация логических функций — такое преобразование логических функций, которое упрощает их в смысле заданного критерия.

МНОП — транзистор со структурой "металл-нитрид-оксид-полупроводник", в котором при программировании можно создавать или устранять заряд на границе слоев "нитрид-оксид", отображая тем самым логические состояния (0 и 1).

Моделирование — воспроизведение в *модельном времени* поведения модели (математической или программной), соответствующего поведению целевой системы в *системном времени*.

Моделирование с единичной задержкой (Unit Delay) — простейшая форма временного моделирования цифровых схем, при которой задержки всех элементов считаются одинаковыми и равными модельной единице.

Модуль счета — число состояний, которое может иметь счетчик, т. е. емкость счетчика.

Мультиплексирование — передача сигналов от разных источников по одним и тем же линиям в режиме разделения времени.

Мультиплексор — схема, передающая на выход одну из нескольких входных величин под управлением адресующего кода.

Н

Нагрузочная способность — параметр микросхемы, определяемый величинами допустимых выходных токов в состояниях логического нуля и единицы. Значение максимально допустимой емкостной нагрузки обычно задается отдельно.

О

Однофазная синхронизация — система синхронизации, в которой на все элементы памяти (триггеры) подаются одни и те же тактирующие сигналы.

Операция монтажной логики — логическая операция, реализуемая путем соединения в одной точке выходов нескольких логических элементов с открытым коллектором или эмиттером.

Организация ЗУ — параметр ЗУ, выражаемый произведением максимально возможного числа хранимых слов на их разрядность.

Основная память — память, работающая в режиме оперативного обмена данными с процессором и, в отличие от кэш-памяти, хранящая весь объем

требуемых для этого данных. В ЭВМ в качестве основной используется, как правило, память динамического типа.

Открытый коллектор — тип выходной цепи логических элементов, один из вариантов выходных цепей, допускающих подключение к магистрали. Может быть использован для реализации операций монтажной логики.

Отрицательный фронт — перепад сигнала при его переходе от высокого уровня к низкому.

П

Параллельный периферийный адаптер (PPI, Parallel Peripheral Interface) — устройство, обслуживающее обмен параллельными данными между процессором и внешними устройствами.

Передний фронт — перепад сигнала при его переходе от пассивного уровня к активному. Для H-активного сигнала передним является положительный фронт, для L-активного — отрицательный.

Перекрестная помеха — помеха, порождаемая взаимным влиянием близлежащих сигнальных линий.

Периферийное сканирование — синоним термина "Граничное сканирование", в этой книге не употребляется.

Поведенческая модель — высокоуровневое представление электронного проекта, которое описывает поведение различных модулей или подсистем проекта (обычно без учета технологии используемой при его реализации).

Полиномиальный счетчик — сдвигающий регистр с линейными обратными связями, т. е. связями, реализованными с помощью элементов сложения по модулю два. Используются в качестве генераторов псевдослучайных последовательностей.

Полностью заказная БИС/СБИС — микросхема, которая целиком проектируется по конкретному заказу и изготовляется с помощью индивидуального набора фотошаблонов для всех этапов процесса производства.

Полузаказная БИС/СБИС — микросхема, которая реализуется с использованием стандартного полуфабриката (БМК), требуемое функционирование которого обеспечивается индивидуальными операциями только на заключительных этапах процесса производства. Для изготовления такой микросхемы нужен существенно уменьшенный набор фотошаблонов (в сравнении с требованиями изготовления полностью заказных БИС/СБИС).

Пороговое напряжение — значение входного напряжения элемента, переход через который трактуется как переход сигнала из одного логического состояния в другое.

Порождающая функция — функция, реализуемая настраиваемым логическим модулем, когда все его входы используются как информационные, т. е. для подачи на них аргументов.

Порт тестирования (Test Access Port) — четыре (или пять) специально выделенных для тестирования по интерфейсу JTAG вывода БИС/СБИС.

Последовательные репрограммируемые ЗУ — репрограммируемые запоминающие устройства типов EEPROM и Flash с последовательным интерфейсом, принимающие и выдающие команды и данные по одноразрядным линиям.

Принстонская архитектура процессора (архитектура фон Неймана) — архитектура с общей памятью команд и данных.

Приоритетный шифратор — устройство, вырабатывающее двоичный номер старшего из имеющихся на входах запросов (прерывания, прямого доступа к памяти и др.).

Программируемая логическая матрица (PLA, Programmable Logic Array) — микросхема для реализации системы переключательных функций, представленных в ДНФ и составляемых из единого набора конъюнктивных термов. Основа ПЛМ — последовательно включенные программируемые матрицы элементов И и ИЛИ.

Программируемая матричная логика (PAL, Programmable Array Logic) — микросхема для реализации системы переключательных функций, представленных в ДНФ, каждая из которых составляется из индивидуального набора относительно небольшого числа конъюнктивных термов. Основа ПМЛ — последовательное включение программируемой матрицы элементов И и фиксированной матрицы элементов ИЛИ.

Программируемость в системе (In System Programmable) — свойство БИС/СБИС программируемой логики конфигурироваться непосредственно в системе, т. е. без изъятия из схемы.

Программируемый интервальный таймер (Programmable Interval Timer) — микросхема, выполняющая в системе операции, связанные с временами, частотами и интервалами.

Программируемый контроллер прерываний (Programmable Interrupt Controller) — микросхема, обслуживающая векторные прерывания по запросам множества источников. Реализует разнообразные способы арбитража и маскирования запросов.

Программируемый связной адаптер (Programmable Communication Interface) — микросхема, обслуживающая обмен данными между процессором и внешним устройством, оперирующим последовательными данными. Выполняет преобразования параллельных данных в последовательные, и наоборот, и необходимые интерфейсные функции.

Проектирование методом "стандартных ячеек" — проектирование БИС/СБИС, изготовляемых с помощью полного набора фотошаблонов, фрагменты которых могут заимствоваться из библиотеки готовых решений.

Прототипирование — как правило, использование программируемых или конфигурируемых стандартно выпускаемых схем для работы с проектом (отладки) перед его заказной реализацией в конструктивно законченной форме.

Псевдослучайная последовательность — детерминированная и, как правило, циклическая последовательность, состоящая из нулей и единиц, характеристики которой близки к характеристикам истинно случайной последовательности.

Р

Радиальное прерывание — прерывание, местоположение подпрограммы обслуживания которого заранее известно, и передача в процессор сведений о нем не требуется.

Разделение термов — применяемый в микросхемах программируемой логики типа ПМЛ прием, благодаря которому тракты выработки воспроизводимых функций могут заимствовать друг у друга термы, сформированные в матрице элементов И.

Реверсивный счетчик — счетчик, направление счета в котором может изменяться под воздействием управляющего сигнала.

Регенерация данных — необходимый для динамических ЗУ режим восстановления хранимых данных, периодическая реализация которого предотвращает потерю информации вследствие перезаряда запоминающих конденсаторов токами утечки.

Регистр — типовой функциональный узел цифровых устройств, выполняющий операции приема, хранения и выдачи данных, причем прием и выдача могут осуществляться для параллельных и (или) последовательных данных.

Регистровый файл — запоминающее устройство, реализованное на основе набора регистров.

Резистор-терминатор — резистор, имеющий сопротивление, равное волновому сопротивлению линии передачи сигнала, включаемый в ее конце для подавления отраженных волн.

Реинжиниринг — воспроизведение (несанкционированное) проекта с той или иной степенью раскрытия его внутренней структуры и принципов функционирования.

Репрограммируемое ПЗУ с ультрафиолетовым стиранием (РПЗУ-УФ, EPROM, Electrically Programmable Read-Only Memory) — запоминающее

устройство, в котором перед записью новой информации старая стирается с помощью облучения кристалла ультрафиолетовыми лучами на специальном стенде в течение довольно длительного времени.

Репрограммируемое ПЗУ с электрическим стиранием (РПЗУ-ЭС, EEPROM, Electrically Erasable Programmable Read-Only Memory) — запоминающее устройство, в котором перед записью новой информации старая стирается с помощью электрических сигналов, что может быть осуществлено без изъятия ЗУ из схемы устройства.

С

Самовосстановление после сбоя — свойство автомата входить в рабочий цикл после попадания в "лишние" (неиспользуемые) состояния без воздействия специальных сигналов установок.

Свертка по модулю — сложение по модулю значений разрядов кодовой комбинации.

Сегментированная система межсоединений — система коммутации, свойственная главным образом схемам FPGA, в которой линии связей составляются из отдельных сегментов, т. е. проводящих участков, не содержащих программируемых ключей. Сами сегменты соединяются друг с другом программируемыми ключами.

Семисегментный индикатор — индикатор для визуального восприятия символов, в котором эти символы отображаются с помощью семи отрезков прямых (сегментов).

Синдром ошибки — слово, составленное из разрядов, значения которых определяются результатами проверок групп, входящих в кодовые комбинации кода Хемминга. Синдром указывает номер неверного разряда, подлежащего исправлению.

Симуляция — моделирование поведения системы (целевой) при помощи системы с иной физической природой.

Синтез — процесс перевода описания проекта от одного уровня абстракции к другому (как правило, в направлении к определенной физической реализации).

Синхронизатор одиночных импульсов — схема выработки по команде одиночного импульса, принадлежащего тактовой последовательности системы.

Синхронность — скоординированное изменение состояния нескольких элементов схемы.

Синхронный автомат — автомат, элементы памяти которого принимают информацию только в определенные моменты времени, задаваемые синхросигналами.

Системный интерфейс — интерфейс межмодульного обмена в пределах микропроцессорной системы.

Системный эквивалентный вентиль — единица измерения сложности программируемых БИС/СБИС. Определение "системный" означает, что через число таких эквивалентных вентилей выражаются и сложности блоков, не относящихся к числу логических, прежде всего блоков памяти.

Сквозной ток — кратковременный импульс тока потребления микросхемы, характерный для элементов ТТЛ(Ш) и КМОП и возникающий при их переключении.

Событийно управляемое моделирование — моделирование, при котором определение состояния модели происходит только при возникновении события (изменении состояния каких-либо внутренних или внешних элементов, таких как новый такт работы). Отличается от временного моделирования, при котором вычисление состояния системы осуществляется через одинаковые кванты астрономического времени.

Совершенная дизъюнктивная нормальная форма (СДНФ) — форма представления переключательных (логических) функций, дизъюнкция конъюнкций одинаковой размерности, включающих литералы всех аргументов.

Статическая помехоустойчивость — устойчивость к воздействию помех, длительность которых не ограничивается. Определяется амплитудами помех, не нарушающих работу элемента.

Статический риск — кратковременные "ложные" сигналы, появляющиеся в переходных процессах на выходах схем в ситуациях, в которых согласно логическим уравнениям выходные сигналы должны оставаться неизменными. Возникают как следствие задержек сигналов в цепях схемы.

Статическое ОЗУ (SRAM) — оперативное запоминающее устройство, основой запоминающего элемента которого является триггер. Отличается высоким быстродействием.

Сторожевой таймер — таймер, предназначенный для отслеживания ситуаций, в которых программа выполняется неправильно, и вызывающий перезапуск программы в этих случаях.

Страничная организация памяти — организация памяти, при которой адрес ячейки рассматривается как состоящий из двух частей, причем старшая часть указывает на страницу (субмодуль), а младшая является адресом слова на данной странице (в данном субмодуле).

Стробирующий сигнал — сигнал, своим уровнем определяющий интервал времени, на котором выполняется та или иная операция.

Схема ускоренного умножения — в данном контексте схема, реализующая алгоритм умножения "сразу на два разряда".

Схемы с переключаемыми конденсаторами — аналоговые схемы, в которых масштабирующими элементами, имитирующими резисторы, являются цепочки с периодическими переключениями конденсатора на два направления.

Счетчик — автономный автомат, который под действием входных (тактирующих) сигналов переходит из одного состояния в другое, фиксируя по модулю в том или ином коде число поступивших на его вход сигналов, т. е. автомат с кольцевой диаграммой состояний.

Счетчик асинхронный — счетчик, разряды которого при переходе в новое состояние формируются не одновременно.

Счетчик Джонсона (счетчик Мебиуса, сдвигающий регистр с перекрестной обратной связью) — счетчик, работающий в коде Либау—Крейга.

Счетчик синхронный — счетчик, разряды которого при переходе в новое состояние переключаются одновременно под воздействием входного (тактирующего) сигнала.

Т

Табличный функциональный преобразователь (LUT, Look-Up Table) — логический блок программируемых БИС/СБИС, реализованный на основе схем программируемой памяти.

Тег — дополнительные данные, сопровождающие хранимую в кэш-памяти единицу информации и определяющие, копией содержимого какой ячейки основной памяти является эта единица информации.

Терм — в данной книге под этим термином понимается конъюнктивный терм, т. е. логическое произведение переменных (их прямых или инверсных значений).

Тестирование — проверка совпадения требуемого и реального поведения устройств и систем, проводимая с использованием определенных методов и средств.

Третье состояние — состояние "отключено", в котором выход логического элемента практически отсоединяется от нагрузки. Элементы с тремя состояниями выхода (0, 1 и "отключено") могут подключаться к магистралям систем с магистрально-модульной структурой.

Триггер — элементарный автомат, содержащий элемент памяти с емкостью один бит и схему управления записью в этот элемент памяти.

Триггер асинхронный — триггер, воспринимающий воздействия информационных входных сигналов непосредственно в моменты их изменений.

Триггер-защелка — триггер типа D, имеющий режим "прозрачности" при одном уровне управляющего сигнала и режим хранения при другом.

Триггер синхронный — тактируемый триггер, воспринимающий воздействия информационных сигналов только при разрешении их приема специальным тактовым сигналом.

Триггер, управляемый уровнем — триггер, для которого сигналом разрешения приема информации является тот или иной уровень управляющего (тактирующего) сигнала. Такой триггер называют также синхронным триггером со статическим управлением.

Триггер, управляемый фронтом — триггер, для которого сигналом разрешения приема информации является перепад управляющего (тактирующего) сигнала. Такой триггер называют также синхронным триггером с динамическим управлением.

Триггер D — синхронный триггер с одним информационным входом, принимающий состояние, соответствующее входному сигналу, по разрешению тактирующего сигнала.

Триггер JK — триггер, имеющий информационные входы установки и сброса, а также режим счетного триггера.

Триггер RS — триггер, имеющий информационные входы установки и сброса.

Турбо-бит — бит, программированием которого в схемах выбирается один из двух режимов — более быстродействующий (при повышении потребляемой схемой мощности) или менее быстродействующий (более экономичный по потребляемой мощности).

У

Универсальный логический модуль — устройство, воспроизводящее любую функцию заданного числа аргументов.

Ф

Фиксированный приоритет — приоритет, присвоенный данному запросу (входу) и не изменяющийся в процессе работы системы.

Флэш-память — репрограммируемая память, в которой стирание данных производится электрическими сигналами для всего кристалла либо для отдельных блоков (симметричных или несимметричных).

Флэш-память с зеркальным битом (Mirror-bit Memory) — флэш-память с хранением двух битов в одном запоминающем элементе с помощью пространственного разделения областей хранения двух зарядов в плавающем затворе транзистора.

Флэш-память с несимметричными блоками (Boot-Block Flash Memory) — флэш-память для хранения программ и других редко изменяемых данных, в

структуре которой выделяются специализированные блоки различного назначения.

Флэш-память с симметричными блоками (Flash-File Memory) — флэш-память для замены электромеханических запоминающих устройств на основе жестких дисков.

Функции возбуждения триггера — функции, определяющие такие воздействия на триггеры автомата, которые переводят автомат из одного состояния в другое согласно требуемому графу переходов.

Функциональная ячейка — типовое схемное решение, входящее в состав библиотеки БМК и реализуемое на основе одной или нескольких базовых ячеек кристалла.

Функциональное моделирование — моделирование цифровых систем, при котором не учитываются задержки распространения сигналов внутри отдельных компонентов.

Функция генерации — вспомогательная функция, используемая при синтезе сумматоров и некоторых других устройств, в которых используются сигналы переноса. Принимает единичное значение для тех разрядов или групп разрядов, на выходах которых сигнал переноса возникает независимо от наличия или отсутствия входного переноса.

Функция прозрачности — вспомогательная функция, используемая при синтезе сумматоров и некоторых других устройств, в которых используются сигналы переноса. Принимает единичное значение для тех разрядов или групп разрядов, на выходах которых сигнал переноса возникает только при наличии входного переноса.

Ц

Цикл ЗУ — минимальный интервал времени между соседними однотипными обращениями к ЗУ. Соответственно типу обращения различают циклы чтения, записи и др.

Циклический (круговой) приоритет — порядок обслуживания запросов (прерывания, прямого доступа к памяти и др.), для которого источники запросов равноправны. Равноправность источников запросов достигается тем, что их приоритеты изменяются при работе системы — после обслуживания источник получает низший приоритет, который постепенно повышается по мере обслуживания других источников запросов.

Э

Эквивалентный вентиль — группа схемных элементов, соответствующая возможности реализации на ней функции вентиля (чаще всего 2И-НЕ, 2ИЛИ-

НЕ). Понятие "эквивалентный вентиль" используется при оценке сложности (уровня интеграции) БМК и БИС/СБИС программируемой логики.

Энергонезависимость — свойство запоминающего устройства сохранять информацию при отключении питающих напряжений.

Я

Ячейки граничного сканирования (Boundary Scan Cells) — дополнительные схемы в составе БИС/СБИС, обеспечивающие реализуемость их тестирования по интерфейсу JTAG.

Ячейка памяти — схема для хранения слова (многоразрядного или одноразрядного), доступ к которому осуществляется по одному адресу.

Дополнение. Словарь иностранных сокращений и терминов

AGP — Advanced Graphics Port — стандартный интерфейс графических систем.

AHDL — Altera Hardware Description Languages — язык описания аппаратуры фирмы Altera.

ASCII — American Standard Code for Information Interchange — стандартный американский код обмена информацией в последовательных интерфейсах.

ASICs — Application Specific Integrated Circuits — специализированные ИС, изготовляемые тем или иным способом по индивидуальному техническому заданию (для конкретного проекта).

AVR — семейство микроконтроллеров фирмы Atmel.

Back Annotation — корректировка схемы проекта, которая опирается на информацию, полученную после моделирования или любых других видов обработки проектной информации, и отражает требования изменения схемы в зависимости от результатов обработки (смена контактов, обмен вентильных групп).

BEDORAM — Burst Extended Data Out RAM — вариант динамических ОЗУ, близкий к EDORAM и отличающийся от него пакетным доступом к данным, позволяющим сократить цикл обращения внутри пакета.

BIST — Built In Self Test — средства тестирования интегральных схем, встроенные в эту же схему.

BSCs — Boundary Scan Cells — *см. Ячейки граничного сканирования.*

BST — Boundary Scan Testing — *см. Периферийное сканирование.*

Bus — шина, магистраль.

CAD — Computer Aided Designer — общеупотребительный термин для всех программных средств, которые поддерживают создание конструируемых систем. Раньше термин использовался только для электронных версий средств создания чертежей.

CD — Coder — шифратор.

CDR — Clock-Data Recovery — автоматическая взаимная временная подстройка передаваемых данных и синхросигналов.

CDRAM — Cached DRAM — динамическое ОЗУ повышенного быстродействия, достигаемого путем кэширования.

CISC — Complex Instruction Set Computer — архитектура процессора, имеющего обширный набор разнообразных команд.

Clock Boost — умножение частоты тактовых импульсов, одна из функций, выполняемых блоками PLL.

Clock Lock — коррекция временного положения тактовых импульсов, одна из функций, выполняемых блоками PLL.

Clock Skew — временной сдвиг тактового импульса относительно заданного положения, вызванный паразитными задержками в цепях тактирования.

Clock Tree — деревоподобное построение синхронных схем для минимизации эффекта тактового перекоса (Clock Skew).

CPLD — Complex PLD — БИС/СБИС программируемой логики, структура которой представляет собою совокупность блоков типа PAL или GAL, объединенных матрицей программируемых соединений. Программируется пользователем.

CPU — Central Processor Unit — центральный процессор.

CRU — Carry Unit — блок ускоренных переносов.

CSoC — Configurable System on Chip — микросхема типа "конфигурируемая система на кристалле".

DC — Decoder — дешифратор.

DCM — Digital Clock Manager — блок управления параметрами синхросигналов.

DDR — Double Data Rate — передача сигналов с двойной скоростью.

DIMM — Dual In Line Memory Module — модуль памяти с двухрядным расположением выводов.

DLL — Delay Locked Loop — схема коррекции параметров синхросигналов.

DMA — Direct Memory Access — прямой доступ к памяти.

DMUX — Demultiplexer — демультиплексор.

DRAM — Dynamic RAM — динамическое ОЗУ.

DRC — Design Rule Check — проверка проектной документации для разводки ИС или печатных плат на соответствие физическим или электрическим ограничениям для предполагаемой к использованию технологии их изготовления.

DRDRAM — Direct RDRAM — вариант динамического ОЗУ высокого быстродействия типа RDRAM, в котором сокращено характерное для RDRAM запаздывание при доступе к первому слову пакета данных (латентность).

DSP — Digital Signal Processing — цифровая обработка сигналов.

EAB — Embedded Array Block — встроенный блок памяти/обработки в микросхемах программируемой логики.

EDA — Electronic Design Automation — общее название программных средств автоматизации проектирования сложных электронных систем.

EDIF — Electronic Design Interchange Format — формат обмена проектов при разработке электронных схем. Список цепей в этом стандарте может быть получен из описаний проекта на языках VHLD или Verilog HDL с помощью стандартных программ. Файлы в формате EDIF могут формироваться пакетами программных средств ряда САПР для целей моделирования с помощью стандартного пакета моделирования EDIF.

EDORAM — Extended Data Out RAM — вариант динамического ОЗУ повышенного быстродействия.

EEPROM — Electrically Erasable Read Only Memory — электрически стираемое репрограммируемое ЗУ.

EPROM — Electrically Programmable Read Only Memory — репрограммируемое ЗУ с ультрафиолетовым стиранием данных.

EPROM-OTP — однократно программируемое ЗУ с программированием состояний плавающих затворов МОП-транзисторов.

ESB — Embedded System Block — встроенный системный блок памяти/обработки в микросхемах программируемой логики.

FACM — Full Associative Cache Memory — полностью ассоциативная кэш-память.

FCRAM — Fast Cycle Random-Access Memory — динамическое ЗУ с сокращенной длительностью цикла.

FIFO — First-In — First-Out — ЗУ с последовательным доступом к данным типа "очередь" (по правилу "первый вошел — первый вышел").

Firm-ядро — разновидность soft-ядра с меньшей степенью свободы реализации в микросхеме с программируемой структурой.

FLOTOX — Floating Gate—Thin Oxide — технология производства транзисторов с плавающим затвором и тонким подзатворным окислом.

FPGA — Field Programmable Gate Array — БИС/СБИС программируемой логики, структура которой представляет собой матрицу программируемых логических блоков, между строками и столбцами которой реализованы программируемые соединения. Программируется пользователем.

FPM — Fast Page Mode — *см. Быстрый страничный доступ.*

FRAM — Ferroelectric RAM — ферроэлектрическое ОЗУ.

GA — Gate Array — базовый матричный кристалл (вентильная матрица).

GRM — General Routing Matrix — главная матрица соединений.

Hard-ядро — область кристалла с программируемой структурой, в которой реализовано то или иное устройство жесткой структуры, не допускающее репрограммирования.

Hit — сигнал "попадание" в схемах кэш-памяти, свидетельствующий о наличии запрашиваемой единицы информации в этой памяти.

HLD — Hardware Description Language — язык описания аппаратуры.

HSTL — High Speed Transceiver Logic — стандарт сигналов интерфейса.

I²C — Inter-Inter Computer — интерфейс синхронного последовательного обмена для соединения между микроЭВМ или между микроЭВМ и периферийными устройствами.

IP — Intellectual Property — интеллектуальная собственность, термин, принятый для наименования программных ядер (soft-ядер) микросхем с программируемыми структурами.

ISP — In-System Programmable — *см. Программируемость в системе.*

JEDEC — Joint Electronic Device Engineering Council — объединенный инженерный совет по электронным устройствам, в области программируемой логики обозначает текстовый файл, содержащий информацию о программировании схемы в стандартной форме JEDEC.

JTAG — Joint Test Action Group — объединенная группа по вопросам тестирования, по имени которой названы методы тестирования БИС/СБИС без физического доступа к каждому их выводу и программирования микросхем программируемой логики с помощью JTAG-интерфейса.

LIFO — Last-In — Fist-Out — ЗУ с последовательным доступом к данным стекового типа (по правилу "последний вошел — первый вышел").

LPGA — Laser-Programmable Gate Array — базовый матричный кристалл с лазерным программированием межсоединений.

LUT — Look Up Table — *см. Табличный функциональный преобразователь.*

LVCMOS — Low Volt CMOS — стандарт сигналов низковольтного интерфейса для КМОП-схем.

LVDS — Low Volt Differential Signaling — стандарт быстродействующего низковольтного дифференциального интерфейса.

LVPECL — Low Volt Positive ECL — стандарт низковольтного интерфейса для схем положительной эмиттерно-связанной логики.

LVTL — Low Volt Transistor Logic — стандарт интерфейса для низковольтной транзисторной логики.

MDAC — Multiplying Digital-Analog Converter — умножающий цифроаналоговый преобразователь.

MDRAM — Multibank DRAM — многобанковая динамическая память, вариант повышения быстродействия ЗУ с помощью разбиения памяти на части (банки), что при кучности адресов последовательных обращений к памяти позволяет обращаться к банкам поочередно с более высокой скоростью.

MIPS — (1) Mega Instructions Per Second — миллион команд в секунду, мера производительности процессора; (2) Microprocessor without Interlocked Pipeline Stages — название известной фирмы, означающее, в сущности, "Микропроцессор без задержек ожидания конвейера".

MLC — Multilevel Cell — многоуровневая ячейка памяти, позволяющая хранить в ней более одного бита.

MPGA — Mask-Programmable Gate Array — базовый матричный кристалл с масочным программированием межсоединений.

MRAM — Magnetic RAM — перспективное ОЗУ, работа которого основана на применении магнитных материалов.

MUX — Multiplexer — мультиплексор.

Net List — перечень всех компонентов проекта и всех сигналов, соединяющих эти компоненты между собой.

NoBL — No Bus Latency — вариант ОЗУ с устранением задержки при реверсе шин.

Node — одиночный контакт в электрических схемах.

NtRAM — No Turnaround RAM — вариант ОЗУ с устранением задержки при реверсе шин.

NvSRAM — Non-Volatile SRAM — статическое ОЗУ со свойством энергонезависимости.

OUM — Ovonics Unified Memory — перспективное ЗУ фирмы Ovonics, работа которого основана на применении материалов с изменяемыми фазовыми состояниями.

OTP — One-Time Programmable — "однократно программируемая", определение относится к микросхемам памяти типа РПЗУ-УФ, корпус которых для удешевления не имеет прозрачного окна для стирания данных путем воздействия на кристалл ультрафиолетовым облучением. В таких ЗУ можно произвести лишь однократное программирование путем необратимого заряда плавающих затворов запоминающих транзисторов.

PAL — Programmable Array Logic — *см. Программируемая матричная логика.*

PCI — Peripheral Component Interconnect — популярная шина расширения для соединения периферийных компонентов системы.

PFRAM — Polimeric FRAM — полимерная ферроэлектрическая память.

PIC — Programmable Interruption Controller — программируемый контроллер прерываний.

PIP — Programmable Interconnect Point — программируемая точка соединений.

PLA — Programmable Logic Array — *см. Программируемая логическая матрица.*

PLD — Programmable Logic Device — общее наименование для схем PAL и PLA.

PLL — Phase Locked Loop — схема следящей системы с чувствительным элементом, реагирующим на разность фаз импульсных последовательностей, используемая для управления временными параметрами синхросигналов цифровых устройств.

PREP — Programmable Electronics Performance Corporation — консорциум компаний, предложивший набор эталонных схем и методику оценки сложности БИС/СБИС программируемой логики.

PSOC — Programmable System On Chip — программируемая система на кристалле.

Pull-down Resistor — "заземляющий" резистор, фиксирующий низкий уровень потенциала на выводе в отсутствие на нем активного сигнала.

Pull-up Resistor — "подтягивающий" резистор, фиксирующий высокий уровень потенциала на выводе в отсутствие на нем активного сигнала.

RAM — Random-Access Memory — оперативное ЗУ (ЗУ с произвольным доступом).

RDRAM — Rambus DRAM — динамическое ОЗУ высокого быстродействия, разработанное фирмой Rambus.

RISC — Reduced Instruction Set Computer — вариант архитектуры быстродействующих процессоров с сокращенным набором команд.

RLDRAM — Reduced Latency DRAM — быстродействующее динамическое ОЗУ.

ROM — Read-Only Memory — постоянная память (память с рабочим режимом только для чтения).

SDRAM — Synchronous DRAM — синхронное ОЗУ динамического типа высокого быстродействия, в котором высокий темп передачи данных обеспечивается конвейерной организацией тракта передачи, тактируемого от синхросигналов, общих для процессора и памяти. Широко применяется в современных компьютерах.

SERDES — Serialiser-Deserialiser — блок преобразования параллельных данных в последовательные, и наоборот.

SIMM — Single In-line Memory Module — модуль памяти с однорядным расположением выводов.

SOC — System On Chip — БИС/СБИС высшего уровня сложности, в которой можно реализовать целую систему, т. е. совокупность разных модулей, образующих целостную систему обработки информации.

SOI — Silicon On Insulator — схемотехнология интегральных схем, обеспечивающая минимальность паразитных параметров схемы, что, в конечном счете, приводит к улучшению ее технических характеристик.

SOPC — System On Progammable Chip — БИС/СБИС программируемой логики высшего уровня сложности, на которой можно реализовать целую систему, т. е. совокупность разных модулей, образующих целостную систему обработки информации.

SPI — Serial Peripheral Interconnect — синхронный последовательный интерфейс.

SPLD — Simple Programmable Logic Device — программируемая логическая схема невысокой сложности.

SRAM — Static RAM — статическое ОЗУ.

SSTL — Stub Series Terminated Logic — стандарт сигналов интерфейса, используемый для микросхем быстродействующих динамических ОЗУ.

StrataFlash — запоминающее устройство флэш-типа с запоминанием двух битов в одном запоминающем элементе с помощью многоуровневого заряда плавающих затворов транзисторов ЛИЗМОП.

TAP — Test Access Port — *см. Порт тестирования.*

Terminator — резистор в конце линии связи, обеспечивающий для нее согласование волнового сопротивления с нагрузкой.

UART — Universal Asynchronous Receiver – Transmitter — программируемый связной адаптер, реализующий асинхронные протоколы передачи последовательных данных.

Verilog HDL — язык описания аппаратуры фирмы Cadence. Наряду с языком VHDL относится к самым популярным языкам описания аппаратуры высокого уровня.

VHDL — Very-High-Speed Hardware Description Language — язык описания аппаратуры, стандарт IEEE, по-видимому, наиболее популярный язык описания аппаратуры высокого уровня.

VLIW — Very Long Instruction Word — архитектура процессора, имеющего набор очень длинных (многобайтовых) команд.

WDT — Watchdog Timer — сторожевой таймер.

XACT — пакет программных средств для проектирования БИС/СБИС программируемой логики фирмы Xilinx.

ZBT — Zero Bus Turnaround — вариант ОЗУ с устранением задержки реверса шины.

Принятые сокращения

А	адрес
АЗУ (CAM)	ассоциативное ЗУ
АП	автомат с памятью
	адресное пространство
АЦП (ADC)	аналого-цифровой преобразователь
БВВ (IOB)	блок ввода/вывода
БиКМОП	схемотехнология, сочетающая использование биполярных и МОП-транзисторов
БИС (LSI)	большая интегральная схема
БИСМ	БИС, реализованная на основе БМК
БМК (GA)	базовый матричный кристалл
БЯ	базовая ячейка
ВБП	встроенный блок памяти
ВУ	внешнее устройство
ГПСП	генератор псевдослучайных последовательностей
ГПСЧ	генератор псевдослучайных чисел
ДКОИ-8	восьмиразрядный двоичный код обмена информацией
ДНФ (SOP)	дизъюнктивная нормальная форма
ДОЗУ (DRAM)	динамическое ОЗУ
ЖКИ	жидкокристаллический индикатор
ЗГ	задающий генератор системы синхронизации
ЗУ	запоминающее устройство
ЗУПВ	запоминающее устройство с произвольной выборкой
ЗЭ	запоминающий элемент
ЗЯ	запоминающая ячейка
ИС	интегральная схема

ИСПС	интегральная схема с перестраиваемой структурой
КЛБ (LAB)	конфигурируемый логический блок
КМОП (CMOS)	комплементарная МОП-структура
КОИ-7	семиразрядный код обмена информацией
КПДП (DMA)	контроллер прямого доступа к памяти
КЦ	командный цикл
ЛБ (LB)	логический блок
ЛВ	линия выборки (словарная)
ЛЗС	линия записи/считывания (разрядная)
ЛИЗМОП	МОП-структура с лавинной инжекцией заряда
ЛПМС	локальная программируемая матрица соединений
ЛФ	логическая функция
ЛЭ (LE)	логический элемент
М	модуль счета
	информационная емкость ЗУ
МАБИС	матричная БИС
МБ	множительный блок
МБЯ	матричная базовая ячейка
МИС (LSI)	малая интегральная схема
МК (MC)	микроконтроллер
МНОП	структура "металл-нитрид-оксид-полупроводник"
МОП (MOS)	структура "металл-оксид-полупроводник"
МП (MP)	микропроцессор
МПК	микропроцессорный комплект
МПС (MCS)	микропроцессорная система
МРТ	матрица распределения термов
МСБ	множительно-суммирующий блок
МЦ	машинный цикл
МЭТ	многоэмиттерный транзистор
ОЗУ (RAM)	оперативное ЗУ
ОК (OC)	открытый коллектор

ОС	обратная связь
	открытый сток
ПАИС	программируемая аналоговая интегральная схема
ПБ	переключательный блок
ПБЯ	периферийная базовая ячейка
ПДП (DMA)	прямой доступ к памяти
ПЗУ (ROM)	постоянное ЗУ
ПЗУМ	постоянное ЗУ с масочным программированием
ПИТ (PIT)	программируемый интервальный таймер
ПКП (PIC)	программируемый контроллер прерываний
ПЛ	программируемая логика
ПЛИС (PLD)	программируемая логическая интегральная схема
ПЛМ (PLA)	программируемая логическая матрица
ПЛУ (PLD)	программируемое логическое устройство
ПМЛ (PAL)	программируемая матричная логика
ПМС (PIA)	программируемая матрица соединений
ППА (PPI)	программируемый параллельный адаптер
ППВМ (FPGA)	программируемая пользователем вентильная матрица
ППЗУ (PROM)	программируемое постоянное ЗУ
ПС (BS)	периферийное сканирование
ПСА (PCI)	программируемый связной адаптер
ПТ (TAP)	порт тестирования (для интерфейса JTAG)
ПТС (PIP)	программируемая точка связи
ПЯ	периферийная ячейка
РВВ	регистр ввода/вывода
РИ	распределитель импульсов
РМП	реконфигурируемая матрица памяти
РОН	регистр общего назначения
РПЗУ-УФ (EPROM)	репрограммируемое ПЗУ со стиранием данных ультрафиолетовыми лучами

РПЗУ-ЭС (EEPROM)	репрограммируемое ПЗУ с электрическим стиранием данных
РСС	регистр слова состояния
РТС	распределитель тактовых сигналов
РУ	распределитель уровней
РУС	регистр управляющего слова
САПР	система автоматизированного проектирования
СБ	связной блок
СБИС (VLSI)	сверхбольшая интегральная схема
СБИС ПЛ	СБИС программируемой логики
СДНФ	совершенная дизъюнктивная нормальная форма
СИС (MSI)	средняя интегральная схема
СОЗУ (SRAM)	статическое ОЗУ
СпИС (ASIC)	специализированная ИС
СПЛИС (CPLD)	сложная программируемая логическая ИС
ССИ (SSI)	семисегментный индикатор
СФ-блок (IP)	сложный функциональный блок
Т	транзистор
Т	такт
Т	триггер
ТИ (CLK)	тактовые импульсы
ТС	третье состояние логического элемента
ТТЛ (TTL)	транзисторно-транзисторная логика
ТТЛШ (TTLS)	ТТЛ с диодами Шотки
УАПП (UART)	универсальный асинхронный приемопередатчик
УВВ	устройство ввода/вывода
УЛМ	универсальный логический модуль
УС (CW)	управляющее слово
УСАПП (USART)	универсальный синхронно-асинхронный приемопередатчик
ФБ	функциональный блок

ФЯ	функциональная ячейка
ЦАП (DAC)	цифроаналоговый преобразователь
ЦОС (DSP)	цифровая обработка сигналов
ЦУ	цифровой узел
	цифровое устройство
ША (AB)	шина адреса
ШД (DB)	шина данных
ШУ (CB)	шина управления
ШФ	шинный формирователь
ЭВ	эквивалентный вентиль
ЭСЛ (ECL)	эмиттерно-связанная логика

Литература и источники в Интернете

Краткая библиография

1. Автоматизация проектирования БИС. В 6 кн.: Практ. пособие/Под ред. Казеннова. — М.: Высш. шк., 1990.
2. Алексенко А. Г. Основы микросхемотехники. — 3-е изд., перераб. и доп. — М.: ЮНИМЕДИАСТАЙЛ, 2002. — 448 с.
3. Алексенко А. Г., Шагурин И. И. Микросхемотехника. — М.: Радио и связь, 1990. — 496 с.
4. Антонов А. П. Язык описания цифровых устройств AlteraHDL. Практический курс. — М.: ИП РадиоСофт, 2001. — 224 с.
5. Антонов А. П., Мелехин В. Ф., Филиппов А. С. Обзор элементной базы фирмы Altera. — СПб.: ЭФО, 1997. — 142 с.
6. Армстронг Д. Моделирование дискретных систем на языке VHDL: Пер. с англ. Т. А. Теплицкого/Под ред. Ю. А. Татарникова. — М.: Мир, 1992.
7. Домрачев Г., Мальцев П. П., Новаченко И. В., Пономарев С. Н. Базовые матричные кристаллы и матричные БИС/В. — М.: Энергоатомиздат, 1992. — 224 с.
8. Бибило П. Н. Основы языка VHDL. — М.: Солон-Р, 1999. — 200 с.
9. Бирюков С. А. Применение цифровых микросхем серий ТТЛ и КМОП. — 2-е изд., стер. — М.: ДМК, 2000. — 240 с.
10. Бродин В. Б., Калинин А. В. Системы на микроконтроллерах и БИС программируемой логики. — М.: Издательство ЭКОМ, 2002. — 400 с.
11. Бродин В. Б., Шагурин И. И. Микроконтроллеры: Справочник. — М.: ЭКОМ, 1999. — 395 с.
12. Букреев И. Н., Мансуров Б. М., Горячев В. И. Микроэлектронные схемы цифровых устройств. — 3-е изд. — М.: Радио и связь, 1990. — 415 с.
13. Вычислительные машины и системы: Учеб. для вузов/В. Д. Ефремов, В. Ф. Мелехин, К. П. Дурандин и др./Под ред. В. Д. Ефремова, В. Ф. Мелехина. — М.: Высшая школа, 1993. — 292 с.
14. Гладштейн М. А. Микроконтроллеры семейства Z86 фирмы ZILOG. — М.: ДОДЭКА, 1999. — 96 с.

15. Григорьев В. Л. Программное обеспечение микропроцессорных систем. — М.: Энергоатомиздат, 1983. — 208 с.

16. Грушвицкий Р. И., Мурсаев А. Х., Угрюмов Е. П. Проектирование систем на микросхемах программируемой логики. — СПб.: БХВ-Петербург, 2002. — 608 с.

17. Грушин С. И., Душутин И. Д., Мелехин В. Ф. Проектирование аппаратных средств микропроцессорных систем: Учеб. пособие. — Л.: ЛПИ им. Калинина, 1990. — 78 с.

18. Гук М. Ю. Аппаратные средства IBM PC: Энциклопедия. — 2-е изд. — СПб.: Питер, 2003. — 928 с.

19. Гутников В. С. Интегральная электроника в измерительных устройствах. — Л.: Энергоатомиздат, 1988. — 304 с.

20. Евстифеев А. В. Микроконтроллеры AVR семейства Classic фирмы Atmel. — М.: Издательский дом Додэка-XXI, 2002. — 288 с.

21. Каган Б. М., Сташин В. В. Основы проектирования микропроцессорных устройств автоматики. — М.: Энергоатомиздат, 1988. — 304 с.

22. Киносита К., Асада К., Карацу О. Логическое проектирование СБИС: Пер. с япон. — М.: Мир, 1988. — 309 с.

23. Кнышев Д. А., Кузелин М. О. ПЛИС фирмы Xilinx: описание структуры основных семейств. — М.: Издательский дом Додэка-XXI, 2000. — 240 с.

24. Лебедев О. Н. Применение микросхем памяти в электронных устройствах: Справочное пособие. — М.: Радио и связь, 1994. — 216 с.

25. Лебедев О. Н., Мирошниченко А. И., Телец В. А. Изделия электронной техники. Цифровые микросхемы. Микросхемы памяти. Микросхемы ЦАП и АЦП: Справочник. — М.: Радио и связь, 1994. — 248 с.

26. Логические ИС КР1533, КР1554: Справочник: В 2-х частях/И. И. Петровский, А. В. Прибыльский, А. А. Троян, В. С. Чувелев. — М.: БИНОМ, 1993. — 496 с.

27. Микропроцессорные системы: Учеб. пособие для вузов/Под общ. ред. Д. В. Пузанкова. — СПб.: Политехника, 2002. — 935 с.

28. Микропроцессоры и микропроцессорные комплекты интегральных микросхем: Справочник: в 2-х т./Н. Н. Аверьянов, А. И. Березенко, Ю. Н. Борщенко и др./Под ред. В. А. Шахнова. — М.: Радио и связь, 1988. — Т. 1 — 368 с. Т. 2 — 368 с.

29. Микросхемы памяти, ЦАП и АЦП: Справочник. — 2-е изд./О. Н. Лебедев, А-Й. К. Марцинкявичус, Э-А. К. Багданскис и др. — М.: КУбК-а, 1996. — 384 с.

30. Новиков Ю. В. Основы цифровой схемотехники. — М.: Мир, 2001. — 379 с.

31. Норенков И. П., Манчев В. Б. Основы теории и проектирования САПР: Учеб. для втузов по спец. "Вычислительные машины, комплексы, системы и сети". — М.: Высшая школа, 1990. — 335 с.

32. Перспективы развития вычислительной техники: В 11 кн.: Справ. пособие./Под ред. Ю. М. Смирнова. Кн. 7: Полупроводниковые запоминающие устройства/А. Б. Акинфиев, В. И. Миронцев, Г. Д. Софийский, В. В. Цыркин. — М.: Высшая школа, 1989. — 160 с.

33. Потемкин И. С. Функциональные узлы цифровой автоматики. — М.: Энергоатомиздат, 1988. — 320 с.

34. Предко М. Руководство по микроконтроллерам: В 2-х т. — Пер. с англ. — М.: Постмаркет, 2001. — Т.1 — 415 с., Т.2 — 487 с.

35. Применение интегральных микросхем памяти: Справочник/А. А. Дерюгин, В. В. Цыркин, Е. В. Красовский и др./Под ред. А. Ю. Гордонова, А. А. Дерюгина. — М.: Радио и связь, 1994. — 232 с.

36. Программируемые логические ИМС на КМОП-структурах и их применение/П. П. Мальцев, Н. И. Гарбузов, А. П. Шарапов, А. А. Кнышев. — М.: Энергоатомиздат, 1998. — 158 с.

37. Пухальский Г. И. Проектирование микропроцессорных систем: Учеб. пособие для вузов. — СПб.: Политехника, 2001. — 544 с.

38. Пухальский Г. И., Новосельцева Т. Я. Цифровые устройства: Учеб. пособие для втузов. — СПб.: Политехника, 1996. — 885 с.

39. Разевиг В. Д. Система проектирования цифровых устройств OrCad. — М.: Солон-Р, 2000. — 160 с.

40. Разевиг В. Д. Система сквозного проектирования электронных устройств DesignLab 8.0. — М.: Солон-Р, 2000. — 698 с.

41. Соловьев В. В., Васильев А. Г. Программируемые логические интегральные схемы и их применение. — Мн.: Беларуская навука, 1998. — 270 с.

42. Стешенко В.Б. ПЛИС фирмы ALTERA: проектирование устройств обработки сигналов. — М.: ДОДЭКА, 2000. — 128 с.

43. Суворова Е. А., Шейнин Ю. Е. Проектирование цифровых систем на VHDL. — СПб.: БХВ-Петербург, 2003. — 576 с.: ил.

44. Титце У., Шенк К. Полупроводниковая схемотехника: Справочное руководство: Пер. с нем. — М.: Мир, 1982. — 512 с.

45. Угрюмов Е. П. Проектирование элементов и узлов ЭВМ: Учеб. пособие для вузов. — М.: Высшая школа, 1987. — 318 с.

46. Хоровиц П., Хилл У. Искусство схемотехники: Пер с англ. — 6-е изд. — М.: Мир, 2001. — 830 с.

47. Цифровые интегральные микросхемы: Справочник/П. П. Мальцев, Н. С. Долидзе, М. И. Критенко и др. — М.: Радио и связь, 1994. — 240 с.

48. Шалыто А. А. Логическое управление. Методы аппаратной и программной реализации алгоритмов. — СПб.: Наука, 2000. — 780 с.

49. Шило В. Л. Популярные цифровые микросхемы: Справочник. 2-е изд. — Челябинск: Металлургия, Челябинское отд., 1989. — 352 с.

50. Щелкунов Н. Н., Дианов А. П. Микропроцессорные средства и системы. — М.: Радио и связь, 1989. — 282 с.

51. Asheden P. J. The designer's guide to VHDL. — San Francisco: Morgan Kaufman Publishers. — 1996. — 688 p.

52. Bostock G. Programmable Logic Devices. — N-Y.: McGraw Hill, 1988. — 243 p.

53. Brey B. The 8085A Microprocessor: Software, Programming and Architecture. — Prentice-Hall, Englewood Cliffs. — N. J., 1986. — 220 p.

54. Bursky D. Embedded Logic And Memory Find A Home In FPGA// Electronic Design. — 1999. — № 14. — pp. 43—56.

55. Bursky D. High-Density FPGA Family Delivers Megagate Capacity// Electronic Design. — 1997. — № 25. — pp. 67—70.

56. Bursky D. Programmable Logic Challenges Traditional ASIC SoC Designs// Electronic Design. — 2002. — April 15.

57. Chang D., Mazek-Sadowska M. Dinamically Reconfigurable FPGA//IEEE Transaction on Computers. — 1999. — № 6. — pp. 565—578.

58. Dipert B. Third Annual Programmable-Logic Directory. — Elecgtronic Design News, September 5. — 2002. — pp. 43—64.

59. Gajsky D. Principles of Digital Design. Prentice Hall, New Jersey, 1997. — 447 p.

60. Kang S., Lebelevici Y. CMOS Digital Integrated Circuits. Analysis and Design. Boston, McGrow-Hill, 1999.

61. Kresta D., Johnson T. High-Level Design Methodology Comes Into Its Own//Electronic Design. — 1999. — № 12. — pp. 57—60.

62. Manni V. Best of Both Worlds in Parallel Digital Adders//IEEE Circuits & Devices. — Vol. 18, 2002, № 5. — pp. 20—23.

63. Oshima Y., Sheu B., Jen S. High-Speed Memory Architectures For Multimedia Applications//IEEE Circuits & Devices. — Vol. 13, — № 1. — Jan. 1997. — pp. 8—13.

64. Perry D. L. VHDL: Programming by Example. — Fourth Edition. McGraw-Hill, 2002. — 497 p.

65. Rabaey J. M. Digital Integrated Circuits: A Design Perspective. — Prentice Hall. 1997. — 734 p.

66. Short K. Microprocessors And Programmed Logic. — 2-nd Ed. — Englewuod Cliffs: Prentice-Hall, 1987. — 515 p.

67. Takai Y., Nagase M., Kitamura M. a. o. 250 Mbyte/s Synchronous DRAM Using a 3-Stage-Pipelined Architecture//IEEE Journal of Solid-State Circuits. — Vol. 29. — № 4. — April 1994. — pp. 426—429.

Интернет-ресурсы

Адреса в сети Интернет, по которым публикуются сведения фирм о выпускаемых ими микросхемах и средствах автоматизированного проектирования цифровых устройств.

I.	**www.actel.com**	XIX.	**www.micron.com**
II.	**www.aldec.com**	XX.	**www.mips.com**
III.	**www.altera.com**	XXI.	**www.mitsubishichips.com**
IV.	**www.amd.com**	XXII.	**www.model.com**
V.	**www.arm.com**	XXIII.	**www.motorola.com**
VI.	**www.atmel.com**	XXIV.	**www.quicklogic.com**
VII.	**www.cadence.com**	XXV.	**www.rambus.com**
VIII.	**www.cypress.com**	XXVI.	**www.ramtron.com**
IX.	**www.exemplar.com**	XXVII.	**www.synopsis.com**
X.	**www.fujitsumicro.com**	XXVIII.	**www.synplicity.com**
XI.	**www.hitachi.com**	XXIX.	**www.ti.com**
XII.	**www.infineon.com**	XXX.	**www.toshiba.com**
XIII.	**www.innoveda.com**	XXXI.	**www.triscend.com**
XIV.	**www.intel.com**	XXXII.	**www.usa.samsungsemi.com**
XV.	**www.latticesemi.com**	XXXIII.	**www.veribest.com**
XVI.	**www.lsilogic.com**	XXXIV.	**www.xilinx.com**
XVII.	**www.mentor.com**	XXXV.	**www.zilog.com**
XVIII.	**www.microchip.com**		

В списке литературы приведены работы, изданные в течение последних 15 лет (за исключением единичных, изданных раньше, но наиболее близких по теме к данному пособию). Перечислены в основном отечественные книги, среди которых мало современных работ, что отражает объективное состояние дел в области издания научно-технической литературы в России. В перечне работ на английском языке преобладают ссылки на адреса сети Интернет и журнальные публикации, т. к. иностранные книги, хотя и имеются в природе, в настоящее время практически недоступны для большинства российских читателей (в том числе и для авторов).

Многочисленные ссылки на отдельные журнальные статьи в списке литературы для учебного пособия нецелесообразны, но перечисление основных журналов, помещающих полезные практические материалы по цифровой и аналоговой схемотехнике, представляется интересным для читателей. К таким журналам относятся следующие: зарубежные Electronic Design (USA) и Electronic Design News (имеется сетевой вариант) и отечественные журналы Chip News, Электроника-НТБ (наука, технология, бизнес), Электронные компоненты, Схемотехника, Компоненты и технологии, Компьютер пресс, BYTE-Россия. Существуют периодические научно-теоретические издания, но их перечисление здесь не приводится, поскольку они представляют интерес для достаточно узкого круга специалистов, которые и без информации со стороны хорошо обо всем осведомлены.

Предметный указатель

1

1T-SRAM 313

A

Active Interconnect Technology 609
AGP 46
AHB 598
AMBA 598
AMI 626
ANSI 45
APB 598
Aplac 658
ASB 598
ASICs 645
ASSP 643

B

BEDORAM 316
BIOS 277
BSC 472
BST 472

C

CAB 589
CDR 188
CDRAM 329
Charge Pump 187
Clock Boost 184
Clock Lock 184
Clock Managers 184
Clock Skew 184, 186
CoCentric SystemC Compiler 666
CoreConnect 598
CPLD 494, 533, 543
CSOC 592, 617
CTT 46

D

DCM 608
DDR LVDS 50
DDR SDRAM 320
Design Flow 640
DesignLab 658, 660
Development Board 661
DIMM 334
DLL 184, 186, 188
DMA 450
DRAM 233, 303
DRDRAM 325

E

EABs 571, 578
ECL 47
ECU 610
Editor of FSM 687
EDORAM 315
EDRAM 329
EEPROM 233, 537
EIA 45
Electronics Workbench 658
End-Front Design 581
EPROM 231
EPROM-OTP 537
ESB 600
ESP 609
Evaluation Board 661
Excalibur 619

F

FACM 248
FCRAM 332
FLASH 233, 537
FLEX 533
FPAA 589
FPAD 589

FPGA 494, 533, 553, 612
FPGA-процессоры 542
FPM 314, 315
FPM DRAM 315
FPSLIC 592, 611, 616
FRAM 236, 335
FTI 571

G

GA 494, 520
GA/SCI 647
GAL 507
GPSS 654
GRM 605
GTL+ 46

H

HDL 667
HSTL 46

I

Idle 490
IEEE 45, 674
IOB 544
IOE 601
ISP 537, 628

L

LAB 600
LI 600
LPGA 645
LUTs 77, 554
LVCMOS 45
LVDM 48
LVDS 45, 46, 48
LVDT 48
LVPECL 45
LVTL 45

M

MathCAD 654
MATLAB 654
MAX+Plus II 692
MDAC 586
MDRAM 316
Microblaze 607
Micro-Cap 658
M-LVDS 46, 48
ModelSim SE 5.7b 692
MPGA 533, 645
MRAM 236, 338

N

NoBL 298
NtRAM 298
NV-SRAM 301

O

One Hot 693
OPB 598
OrCAD 655
OUM 236, 340

P

PAL 77, 494, 507
P-CAD 655
PCI 45
PCI-X 45
PC-LOGS 660
PECL 47
PFRAM 236, 337
PIA 543
PIO 617
PIP 538
Pipelined Burst SRAM 298
PLA 77, 494
PLB 598
PLD 494, 517
PLL 184, 186, 188
Power Down 490, 540
PREP 632
PROM 231, 259
Protel 655
Prototype Plate 661
PSoC 584, 591
Pull-down Resistors 24
Pull-up Resistors 24

R

RC-цепочки 54
RDRAM 318, 324
RIMM 335
RLDRAM 331
ROM(M) 231, 256
RS-422 46
RS-485 46
RTL 679
RC-цепочки 55

S

SDRAM 46, 318
SERDES 610
SignalTap Logic Analysis 673
SIMM 334
Simula 654
SIMULINK 654
Single-gate Logic 70
SLC 77, 554
SOI 540
SOP 600
SOPC 494, 533, 591
Spice 654
SPLD 494
SRAM 233, 291
SSTL-2 46
SSTL-3 46
Standby Power 540
Starter Kit 661
State Machins 668
Synchronous Burst SRAM 298

T

Test-Bench 671
TIA 45
TriMatrix 602

U

UART 424, 433
USART 424

V

VersaRing 606

W

Waveform Editor 673

Z

ZBT 298

А

Автомат:
автономный 169
линейный 222
Мили 169
Мура 169, 171, 173
на триггерах с мультиплексным управлением 175
с кодированием состояний в коде "1 из N" 177
с памятью (АП) 73, 143, 167, 168
асинхронные 168
кодирование состояний 170
проектирование 169, 172
синхронные 168, 169
Адаптеры:
типа УАПП (UART) 424, 433
типа УСАПП (USART) 424
Адресация 357
абсолютная 355
косвенная 354
относительная 355
простая 355
микроконтроллеры
косвенной адресации 489
прямой адресации 488
неабсолютная 355
линейная селекция 355
непосредственная 355
прямая 354
страница 355
Алгоритм Бута 135
АЛУ 114, 129
Алфавит:
двухсимвольный 9
девятисимвольный 9
для описания логических сигналов 9
семисимвольный 9
трехсимвольный 9
четырехсимвольный 9
Аналоговая схемотехника 581
имитация резисторов 582
отношение сопротивлений точных резисторов 582
цепи с переключаемыми конденсаторами 582, 583
Аномалии:
колебательные 162
метастабильные 162
Арифметико-логические устройства (АЛУ) 129
Архитектура МПС:
Гарвардская 346
Принстонская 346
фон Неймана 346
Ассоциативный доступ 235
Атрибут сигнала 692

Б

Базис:
Буля 8
Пирса 8
Шеффера 8
Базовые матричные кристаллы (БМК) 494, 520, 645
аналоговые 526
базовая ячейка (БЯ) 522
бесканальные 525
библиотека функциональных ячеек 522
блочные 525
каналы трассировки 523
канальной архитектуры 520
канальные 523
классификация 523
матричные базовые ячейки (МБЯ) 522
параметры 528
периферийные базовые ячейки (ПБЯ) 522
разновидности 520
типа MPGA 533
типа БиКМОП 526
трассировочная способность 521
функциональная ячейка (ФЯ) 522
цифроаналоговые 526
цифровые 526
число слоев межсоединений 526
эквивалентный вентиль (ЭВ) 523

БИС 5, 69
БИС/СБИС:
полузаказные 520
типа CPLD 533, 543
блоки ввода/вывода 551
параллельные логические расширители 546
последовательные логические расширители 545
системы коммутации 549
структура 543
функциональные блоки 544
типа FLEX 533
типа FPGA 533, 553
VersaRing 567
адресация программируемых перемычек 563
блоки ввода/вывода 559
зернистость логических блоков 554
иерархическая система связей 561
крупнозернистые логические блоки 556, 557
логические блоки 554, 557
мелкозернистые логические блоки 554
обобщенная структура 567
система межсоединений Actel 561
система межсоединений Xilinx 563
системы межсоединений (коммутации) 561
типа SOPC 533, 593
БИСМ 521
Бистабильный атом 335
Блоки:
CDR 52
CRU 52
DES 52
SERDES 51
ускоренного переноса 131
Блочные структуры:
несимметричные 271
симметричные 271
Бод 420
Буфер:
FIFO 52, 235, 246
LIFO 235
Буферные регистры 403
Быстрые сдвигатели 139
управляемые двоичным кодом (логарифмический) 141
управляемые кодом "1 из N" 139
Быстрый страничный доступ 314

В

Ввод/вывод отображенный на память 356
Вентиль 78
Вентильные матрицы (ВМ) 494, 520
Вес комбинации 106
Видеопамять 235, 244
Витая пара 39, 40
Временные:
диаграммы 149
перекосы 683
состязания 9
состязания сигналов 157
Время:
доступа 231
предустановки 230, 685
сохранения 230
удержания 230

Г

Генераторы:
импульсов 56
псевдослучайной последовательности 224
псевдослучайных чисел (ГПСЧ) 226
с кварцевыми резонаторами 58
Гигантский магниторезистивный эффект 339
Граничное сканирование 472
Граф переходов 685

Д

Дейзи-цепочка 88, 440
Демультиплексоры 91

Деревья Уоллеса 135
Дешифратор:
двоичный 81
наращивание размерности 83
неполный 81
схемотехническая реализация 83
условное обозначение 81
Дизъюнктивная нормальная форма (ДНФ) 78, 495
Дифференциальная передача сигнала 39
Дребезг контактов 163
способы устранения 163

Е

Емкость:
паразитная 29

З

Задержка:
на логических элементах 53
на одновибраторах 55
на счетчиках 55
с помощью RC-цепочки 54
Запрещенная комбинация сигналов 144
Запрос на обслуживание 86
ЗУ (запоминающие устройства) 227
Boot Block Flash Memory 271
Flash-File Memory 271
асинхронные 234
ассоциативные 235, 248
быстродействие 228
время деактивации 317
время доступа 317
время цикла 317
входные и выходные сигналы 229
двухбанковые структуры 255
динамические 233, 234, 303
основные параметры 333
повышенного быстродействия 313
регенерация данных 311
режим "считывание-модификация-запись" 311
с фрагментированной структурой 329
схема 309
типа BEDORAM 316
типа CDRAM 329
типа DDR SDRAM 320
типа EDORAM 315
типа FCRAM 332
типа FPM 314
типа MDRAM 316
типа RDRAM 324
типа RLDRAM 331
типа SDRAM 318
усилители-регенераторы 307
задержка сигнала CAS относительно RAS 317
запоминающая ячейка (ЗЯ) 228
запоминающий элемент (ЗЭ) 228
информационная емкость 228
квазистатические 233, 313
классификация 231
конвейеризация продвижения данных 318
кучность адресов 252
латентность памяти 317
магниторезистивные 236
масочные типа ROM(M) 231, 256
многобанковые структуры 255
многопортовые 234
нетактируемые 234
однопортовые 234
оперативные (ОЗУ) 231
основные структуры 237
блочного типа 243
типа 2D 237
типа 2DM 241
типа 3D 238
типа FIFO 246
организация 228
полимерные ферроэлектрические 236
последовательные 286
постоянные (ПЗУ) 231
внешняя организация 308
запоминающие элементы (ЗЭ) 303
мультиплексирование шины адреса 307
произвольный доступ 253

с последовательным доступом 235
синхронные 234
конвейерная передача данных 297
конвейризованные с пакетным обменом 298
с пакетным обменом 298
сквозная передача данных 297
статические 297
статические 233, 234, 291
запоминающие элементы 292, 293
искусственная энергонезависимость 299
с ускоренным реверсом шины 298
типа NV-SRAM 301
типа БиКМОП 302
стековые 235
структурные методы повышения быстродействия 252
быстрый страничный метод 253
конвейеризация 255
пакетная передача данных и команд 253
технология DDR 254
технология QDR 254
тайминг (Timing) 254, 316, 317
тактируемые 234
типа EEPROM 262
типа EPROM 262
типа EPROM-OTP 266
типа FIFO 235, 246
типа FRAM 335
типа MRAM 338
типа MTJ 339
типа OUM 236, 340
типа PFRAM 337
типа PROM 259
программирование 261
типа QDR 254
типа ROM 256
файловые 235
ферроэлектрического типа 236
циклические 235
энергонезависимость 229

И

Импульс сквозного тока 14
Индикаторы:
жидкокристаллические (ЖКИ) 61, 62
инверсии 7
семисегментные 59
Интегральные схемы (ИС) 5
на стандартных ячейках 646
полностью заказные 645
полузаказные 646
специализированные 645
стандартные 643
Интерфейс 397
hard-ядра 398
I²C 437
I²C 401
JTAG 471
JTAG-цепочка 475
команды граничного сканирования 477
механизм граничного сканирования 477
порт доступа TAP 474
схема ячейки BSC 473
транспортный механизм 475
устройство управления граничным сканированием 475
Microbus 398, 399
Multibus 399
SPI 401, 434
задатчик (активный) 399
И-41 399
ИРПР 416
исполнитель (пассивный) 399
мегафункции 398
IP 398
soft-ядра 398
механическая совместимость 397
протокол обмена 397
с общей шиной 355
с раздельной шиной 355
системный 398
функциональная совместимость 397
шина Q-bus 399
шина VME 400
электрическая совместимость 397

Интерфейсы ПК:
RS-232C 401
RS-485 401
шина CAN 401
шина EISA 400
шина ISA 400
шина MCA 400
шина PCI 400
шина USB 401
шина VL-Bus 400
Искажения сигналов в несогласованных линиях 29
ИСПС 534

К

Код 106
"1 из N" 170, 172
ASCII 421
Грея 172, 213
ДКОИ-8 421
КОИ-7 421
Хемминга 110
модифицированный 110, 112
схема кодирования и декодирования 112
циклический 226
Кодовая комбинация 106
Кодовое расстояние 106
минимальное 106
Комбинационные цепи (КЦ) 73, 167
Компараторы 100
на больше 101
на равенство 101, 103
Компилятор 657
Конвольверы 614
Контроллер:
прерываний 350
каскадное включение 449
прямого доступа к памяти 351, 450
временный регистр 455
выводы и сигналы 457
две фазы работы 455
программирование 453
регистр запросов 455
регистр масок 455
регистр режима 453
регистр состояния 453
регистр управления 453
режим автоинициализации 453
структура 451
увеличение числа каналов ПДП 458
Контроль:
по модулю 2 105, 107, 109
по четности (нечетности) 106, 107
с использованием кодов Хемминга 110
Коэффициент отражения волны 31
Кратность ошибки 106
Критерии качества ЦУ 80
АТ 80
общий 80
по Квайну 79
частный 80
Кусочно-линейная аппроксимация 290
Кэш-память 227, 247
второго уровня (L2) 252
набор 251
наборно-ассоциативная 249, 251
первого уровня (L1) 252
полностью ассоциативная 248
с прямым размещением 249, 251
строки 249
тег 248, 251

Л

Линейная селекция 355
Линии связи:
с гальваническими развязками 41
типа "токовая петля" 42
Логика:
отрицательная 8
положительная 8

М

МАБИС 521, 645
этапы проектирования 529
Матричные перемножители 132
Меандр 470
Межсоединения элементов 37

Метод Эйхельбергера 9
Микроконтроллеры (МК) 378, 479
  выполнение команд 489
  программирование 491
  программирование по последовательному каналу 491
  режим глубокого понижения мощности 490
  режим покоя 490
  режимы потребления мощности 490
  семейства AVR 481 482
  сигналы сброса 489
  система приоритетных прерываний 491
  спящие режимы 490
Микропроцессор (МП) 5, 345
  АЛУ 360
  блок регистров 361
    программный счетчик (PC) 362
    регистр адреса (RA) 363
    регистры общего назначения (РОН) 361
    указатель стека (SP) 361
  блок синхронизации и управления 362
  исключения 369
  код операции (КОП) 367
  командный цикл (КЦ) 363, 367
  машинные циклы (МЦ) 363, 368
    типы 363
  мультиплексируемые шины 353
  прерывания:
    аппаратные 369
    маскирование 370
    программные 369
  разрядность 351
  регистр команд (IR) 362
  регистр флажков (RF) 361, 376
  регистр-аккумулятор 359
  сигнал $\overline{\text{INTA}}$ 365
  сигнал ALE 353, 364, 367
  сигнал CLK 365
  сигнал HLDA 365
  сигнал HOLD 365
  сигнал INT 351
  сигнал INTA 351
  сигнал INTR 365
  сигнал IO/M 364
  сигнал READY 364
  сигнал RESET 365
  сигнал TRAP 365
  сигналы S_0 и S_1 364
  система команд 373
  стек 361
    типа LIFO 361
  такты 363, 368
  типа CISC 347, 481
  типа K1821BM85A 359
  типа RISC 347, 481
  типа VLIW 347
  цифровой обработки сигналов 378
Микропроцессорная система (МПС) 345
  адресация памяти 380
  адресное пространство 354
  вектор прерывания 351, 371
  обмен с квитированием 394
  прямой доступ к памяти 351
  система прерываний 369
  структура 349
    магистрально-модульная 349
    трехшинная 349
Микропроцессорный комплект (МПК) 348
МикроЭВМ 479
Минимизация логических функций 78, 79
МИС 5, 69, 493
Множительно-суммирующий блок 132, 133
Множительные блоки 132
МНОП-транзистор 262
Модемы 419
Модули памяти 356
  DIMM 334
  RIMM 335
  SIMM 334
Монтажная логика 18
Мосты 400
Мощность:
  динамическая 13
  полная 13
  статическая 13
М-последовательность 224

Мультиплексная формула 89
Мультиплексное управление 63
Мультиплексоры 88
наращивание размерности 90

О

Области применения FPGA 541
Одновентильные схемы 71
Одновибраторы 55
Операция "исключающее ИЛИ" 123
Оптрон 41
Отказы 103

П

ПАИС 534, 581
микросхемы семейства ispPAC 584
динамическая реконфигурация 584
основные параметры 587
микросхемы типа FPAD 589
Память:
внешняя 228
кэш-память 227
основная 227
программируемая пользователем 231
регистровые ЗУ 227
специализированная 227
типа FreeRam 614
Параллельные периферийные адаптеры (ППА) 410
пример использования 417
режимы работы 411
Передача информации:
асинхронный обмен 421
дуплексная 421
контроль четности 423
марка 422
ошибка пропуска 423
ошибка формата 423
полудуплексная 421
посылка (кадр) 422
разновидности синхронных передач 423
симплексная 48, 421
типа Multidrop 48
типа Multipoint 48
типа Point-to-Point 48
синхронный обмен 421
старт-бит 422
стоп-бит 422
Перекрестные помехи 28
Переход к логическому базису 79
Периферийное сканирование 472
Периферийные адаптеры 410
Персональные компьютеры (ПК) 400
Пиксел 244
Плавающий затвор 264
ПЛИС 534, 645
классификация 535
области применения 541
понижение потреблемой мощности 539
применение 534
программирование 538
ключевых МОП-транзисторов триггером памяти 537
перемычками типа antifuse 535
перемычками типа ONO 535
плавающих затворов транзисторов 537
программируемая точка связи (ПТС) 538
программируемость 535
режим Power Down 540
режим Standby Power 540
турбо-бит 540
Плоские дисплеи 63
Поллинг 439
аппаратный 439
Поляризация материала 335
Порты параллельные 404
Последовательностные схемы 167
Преобразователь параллельного кода в последовательный 200
Прерывания:
вектор 442
вложенность 441
круговой приоритет 441
маскирование запросов 442
специальное маскирование 442
Приоритет 86

Программа:
HDL Designer 669, 687
LeonardoSpectrum 695
StateCAD Version 3.2 669, 709
Synplify 695
SystemVision 695
Программирование ЗУ 261
Программируемая логика 650
Программируемая матричная логика (ПМЛ) 494
серии К1556 512
структура 507
типа PAL 22V10 516
Программируемые интервальные таймеры 351, 459
инициализация 467
описание работы 466
режим генератора одиночного строба 470, 471
режим генератора частоты 469
режим генерации меандра 470
режим одновибратора 469
режим с прерыванием по окончании счета 468
режимы работы 468
сторожевые 463
таймер-счетчик 0 460
таймер-счетчик 1 461
типа ВИ54 464
чтение содержимого счетчика 467
Программируемые контроллеры прерываний 439, 440
включение в МПС 442
маскирование запросов 442
программирование 445
структура 443
Программируемые логические матрицы (ПЛМ) 494
биполярные 497
воспроизведение скобочных форм 503
на МОП-транзисторах 499
наращивание 504
программирование 501
схемотехника 497
Программируемые пользователем вентильные матрицы (FPGA) 553
Программируемый связной адаптер 419, 424
временные диаграммы 430
пример подключения к МП и терминалу 433
программирование 428
Программно-управляемый обмен 358
Программный безусловный ввод/вывод 392
Проектирование 637
аналоговые и аналого-цифровые фрагменты 660
восходящее 640
выбор САПР 663
групповое использование САПР 664
инструментальные средства 652
интерпретирующие редакторы 671
исходные модули 640
компилирующие редакторы 671
компиляция проекта 670
конструкторско-топологическое представление 638
кросс-ассемблирование 657
микропроцессорная система 650
на дискретных ИС 658
на основе МИС и СИС 649
нисходящее 639
определение временных характеристик 671
основные идеи 653
основные составляющие 640
отладка на прототипе 657
платформа 653
программная модель 671
проекты на ПЛИС и SOPC 662
прототипные платы 672
процесс 639
разработка процессорного блока 655
с использованием языков описания 705
симуляция 657
системный этап 653
спецификация проекта 664
(окончание рубрики см. на с. 778)

Проектирование *(окончание)*:
способ описания проекта:
графический 666
поведенческий 667
структурный 667
текстовый 666, 667
среда и система разработки 657
средства описания автоматов 668
граф-схемы 668
средства отладки 661
структурное описание 638
тестирование проекта 670
тип обрабатываемой информации 641
функциональное моделирование 670
функциональное описание 638
ЦУ на базе ПЛИС 663
экспериментальная проверка 672
эмуляция 657
этап системного проектирования 641
этапы 638
этапы проектирования с использованием САПР 670
языки описания аппаратуры 667
высокого уровня 668
низкого уровня 667
Пропускная способность шины 50
Прямой доступ к памяти 358, 450
блочные передачи 450
одиночные передачи 450
Псевдослучайные последовательности 223

P

Распределители:
импульсов 214
тактов 214, 215
уровней 214
Регистровые файлы 195
Регистры 194
PISO 194
SIPO 52, 194
буферные 403
классификация 194
кольцевые
с перекрестной обратной связью 218
параллельные 194
последовательно-параллельные 194
последовательные (сдвигающие) 194
нереверсивные 194
реверсивные 194
сдвигающие 197
многотактные 198
однотактные 197
реверсивные 197
с линейной обратной связью 223, 226
универсальные 198
Режимы:
"высокого импеданса" 9
неиспользуемых входов 64
"отключено" 9
Резисторы:
"заземляющие" 24
"подтягивающие" 24
Риски 73
динамические 74
статические 73

C

Самосинхронизирующиеся устройства 169
САПР 2, 6, 640, 652
этапы работы 654
СБИС 5, 69
СБИС ПЛ 534
firm-ядра 593
hard-ядра 593, 594, 595
аппаратные ядра 594, 596
процессорные 597
soft-ядра 593, 594
процессорные 596
блоки IP 593
виртуальные компоненты 593
защищенность проектов 629
кристаллы с триггерной памятью 631
однократно программируемые кристаллы 630

репрограммируемые кристаллы 630
клонирование 629
конвертация проектов 623, 624, 625
конфигурирование микросхем 626
активный последовательный режим 627
байт-последовательный режим 627
пассивный последовательный режим 627
режим граничного сканирования 628
реконфигурирование 628
способы 627
микросхемы HardCopy 625
однократно программируемые микросхемы 724
оценка быстродействия 634
градация быстродействия 635
оценка логической сложности 632
методика PREP 632
набор эталонных схем 632
параметры 633
реконструкция проектов 629
репрограммируемые с триггерной памятью 726
с комбинированной архитектурой 569
блоки ввода/вывода 580
встроенные блоки памяти 570, 578
двухуровневая коммутация 570
логические блоки и их коммутация 576
логические элементы 571
обратные связи 572
структура 569
тракты переноса и каскадирования 571
с программированием плавающих затворов 725
системная частота тактирования 634
СФ-блоки 593
типа SOPC 591
блочные 594, 609
блочные беспроцессорные 609
блочные процессорные 610
блочные типа ESP 609
взаимодействие блоков 615
логические ячейки FPGA 612, 614
мегаблоки 600
микросхемы EPSLIC 611, 616, 617
микросхемы Excalibur 619, 620
микросхемы QuickDSP 610
микросхемы QuickPC 610
микросхемы QuickPCI 609
микросхемы QuickRAM 609
микросхемы QuickSD 610
модули пользователя 622
однородные 593, 594
процессорные ядра 596
с аналоговыми и цифровыми блоками 620
с флэш-памятью 608
семейства APEX 20K, APEX II 598
семейство Eclipse 608
семейство Stratix 601
семейство Virtex 604
системы на кристалле 592, 598 617
эквивалентный вентиль 632
Сбои 103
Свертка по модулю 2, 107
Сдвигатель:
типа Barrel Shifter (управляемый кодом "1 из N") 139, 140
СДНФ 76, 77
Сигнал:
случайный 222
стробирующий 294
Сигнатура 226
Сигнатурные анализаторы 226
Сила сигнала 9
Синдром ошибки 112
Синтез 637
Синхронизаторы одиночных импульсов 164
Синхронизация:
двухфазная 192, 193
многофазная 194
однофазная 188, 189
расчетные соотношения 190

Синхронизация в цифровых устройствах 179
СИС 5, 69, 493
Система переключательных функций 288
Системы синхронизации 182
  устройства типа DLL 184
  устройства типа PLL 184
Согласование волновых сопротивлений:
  параллельное 32
  последовательное 36
Сопротивление волновое 31
Стандарт IEEE 1149.1 472
Стек 361
Сужение шин 324
Сумматор 114
  групповой структуры 124
  накапливающие 127
  одноразрядный 114
  параллельный с параллельным переносом 119
  параллельный с последовательным переносом 118
  последовательный 117
  с передачей сигнала переноса по цепочке замкнутых ключей 122
  с условным переносом 127
Схемы:
  "Деревья" Уоллеса 135
  самосинхронизирующиеся 188
  свертки 107
  типа Weak pin-keeper 24, 25
  ускоренного умножения 135
Счетчики 201
  асинхронные 204
  быстродействие 202
  в коде "1 из N" 214
    на основе кольцевых регистров 215
  в коде Грея 213, 214
  двично-десятичные 211
  двоичные 202
  Джонсона 218, 220
  исключение лишних состояний 208
  классификация 202
  лишние состояния 210
  модуль счета 201
  полиномиальные 222
  реверсивные 204
  режимы работы 202
  с групповой структурой 206
  с произвольным модулем 208
    способы построения 209
  с управляемым сбросом 212
  свойство самозапуска 210
  синхронные с параллельным переносом 205
  синхронные с последовательным переносом 207

T

Таблица:
  истинности 76
  функционирования 76
Тактовые импульсы:
  параметры 180
  размножение 183
Тег 235
Терм 495
Терминатор 32
Термы 78
Технология:
  DDR 50, 254
  SDR 50, 254
  БиКМОП 302
Токовые импульсы в цепях питания 26
Тракт обработки информации 179
Транзисторы:
  ЛИЗМОП 264
  с плавающим затвором 264
    стирание информации 265
  типа ETOX 265
  типа FLOTOX 264, 265
Триггеры 143
  аномальные состояния 162
  асинхронные 145
  время выдержки 147
  время предустановки 147
  двухступенчатые 146, 156

диаграмма состояния 148
запрещенная комбинация входных сигналов 152
карта Карно 148
классификация 144
комбинированные 145
на элементах КМОП 153
двунаправленный ключ 153
одноступенчатые 146, 158
с динамическим управлением 146, 155
синхронные 145
словарь триггера 148
со сложной входной логикой 145
способы описания функционирования 147
структурное уравнение 148
схемотехника 149
счетные 145
тактовые сигналы 145
типа D 145, 149, 152, 154, 160
типа JK 145, 151, 159
типа RS 144
асинхронные 152
синхронные 152
типа T 145, 149
триггер-защелка 146
упарвляемые уровнем 145
управляемые фронтом 146
установочные сигналы 144
характеристическое уравнение 148
Шмитта 39
Троичная алгебра 10
Туннелирование носителей заряда 339

У

Указатели старшей единицы 86, 88
Универсальные логические модули на основе мультиплексоров 91
Условные графические обозначения 7

Ф

Фазность 182
Фильтрация напряжений питания 26
Флэш-память 269
командный интерфейс пользователя (CUI) 282
микросхемы MLC 283
с зеркальным битом 284
с несимметричной блочной структурой 277
с симметричной блочной структурой 280
структуры с ячейками ИЛИ-НЕ и И-НЕ 271
структуры типа Concurrent Flash Memory 277
типа StrataFlash 283
улучшение параметров 274
управление словами-командами 275
файловая 280
Формирование импульсов по длительности 55
Функциональные разновидности ПЛМ и ПМЛ 507
структуры PLD 517
Функциональные узлы:
комбинационные 73
последовательностные 73
Функция:
генерации 120
прозрачности 120

Х

Халкогенидные сплавы 340

Ц

Цепь с двумя устойчивыми состояниями 143
Цифроаналоговые преобразователи (ЦАП) 586
Цифровые устройства (ЦУ) 5
синхронизация 179

Ч

Чипсет 400

Ш

Шина 349
AHB 598
APB 598
ASB 598
CSI 617
OPB 598
PLB 598
SPI 286
Шинная система:
AMBA 598
CoreConnect 598
Шинные формирователи 401
Шифраторы
двоичные 85
приоритетные 85
наращивание размерности 87

Э

Элемент памяти (ЭП) 167
Элемент сложения по модулю 2 62
Элементы:
мажоритарные 104
с открытым коллектором (ОК) 17—19
с открытым стоком (ОС) 18, 19
с тремя состояниями выхода (ТС) 15, 16
типа ЭСЛ 22
цифровые 13
задержки 53
Элементы индикации:
на жидких кристаллах 61
на светодиодах 58

Я

Языки:
ABEL 667
AHDL 667
EDIF 667
PLDASM 667
SystemC 666
Verilog 668
VHDL 668, 673, 674
для моделирования и синтеза 679
оператор присваивания 677
описание проекта 675
поведенческий вариант описания проекта 680
понятие сигнала 676
примеры описаний элементов 678
синтаксические конструкции понятия 674
стиль описания 677
структурный вариант описания проекта 680
VHDL-AMS 674, 695
типа HDL 529
Ярусность логических схем 80
Ячейки периферийного сканирования 472

www.bhv.ru

Учебная литература издательства "БХВ-Петербург" в продаже:

Магазин "Новая техническая книга": СПб., Измайловский пр., д. 29, тел.: (812) 251-41-10
Отдел оптовых поставок: e-mail: opt@bhv.spb.su, www.techkniga.com

Серия "Учебное пособие"/ "Учебник"

Арбузов С., Гук И., Соловьева И., Солонина А., Улахович Д. Основы цифровой обработки сигналов. Курс лекций	576 с.
Барсегян А., Куприянов М., Степаненко Ю. Методы и модели анализа данных: OLAP и Data Mining (+CD-ROM)	336 с.
Барсегян А. Технологии анализа данных: Data Mining, Visual Mining, Text Mining, OLAP	384 с.
Бертяев В. Теоретическая механика на базе Mathcad. Практикум	752 с.
Белов Д. и др. Разработка экспертных систем. Среда CLIPS	608 с.
Бенькович Е. Практическое моделирование динамических систем (+CD-ROM)	464 с.
Бенькович Е. и др. Практическое моделирование динамических систем (+CD-ROM)	464 с.
Боев В. Моделирование систем. Инструментальные средства GPSS World	368 с.
Бойко В. Схемотехника электронных систем. Микропроцессоры и микроконтроллеры	464 с.
Бойко В. Схемотехника электронных систем. Цифровые устройства	512 с.
Бойко В. Схемотехника электронных систем. Аналоговые и импульсные устройства	450 с.
Большаков В. Инженерная и компьютерная графика. Практикум	592 с.
Бутиков Е. Оптика: Учебное пособие для студентов физических специальностей вузов	480 с.
Вуль В. Электронные издания (+дискета)	560 с.
Гольдштейн Б. Системы коммутации, 2-е изд.	314 с.
Гомоюнов К. Транзисторные цепи	240 с.
Грушвицкий Р. и др. Проектирование систем на микросхемах программируемой логики	608 с.
Грушвицкий Р. и др. Проектирование систем на микросхемах с программируемой структурой, 2-е изд.	650 с.
Гуревич А. Физика твердого тела	320 с.
Давыдов В. С++. Технологии программирования	672 с.
Даугавет И. Теория приближенных методов. Линейные уравнения, 2-е изд.	288 с.
Жмакин А. Архитектура ЭВМ	320 с.

Иглин С. Математические расчеты на базе Matlab	640 с.
Ирвин Дж., Харль Д. Передача данных в сетях: инженерный подход	448 с.
Иртегов Д. Введение в сетевые технологии	560 с.
Карпов Ю. Имитационное моделирование систем. Введение в моделирование	400 с.
Колесов Ю., Сениченков Ю. Моделирование систем. Динамические и гибридные системы	224 с.
Колесов Ю., Сениченков Ю. Моделирование систем. Объектно-ориентированный подход	192 с.
Колесов Ю., Сениченков Ю. Моделирование систем. Практикум по компьютерному моделированию	352 с.
Кочетов В., Кочетов М., Павленко А. Сопротивление материалов, 3-е изд.	544 с.
Крылов В., Самохвалова С. Теория телетрафика и ее приложения	288 с.
Кубенский А. Структуры и алгоритмы обработки данных: объектно-ориентированный подход и реализация на C++ (+CD-ROM)	464 с.
Кузнецов А. и др. Математика для экономистов на базе Mathcad	496 с.
Культин Н. Delphi в задачах и примерах (+CD-ROM)	288 с.
Культин Н. C/C++ в задачах и примерах	288 с.
Культин Н. Turbo Pascal в задачах и примерах	256 с.
Литвиненко Н. Технология программирования на C++. Начальный курс	288 с.
Лурье Б., Энрайт П. Классические методы автоматического управления	640 с.
Макаров Е. Сопротивление материалов на базе Mathcad (+CD-ROM)	512 с.
Максименко А. Эксплуатация строительных и дорожных машин	400 с.
Малыхина М. Базы данных: основы, проектирование, использование, 2-е изд.	512 с.
Малыхина М. Программирование на языке высокого уровня. Turbo Pascal	544 с.
Молдовян Н. Практикум по криптосистемам с открытым ключом	304 с.
Никулин Е. Основы теории автоматического управления. Частотные методы анализа и синтеза систем	640 с.
Никулин Е. Компьютерная геометрия и алгоритмы машинной графики	560 с.
Опалева Э., Самойленко В. Языки программирования и методы трансляции	480 с.
Петров Ю. История и философия науки. Математика, вычислительная техника, информатика	448 с.
Погорелов В. Строительная механика тонкостенных конструкций	528 с.
Порев В. Компьютерная графика	432 с.
Поршнев С. Вычислительная математика. Курс лекций	320 с.
Поршнев С., Беленкова И. Численные методы на базе Mathcad	464 с.
Пышкин Е. Основные концепции и механизмы объектно-ориентированного программирования	640 с.
Рапаков Г., Ржеуцкая С. Программирование на языке Pascal	480 с.

Романов Е. Практикум по программированию на C++	432 с.
Романовский И. Дискретный анализ. 3-е изд.	320 с.
Сафронов И. Бейсик в задачах и примерах	224 с.
Сеннов А. Курс практической работы на ПК	576 с.
Солонина А. и др. Цифровые процессоры обработки сигналов фирмы MOTOROLA	512 с.
Солонина А. Основы цифровой обработки сигналов. Курс лекций	608 с.
Солонина А., Улахович Д., Яковлев Л. Алгоритмы и процессоры цифровой обработки сигналов	464 с.
Сорокина С. и др. Программирование драйверов и систем безопасности	256 с.
Стешенко В. P-CAD. Технология проектирования печатных плат	720 с.
Суворова Е., Шейнин Ю. Проектирование цифровых систем на VHDL	576 с.
Угрюмов Е. Цифровая схемотехника, 2-е изд.	800 с.
Ускова О. и др. Программирование алгоритмов обработки данных	192 с.
Файншмидт В. Дифференциальное и интегральное исчисление функций одного аргумента	224 с.
Хабибуллин И. Программирование на языке высокого уровня. C/C++	512 с.
Хейфец А. 3-d технологии построения чертежа. AutoCAD. 3-е изд.	256 с.
Хейфец А. Инженерная компьютерная графика AutoCAD	336 с.
Хрящев В., Шипова Г. Моделирование и создание чертежей в системе AutoCAD	224 с.
Частиков А. и др. Разработка экспертных систем. Среда CLIPS	608 с.
Чемоданова Т. Pro/Engineer: деталь, сборка, чертеж	560 с.
Черноруцкий И. Методы принятия решений	416 с.
Черняк А. и др. Математика для экономистов на базе Mathcad	496 с.
Черняк А. и др. Высшая математика для экономистов на базе Mathcad. Общий курс	608 с.
Шапорев С. Дискретная математика. Курс лекций и практических занятий	400 с.
Шапорев С. Математическая логика. Курс лекций и практических занятий	416 с.
Шелест В. Программирование	592 с.
Щука А. Электроника	800 с.
Юревич Е. Основы робототехники, 2-е изд. (+CD-ROM)	416 с.

Серия "Для студента"

Бекаревич Ю., Пушкина Н. Microsoft Access за 21 занятие	544 с.
Герман О., Герман Ю. Программирование на Java и C#	512 с.
Дунаев В. Базы данных. Язык SQL	288 с.
Едомский Ю. Техника Web-дизайна	400 с.
Магда Ю. UNIX для студента	480 с.

Пахомов Б. С и Borland C++ Builder для студента	448 с.
Побегайло А. C/C++ для студента	528 с.
Половко А. Derive для студента	352 с.
Половко А., Бутусов П. MATLAB для студента	320 с.
Половко А., Ганичев И. Mathcad для студента	336 с.
Пономаренко С. Adobe Photoshop CS2 для студента	464 с.
Рудикова Л. Microsoft Excel для студента	368 с.
Рудикова Л. Базы данных. Разработка приложений	496 с.
Рудикова Л. Microsoft Word для студента	400 с.
Федорова А. CorelDRAW для студента	576 с.
Федорова А. Adobe Illustrator для студента	640 с.

Серия "На примерах"

Адаменко А. Pascal на примерах из математики	416 с.
Васильев А. Mathcad 13 на примерах	528 с.
Голышев С., Кузнецов А., Симдянов И. PHP 5 на примерах	576 с.
Дударева Н. SolidWorks 2007 на примерах	528 с.
Кашаев С. Программирование в Microsoft Excel на примерах	320 с.
Климов А. JavaScript на примерах	256 с.
Левковец Л. AutoCAD 2007. Базовый курс на примерах	448 с.
Мержевич В. HTML и CSS на примерах	448 с.
Пестриков В., Маслобоев А. Delphi на примерах	496 с.
Пирогов В. Ассемблер на примерах	416 с.
Поляк-Брагинский А. Администрирование сети на примерах	320 с.
Потопахин В. Язык С. Освой на примерах	320 с.
Слепченко О. Macromedia Flash Professional 8 на примерах	416 с.
Шпрута В. Delphi 2006 на примерах	528 с.

Серия "В задачах и примерах"

Культин Н. Delphi .NET в задачах и примерах	256 с.
Культин Н. Delphi в задачах и примерах (+CD-ROM)	288 с.
Культин Н., Цой Л. C/C++ Builder в задачах и примерах	336 с.
Культин Н. Turbo Pascal в задачах и примерах	256 с.
Сафронов И. Бейсик в задачах и примерах	400 с.
Сафронов И. Visual Basic в задачах и примерах, 2-е изд.	320 с.

www.bhv.ru

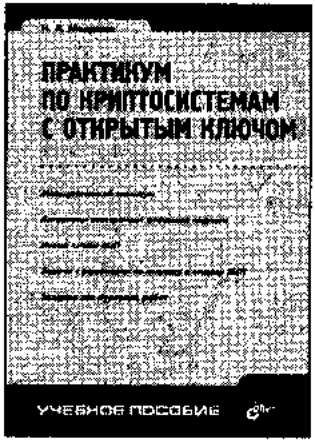
Молдовян Н.А.
Практикум по криптосистемам с открытым ключом

Магазин "Новая техническая книга"

СПб., Измайловский пр., д. 29, тел.: (812) 251-41-10

Отдел оптовых поставок

E-mail: opt@bhv.spb.su



Среди большого обилия книг, посвященных теоретическим вопросам классической и современной криптографии, трудно найти материал для практических занятий. Целью данной книги является восполнение этого пробела по отношению к криптосистемам с открытым ключом. В ней приводится около 200 вариантов заданий для выполнения курсовых работ и проектов.

На основе многолетнего преподавательского опыта автором составлен список задач по элементам теории чисел и двухключевой криптографии, включая тематику анализа и синтеза систем электронной цифровой подписи различного типа, для которых приводятся ответы, указания и решения.

- Математический минимум
- Алгоритмы электронной цифровой подписи
- Новые схемы ЭЦП
- Задачи с решениями по анализу и синтезу ЭЦП
- Задания для курсовых работ

Молдовян Николай Андреевич, доктор технических наук, профессор, заслуженный изобретатель РФ, главный научный сотрудник Научного филиала ФГУП НИИ «Вектор» – СЦПС «Спектр». Известный специалист, ведущий открытые исследования в области криптографии.

Автор более 250 печатных работ в области защиты информации и криптографии и более 60-ти патентов на изобретения, которые патентуются в 21 стране (Россия, США, Германия, Франция, Великобритания, Китай и др.). Его новые подходы к построению скоростных шифров опубликованы в ведущих научных журналах мира.

Магазин-салон
«НОВАЯ ТЕХНИЧЕСКАЯ КНИГА»

190005, Санкт-Петербург, Измайловский пр., 29

В МАГАЗИНЕ ПРЕДСТАВЛЕНА ЛИТЕРАТУРА

ПО

компьютерным технологиям
радиотехнике и электронике
физике и математике
строительству и архитектуре
транспорту
машиностроению

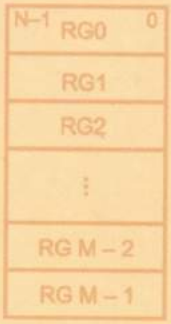
и другим
естественно-научным
и техническим направлениям

- ✓ **Низкие цены**
- ✓ **Прямые поставки от издательств**
- ✓ **Ежедневное пополнение ассортимента**
- ✓ **Подарки и скидки покупателям**

Магазин работает с 10.00 до 20.00 без обеденного перерыва выходной день – воскресенье

Тел.: (812) 251-41-10, e-mail: trade@techkniga.com
www.techkniga.com

Угрюмов Евгений Павлович, доктор технических наук, заслуженный деятель науки и техники РФ, профессор кафедры вычислительной техники Санкт-Петербургского государственного электротехнического университета.



ЦИФРОВАЯ СХЕМОТЕХНИКА

2-Е ИЗДАНИЕ

В учебном пособии рассматриваются основные вопросы дисциплины "Схемотехника ЭВМ", изучающей современную элементную базу цифровой техники от принципов построения и проектирования функциональных узлов и устройств до их практической реализации. Во втором издании отражены изменения, произошедшие со времени выхода первого издания, в такой динамичной области, как схемотехника, особенно в разделах, посвященных программируемой логике, запоминающим устройствам, проблемам скоростной передачи данных и системам синхронизации в быстродействующих устройствах.

Поскольку на единой элементной базе строятся все разнообразные средства цифровой техники: ЭВМ, микропроцессорные системы измерений и автоматизации технологических процессов, цифровая связь, телевидение и бытовая аппаратура – книга может быть полезна студентам всех специальностей, связанных с использованием и разработкой цифровых средств обработки информации, а также работникам промышленности и научных учреждений.

На мой взгляд, очень полезная книга. Мне, как студенту 3-го курса по специальности 2202, она очень помогла. Книга полностью выполняет свое назначение как учебное пособие. Не нашел ни одной ошибки, которая могла бы ввести меня в заблуждение относительно изложенного материала. Рекомендую начинающим схемотехникам, имеющим представление об электронных элементах.

Сабир Израфилов,
Россия
28 июля 2003 г.

ISBN 978-5-94157-397-4

БХВ-Петербург
190005, Санкт-Петербург,
Измайловский пр., 29
E-mail: mail@bhv.ru
Internet: www.bhv.ru
тел.: (812) 251-42-44
факс: (812) 251-12-95