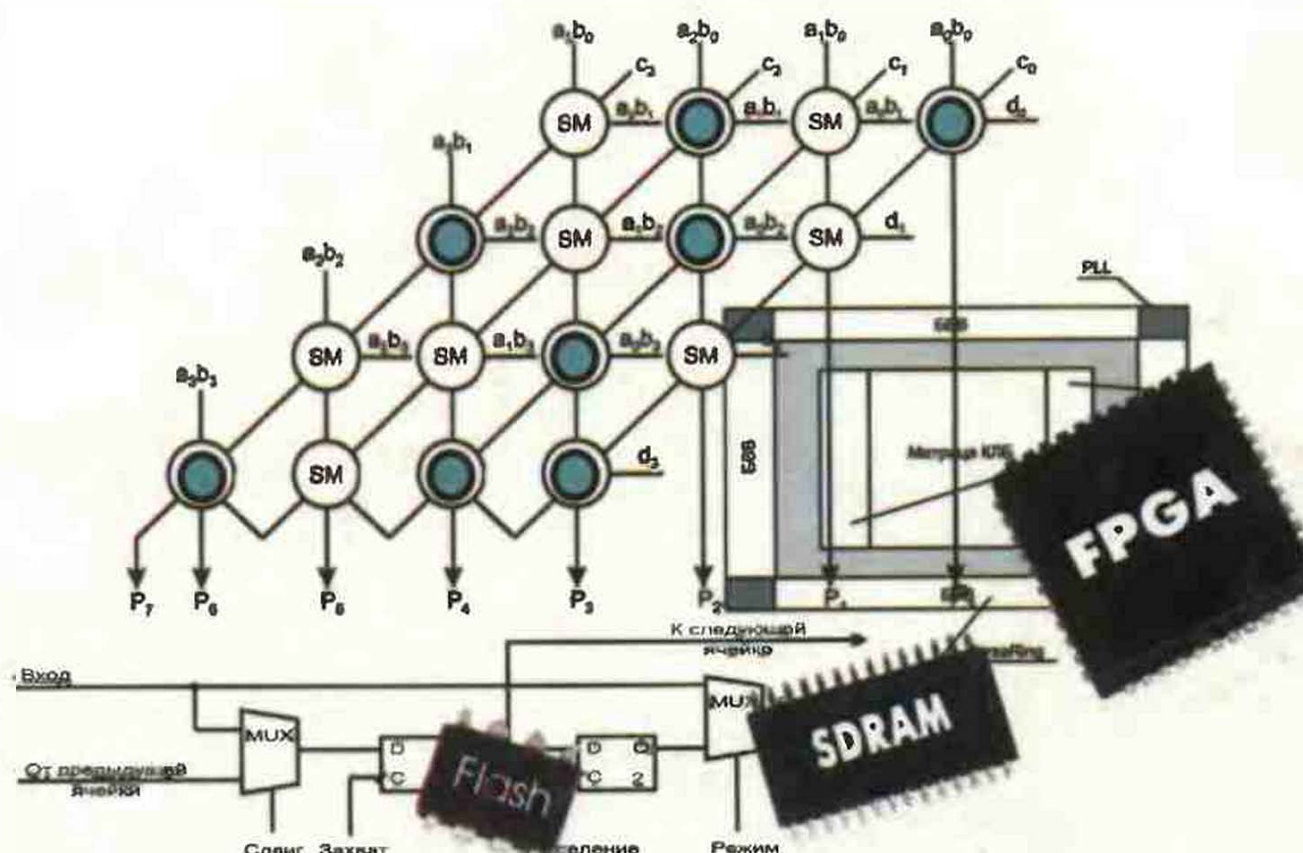


Цифровая схемотехника

учебное
пособие

**От логического элемента
до перспективных БИС/СБИС
с программируемыми структурами**

Natahaus



Е. Угрюмов

Цифровая схемотехника



Автор книги, Угрюмов Евгений Павлович, доктор технических наук, Заслуженный деятель науки и техники РСФСР, профессор кафедры Вычислительной техники Санкт-Петербургского государственного электротехнического университета, специализируется на разработке вычислительных устройств и имеет многолетний опыт преподавания в области цифровой и аналоговой схемотехники. Среди его трудов по схемотехническому направлению учебные пособия для вузов, выпущенные издательством "Вышая школа" в 1976 г. "Элементы и узлы ЭЦВМ" и в 1987 г. "Проектирование элементов и узлов ЭЦВМ".

Книга посвящена ознакомлению с элементной базой, на основе которой реализуется аппаратная часть (hardware) различных средств обработки информации (компьютеров, систем автоматики и управления, связи, бытовой техники и т. д.).

Рассмотрены методы построения функциональных блоков цифровой аппаратуры и имеющийся схемотехнический опыт в этой области.

- Цифровые элементы и узлы
- Запоминающие устройства
- Микропроцессорные и интерфейсные БИС/СБИС
- Программируемая логика
- Проектирование цифровых узлов и устройств

ISBN 5-8206-0100-9



9 785820 601002



Книга-почтой BHV

Адрес: 199397, Санкт-Петербург, а/я 194

Тел. (812) 541-8551. Факс (812) 541-8461

E-mail: trade@bhv.spb.su, Internet: www.bhv.ru

Евгений Угрюмов

Цифровая схемотехника

*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")*



Дюссельдорф • Киев • Москва • Санкт-Петербург

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. Описывается использование в схемотехнике стандартных элементов, типовых функциональных узлов и микросхем программируемой логики, которые, согласно прогнозам, в ближайшие годы произведут в цифровой схемотехнике такой же переворот, как микрокомпьютеры в 70-е гг.

Приведены структуры и схемотехника полупроводниковых запоминающих устройств, простых микропроцессоров и БИС/СБИС микропроцессорных комплектов. 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

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зав. редакцией	<i>Наталья Таркова</i>
Редактор	<i>Юрий Рожко</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн обложки	<i>Ангелины Лужиной</i>
Зав. производством	<i>Николай Тверских</i>

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)

Угрюмов Е. П.

Цифровая схемотехника. — СПб.: БХВ — Санкт-Петербург, 2000. — 528 с.: ил.

ISBN 5-8206-0100-9

© Е. П. Угрюмов, 2000

© Оформление, издательство "БХВ — Санкт-Петербург", 2000

Лицензия ЛР № 065953 от 15.06.98. Подписано в печать 14.07.00

Формат 70×100¹/₁₆. Печать офсетная. Усл. печ. л. 42,57.

Тираж 3000 экз. Заказ № 3370.

"БХВ — Санкт-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар, № 77.99.1.953.П.950.3.99 от 01.03.1999 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН.
199034, Санкт-Петербург, 9-я линия, 12.

Содержание

Предисловие	1
Введение.....	3
Глава 1. Схемотехнические проблемы построения цифровых узлов и устройств	5
§ 1.1. Простейшие модели и система параметров логических элементов.....	5
Простейшие модели логических элементов.....	5
Статические параметры логических элементов	6
Быстродействие логических элементов	7
Мощности потребления логических элементов.....	8
§ 1.2. Типы выходных каскадов цифровых элементов	9
Логический выход	9
Элементы с тремя состояниями выхода	10
Выход с открытым коллектором	11
Выход с открытым эмиттером	14
§ 1.3. Паразитные связи цифровых элементов по цепям питания.	
Фильтрация питающих напряжений в схемах ЦУ.....	15
§ 1.4. Передача сигналов в цифровых узлах и устройствах. Помехи в сигнальных линиях. Сигнальные линии повышенного качества	17
Перекрестные помехи.....	18
Искажения сигналов в несогласованных линиях	19
Параллельное согласование волновых сопротивлений.....	21
Последовательное согласование волновых сопротивлений.....	21
Линии передачи сигналов	22
§ 1.5. Вспомогательные элементы цифровых узлов и устройств	24
Элементы задержки.....	25
Формирование импульсов по длительности	28
Генераторы импульсов.....	29
Элементы индикации.....	30
§ 1.6. О некоторых типовых ситуациях при построении узлов и устройств на стандартных ИС.....	34
Режимы неиспользуемых входов	34
Режимы неиспользуемых элементов	36
Наращивание числа входов.....	36
Снижение нагрузок на выходах логических элементов	36

Глава 2. Функциональные узлы комбинационного типа.....	39
§ 2.1. Введение в проблематику проектирования ЦУ комбинационного типа	39
§ 2.2. Двоичные дешифраторы	46
Схемотехническая реализация дешифраторов.....	48
§ 2.3. Приоритетные и двоичные шифраторы.	
Указатели старшей единицы	50
§ 2.4. Мультиплексоры и демультиплексоры	54
§ 2.5. Универсальные логические модули на основе мультиплексоров	56
Первый способ настройки УЛМ	57
Второй способ настройки УЛМ.....	58
Пирамидальные структуры УЛМ.....	60
§ 2.6. Компараторы	64
§ 2.7. Схемы контроля	67
Мажоритарные элементы	69
Контроль по модулю 2.....	69
Схемы свертки	71
Передача данных с контролем по модулю 2	73
Контроль логического преобразователя	73
Контроль с использованием кодов Хемминга	73
Схемы кодера и декодера для кода Хемминга.....	76
§ 2.8. Сумматоры	77
Одноразрядный сумматор	78
Последовательный сумматор.....	80
Параллельный сумматор с последовательным переносом.....	81
Параллельный сумматор с параллельным переносом.....	82
Сумматоры групповой структуры	85
§ 2.9. Арифметико-логические устройства и блоки ускоренного переноса ..	90
§ 2.10. Матричные умножители	93
Множительно-суммирующие блоки	93
Схемы ускоренного умножения	96
Глава 3. Функциональные узлы последовательностного типа (автоматы с памятью)	101
§ 3.1. Триггерные устройства (элементарные автоматы). Классификация.	
Основные сведения.....	101
Классификация триггеров.....	102
Времена предустановки и выдержки.....	104
Способы описания триггеров	105
§ 3.2. Схемотехника триггерных устройств.....	107
§ 3.3. Аномальные состояния триггеров.....	116
§ 3.4. Применение триггеров в схемах ввода и синхронизации логических сигналов	117
Ввод логических сигналов от механических ключей	117
Синхронизаторы одиночных импульсов.....	119
Ввод асинхронных данных.....	120

§ 3.5. Введение в проблематику и методику проектирования автоматов с памятью.....	121
Проектирование автоматов	123
Пример проектирования	126
§ 3.6. Синхронизация в цифровых устройствах	132
Параметры тактовых импульсов	133
Структура устройств синхронизации	135
Размножение тактовых импульсов	136
Коррекция расфазирования синхросигналов	137
Однофазная синхронизация	138
Расчетные соотношения для проектирования однофазной системы синхронизации.....	140
Двухфазная синхронизация.....	142
§ 3.7. Регистры и регистровые файлы	143
Регистровые файлы	145
Сдвигающие регистры	146
Универсальные регистры.....	148
§ 3.8. Основные сведения о счетчиках. Двоичные счетчики.....	150
Классификация счетчиков	151
Двоичные счетчики.....	152
Счетчики с групповой структурой	155
§ 3.9. Двоично-кодированные счетчики с произвольным модулем.....	157
§ 3.10. Счетчики с недвоичным кодированием.....	162
Счетчики в коде Грея	162
Счетчики в коде "1 из N"	163
§ 3.11. Полиномиальные счетчики	170
Схемы генераторов псевдослучайной последовательности (ГПСП)	172
Глава 4. Запоминающие устройства	175
§ 4.1. Основные сведения. Система параметров. Классификация.....	175
Важнейшие параметры ЗУ.....	176
Классификация ЗУ.....	178
§ 4.2. Основные структуры запоминающих устройств.....	182
Структура 2D	182
Структура 3D	184
Структура 2DM.....	186
Память с последовательным доступом	187
Видеопамять	187
Буфер FIFO.....	189
Кэш-память.....	190
§ 4.3. Запоминающие устройства типа ROM(M), PROM, EPROM, EEPROM.....	195
Масочные ЗУ	195
ЗУ типа PROM	197
ЗУ типов EPROM и EEPROM.....	200
Импульсное питание ROM	204

§ 4.4. Флэш-память	205
Память типа Bulk Erase	209
Флэш-память с несимметричной блочной структурой	211
Файловая Флэш-память	213
Память типа StrataFlash	216
§ 4.5. Использование программируемых ЗУ для решения задач обработки информации	217
Реализация логических (переключательных) функций	218
Реализация конечных автоматов	219
Воспроизведение арифметических операций и функциональных зависимостей	219
§ 4.6. Статические запоминающие устройства	221
Запоминающие элементы статических ЗУ	222
Выходной каскад с третьим состоянием	223
Внешняя организация и временные диаграммы статических ЗУ	224
Искусственная энергонезависимость статических ЗУ	226
§ 4.7. Динамические запоминающие устройства — базовая структура	229
Запоминающие элементы	229
Усилители-регенераторы	232
Мультиплексирование шины адреса	233
Внешняя организация и временные диаграммы	233
Схема динамического ЗУ	234
§ 4.8. Динамические запоминающие устройства повышенного быстродействия	236
Вариант FPM	237
Структуры типа EDORAM	238
Структуры типа BEDORAM	238
Структура типа MDRAM	239
Структуры типа SDRAM	239
Структуры типа RDRAM	241
Структура DRDRAM	242
Структура типа CDRAM	243
§ 4.9. Регенерация данных в динамических запоминающих устройствах	243
§ 4.10. Заключительные замечания	245
Глава 5. Микропроцессорные бис/сбис и их применение в микропроцессорных системах	249
§ 5.1. Микропроцессорные комплекты БИС/СБИС. Структура и функционирование микропроцессорной системы. Микроконтроллеры	249
Микроконтроллеры	252
§ 5.2. Управление памятью и внешними устройствами. Построение модуля памяти	255
Модуль памяти	257
Сигналы управления	258
Виды обмена	259

§ 5.3. Микропроцессор серии 1821 (Intel 8085A)	260
Структура микропроцессора K1821BM85A	260
Блок регистров	262
Синхронизация и последовательность действий МП	267
Система прерываний	270
Система команд МП	273
Пример выполнения команды	281
§ 5.4. Схемы подключения памяти и внешних устройств	
к шинам микропроцессорной системы	282
Анализ нагрузочных условий	289
Согласование временных диаграмм МП и ЗУ	290
Схемы реализации безусловного программного ввода/вывода	293
Схемы реализации условного программного ввода/вывода	296
Глава 6. Интерфейсные бис/сбис микропроцессорных комплектов	299
§ 6.1. Интерфейсы микропроцессорных систем	299
Интерфейс (шина) Microbus	299
Интерфейс И-41	300
Интерфейс МПИ	301
§ 6.2. Шинные формирователи и буферные регистры	302
Шинные формирователи	302
Буферные регистры	304
§ 6.3. Параллельные периферийные адаптеры	306
Режим 0	309
Режим 1	309
Режим 2	312
§ 6.4. Программируемые связанные адаптеры	315
Структура ПСА	320
Выводы и сигналы ПСА	321
Передачик ПСА	323
Приемник ПСА	323
§ 6.5. Программируемые контроллеры прерываний	329
Вложенные прерывания с фиксированными приоритетами входов	329
Прерывания с круговым (циклическим) приоритетом	329
Структура ПКП	331
Каскадное включение контроллеров	337
§ 6.6. Контроллеры прямого доступа к памяти	338
Структура и функции КПДП	340
Выводы и сигналы контроллера	345
§ 6.7. Программируемые интервальные таймеры	348
Структура ИС ВИ54	348

Глава 7. Программируемые логические матрицы, программируемая матричная логика, базовые матричные кристаллы357

§ 7.1. Вводные замечания	357
--------------------------------	-----

§ 7.2. Программируемые логические матрицы и программируемая матричная логика (ПЛМ и ПМЛ).....	358
Программируемые логические матрицы	358
Схемотехника ПЛМ	359
Подготовка задачи к решению с помощью ПЛМ	361
Программирование ПЛМ	362
Упрощенное изображение схем ПЛМ	363
Воспроизведение скобочных форм переключательных функций.....	364
Программируемая матричная логика.....	367
Функциональные разновидности ПЛМ и ПМЛ.....	368
Схемы с программируемым выходным буфером.....	368
Схемы с двунаправленными выводами.....	370
Схемы с памятью.....	371
ПМЛ с разделяемыми конъюнкторами	372
ПМЛ серии K1556.....	373
Пример подготовки задачи к решению с помощью ПМЛ.....	375
Пример более сложной структуры PLD.....	377
§ 7.3. Базовые матричные кристаллы (вентильные матрицы с масочным программированием).....	380
Классификация БМК	383
Параметры БМК.....	388
Глава 8. Современные и перспективные БИС/СБИС со сложными программируемыми и репрограммируемыми структурами (FPGA, CPLD, FLEX, SOC и др.)	391
§ 8.1. Общие сведения	391
Классификация по конструктивно-технологическому типу программируемых элементов	392
§ 8.2. Программируемые пользователем вентильные матрицы (FPGA)	397
Логические блоки FPGA	398
Блоки ввода/вывода FPGA.....	402
Системы межсоединений FPGA.....	404
Области применения FPGA и других СБИС ПЛ	410
Построение реконфигурируемых систем.....	410
Задачи логической эмуляции.....	410
Построение динамически реконфигурируемых систем	411
Обогащение цифровой элементной базы	412
§ 8.3. Сложные программируемые логические схемы (CPLD) и СБИС программируемой логики смешанной архитектуры (FLEX и др.).....	412
Функциональные блоки CPLD	413
Системы коммутации CPLD	413
CPLD MAX 7000.....	415
Микросхемы семейства FLEX 10K	421
Логический элемент	423

§ 8.4. СБИС программируемой логики типа "система на кристалле"	425
Семейство СБИС типа APXH 20K/KE.....	427
Семейство СБИС типа Virtex.....	428
§ 8.5. Параметры и популярные семейства СБИС программируемой логики	431
Уровень интеграции (сложность)	431
Быстродействие СБИС.....	432
§ 8.6. Интерфейс JTAG. Периферийное сканирование. Программирование в системе (ISP). Конфигурирование СБИС ПЛ.....	439
Интерфейс JTAG и периферийное сканирование.....	439
Программирование в системе.....	442
Требования к числу допустимых для микросхемы циклов репрограммирования.....	443
Настройка микросхем программируемой логики.....	443
Глава 9. Методика и средства проектирования цифровых устройств.....	445
§ 9.1. Общие сведения	445
Классификация цифровых ИС с точки зрения методов проектирования	446
Области применения СпИС различных типов	448
§ 9.2. Пример "ручного" проектирования цифрового устройства с использованием программируемой матричной логики (ПМЛ)	451
Первый этап проектирования.....	454
Второй этап проектирования.....	455
Третий этап проектирования	455
Четвертый этап	457
Последний этап проектирования.....	457
§ 9.3. Методика и средства автоматизированного проектирования цифровых устройств	458
Средства описания проекта.....	459
Языки низкого уровня	460
Языки высокого уровня.....	460
Разделение устройства на операционный блок и блок управления	460
Этапы проектных процедур	461
Основные сведения о языке VHDL	464
Синтаксические конструкции и основные понятия языка	465
Описание проекта на языке VHDL.....	466
Примеры поведенческих описаний элементов на языке VHDL	467
Структурный и поведенческий варианты описания проекта.....	468
О возможностях и средствах описания типовых узлов цифровой техники....	469
§ 9.4. Пример автоматизированного проектирования цифрового устройства с использованием языков описания аппаратуры	473
Первый этап. Рассмотрение ТЗ на разрабатываемое устройство	473
Второй этап. Разработка общей структуры операционного блока	474
Третий этап. Описание работы управляющего автомата	476
Пояснения к синтаксису AHDL и VHDL программ устройства управления	478

Четвертый этап. Компиляция проекта и основные параметры устройства...	484
Пятый этап. Тестирование проекта.....	485
Шестой этап. Автоматическое определение временных характеристик устройства.....	487
Седьмой этап. Практическое использование результатов проектирования ...	487
Глоссарий	489
Дополнение. Словарь иностранных сокращений и терминов	502
Принятые сокращения.....	507
Литература	511
Предметный указатель	515

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 около 2/3 имевшихся мощностей. Перспективы отечественной микроэлектроники предсказать сложно, поскольку вывод ее на современный уровень требует вложения огромных средств. Государственная поддержка на требуемом уровне пока не реальна (даже крупнейшие мировые фирмы вынуждены объединяться в альянсы для аккумуляирования 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. Крупнейший специалист в области информатики академик Е. П. Велихов в одной из своих статей¹ присоединился к тезису: *"Тот, кто умеет делать компьютеры, владеет миром"*.

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

¹ Газета "Известия", 25 ноября 1999 г.

действиями над двоичными числами, а также с принципами работы транзисторов и транзисторных переключающих каскадов.

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.

В пособии рассмотрен широкий круг вопросов, связанных с изучением и применением современной элементной базы цифровой техники, что соответствует основному содержанию дисциплины "Схемотехника ЭВМ" специальности 2201. Учебники и учебные пособия по этой тематике не издавались уже более 10 лет. Исключение составляет учебное пособие Г. И. Пухальского и Т. Я. Новосельцевой "Цифровые устройства" (издательство Политехника, СПб, 1996 г., 885 с.), отличающееся от данного пособия как по характеру изложения, так и по тематике, поскольку в нем отсутствуют разделы, посвященные схемам памяти, микропроцессорам, схемам программируемой логики и др.

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

Параграф 4.2 написан автором совместно с А. Х. Мурсаевым, глава 9 совместно с Р. И. Грушвицким и А. Н. Альшевским.

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 могут быть реализованы 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.

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, так как их универсальность снижается. В развитых сериях стандартных ИС насчитываются сотни типов СИС.

С появлением БИС и СБИС схемы с тысячами и даже миллионами логических элементов стали размещаться на одном кристалле. 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, так как 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 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 возможность 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 происходит без разбиения этого процесса на последовательно выполняемые элементарные действия. Задача решается "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.

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 изготавливаются по схемотехнологиям КМОП, ТТЛШ и др. Элементы КМОП обладают рядом уникальных параметров (малая потребляемая мощность при невысоких частотах переключения, высокая помехоустойчивость, широкие допуски на величину питающих напряжений, высокое быстродействие при небольших емкостных нагрузках). Эти элементы доминируют в схемах внутренних областей БИС/СБИС. За ТТЛШ осталась в основном область периферийных схем, где требуется передача сигналов по внешним цепям, испытывающим значительную емкостную нагрузку. Элементы ЭСЛ (эмиттерно-связанная логика) обеспечивают максимальное быстродействие, но ценой повышения потребляемой мощности, что снижает достижимый уровень интеграции.

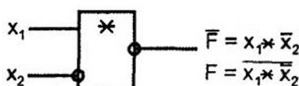
Глава 1

Схемотехнические проблемы построения цифровых узлов и устройств

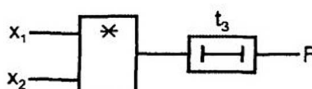
§ 1.1. Простейшие модели и система параметров логических элементов

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.



а



б

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 опера-

ции 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_3 (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 и т. д. Например, для элементов КМОП задержка пропорциональна емкости нагрузки. Простейшая модель не учитывает также фильтрующих свойств реальных элементов, благодаря которым короткие входные импульсы, обладающие малой энергией, не способны вызвать переключение элемента

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

Для правильного проектирования и эксплуатации ЦУ необходимо знать систему параметров логических элементов (статических и динамических)

Статические параметры логических элементов

В качестве важнейших статических параметров приводятся четыре значения напряжений и четыре значения токов.

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 $U_{\text{вх.1.min}}$ и $U_{\text{вх.0.max}}$. Входные напряжения данного элемента есть выходные напряжения предыдущего (источника сигналов). Уровни, гарантируемые на выходе элемента при соблюдении допустимых нагрузочных условий, задаются параметрами $U_{\text{вых.1.min}}$ и $U_{\text{вых.0.max}}$. Выходные уровни несколько "лучше" входных, что обеспечивает определенную помехоустойчивость элемента. Для уровня U_1 опасны отрицательные помехи, снижающие его, причем допустимая статическая помеха (т. е. помеха любой длительности)

$$U_{\text{пом}}^- = U_{\text{вых.1.min}} - U_{\text{вх.1.min}}.$$

Для уровня U_0 опасны положительные помехи, причем допустимая статическая помеха

$$U_{\text{пом}}^+ = U_{\text{вых.0.max}} - U_{\text{вх.0.max}}.$$

Четыре значения токов — входные и выходные токи в обоих логических состояниях. При высоком уровне выходного напряжения из элемента — источника ток вытекает, 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 $I_{\text{вых.1.max}}$ и $I_{\text{вых.0.max}}$, характеризующие возможности элемента — источника сигнала, и токи $I_{\text{вх.1.max}}$ и $I_{\text{вх.0.max}}$, 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. 1.2) показывают длительности характерных этапов переходных процессов, 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 $t_3 = 0,5(t^{10} + t^{01})$.

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 опре-

деленным условиям измерений, указанным в справочниках. Если условия работы элемента отличаются от условий измерения, то может потребоваться коррекция справочных данных.

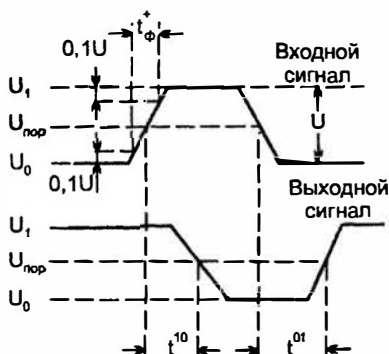


Рис. 1.2. Временные диаграммы процессов переключения логического элемента

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 $t_3 = t_{3н} + k\Delta C$, где $t_{3н}$ — 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 и конструкции 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. Обычно не возникает трудностей при подсчете мощностей, потребляемых биполярными схемами. При подсчете мощностей, потребляемых элементами типа КМОП, положение намного сложнее и данных, приведенных в справочниках, может не хватить. Здесь следует отметить, что в настоящее время только справочник под редакцией И. И. Петровского [21] предоставляет удобные данные для расчета мощностей ЦУ на элементах КМОП (для серии элементов КР1554).

§ 1.2. Типы выходных каскадов цифровых элементов

Цифровые элементы (логические, запоминающие, буферные) могут иметь выходы следующих типов: логические, с открытым коллектором (стоком), с третьим состоянием, с открытым эмиттером (исток).

Наличие четырех типов выходов объясняется различными условиями работы элементов в логических цепях, в магистрально-модульных микропроцессорных 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 комбинационных цепях.

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. 1.3, а).

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 ноль, не будет нормального результата. Во-вторых, при соединении выходов, находящихся в различных логических состояниях, возникло бы их "противоборство". Вследствие малых величин выходных сопротивлений уравнивающий ток при этом может достигать достаточно большой величины, что может вывести из строя электрические элементы выходной цепи.

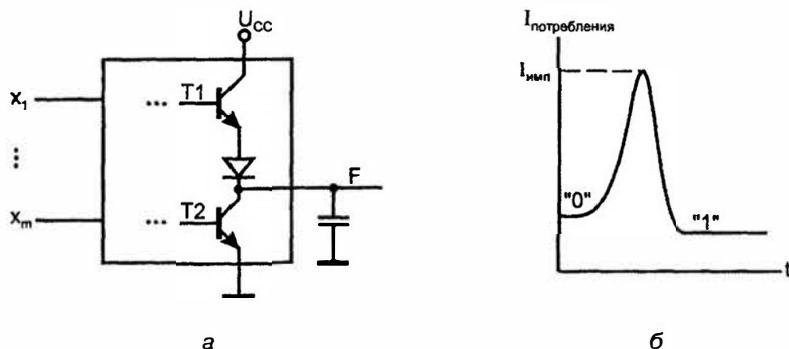


Рис. 1.3. Схема выходной цепи цифрового элемента (а) и график изменения потребляемого им тока в процессе переключения (б)

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 T1 и T2 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. 1.3, б).

Элементы с тремя состояниями выхода

Элементы с тремя состояниями выхода (типа ТС) кроме логических состояний 0 и 1 имеют состояние "отключено", в котором ток выходной цепи пренебрежимо мал. В это состояние (третье) элемент переводится специальным управляющим сигналом, обеспечивающим запертое состояние обоих транзисторов выходного каскада (T1 и T2 на рис. 1.3, а). 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". 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 — H-активными или L-активными, что ведет к наличию четырех типов буферных каскадов (рис. 1.4).

Выходы типа ТС отмечаются в обозначениях элементов значком треугольника, как на рис. 1.4, 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.

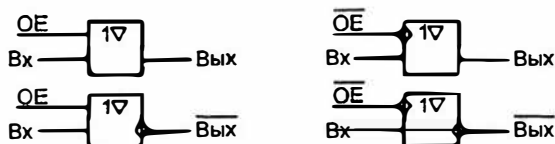


Рис. 1.4. Типы буферных каскадов с третьим состоянием

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 $OE_1 + OE_2 + \dots + OE_n \leq 1$ при объединении n выходов. Нарушение этого условия может привести даже к выходу из строя самих элементов.

Выход с открытым коллектором

Элементы с открытым коллектором имеют выходную цепь, заканчивающуюся одиночным транзистором, коллектор которого не соединен с какими-либо цепями внутри микросхемы (рис. 1.5, а). Транзистор управляется от предыдущей части схемы элемента так, что может находиться в насыщенном или запертом состоянии. Насыщенное состояние трактуется как отображение логического нуля, запертое — единицы.

Насыщение транзистора обеспечивает на выходе напряжение U_0 (малое напряжение насыщения "коллектор-эмиттер" $U_{кэп}$). Запирание же транзистора какого-либо уровня напряжения на выходе элемента не задает, выход при этом имеет фактически неизвестный "плавающий" потенциал, т. к. не подключен к каким-либо цепям схемы элемента. Поэтому для формирования высокого уровня напряжения при запирании транзистора на выходе элементов с открытым коллектором (типа ОК) 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. 1.5, б). 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 эле-

ментов, выходы которых соединены, то можно получить дополнительную логическую операцию, называемую операцией монтажной логики.

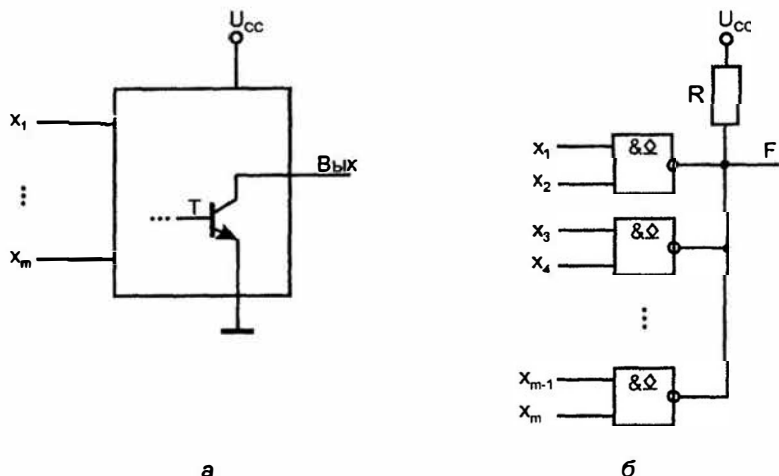


Рис. 1.5. Схема выходной цепи цифрового элемента с открытым коллектором (а) и реализации монтажной логики (б)

При реализации монтажной логики высокое напряжение на общем выходе возникает только при запираании всех транзисторов, т. к. насыщение хотя бы одного из них снижает выходное напряжение до уровня $U_0 = U_{кэл}$. 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

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

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

При использовании элементов с ОК 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 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 входе разрешит работу элемента.

Положительной чертой элементов с ОК при работе в магистрально-модульных системах является их защищенность от повреждений из-за ошибок управления, приводящих к одновременной выдаче на 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 происходит заряд выходной емкости 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_3^{01} .

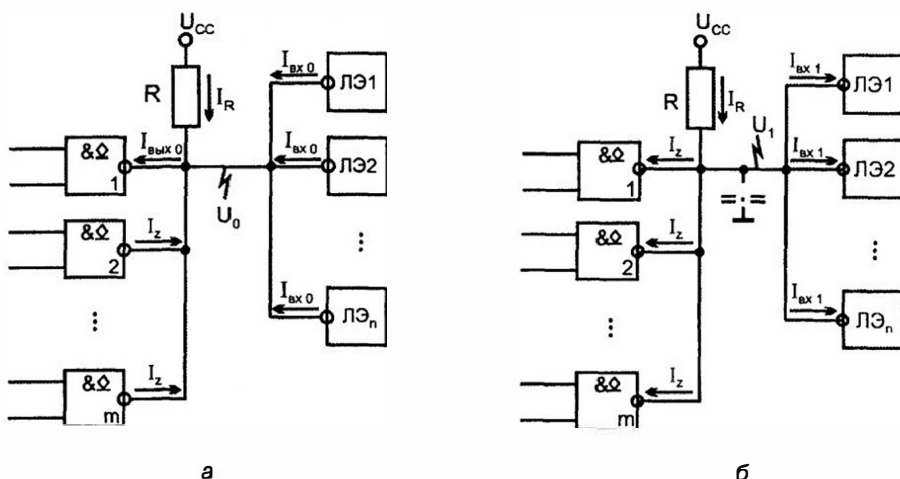


Рис. 1.6. Схемы для расчета минимального (а) и максимального (б) значений сопротивления внешней цепи в каскадах с открытым коллектором

При работе с элементами типа ОК проектировщик должен задать сопротивление резистора R , которое не является стандартным, а определяется для конкретных условий. Анализ статических режимов задает ограничения величины сопротивления R снизу и сверху. Значение сопротивления резистора R выбирается в этом диапазоне с учетом быстродействия схемы и потребляемой ею мощности.

Ограничение снизу величины сопротивления резистора R связано с тем, что ее уменьшение может вызвать перегрузку насыщенного транзистора по току. На рис. 1.6, а показан режим, в котором нулевое состояние выхода схемы обеспечивается элементом 1 с ОК. Из этого рисунка видно, что через выход

элемента протекает суммарный ток, складывающийся из токов резистора, входных токов логических элементов ($I_{Э1}...I_{Эn}$) и токов выходов запертых транзисторов элементов с ОК $2...m$, т. е.

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

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

$$R \geq (U_{\text{сс}} - U_0)/(I_{\text{вх } 0 \text{ max}} - nI_{\text{вх } 0 \text{ max}}).$$

Ограничение сверху величины сопротивления резистора R связано с необходимостью гарантировать достаточно высокий уровень напряжения U_1 , формируемого в схеме при запертом состоянии всех выходов элементов с ОК. Из схемы (рис. 1.6, б) видно, что $U_1 = U_{\text{сс}} - I_R R$.

В то же время $I_R = mI_z + nI_{\text{вх } 1 \text{ max}}$.

Из полученных выражений следует

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

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

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

Выход с открытым эмиттером

Выход с открытым эмиттером характерен для элементов типа ЭСЛ. Для работы на магистраль такие элементы не используются. Возможность соединять друг с другом выходы с открытым эмиттером при объединении эмиттерных резисторов в один общий резистор приводит к схеме рис. 1.7, иногда называемой "эмиттерный дот" и используемой при построении логических схем для получения дополнительной операции монтажной логики. Элементы ЭСЛ имеют противофазные выходы, на одном из которых реализуется функция ИЛИ, на другом — 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

$$F = \overline{x_1} \sqrt{x_2} \sqrt{x_3} \sqrt{x_4} = \overline{x_1} \overline{x_2} \sqrt{x_3} \overline{x_4}.$$

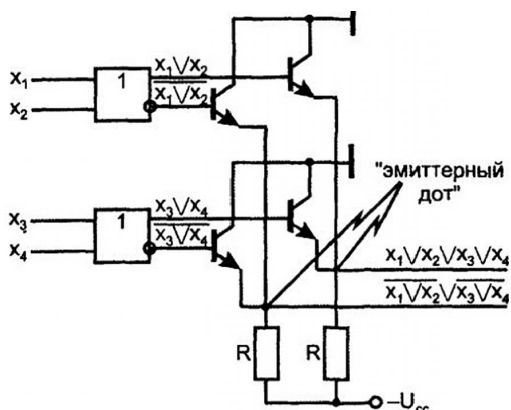


Рис. 1.7. Схема "эмиттерного дота"

Соединяя прямой выход с инверсным, можно получить функцию вида

$$F = x_1 \vee x_2 \vee \overline{x_3} \vee \overline{x_4} = x_1 \vee x_2 \vee \overline{x_3} \overline{x_4}$$

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

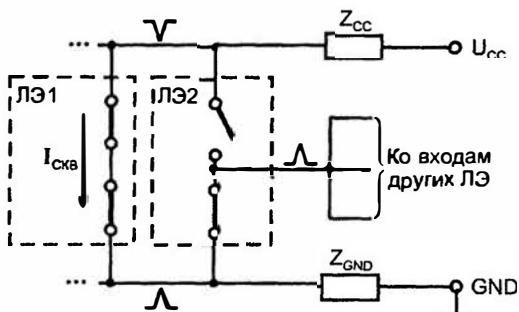
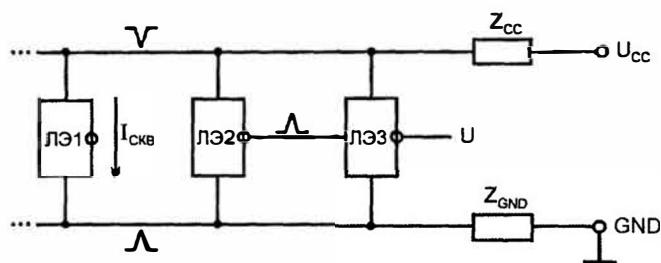
Одной из важнейших задач при проектировании и эксплуатации ЦУ является борьба со сбоями из-за помех. Типовой проблемой здесь является, в частности, наличие токовых импульсов в цепях питания ИС.

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.

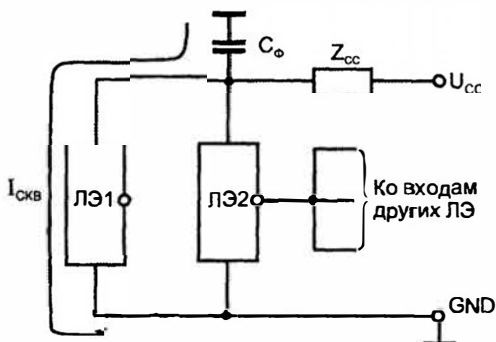
Для определенности далее будем говорить о сквозных токах, хотя практически то же самое можно говорить и о токах перезаряда емкостей.

Импульс сквозного тока переключающегося элемента 1 (рис. 1.8, 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 полное сопротивление 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

1 элемент 2. Если, как показано на рисунке, элемент 2 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 2, откуда сможет распространяться и далее по обычным сигнальным цепям. При единичном состоянии элемента 2 на его выход пройдет отрицательный импульс помехи с линии источника питания.



а



б

Рис. 1.8. Схемы, поясняющие процесс возникновения импульсных помех при переключении цифрового элемента (а), и пути протекания сквозного тока при наличии в схеме фильтрующего конденсатора (б)

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.

Для шин питания схемы наряду с конструктивными методами применяют и схемотехнические: 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 обрабатывать короткие импульсные помехи. Поэтому для коротких помех выходное сопротивление источника не обеспечивает того низкого уровня, которое оно имеет в статике. Установка фильтрующих конденсаторов $C_{\text{ф}}$ создает путь (рис. 1.8, 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 руководящих материалах по применению конкретных типов ИС.

§ 1.4. 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, электромагнит-

ные наводки и паразитные колебания из-за несогласованности волновых сопротивлений линий связи.

Перекрыстные помехи

Перекрыстные помехи 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 $C_{\text{пом}}$ (рис. 1.9, а). Схема замещения рассматриваемой цепи может быть представлена в виде рис. 1.9, б, где

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

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

$$a = (U_1 - U_0)/t_{\text{ф}} = U/t_{\text{ф}},$$

то напряжение помехи на входе элемента ЛЭ2 будет определяться соотношением (для времен от нуля до $t_{\text{ф}}$)

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

т. е. пропорционально крутизне фронта.

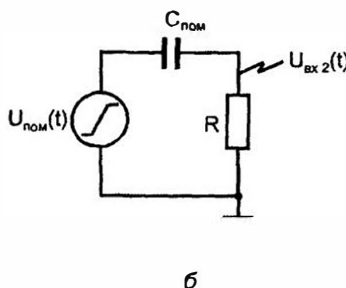
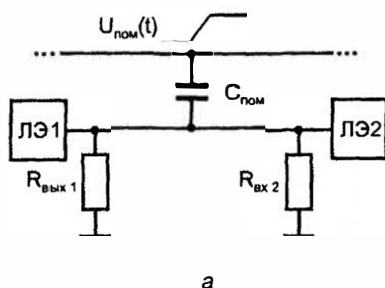


Рис. 1.9. Схема, поясняющая процесс возникновения перекрыстных помех в цифровых устройствах (а), и схема замещения (б)

Борьба с перекрыстными 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.

Скорость распространения сигнала \bar{v} в линии равна $V = V_c / \sqrt{\epsilon}$, где V_c — скорость света в вакууме (30 см/нс); ϵ — диэлектрическая постоянная среды, в которой распространяется сигнал. Практически $V = 15 \dots 20$ см/нс. Поведение длинной линии резко отличается от поведения короткой.

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. 1.10, а) имеет волновое сопротивление $Z_0 = \sqrt{L/C}$, величина которого зависит от конструкции линии. 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 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

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

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. 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).

Для устранения паразитных колебаний в длинной линии используют параллельное или последовательное согласование волновых сопротивлений

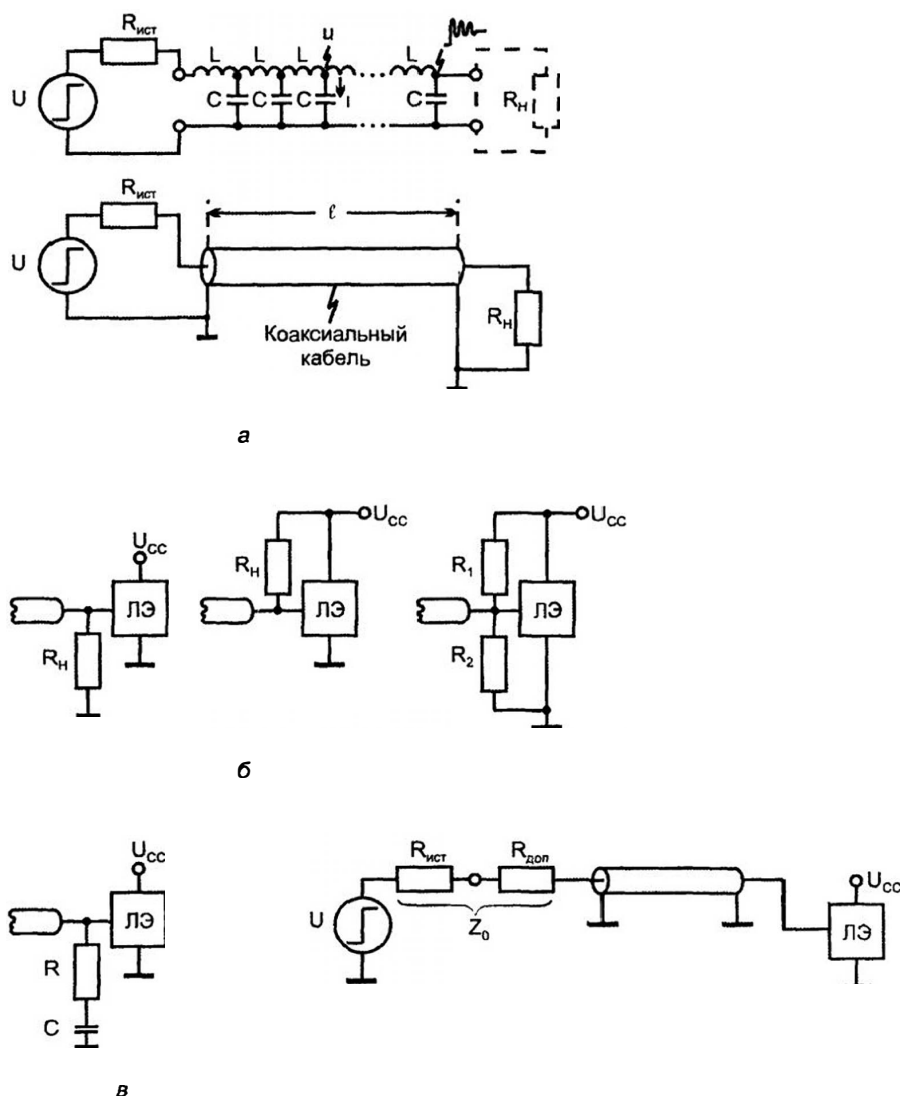


Рис. 1.10. Схема замещения длинной линии без потерь и схема с реализацией линии в виде коаксиального кабеля (а), варианты согласования волновых сопротивлений при передаче цифровых сигналов (б), (в), (г)

Параллельное согласование волновых сопротивлений

При параллельном согласовании в конце линии включают резистор (иногда называемый терминатором), чтобы сделать сопротивление нагрузки линии 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 должно установиться напряжение U_1 или U_0 , в зависимости от логического состояния элемента — источника сигнала. Под этим напряжением находится резистор-терминатор, сопротивление которого мало (типичные значения волновых сопротивлений линий передачи сигналов 50...100 Ом). Ток через резистор-терминатор может оказаться неприемлемо большим. Для поиска наиболее подходящего варианта включения резистора на выходе линии можно просмотреть несколько схемных вариантов (рис. 1.10, б). Пользуются также включением последовательно с резистором емкости C , которая предотвращает потребление тока в статике (рис. 1.10, в).

Последовательное согласование волновых сопротивлений

При последовательном согласовании в начале линии последовательно включается резистор $R_{\text{доп}}$, сопротивление которого совместно с выходным сопротивлением источника сигнала $R_{\text{ист}}$ дает величину Z_0 (рис. 1.10, г). При этом на выходе линии действует высокое входное сопротивление элемента-приемника, следовательно, там коэффициент отражения приблизительно равен единице, и амплитуда отраженной волны приблизительно равна амплитуде падающей.

Переходный процесс в этом случае протекает следующим образом.

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

При последовательном согласовании отсутствуют токи нагрузки на источник сигнала, характерные для параллельного согласования. Повышенное значение сопротивления в цепи передачи сигнала может уменьшать амплитуду передаваемых напряжений, 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 циф-

ровых элементов зачастую непостоянны и зависят от логического состояния элемента, уровня сигнала и т. д. То же самое можно сказать и о входных сопротивлениях элементов.

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 выше ("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.

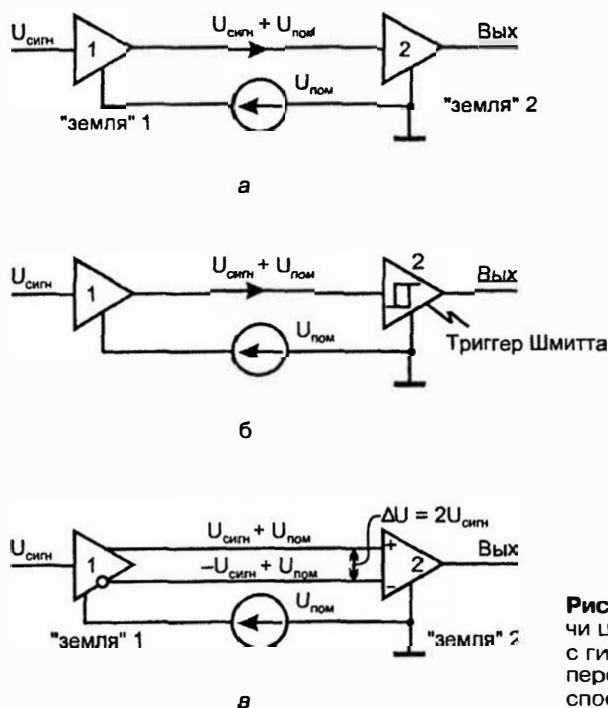
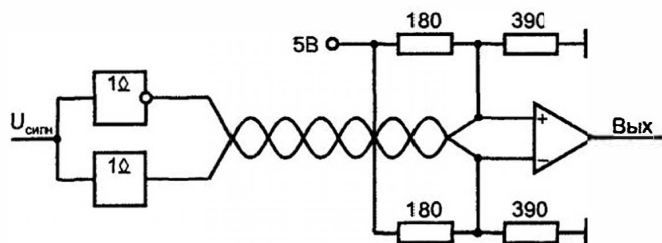
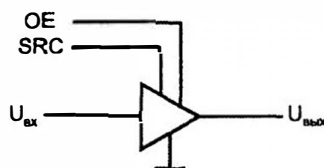


Рис. 1.11. Простейшая схема передачи цифрового сигнала (а), схема с гистерезисным приемником (б), передача сигнала дифференциальным способом (в)



г



д

Рис. 1.11. (окончание) Пример схемы помехоустойчивой передачи сигнала (г), буфер с регулируемой крутизной фронта (д)

Схема соединения одиночным проводником (рис. 1.11, а) изображена с учетом напряжения помехи, которая может возникнуть между "землями" двух элементов. В этом случае помеха передается на вход приемника сигнала.

Помехоустойчивость передачи повышается, если элемент-приемник обладает гистерезисными свойствами, как, например, триггер Шмитта (рис. 1.11, 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. 1.11, в. Приемником сигнала служит дифференциальный усилитель (или компаратор). На его верхнем входе действует напряжение $U_{\text{сигн}} + U_{\text{пом}}$, а на нижнем $-U_{\text{сигн}} + U_{\text{пом}}$. Дифференциальный приемник воспринимает разность напряжений между входами, которая равна $2U_{\text{сигн}}$ 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. 1.11, 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 $2 \times 0,05 \text{ мм}^2$: 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. 1.11, д изображен буфер с третьим состоянием и регулировкой крутизны нарастания выходного сигнала. Введением/снятием третьего состояния управляет вход ОЕ (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 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 соответствует 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 в длинных связях иногда включают через определенные расстояния усилители-повторители сигнала. Для оценки положения, начиная с которого основная доля задержки приходится на проводник, приведем цифры для технологии с минимальным размером 0,5 мкм: это 0,01; 0,02 и 0,5 мм соответственно для поликремниевых, диффузионных и металлизированных проводников.

§ 1.5. Вспомогательные элементы цифровых узлов и устройств

К числу вспомогательных отнесем элементы, не выполняющие логические операции или запоминание данных, но 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 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 последовательной 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 ограничивается разбросом собственных задержек элементов и невысока.

Задержку на большее время можно получить с помощью RC-цепочки, включаемой в цепь передачи сигнала (рис. 1.12), 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 английского *delay*, что означает задержку). После RC-цепочки в схеме включены три инвертора для формирования достаточно крутых фронтов на выходе элемента задержки.

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.

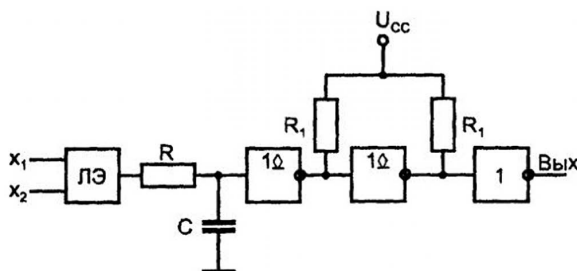


Рис. 1.12. Схема задержки с RC-цепочкой

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 (рис. 1.12) первые элементы цепи формирования имеют выход с ОК, 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 время-задающих 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 одновиб-

ратора, 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.

Сравнительно недавно в номенклатуре ИС появились специальные элементы задержки. На рис. 1.13 показан фрагмент схемы такого элемента, предназначенный для задержки отрицательного фронта. Положительные фронты входного импульсного сигнала задерживаются другой схемой подобного вида.

В схеме (рис. 1.13) в исходном состоянии высокий уровень входного напряжения насыщает транзистор T , и на неинверсный вход 2 дифференциального усилителя-компаратора поступает малое напряжение "коллектор-эмиттер" этого транзистора. На инверсный вход 1 поступает более высокое напряжение с делителя, образованного резисторами R_2 и R_{np} , причем в схеме имеется возможность регулирования этого напряжения, т. к. сопротивление R_{np} может программироваться пропусканием через него тока I_{np} . После завершения режима программирования значение R_{np} остается неизменным.

Поступление отрицательного фронта входного напряжения запирает транзистор T , и емкость начинает заряжаться от источника питания через резистор R_1 с постоянной времени R_1C . Когда напряжение на емкости достигнет напряжения, установленного на верхнем входе усилителя-компаратора, он переключится и выработает выходной сигнал.

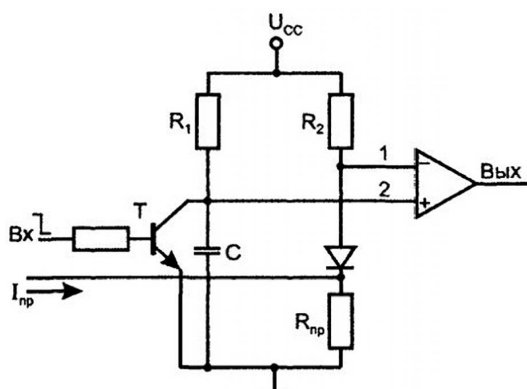


Рис. 1.13. Фрагмент схемы интегрального элемента задержки

В номенклатуре отечественных ИС появились три элемента задержки на 100, 125 и 150 нс с пятью равномерными отводами у каждого.

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 реализуются схемой (рис. 1.14, а). Если конкретизировать функцию F , считая ее дизъюнкцией, то, как видно из временных диаграмм на рис. 1.14, б, схема будет расширять входной импульс на интервал, равный времени задержки t_d . Если понимать под функцией F конъюнкцию и рассмотреть временные диаграммы (рис. 1.14, 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. 1.14, г. Заметим, что схема при $F = x_1 \bar{x}_2$ может быть заменена сочетанием обычного конъюнктора и инвертирующей задержки.

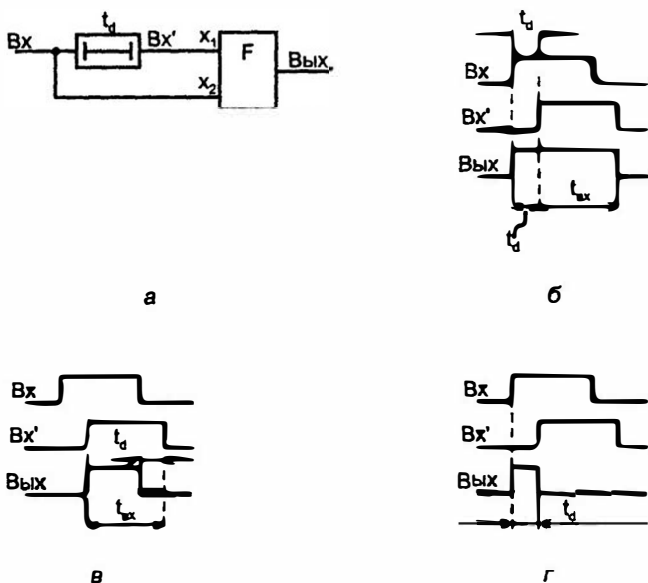


Рис. 1.14. Схема формирования импульса по длительности (а) и временные диаграммы реализации операций расширения (б), сужения (в) и стандартизации (г) импульсов

Генераторы импульсов

На элементах задержки и логических элементах строятся генераторы импульсных последовательностей. Простейший вариант показан на рис. 1.15, 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. 1.15, б).

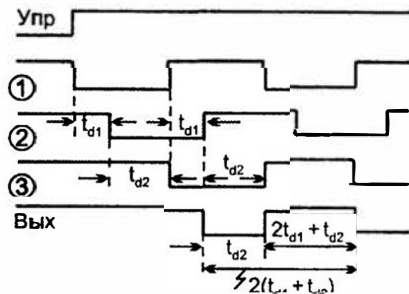
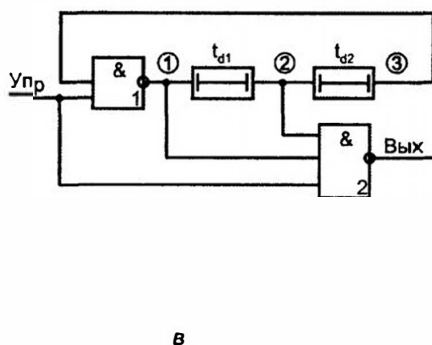
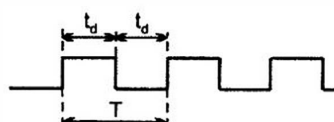
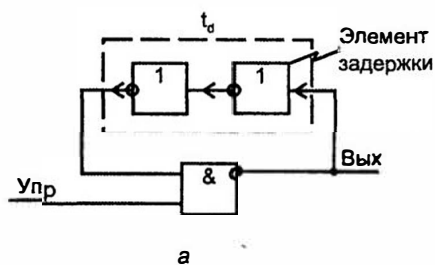


Рис. 1.15. Схемы генераторов симметричных (а) и несимметричных (в) импульсов и соответствующие временные диаграммы их выходных сигналов (б, г)

Очень часто требуются импульсы, в которых длительности импульса и паузы должны быть различны. На рис. 1.15, в показана схема, в которой возможно отдельное задание длительностей импульса и паузы.

Работу схемы легко уяснить из рассмотрения временных диаграмм на рис. 1.15, г. Видно, что длительность паузы устанавливается элементом задержки 2, после чего можно задать необходимую длительность импульса элементом задержки 1. При этом $t_n = t_{d2}$ и $t_n = 2t_{d1} + t_{d2}$. 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 генераторов, рассмотренных выше.

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

Элементы индикации

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 изготавливаются 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 двадцати 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 ко-

торых семь сегментов-диодов расположены так, что при зажигании определенной их комбинации высвечивается тот или иной символ (рис. 1.16, а).

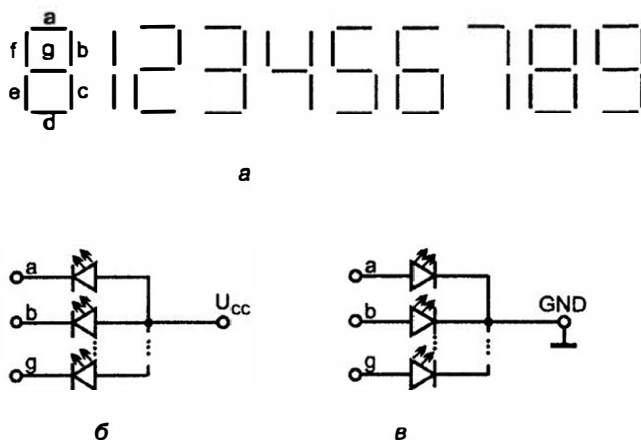


Рис. 1.16. Семисегментный индикатор и отображаемые им цифры (а), варианты индикатора с общим анодом (б) и общим катодом (в)

Выпускаются семисегментные индикаторы (ССИ) с общим анодом или общим катодом (рис. 1.16, 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, поскольку при их использовании имеется внешняя цепочка с резистором, сопротивление которого можно задать с учетом характеристик применяемых светодиодов.

В схеме (рис. 1.17, а) показано управление одним из сегментов ССИ. Диод зажигается, когда на выходе управляющего элемента напряжение равно U_0 . Через диод будет протекать ток $I_d = (U_{CC} - U_d - U_0)/R$, следовательно для его задания требуется условие $R = (U_{CC} - U_d - U_0)/I_d$. Для этой схемы требуются ССИ с общим анодом. Необходим управляющий элемент с достаточно большим выходным током в нулевом состоянии ($I_{\text{вых.0}} \geq I_d$).

В схеме (рис. 1.17, б) диод зажигается, когда выходной транзистор управляющего элемента запирается. Через диод течет ток $I_d = (U_{CC} - U_d)/R$, откуда следует $R = (U_{CC} - U_d)/I_d$. Для этой схемы требуется ССИ с общим катодом. Выход управляющего элемента должен удовлетворять условию $I_{\text{вых.0}} \geq (U_{CC} - U_0)/R$.

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. 1.17, *в, г*

Для логического управления ССИ имеются стандартные ИС-дешифраторы ССИ, работающие согласно табл. 1.1.

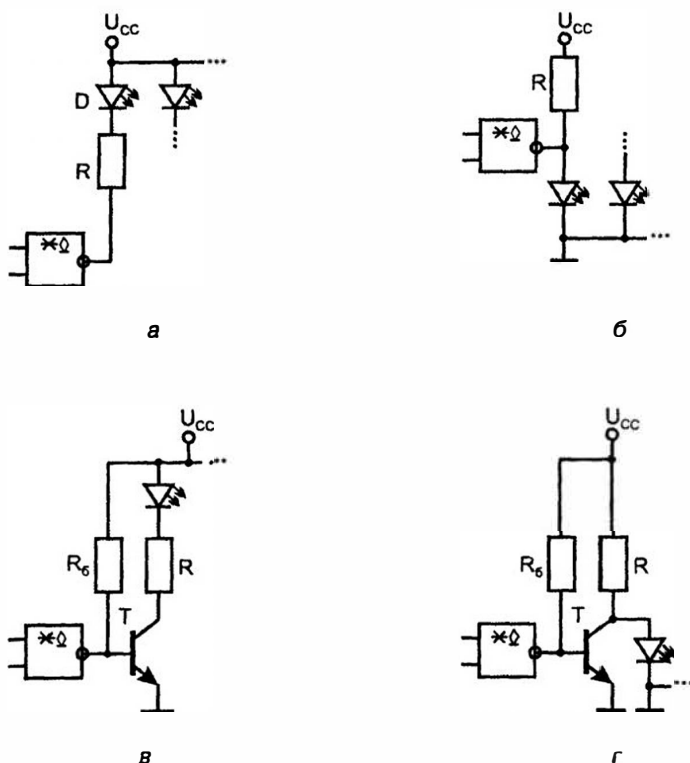


Рис. 1.17. Схемы управления сегментом индикатора с общим анодом (*а*), общим катодом (*б*) и использованием усилительных каскадов (*в, г*)

Таблица 1.1

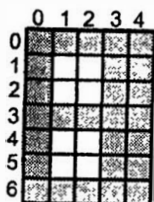
Десятичная цифра	Входной двоичный код	Возбуждаемые сегменты						
		a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0

Таблица 1.1 (окончание)

Десятичная цифра	Входной двоичный код	Возбуждаемые сегменты						
		a	b	c	d	e	f	g
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

Второй тип индикаторов, имеющих обычные для ИС уровни управляющих сигналов, — жидкокристаллические. 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.

На основе светодиодов или жидкокристаллических индикаторов изготавливаются 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. 1.18 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. 1.18. 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 человеческого зрения. Если частота возбуждения символов составляет десятки герц (современные средства визуальной индикации имеют частоты в 70...100 Гц), то мерцания изображений неощутимы.

В отличие от светодиодных, жидкокристаллические индикаторы не светятся. В темноте они не видны. В них под действием электрических полей меняются лишь свойства отражения света, благодаря чему и можно видеть отображаемые символы.

§ 1.6. 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 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. 1.19, а). Если же учесть особенности той или иной схемотехнологии, то выбор варианта действий становится определенным.

Для ЭСЛ 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.

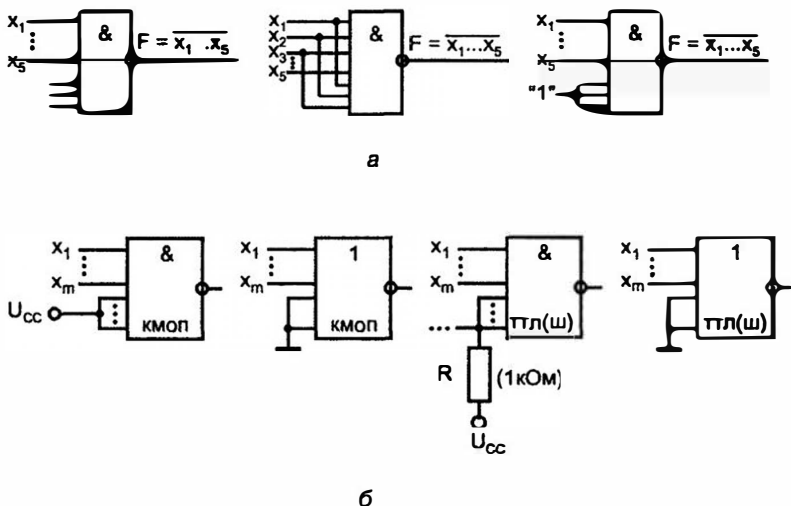


Рис. 1.19. Принципиально возможные (а) и рекомендуемые (б) режимы неиспользуемых входов логических элементов, схема формирования сигналов логической единицы (в)

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.

Таким образом, для КМОП и ТТЛ(Ш) 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 совпадают с уровнями U_{cc} и "земли", к которым и подключают неиспользуемые входы. У элементов ТТЛ(Ш) уровень U_1 на 1,5...2 В ниже уровня U_{cc} , поэтому для предотвращения пробоев неиспользуемые входы подключают к источнику питания U_{cc} через резисторы R , (обычная рекомендация: $R = 1$ кОм), причем к одному резистору разрешается подключать до 20 входов.

Примеры, иллюстрирующие перечисленные способы подключения неиспользуемых выводов ИС, показаны на рис. 1.19, б. Сигналы логической единицы можно получать от специального элемента (рис. 1.19, в), причем, если это мощный элемент, то он может иметь коэффициент разветвления до 30.

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 элементами 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. Нарращивание числа входов для операций И-НЕ, ИЛИ-НЕ, в сущности, производится аналогичным методом, но в схеме появляются дополнительные инверторы (рис. 1.20, а). На этом рисунке звездочка обозначает операцию Шеффера или Пирса.

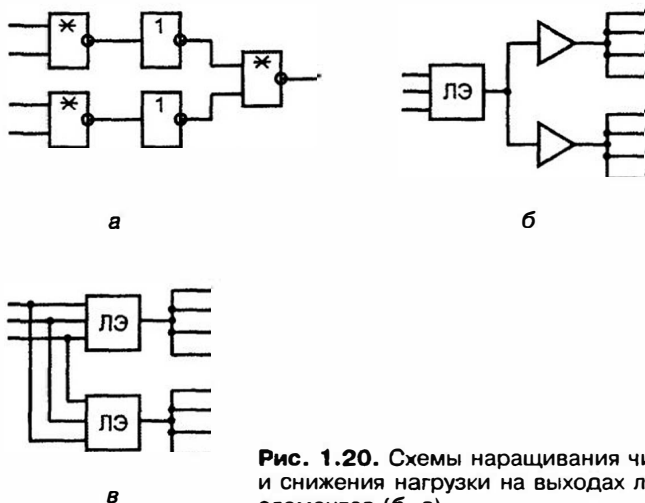


Рис. 1.20. Схемы наращивания числа входов (а) и снижения нагрузки на выходах логических элементов (б, в)

Снижение нагрузок на выходах логических элементов

Это может понадобиться, если нагрузки превышают допустимые значения, а также для повышения быстродействия схем, на которое нагрузки элементов

оказывают самое непосредственное влияние. Чем больше число нагрузок у элемента — источника сигнала (или нестандартная внешняя нагрузка), 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. 1.20, 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 приема должна оцениваться конкретным расчетом.

Литература к главе: [37], [2], [28], [36], [21], [46].

Глава 2

Функциональные узлы комбинационного типа

§ 2.1. Введение в проблематику проектирования ЦУ комбинационного типа

Функциональные узлы выполняют типовые для цифровых устройств микрооперации. Микрооперации соответствуют низшему иерархическому уровню внутреннего языка цифрового устройства, они обозначены в этом языке и не содержат других операций, обозначенных в нем.

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 чему 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 $F = x\bar{x} = 1$. В статике при любом значении x на одном из входов элемента И-НЕ имеется логический ноль, обеспечивающий единичное значение выхода. В переходных процессах возможен статический 1-риск.

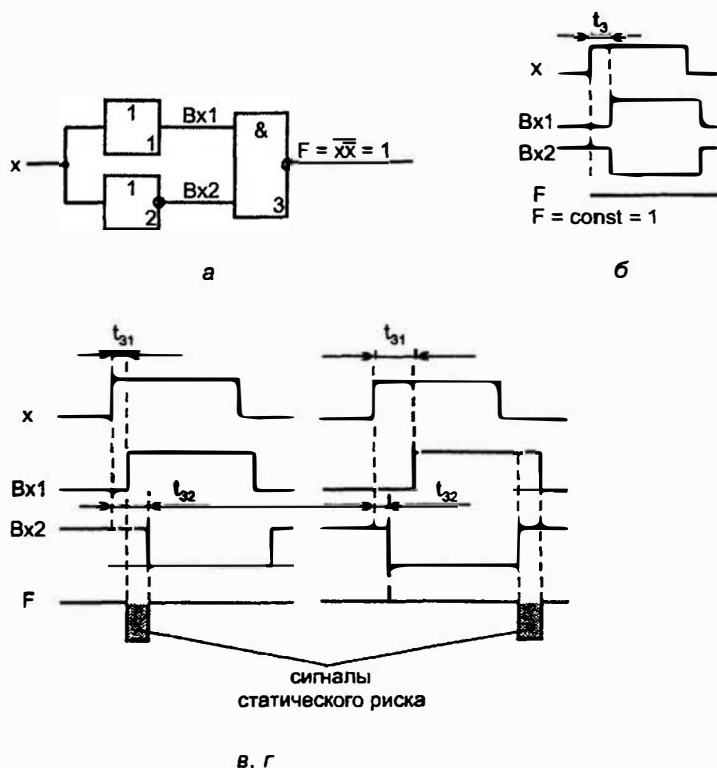


Рис. 2.1. Схема, иллюстрирующая механизм возникновения статического риска в комбинационной цепи (а), и временные диаграммы ее работы (б, в, г)

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 в 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 $t_{3.2\min.}$, которое и обозначит начало интервала переходных процессов. На самом длинном пути (до выхода F_1) сигнал задержится не более чем на время $t_3 = t_{3.1.\max} + t_{3.3.\max} + t_{3.4.\max}$, по истечении которого переходные процессы завершатся.

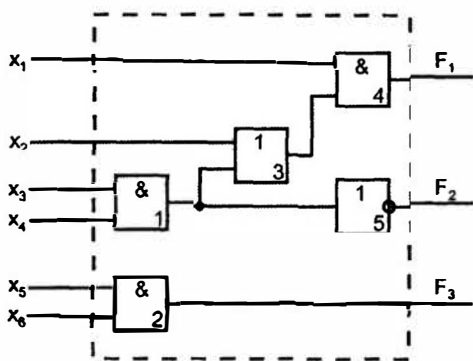


Рис. 2.2. Схема, иллюстрирующая расчет длительности переходного процесса в комбинационной цепи

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.

Из приведенного примера видно, что для расчета переходных процессов в ЦУ нужны сведения о минимальных и максимальных значениях задержек элементов. К сожалению, изготовитель часто указывает только максимальные значения задержек, нередко 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 $t_{3 \max}/t_{3 \min}$ равно обычно 2...3, для элементов одного кристалла между задержками элементов возникает сильная корреляция, и отношение, $t_{3 \max}/t_{3 \min}$ может существенно снижаться.

В состав ЦУ, как правило, входят типовые функциональные узлы и некоторое количество логических схем, специфичных для данного конкретного проекта (как иногда говорят — произвольной логики). *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

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

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

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

1. Логические блоки табличного типа (LUTs, Look-Up Tables).

2. Логические блоки в виде последовательности матриц элементов И и ИЛИ (PLA, Programmable Logic Array; PAL, Programmable Array Logic).
3. Универсальные логические блоки на основе мультиплексоров.
4. 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,

$$\begin{aligned} F(x_1 x_2 x_3) &= x_1 \vee \bar{x}_2 \bar{x}_3 = x_1 (x_2 \vee \bar{x}_2) (x_3 \vee \bar{x}_3) \vee x_2 \bar{x}_3 (x_1 \vee \bar{x}_1) = \\ &= x_1 x_2 x_3 \vee x_1 \bar{x}_2 \bar{x}_3 \vee x_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3. \end{aligned}$$

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.

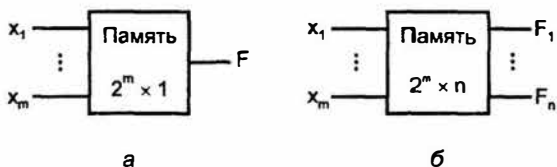


Рис. 2.3. Блоки памяти для воспроизведения одной (а) и нескольких (б) логических функций

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 данный 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 predetermined 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 "ventиль" 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

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

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

$$F = x_1 x_2 \vee x_3 x_4, \quad \bar{F} = \overline{x_1 x_2 \vee x_3 x_4} = \overline{\overline{x_1 x_2 \vee x_3 x_4}} = \overline{\overline{x_1} \vee \overline{x_2} \vee \overline{x_3} \vee \overline{x_4}}.$$

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 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 [28] сказано (с. 6): "Примером исчисления, которым широко пользуются в процессе синтеза логических схем, являются преобразования ал-

гебры логики. Набор правил говорит лишь о том, как можно преобразовать исходное булево 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 — нулевыми. Например, код "1 из N", содержащий 4 кодовых комбинации, будет представлен следующим образом:

1	0	0	0	Из сказанного видно, что двоичный дешифратор, имеющий n входов, должен иметь 2^n выходов, соответствующих числу разных комбинаций в n -разрядном двоичном коде.
0	1	0	0	
0	0	1	0	
0	0	0	1	

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 .

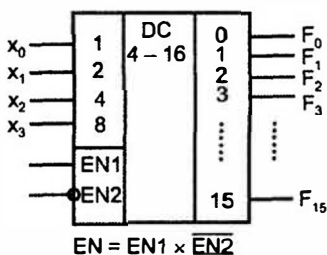
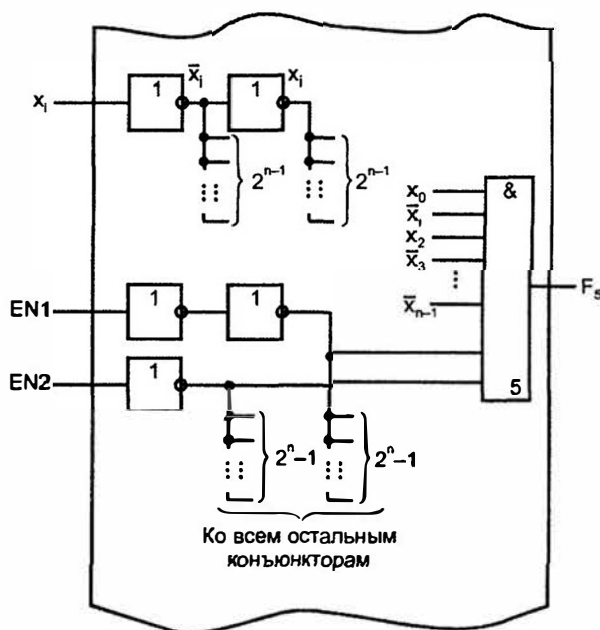


Рис. 2.4. Условное обозначение (а) и схемная реализация (б) двоичного дешифратора

а



б

В условном обозначении дешифраторов проставляются буквы DC (от английского Decoder). Входы дешифратора принято обозначать их двоичными весами. Кроме информационных входов дешифратор обычно имеет один или более входов разрешения работы обозначаемых как EN (Enable). При наличии разрешения по этому входу дешифратор работает описанным выше

удачным 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.

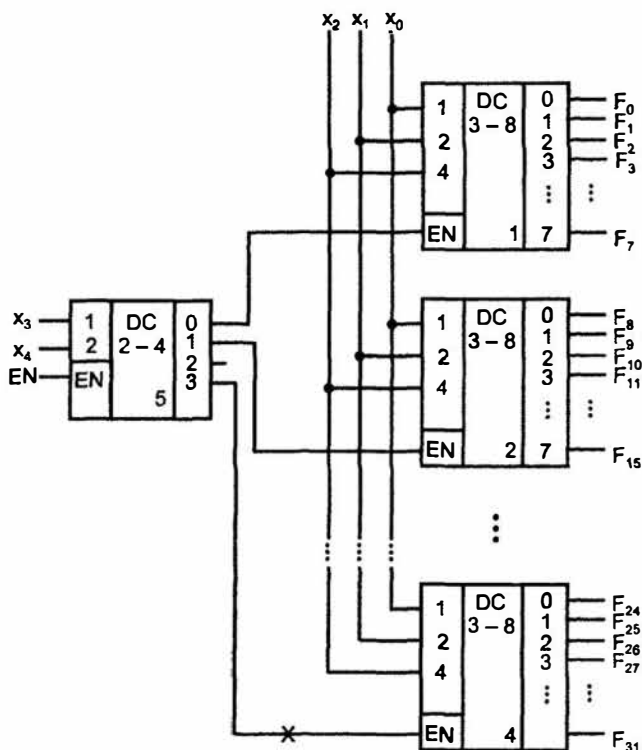


Рис. 2.5. Схема наращивания размерности двоичного дешифратора

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 получения 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 единственный 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.

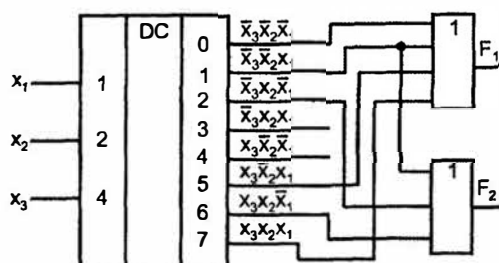


Рис. 2.6. Схема воспроизведения произвольных логических функций с помощью дешифратора и дизъюнкторов

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.

§ 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 указанной задачи — присвоение каждому источнику запросов фиксированного приоритета. Например, группа из восьми запросов $R_7 \dots R_0$ (R от английского 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 и т. д.

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

Таблица 2.2

Е1	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

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

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

$$\begin{aligned}a_2 &= R_7 \vee \bar{R}_6 \vee R_5 \vee \bar{R}_4; \\a_1 &= R_7 \vee \bar{R}_6 \vee \bar{R}_5 \bar{R}_4 R_2 \vee \bar{R}_5 \bar{R}_4 R_3; \\a_0 &= (R_7 \vee \bar{R}_6 R_5 \vee \bar{R}_6 \bar{R}_4 R_3 \vee \bar{R}_6 \bar{R}_4 \bar{R}_2 R_1,\end{aligned}$$

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 EI_2 . Если на входах $\bar{R}_8...R_{15}$ есть хотя бы один запрос, то разрешения на работу младшего шифратора 1 нет ($EO_2 = 1$). 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_i ($i = 0, 1, 2$). 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 EO_2 дает номера от 8 до 15.

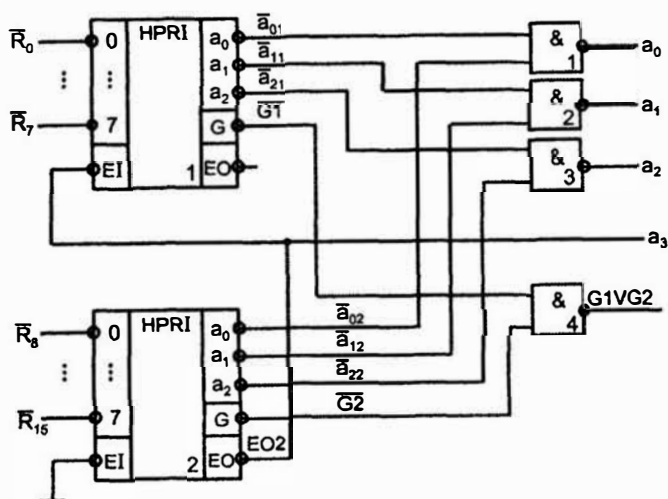


Рис. 2.7. Схема наращивания размерности приоритетного шифратора

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 \bar{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.

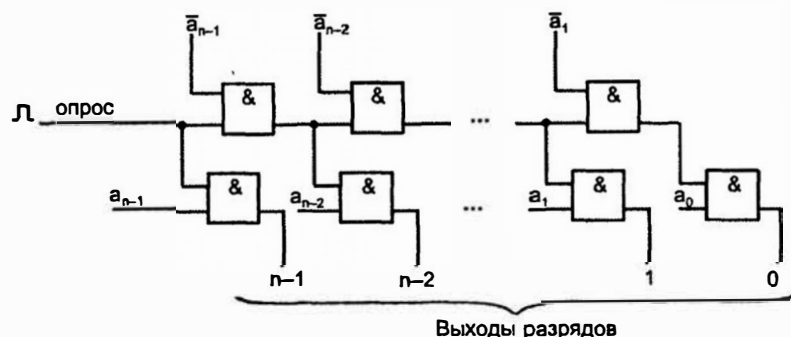


Рис. 2.8. Схема указания старшей единицы

Указатели старшей единицы могут быть реализованы подключением двоичного дешифратора к выходу шифратора приоритета, но эту же задачу можно

решить с помощью специальной цепочечной схемы (рис. 2.8) путем последовательного опроса разрядов, начиная со старшего, и прекращения дальнейшего опроса при выявлении первой же единицы.

В этой схеме единичный сигнал опроса, подаваемый со стороны старшего разряда a_{n-1} может распространяться вправо только до первого разряда, содержащего единицу. Разряд, содержащий ноль, пропускает сигнал опроса, на его выходе остается нулевой уровень. На выходе единичного разряда конъюнктор блокируется нулевым значением инвертированной переменной, и дальнейшее распространение переноса прекращается. Одновременно на выходе разряда возникает единичный сигнал.

§ 2.4. Мультиплексоры и демультиплексоры

Мультиплексоры осуществляют подключение одного из входных каналов к выходному под управлением управляющего (адресующего) слова. Разрядности каналов могут быть различными, мультиплексоры для коммутации многоразрядных слов составляются из одноразрядных.

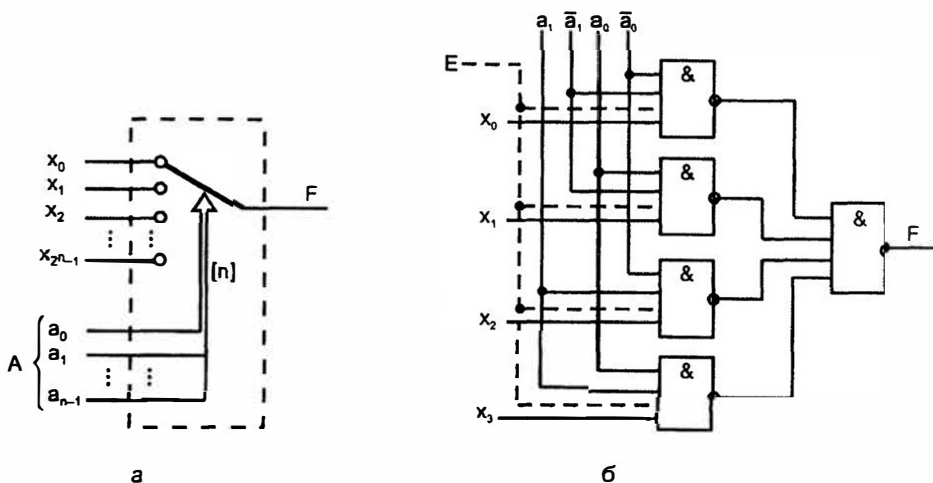


Рис. 2.9. Упрощенное представление мультиплексора многопозиционным ключом (а) и реализация мультиплексора на элементах И-НЕ (б)

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. 2.9, 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

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

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. 2.9, 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 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. 2.10 (сокращение MUX от английского MultipleXer).

Демультимплексоры выполняют операцию, обратную операции мультиплексоров — передают данные из одного входного канала в один из нескольких каналов-приемников. Многоадресные демультимплексоры состоят из нескольких одноадресных. Условное обозначение демультимплексоров на примере размерности "1–4" показано на рис. 2.11.

Нетрудно заметить, что дешифратор со входом разрешения работы будет работать в режиме демультимплексора, 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.

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. Приравнявая число воспроизводимых функций к числу комбинаций кода настройки, имеем для числа настроечных входов соотношение $m = 2^n$.

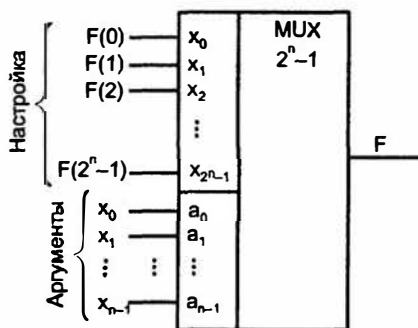
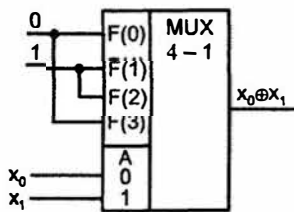
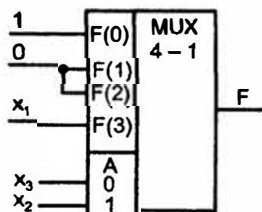


Рис. 2.12. Схема использования мультиплексора в качестве УЛМ (а), примеры воспроизведения функций при настройке константами (б) и при переносе одного аргумента в число сигналов настройки (в)



б



в

Полученному выражению отвечает соотношение между числом входов разного типа для мультиплексора. При этом на адресные входы следует пода-

вать аргументы функции, а на информационные входы — сигналы настройки (рис. 2.12, а). Таким образом, *для использования мультиплексора в качестве УЛМ следует изменить назначение его входов.*

Рис. 2.12, а — 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, б 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 настройки. 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.

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 $F = x_1 x_2 x_3 \vee \bar{x}_2 \bar{x}_3$. 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	$F_{\text{ост}}$
0	0	1
0	1	0
1	0	0
1	1	x_1

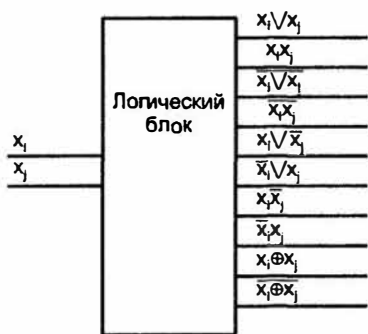
а

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

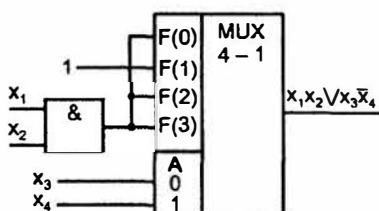
б

Схема УЛМ приведена на рис. 2.12, а.

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 2-х 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 $F = x_1x_2\sqrt{x_3\bar{x}_4}$ 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. 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.

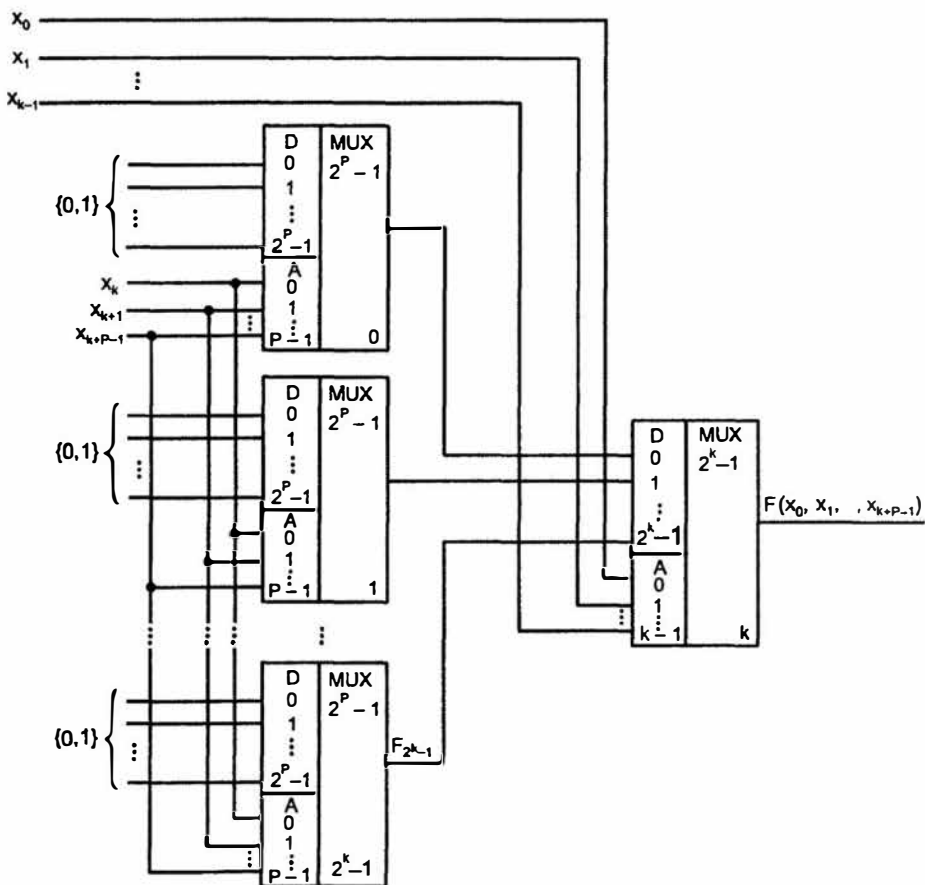


Рис. 2.14. Структура УЛМ, построенного на нескольких мультиплексорах

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 относятся 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 $0...2^k - 1$ 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

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

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

Разложение функции по двум аргументам

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

и, наконец, разложение по k аргументам

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

где

$$F_0 = F(0, 0, \dots, 0, x_k, \dots, x_{n-1}), \\ F_1 = F(0, 0, \dots, 0, 1, x_k, \dots, x_{n-1}),$$

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

Структура формул разложения полностью соответствует реализации двухъярусным УЛМ. В первом ярусе реализуются функции F_i , ($i = 0, \dots, 2^k - 1$), 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 4-х 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 $F = x_0x_1 \vee x_2x_3$. 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
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 \tilde{x}_0 в сигналы настройки (алфавит настройки $\{0, 1, \tilde{x}_0\}$) 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	$F_{\text{ост}}$
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.

Таблица 2.6

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

просмотреть четверки смежных строк таблицы с неизменными наборами $x_2 x_3$ — аргументами, подаваемыми на адресные входы УЛМ. Этот просмотр приводит к следующей таблице (табл. 2.6).

Из таблицы видно, что для воспроизведения функции достаточно использовать мультиплексор "4—1" с дополнительным конъюнктом для получения произведения $x_1 x_0$. 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

$$F = \overline{S_0} \overline{S_1} (\overline{S_A} A_0 \vee S_A A_1) \vee (S_0 \vee S_1) (\overline{S_B} B_0 \vee S_B B_1).$$

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.

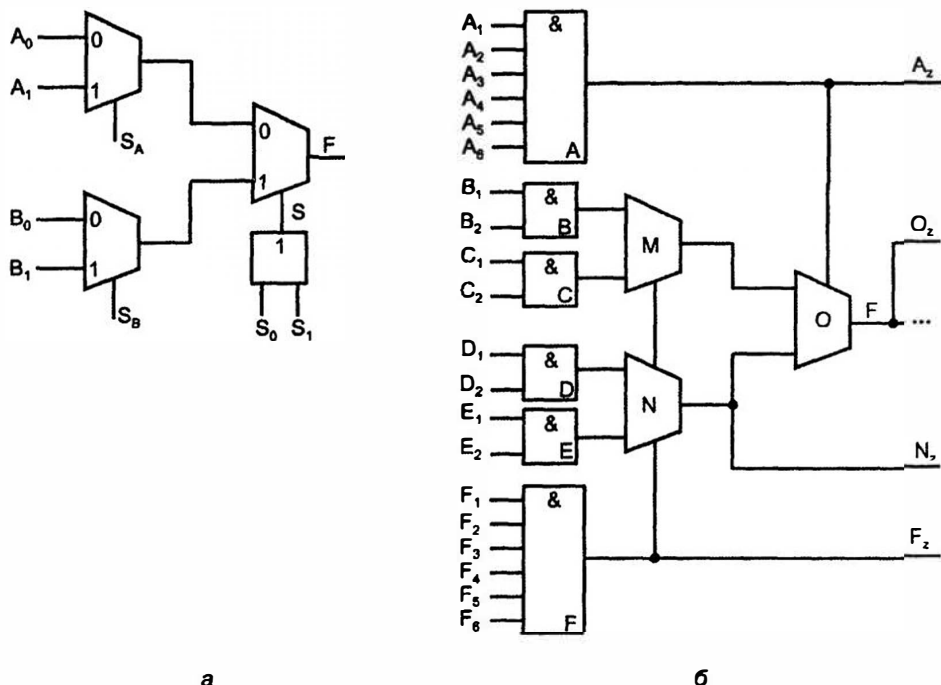


Рис. 2.15. Мультиплексорные логические блоки, используемые в микросхемах фирм Actel (а) и Quicklogic (б)

§ 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:

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

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.

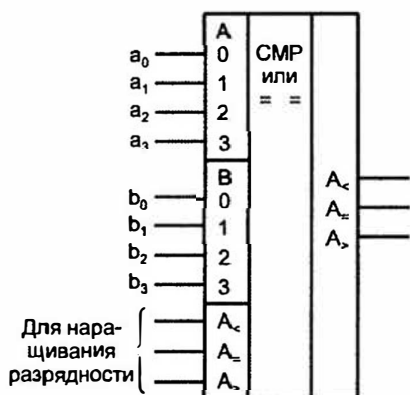


Рис. 2.16. Условное обозначение компаратора с тремя выходами

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

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

Признак неравенства разрядов

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

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

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

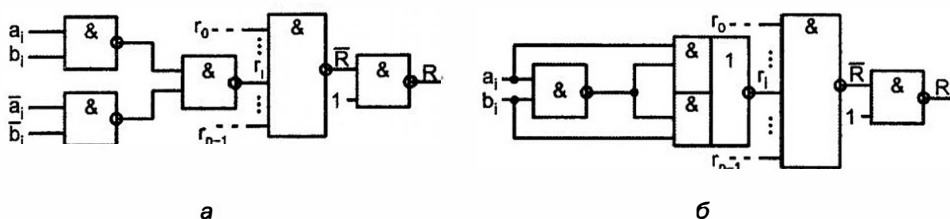


Рис. 2.17. Схемы компараторов на равенство (а, б)

Схема без парафазных входов (рис. 2.17, б) основана на выражениях для g_i , преобразованных следующим образом:

$$g_i = \overline{a_i b_i} \vee \overline{a_i} b_i = \overline{a_i (a_i \vee b_i)} \vee b_i (a_i \vee b_i) = \overline{a_i a_i} \vee \overline{b_i a_i} \vee b_i a_i \vee b_i b_i.$$

Таблица 2.7

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

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

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 $F_{A>B} = a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0$.

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

$$F_{A>B} = a_{n-1} \bar{b}_{n-1} \vee g_{n-1} a_{n-2} \bar{b}_{n-2} \vee \dots \vee g_{n-1} g_{n-2} \dots g_1 a_0 \bar{b}_0.$$

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 g_i можно заменить более простыми функциями $d_j = a_j \bar{b}_j$. Однако для построения компаратора с тремя выходами ("равно", "больше" и "меньше") полученный нами вариант остается предпочтительным, поскольку функции g_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. Выработка признака $A > B$ в этой схеме производится по соотношению (штрихом отмечены функции с выходов младшей группы)

$$F_{A>B} = a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0 = \overline{a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0} \vee (a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0) = \overline{a_1 \bar{b}_1} \cdot \overline{g_1 a_0 \bar{b}_0} \vee (a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0) = \overline{a_1 \bar{b}_1} \cdot \overline{g_1 a_0 \bar{b}_0} \vee (a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0) = \overline{a_1 \bar{b}_1} \cdot \overline{g_1 a_0 \bar{b}_0} \vee (a_1 \bar{b}_1 \vee g_1 a_0 \bar{b}_0).$$

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

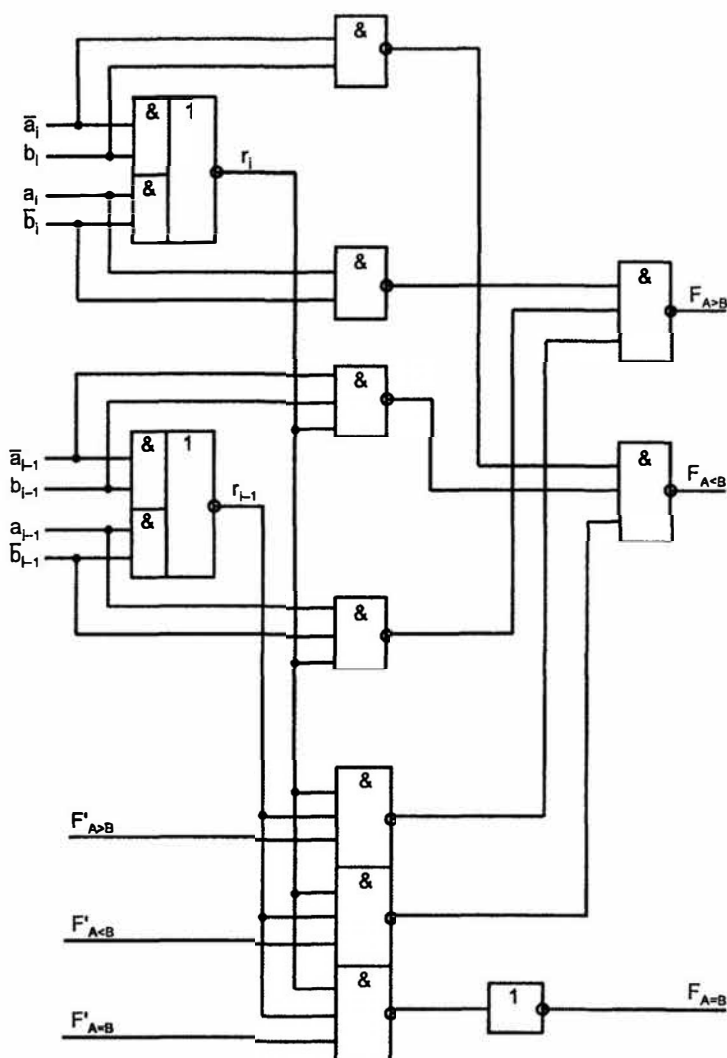


Рис. 2.18. Пример построения компаратора

§ 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, трехкратное 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

Таблица 2.8

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.

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. 2.19).

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

$$F = F_1 F_2 \vee F_1 F_3 \vee F_2 F_3, \quad a_1 = F_2 \oplus F_3, \quad a_0 = F_1 \oplus F_3.$$

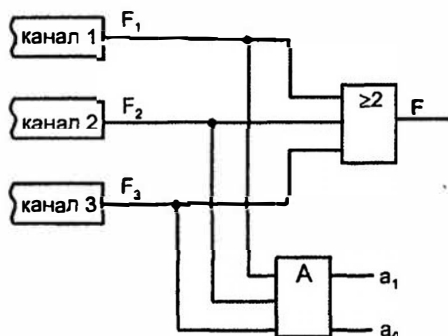


Рис. 2.19. Схема голосования с мажоритарным элементом

В схемах типа рис. 2.19 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 применяе-

мым 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:

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

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

Существует также понятие *веса комбинации*, под которым понимается 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 r при контроле по четности ($r_{\text{ч}}$) и нечетности ($r_{\text{н}}$) приведены для четырехразрядного информационного слова в табл. 2.9.

Таблица 2.9

Как видно из таблицы, $\rho_4 = a_3 \oplus a_2 \oplus a_1 \oplus a_0$;
 $\rho_n = a_3 \oplus a_2 \oplus a_1 \oplus a_0$.

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.

a_3	a_2	a_1	a_0	ρ_4	ρ_n
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

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. Для практики типична многоярусная схема свертки пирамидального типа (рис. 2.20, а).

На рис. 2.20, 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:

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

где $N_{лз}$ — 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. 2.20, б), которая выдает результат всего через одну задержку после поступления последнего разряда a_7 .

Примером ИС свертки по модулю 2 может служить микросхема ИП5 серии КР1533 (рис. 2.21, а). Схема имеет 9 входов, что допускает свертку байта с девятым контрольным разрядом. Двумя выходами схемы являются Е (Even) и О (Odd). Если вес входной комбинации четный, то $E = 1$ и $O = 0$, и наоборот, если вес нечетный.

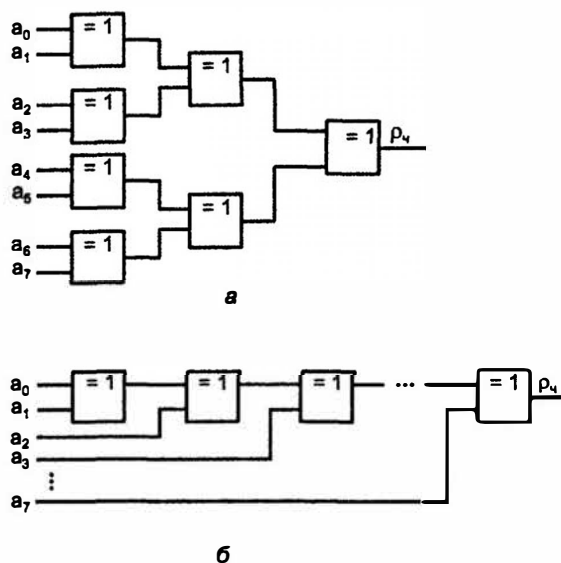
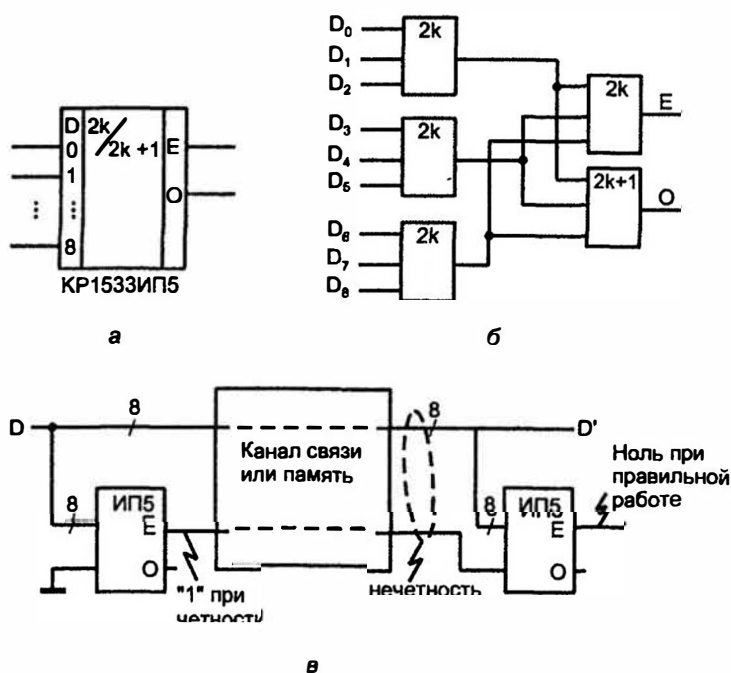


Рис. 2.20. Схемы свертки пирамидального (а) и последовательного (б) типов



Схемотехнически ИС КР1533ИП5 представляет собою пирамидальную структуру из трехходовых элементов типа четность/нечетность (рис. 2.21, б).

Передача данных с контролем по модулю 2

Передача данных или их запись/считывание (если речь идет о памяти) с контролем показаны на рис. 2.21, 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. 2.22 показан пример контроля логического преобразователя ЛП, воспроизводящего систему переключательных функций от m переменных. Для осуществления контроля к системе добавляется еще одна функция $F_{\text{доп}} = F_1 \oplus F_2 \oplus \dots \oplus F_n$, которая воспроизводится на индивидуальных элементах (во избежание маскирования контролируемых ошибок). Затем выработанные функции $F_1 \dots F_n$ свертываются по модулю 2, и результат сравнивается с дополнительной функцией $F_{\text{доп}}$. 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.

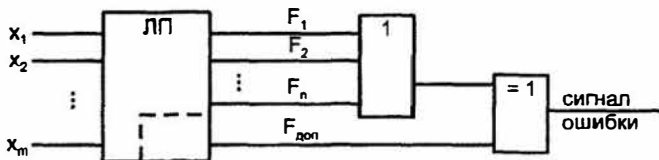


Рис. 2.22. Схема контроля логического преобразователя по модулю 2

Контроль с использованием кодов Хемминга

Применение кодов Хемминга позволяет *исправлять единичные ошибки*. Добавление к коду Хемминга контрольного разряда, обеспечивающего чет-

ность/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 ($i = 0, 1, 2, \dots$).

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, ...

Таблица 2.10

8	7	6	5	4	3	2	1	
p_3	a_3	a_2	a_1	p_2	a_0	p_1	p_0	
0	0	0	0	0	0	0	0	
1	0	0	0	0	1	1	1	
1	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	
0	0	1	1	0	0	1	1	
1	0	1	1	0	1	0	0	
0	1	0	0	1	0	1	0	
1	1	1	1	1	1	1	1	

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 p в таблице обозначен общий контрольный разряд для всей кодовой комбинации, через p_1, p_2, p_3 — первый, второй и третий групповые контрольные разряды.

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 p , 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 резуль-

таты 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. 2.23 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 собою декодирующее устройство для кода Хемминга.

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

Кодирование-декодирование для 16-разрядных слов с формированием 6 контрольных разрядов модифицированного кода Хемминга реализуется микросхемой ВЖ1 серий K555, 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 Рид-Соломона и др.).

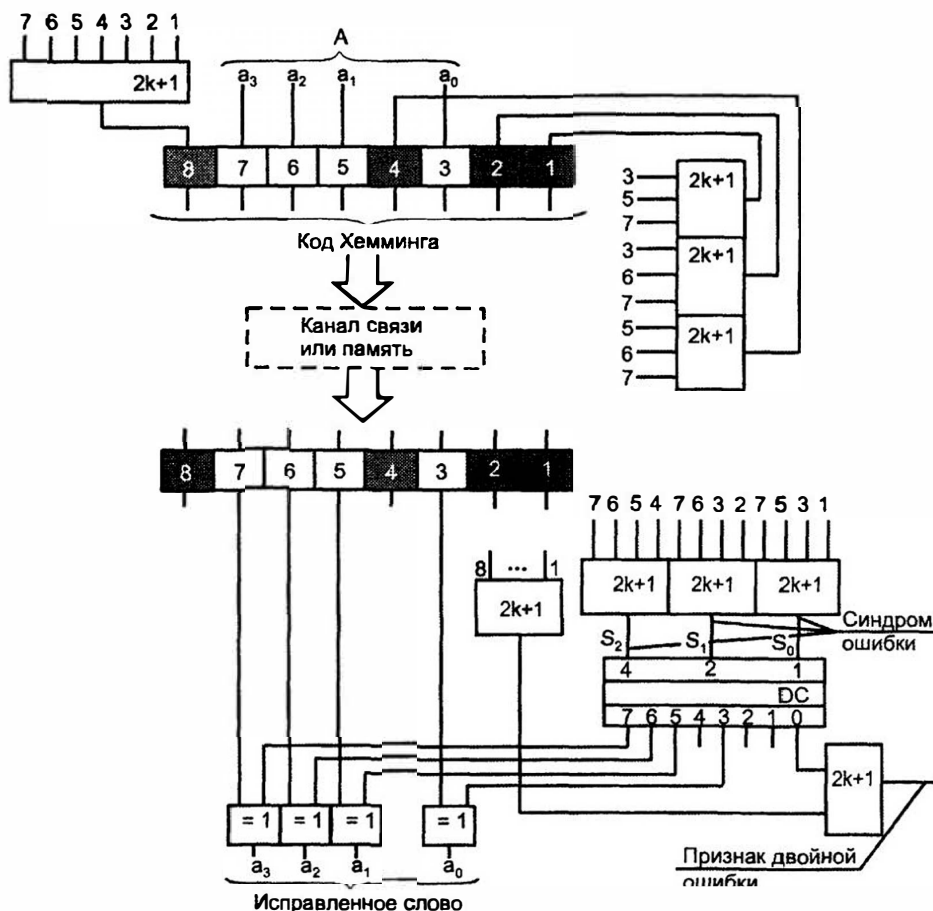


Рис. 2.23. Схема кодирования и декодирования для кодов Хемминга

§ 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 неперменной 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 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 через сложение с применением дополнительных либо обратных кодов.

Одноразрядный сумматор

Таблица 2.12

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 следующий вид (табл. 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 C от английского carry) имеют вид

$$S_i = \bar{a}_i \bar{b}_i c_{i-1} \vee \bar{a}_i b_i \bar{c}_{i-1} \vee a_i \bar{b}_i \bar{c}_{i-1} \vee a_i b_i c_{i-1}, \quad C_i = a_i b_i \vee a_i c_{i-1} \vee b_i c_{i-1}.$$

В базисе Шеффера функции S_i и C_i выражаются следующим образом:

$$S_i = \overline{a_i \bar{b}_i c_{i-1} \cdot \bar{a}_i b_i \bar{c}_{i-1} \cdot a_i \bar{b}_i \bar{c}_{i-1} \cdot a_i b_i c_{i-1}},$$

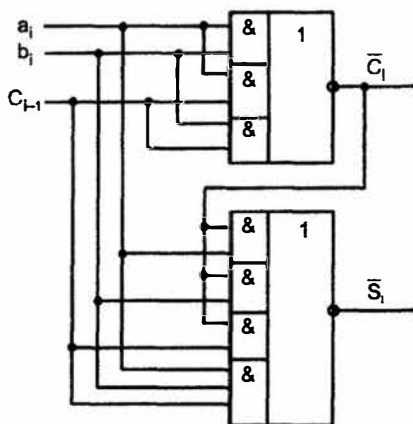
$$C_i = \overline{a_i \bar{b}_i \cdot a_i c_{i-1} \cdot b_i c_{i-1}}.$$

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 3-3-3И-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 .

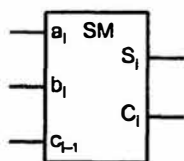
Из табл. 2.12 видно, что во всех строчках, кроме первой и последней, $S_i = \bar{C}_i$. Чтобы сделать формулу справедливой также в первой и последней строчках, нужно убрать единицу в строчке нулевых входных величин и добавить единицу в строчку единичных входных величин, что приводит к соотношению

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

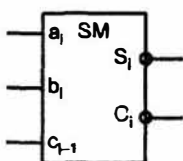
Схема сумматора, построенного по этому соотношению, показана на рис. 2.24, а.



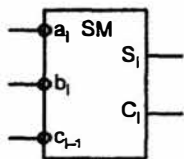
а



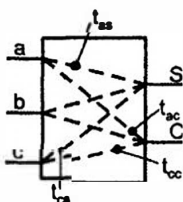
б



в



г



д

Рис. 2.24. Схема (а), условные обозначения (б, в, г) и пути распространения сигналов одноразрядного сумматора (д)

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.

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

Условное обозначение одноразрядного сумматора показано на рис. 2.24, 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. 2.24, 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. 2.24, д). Так как тракты от обоих слагаемых обычно одинаковы, остаются четыре задержки, отмеченные надписями t_{as} , t_{ac} , t_{cs} и t_{cs} на рис. 2.24, д.

На рис. 2.25 показана схема сумматора, входящая в библиотеку схемных решений семейства СБИС FLEX8000 фирмы Altera.

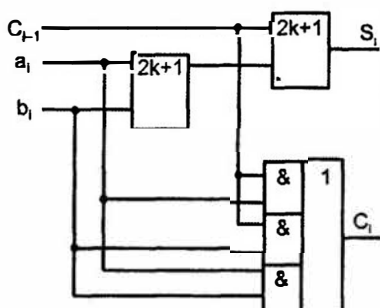


Рис. 2.25. Схема одноразрядного сумматора из библиотеки схемных решений для СБИС FLEX 8000

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. 2.26, а), помимо сумматора, содержит сдвигающие регистры слагаемых и суммы, а также триггер запоминания переноса. Регистры и триггер тактируются синхронными импульсами СИ.

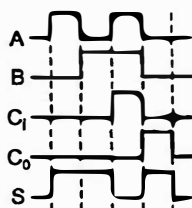
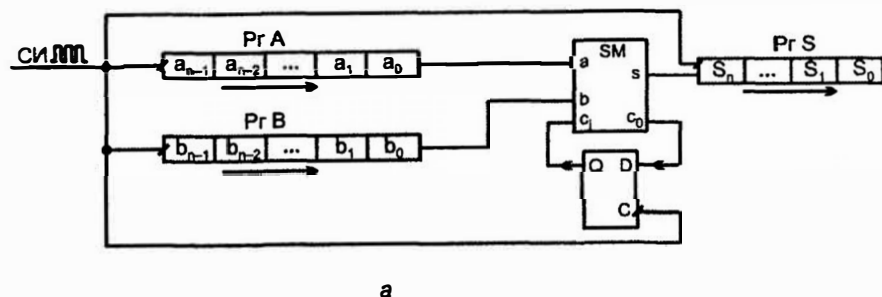


Рис. 2.26. Схема сумматора для последовательных операндов (а) и ее временная диаграмма (б)

На рис. 2.26, б показана временная диаграмма, соответствующая операции сложения двух операндов $101 + 110 = 1011$ или в десятичном выражении $5 + 6 = 11$.

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. 2.27, 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.

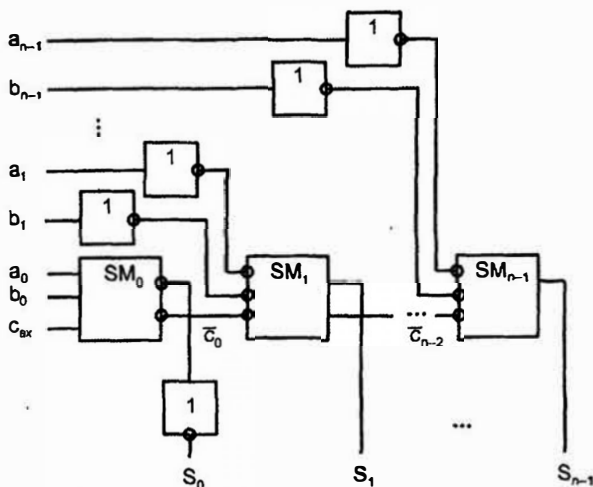


Рис. 2.27. Схема сумматора с последовательным переносом

Если одноразрядные сумматоры выполнены по схеме (см. рис. 2.24, 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. 2.25), 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. 2.28). Обозначение CR от слова carry (перенос).

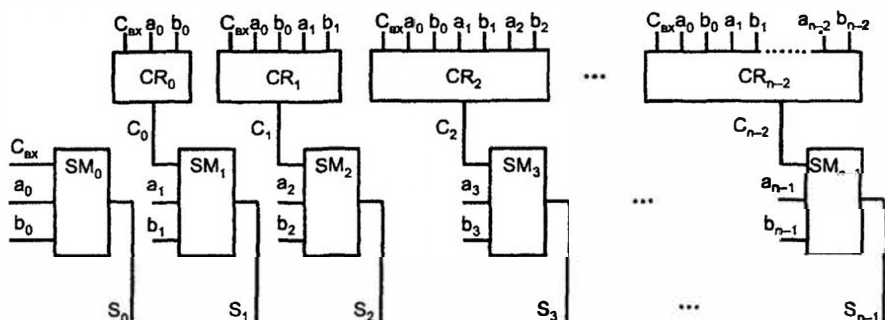


Рис. 2.28. Структура сумматора с параллельным переносом

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, $h_i = a_i b_i \vee \bar{a}_i \bar{b}_i$, 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 выведем функции переноса C 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

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

Для следующего разряда произведем те же действия

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

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

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

Функции переноса имеют нормальную дизъюнктивную форму и могут быть реализованы элементами И-ИЛИ (либо И-ИЛИ-НЕ, для \bar{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

$$\begin{aligned} C_0 &= \overline{g_0 \cdot C_{\text{вх}} h_0} = \overline{a_0 b_0 \cdot C_{\text{вх}} h_0}, \\ C_1 &= \overline{a_1 b_1 \cdot a_0 b_0 h_1 \cdot C_{\text{вх}} h_1 h_0}, \\ C_2 &= \overline{a_2 b_2 \cdot a_1 b_1 h_2 \cdot a_0 b_0 h_2 h_1 \cdot C_{\text{вх}} h_2 h_1 h_0}. \end{aligned}$$

Схема сумматора (рис. 2.29) соответствует полученным выражениям.

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

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.

Диапазон разрядностей, в которых проявляются достоинства сумматоров с параллельным переносом, невелик. До $n = 3 \dots 4$ преимущества имеют более простые схемы сумматоров с последовательным переносом, после $n = 8$ по-

являются перегруженные элементы и элементы с большим числом входов, что замедляет работу сумматора, требует введения развязывающих элементов с их задержками и т. п.

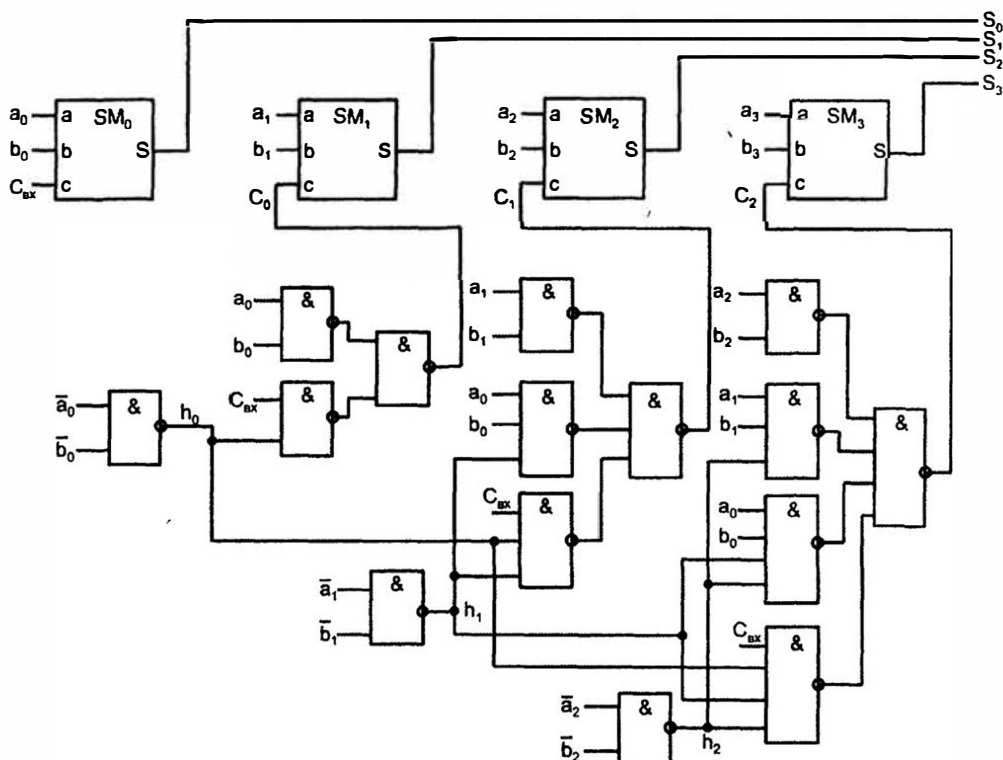


Рис. 2.29. Вариант схемы сумматора с параллельным переносом

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. 2.30). Слагаемые разбиты на m -разрядные поля, суммируемые в группах. Результат также составляется из m -разрядных полей.

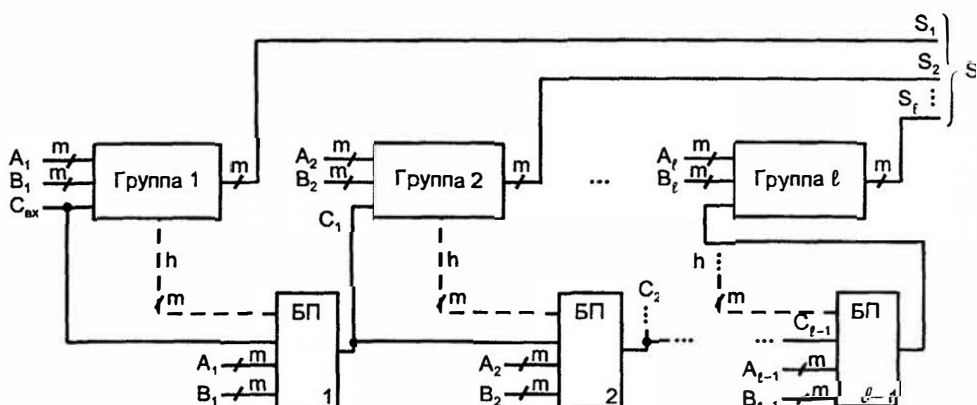


Рис. 2.30. Схема группового сумматора с цепным переносом

Блоки переноса БП_i ($i = 1 \dots$) 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 параллельными переносами сложность схем переноса непрерывно возрастает при переходе от предыдущего разряда к последующему (последняя схема переноса требует анализа операндов с разрядностью $n-1$).

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

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. 2.30). Имея в виду реализацию блоков переноса и групп, показанную выше для базиса И-НЕ, формулу для времени суммирования можно представить в виде:

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

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

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

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

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 соотношения:

$$H = h_{m-1} h_{m-2} \dots h_0,$$

согласно которому группа прозрачна при прозрачности всех ее разрядов, и

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

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 :

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

Схемы выработки переноса усложняются с ростом i . Структура группового сумматора с параллельными межгрупповыми переносами показана на рис. 2.31, 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).

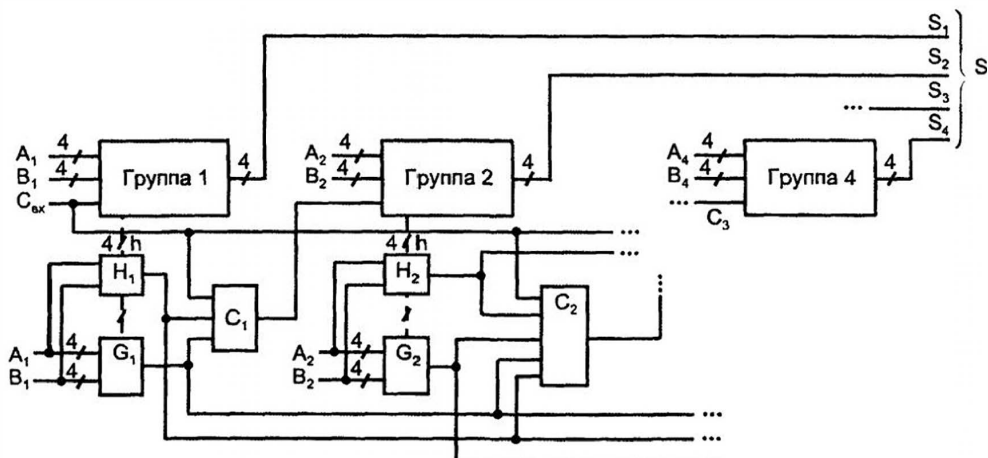


Рис. 2.31. Схема группового сумматора с параллельным переносом

Время суммирования для схемы (рис. 2.31)

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

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

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 организовав параллельный перенос между новыми группами. Сумматоры разных типов подробно описаны в работе [28].

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 FLEX 8000 была реализована цепь последовательных переносов с малыми задержками (1 нс на разряд). Это возродило интерес к структурам с последовательным переносом и, соответственно, к методам улучшения их быстродействия.

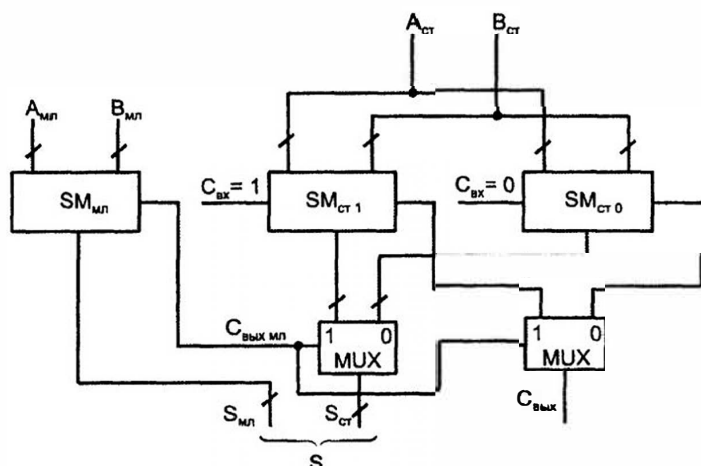


Рис. 2.32. Схема сумматора с условным переносом

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 $A_{\text{мл}}$ и $B_{\text{мл}}$. На втором — старшие поля операндов при условии $C_{\text{вх}} = 1$, в третьем — старшие поля операндов при условии $C_{\text{вх}} = 0$. 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. 2.32).

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. 2.33. Очередное прибавление слагаемого тактируется синхросигналами СИ. Учитывая особенности функционирования, накапливающие сумматоры называют иногда аккумуляторами.

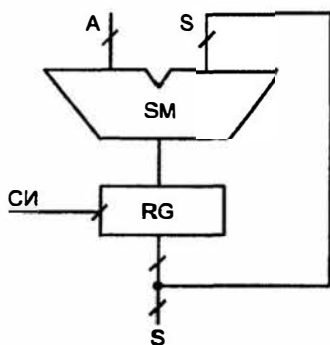


Рис. 2.33. Структура накапливающего сумматора

В сериях элементов имеются одноразрядные сумматоры, в том числе с дополнительной входной логикой, двухразрядные и четырехразрядные. Примером стандартных ИС сумматоров могут служить микросхемы ИМЗ серии К555, содержащие четырехразрядный сумматор с последовательным переносом и блок переноса (рис. 2.34), которые непосредственно пригодны для составления из них группового сумматора с цепным переносом.

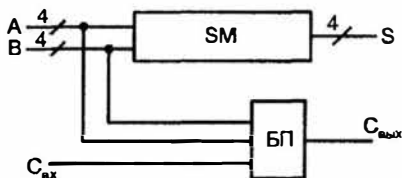


Рис. 2.34. Структура микросхемы К555ИМЗ

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

В некоторых сериях элементов сумматоры отсутствуют. Причиной этого обычно является наличие арифметико-логического устройства, 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. 2.35) имеет входы операндов А и В, входы выбора операций S, вход переноса \bar{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 \bar{C}_0 — выходной перенос, а выход $A = B$ есть выход сравнения на равенство с открытым коллектором.

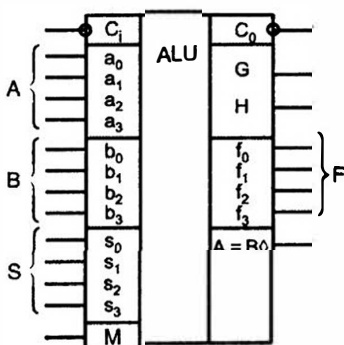


Рис. 2.35. Условное обозначение АЛУ

Перечень выполняемых АЛУ операций дан в табл. 2.13. Для краткости двоичные числа $s_3s_2s_1s_0$ представлены их десятичными эквивалентами. Под утолщенными обозначениями 1 и 0 следует понимать наборы 1111 и 0000, входной перенос поступает в младший разряд слова, т. е. равен $000C_1$. 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.

Таблица 2.13

S	Логические функции (M = 1)	Арифметико-логические функции (M = 0)
0	\bar{A}	$A + C_1$
1	$A \vee \bar{B}$	$A \vee B + C_1$
2	$\bar{A}B$	$A \vee \bar{B} + C_1$
3	0	$1 + C_1$
4	$\bar{A}\bar{B}$	$A + \bar{A}B + C_1$
5	\bar{B}	$A \vee B + \bar{A}\bar{B} + C_1$
6	$A \oplus B$	$A + \bar{B} + C_1$
7	$A\bar{B}$	$\bar{A}\bar{B} + 1 + C_1$
8	$\bar{A} \vee B$	$A + AB + C_1$
9	$\bar{A} \oplus \bar{B}$	$A + B + C_1$
10	B	$A \vee \bar{B} + AB + C_1$
11	AB	$AB + 1 + C_1$
12	1	$A + A + C_1$
13	$A \vee \bar{B}$	$A \vee B + A + C_1$
14	$A \vee B$	$A \vee \bar{B} + A + C_1$
15	A	$A + 1 + C_1$

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

Запись типа $A \vee \bar{B} + AB$ следует понимать так: вначале поразрядно выполняются операции инвертирования (\bar{B}), логического сложения ($A \vee \bar{B}$) и умножения (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. 2.36, а) или параллельных (рис. 2.36, 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 пе-

ренос и вырабатывающие сигналы переноса по формулам, приведенным в предыдущем параграфе.

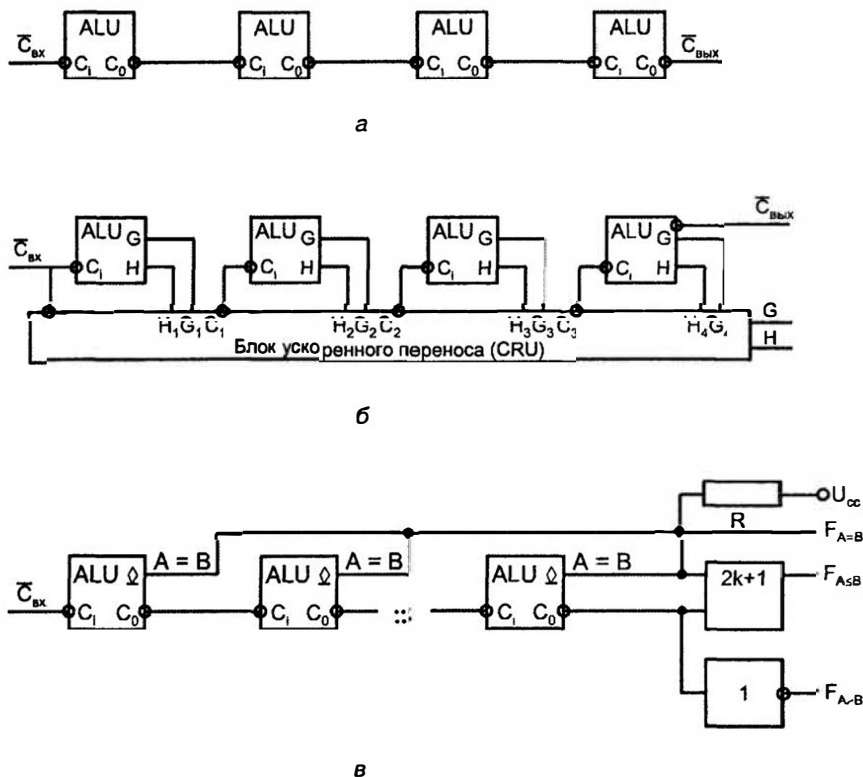


Рис. 2.36. Схемы наращивания АЛУ при последовательном (а) и параллельном (б) переносах и реализация функций компаратора для группы АЛУ (в)

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. 2.36, 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 годах, 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 $A_m = a_{m-1} \dots a_0$ и $B_n = b_{n-1} \dots b_0$. 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 $P_{m+n} = p_{m+n-1} p_{m+n-2} \dots p_0$.

Члены вида $a_i b_j$, где $i = 0 \dots (m-1)$ и $j = 0 \dots (n-1)$ 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.

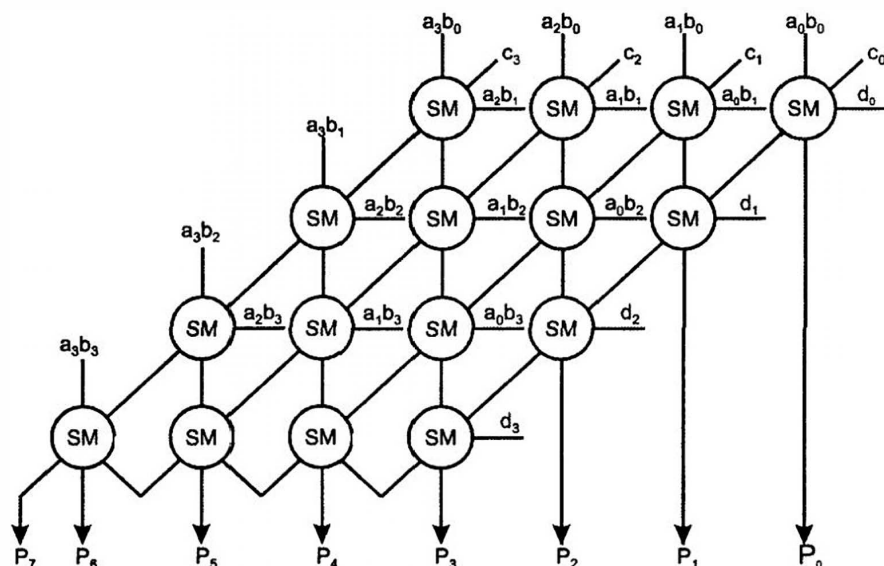
Матричные перемножители могут быть просто *множительными* блоками (МБ) или *множительно-суммирующими* (МСБ), последние обеспечивают удобство наращивания размерности умножителя.

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.

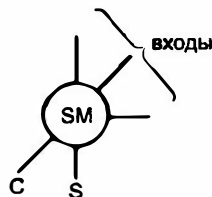
Множительно-суммирующие блоки

Множительно-суммирующий блок для четырехразрядных операндов без набора конъюнкторов, вырабатывающих члены вида $a_i b_j$, показан на рис. 2.37, а, где для одноразрядного сумматора принято обозначение (рис. 2.37, б).

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



а



б

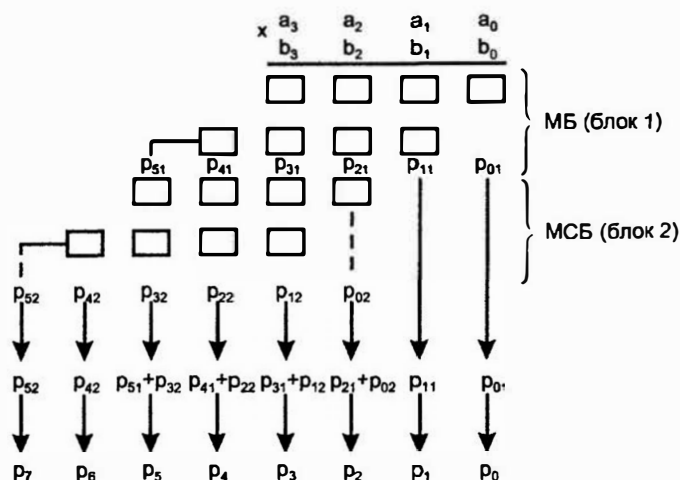
Рис. 2.37. Схема множително-суммирующего блока для четырехразрядных сомножителей (а), обозначение одноразрядного сумматора для данной схемы (б)

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, $t_{MPL} = t_K + (2n-1)t_{SM}$.

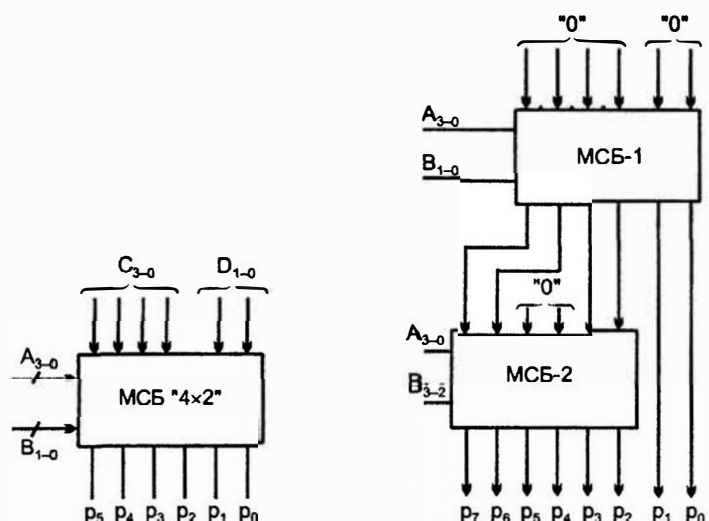
Схема множительного блока отличается от схемы МСБ тем, что в ней отсутствуют сумматоры правой диагонали, т. к. при $C_m = 0$ и $D_n = 0$ они не требуются.

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. 2.38, а на при-

мере 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).



а



б

в

Рис. 2.38. 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)

Перемножение в пределах зон дает частичные произведения $p_1 = p_{51}p_{41}p_{31}p_{21}p_{11}p_{01}$ и $p_2 = p_{52}p_{42}p_{32}p_{22}p_{12}p_{02}$. 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. 2.38, *в*. В ней использовано условное обозначение МСБ (рис. 2.38, *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 выше 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

$$P = A \times B = A \times (b_{n-1}2^{n-1} + b_{n-2}2^{n-2} + \dots + b_02^0). \quad (a)$$

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

Выражение (a) можно видоизменить с помощью соотношения

$$b_i2^i = b_i2^{i+1} - 2b_i2^{i-1}, \quad (б)$$

справедливость которого очевидна.

Это соотношение позволяет разреживать последовательность (спектр) степеней в сумме частичных произведений. Можно, например, исключить четные степени, как показано на рис. 2.39, *а*. 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.

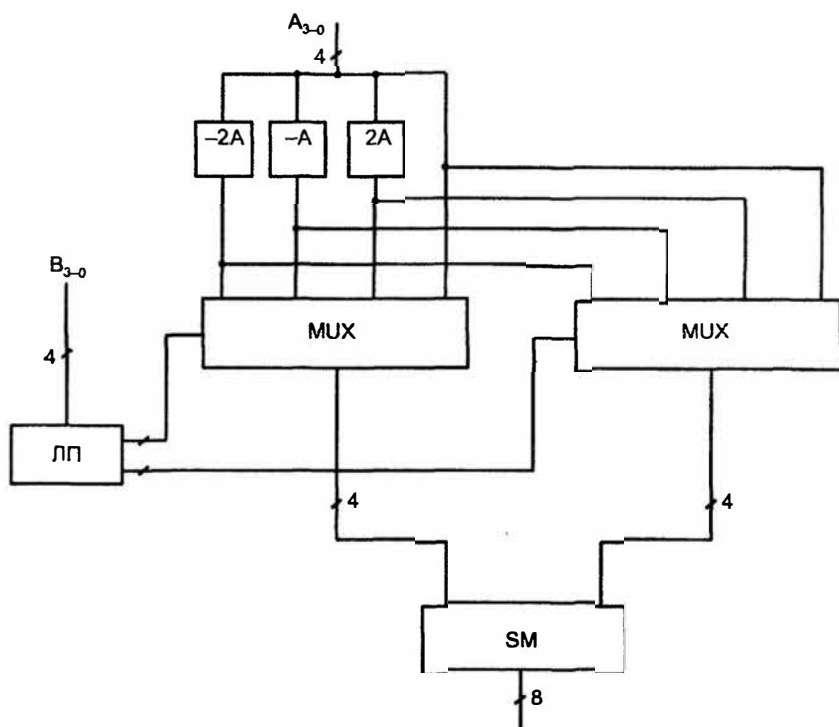
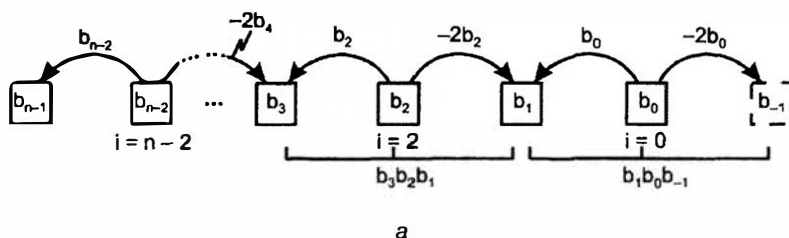


Рис. 2.39. К пояснению принципа быстрого умножения "сразу на два разряда" (а) и схема быстрого умножения (б)

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. 2.39, 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 — $-2A \cdot 2^{-1} = -A$, 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 $A \cdot 2^3 = 01010000$.

Для получения результата заданного умножения требуется сложить частичные произведения:

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

Схема, реализующая алгоритм быстрого умножения сразу на два разряда, показана на рис. 2.39, 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 сигналы выбора того или иного варианта частичных произведений. Окончательный результат получается суммированием частичных произведений с учетом их взаимного сдвига в разрядной сетке. Размерность множителя 4×4 .

Приведенные выше 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 с знаковыми разрядами, а, например, дополнительными кодами, то, имея рассмотренные выше 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 (ВРЗ, ВР4 и ВР5 соответственно). В схемотехнике ЭСЛ выполнен умножитель 1800ВР1 (8×8 за 17 нс). Зарубежные фирмы разработали умножители (фирмы ВІТ, 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.

Литература к главе: [37], [2], [28], [22], [29], [32], [36], [42], [43], [8], [33], [20], [35], [18], [44].

Глава 3

Функциональные узлы последовательного 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 ноль, 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.

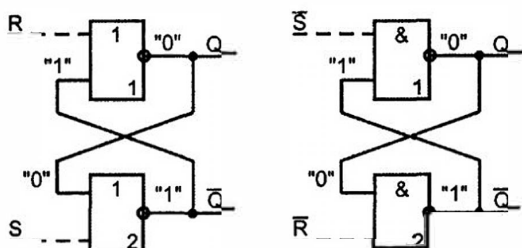


Рис. 3.1. Схемы фиксаторов с входами управления на элементах ИЛИ-НЕ и И-НЕ

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 ноль (сброса), а буквой S (от Set) — сигнал установки в состояние логической единицы (установки). Состояние триггера считывается по значению прямого выхода, обозначаемого как Q. Чаше всего триггер имеет и второй выход с инверсным сигналом \bar{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 важное значение приобретают классификация триггеров, изучение их параметров и особенностей функционирования.

Классификация триггеров

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

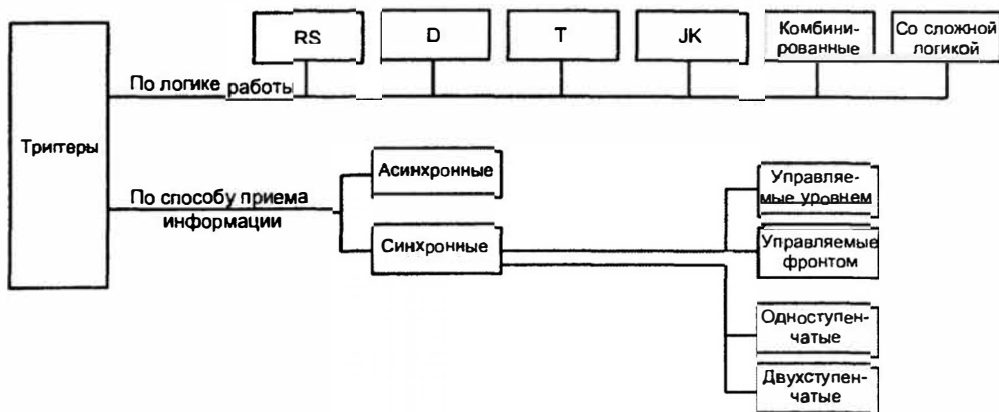


Рис. 3.2. Классификация триггеров, используемых в практической схемотехнике

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 ноль (R).

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

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. Имеет один вход, 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 $J_1 J_2 J_3$ и $K_1 K_2 K_3$, соединенными операцией конъюнкции:

$$J = J_1 J_2 J_3, K = K_1 K_2 K_3.$$

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, 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 С (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-зашелка" (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 английского 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 вы-

держки 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.

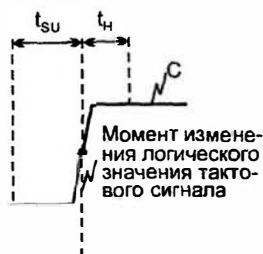


Рис. 3.4. К пояснению параметров предустановки и выдержки для синхронных триггеров

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 ниже.

Таблица 3.1

J	K	Q	Q _н
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

Таблицу истинности триггера JK можно записать в полном (табл. 3.1) или сокращенном виде (табл. 3.2). Через Q_н обозначено новое состояние триггера (после переключения).

J	K	Q _н
0	0	Q1
0	1	0
1	0	1
1	1	\bar{Q}

Карта Карно для JK-триггера показана на рис. 3.5. Из нее можно получить *характеристическое уравнение* триггера $Q_H = \bar{J}Q \vee \bar{Q}K$.

Переведя уравнение в логический базис элементов, на которых строится триггер, получим *структурное уравнение* триггера, определяющее конфигурацию его схемы.

JK	00	01	11	10
Q				
0	0	0	1	1
1	1	0	0	1

Рис. 3.5. Карта Карно для JK-триггера

Таблица 3.3

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.

Переход	J	K
0→0	0	X
0→1	1	X
1→0	X	1
1→1	X	0

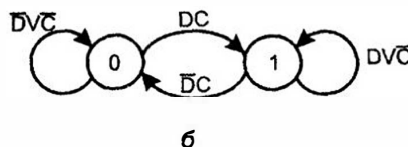
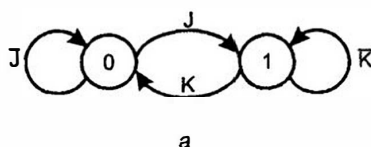


Рис. 3.6. Диаграммы состояний (графы переходов) для триггеров JK (а) и D (б)

Для D-триггера сокращенная таблица истинности дана в табл. 3.4, а словарь в табл. 3.5. Характеристическое уравнение триггера $Q_H = D$.

Таблица 3.4

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 С представлена на рис. 3.6, б. Синхросигнал С показан, т. к. триггеры типа D всегда тактируются.

Словари триггеров RS и Т имеют вид (табл. 3.6).

Таблица 3.6

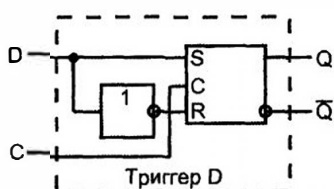
Переход	R	S	T
0→0	X	0	0
0→1	0	1	1
1→0	1	0	1
1→1	0	X	0

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.

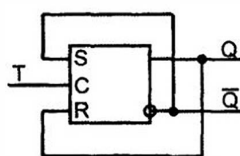
§ 3.2. Схемотехника триггерных устройств

Знание основ схемотехники триггерных устройств облегчает правильное их применение в различных условиях, в том числе и не всегда оговоренных в справочной литературе.

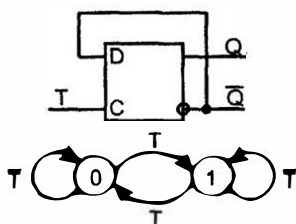
Прежде всего, отметим, что между триггерами RS и D, с одной стороны, и Т и JK с другой, имеется существенная разница. Первые имеют разомкнутую структуру (о внутренних обратных связях в схеме фиксатора сейчас речь не идет), а вторые используют выходные сигналы для воздействия на свои входы.



а



б



в

Рис. 3.7. Схемы информационных связей, образующих триггеры D (а) и Т (б, в), и диаграмма состояний счетного триггера

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 тактирования 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).

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 \bar{Q} , и он переключится. После этого, если T остается единичным, 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 _н	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

$$S = JK\bar{Q}/JK\bar{Q} = J\bar{Q}, R = \bar{J}KQ/JKQ = KQ.$$

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 Т-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.

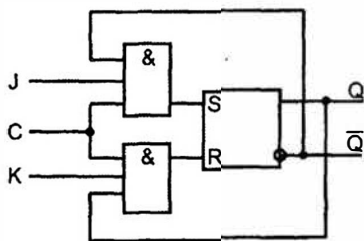


Рис. 3.8. Схема информационных связей, образующих JK-триггер

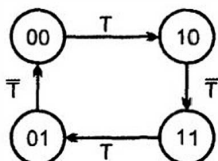


Рис. 3.9. Диаграмма состояний Т-триггера, построенного на двухступенчатых триггерах

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 \bar{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.

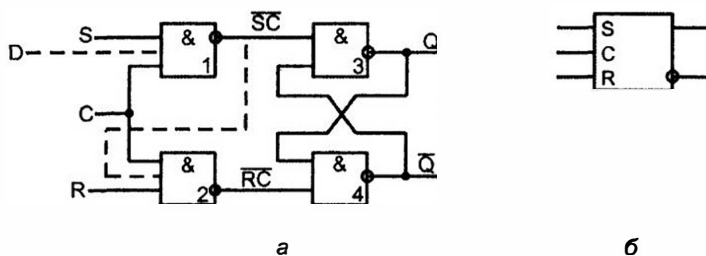


Рис. 3.10. Схема синхронного RS-триггера с управлением уровнем (а) и ее условное обозначение (б)

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-зашелки 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 $Q = 0$.

Триггеры с динамическим управлением воспринимают информационные сигналы только в момент перепада тактового сигнала (точнее в окрестностях этого момента).

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. 3.12), с инвертором (рис. 3.13), с запрещающими связями.

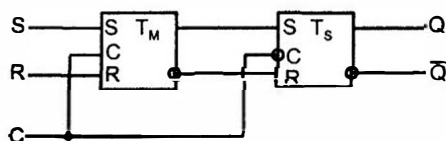


Рис. 3.12. Схема двухступенчатого триггера с разнополярным управлением

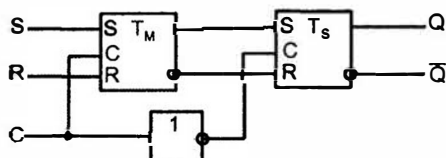
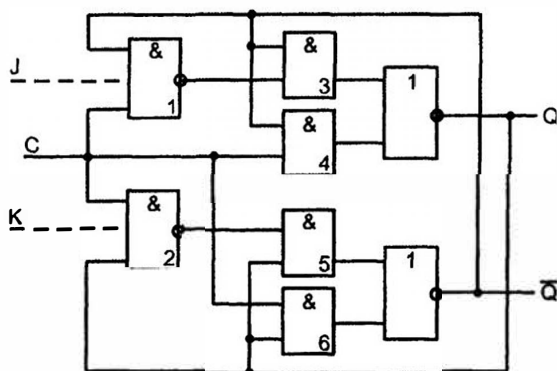


Рис. 3.13. Схема двухступенчатого триггера с инвертором

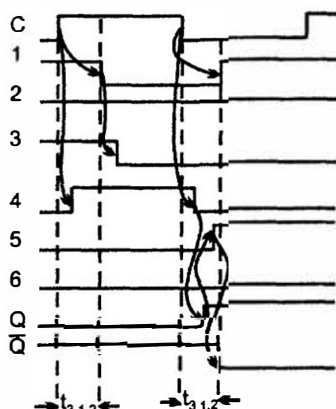
В первом варианте антисинхронное тактирование ступеней очевидно, поскольку ступени имеют соответствующие синхровходы. Во втором варианте ступени идентичны по синхровходам, а для их антисинхронного управления в цепь тактовых сигналов включен инвертор. В такой схеме возможны временные состязания сигналов: 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 допустимостью смены информационных сигналов при любом уровне тактового сигнала. Старые разновидности двухступенчатых триггеров из-за явлений "захвата единицы" и "захвата нуля" [28] 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.



а



б

Рис. 3.14. Схема JK-триггера с внутренними задержками (а) и временные диаграммы ее работы (б)

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 шестизлементные с управле-

нием фронтом. Принцип работы шестизадающих триггеров рассмотрен ранее. Функционирование одноступенчатого триггера с внутренней задержкой (рис. 3.14, а) можно рассмотреть с помощью временных диаграмм (рис. 3.14, 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 $t_{3.1, 2} > t_{\text{И}} + T_{\text{или-НЕ}}$ (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. 3.15 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_H , что облегчает построение некоторых узлов на основе таких триггеров.

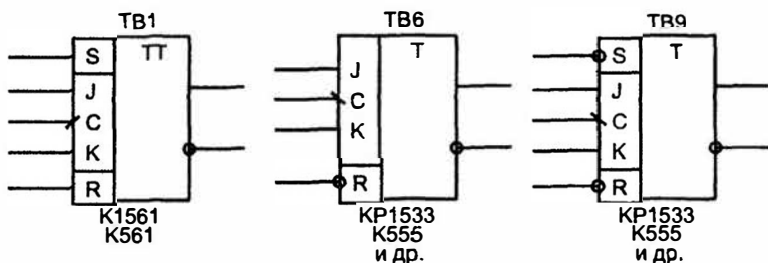


Рис. 3.15. Схемы стандартных триггеров типа JK

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 \bar{S} и \bar{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. Например, для триггера типа K561TB9 функционирование триггера представлено табл. 3.8.

Таблица 3.8

\bar{S}	\bar{R}	C	J	K	Q_H	\bar{Q}_H	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	\downarrow	H	H	\bar{Q}	Q	Счетный
H	H	\downarrow	L	H	L	H	Загрузка нуля (сброс)
H	H	\downarrow	H	L	H	L	Загрузка единицы (установка)
H	H	\downarrow	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 \downarrow 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
K555TB9	ТТЛШ	2	15	30	40
K531TB9	ТТЛШ	2	6,2	80	250
K500TB135	ЭСЛ	2	3	170	290
K1533TB15	ТТЛШ	2	12	40	24
K561TB1	КМОП	2	170	3	0,02 статическая
K1554TB9	КМОП	2	10	140	0,035 статическая

На рис. 3.16 и рис. 3.17 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.

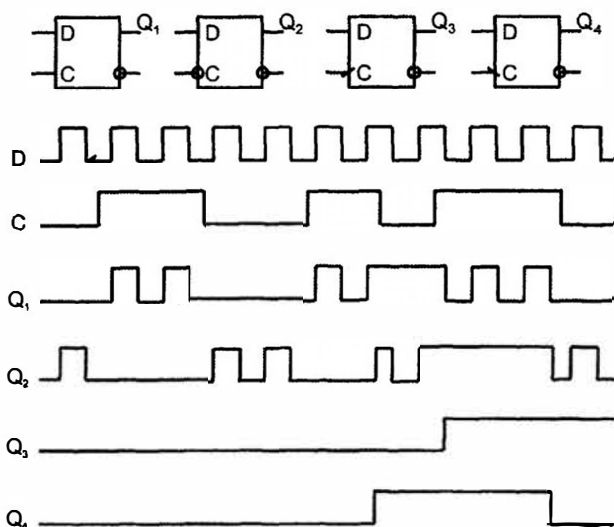


Рис. 3.16. Иллюстрация к работе триггеров типа D

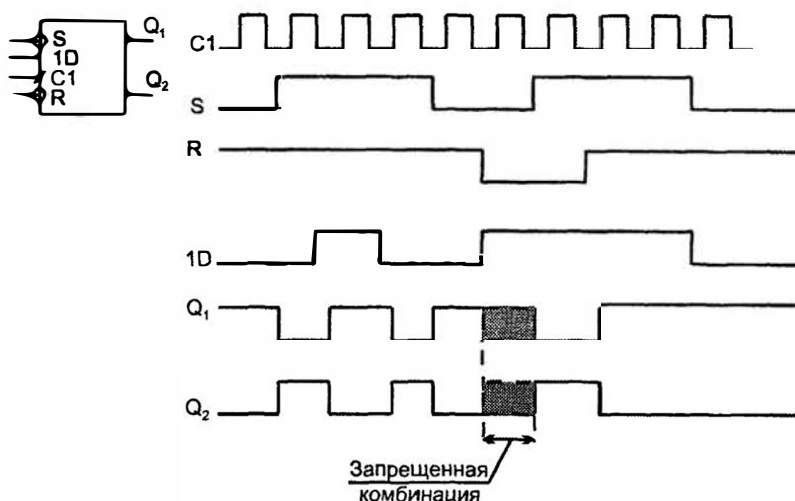


Рис. 3.17. Иллюстрация к работе триггера типа RSD

§ 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 — неустраняемые 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 [1].

§ 3.4. Применение триггеров в схемах ввода и синхронизации логических сигналов

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. 3.18, 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.

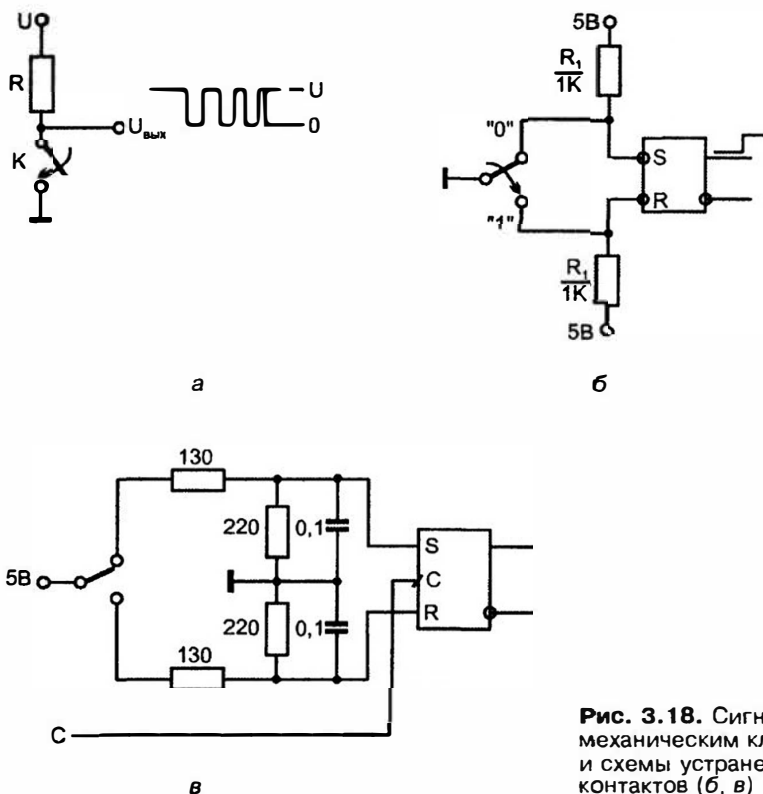


Рис. 3.18. Сигнал, формируемый механическим ключом (а), и схемы устранения дребзга контактов (б, в)

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. 3.18, б), в которой верхнее положение ключа устанавливает триггер (на входе \bar{S} нулевое напряжение, на входе \bar{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 \bar{R} нулевое напряжение, на входе \bar{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. 3.18, 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 $I_{вх1}$.

Синхронизированный с тактовыми сигналами ввод (рис. 3.18, 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 $I_{вх0}$. 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. 3.19. Рассмотрение временных диаграмм его работы свидетельствует о сужении входного импульса относи-

§ 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 $Q_n = f(Q, X)$ и $Y = \varphi(Q, X)$, где Q_n 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 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. 3.21, 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. 3.21, б) имеются специальные синхросигналы (тактирующие импульсы) C , которые разрешают элементам памяти прием данных только в определенные моменты времени. Элементами памяти слу-

жат 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.

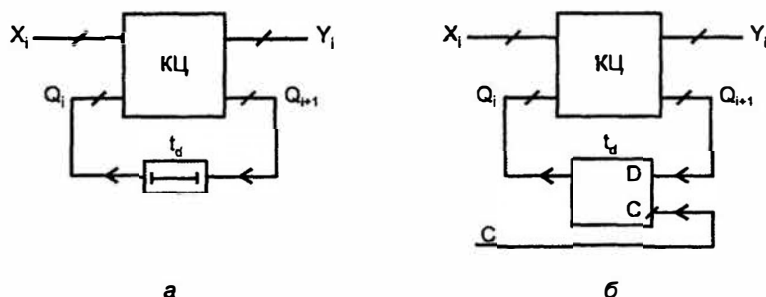


Рис. 3.21. Асинхронный (а) и синхронный (б) автоматы с памятью

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 асинхронные 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.

В работах отечественных и зарубежных ученых 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 $Q_n = f(Q, X)$ 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 чис-

лом триггеров, экономия их числа при построении автомата незначительна. 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.

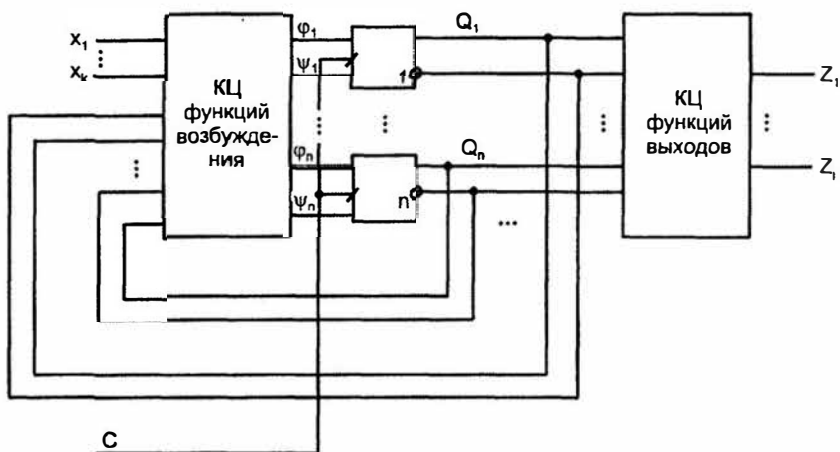


Рис. 3.22. Структурная схема автомата Мура

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. 3.22).

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.

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, \dots, x_k и текущего состояния Q_1, Q_2, \dots, Q_n . Предмет синтеза — получение функций возбуждения φ_i и ψ_i 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, \dots, Z_j осуществляется комбинационной схемой, на которую подаются только выходы триггеров Q_1, Q_2, \dots, 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 _н (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 $\varphi_1, \psi_1, \dots, \varphi_n, \psi_n$ 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). Ниже даны 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 М. 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

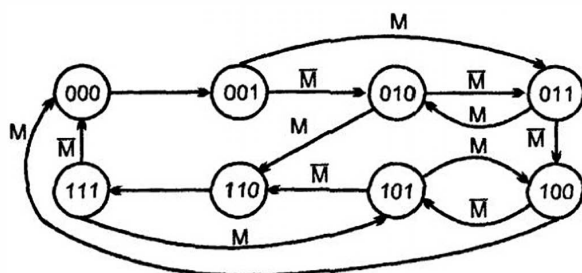


Рис. 3.23. Диаграмма состояний автомата для примера его проектирования

Кодирование состояний автомата, являющегося автоматом Мура, определяется здесь самой постановкой задачи. Диаграмма состояний автомата показана на рис. 3.23. Изменение управляющего сигнала М сразу же ведет к изменению режима, т. е. следующее состояние будет принадлежать уже другому коду.

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
М	Q ₂ Q ₁ Q ₀	Q _{2H} , Q _{1H} , Q _{0H}	М	Q ₂ Q ₁ Q ₀	Q _{2H} , Q _{1H} , Q _{0H}
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

$$Q_{ni} = F(x_1, x_2, \dots, x_k, Q_1, Q_2, \dots, Q_n).$$

Эту функцию можно разложить следующим образом

$$Q_{ni} = f_i \bar{Q}_i \vee g_i Q_i, \quad (a)$$

где функции f и g уже не содержат соответственно переменных Q_i и \bar{Q}_i .

Характеристическое уравнение триггера типа JK имеет вид:

$$Q_{ni} = J_i \bar{Q}_i \vee \bar{K}_i Q_i, \quad (б)$$

Сопоставляя выражения (а) и (б), получим $f_i = J_i$, $\bar{g}_i = K_i$.

Следовательно, положив в функции переходов триггера $Q_i = 0$, сразу получаем функцию возбуждения для входа J:

$$Q_{ni} \Big|_{Q_i = 0} = f_i = J_i,$$

а приняв условие $Q = 1$, можно получить функцию возбуждения для входа K_i :

$$Q_{ni} \Big|_{Q_i = 1} = g_i = \bar{K}_i, \text{ т. е. } \bar{K}_i = g_i.$$

При этом члены формул для Q_{ni} , не содержащие \bar{Q}_i и Q_i , преобразуются путем умножения на $Q_i \vee \bar{Q}_i = 1$ в расширенную форму, в которой все слагаемые содержат переменные \bar{Q}_i или Q_i .

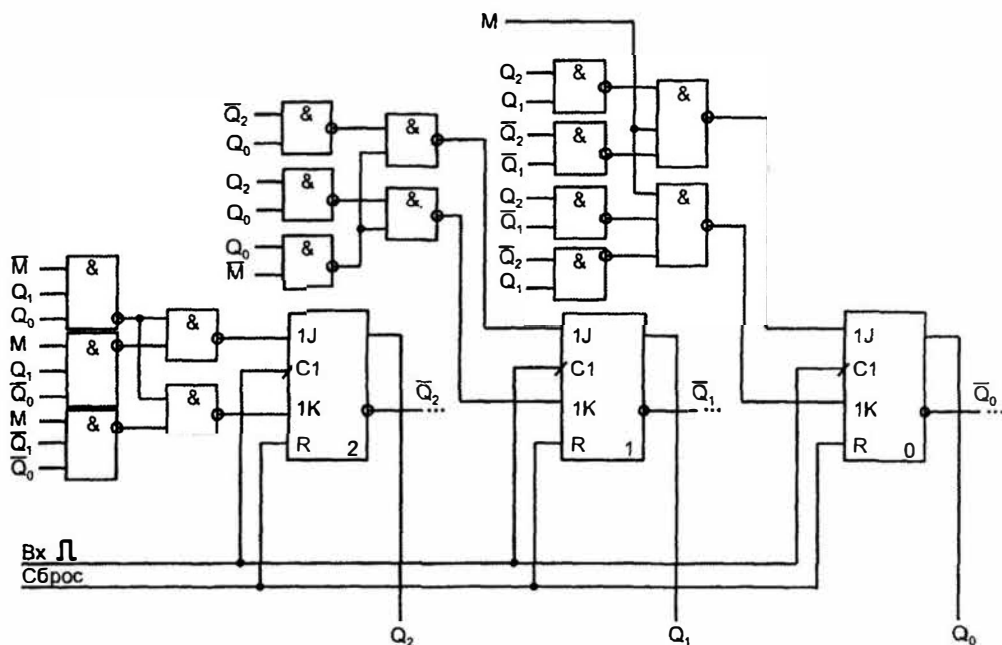


Рис. 3.24. Схема автомата на триггерах JK для примера проектирования

В результате можно получить выражения:

$$J_2 = \overline{MQ_1Q_0} \vee MQ_1\bar{Q}_0 = \overline{MQ_1Q_0} \cdot \overline{MQ_1Q_0};$$

$$K_2 = \overline{MQ_1Q_0} \vee MQ_1\bar{Q}_0 = \overline{MQ_1Q_0} \cdot \overline{MQ_1Q_0};$$

$$J_1 = \overline{M}Q_0\overline{Q_2}Q_0 = \overline{M}Q_0 \cdot \overline{Q_2}Q_0;$$

$$K_1 = \overline{M}Q_0\overline{Q_2}Q_0 = \overline{M}Q_0 \cdot \overline{Q_2}Q_0;$$

$$J_0 = \overline{M}\overline{Q_2}\overline{Q_1}\overline{Q_2}Q_1 = \overline{M} \cdot \overline{Q_2}\overline{Q_1} \cdot \overline{Q_2}Q_1;$$

$$K_0 = \overline{M}\overline{Q_2}Q_1\overline{Q_2}Q_1 = \overline{M} \cdot \overline{Q_2}Q_1 \cdot \overline{Q_2}Q_1.$$

Схема автомата приведена на рис. 3.24.

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. 3.25. Входные сигналы $x_0 \dots x_{m-1}$ и значения разрядов слова старого состояния $Q_0 \dots Q_{n-1}$ образуют управляющее (адресное) входное слово мультиплексора, по которому выбираются значения разрядов нового слова состояния. Поступление тактового импульса ТИ вводит новое слово состояния в триггеры.

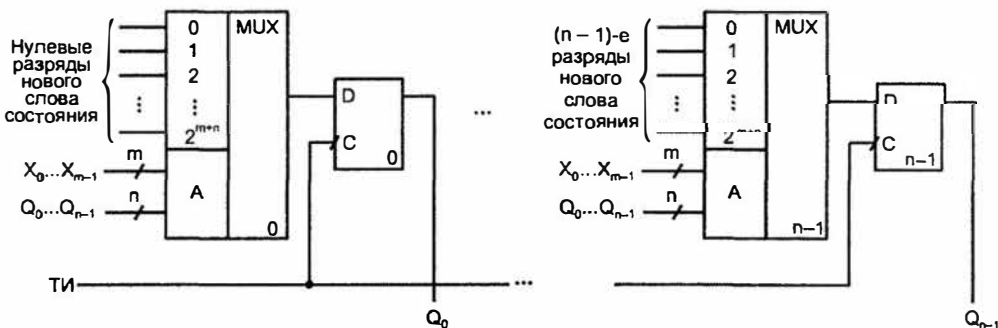


Рис. 3.25. Структура автомата на триггерах с мультиплексным управлением

Для рассматриваемого примера автомат реализуется структурой (рис. 3.27) с числом триггеров 8. Переход в следующее состояние происходит как переход единственной единицы из триггера в соседний триггер, что осуществляется крайне не 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 М. 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.

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. 3.28).

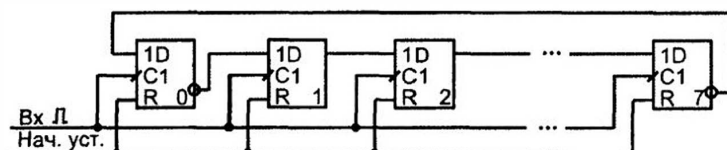


Рис. 3.28. Схема установки автомата в исходное состояние при использовании кода "1 из N"

§ 3.6. Синхронизация в цифровых устройствах

Как уже отмечалось, основным методом построения работоспособных цифровых устройств в настоящее время является синхронизация, устраняющая критические временные соотношения 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.

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. 3.29, а), так и при последовательном выполне-

нии различных операций в разных временных тактах на одном и том же оборудовании (рис. 3.29, 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 $t_{\text{кц.min}}$ и $t_{\text{кц.max}}$.

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.

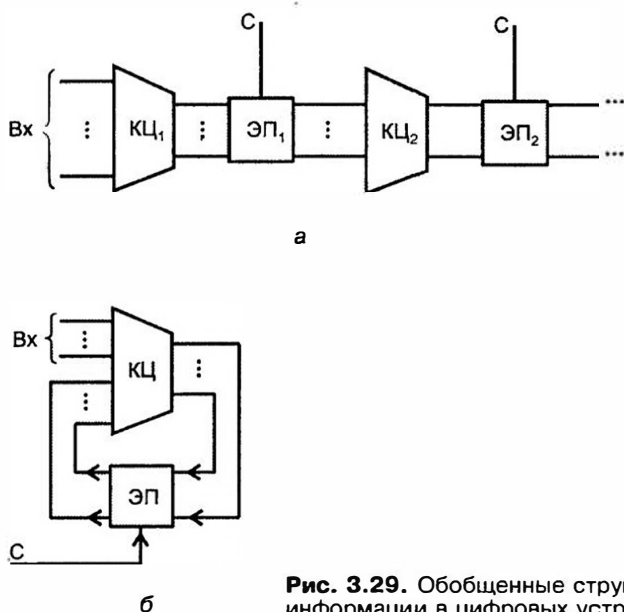


Рис. 3.29. Обобщенные структуры тракта обработки информации в цифровых устройствах (а, б)

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 $t_n \geq t_{wc}$.

Новое состояние триггеры примут по истечении максимальной из задержек t_3^{01} и t_3^{10} их переключения. Параметры t_{wc} и $\max\{t_3^{01}, t_3^{10}\}$ зачастую близки, но могут и отличаться в два и более раз. Разность $\max\{t_3^{01}, t_3^{10}\} - t_{wc}$ обозначим через Δt_{tr} (рис. 3.30).

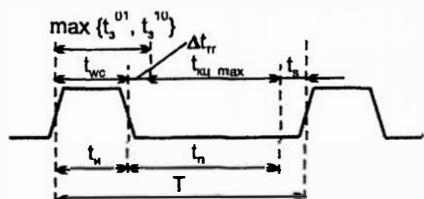


Рис. 3.30. Иллюстрация к определению параметров синхросигналов

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:

$$t_n \geq \Delta t_{tr} + t_{кц. max} + t_s.$$

Минимальный период тактовых импульсов $T_{min} = t_{н. min} + t_{п. min}$, 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 $10^{-4} \dots 10^{-5}$. В более простых генераторах нестабильность частоты существенно выше. Она, в конечном счете, приводит к потере быстродействия устройства. Действительно, частоту синхросигналов 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 воз-

можная 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

$$t_2 - t_1 = (U_{\text{пор}2} - U_{\text{пор}1})/K,$$

где K — крутизна фронта синхроимпульса; $U_{\text{пор}2}$ и $U_{\text{пор}1}$ — 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. 3.31) 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.

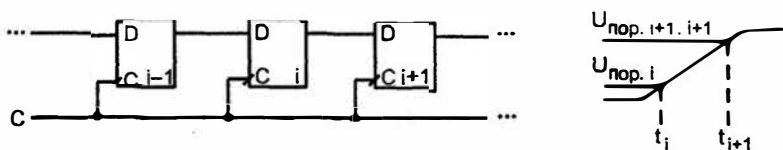


Рис. 3.31. Схема передачи данных в цепочке синхронных триггеров

Структура устройств синхронизации

Обобщенная структура устройства синхронизации (рис. 3.32) 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 син-

хронизации, 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.

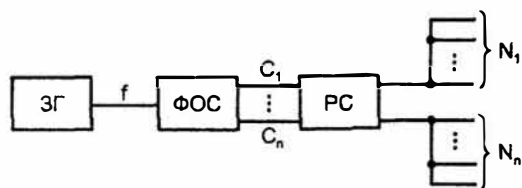


Рис. 3.32. Обобщенная структура блока синхронизации

Различают следующие системы синхронизации:

- ☐ однофазную;
- ☐ двухфазную;
- ☐ многофазную.

Размножение тактовых импульсов

Тактовыми импульсами обычно требуется обеспечить большое число элементов памяти. Обычно тактовые импульсы 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 сильнооточными цепями синхронизации.

Типовое решение — размножение тактовых импульсов с помощью разветвляющейся пирамидальной схемы (рис. 3.33, а), число ярусов которой зависит от числа тактируемых элементов памяти и коэффициентов разветвления задающего генератора и буферных каскадов БК. Кроме того, при определении числа ярусов целесообразно учитывать конструкцию устройства, ставя ярусы в соответствие каким-либо конструктивным единицам (ТЭЗам, панелям, рамам и т. п.).

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 $\Delta t_{\max} = m \Delta t_{\text{БК}}$, где m — число ярусов в схеме РС; $\Delta t_{\text{БК}} = (t_{\text{БК max}} - t_{\text{БК min}})$ — разброс задержек БК.

Временные сдвиги между синхроимпульсами, подаваемыми на различные ЭП, приводят к эффектам, равноценным сокращению одних интервалов и удли-

нению 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.

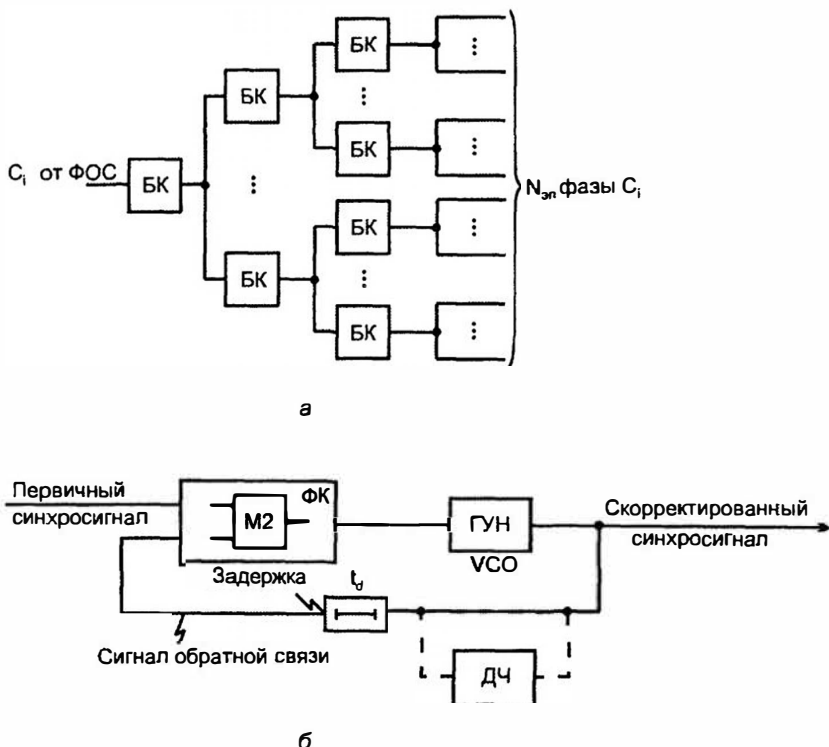


Рис. 3.33. Схема размножения (а) и коррекции (б) тактовых импульсов

Задержки синхросигналов возникают не только в схемах их размножения, но и в цепях передачи тактовых сигналов самих ЦУ.

Коррекция расфазирования синхросигналов

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).

Такие схемы (рис. 3.33, б) содержат фазовый компаратор ФК, генератор, управляемый напряжением ГУН (VCO, Volt-Controlled Oscillator), с выхода которого берутся скорректированные синхроимпульсы, и цепь обратной связи, в которую могут быть включены не только элементы задержки, но и делители частоты ДЧ. При этом на PLL могут возлагаться две функции — коррекция фазовых сдвигов синхросигналов (Clock Skew), осуществляемая замкнутым контуром с элементом задержки в обратной связи (функция Clock Lock), и получение удвоенной частоты синхросигналов при введении в цепь обратной связи делителя частоты ДЧ (функция Clock Boost). Удвоение внутренней частоты работы устройств относительно внешней частоты передачи данных часто используется в микропроцессорах и СБИС программируемой логики высокой сложности.

Благодаря введению схем PLL, удается снижать расфазирование тактовых сигналов системы до очень малых значений.

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 безинерционных 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 ($t_{\text{и}} \geq t_{\text{wc}}$). После переключения триггеров на входах

КЦ появляются новые значения аргументов, а по истечении $t_{\text{кц min}}$ 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

$$t_{\text{wc}} \leq t_{\text{и}} \leq t_{\text{тг min}} + t_{\text{кц min}},$$

где $t_{\text{тг min}}$ — минимальное время переключения триггера.

Как видно, в данном случае необходимо строгое ограничение длительности импульсов снизу и сверху, т. к. за время существования импульса должен 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.

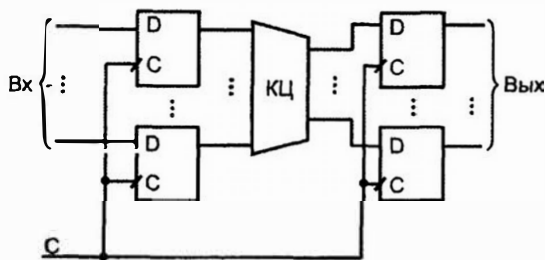


Рис. 3.34. Схема однофазной синхронизации триггеров с динамическим управлением

При использовании триггеров с динамическим управлением (рис. 3.34) информация принимается по фронту синхроимпульса, а чувствительность триггера к информационным сигналам сохраняется лишь в малом интервале времени в окрестности фронта (в течение времени выдержки $t_{\text{н}}$). Триггеры должны потерять чувствительность к изменениям информационных сигна-

лов, прежде чем до их входов по кратчайшему пути может прийти такое изменение. Если это не обеспечивается, возможен сбой. Таким образом, и в этом варианте однофазной системы синхронизации требуется соблюдение определенного условия работоспособности: $t_{\text{тг min}} + t_{\text{кш. min}} \geq t_{\text{н}}$.

Легко заметить, что обеспечить это условие работоспособности гораздо проще, чем предыдущее, т. к. величина $t_{\text{н}}$ мала. Более того, для ряда триггеров, в частности, для JK-триггеров, реализованных по схеме с внутренними задержками, $t_{\text{н}} = 0$. А это значит, что при их применении работоспособность систем с однофазной синхронизацией гарантирована.

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 системы 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 заставить частоту генератора с точностью, 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. 3.35) 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 $t_{Tr} + t_{Kл}$ 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 $\Delta + t_{Tr \max} + t_{Kл \max}$, а наиболее раннее в момент $-\Delta + t_{Tr \min} + t_{Kл \min}$.

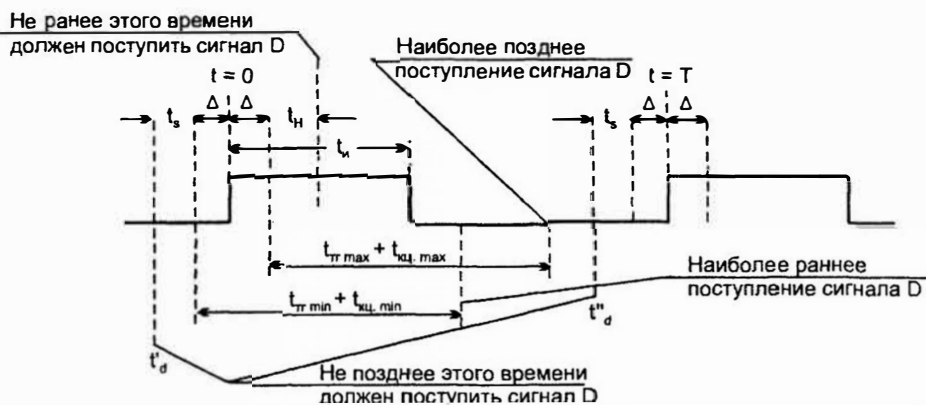


Рис. 3.35. Временная диаграмма синхросигнала однофазной системы синхронизации

Чтобы наиболее позднее поступление информационного сигнала оказалось в допустимой области, необходимо соблюдение условия

$$T - \Delta - t_S \geq \Delta + t_{Tr \max} + t_{Kл \max}$$

Из этого неравенства следует

$$T \geq 2\Delta + t_S + t_{T \max} + t_{K \max}. \quad (a)$$

Условие (а) 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

$$-\Delta + t_{T \min} + t_{K \min} \geq \Delta + t_H,$$

из которого следует условие

$$t_{K \min} \geq 2\Delta + t_H - t_{T \min}. \quad (б)$$

Еще одним необходимым условием является требование длительности импульса, достаточной для надежного переключения триггера

$$t_H \geq 2\Delta + t_{H \min}. \quad (в)$$

Порядок определения параметров синхроимпульсов: выбор t_H по условию (в), выбор T по условию (а), проверка выполнения условия (б).

Слагаемое 2Δ в выражении (в) отражает возможность запаздывания переднего и опережения заднего фронта синхроимпульсов. Нарушение условия (б) 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. 3.36, а), сдвинутых во времени друг относительно друга. Интервал между импульсами обеих последовательностей отводится для работы комбинационных цепей. Соседние каскады получают разноименные серии синхроимпульсов (рис. 3.36, 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.

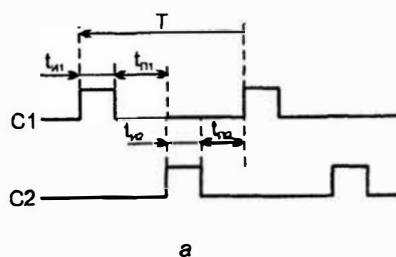
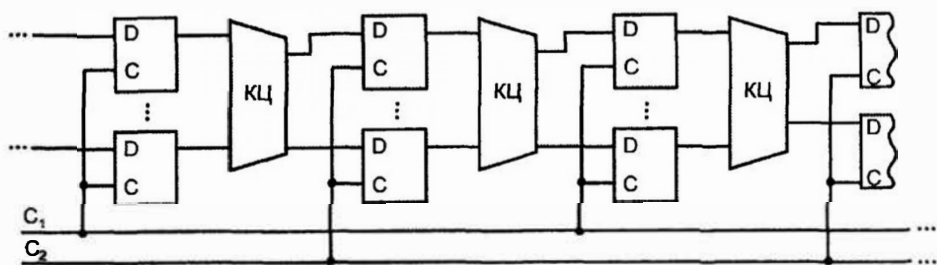


Рис. 3.36. Временная диаграмма синхросигналов (а) и схема тактирования элементов памяти для двухфазной системы синхронизации (б)



б

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

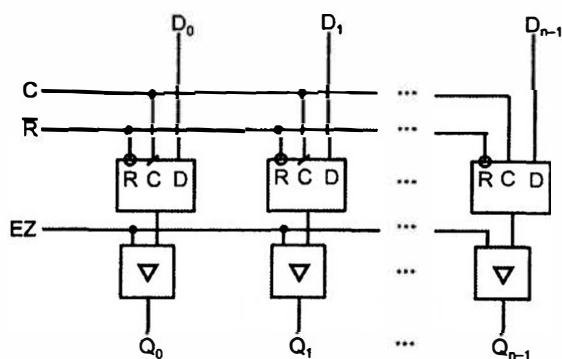
Регистры — самые распространенные узлы цифровых устройств. Они оперируют с множеством связанных переменных, составляющих слово. Над словами выполняется ряд операций: прием, выдача, хранение, сдвиг в разрядной сетке, поразрядные логические операции.

Регистры состоят из разрядных схем, в которых имеются триггеры и, чаще всего, также и логические элементы.

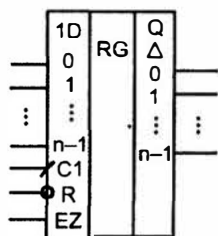
По количеству линий передачи переменных регистры делятся на однофазные и парафазные, по системе синхронизации на одноктактные, двухтактные и многотактные. Однако главным классификационным признаком является способ приема и выдачи данных. По этому признаку различают *параллельные* (статические) регистры, *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).



а



б

Рис. 3.37. Схема статического регистра (а) и его условное графическое обозначение (б)

Последовательно-параллельные регистры имеют входы-выходы одновременно последовательного и параллельного типа. Имеются 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 \bar{R} и выходы с третьим состоянием, управляемые сигналом EZ, показан на рис. 3.37.

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 (серии КР1533, К555 и др.) можно хранить 4 четырехразрядных слова с возможностью независимой и одновременной записи одного слова и чтения другого. Информационные входы регистров соединены параллельно (рис. 3.38). Входы адресов записи WA и WB (от Write) дают четыре комбинации, каждая из которых разрешает "зашелкнуть" данные, присутствующие в настоящее время на выходах D_{1-4} .

Содержимое файла (регистра) вызывается на выходы блока Q_{1-4} с помощью дешифратора считывания (адресных входов мультиплексора) адресами RA и RB (от английского 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 парал-

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.

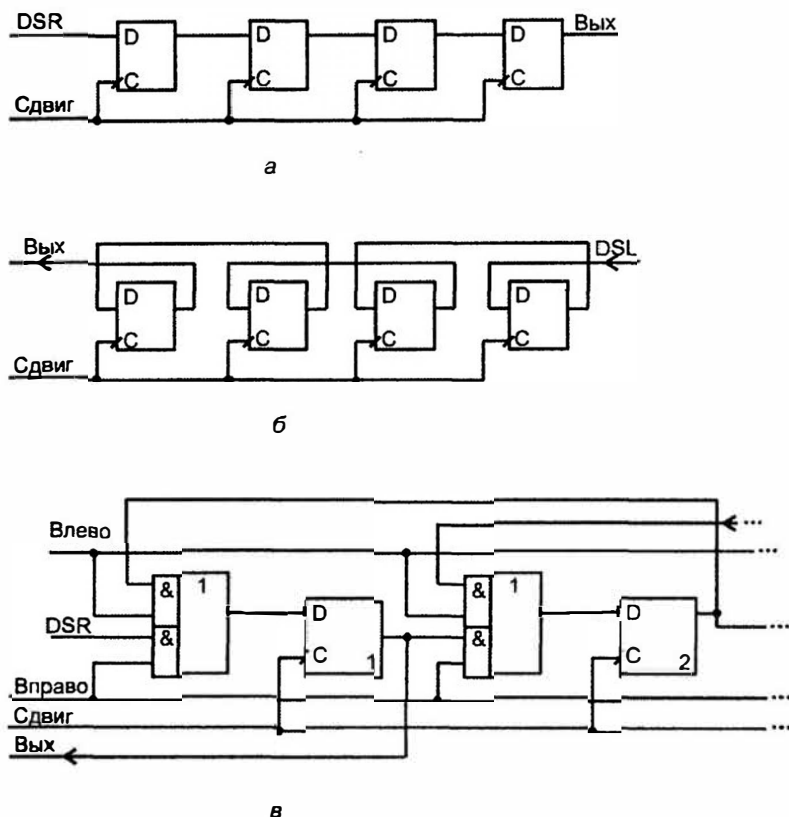


Рис. 3.39. Схемы регистров сдвига вправо (а), влево (б) и реверсивного (в)

Многотактные сдвигающие регистры управляются несколькими синхронными последовательностями. Из их числа наиболее известны двухтактные с основным и дополнительным регистрами, построенными на простых одноступенчатых триггерах, управляемых уровнем. По такту С1 содержимое основного регистра переписывается в дополнительный, а по такту С2 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.

Таблица 3.13

Режим	Входы							Выходы				
	C	\bar{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	\bar{L}	H	L	L	X	X	X	Q_0	Q_1	...	Q_6	Q_7
Сдвиг влево	\bar{L}	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	\bar{L}	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
Параллельная загрузка	\bar{L}	H	H	H	X	X	D_n	D_0	D_1	...	D_6	D_7

Типичным представителем многорежимных регистров является микросхема ИР13 серии КР1533 и других (рис. 3.40). 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 \bar{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. 3.41.

Регистры, имеющие разнотипные вход и выход, служат основными блоками преобразователей параллельных кодов в последовательные и обратно. На рис. 3.42 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 загружается преобразуемое слово D_{1-7} , а в нулевой разряд — константа 0. На последовательный вход DSR подана константа 1.

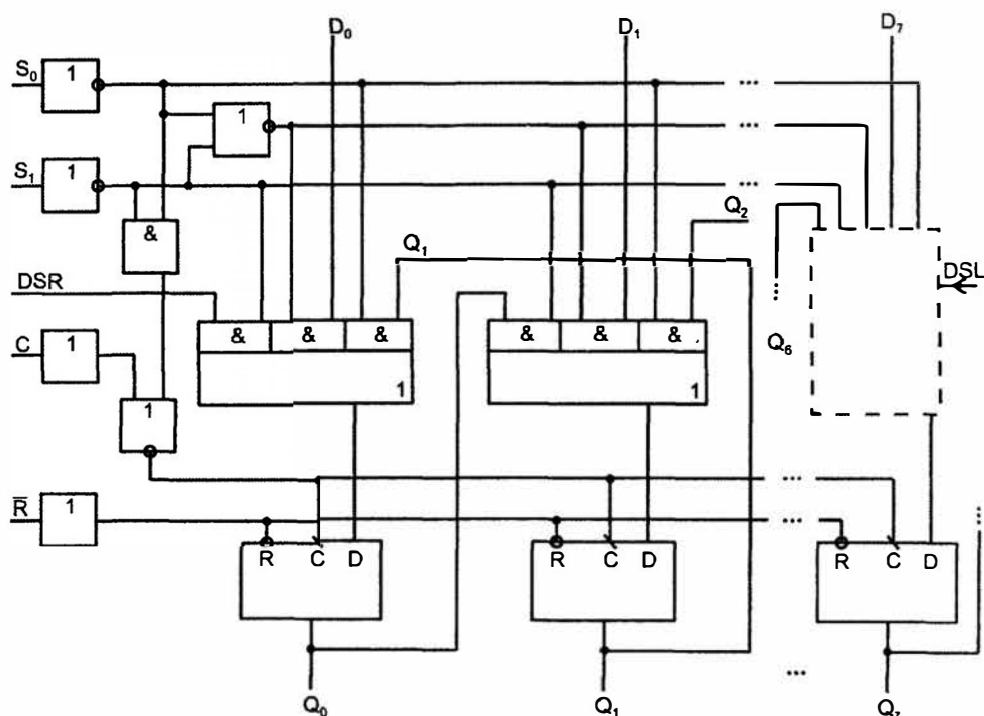


Рис. 3.40. Схема многорежимного регистра

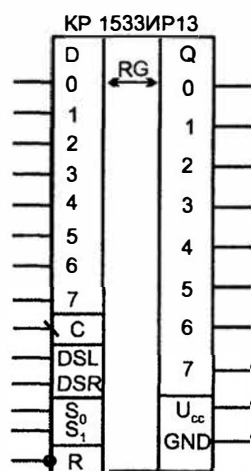


Рис. 3.41. Условное обозначение универсального регистра

Таким образом, после загрузки в регистре формируется слово $0D1D2...D7$. Тактовые импульсы, поступающие на вход C , вызывают сдвиги слова вправо (для условного обозначения это соответствует сдвигу вниз). Сдвиги выводят слово в

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 M (емкостью)*. Модуль определяет число возможных состояний счетчика. После поступления на счетчик M 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 Т (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 со-

стояния отображаются последовательностью двоичных чисел, десятичными эквивалентами которых будут числа $0, 1, 2, 3, \dots, M-1$).

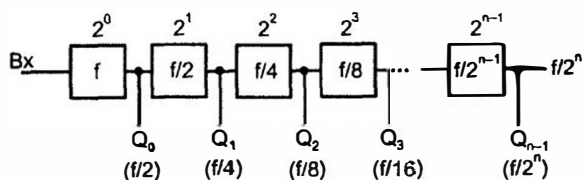
Двоичные счетчики

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 $01010101\dots$, 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 $00110011\dots$. В третьем разряде образуется последовательность из четверок нулей и единиц $00001111\dots$ 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. 3.43, *а*). Заметим, кстати, что согласно ГОСТу входы элементов изображаются слева, а выходы справа. Соблюдение этого правила ведет к тому, что в числе, содержащемся в счетнике, младшие разряды расположены левее старших.

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 $1-0$, а при обратном — при переключении $0-1$. 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. 3.43, *б*.

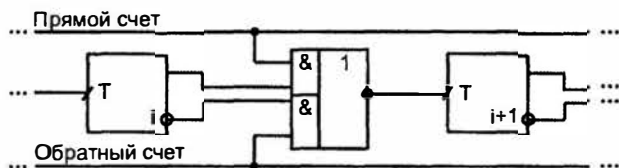
Из различия вариантов прямого и обратного счета следует также и способ построения *реверсивного счетчика* (рис. 3.43, *в*) путем переноса точки съема сигнала с триггера на противоположный выход под действием управляющего сигнала и с помощью элемента И-ИЛИ-НЕ, как показано на рисунке, либо элемента И-ИЛИ.



а



б



в

Рис. 3.43. Структура последовательного счетчика (а), ее реализация на триггерах с прямым динамическим управлением (б) и межразрядные связи реверсивного счетчика (в)

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 $t_{уст} \leq nt_{тг}$. 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 счетчиков является возникновение в переходных процессах ложных состояний из-за задержек переключения триггеров.

На рис. 3.44 показана временная диаграмма работы двухразрядного суммирующего счетчика на триггерах с прямым динамическим управлением, построенная с учетом задержек переключения триггеров t_{π} . 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.

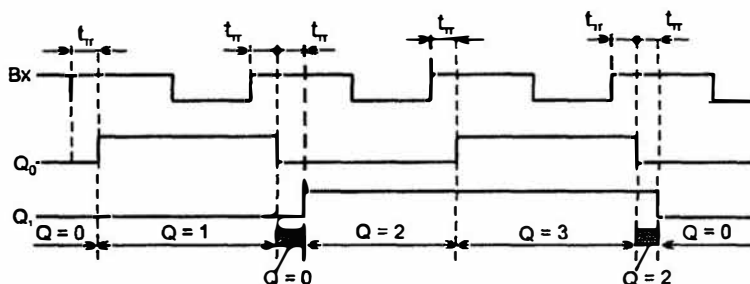


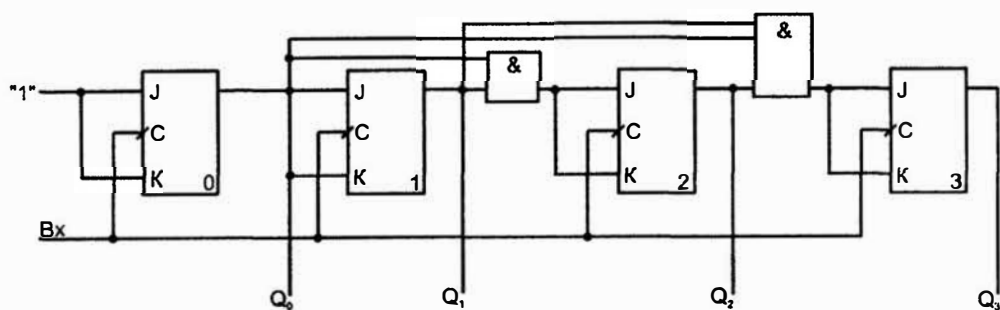
Рис. 3.44. Временные диаграммы работы последовательного двоичного счетчика

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:

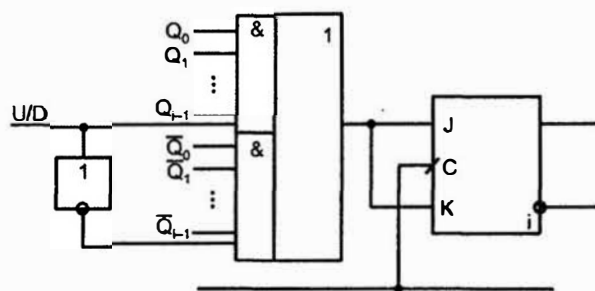
$$\begin{array}{r}
 0 \ 1 \ 1 \ 0 \\
 + \\
 \hline
 0 \ 1 \ 1 \ 0
 \end{array}
 \quad
 \begin{array}{|c|}
 \hline
 0 \ 1 \ 1 \ 1 \\
 \hline
 \end{array}
 \quad
 \begin{array}{r}
 0 \ 1 \ 1 \ 0 \\
 - \\
 \hline
 0 \ 1 \ 1 \ 0
 \end{array}
 \quad
 \begin{array}{|c|}
 \hline
 1 \ 0 \ 0 \ 0 \\
 \hline
 \end{array}
 \quad
 \begin{array}{|c|}
 \hline
 1 \\
 \hline
 \end{array}$$

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 таких счетчиков не зависит от разрядности n и равно $t_{\text{уст}} = t_k + t_{\pi}$, где t_k — задержка конъюнктора. Структура суммирующего синхронного счетчика с параллельным переносом, реализованного на триггерах с управлением фронтом, показана на рис. 3.45, а. Схема межразрядной связи для реверсивного счетчика с сигналом U/D ($U_p/Down$, т. е. прямо/обратно) показана рис. 3.45, б.



а



б

Рис. 3.45. Схемы параллельных счетчиков прямого счета (а) и реверсивного (б)

С ростом числа разрядов реализация параллельных счетчиков затрудняется — требуются вентили с большим числом входов, растет нагрузка на выходы триггеров.

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. 3.46, а). При единичном состоянии всех триггеров группы приход очередного входного сигнала создаст перенос из этой группы. Эта ситуация подготавливает межгрупповой конъюнктор к прямому пропуску входного сигнала на следующую группу. В наихудшем для быстродействия случае, когда перенос проходит через все группы и поступает на вход последней,

$$t_{\text{уст}} = t_k(\ell - 1) + t_{\text{гр}},$$

где ℓ — число групп; $t_{\text{гр}}$ — время установления кода в группе.

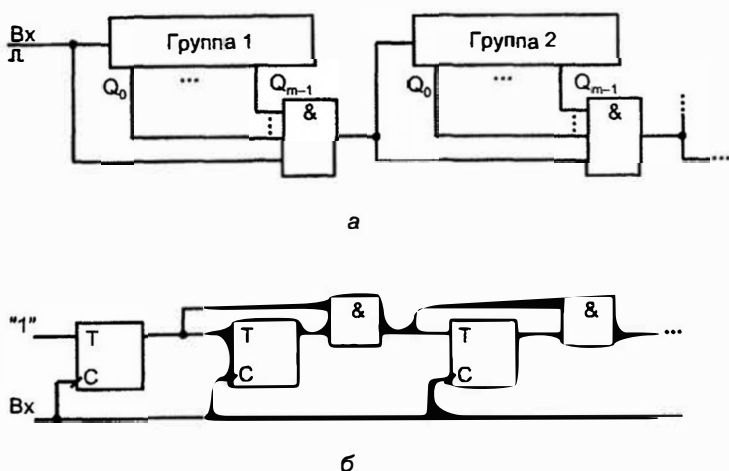


Рис. 3.46. Схемы счетчиков групповой структуры (а, б)

Если уменьшить разрядность группы до единицы и использовать синхронные Т-триггеры, то получится схема *синхронного счетчика с последовательным переносом* (рис. 3.46, 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 дать цепочке вентилях установиться в новое состояние путем их последовательного переключения.

В развитых сериях ИС обычно имеется по 5...10 вариантов двоичных счетчиков, выполненных в виде 4-х разрядных 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, [32]).

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. § 1.3).* Поэтому в руководящих материалах по использованию некоторых БИС/СБИС программируемой логики, в частности, имеется ограничение на раз-

рядность 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. 3.47.

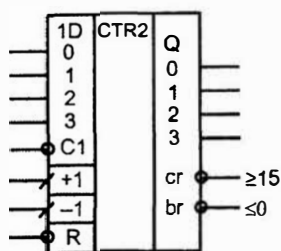


Рис. 3.47. Условное обозначение двоичного реверсивного счетчика со сбросом, параллельной загрузкой и выходами переноса и заема

§ 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 $n = \lceil \log_2 M \rceil$, 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 предполагаем 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 первым способом

Построение счетчика первым способом 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 запи-

шем: $J_2 = Q_1 Q_0$ (в столбце J_2 оставлена всего одна единица), $J_1 = Q_0$, $J_0 = \bar{Q}_2$. 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. 3.48.

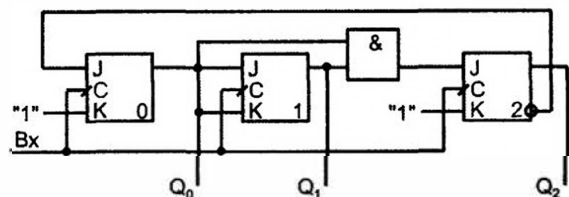


Рис. 3.48. Схема счетчика с модулем 5

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 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. 3.49. 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.

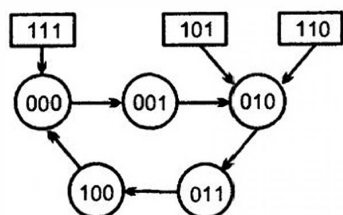


Рис. 3.49. Диаграмма состояний счетчика с модулем 5

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.

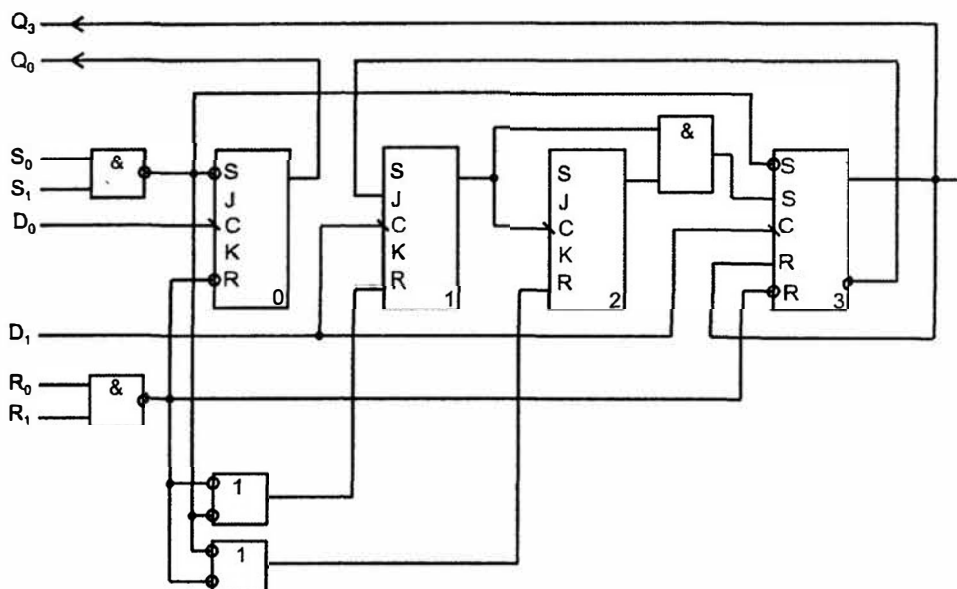


Рис. 3.50. Схема счетчика IE2 серии KP1533

Схема счетчика IE2, которая появилась в самых первых сериях ИС и до сих пор повторяется во всех более новых (рис. 3.50), 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 скажностью 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 ($t_{\text{и}} = T/5$) и симметричных импульсов ($t_{\text{и}} = T/2$), 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).

Построение счетчика вторым способом

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 $J_i = K_i = Q_0 Q_1 \dots Q_{i-1}$ (в младшем триггере $J_0 = K_0 = 1$). Введем в эти функции сигнал сброса R , изменив их следующим образом:

$$J_i = (Q_0 Q_1 \dots Q_{i-1}) \bar{R},$$

$$K_i = J_i \vee R.$$

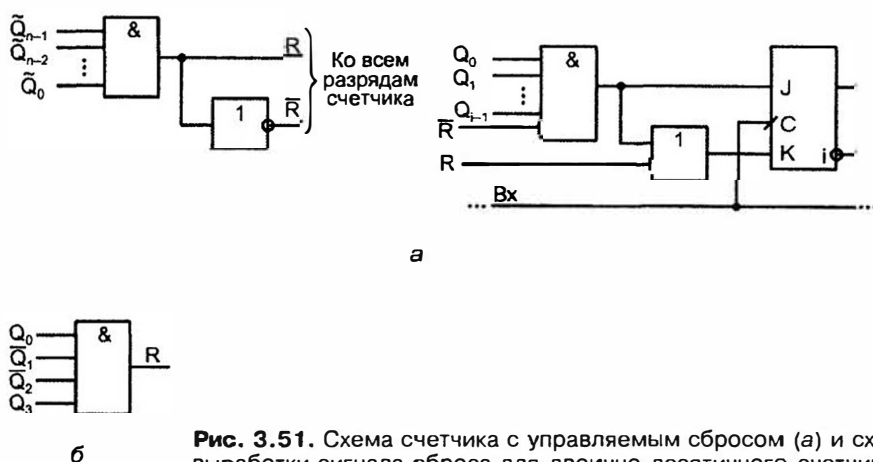


Рис. 3.51. Схема счетчика с управляемым сбросом (а) и схема выработки сигнала сброса для двоично-десятичного счетчика (б)

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 $0, 1, 2, \dots, M-1, 0, \dots$, 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. 3.51, *а*).

Если, например, имеется четырехразрядный счетчик, и на входы конъюнктора выработки сигнала сброса подключены выходы триггеров, как показано на рис. 3.51, *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. § 1.3). 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 § 3.4), 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.

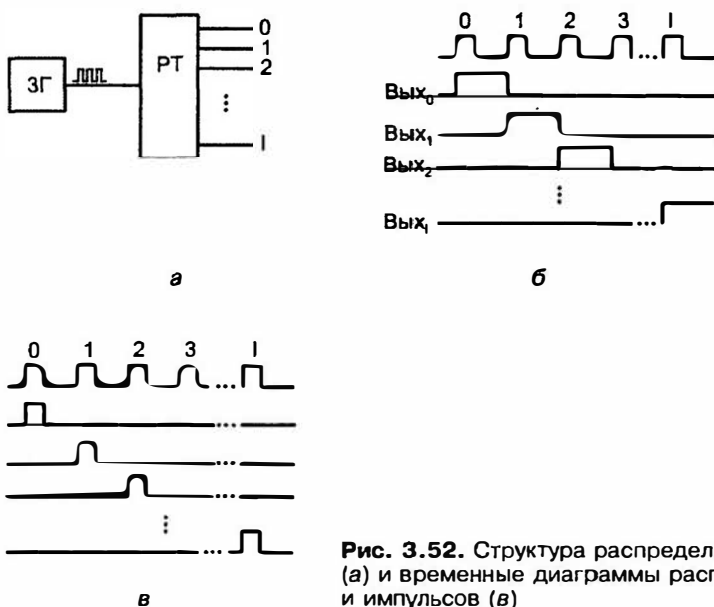


Рис. 3.52. Структура распределителя тактовых сигналов (а) и временные диаграммы распределения уровней (б) и импульсов (в)

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. 3.52, а, б, в, причем диаграмма на рис. 3.52, б соответствует режиму распределения уровней РУ (паузы между активными состояниями каналов отсутствуют), а диаграмма на рис. 3.52, в — режиму распределения импульсов РИ. Распределители импульсов не имеют самостоятельной схемотехни-

ки, они реализуются на основе распределителей уровней путем включения в их выходные цепи конъюнкторов, на вторые входы которых подаются импульсы задающего генератора.

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.

Распределителем тактов 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.

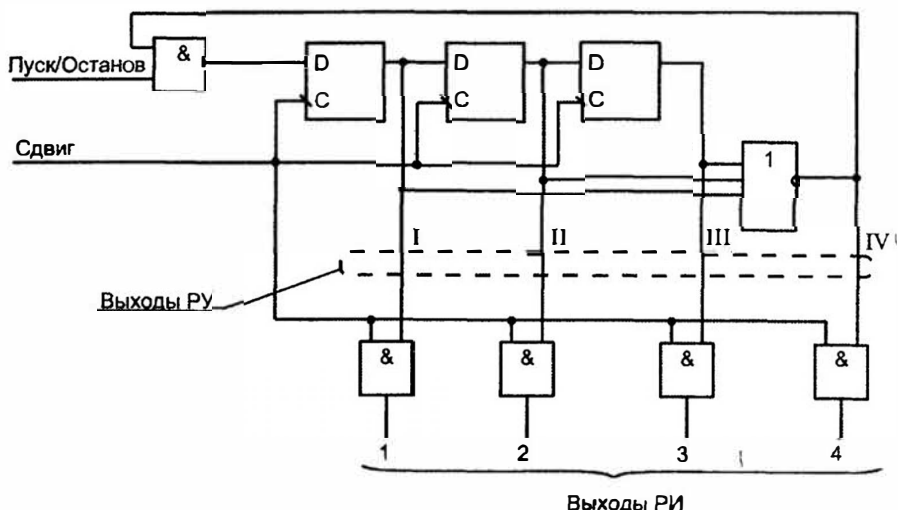


Рис. 3.53. Схема распределителя с автоматическим входением в рабочий цикл

Возможны варианты с *самовосстановлением* работы РТ на кольцевом регистре. Схема такого РТ с самовосстановлением за несколько тактов (рис. 3.53) основана на том, что на вход регистра подаются нули, пока в нем имеется хотя бы одна единица. Таким образом, лишние возникшие единицы будут устранены.

Когда регистр очистится, сформируется сигнал записи единицы на его входе. Следовательно, потеря единственной единицы также будет исключена. Выход логического элемента, выполняющего самовосстановление схемы, даст еще один дополнительный канал. На схеме, приведенной на рис. 3.53, 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. 3.54, а. Ей соответствует следующая таблица истинности (табл. 3.15):

Таблица 3.15

Q ₁	Q ₂	Q ₃	Q _{1Н}	Q _{2Н}	Q _{3Н}	Q ₁	Q ₂	Q ₃	Q _{1Н}	Q _{2Н}	Q _{3Н}
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 $D_1 = Q_{1Н}$ имеем следующие соотношения:

$$D_3 = \bar{Q}_1 \bar{Q}_2 \bar{Q}_3, D_2 = Q_1 \bar{Q}_2 \bar{Q}_3 \text{ и } D_1 = Q_3 \vee Q_1 Q_2 \vee \bar{Q}_1 \bar{Q}_2.$$

Схема распределителя показана на рис. 3.54, 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 аппаратным 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).

Кольцевой регистр с перекрестной обратной связью (счетчик Джонсона, счетчик Мебиуса, счетчик Либау-Крейга) имеет обратную связь на первый триггер от инверсии выходного сигнала (рис. 3.55, 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 обуславливает применение счетчиков Джонсона в составе распределителей.

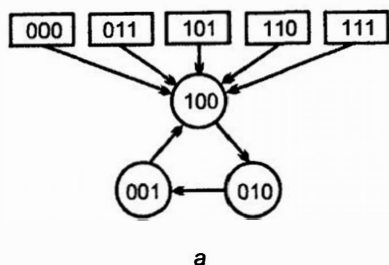
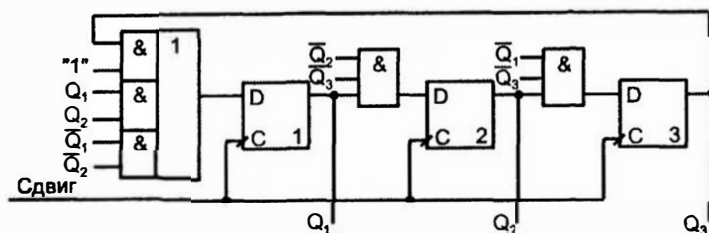


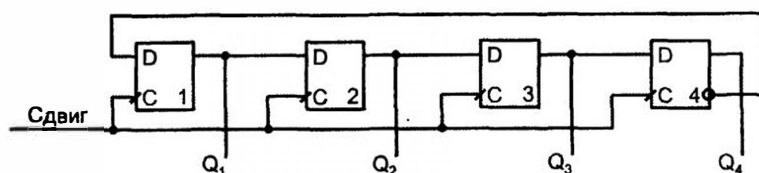
Рис. 3.54. Диаграмма состояний (а) и схема (б) распределителя с автоматическим входением в рабочий цикл за один такт



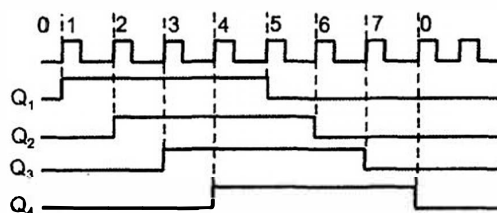
Показанный на рисунке четырехразрядный счетчик Джонсона при начальном нулевом состоянии работает следующим образом. Первый импульс сдвига S установит первый триггер в единичное состояние ($\bar{Q}_4 = 1$), в остальных разрядах будут нули как результат сдвига нулей от соседних слева разрядов. Второй импульс сдвига сохраняет единичное состояние первого триггера, т. к. по-прежнему $\bar{Q}_4 = 1$. Второй разряд окажется в единичном состоянии, 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 ноль, т. к. теперь $\bar{Q}_4 = 0$. 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. 3.55, 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).



а



б

Рис. 3.55. Схема счетчика Джонсона (а) и временные диаграммы его работы (б)

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

$$F_0 = \bar{Q}_1 \bar{Q}_4, F_1 = Q_1 \bar{Q}_2, F_2 = Q_2 \bar{Q}_3, F_3 = Q_3 \bar{Q}_4;$$

$$F_4 = Q_1 Q_4, F_5 = \bar{Q}_1 Q_2, F_6 = \bar{Q}_2 Q_3, F_7 = \bar{Q}_3 Q_4,$$

где F_i ($i = 0 \dots 7$) — 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. 3.56, а), можно запретить начало импульса в последующем канале до его завершения в предыдущем.

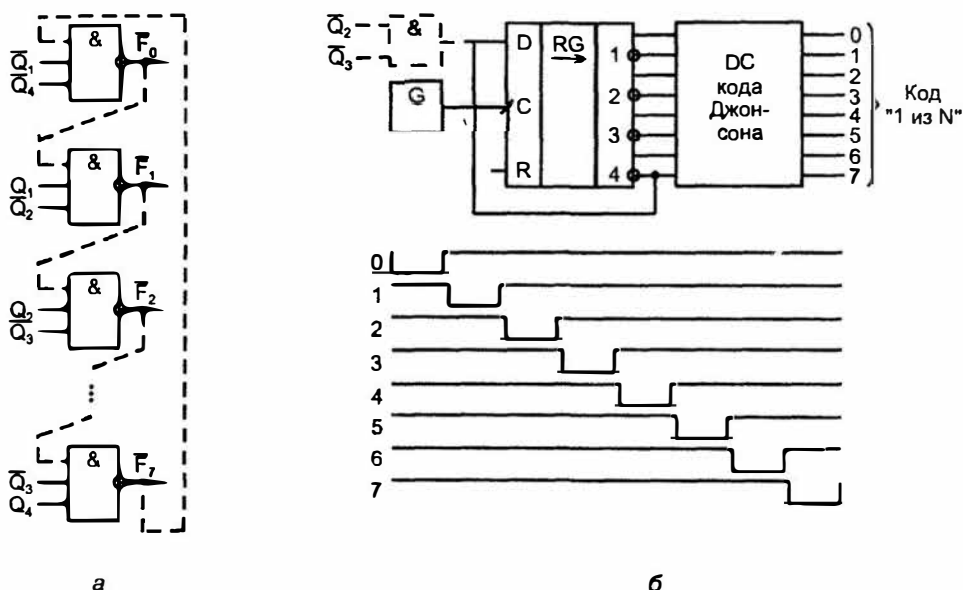


Рис. 3.56. Схемы преобразования кода Джонсона в код "1 из N" (а) и распределителя на основе счетчика Джонсона (б)

Распределитель тактов в целом (рис. 3.56, 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. ноль на инверсном выходе этого разряда. Подавая этот нулевой сигнал на вход счетчика вместе с основным сигналом обратной связи через конъюнктор (показан на рис. 3.56, 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 выше, 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

$$F_{ос} = D_1 = \overline{Q_n} \vee \overline{Q_{n-1}} \dots \overline{Q_2} Q_1.$$

Распределители на основе счетчиков Джонсона характеризуются небольшими аппаратными затратами (1/2 триггера и один двухходовой 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 аппаратной 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

$$F(x_1, \dots, x_n) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus \dots \oplus a_n x_n,$$

где a_i принимают значения 0 или 1 ($i = 0 \dots n$).

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. 3.57. 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

$$F(x) = x^n + \alpha_1 x^{n-1} + \dots + \alpha_{n-1} x + 1.$$

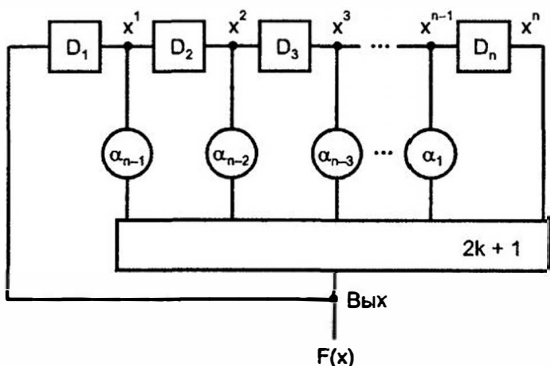


Рис. 3.57. Структура сдвигающего регистра с линейной обратной связью

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 повторяться, т. е. устройство работает периодически. 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 *M-последовательностью*.

К M -последовательности приводят многие варианты схемы. Их поиск основан на изучении полинома, соответствующего схеме. Чтобы генерировалась M -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 отыскивать такие, у которых ми-

нимальное число ненулевых коэффициентов α_i , поскольку это упрощает схему.

Для генерации M -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

п	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. 3.58, а), останавливается сбросом всех триггеров и запускается импульсом старта, записывающим единицу через элемент сложения по модулю 2 в левый триггер. На рис. 3.58, 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.

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 кор-

реляции 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 .

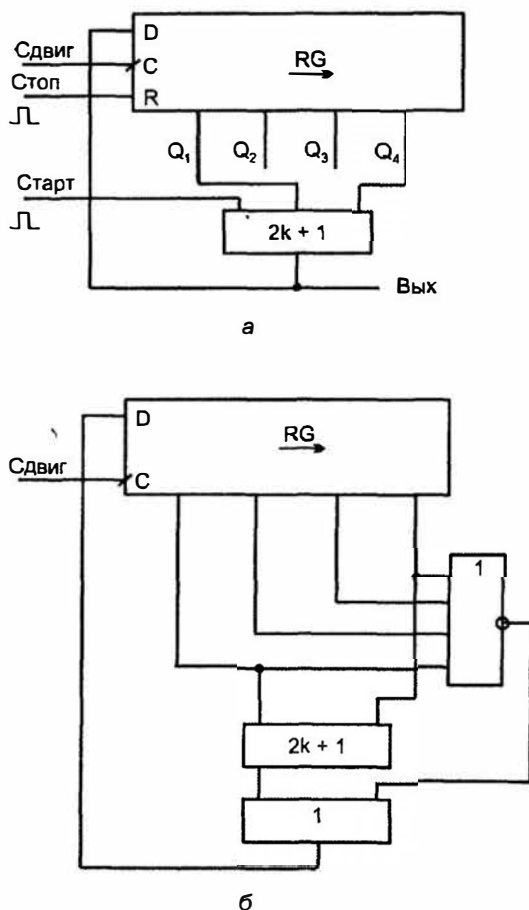


Рис. 3.58. Схемы четырехразрядных генераторов М-последовательностей, запускающегося стартовым импульсом (а) и самозапускающегося (б)

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 (см., например, [40]).

Линейные автоматы на основе сдвигающих регистров используются также в *сигнатурных анализаторах*, являющихся средствами тестового диагностирования цифровых устройств, требующих подачи на них специальных воздейст-

вий, 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: [1], [2], [8], [18], [21], [26], [28], [32], [35], [36], [37], [40], [42], [43], [44].

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 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 наиболее развитой иерархии памяти ЭВМ можно выделить следующие уровни:

- *регистровые ЗУ*, находящиеся в составе процессора или других устройств (т. е. внутренние для этих блоков), 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. Высокое быстродействие кэш-памяти повышает производительность ЭВМ;
- *основная память* (оперативная, постоянная, полупостоянная), работающая в режиме непосредственного 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 $2^{20} = 1048576$.

Организация ЗУ — произведение числа хранимых слов на их разрядность. Видно, что это дает информационную емкость ЗУ, однако при одной и той же информационной емкости организация ЗУ может быть различной, так что организация является самостоятельным важным параметром.

Быстродействие (производительность) ЗУ оценивают временами считывания, записи и длительностями циклов чтения/записи. *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 чтениями или записями образует соответствующий цикл. Длительности циклов могут превышать времена чтения или записи, т. к. после этих операций может потребоваться время для восстановления необходимого начального состояния ЗУ.

Время чтения, записи и длительности циклов — традиционные параметры. Для некоторых современных ЗУ они должны быть дополнены новыми. Причиной является более сложный характер доступа к хранимым данным, когда обращение к первому слову некоторой группы слов (пакета) требует большего времени, чем обращение к последующим. Для таких режимов вводят параметр *времени доступа при первом обращении* (Latency) и *темпа передач* для последующих слов пакета (Bandwidth). 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.

Помимо указанных основных параметров для ЗУ указывают еще целый набор временных интервалов. Перечисленные выше динамические параметры являются *эксплуатационными* (измеряемыми). Кроме них, существует ряд *режимных параметров*, обеспечение которых необходимо для нормального функционирования ЗУ, поскольку оно имеет несколько сигналов управления, для которых должно быть обеспечено определенное взаимное расположение во времени. Для этих сигналов задаются длительности и ограничения по взаимному положению во времени.

Один из возможных наборов сигналов ЗУ (рис. 4.1, а) включает следующие сигналы:

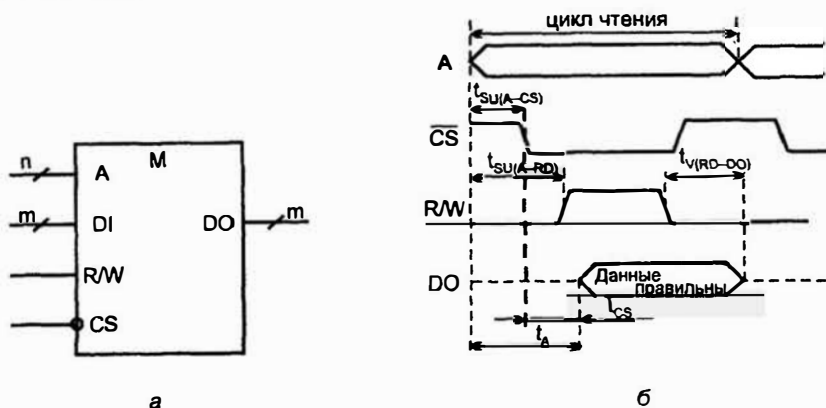


Рис. 4.1. Типичные сигналы ЗУ (а) и их временные диаграммы (б)

- **A** — адрес, разрядность которого n определяется числом ячеек ЗУ, т. е. максимально возможным числом хранимых в ЗУ слов. Для ЗУ типично число ячеек, выражаемое целой степенью двойки. Адрес является номером ячейки, к которой идет обращение. Очевидно, что разрядность адреса связана с числом хранимых слов N соотношением $n = \log_2 N$ (имеется в виду максимально возможное число хранимых слов). Например, ЗУ с информационной емкостью 64К слов имеет 16-разрядные адреса, выражаемые словами

$$A = A_{15}A_{14}A_{13} \dots A_0;$$

- **CS** — (Chip Select) или **CE** (Chip Enable), который разрешает или запрещает работу данной микросхемы;
- **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.

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 другой 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 А (от слова Access) обозначаются согласно стандарту времени 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.

Кроме отмеченных параметров для ЗУ, используется и ряд других (уровни напряжений, токи, емкости выводов, температурный диапазон и т. д.), которые не требуют специального рассмотрения, т. к. они традиционны для цифровой схемотехники. Исключение составляет *свойство энергонезависимости*, т. е. способность ЗУ сохранять данные при отключении напряжения питания. Энергонезависимость может быть естественной, т. е. присущей самим ЗУ, или искусственной, достигаемой введением резервных источников питания, автоматически подключаемых к накопителю ЗУ при снятии основного питания.

Классификация ЗУ

Для классификации ЗУ (рис. 4.2) важнейшим признаком является способ доступа к данным.

При адресном доступе код на адресном входе 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. Запоминающие элементы ОЗУ, как правило, не обладают энергонезависимостью.

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".

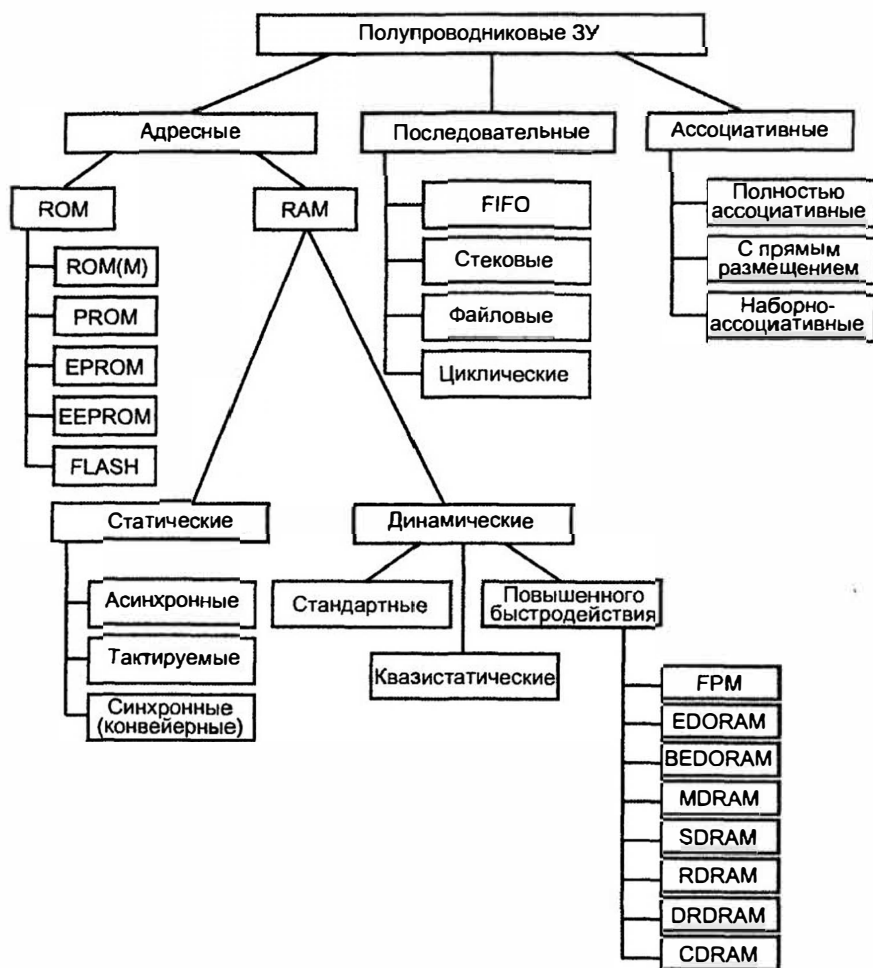


Рис. 4.2. Классификация полупроводниковых ЗУ

RAM делятся на статические и динамические. В первом варианте запоминающими элементами являются триггеры, сохраняющие свое состояние, пока схема находится под питанием и нет новой записи данных. Во втором варианте данные хранятся в виде зарядов конденсаторов, образуемых элементами МОП-структур. Саморазряд конденсаторов ведет к разрушению данных, поэтому они должны периодически (каждые несколько миллисе-

кунд) регенерироваться. В то же время плотность упаковки динамических элементов памяти в несколько раз превышает плотность упаковки, достижимую в статических RAM.

Регенерация данных в динамических ЗУ осуществляется с помощью специальных контроллеров. Разработаны также ЗУ с динамическими запоминающими элементами, имеющие внутреннюю встроенную систему регенерации, у которых внешнее поведение относительно управляющих сигналов становится аналогичным поведению статических ЗУ. Такие ЗУ называют *квазистатическими*.

Статические ЗУ называются SRAM (Static RAM), а динамические — DRAM (Dynamic RAM).

Статические ОЗУ можно разделить на асинхронные, тактируемые и синхронные (конвейерные). В *асинхронных* сигналы управления могут задаваться как импульсами, так и уровнями. Например, сигнал разрешения работы \overline{CS} может оставаться неизменным и разрешающим на протяжении многих циклов обращения к памяти. В *тактируемых* ЗУ некоторые сигналы обязательно должны быть импульсными, например, сигнал разрешения работы \overline{CS} в каждом цикле обращения к памяти должен переходить из пассивного состояния в активное (должен формироваться фронт этого сигнала в каждом цикле). Этот тип ЗУ называют часто синхронным. Здесь использован термин "тактируемые", чтобы "освободить" термин "синхронные" для новых типов ЗУ, в которых организован *конвейерный* тракт передачи данных, синхронизируемый от тактовой системы процессора, что дает повышение темпа передач данных в несколько раз. *Подробнее сущность конвейерной организации ЗУ рассмотрена в § 4.8*, т. к. она играет важную роль в повышении быстродействия динамических ЗУ (вариант SDRAM).

Динамические ЗУ характеризуются наибольшей информационной емкостью и высокой стоимостью, поэтому именно они используются как основная память ЭВМ. Поскольку от этой памяти требуется высокое быстродействие, разработаны многочисленные архитектуры повышенного быстродействия, перечисленные в классификации. Подробнее эти архитектуры рассмотрены в § 4.7.

Статические ЗУ в 4...5 раз дороже динамических и приблизительно в столько же раз меньше по информационной емкости. Их достоинством является высокое быстродействие, а типичной областью использования — схемы кэш-памяти.

Постоянная память типа ROM (М) программируется при изготовлении методами интегральной технологии с помощью одной из используемых при этом масок. В 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 трех разновидностях ROM в обозначениях присутствует буква P (от Programmable). Это *программируемая пользователем память* (в русской терминологии ППЗУ — программируемые ПЗУ). Ее содержимое записывается либо однократно (в PROM), либо может быть заменено путем стирания старой информации и записи новой (в EPROM и EEPROM). В EPROM стирание выполняется с помощью облучения кристалла ультрафиолетовыми лучами, ее русское название РПЗУ-УФ (репрограммируемое ПЗУ с УФ-стиранием). В EEPROM стирание производится электрическими сигналами, ее русское название РПЗУ-ЭС (репрограммируемое ПЗУ с электрическим стиранием). Английские названия расшифровываются как Electrically Programmable ROM и Electrically Erasable Programmable ROM. Программирование PROM и репрограммирование EPROM и EEPROM производится в обычных лабораторных условиях с помощью либо специальных программаторов, либо специальных режимов без специальных приборов (для EEPROM).

Память muna Flash по запоминающему элементу подобна памяти типа EEPROM (или иначе E²PROM), но имеет структурные и технологические особенности, позволяющие выделить ее в отдельный вид.

Запись данных и для EPROM и для E²PROM производится электрическими сигналами.

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. 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 английского слова 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.

§ 4.2. Основные структуры запоминающих устройств

Адресные ЗУ представлены в классификации статическими и динамическими оперативными устройствами и памятью типа ROM. Многочисленные варианты этих ЗУ имеют много общего с точки зрения структурных схем, что делает более рациональным не конкретное рассмотрение каждого ЗУ в полном объеме, а изучение некоторых обобщенных структур с последующим описанием запоминающих элементов для различных ЗУ.

Общность структур особенно проявляется для статических ОЗУ и памяти типа ROM. *Структуры динамических ОЗУ имеют свою специфику и рассмотрены в § 4.7.* Для статических ОЗУ и памяти типа ROM наиболее характерны структуры 2D, 3D и 2DM.

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.

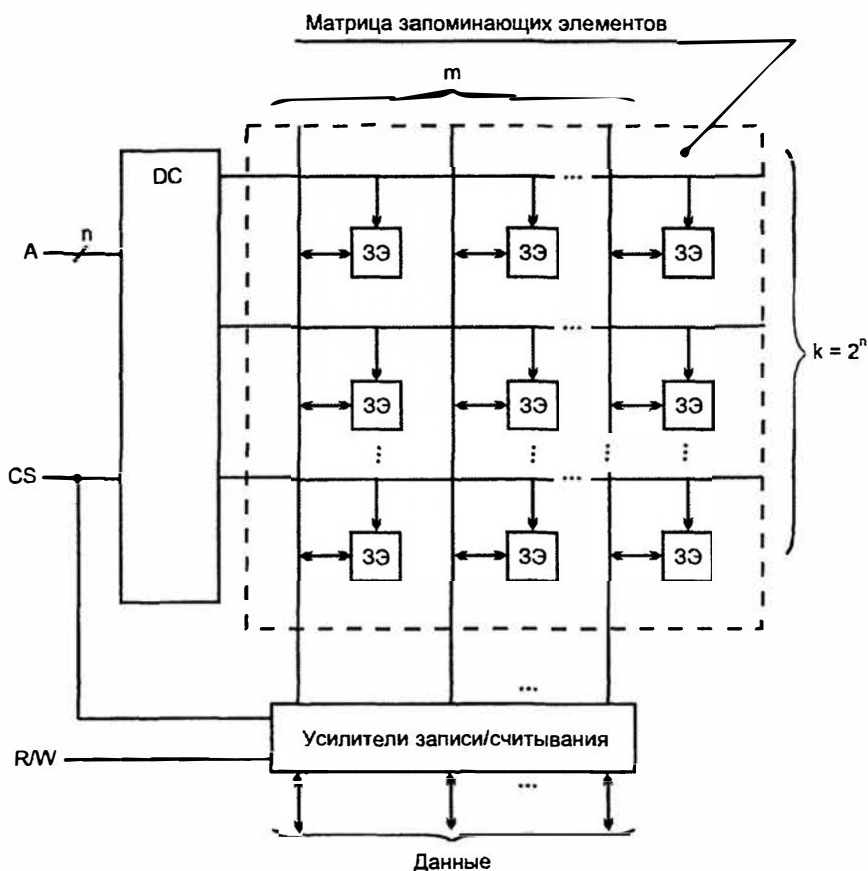


Рис. 4.3. Структура ЗУ типа 2D

Дешифратор адресного кода DC при наличии разрешающего сигнала CS (Chip Select — сигнала выбора микросхемы) активизирует одну из выходных линий, разрешая одновременный доступ ко всем элементам выбранной строки, хранящей слово, адрес которого соответствует номеру строки. Элементы одного 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, наиболее очевидным из которых является чрезмерное усложнение дешифратора адреса (число выходов дешифратора равно числу хранимых слов).

Структура 3D

Структура 3D позволяет резко упростить дешифраторы адреса с помощью двухкоординатной выборки запоминающих элементов. Принцип двухкоординатной выборки поясняется (рис. 4.4, а) на примере 3У типа ROM, реализующего только операции чтения данных.

Здесь код адреса разрядностью n делится на две половины, каждая из которых декодируется отдельно. Выбирается запоминающий элемент, находящийся на пересечении активных линий выходов обоих дешифраторов. Таких пересечений будет как раз

$$2^{n/2} \times 2^{n/2} = 2^n.$$

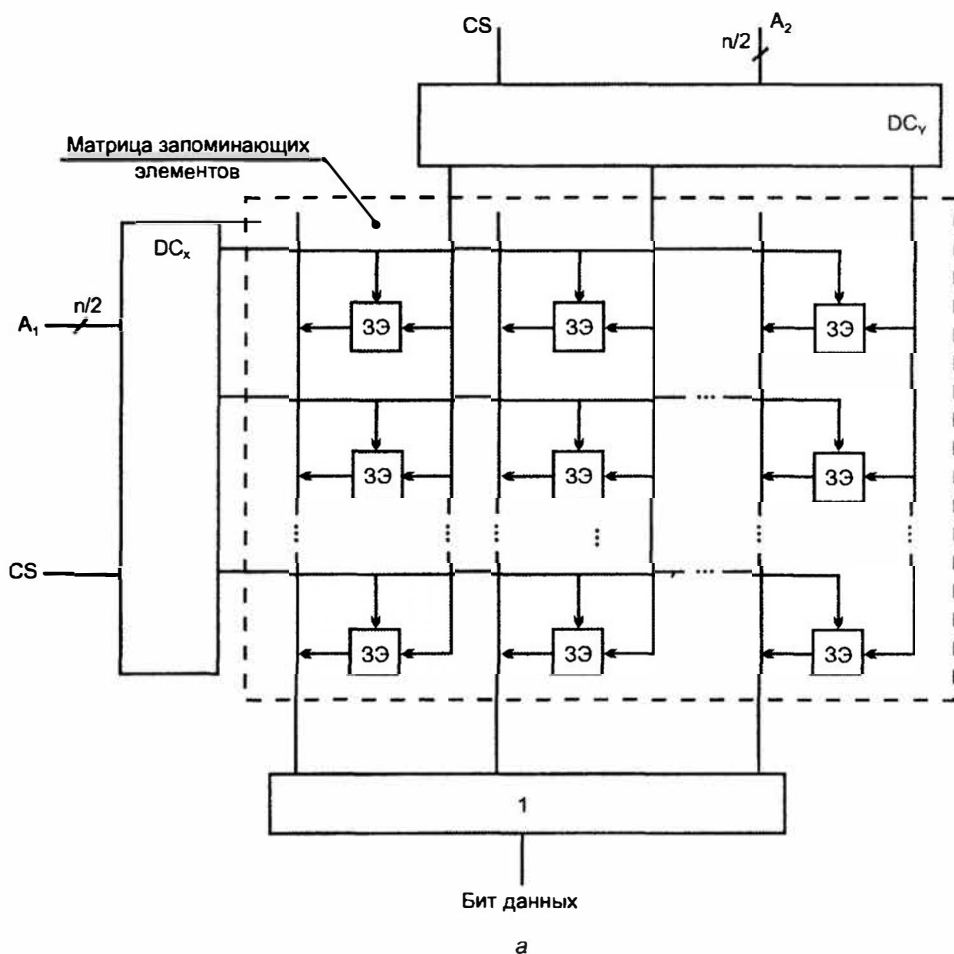
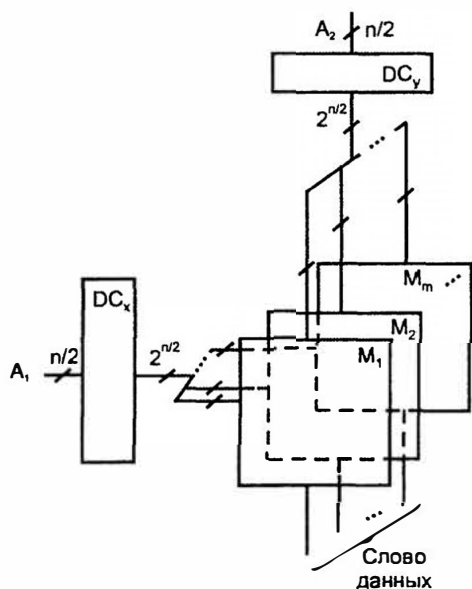


Рис. 4.4. Структура 3У типа 3D с одноразрядной (а) организацией



б

Рис. 4.4. (окончание) Структура ЗУ типа 3D с многоразрядной (б) организацией

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 . Уже для ЗУ небольшой емкости видна эта существенная разница: для структуры 2D при хранении 1К 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.

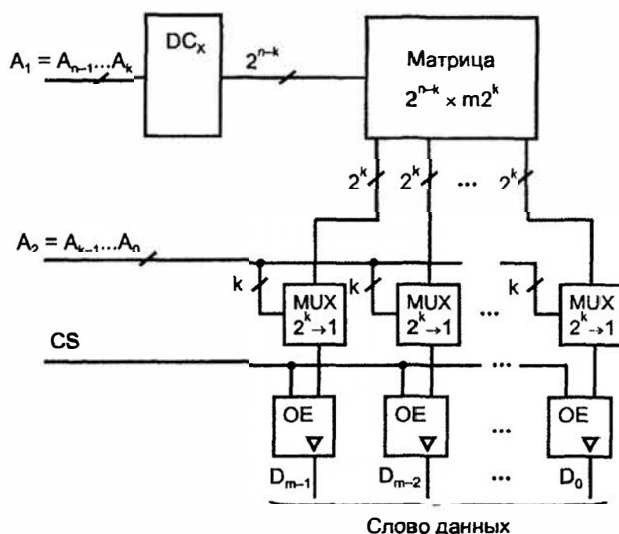
Структура типа 3D, показанная на рис. 4.4, а для ЗУ с одnorазрядной организацией, может применяться и в ЗУ с многоразрядной организацией (рис. 4.4, б), приобретая при этом "трехмерный" характер. В этом случае несколько матриц управляются от двух дешифраторов, относительно которых они включены параллельно. 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 (2D модифицированная) сочетаются достоинства обеих рассмотренных структур — упрощается дешифрация адреса и не требуются запоминающие элементы с двухкоординатной выборкой.

Структура 2DM

ЗУ типа ROM (рис. 4.5, а) структуры 2DM для матрицы запоминающих элементов с адресацией от дешифратора DC_x 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 $A_{n-1} \dots A_k$. 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 $A_{k-1} \dots A_0$. 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, б в более общем виде структура 2DM 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)

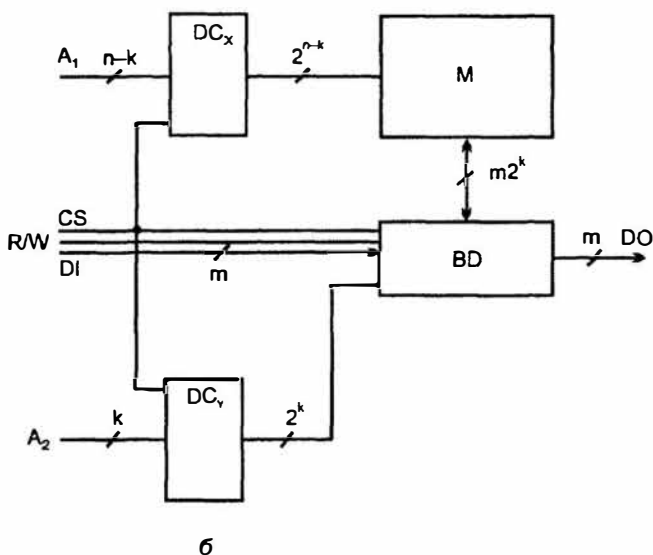


Рис. 4.5. (окончание) Структура ЗУ типа 2DM для RAM (б)

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**.

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 с необходимым управлением адресом доступа.

Основными представителями этого вида памяти являются видеопамять, буфер **FIFO** и стек.

Видеопамять

Видеопамять работает циклично, на ее выходе последовательно в порядке сканирования экрана монитора лучом появляются коды, задающие параметры светимости (цвет, яркость) элементарных точек экрана — пикселей. Текущее изображение на мониторе — кадр — представлено последовательностью слов, длина которой равна числу пикселей экрана. Слово, соответствующее одному пикселу, может иметь разрядность от 8 (для черно-белых мониторов) до 24 (для полноцветного режима).

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 сигналом начала передачи блока данных из основной памяти или видеобuffers. 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. 4.6) показано с представлением элементов хранения и перезаписи данных в виде статических регистров.

При считывании 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. 4.6 для пояснения принципа показан только кадровый). Появление кода синхросигнала на выходе обнаруживается компаратором и синхронизирует запуск развертки монитора.

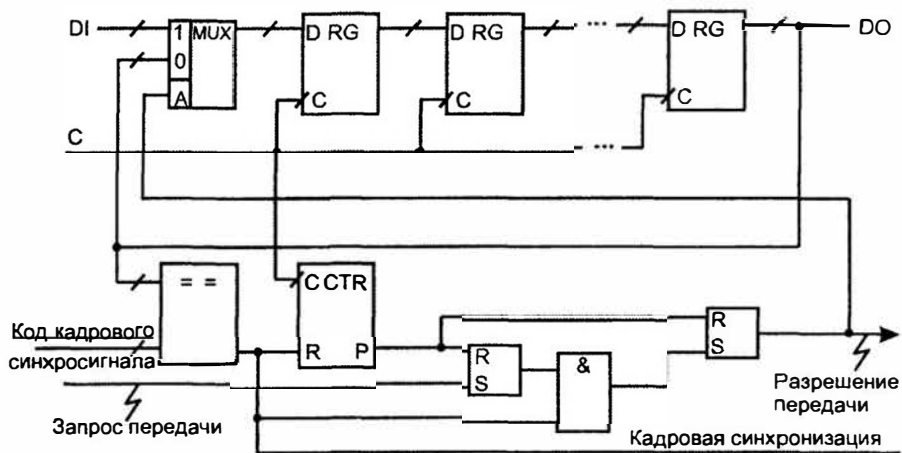


Рис. 4.6. Структура видеопамати

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 мультиплек-

сор 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 переключается на верхний канал на один такт работы, чем обеспечивается замена всего одного слова.

Буфер FIFO

Буфер FIFO, пример структуры которого приведен на рис. 4.7, 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.

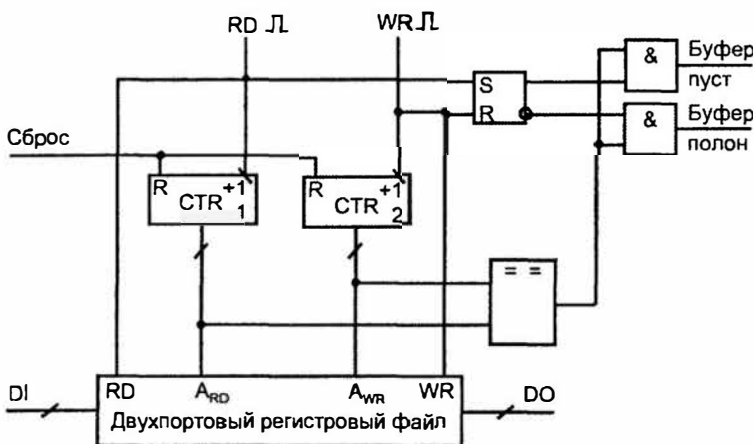


Рис. 4.7. Структура буфера FIFO

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 (рис. 4.7) перед началом работы оба счетчика адресов CTR₁ и CTR₂ 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 сравниваются при чтении, то буфер пуст. Если адреса сравниваются 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).

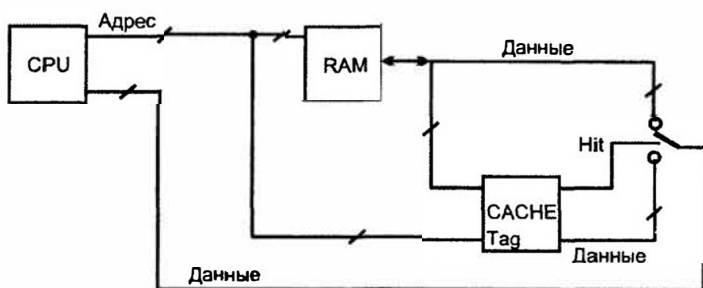


Рис. 4.8. Структура кэшированной памяти

При чтении данных сначала выполняется обращение к кэш-памяти (рис. 4.8). 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.

Эффективность кэширования обуславливается тем, что большинство прикладных программ имеют циклический характер и многократно используют одни и те же данные. Поэтому после первого использования данных из относительно медленной основной памяти повторные обращения 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 является эта единица информации.

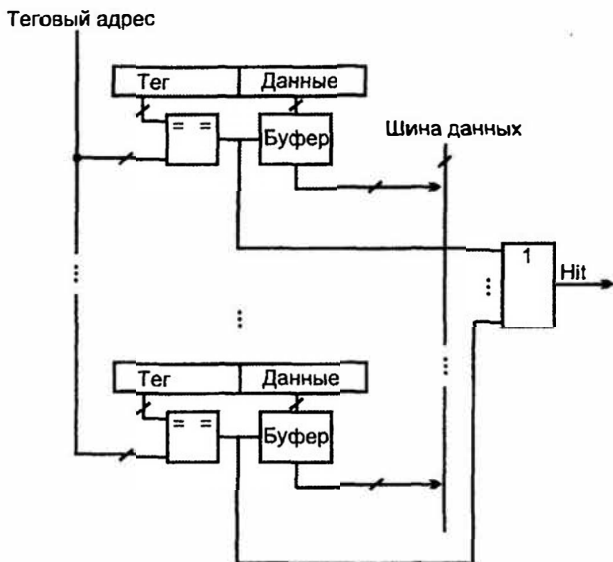


Рис. 4.9. Структура полностью ассоциативной кэш-памяти

В полностью ассоциативной кэш-памяти (FACM, Fully Associated Cache Memory), структура которой показана на рис. 4.9, 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.

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 функцио-

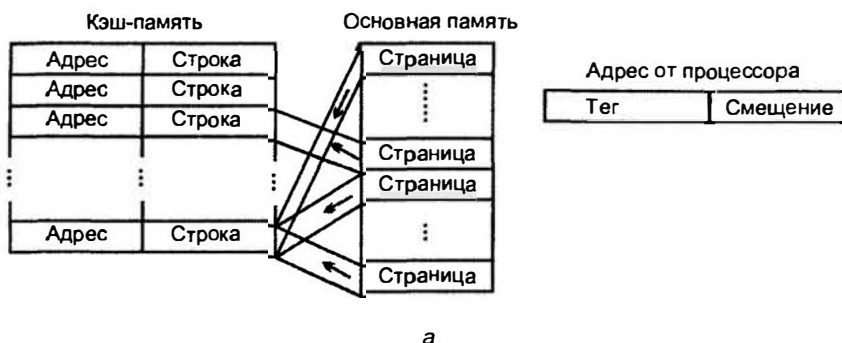
нальную гибкость и бесконфликтность адресов, т. к. любую единицу информации можно загрузить в любую ячейку кэш-памяти.

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). Для конкретного рассмотрения этих структур укажем, что главными параметрами кэш-памяти являются размер строки (Cache Line) и их число (рис. 4.10). Строка представляет собою некоторый набор слов. Ее емкость будем считать соответствующей странице основной памяти.



Рис. 4.10. 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. 4.11, а). 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.



а

Рис. 4.11. Пояснения к организации кэш-памяти с произвольной загрузкой (а)

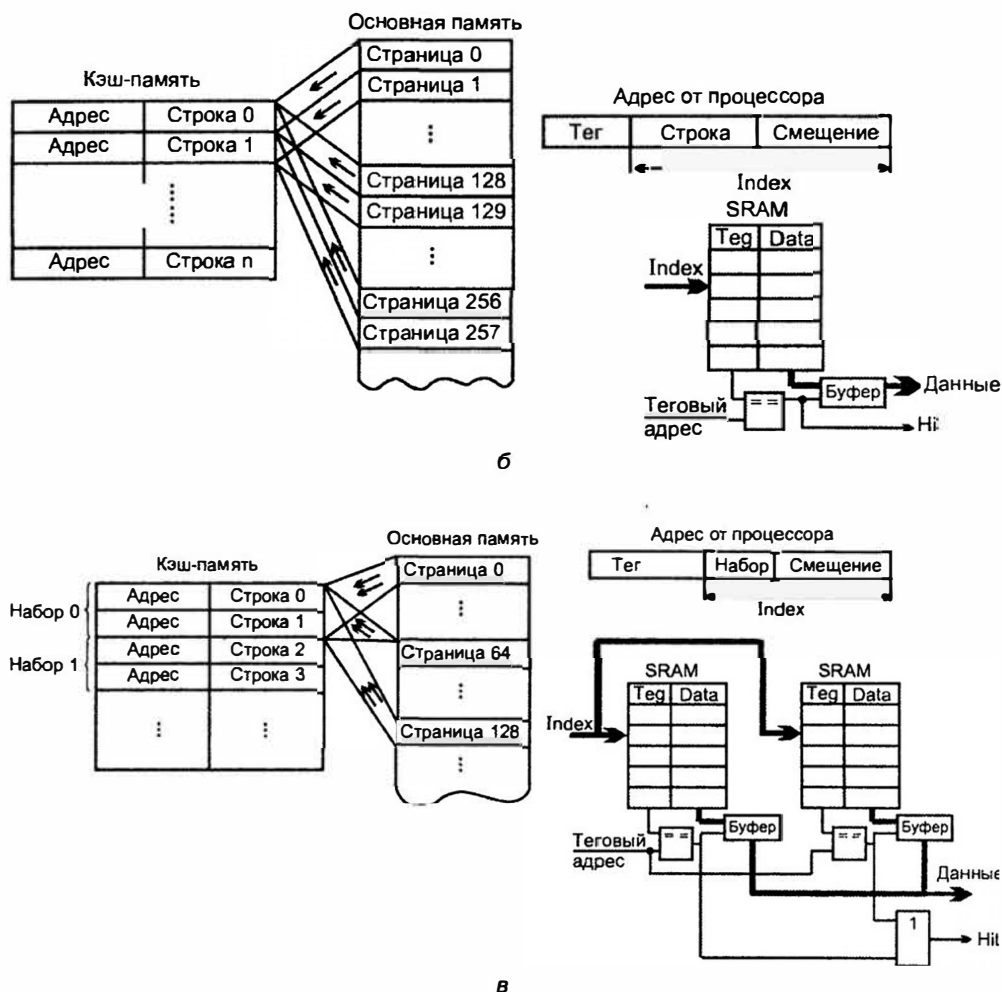


Рис. 4.11. (окончание) Пояснения к организации кэш-памяти с прямым размещением (б) и наборно-ассоциативной (в)

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. 4.11, 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. Ос-

тавшие 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. 4.11, 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 FASM 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. 4.11, 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. 4.11, 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. Например, одна из микросхем фирмы Sugix имеет 4К строк, 15-разрядный теговый адрес и 16-разрядный выход. Для построения кэш-памяти используют чаще всего обычные SRAM 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. Кстати говоря, именно МП Power 3 использован в суперкомпьютере, который сумел обыграть чемпиона мира по шахматам Гарри Каспарова.

§ 4.3. Запоминающие устройства типа ROM(M), PROM, EPROM, EEPROM

Запоминающие устройства типа ROM (память только для чтения) хранят информацию, которая либо вообще не изменяется (в ЗУ типов ROM(M) и PROM), либо изменяется редко и не в оперативном режиме (в ЗУ типов EPROM и EEPROM).

В масочные ЗУ типа ROM(M) информация записывается при изготовлении микросхем на промышленных предприятиях с помощью шаблона (маски) на завершающем этапе технологического процесса.

ЗУ типа PROM программируются после изготовления их предприятием электронной промышленности в лабораториях потребителей без использования сложных технологических процессов. Для этого используются несложные устройства (программаторы).

Программирование постоянной памяти заключается в том или ином размещении элементов связи между горизонтальными и вертикальными линиями матрицы запоминающих элементов.

Запоминающие устройства типа ROM имеют многоразрядную организацию (чаще всего 8-разрядную или 4-разрядную, для некоторых ИС 16-разрядную) и обычно выполняются по структуре 2DM. Простейшие ЗУ могут иметь структуру 2D. Технологии изготовления постоянных ЗУ разнообразны — ТТЛ(Ш), КМОП, n-МОП и др.

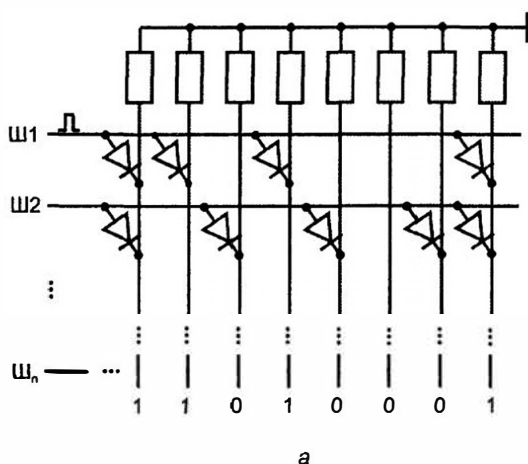
Масочные ЗУ

Элементом связи в масочных ЗУ могут быть диоды, биполярные транзисторы, МОП-транзисторы и т. д.

В матрице диодного ROM(M) (рис. 4.12, а) горизонтальные линии являются линиями выборки слов, а вертикальные — линиями считывания. Считывае-

мое 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 Ш1¹ считывается слово 11010001 (в ячейке номер один хранится это слово). При возбуждении 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 матрицы диоды изготавливаются, в других — нет. При этом, чтобы удешевить производство, при изготовлении ЗУ стремятся варьировать только один шаблон, так чтобы одни элементы связи были законченными и работоспособными, а другие — не законченными и как бы отсутствующими. Для матриц с МОП-транзисторами часто 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. 4.12, б.



а

Рис. 4.12. Матрица диодных запоминающих элементов масочного ЗУ (а)

¹ В литературе, посвященной памяти, "шинами" часто называют отдельные линии (в противоположность литературе по микропроцессорной технике).

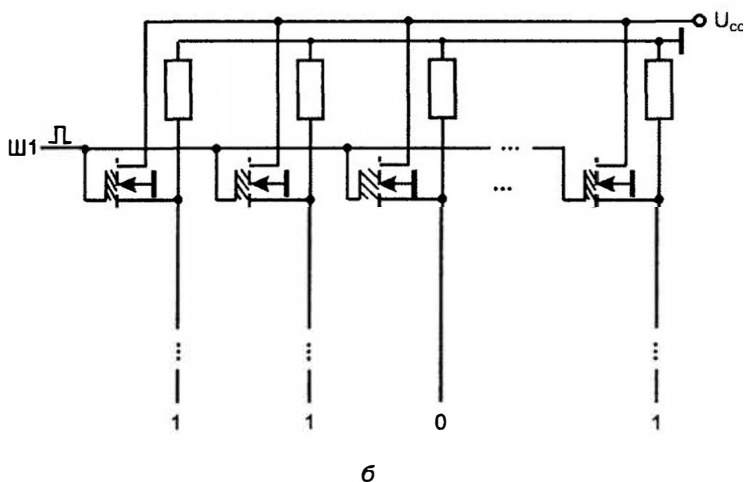


Рис. 4.12. (окончание) Матрица МОП-транзисторных элементов (б)

ЗУ с масочным программированием отличаются компактностью запоминающих элементов и, следовательно, высоким уровнем интеграции. При больших объемах производства масочное программирование предпочтительно, однако при недостаточной тиражности ЗУ затраты на проектирование и изготовление шаблона для технологического программирования ЗУ окажутся чрезмерно высокими. Отсюда видна и область применения масочных ЗУ — хранение стандартной информации, имеющей широкий круг потребителей. В частности, масочные ЗУ имеют в качестве "прошивки"¹ коды букв алфавитов (русского и латинского), таблицы типовых функций (синуса, квадратичной функции и др.), стандартное программное обеспечение и т. п.

ЗУ типа PROM

В ЗУ типа PROM микросхемы программируются устранением или созданием специальных перемычек. В исходной заготовке имеются (или отсутствуют) все перемычки. После программирования остаются или возникают только необходимые.

Устранение части перемычек свойственно ЗУ с плавкими перемычками (типа fuse — предохранитель). При этом в исходном состоянии ЗУ имеет все перемычки, а при программировании часть их ликвидируется путем расплавления импульсами тока достаточно большой амплитуды и длительности.

¹ Термином "прошивка" иногда называют содержимое постоянной памяти. Это название появилось во времена памяти на ферритовых сердечниках, когда информация заносилась в ЗУ путем пропуска провода через определенные сердечники.

В ЗУ с плавкими перемычками эти перемычки включаются в электроды диодов или транзисторов. Перемычки могут быть металлическими (вначале изготавливались из нихрома, позднее из титановольфрамовых и других сплавов) или поликристаллическими (кремниевыми). В исходном состоянии запоминающий элемент хранит логическую единицу, логический ноль нужно записать, расплавляя перемычку.

Создание части перемычек соответствует схемам, которые в исходном состоянии имеют непроводящие перемычки 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, *которые рассмотрены в § 8.1*. В номенклатуре продукции стран СНГ ЗУ с перемычками типа antifuse отсутствуют.

Второй тип запоминающего элемента PROM — два встречно включенных диода. В исходном состоянии сопротивление такой цепочки настолько велико, что практически равноценно разомкнутой цепи, и запоминающий элемент хранит логический ноль. 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. 4.13, а, б в исходном состоянии и после программирования.

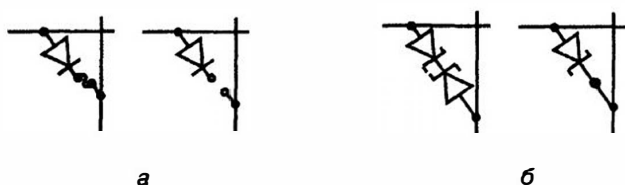


Рис. 4.13. Запоминающие элементы с плавкими перемычками (а) и диодными парами (б)

Матрица запоминающих элементов ЗУ с плавкими перемычками в технике ТТЛ (микросхемы K155PE3) показана на рис. 4.14. 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. 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 ноль 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.

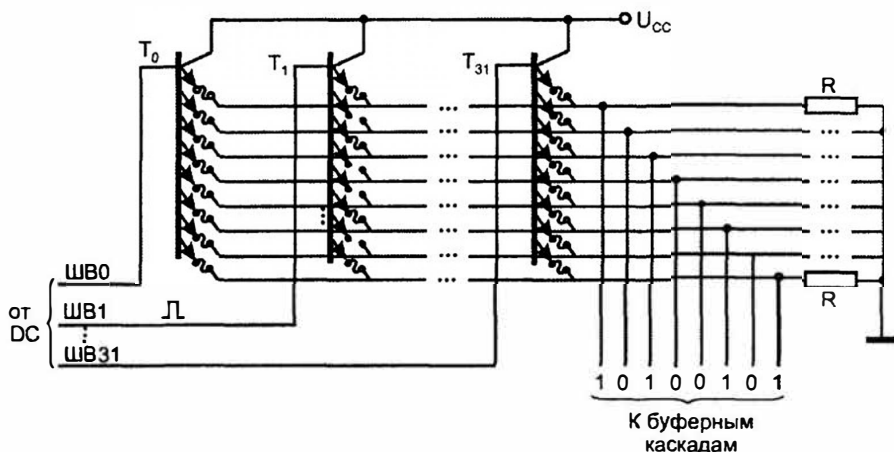


Рис. 4.14. Матрица запоминающих элементов с плавкими перемычками в технике ТТЛ

Программирование ЗУ с плавкими перемычками реализуется простыми аппаратными средствами и может быть доступно схемотехникам даже при отсутствии специального оборудования. На рис. 4.15 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 ОЕ. 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.

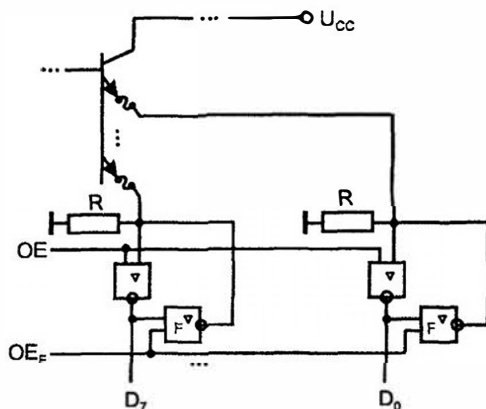


Рис. 4.15. Схема запоминающей ячейки с элементами программирования плавких перемычек

Для прожигания перемычек на них подают токи в десятки миллиампер в виде серии импульсов (для большей надежности прожигания). Не все перемычки удастся пережечь надлежащим образом, коэффициент программируемости для серии K556, например, составляет 0,5...0,7. В ЗУ с плавкими перемычками возможно восстановление проводимости перемычек через некоторое время из-за 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. Невысокая стоимость программируемых пользователем ЗУ объясняется тем, что изготовитель выпускает микросхемы без учета конкретного содержимого ЗУ, т. е. освобожден от проектирования по специализированным заказам и, следовательно, связанных с этим затрат.

Среди отечественных PROM ведущее место занимают микросхемы серии K556, имеющие информационную емкость 1...64 Кбит и время 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.

ЗУ типов EPROM и EEPROM

В репрограммируемых ЗУ типов EPROM и EEPROM (или E²PROM) возможно стирание старой информации и замена ее новой в результате специ-

ального процесса, для проведения которого ЗУ выводится из рабочего режима. Рабочий режим (чтение данных) — процесс, выполняемый с относительно высокой скоростью. Замена же содержимого памяти требует выполнения гораздо более длительных операций.

По способу стирания старой информации различают ЗУ со стиранием ультрафиолетовыми лучами (EPROM или в русской терминологии РПЗУ-УФ, т. е. репрограммируемые ПЗУ с ультрафиолетовым стиранием) и электрическим стиранием (E²PROМ или РПЗУ-ЭС).

Запоминающими элементами современных РПЗУ являются транзисторы типов МНОП и ЛИЗМОП (добавление ЛИЗ к обозначению МОП происходит от слов Лавинная Инжекция Заряда).

МНОП-транзистор отличается от обычного МОП-транзистора 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. 4.16, а). На границе диэлектрических слоев возникают центры захвата заряда. Благодаря туннельному эффекту, носители заряда могут проходить через тонкую пленку окисла толщиной не более 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.

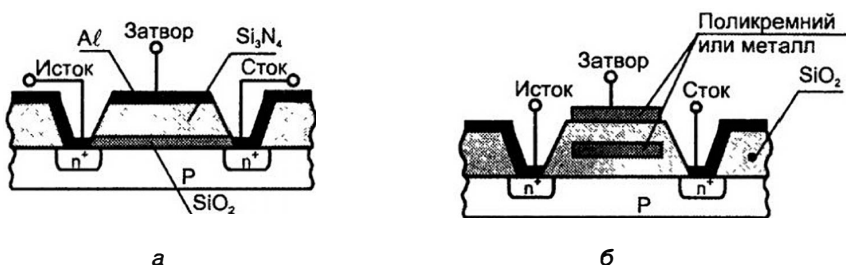


Рис. 4.16. Структуры транзисторов типов МНОП (а) и ЛИЗМОП с двойным затвором (б)

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, лег-

ко открывается рабочим значением напряжения. Так осуществляется хранение бита в МНОП: одно из состояний трактуется как отображение логической единицы, другое — нуля.

При программировании ЗУ используются относительно высокие напряжения, около 20 В. После снятия высоких напряжений туннельное прохождение носителей заряда через диэлектрик прекращается и заданное транзистору пороговое напряжение остается неизменным.

После $10^4 \dots 10^6$ перезаписей МНОП-транзистор перестает устойчиво хранить заряд. РПЗУ на МНОП-транзисторах энергонезависимы и могут хранить информацию месяцами, годами и десятками лет.

Перед новой записью старая информация стирается записью нулей во все запоминающие элементы. Тип ЗУ — РПЗУ-ЭС.

Транзисторы типа ЛИЗМОП всегда имеют так называемый плавающий затвор, который может быть единственным или вторым, дополнительным к обычному (управляющему) затвору. Транзисторы с одним плавающим затвором используются в ЗУ типа РПЗУ-УФ, а транзисторы с двойным затвором пригодны для применения как в РПЗУ-УФ, так и в РПЗУ-ЭС. Рассмотрим более современный тип — ЛИЗМОП-транзистор с двойным затвором (рис. 4.16, б).

Принцип работы ЛИЗМОП с двойным затвором близок к принципу работы МНОП-транзистора — здесь также между управляющим затвором и областью канала помещается область, в которую при программировании можно вводить заряд, влияющий на величину порогового напряжения транзистора. Только область введения заряда представляет собою не границу раздела 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 называется плавающим затвором.

При подаче на управляющий затвор, исток и сток импульса положительного напряжения относительно большой амплитуды 20...25 В в обратном смещенных р-п переходах возникает лавинный пробой, область которого насыщается электронами. Часть электронов, имеющих энергию, достаточную для преодоления потенциального барьера диэлектрической области, проникает в плавающий затвор. Снятие высокого программирующего напряжения восстанавливает обычное состояние областей транзистора и запирает электроны в плавающем затворе, где они могут находиться длительное время (в высококачественных приборах многие годы).

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 может производиться двумя способами — ультрафиолетовым облучением или электрическими сигналами.

В первом случае 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. Операция стирания информации этим способом занимает десятки минут, информация стирается сразу во всем кристалле. В схемах с УФ-стиранием число циклов перепрограммирования существенно ограничено, т. к. под действием ультрафиолетовых лучей свойства материалов постепенно изменяются. Число циклов перезаписи у отечественных ИС равно 10...100.

Электрическое стирание информации осуществляется подачей на управляющие затворы низкого (нулевого) напряжения, а на стоки — высокого напряжения программирования. Электрическое стирание имеет преимущества: можно стирать информацию не со всего кристалла, а выборочно (индивидуально для каждого адреса). Длительность процесса "стирание-запись" значительно меньше, сильно ослабевают 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. В последнее время эти недостатки быстро преодолеваются и ЭС-стирание вытесняет УФ-стирание.

Предшественниками двухзатворных ЛИЗМОП-транзисторов были однозатворные, имевшие только плавающий затвор. Эти транзисторы изготавливались обычно с р-каналом, поэтому введение электронов в плавающий затвор приводило к созданию в транзисторе проводящего канала, а удаление заряда — к исчезновению такого канала. При использовании таких транзисторов запоминающие элементы состоят из двух последовательно включенных транзисторов: ключевого МОП-транзистора обычного типа для выборки адресованного элемента и ЛИЗМОП-транзистора, состояние которого определяет хранимый бит. Стирание информации производится ультрафиолетовыми лучами.

Подключение двухзатворных ЛИЗМОП-транзисторов к линиям выборки строк и линиям чтения в матрицах ЗУ показано на рис. 4.17. Запись логического нуля осуществляется путем заряда плавающего затвора инъекцией "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 РПЗУ-УФ (в маркировке они имеют буквы РФ) наиболее известна серия К573 с широким набором типономиналов, а среди РПЗУ-ЭС (в маркировке имеют буквы РР) имеются серии КР558 (на основе п-МНОП), К1609, К1624, К1626 на ЛИЗМОП с двумя затворами.

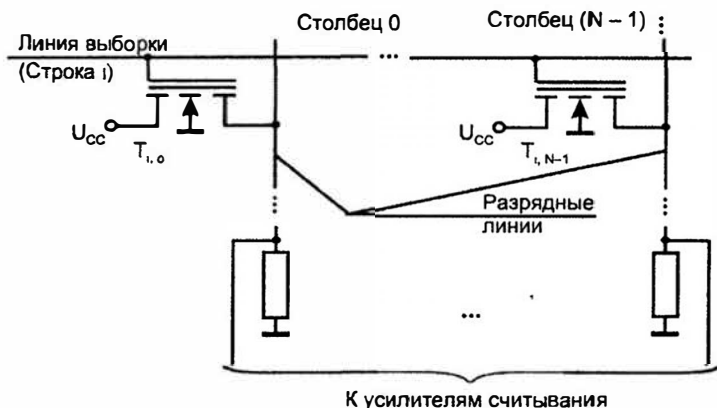


Рис. 4.17. Схема подключения ЛИЗМОП-транзисторов с двойным затвором к линиям выборки и считывания в РПЗУ

Отечественные ROM характеризуются в настоящее время следующими параметрами: масочные ИС имеют информационную емкость до 1 Мбита при временах доступа около 200 нс, микросхемы с плавкими перемычками соответственно 64 Кбита и 80 нс, РПЗУ-УФ 1 Мбит и 350 нс, РПЗУ-ЭС 64 Кбита и 250 нс.

На уровне мировой техники имеются 3У типа РПЗУ-УФ с информационной емкостью до 8 Мбит при временах доступа 45 нс (фирма Atmel), 3У типа РПЗУ-ЭС с информационной емкостью до 256 Кбит при временах доступа 90 нс и допустимом числе циклов перезаписи 10^5 с временем сохранения данных более 10 лет. Это 3У использует один источник питания 5 В и потребляет ток 2 мА в активном режиме и 100 мкА при отсутствии обращений. Возможна байтовая или страничная запись за время 3 мс (фирма SGS-Thomson).

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. 4.18 показан обычный вариант по-

строения 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.

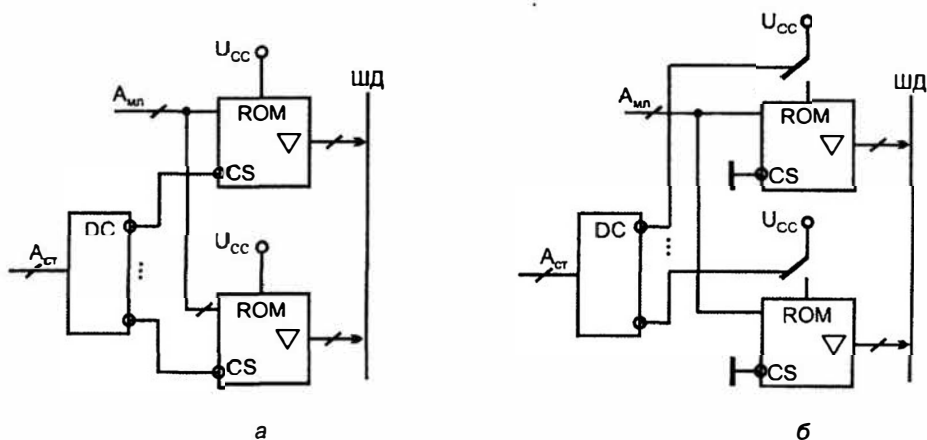


Рис. 4.18. Модули постоянной памяти с обычным (а) и импульсным (б) питанием

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, потеря времени не возникает.

§ 4.4. Флэш-память

Флэш-память (Flash-Memory) по типу запоминающих элементов и основным принципам работы подобна памяти типа $E^2\text{PROM}$, однако ряд архитектурных и структурных особенностей позволяют выделить ее в отдельный класс. Разработка Флэш-памяти считается кульминацией десятилетнего развития схемотехники памяти с электрическим стиранием информации.

В схемах Флэш-памяти не предусмотрено стирание отдельных слов, стирание информации осуществляется либо для всей памяти одновременно, либо для достаточно больших блоков. Понятно, что это позволяет 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.

Термин Flash по одной из версий связан с характерной особенностью этого вида памяти — возможностью одновременного стирания всего ее объема. Согласно этой версии ещё 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 многих применений это неудобно. Поэтому наряду со схемами с одновременным стиранием всего содержимого имеются схемы с блочной структурой, в которых весь массив памяти делится на блоки, стираемые независимо друг от друга. Объем таких блоков сильно разнится: от 256 байт до 128 Кбайт.

Число циклов репрограммирования для Флэш-памяти хотя и велико, но ограничено, т. е. ячейки при перезаписи "изнашиваются". Чтобы увеличить долговечность памяти, в ее работе используются специальные алгоритмы, способствующие "разравниванию" числа перезаписей по всем блокам микросхемы.

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) и замена памяти на 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. Стирание в этих схемах может быть как одновременным для всей памяти, так и блочным. Среди устройств с блочным стиранием выделяют схемы со специализированными блоками (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 для замены жестких магнитных дисков (*Flash-File Memory*) содержат более развитые средства перезаписи информации и имеют идентичные блоки (симметричные блочные структуры).

Одним из элементов структуры Флэш-памяти является накопитель (матрица запоминающих элементов). В схемотехнике накопителей развиваются два направления: на основе ячеек типа ИЛИ-НЕ (NOR) и на основе ячеек типа И-НЕ (NAND).

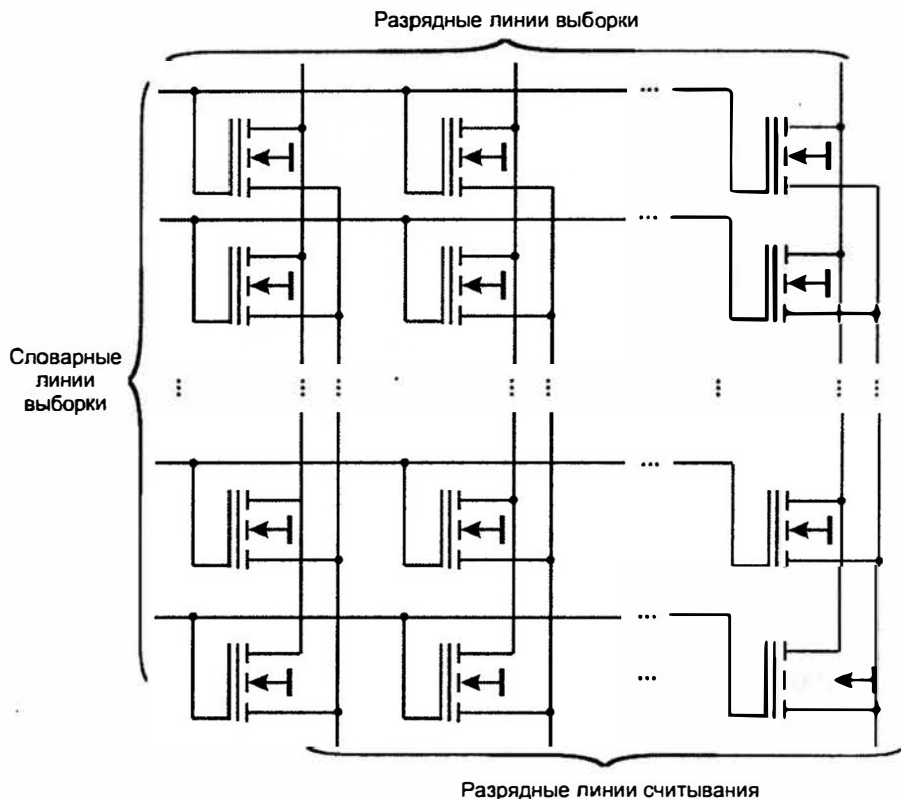


Рис. 4.19. Структура матрицы накопителя Флэш-памяти на основе ячеек ИЛИ-НЕ

Накопители на основе ячеек ИЛИ-НЕ (с параллельным включением ЛИЗ-МОП-транзисторов с двойным затвором) обеспечивают быстрый доступ к словам при произвольной выборке. Они приемлемы для разных применений, но наиболее бесспорным считается их применение в памяти для хранения редко обновляемых данных. При этом возникает полезная преемственность с применявшимися ранее ROM и EPROM, сохраняются типичные сигналы управления, обеспечивающие чтение с произвольной выборкой. Структура матрицы накопителя показана на рис. 4.19. 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.

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 практически используются только в схемах замены магнитных дисков. В схемах на этих ячейках сам накопитель компактнее, но увеличивается количество логических элементов обрामления накопителя.

Для улучшения технико-экономических характеристик в схемах Флэш-памяти применяются различные средства и приемы:

1. 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.
2. Внутренняя очередь команд, управляющих работой Флэш-памяти, которая позволяет организовать конвейеризацию выполняемых операций и ускорить процессы чтения и записи.
3. Программирование длины хранимых в ЗУ слов для согласования с различными портами ввода/вывода.
4. 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.
5. 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.
6. 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.
7. Различные меры защиты от случайного или несанкционированного доступа.

Флэш-память с адресным доступом, ориентированная на хранение не слишком часто изменяемой информации, может иметь одновременное стирание

всей информации (архитектура Bulk Erase) или блочное стирание (архитектура Boot Block Flash-Memory).

Имея преимественность с 3У типов E²PROM и EPROM, разработанными ранее, схемы Флэш-памяти предпочтительнее E²PROM 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 к тому же происходит гораздо быстрее.

Память типа Bulk Erase

Память типа Bulk Erase фирмы Intel, наиболее известной среди разработчиков Флэш-памяти, имеет время записи байта около 10 мкс, допускает до 10⁵ циклов стирания, напряжение программирования для нее составляет 12 В ± 5%, ток активного режима около 10 мА, в режиме покоя около 50 мкА. Время доступа при чтении равно приблизительно 100 нс, время стирания и время программирования всего кристалла составляет 0,6...4 с для кристаллов емкостью 256 Кбит...2 Мбит.

В отличие от традиционного управления схемами памяти с помощью адресных и управляющих сигналов. Флэш-память имеет *дополнительное управление словами-командами*, записываемыми процессором в специальный регистр, функционирующий только при высоком уровне напряжения на выводе микросхемы, обозначаемом U_{pp} (напряжении программирования). При отсутствии такого уровня U_{pp} схема работает только как память для чтения под управлением традиционных сигналов, задающих операции чтения, снижения мощности, управления третьим состоянием и выдачи идентификатора.

На рис. 4.20 показана структура Флэш-памяти типа Bulk Erase (схемы 28F010, 28F020 фирмы Intel и др.).

Входы А являются адресными, причем в течение цикла записи адреса фиксируются в регистре-защелке по сигналу строба STB. Ввод/вывод данных (линии DQ) осуществляется через буферы с третьим состоянием. В течение цикла записи данные фиксируются в регистре-защелке.

Сигналы \overline{CE} , \overline{OE} и \overline{WE} L-активны. Сигнал \overline{CE} активизирует управляющую логику, буферы ввода/вывода данных, дешифраторы адреса DC_y, DC_x и усилители чтения. При высоком уровне сигнала \overline{CE} (схема не выбрана) буферы входят в третье состояние, а потребление мощности снижается до уровня покоя (Standby).

Сигнал \overline{OE} низким уровнем разрешает вывод данных через буферы в течение циклов чтения (естественно, только при низком уровне сигнала \overline{CE}).

Сигнал \overline{WE} разрешает запись в регистр команд и матрицу запоминающих ячеек и своими фронтами загружает регистры-защелки (отрицательным — регистр-защелку адреса, положительным — данных).

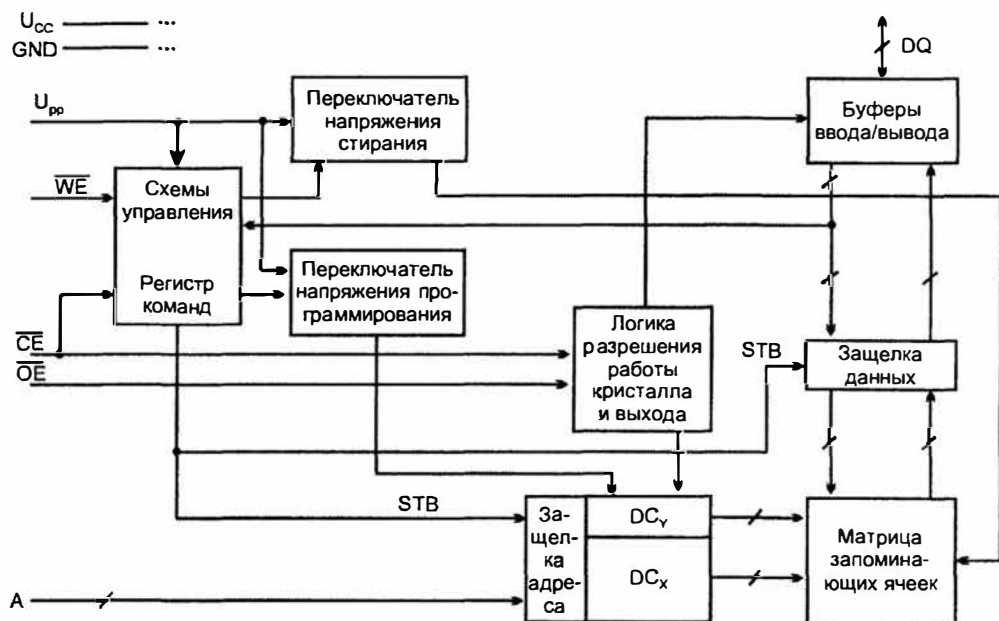


Рис. 4.20. Структура Флэш-памяти со стиранием данных одновременно со всего кристалла (типа Bulk Erase)

Схемы управления и содержимое регистра команд определяют состояние переключателей уровней напряжения U_{PP} , используемых в разных режимах работы (при стирании, программировании или чтении).

Для получения выходных данных при чтении требуется активное состояние сигналов \overline{CE} и \overline{OE} . При этом низкий уровень U_{PP} делает возможным чтение только данных памяти, а высокий позволяет считывать также коды идентификаторов и информацию для проверки операций стирания/программирования. Операции с идентификатором выводят код фирмы-изготовителя и микросхемы. Эти сведения позволяют согласовать алгоритмы стирания и программирования схемы и программирующего оборудования, что производится автоматически.

Коды идентификаторов находятся в двух ячейках памяти и могут считываться с помощью определенной комбинации сигналов или регистра команд (чтением после подачи в регистр команды 90H).

При выполнении операций записи коды адресов и данных фиксируются во внутренних регистрах-защелках. При высоком уровне U_{PP} выполняются те же операции и дополнительно разрешается стирание и программирование памяти. Все действия, связанные с изменением содержимого памяти, производятся с использованием регистра команд. Регистр команд не занимает какой-либо позиции в адресном пространстве и загружается обычным циклом записи от процессора при низком уровне U_{PP} . Его содержимое играет роль входной информации для внутреннего автомата управления схемами стирания и программирования памяти. Используются 7 команд, две из которых задают операции чтения (данных и кодов идентификатора), две другие относятся к операции стира-

ния (подготовка стирания/стирание и проверка стирания), две команды относятся к операции программирования (подготовка программирования/программирование и проверка программирования) и одна команда задает операцию сброса микросхемы.

При снижении уровня U_{pp} регистр команд сбрасывается, разрешая микросхеме только операции чтения.

По команде стирания стираются все байты матрицы параллельно, после чего все они должны быть проверены. Для этого байты адресуются и активизируются подачей специального напряжения. Чтение из ячейки кода OFFH показывает, что все биты байта стерты. Если считывается иной код, выполняется повторная операция стирания. Затем проверка возобновляется с адреса последнего проверенного байта. Процесс проверки продолжается до достижения последнего адреса.

Программирование памяти ведется байт за байтом (последовательно или при произвольном доступе). Цикл чтения от процессора выводит данные байта, которые сравниваются с заданными. Равенство байтов свидетельствует об успешном программировании. После этого процесс программирования переходит к следующему байту.

Команда сброса является средством надежного устранения действия команд стирания/программирования. После каждой из этих команд в регистр команд можно записать код операции сброса, что устранил возможность каких-либо действий, связанных с указанными командами. Содержимое памяти не сможет измениться. Для дальнейшего приведения схемы в желаемое состояние в регистр команд нужно записать соответствующую команду.

При переходе сигнала СЕ к высокому уровню вводится режим пониженной мощности. Если это происходит при стирании, программировании или проверках данных, то активный ток сохраняется до завершения указанных операций.

Флэш-память с несимметричной блочной структурой

Схемам типа Boot Block Flash Memory (Boot-блок Флэш-память, сокращенно ББФП) присуще блочное стирание данных и несимметричная блочная архитектура. Блоки специализированы и имеют разные размеры. Среди них имеется так называемый Boot-блок (ББ), содержимое которого аппаратно защищено от случайного стирания. В ББ 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 БП (блоки параметров) и ГБ (главные блоки), не снабженные аппаратными средствами защиты от непредусмотренной записи. Блоки БП хранят относительно часто меняемые параметры системы (коды идентификаторов, диагностические программы и т. п.). Блоки ГБ хранят основные управляющие программы и т. п.

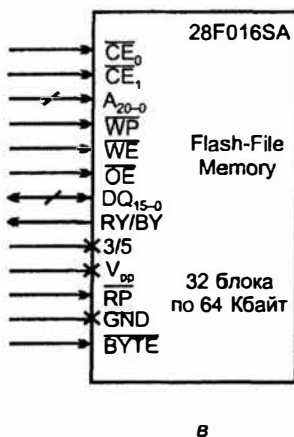
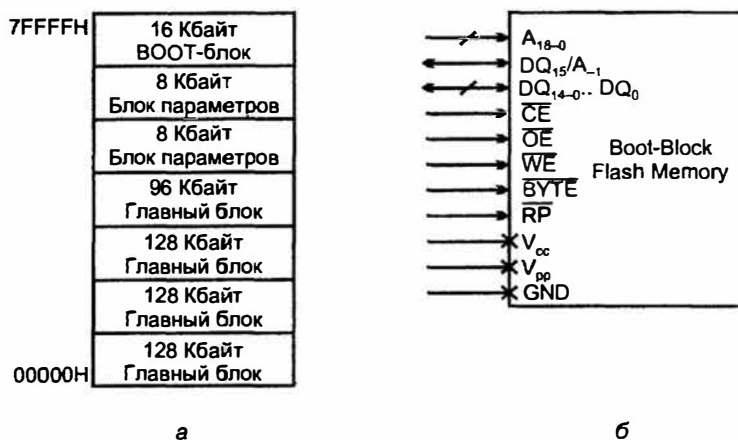


Рис. 4.21. Распределение адресного пространства и внешняя организация Флэш-памяти с несимметричной блочной структурой (а, б) и внешняя организация файловой Флэш-памяти (в)

Микросхемы ББФП предназначены для работы с разными микропроцессорами и для соответствия им имеют два варианта расположения ББ в адресном пространстве: вверху и внизу, что отображается в маркировке ИС буквами Т (Top) или В (Bottom). На рис. 4.21, а для примера приведена карта памяти (распределение адресного пространства) для ИС емкостью 4 Мбит с верхним расположением ББ.

В настоящее время выпускаются ББФП с емкостями 1...16 Мбит, в последующих поколениях ожидаются ИС с информационными емкостями до 256 Мбит.

По своему функционированию ББФП близки к памяти типа Bulk Erase, в обоих типах ИС операции стирания/программирования ведутся под управлением внутреннего автомата, входной информацией для которого служат

команды, вводимые от процессора. В схемах ББФП эту роль играет так называемый командный интерфейс пользователя CUI (Command User Interface).

Внешняя организация типичной ББФП показана на рис. 4.21, б на примере ИС с информационной емкостью 4 Мбита.

Адреса задаются 19-разрядным кодом A_{18-0} , т. е. в памяти хранится до 512 Кслов. Сигнал **BYTE** задает 8-разрядную или 16-разрядную организацию памяти. При байтовой организации байты передаются по линиям DQ_{7-0} , а линия DQ_{15} играет роль самого младшего разряда адреса A_{-1} , определяющего, какой байт данной ячейки передается (старший или младший). При словарной организации выходы DQ_{15-0} являются линиями ввода-вывода данных.

Напряжение на выводе **RP** (Reset/Power Down) может иметь три уровня: $12\text{ В} \pm 5\%$, уровень логической единицы **H** и низкий уровень **L**. При напряжении $12\text{ В} \pm 5\%$ ББ открыт и в нем могут выполняться операции стирания и программирования. При напряжении ниже 6,5 В ББ заперт.

Имея ряд режимов экономии мощности, схемы ББФП, в частности, реализуют режим **APS** (Automatic Power Saving), благодаря которому после завершения цикла чтения схема автоматически входит в статический режим с потреблением тока около 1 мА, в котором находится до начала следующего цикла чтения.

Когда схема не выбрана (при высоком уровне сигнала на выводе **CE** и выводе **RP**, т. е. $CE = RP = H$) потребление мощности снижается до уровня покоя (десятки мкА). При $RP = L$ не только запрещается запись, но и вводится режим глубокого снижения мощности, в котором ток потребления снижается до долей мкА.

Активному режиму соответствует комбинация сигналов $\overline{CE} = L$ и $\overline{RP} = H$. Сигналы **OE** и **WE** имеют обычное назначение. Микросхемы Boot-блок Флэш-памяти могут работать с разными напряжениями питания и программирования (технология **Smart Voltage**), имеют времена доступа при чтении 60...70 нс, токи активных режимов 15...25 мА и крайне малые токи в режиме глубокого понижения мощности (около 0,2 мкА).

Файловая Флэш-память

Важное место в иерархии ЗУ занимает файловая Флэш-память (ФФП). В течение многих лет хранение больших объемов данных возлагалось в микроЭВМ на хорошо отработанные и сравнительно недорогие внешние ЗУ на магнитных, а впоследствии и оптических дисках. Во многих компьютерах система памяти организована как сочетание жесткого магнитного диска (винчестера) с динамическим полупроводниковым ОЗУ.

Имея значительные достоинства, дисковые ЗУ как электромеханические устройства не свободны от ряда недостатков: чувствительности к ударам и

вибрациям, загрязнению, ограниченного быстродействия и значительного потребления 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 ориентирована на замену твердых дисков, которая 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 для замены дисковой памяти в портативных компьютерах — один из важнейших факторов, способствующих развитию этого направления. При этом традиционное сочетание "жесткий диск — динамическое ОЗУ" может заменяться сочетанием "Флэш-память — статическое ОЗУ". Команды программы, хранимые в ФФП, читаются в этом случае непосредственно процессором, результаты тоже записываются прямо в ФФП, а операции с интенсивными вычислениями, требующие быстрее 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, отражаемых в операционной системе MS-DOS. Разработаны программные средства, которые обеспечивают обмен между Флэш-блоками, подобно тому как операционная система MS-DOS обеспечивает обмен между 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.

Микросхемы ФФП фирмы Intel имеют информационную емкость 4...32 Мбит при временах доступа 70...150 нс, напряжения питания 5; 3,3 или даже 2,7 В. Они имеют байтовую или управляемую разрядность (8 или 16), напряжение программирования у них также, как правило, многовариантно (3,3; 5; 12 В).

Внешняя организация ФФП показана на рис. 4.21, в, на примере микросхемы с информационной емкостью 16 Мбит (ИС типа 28F016SA фирмы Intel).

Накопитель схемы с общей информационной емкостью 16 Мбит разбит на 32 блока по 64 Кбайт.

Поясним смысл некоторых выводов и сигналов. Шина адреса: линии A_{20-16} выбирают один из блоков, линии A_{15-1} выбирают слово в пределах одного блока (блок с емкостью 64 Кбайта содержит 32 Кслов), линия A_0 — бит выборки байта, определяющий старший и младший байты при байтовой организации памяти и отключаемый при ее словарной организации. От процессора поступает начальный адрес блока данных, который запоминается в очереди адресов. Текущий адрес ячейки памяти для обмена формируется адресным счетчиком.

В шине данных DQ_{15-0} линии DQ_{7-0} предназначены для ввода и вывода младшего байта данных, передачи команды в командный интерфейс пользователя CUI в цикле записи и вывода данных из буфера, регистров идентификатора или состояния в соответствующих режимах чтения. Линии DQ_{15-8} предназначены для передачи старшего байта при словарной организации памяти. По ним выводят данные накопителя, буфера или идентификатора в соответствующем режиме чтения, но эти линии не используются для чтения из регистров состояния. Если кристалл не выбран или запрещен вывод, линии шины данных переходят в третье состояние.

Линии \overline{CE}_0 и CE_1 — входы разрешения кристалла, при высоком уровне любого из них 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 WE управляет доступом к командному интерфейсу пользователя CUI, страничным буферам, регистрам очереди данных и защелкам очереди адресов.

Сигнал 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 схемы составляет 400 нс. 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 (кстати говоря, сигнал вырабатывается каскадом с открытым стоком, требующим подключения внешней цепочки $U_{cc} - R$ для формирования высокого уровня) 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 имеет бит запрещения записи (Lock-bit). Низкий уровень WP разрешает защиту,

т. е. запись или стирание в блоке могут выполняться только при $\text{Lock-bit} = 0$. При высоком уровне $\overline{\text{WP}}$ в блоках могут выполняться операции записи и стирания независимо от состояния блокирующих битов.

Сигнал $\overline{\text{BYTE}}$ низким уровнем вводит схему в байтовый режим, высоким — в словарный и выключает буфер линии A_0 .

Напряжение программирования U_{pp} и вывод напряжения питания (это может быть 3,3 или 5 В — вход обозначен дробью 3/5) поступают в схему через переключатель напряжения, который находится внутри схемы.

Для примера приведем параметры ФФП фирмы Intel/28F032SA (1997 г.):

- ❑ организация 2М×16 или 4М×8 (по выбору потребителя), напряжение питания 3,3 или 5 В (по выбору потребителя), напряжение программирования 12 В, до 10^6 циклов стирания на блок, 64 независимо запираемых блока по 64 Кбайт или 64 блока по 32 Кслов;
- ❑ корпус типа TSOP размерами 1,2×14×20 мм с 56 выводами;
- ❑ технология с топологической нормой 0,6 мкм;
- ❑ время доступа при чтении 70 или 150 нс при питании от 5 В и 3 В соответственно;
- ❑ время записи слова/байта не более 9 мкс;
- ❑ время записи блока не более 2,1 с для байтового режима и не более 1 с для словарного режима;
- ❑ время стирания блока не более 10 с и стирания кристалла не более 25,6 с.

Память типа StrataFlash

В 1997 г. компания Intel представила новый вид Флэш-памяти, названный СтратаФлэш (StrataFlash), в которой впервые в одном элементе памяти хранятся два бита, а не один. 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.

До изобретения памяти СтратаФлэш для увеличения емкости ЗУ шли путем уменьшения размеров схемных элементов и других усовершенствований технологических процессов литографии. СтратаФлэш 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 вводимых в плавающий затвор зарядов. Во второй половине 90-х гг. появились коммерческие образцы памяти СтратаФлэш. При этом от емкости 32 Мбита перешли к емкости 64 Мбита без заметных изменений площади кристалла.

Запоминающие элементы программируются введением в плавающий затвор одного из 4-х количеств заряда, каждое из которых соответствует паре двоичных цифр 11, 10, 01, 00. В зависимости от заряда, запоминающий транзистор имеет одно из четырех пороговых напряжений. При считывании информации к затвору транзистора прикладывают напряжение считывания. Ток запоминающего транзистора зависит от порогового напряжения. Определяя ток, можно выявить состояние плавающего затвора.

На рис. 4.22 показаны распределение пороговых напряжений в четырехуровневом запоминающем элементе (а) и схема чтения состояния запоминающего транзистора (б).

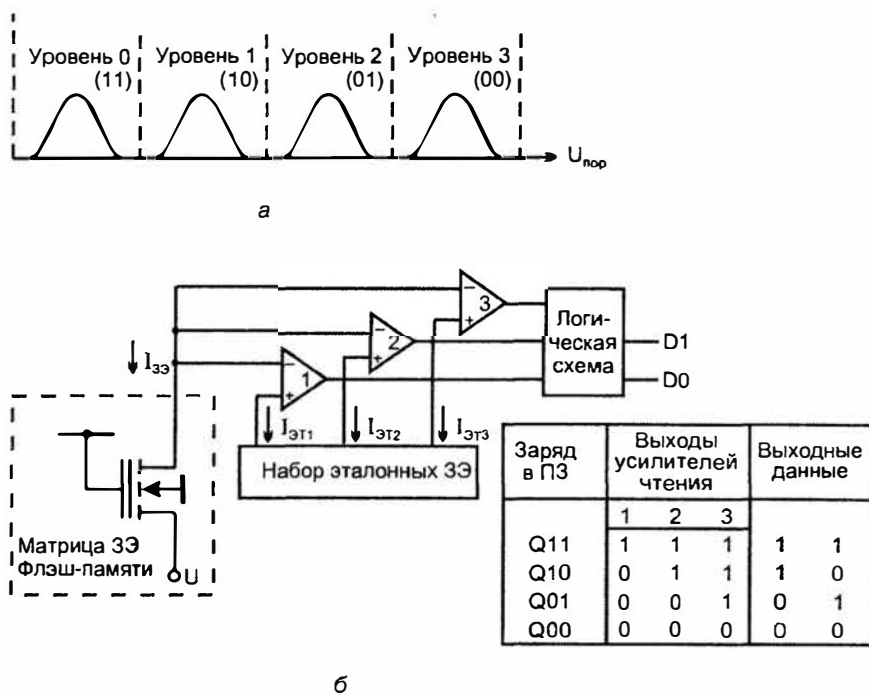


Рис. 4.22. Графики распределения пороговых напряжений в четырехуровневом запоминающем элементе (а) и схема чтения данных из этого элемента (б)

§ 4.5. Использование программируемых ЗУ для решения задач обработки информации

В предыдущих параграфах запоминающие устройства рассматривались с точки зрения основной для них задачи хранения информации. Однако програм-

мируемая память есть также универсальное средство решения самых разных задач обработки информации. Применимость этого средства 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 одно-разрядный 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 — 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. 4.23.

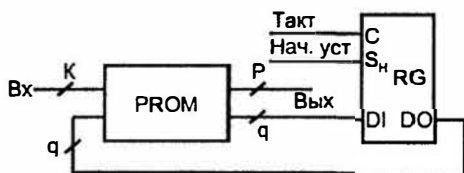


Рис. 4.23. Структура автомата, реализованного на основе микросхем памяти

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.

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. 4.24, а). Если разрядность кода, представляющего функцию, равна n , то, очевидно, емкость памяти в битах будет равна $n2^m$.

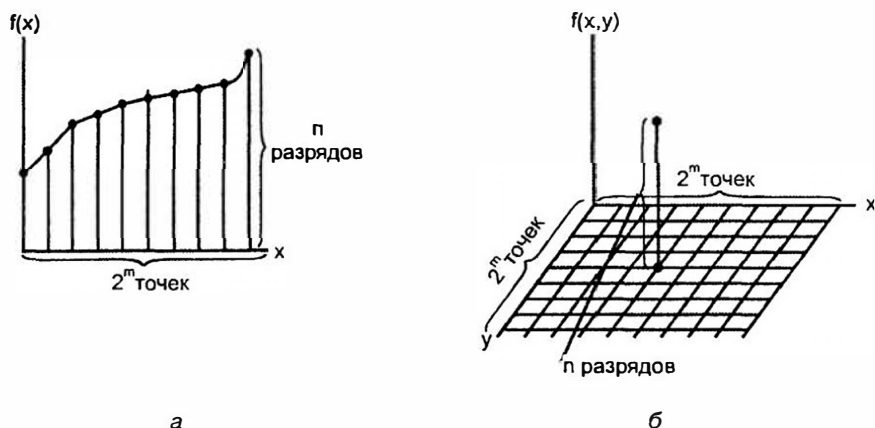


Рис. 4.24. К определению емкости памяти при воспроизведении табличным методом числовых функций одного (а) и двух (б) аргументов

С ростом числа аргументов объем памяти для запоминания таблиц функций быстро растет. Для функции двух аргументов разрядностей m_1 число точек, в которых задана функция, определится как произведение чисел точек по каждой из координат и составит 2^{2m} (рис. 4.24, 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.

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:

$$x = x_i + \Delta x, \quad f(x) = f(x_i) + \Delta f(x_i) \Delta x h^{-1},$$

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).

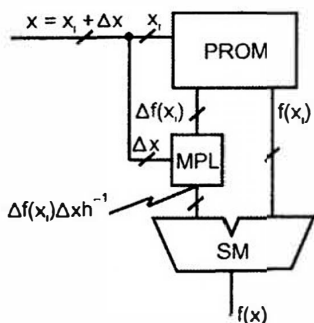


Рис. 4.25. Структура преобразователя с кусочно-линейной аппроксимацией функций

Согласно приведенным формулам структура функционального преобразователя с кусочно-линейной аппроксимацией имеет вид, приведенный на рис. 4.25

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.

§ 4.6. Статические запоминающие устройства

Область применения относительно дорогостоящих статических ОЗУ в системах обработки информации определяется их высоким быстродействием. В частности, они широко используются в кэш-памяти, которая при сравнительно малой емкости должна иметь максимальное быстродействие.

Статические ОЗУ (SRAM), как правило, имеют структуру 2DM, часть их при небольшой информационной емкости строится по структуре 2D.

Запоминающими элементами статических ОЗУ служат триггеры с цепями установки и сброса. В связи с этим статические ОЗУ называют также триггерными. Триггеры можно реализовать по любой схмотехнологии (ТТЛ(Ш), И²Л, ЭСЛ, n-МОП, КМОП, AsGa и др.), соответственно которой существуют разнообразные схемы ЗУ. Различие в параметрах этих ЗУ отражает специфику той или иной схмотехнологии. В последнее время наиболее интенсивно развиваются статические ЗУ, выполненные по схмотехнологии КМОП, которая по мере уменьшения топологических норм технологического процесса приобретает высокое быстродействие при сохранении своих традиционных преимуществ.

Среди отечественных серий микросхем хорошо развитыми являются серии К537 технологии КМОП и К132 технологии n-МОП.

При подаче нуля на выход \bar{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 столбцовыми шинами соединяются все триггеры строки, но только одна пара шин связывается с выходными цепями считывания или входной цепью записи в соответствии с адресом столбца.

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 двухполюсники, показанные на рис. 4.26, б. В первом случае это п-МОП транзистор со встроенным каналом и нулевым напряжением затвора, т. е. обычный элемент нагрузки в схемах с п-каналом.

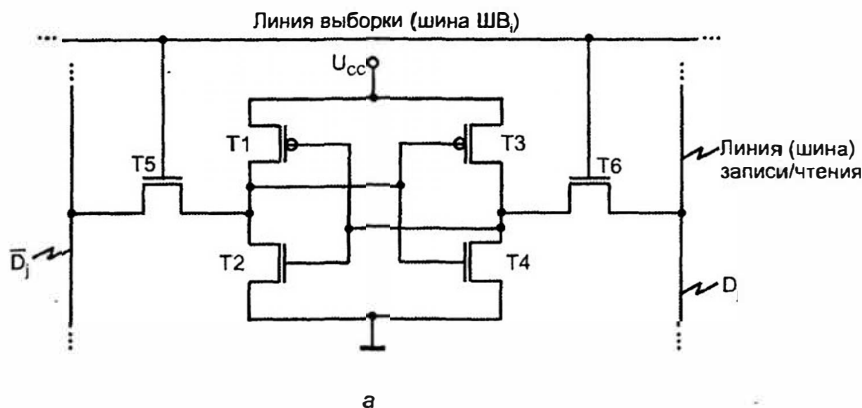
Стремление к режиму микротоков привело к схеме с нагрузочным поликремниевым резистором (второй случай, нагрузка типа рис. 4.26, в). Высокоомные нагрузочные резисторы изготавливаются из поликристаллического кремния и пространственно расположены над областью транзисторов, что придает схеме также и высокую компактность. Режим микротоков нужен для кристаллов высокого уровня интеграции, но создает и ряд трудностей, в первую очередь низкую скорость переключения триггера (микротоки не в состоянии быстро перезаряжать паразитные емкости схемы) и маломощность выходных сигналов. Первый недостаток преодолевается тем, что триггер переключается под воздействием мощных сигналов записи информации через ключевые транзисторы, а не за счет только внутренних токов цепей обратных связей. Вторая особенность требует применения высокочувствительных усилителей считывания. Это объясняет использование так называемых усилителей-регенераторов в статических ЗУ (ранее они были характерны только для динамических).

Запоминающие элементы статических ОЗУ, выполненных по КМОП технологии, показаны на рис. 4.27, а в обозначениях США. Эти элементы построены так же, как и элементы на п-МОП транзисторах, и не требуют дополнительных пояснений.

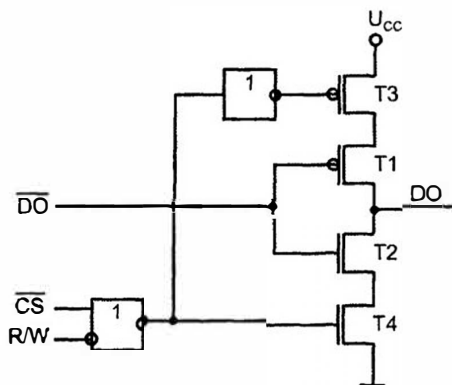
Выходной каскад с третьим состоянием

На рис. 4.27, б показан выходной каскад с третьим состоянием, используемый в КМОП ЗУ. Низкий уровень сигнала \overline{CS} и высокий уровень сигнала R/W , означающие разрешение операции чтения, создают на выходе элемента ИЛИ-НЕ высокий уровень логической единицы, открывающий транзи-

сторы T3 и T4 и, тем самым, позволяющий нормально работать инвертору на транзисторах T1 и T2, через который данные передаются на выход. При всех иных комбинациях сигналов \overline{CS} и R/W выход элемента ИЛИ-НЕ имеет низкий уровень логического нуля, при котором транзисторы T3 и T4 заперты и выход DO находится в состоянии "отключено".



а



б

Рис. 4.27. Схемы триггерного запоминающего элемента (а) и выходного каскада (б) в схемотехнике КМОП

Внешняя организация и временные диаграммы статических ЗУ

В номенклатуре статических ЗУ представлены ИС с одноразрядной и словарной организацией. Внешняя организация статического ЗУ емкостью 64 Кбита ($8K \times 8$) показана на рис. 4.28. Состав и функциональное назначение сигналов адреса A_{12-0} , 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).

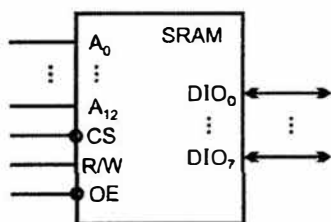


Рис. 4.28. Пример внешней организации статического ЗУ

Таблица 4.1

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.

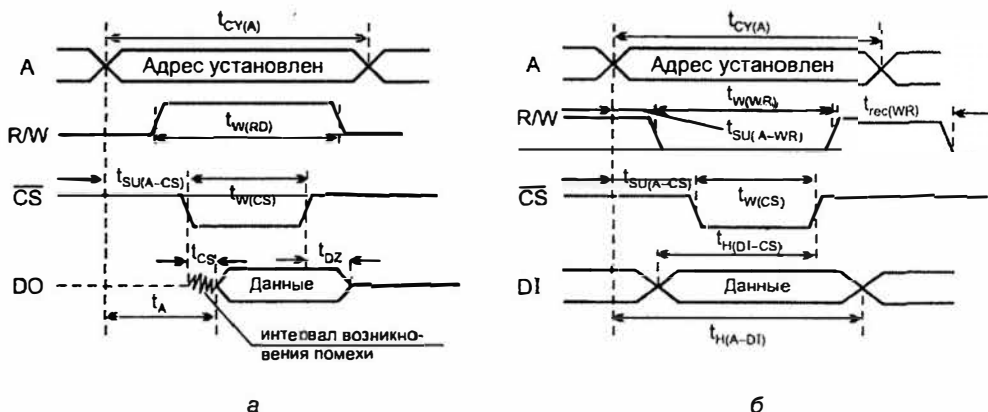


Рис. 4.29. Временные диаграммы процессов чтения (а) и записи (б) в статическом ЗУ

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} .

Статические ЗУ подразделяются на асинхронные и тактируемые. В тактируемых ЗУ к определенным сигналам (как правило, к сигналу \overline{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 асинхронных ЗУ такие требования отсутствуют 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 статическом ЗУ показан на рис. 2.29, а, б. На них показаны времена выборки относительно адреса t_A и выбора t_{CS} , длительности импульсов t_W различных сигналов и цикла адреса $t_{SY(A)}$, задержка 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 $C_{L\max}$.

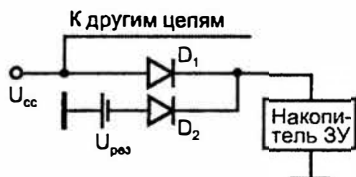
Искусственная энергонезависимость статических ЗУ

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.

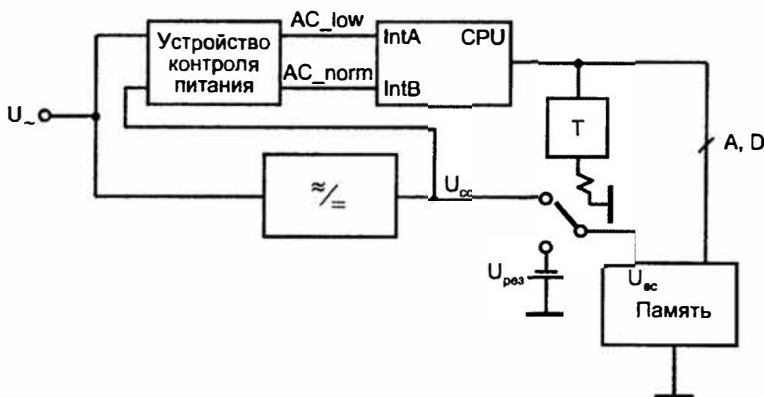
Для подключения к накопителю ЗУ резервного источника питания разработчики памяти рекомендуют схему, приведенную на рис. 4.30, а. В этой схеме напряжение резервного источника несколько ниже напряжения основного источника U_{CC} . В рабочем режиме накопитель питается от напряжения U_{CC} , при этом диод $D1$ 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 разработке микропроцессорных систем вариант (рис. 4.30, а) недостаточно надежен в связи со следующим обстоятельством. Напряжение пита-

ния 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 относительно медленно снижается. На начальном этапе этого процесса система продолжает работать, но в ее работе возможны ошибки. Желательно быстрее отреагировать на аварию питания. Это достигается с помощью схем (рис. 4.30, б).



а



б

Рис. 4.30. Схемы подключения резервных источников питания к накопителям ЗУ (а, б)

Здесь нарушение нормальной работы источника питания обнаруживается контролем напряжения переменного тока (AC — Alternate Current). Нарушение можно выявить за один-два периода переменного напряжения, пока постоянное напряжение U_{cc} еще не изменилось. Признак нарушения AC_{low} служит запросом прерывания для процессора CPU. Получив запрос, процессор выполняет подпрограмму обслуживания прерывания А (Interrupt A), в ходе которого передает содержимое своих регистров в стек накопителя (выполняет так называемое контекстное переключение) и заканчивает подпрограмму установкой триггера Т, что воздействует на обмотку реле, управляющего ключом. В результате память подключается к резервному источнику.

При восстановлении нормального питания признак АС_погт вызывает программу обслуживания прерывания В, в ходе которой из стека возвращаются в процессор данные для регистров процессора и сбрасывается триггер, что ведет к подключению памяти к основному источнику питания.

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. 4.31 показана для примера ячейка двухпортового ЗУ с организацией $4K \times 1$ и временем доступа 4 нс, выполненная 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.

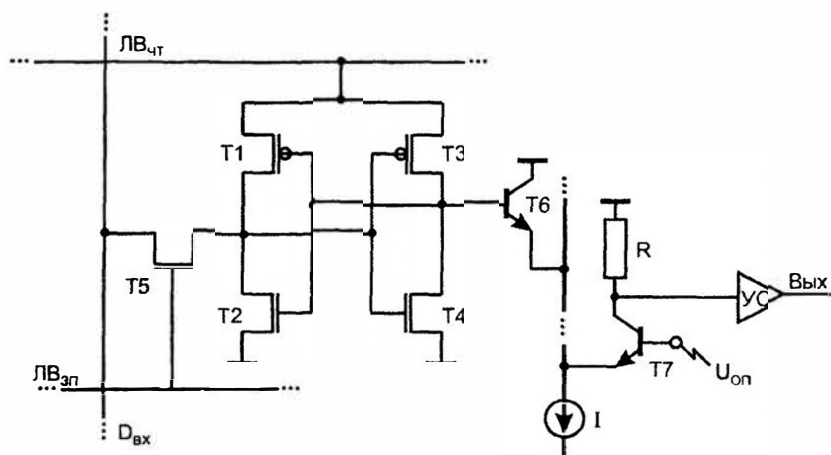


Рис. 4.31. Схема ячейки статического ЗУ в схемотехнике БикМОП

Ячейка имеет две линии выборки — для чтения (ЛВ_{чт}) и для записи (ЛВ_{зн}). 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 $ЛВ_{\text{чт}}$. В отсутствие выборки для чтения этот потенциал невысок и любые переключения триггера не могут настолько повысить потенциал базы T_6 , чтобы он открылся. Запись данных производится сигналом $D_{\text{вх}}$ при выборке ячейки по линии $ЛВ_{\text{эл}}$. Транзистор T_5 изготавливается как низкоомный, что позволяет ему диктовать состояние триггера.

Для чтения напряжение на линии $ЛВ_{\text{чт}}$ повышают на 0,55 В. Если триггер хранит единицу, то T_3 открыт, а T_4 заперт. Так как при этом перепад напряжения на $ЛВ_{\text{чт}}$ передается на базу T_6 , он открывается, и ток I переключается из опорного транзистора T_7 в транзистор T_6 . Напряжение на коллекторе T_7 повышается, что и служит входным сигналом чтения единицы для последующих каскадов усилителя чтения, обозначенных как $УС$. Если триггер хранит логический ноль, то T_3 заперт и T_4 открыт. Ясно, что в этом случае перепад напряжения на линии $ЛВ_{\text{чт}}$ никак не повлияет на потенциал базы T_6 , переключения тока I не возникнет и перепада выходного напряжения схемы не будет.

§ 4.7. 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 6 транзисторов, что позволяет разместить на кристалле намного больше ЗЭ (в 4–5 раз) и обеспечивает динамическим ЗУ максимальную емкость. В то же время конденсатор неизбежно теряет со временем свой заряд, и хранение данных требует их периодической регенерации (через несколько миллисекунд).

Запоминающие элементы

Известны конденсаторные ЗЭ разной сложности. В последнее время практически всегда применяют одностранзисторные ЗЭ — лидеры компактности, размеры которых настолько малы, что на их работу стали влиять даже α -частицы, излучаемые элементами корпуса ИС.

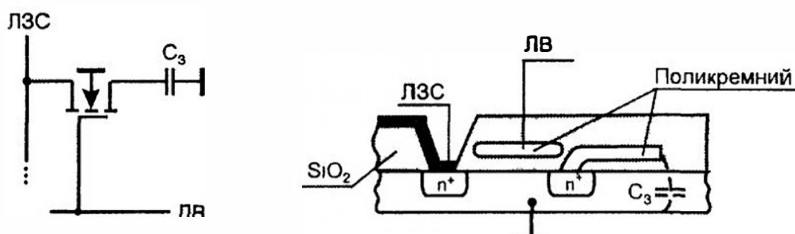


Рис. 4.32. Схема и конструкция запоминающего элемента динамического ЗУ

Электрическая схема и конструкция однотранзисторного ЗЭ показаны на рис. 4.32. 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 .

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. Фрагмент ЗУ (рис. 4.33) показывает ЗЭ, усилитель считывания УС а также ключи К1 и К0 соответственно записи единицы и нуля. К линии записи-считывания (ЛЗС) подключено столько ЗЭ, сколько строк имеется в запоминающей матрице. Особое значение имеет емкость ЛЗС $C_{\text{л}}$, в силу большой протяженности линии и большого числа подключенных к ней транзисторов многократно превышающая емкость ЗЭ.

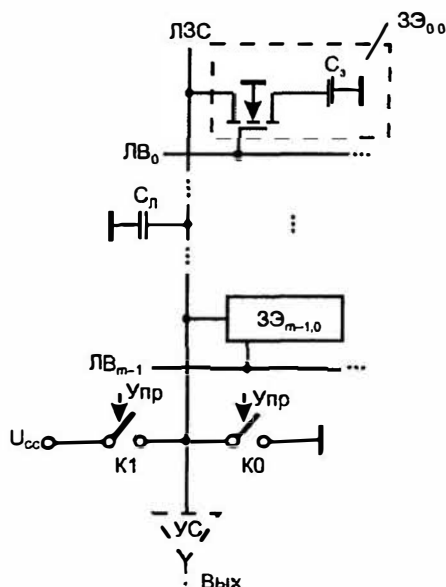


Рис. 4.33. Фрагмент схемы динамического ЗУ

Перед считыванием производится 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_L заряжается до уровня $U_{cc}/2$. Будем считать, что хранение единицы соответствует заряженной емкости C_3 , а хранение нуля — разряженной.

При считывании нуля к ЛЗС подключается емкость C_3 , имевшая нулевой заряд. Часть заряда емкости C_L перетекает в емкость C_3 , 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_3 составляло вначале величину U_{cc} и превышало напряжение на ЛЗС. При подключении C_3 к ЛЗС часть заряда стекает с запоминающей емкости в C_L и напряжение на ЛЗС увеличивается на ΔU . Графики сигналов при считывании нуля и единицы показаны на рис. 4.34.

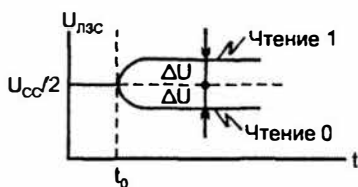


Рис. 4.34. Временные диаграммы сигналов при считывании данных в динамических ЗУ

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

$$Q = C_L U_{cc}/2.$$

После выборки ЗЭ этот же заряд имеет суммарная емкость $C_L + C_3$ и можно записать следующее соотношение:

$$Q = (C_L + C_3) (U_{cc}/2 - \Delta U).$$

Приравняв выражения для одного и того же значения заряда Q , получим соотношение

$$C_L U_{cc}/2 = (C_L + C_3) (U_{cc}/2 - \Delta U),$$

из которого следует выражение

$$\Delta U = U_{cc} C_3 / [2(C_3 + C_L)] \approx U_{cc} C_3 / 2C_L.$$

В силу неравенства $C_3 \ll C_L$ сигнал ΔU оказывается слабым.

Кроме того, считывание является разрушающим — подключение запоминающей емкости к ЛЗС изменяет ее заряд.

Мерами преодоления отмеченных недостатков служат способы увеличения емкости C_3 (без увеличения площади ЗЭ), уменьшения емкости ЛЗС и применение усилителей-регенераторов для считывания данных.

В направлении увеличения C_3 можно указать разработку фирмой Сименс нового диэлектрика (диоксида титана TiO_2), имеющего диэлектрическую постоянную в 20 раз большую, чем SiO_2 . Это позволяет при той же емкости сократить площадь ЗЭ почти в 20 раз или увеличить C_3 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. 4.35, а). Очевидно, что такой прием вдвое уменьшает емкость линий, к которым подключаются запоминающие емкости, т. е. вдвое увеличивает сигнал ΔU .

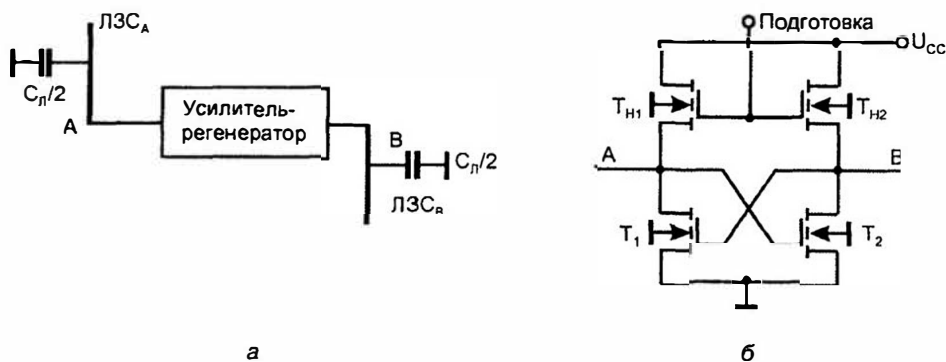


Рис. 4.35. Схема включения усилителя-регенератора в разрыв линии записи-считывания динамического ЗУ (а) и вариант схемной реализации усилителя-регенератора (б)

Усилители-регенераторы

Усилители-регенераторы строятся на основе триггерных схем. Один из возможных вариантов (рис. 4.35, 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 зави-

симости от того, считывается единица или ноль. Неравенство напряжений в точках А и В вносит несимметрию 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 А и В являются одновременно и входами и выходами усилителя-регенератора, после своего срабатывания он восстанавливает на емкости C_3 полное значение считанного сигнала. Тем самым автоматически осуществляется регенерация данных в ЗУ. Состояние триггера определяет также сигналы, выводимые во внешние цепи в качестве считанной информации.

Мультиплексирование шины адреса

Особенностью динамических ЗУ является мультиплексирование шины адреса. Адрес делится на два полуадреса, один из которых представляет собою адрес строки, а другой — адрес столбца матрицы ЗУ. Полуадреса подаются на одни и те же выводы корпуса ИС поочередно. Подача адреса строки сопровождается соответствующим стробом RAS (Row Address Strobe), а адреса столбца — стробом 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 RAS и CAS в некоторых режимах и схемах используются различно (например, в режимах 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 $16M \times 1$ имеет 24-разрядный адрес, а мультиплексирование сократит число адресных линий на 12.

Внешняя организация и временные диаграммы

На рис. 4.36 показаны внешняя организация и временные диаграммы динамического ОЗУ. Циклы обращения к ЗУ начинаются сигналом \overline{RAS} и закрываются 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 RAS и адреса строк. Области безразличных значений сигналов на рисунке заштрихованы.

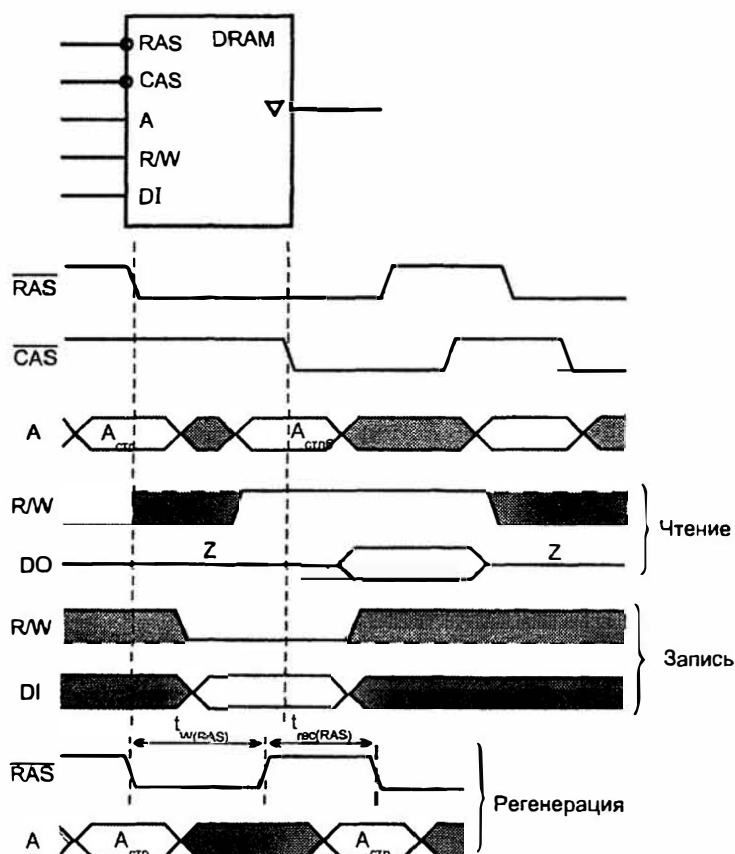


Рис. 4.36. Пример внешней организации и временных диаграмм динамического ЗУ

Схема динамического ЗУ

В схеме динамического ЗУ (рис. 4.37) 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 $\Phi 1$ и $\Phi 2$. Тактирующий сигнал $\Phi 1$ разрешает загрузку регистра RgX и работу дешифратора ДШХ, одна из выходных линий которого возбуждается и выбирает все ЗЭ строки, адрес которой содержится в регистре RgX .

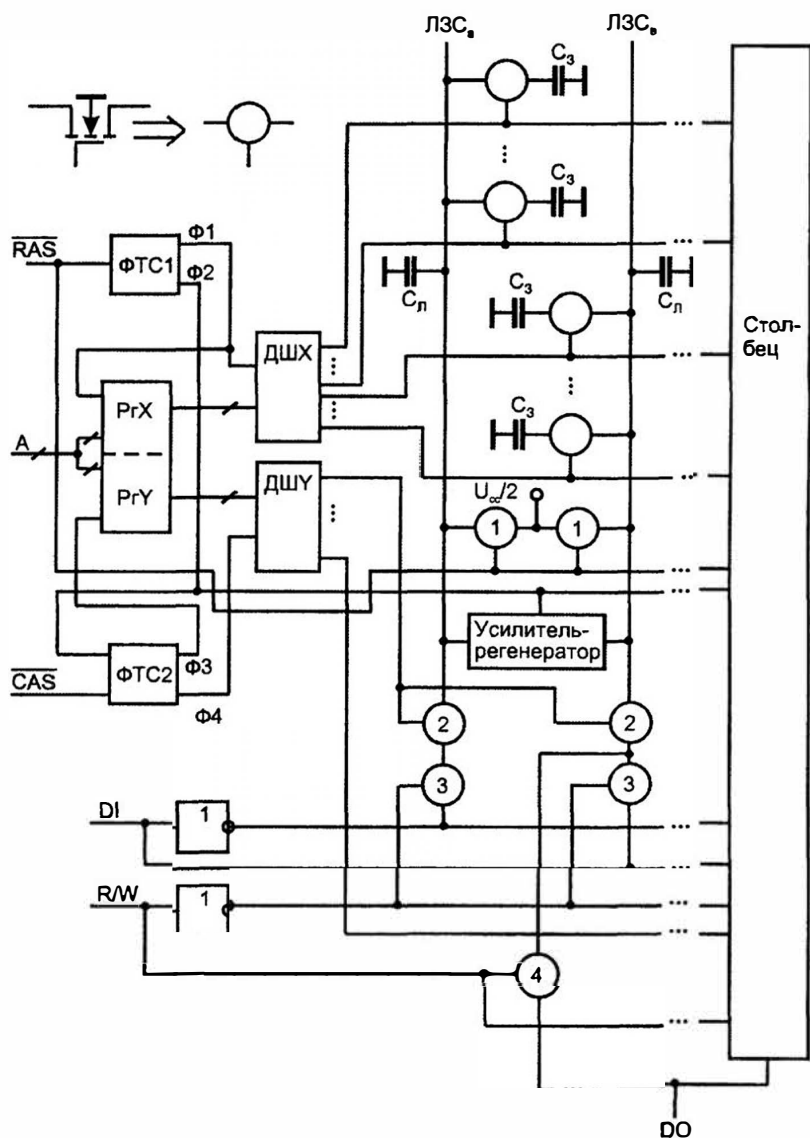


Рис. 4.37. Схема динамического ЗУ

В разрыв между секциями ЛЗС_а и ЛЗС_в включен усилитель-регенератор, для которого подключение ЗЭ, хранящего единицу или ноль, 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 CAS, разрешающего формирователю ФТС2 формирование второй пары тактирующих сигналов Ф3 и Ф4. Сигнал Ф3 загружает в РГУ адрес столбца, а Ф4 активизирует дешифратор ДШУ, 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).

Для операции регенерации, целиком проходящей внутри ЗУ, связь с внешними выводами не требуется, поэтому для нее достаточно подачи только сигнала \overline{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.

§ 4.8. Динамические запоминающие устройства повышенного быстродействия

Современные микропроцессоры характеризуются высоким быстродействием. Это требует и увеличения скорости работы ОЗУ, обменивающихся информацией с процессорами. Особенно остро эта задача стоит перед разработчиками динамических ОЗУ, которые благодаря максимальной информационной емкости и низкой стоимости занимают ведущее место в составе основной памяти компьютеров.

В последнее время предложен ряд вариантов динамических ОЗУ повышенного быстродействия. *Методы, использованные в этих ОЗУ, основаны на предположении о кучности адресов при обращениях к ОЗУ.* Это отвечает тенденции, проявляющейся при выполнении самых разных программ и состоящей в том, что адреса последующих обращений к ОЗУ вероятнее всего расположены рядом с адресом текущего обращения.

Вариант FPM

Вариант FPM (Fast Page Mode, быстрый страничный режим доступа) эффективен, если после обращения к некоторому ЗЭ следующее обращение будет к ЗЭ в той же строке. Сравним такую ситуацию с более общей.

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 обращении к строке (странице), во всех ЗЭ строки проходят процессы, соответствующие двум первым фазам полного цикла обмена (по стробу RAS), и эти элементы готовы к выполнению очередных фаз. При обращении к данным в пределах одной страницы адрес строки остается неизменным, изменяются только адреса столбцов в сопровождении сигнала строба CAS. Изменяет состояние фактически только группа ключей 3 и 4 (см. рис. 4.37). Пока не изменился номер страницы, в циклах обмена исключены некоторые этапы, что сокращает длительность циклов.

Временные диаграммы для режима FPM представлены на рис. 4.38. Видно, что время доступа к данным при неизменности адреса строки RA и изменениях только адреса столбца 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-... .

Режим FPM — начало линии развития методов повышения быстродействия динамических ЗУ. По быстродействию его возможности уже намного превышены более поздними разработками, тем не менее метод FPM находит свою область применения, и соответствующие ЗУ до сих пор занимают достаточно большой сектор рынка.

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 обеспечивают времена обращения к ЗУ 30...40 нс, что допускает их работу с процессорными шинами на тактовой частоте до 33 МГц.

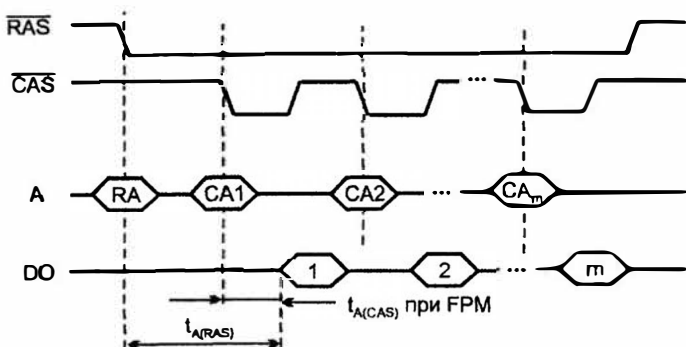


Рис. 4.38. Временные диаграммы режима FPM динамических ОЗУ

Структуры типа EDORAM

Структуры типа EDORAM (Extended Data Out RAM, т. е. ОЗУ с расширенным 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 $\overline{\text{CAS}}$. 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 $\overline{\text{CAS}}$, но длительность его может быть сокращена в сравнении с режимом FPM. Это увеличивает быстродействие ЗУ. В случае применения памяти типа EDORAM характерная пропорциональность времен обращения будет следующей: 5-2-2-....

Разработанные EDORAM допускают работу на частотах до 50 МГц. Такие ЗУ получили 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.

Структуры типа BEDORAM

В структуре типа BEDORAM (Burst EDORAM, т. е. с пакетным расширенным доступом) содержится дополнительно 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 счетчика. Характерная пропорциональность времен первого и последующих обращений 5-1-1-1 (имеется в виду часто применяемый вариант с длиной пакета, равной 4). Память типа BEDORAM не получила широкого распространения из-за появления сильного конкурента — синхронных DRAM (SDRAM), в которых не только достигается пропорциональность времен обращений 5-1-1-1, но и сами времена существенно сокращаются.

Структура типа MDRAM

В структурах MDRAM (Multibank DRAM, многобанковые ОЗУ) память делится на части (банки). Обращение к банкам поочередное, чем исключается ожидание перезаряда 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, тем меньше будет повторных последовательных обращений в один и тот же банк.

Так как процессор чаще всего считывает данные по последовательным адресам, то эффект ускорения работы ЗУ достигается уже при делении памяти всего на два блока, а именно на один с нечетными адресами, другой — с четными. Банки ЗУ типа MDRAM могут строиться на обычных DRAM без каких-либо схемных изменений.

Структуры типа SDRAM

Хотя переход от базовой структуры DRAM к архитектурам FPM и EDORAM повысил быстродействие памяти, этого оказалось недостаточно для современных компьютеров и графических систем. Память типа SDRAM (Synchronous DRAM) заняла сейчас важное место в качестве быстродействующей памяти с высокой пропускной способностью.

В SDRAM синхросигналы памяти тесно увязаны с тактовой частотой системы, в них используется конвейеризация тракта продвижения информации, может применяться многобанковая структура памяти и др.

Синхронные DRAM были предложены в 1994 г. в работе [58] как двухбанковые системы с трехступенчатым конвейером, имевшие пропускную способность 250 Мбайт/с. Эти ЗУ работали на частоте 125 МГц при $U_{cc} = 3,3$ В и топологической норме 0,5 мкм. Причем площадь кристалла (113,7 мм²) практически не отличалась от площади кристаллов обычных DRAM той же емкости.

До более подробного ознакомления с памятью типа SDRAM рассмотрим общий вопрос о конвейеризации трактов обработки информации. Сущность конвейеризации заключается в разбиении трактов обработки информации на ступени. На рис. 4.39 показан тракт обработки данных, содержащий входной и выходной регистры и логическую схему между ними. Исходя из тезиса о возможности подачи новых входных данных только после окончания обработки старых, получим минимальный период тактовых импульсов для этой схемы:

$$T_{\min} = t_{pr} + t_{кц} + t_{su},$$

где t_{pr} — задержка входного регистра на пути "такт-выход"; $t_{кц}$ — задержка сигнала в комбинационной цепи (логической схеме); t_{su} — время предустановки выходного регистра.

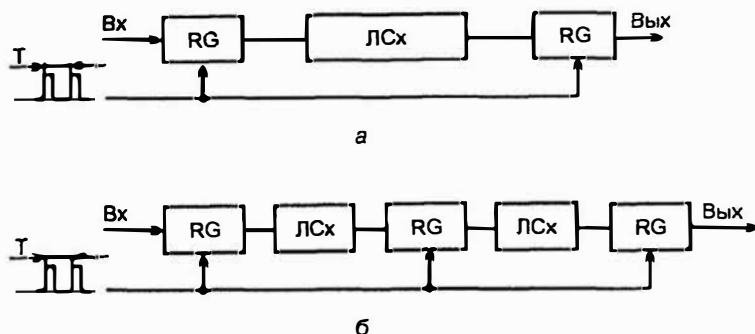


Рис. 4.39. Исходный (а) и конвейеризованный (б) тракты обработки информации

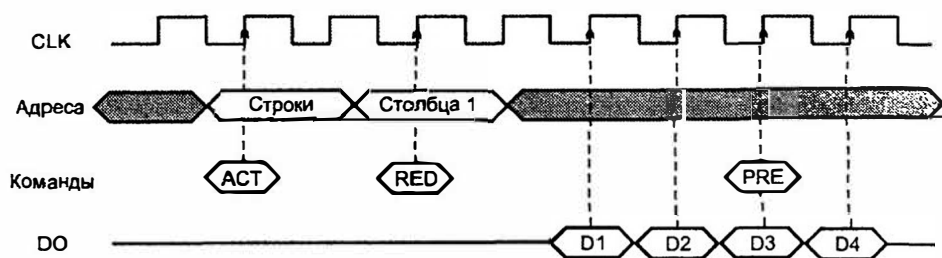
Уменьшения T_{\min} , т. е. повышения частоты тактовых импульсов, можно добиться снижением $t_{\text{лс}}$ путем расщепления логической схемы на ступени, разделенные регистрами (рис. 4.39, б). Если логическая схема расщепляется по глубине ровно пополам, то новое значение минимального периода тактовых импульсов определится тем же соотношением, что и для схемы, показанной на рис. 4.39, а, однако численное значение задержки логической схемы нужно будет уменьшить вдвое

Применение конвейера увеличивает поток информации от входа к выходу за единицу времени, хотя, в то же время, единица информации проходит от входа к выходу за большее время, чем в схеме без конвейеризации.

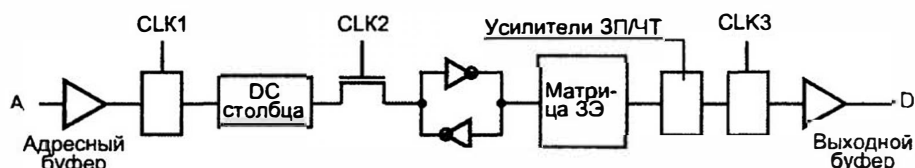
В микросхемах SDRAM внешние управляющие сигналы фиксируются положительными фронтами тактовых импульсов и используются для генерации команд, управляющих процессами в ЗУ. Команда ACT (Active) связана с выбором строки по соответствующему адресу. Команда RED (Read) определяет адрес первого столбца для чтения данных. Команда PRE (Precharge) связана с этапом предзаряда шин.

Первое слово после формирования адреса 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. 4.40, а). Чтобы ускорить темп появления слов, в пакете организуется трехступенчатый конвейер (рис. 4.40, 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. 4.40, в).

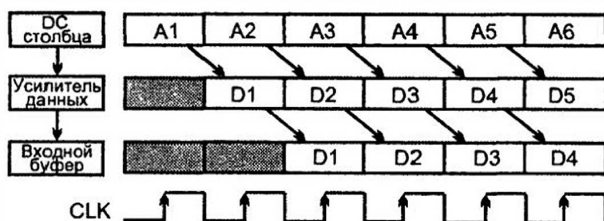
В микросхемах SDRAM предусматривают возможность регулировки запаздывания первого доступа с целью приспособления памяти к частотным требованиям системы и длины пакета, в котором слова читаются или записываются в каждом такте после всего одной команды.



а



б



в

Рис. 4.40. Временные диаграммы (а), трехступенчатый конвейер (б) и временные соотношения обработки информации (в) для синхронных динамических ОЗУ

К достоинствам SDRAM относится 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 (зашелкиваются) фронтами тактовых импульсов, жестко задающими моменты их появления и исчезновения. В SDRAM легко реализуются и многобанковые системы памяти на одном кристалле.

Структуры типа RDRAM

Микросхемы названы по имени фирмы разработчика — Rambus (RDRAM — Rambus DRAM). Они представляют собою байт-последовательную память с очень высоким темпом передачи байтов. Основными новшествами архитектурного плана являются *синхронизация обоими фронтами тактовых импульсов*

и специальный новый интерфейс *Rambus Channel*. Синхронизация принципиально сходна с применяемой в SDRAM.

В первой разработке при частоте тактовых импульсов 250 МГц получен темп передачи байтов 500 МГц (2 нс/байт). В дальнейшем частота еще повысилась в 1,5...3 раза.

Интерфейс *Rambus Channel* имеет всего 13 сигнальных линий, что значительно меньше, чем у традиционных микросхем памяти. В интерфейсе нет специализированных адресных линий. Вместо обычной адресации по интерфейсу посылаются пакеты, включающие в себя команды 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 запаздывание составляло 128 нс. Поэтому при чтении отдельных слов RDRAM совершенно неэффективна. Средняя частота передачи байтов зависит 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.

RDRAM идеально 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.

Структура DRDRAM

Это близкий родственник RDRAM, называемый Direct RDRAM (DRDRAM). В этой разновидности архитектуры RDRAM преодолен такой фактор, как большое время запаздывания при первом доступе к данным. Естественно, это расширило область использования DRDRAM.

Сегодня в области быстродействующих DRAM доминируют синхронные (SDRAM). Для некомпьютерных применений, требующих больших емкостей памяти, эта ситуация может сохраниться на многие годы. В компьютерных схемах DRDRAM представляется сильной альтернативой. Имея времена первого доступа, такие же как у SDRAM, DRDRAM не деградируют по скорости при произвольных обращениях больше, чем обычные синхронные DRAM. Пропускная же способность у них продолжает увеличиваться. Уже имеются микросхемы DRDRAM с 16-разрядным интерфейсом (первоначальные варианты RDRAM имели 8-разрядные). При работе на тактовой частоте 400 МГц и схемотехнике DDR (Double Data Rate), предусматривающей тактирование процессов обоими фронтами импульсов, такие DRDRAM дают пропускную способность (Bandwidth) внутри пакета 1,6 Гбайт/с.

Можно сказать, что в извечной гонке с процессорами 3У впервые из договняющих стали опережающими, поскольку цифру 1,6 Гбайт/с сейчас вряд ли можно использовать в системах.

Структура типа CDRAM

В структурах CDRAM (Cached DRAM, кэшированная DRAM) на одном кристалле с DRAM размещена статическая кэш-память (кэш первого уровня). 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.

§ 4.9. 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 ЗУ. 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 $1M \times 1$, 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. 4.41. 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 T1 и T2 сброшены. Нулевое значение выхода T2 сбрасывает счетчик CTR, блокирует передачу через элемент И-ИЛИ строба $RAS_{\text{рег}}$ и по адресному входу A мультиплексора MUX2 обеспечивает передачу на выход этого мультиплексора адресов от мультиплексора MUX1.

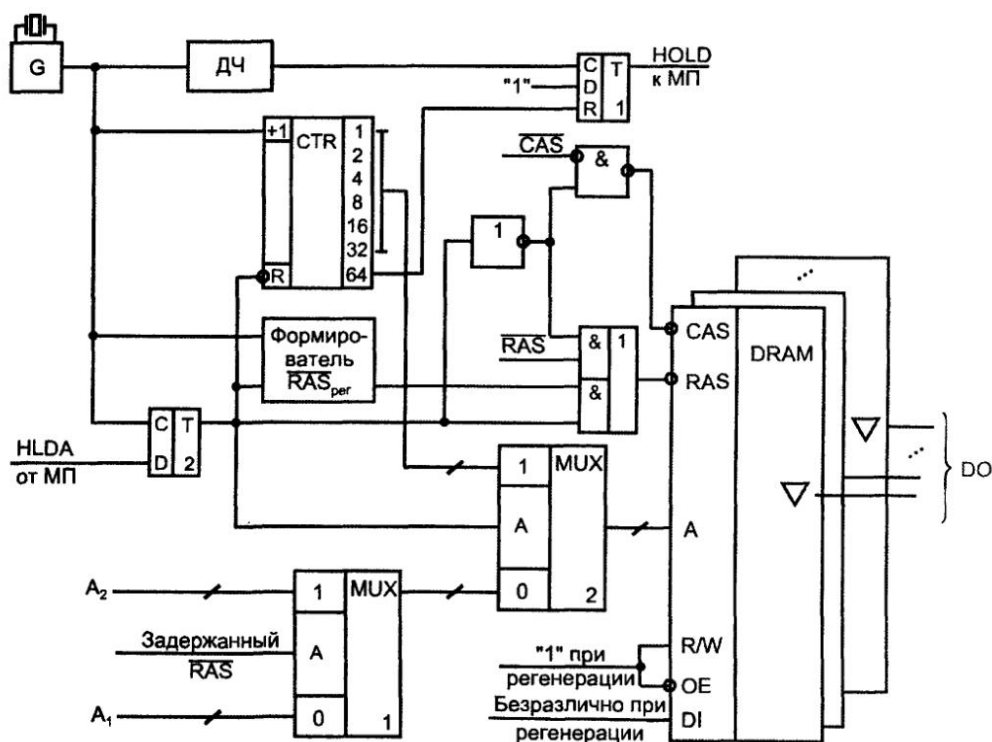


Рис. 4.41. Схема контроллера динамического ОЗУ

При этом модуль памяти получает сигналы \overline{RAS} и \overline{CAS} , соответствующие рабочему режиму, адреса A1 и A2 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 $T_{\text{рег}}$ (составит несколько миллисекунд). Таким образом, с периодом $T_{\text{рег}}$ на выходе ДЧ появляется импульс, что заставляет триггер Т1 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 Т2, в результате чего блокируется передача стробов RAS и CAS на модуль памяти, разрешается передача на вход $\overline{\text{RAS}}_{\text{рег}}$, вырабатываемого формирователем контроллера, мультиплексор MUX2 переключается на передачу адресов со счетчика CTR на адресный вход ЗУ. Одновременно с этим триггер Т2 снимает сигнал асинхронного сброса со входа $\overline{\text{R}}$ счетчика, и он начинает перебирать адреса строк от нулевого до максимального (конкретно в показанной схеме таких адресов 64). Появление импульса переполнения счетчика сбрасывает триггер Т1, обозначая этим окончание операции регенерации и снимая сигнал HOLD. В ответ процессор снимает сигнал HLDA, после чего очередной импульс генератора сбрасывает Т2, 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.

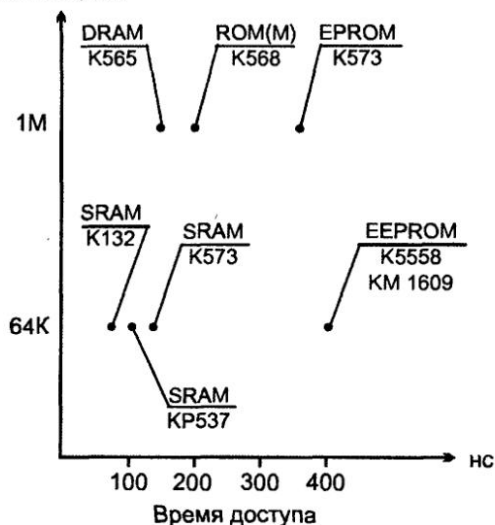
§ 4.10. Заключительные замечания

Архитектуры, технология и схемотехника полупроводниковых ЗУ постоянно развиваются. Поколения динамических ЗУ сменяются приблизительно через пять лет. В 1990 г. доминировали ЗУ с емкостью 1 Мбит, сейчас это ЗУ с емкостью 16 Мбит, а в ближайшие годы будут доминировать ЗУ на 64 Мбита.

Цены на "старые" DRAM (не имеющие максимальных для данного времени емкости и быстродействия) составляют в среднем 2 USD за мегабит, цены на новые ЗУ значительно выше, но имеют тенденцию к быстрому снижению. Например, DRAM емкостью 64 Мбита стоила в 1994 г. 500 USD а в 1996 — 200 USD.

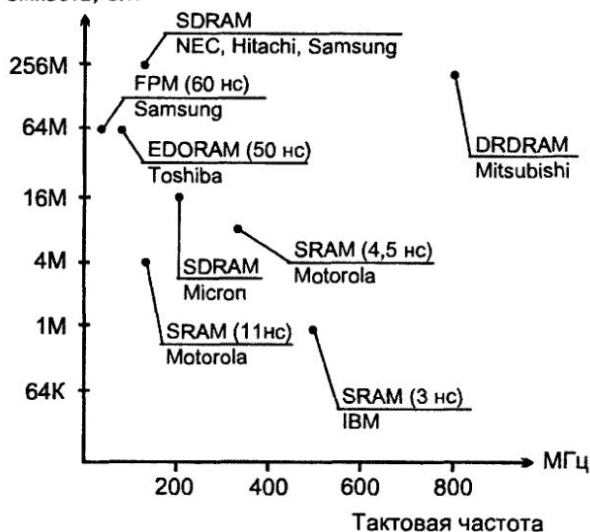
Производство современных ИС ЗУ требует больших инвестиций для создания новых заводов, составляющих миллиарды USD.

Информационная
емкость, бит



а

Информационная
емкость, бит



б

Рис. 4.42. Параметры емкости и быстродействия отечественных (а) и зарубежных (б) запоминающих устройств

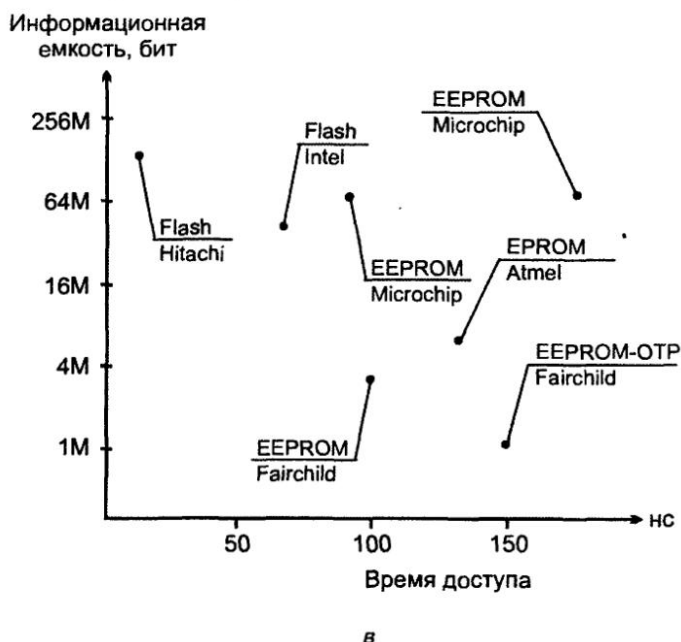


Рис. 4.42. (окончание) Параметры емкости и быстродействия зарубежных (в) запоминающих устройств

Для освоения динамических ЗУ с емкостью 256 Мбит объединили усилия такие известные мощные фирмы, как IBM, Siemens, Toshiba и Samsung. Недавно была продемонстрирована разработка этой группы — DRAM со временем доступа в пакетном режиме 26 нс на кристалле площадью 286 мм² по топологическим нормам 0,25 мкм с 280 миллионами транзисторов на кристалле.

Сравнительные параметры ЗУ различных типов в координатах "максимальная емкость — быстродействие" показаны на рис. 4.42, а, б, в. На рис. 4.42, а приведены параметры отечественных микросхем, на рис. 4.42, б — зарубежных оперативных ЗУ, для которых в связи с пакетными режимами доступа характерным параметром быстродействия является тактовая частота. Для некоторых из этих ЗУ в скобках приведены времена доступа. На рис. 4.42, в даны параметры зарубежных программируемых постоянных ЗУ.

Литература к главе: [9], [10], [16], [19], [20], [24], [25], [27], [29], [35], [37], [56], [58].

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Глава 5

Микропроцессорные БИС/СБИС и их применение в микропроцессорных системах

§ 5.1. Микропроцессорные комплекты БИС/СБИС.

Структура и функционирование микропроцессорной системы. Микроконтроллеры

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 или нескольких БИС/СБИС программно-управляемое устройство, осуществляющее процесс обработки информации и управление им.

Решаемая задача определяется реализуемой МП программой, структура микропроцессорной системы остается неизменной, что и определяет ее универсальность.

Микропроцессоры появились, когда уровень интеграции ИС достиг значений, при которых необходимые для программной реализации алгоритмов блоки удалось разместить на одном или нескольких кристаллах. МП — центральный процессорный элемент микропроцессорной системы (микроЭВМ), в которую также входят память и устройства ввода/вывода (внешние устройства).

Совокупность БИС/СБИС, пригодных для совместного применения в составе микроЭВМ, называют *микропроцессорным комплектом БИС/СБИС (МПК)*. Понятие МПК задает номенклатуру микросхем с точки зрения возможностей их совместного применения (совместимость по архитектуре,

ИР82, обеспечивающем работу шины на нагрузку, образуемую внешними цепями. Собственной нагрузочной способности у выводов МП, как правило, не хватает. Линии AD_{7-0} мультиплексируются. Вначале они передают младший байт адреса, признаком чего служит наличие сигнала ALE (Address Latch Enable), загружающего этот байт в регистр ИР82.

После загрузки регистра сигнал ALE снимается, и содержимое регистра остается неизменным вплоть до новой загрузки в следующем цикле работы процессора. Так формируется 16-разрядная шина адреса, содержащая адрес A_{15-0} . Этот адрес используется блоками постоянной и оперативной памяти ROM и RAM. Адресация портов ввода и вывода данных требует восьмизначного адреса, что соответствует возможности работы не более чем с 256 портами каждого из типов. Адрес портов можно снимать с любой половины адресной шины (во взятом для примера МП состояния обоих полушин адреса при адресации портов дублируются).

После передачи младшего байта адреса шина AD_{7-0} отдается для передачи данных. Эти передачи двунаправлены, направление задается буфером данных BD в зависимости от сигнала T (Transit). При активном состоянии сигнала чтения \overline{RD} (Read) данные передаются справа налево, при пассивном — в обратном направлении. К шине данных подключены информационные выводы всех модулей МПС.

Выводы x_1 и x_2 служат для подключения кварцевого резонатора или иных контуров, задающих частоту тактовому генератору, расположенному в МП. Тактирование системы производится на частоте, равной половине частоты резонанса кварца или иного контура, поскольку генератор работает на триггер, с которого снимаются сигналы тактирования модулей МПС, а триггер делит частоту на 2. Вход \overline{RESIN} является входом асинхронного сброса, приводящим МП в исходное состояние. Сигнал 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{RESIN} нарастает постепенно, и в течение некоторого времени после включения питания остается низким (ниже порогового), что равноценно подаче сигнала \overline{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.

По однонаправленной адресной шине МП посылает адреса, определяя объект, с которым будет обмен, по шине данных (двунаправленной) обменива-

ется данными с модулями (блоками) системы, по шине управления идет обмен управляющей информацией.

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. 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 выполняется специальными блоками, называемыми *адаптерами* (интерфейсными блоками ввода-вывода). Напомним, что интерфейсом называют совокупность аппаратных и программных средств, унифицирующих процессы обмена между модулями системы.

На схеме (рис. 5.1) модули системы показаны укрупненно. Кроме обозначенных блоков, в состав систем входят обычно и более сложные, чем адаптеры, блоки управления внешними устройствами — *контроллеры*. К их числу относятся, прежде всего, контроллеры прерываний и прямого доступа к памяти. Имеются также контроллеры клавиатуры, дисплея, дисковой памяти и т. д.

Контроллеры прерываний обеспечивают обмен с внешними устройствами в режиме прерывания (временной остановки) выполняемой программы для обслуживания запроса от внешнего устройства.

Контроллеры прямого доступа к памяти обслуживают режим прямой связи между внешними устройствами и памятью без участия МП. При управлении обменом со стороны МП пересылка данных между внешними устройствами и памятью происходит в два этапа — сначала данные принимаются микропроцессором, а затем выдаются им на приемник данных. В режиме прямого доступа к памяти МП отключается от шин системы и передает управление ими контроллеру прямого доступа, а передачи данных осуществляются в один этап — непосредственно от источника к приемнику.

В состав МПС часто входят также программируемые таймеры, формирующие различные сигналы (интервалы, последовательности импульсов и т. д.) для проведения операций, связанных со временем.

Микроконтроллеры

Микроконтроллеры (МК) — разновидность микропроцессорных систем (микроЭВМ), ориентированная на реализацию алгоритмов управления техническими устройствами и технологическими процессами. В сравнении с

универсальными микроЭВМ микроконтроллеры проще, и уже около 25 лет тому назад оказалось возможным разместить практически всю схемотехнику МК на одном кристалле, что и дало начало их развитию. Вторым названием МК стало название *"однокристальная микроЭВМ"*. Разработка МК означала появление БИС такой функциональной законченности, которая позволяет решать в полном объеме задачи определенного класса.

Что отличает МК от микроЭВМ универсального назначения? Прежде всего, это малый объем памяти и менее разнообразный состав внешних устройств. В состав универсальной микроЭВМ 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 заранее известные несложные алгоритмы, и для размещения программ им требуются емкости памяти, на несколько порядков меньшие, чем у микроЭВМ широкого назначения. Набор внешних устройств также существенно сужается, а сами они значительно проще. В результате модули микроЭВМ конструктивно самостоятельны, а МК выполняется на одном кристалле, хотя в его составе имеются модули того же функционального назначения.

Сопоставляя микропроцессор (т. е. центральный процессорный элемент системы) и МК (т. е. микросхему простой системы в целом) с точки зрения коммерческих потребностей, можно четко видеть преобладание МК. Число пользователей МК в несколько раз превышает число пользователей отдельных микросхем МП. Применение МК поддерживается такими областями массового производства, как бытовая аппаратура, станкостроение, автомобильная промышленность и т. д.

Первые МК выпущены фирмой Intel в 1976 г. (восьмиразрядный МК 8048). В настоящее время многими поставщиками выпускаются 8-, 16- и 32-разрядные МК с емкостью памяти программ до десятков Кбайт, 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 (фирмы Intel). Очень многие производители выпускают аналоги этих семейств или совместимые с ними МК. В отечественной номенклатуре это K1816BE51, K1830BE51 (восьмиразрядные МК). 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 популярное семейство МК серии AT89 с Флэш-памятью программ, являющееся функциональным аналогом семейства восьмиразрядных МК фирмы Intel.

Небезынтересно, что, несмотря на появление новых 16- и 32-разрядных МК, наибольший успех на рынке остается за 8-разрядными. Сейчас около половины рынка МК (приблизительно 6 млрд долларов) остается за этими МК, что означает их лидирование с большим отрывом относительно представителей других семейств.

В структуре МК семейства AT89C (рис. 5.2) используются отдельные блоки программной памяти типа Флэш и ОЗУ данных (Гарвардская архитектура). Диапазоны емкостей памяти, как и частот генератора тактовых импульсов ГТИ, приведенные на рис. 5.2, характеризуют параметры представителей семейства от младшего до старшего. При необходимости возможно подключение внешних БИС ПЗУ, ОЗУ для расширения пространства памяти. Средства ввода/вывода представлены 4 параллельными портами (32 линии) и линиями TxD (выход передатчика) и RxD (вход приемника) для последовательного ввода/вывода. В состав МК входят 2—3 таймера-счетчика (16-разрядных), которые дают системные метки времени и обрабатывают интервалы. Для сокращения ширины физического интерфейса функции линий параллельных портов совмещены, и в разных режимах имеют разное назначение. Система прерываний с 5 источниками запросов радиального типа (см. § 5.3) обслуживает 2 внешних запроса, 2 запроса от таймеров и 1 от последовательного порта. При частоте ГТИ 12 МГц большинство команд выполняется за 1 мкс, некоторые команды — за 2 мкс.

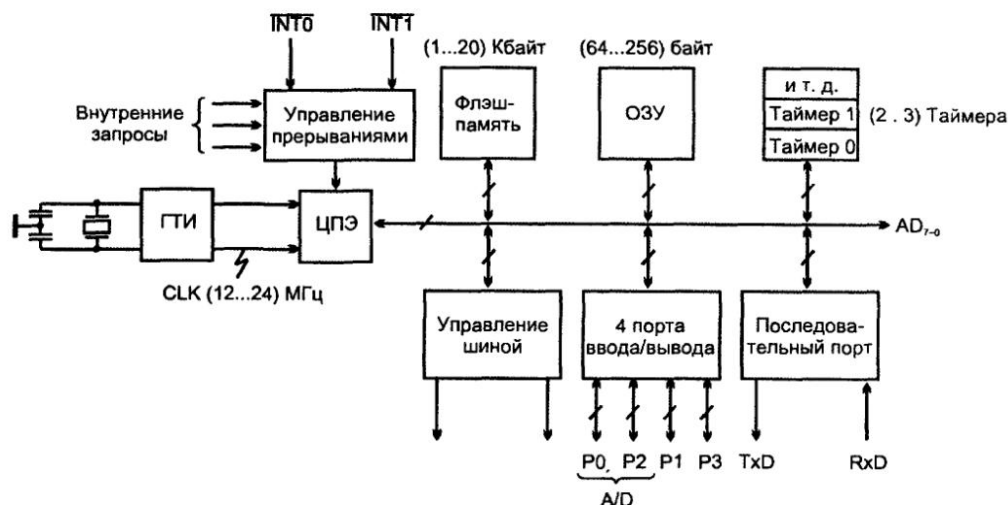


Рис. 5.2. Структура микроконтроллера AT89C

В последующих параграфах этой главы и в следующей главе, несмотря на коммерческое преобладание МК над МП, рассмотрены микропроцессор и набор интерфейсных схем (адаптеров, контроллеров), входящих в микро-

процессорную систему. При этом интерфейсные схемы соответствуют не только отдельным кристаллам или областям кристаллов большой сложности, но и мегафункциям (макрофункциям) библиотек функциональных блоков современных СБИС программируемой логики. Такое решение принято, поскольку указанный комплект микросхем, рассчитанный на построение МПС широкого назначения, полнее иллюстрирует как работу блоков, так и особенности решения задач управления памятью и внешними устройствами, организации системы прерываний, реализации прямого доступа к памяти и т. д. В МК перечисленные и другие задачи решаются в ограниченной степени и более простыми средствами, поэтому знание универсальных микросхем, рассчитанных на построение МПС, позволяет легко осваивать и микроконтроллерную технику.

Подробные сведения о современных микроконтроллерах приведены в справочнике [7].

§ 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 сама память независимо от ее технической реализации может быть условно представлена набором регистров (ячеек), число которых M , а разрядность — N (рис. 5.3).

$N-1$ RG0	0
RG1	
RG2	
⋮	
RG $M-2$	
RG $M-1$	

Рис. 5.3. Условное представление 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 адрес операнда. Команда сохраняет компактность, но для ее выполнения требуется предварительная настройка — загрузка адреса в регистр (регистр косвенного адреса). Косвенная адресация удобна при обработке списков, когда настройка производится однократно, а очередной адрес получается модификацией предыдущего (изменением его на единицу).

При *непосредственной адресации* в команде содержится сам операнд.

Помимо перечисленных имеются и более сложные способы адресации: индексная, относительная и др., однако в простейших МП они не используются.

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 АП "лишнего" пространства. Частным случаем неабсолютной адресации ВУ является так называемая *линейная селекция* (линейный выбор), подробнее рассмотренная ниже.

В простых МПС часто адресный код 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 ВУ 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 ИО/М, указывающий, адресуется память или ВУ. При этом память может использовать все 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 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 определенных объектов.

Для краткости записей адреса в АП обычно выражают в шестнадцатиричной системе счисления, для оценки емкостей АП используется часто единица измерения $K = 2^{10} = 1024$ или $M = 2^{20} = 1048576$.

Модуль 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 — разрядность слов. 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 к 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).

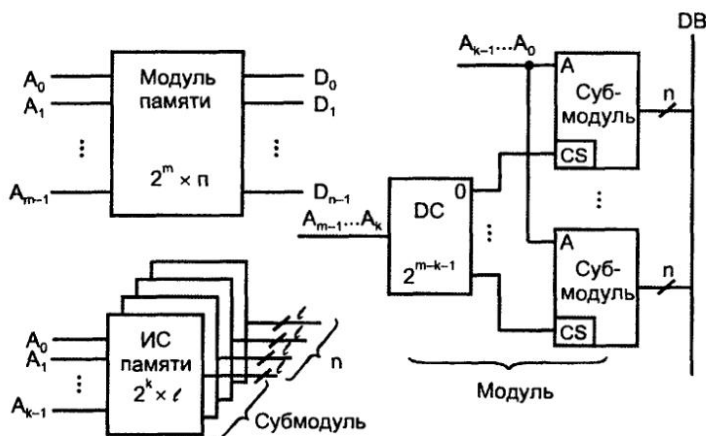


Рис. 5.4. Структуры модуля памяти

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} , \overline{EN}), признак обращения к ВУ или памяти ($\overline{IO/M}$). Процессор обычно вырабатывает минимальную группу сигналов, тогда как в системном интерфейсе может быть предусмотрена несколько иная группа. В частности, МП K1821BM85A дает три сигнала: сигнал чтения (\overline{RD}), записи (\overline{WR}) и сигнал $\overline{IO/M}$, т. е. 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 ОЗУ могут быть асинхронными или тактируемыми. Для тактируемых ОЗУ нужен импульсный характер какого-либо сигнала 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 $CS = \overline{MEMR} \times \overline{MEMW}$. При этом обеспечивается пассивное состояние сигнала \overline{CS} на интервалах, на которых не действуют ни сигнал чтения, ни сигнал записи ($\overline{MEMR} = \overline{MEMW} = 1$).

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 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 ниже после изучения конкретного МП.

Виды обмена

Выполнение процессором операций записи и чтения данных может проходить в режимах *программно-управляемого обмена, прерывания и прямого доступа к памяти (ПДП)*.

В первом случае инициатором обмена является программа. 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.

При обменах по прерываниям 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 повышает эффективность МПС.

§ 5.3. Микропроцессор серии 1821 (Intel 8085A)

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 85...92% от общего объема выпуска микропроцессоров. При изучении МП целесообразно ориентироваться на конкретные образцы. Здесь выбран МП K1821BM85A — 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 выпускается промышленностью и встречается в каталогах фирм. Естественно, что область его применения не являются компьютеры, в которых сейчас применяют гораздо более мощные и производительные МП. Такие МП, как K1821BM85A используются в системах управления различной аппаратурой, где их возможностей хватает.

Структура микропроцессора K1821BM85A

Структура микропроцессора K1821BM85A 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;
- ALU (Arithmetic-Logic Unit) — арифметико-логическое устройство, 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 посту-

пает результат операции¹. 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);

- RF (Register Flags) — регистр флажков, т. е. битов, указывающих признаки результатов арифметических или логических операций, выполненных в АЛУ.

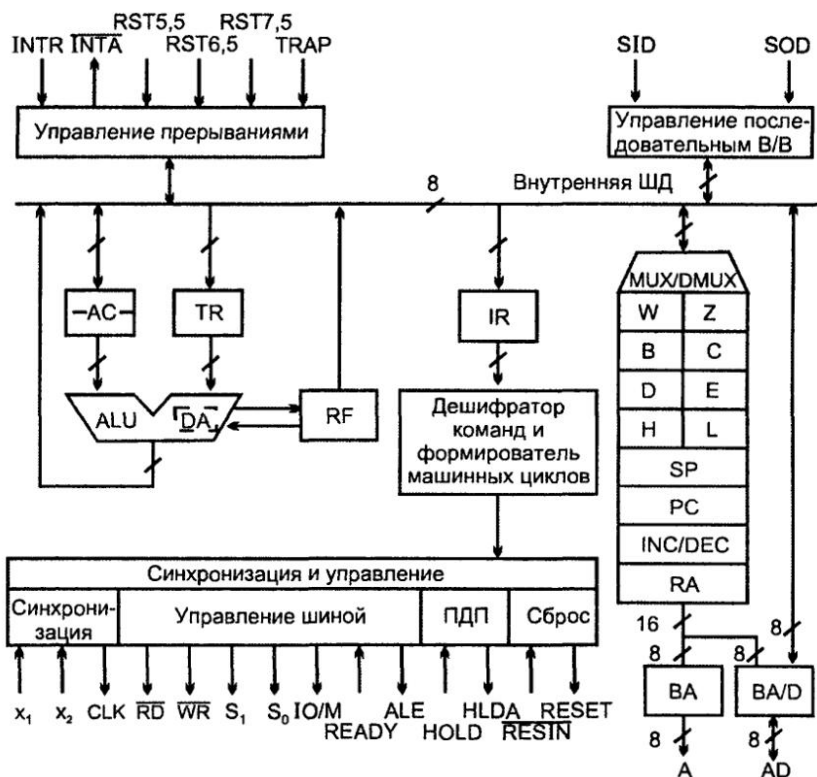


Рис. 5.5. Структура микропроцессора K1821BM85A

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 ос-

¹ Звездочкой обозначен обобщенный символ операции.

тальных 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 регистровой 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" 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.

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.

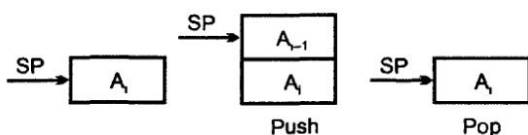


Рис. 5.6. Реализация стека в микропроцессорной системе

Основное назначение стека — обслуживание прерываний программы и выполнения подпрограмм.

Программный счетчик PC (Program Counter) дает адрес команды, и может обращаться в любую из 64К ячеек АП. При сбросе МП 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) 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 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.

Буфер адреса ВА с тремя состояниями выхода выдает сигналы старших разрядов адреса на линии адресной шины A_{15-8} .

Буфер шины адресов/данных ВА/D с тремя состояниями выхода передает на шину A/D с разделением во времени 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 ниже.

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 K1821 имеет следующие типы машинных циклов:

1. Выборки команды (OF, Opcode Fetch).
2. Чтения из памяти (MR, Memory Read).
3. Записи в память (MW, Memory Write).
4. Чтения из ВУ (IOR, Input-Output Read).
5. Записи в ВУ (IQW, Input-Output Write).
6. Подтверждения прерывания (INA, Interrupt Acknowledge).
7. Освобождения шин (BI, Bus Idle).
8. Останов (HALT).

В начале каждого машинного цикла генерируются сигналы состояния, идентифицирующие тип цикла и действующие в течение всего цикла.

Функции выводов и сигналов:

- 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 — 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} — стробы чтения или 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 состоя-

ние 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 **RD**, **WR** и **INTA** для различных машинных циклов имеют следующие значения (табл. 5.1):

Таблица 5.1

Тип мц	Сигналы состояния			Сигналы управления		
	IO/M	S_1	S_0	RD	WR	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;
- **RESIN (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** — 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 stroba $\overline{\text{INTA}}$, если прерывание разрешено программой. Адрес подпрограммы, вызываемой этим входом, выдается внешним устройством. При сбросе прием сигнала запрещается (прерывания запрещены);
- ❑ $\overline{\text{INTA}}$ (Interrupt Acknowledge) — выход stroba подтверждения векторного прерывания после завершения текущего командного цикла. Используется для чтения вектора прерывания;
- ❑ **RST 5,5; RST 6,5; RST 7,5** — 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** — сигнал подтверждения захвата (Hold Acknowledge). 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 RD, 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.

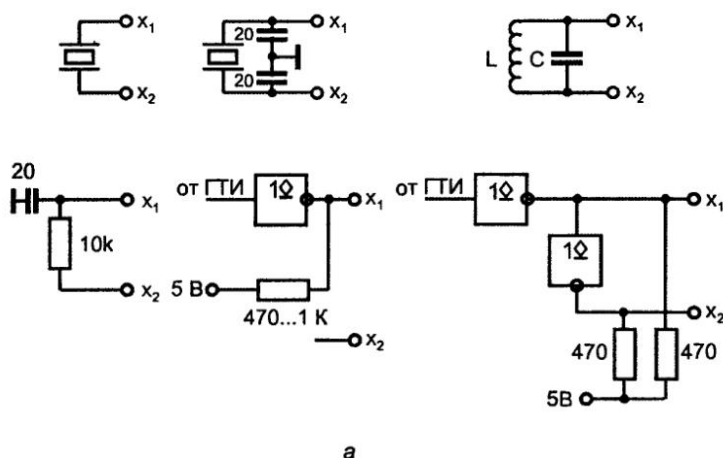


Рис. 5.7. Внешние элементы тактового генератора (а) и формирование синхросигналов (б) в микропроцессоре K1821BM85A

Для образования сигналов синхронизации CLK выход генератора подается на вход счетного триггера (рис. 5.7, б). Триггер формирует две последовательности противофазных импульсов Φ1 и Φ2 для тактирования внутренних схем МП. Сигнал синхронизации системы CLK синфазен импульсам Φ2. Сигнал ALE формируется как один импульс последовательности Φ1, выделяемый из нее в первом такте (Т1) каждого машинного цикла. Буфер выдачи сигнала ALE во внешние цепи имеет вход разрешения 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 М1 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.

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 адре-

са 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.

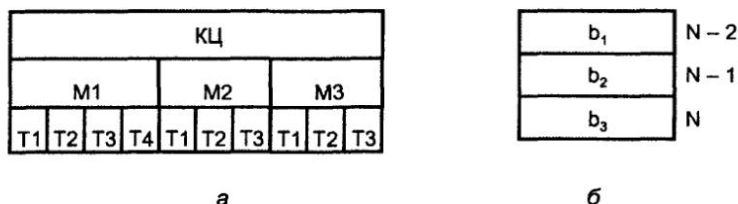


Рис. 5.8. Циклы и такты микропроцессора K1821BM85A (а) и пример размещения команды в памяти микропроцессорной системы (б)

Таблица 5.2

Регистры							Пары регистров			
В	С	Д	Е	Н	Л	А	В	Д	Н	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 ниже. 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 b₃b₂ (Store Accumulator Direct), передающей содержимое аккумулятора в ячейку памяти при прямой адресации, т. е. указании адреса ячейки в самой команде. Команда трехбайтная, для ее передачи в МП требуются три машинных цикла, в первом из которых байт b₁ передается в регистр команд IR, в последующих байты b₂ и b₃ 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 составит из четырех машинных циклов в следующем порядке OF-MR-MR-MW.

Каждый машинный цикл делится на *такты* (состояния) — интервалы между одноименными фронтами тактовых импульсов.

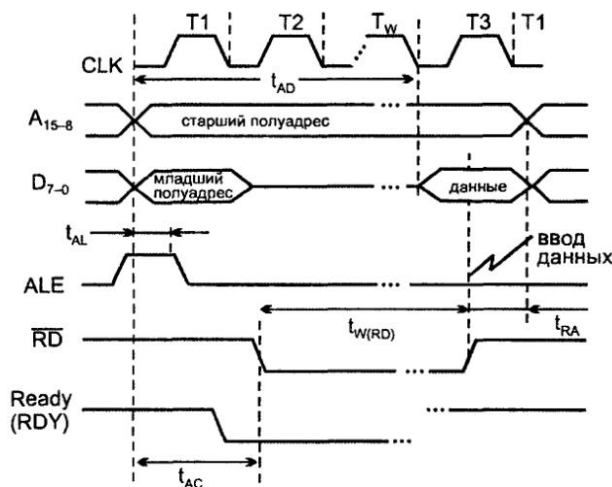


Рис. 5.9. Временные диаграммы цикла чтения микропроцессора

Типовые действия, выполняемые в тактах машинного цикла:

- T₁** Адрес памяти или ВУ выставляется на AD₇₋₀ и A₁₅₋₈, генерируется сигнал ALE для фиксации битов AD₇₋₀. На линиях IO/M, S₁ и S₀ 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.

T_4 Декодируется содержимое регистра команд.

$T_{5,6}$ Используются при необходимости для завершения некоторых команд. Системные шины не используются.

Машинный цикл всегда содержит такты $T_1...T_3$, 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.

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 системы запросы возникают при сбоях в работе, переполнении разрядной сетки, попытке деления на ноль и т. д., а также при требованиях обслуживания от внешних устройств. Извне могут поступать 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 микропроцессором. 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 соответствующей команде основной программы.

Микропроцессор K1821 имеет пять входов прерывания и один выход управления им \overline{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 микропроцессором 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 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 микропроцессора запрещает обслуживание этих запросов, для их последующего разрешения следует подать команду EI. Имеется также возможность отдельного 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 команды 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 маскирования 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

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, b₂ передает в аккумулятор байт b₂, т. е. выполняет действие (A) \rightarrow (b₂) пересылки в аккумулятор данных при непосредственной адресации. Байт b₂ 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 таблице (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 b₃b₂, 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.

Таблица 5.3

Мнемокод	Код	Флажки	Число			Содержание
			бай- тов	так- тов	цик- лов	
1	2	3	4	5	6	7
Команды пересылки						
MOV r ₁ , r ₂	01ПППPIII	—	1	4	1	Пересылка из регистра r ₂ в регистр r ₁

Таблица 5.3 (продолжение)

Мнемокод	Код	Флажки	Число			Содержание
			бай- тов	так- тов	цик- лов	
1	2	3	4	5	6	7
Команды пересылки						
MOV M, r	01110IIII	—	1	7	2	Пересылка из регистра в память
MOV r, M	01ППП110	—	1	7	2	Пересылка из памяти в регистр
MVI r, b ₂	00ППП110	—	2	7	2	Пересылка непосредственных данных в регистр
MVI M, b ₂	36	—	2	10	3	Пересылка непосредственных данных в память
LXI r _p b ₃ b ₂	00ПР0001	—	3	10	3	Загрузка непосредственных данных в пару регистров
LDA b ₃ b ₂	3A	—	3	13	4	Прямая загрузка аккумулятора
STA b ₃ b ₂	32	—	3	13	4	Прямая запись аккумулятора в память
LHLD b ₃ b ₂	2A	—	3	16	5	Прямая загрузка пары регистров
SHLD b ₃ b ₂	22	—	3	16	5	Прямая запись пары регистров H в память
LDAX r _p	00ПР1010	—	1	7	2	Косвенная загрузка аккумулятора посредством пары регистров В или D
STAX r _p	00ПР0010	—	1	7	2	Косвенная запись аккумулятора в память посредством пары регистров В или D
XCHG	EB	—	1	4	1	Обмен между парами регистров H и D
Команды арифметических и логических операций						
ADD r	10000IIII	+	1	4	1	Сложение регистра и аккумулятора
ADD M	86	+	1	7	2	Сложение памяти и аккумулятора

Таблица 5.3 (продолжение)

Мнемокод	Код	Флажки	Число			Содержание
			бай- тов	так- тов	цик- лов	
1	2	3	4	5	6	7
Команды арифметических и логических операций						
ADI b_2	C6	+	2	7	2	Сложение непосредственных данных и аккумулятора
ADC r	10001IIII	+	1	4	1	Сложение регистра и аккумулятора с переносом
ADC M	8E	+	1	7	2	Сложение памяти и аккумулятора
ACI b_2	CE	+	2	7	2	Сложение непосредственных данных и аккумулятора с переносом
SUB r	10010IIII	+	1	4	1	Вычитание регистра из аккумулятора
SUB M	96	+	1	7	2	Вычитание памяти из аккумулятора
SUI b_2	D6	+	2	7	2	Вычитание непосредственных данных из аккумулятора
SBB r	10011IIII	+	1	4	1	Вычитание регистра из аккумулятора с заемом
SBB M	9E	+	1	7	2	Вычитание памяти из аккумулятора с заемом
SBI b_2	DE	+	2	7	2	Вычитание непосредственных данных из аккумулятора с заемом
INR r	00ППП100	(+)	1	4	1	Инкремент регистра
INR M	34	(+)	1	10	3	Инкремент памяти
DCR r	00ППП101	(+)	1	4	1	Декремент регистра
DCR M	35	(+)	1	10	3	Декремент памяти
INX r_p	00ПР0011	—	1	6	1	Инкремент пары регистров

Таблица 5.3 (продолжение)

Мнемокод	Код	Флажки	Число			Содержание
			бай- тов	так- тов	цик- лов	
1	2	3	4	5	6	7
Команды арифметических и логических операций						
DCX r _p	00PR1011	—	1	6	1	Декремент пары регистров
DAD r _p	00PR1001	C	1	10	3	Сложение регистровой пары R с регистровой парой
DAA	27	+	1	4	1	Преобразование аккумулятора в двоично-десятичный код
ANA r	10100IIII	+	1	4	1	Логическое И регистра и аккумулятора
ANA M	A6	+	1	4	1	Логическое И памяти и аккумулятора
ANI b ₂	E6	+	2	7	2	Логическое И непосредственных данных и аккумулятора
XRA r	10101IIII	+	1	4	1	Исключающее ИЛИ регистра и аккумулятора
XRA M	AE	+	1	7	2	Исключающее ИЛИ памяти и аккумулятора
XRI b ₂	EE	+	2	7	2	Исключающее ИЛИ непосредственных данных и аккумулятора
ORA r	10110IIII	+	1	7	2	Логическое ИЛИ регистра и аккумулятора
ORA M	B6	+	1	7	2	Логическое ИЛИ памяти и аккумулятора
ORI b ₂	F6	+	2	7	2	Логическое ИЛИ непосредственных данных и аккумулятора
CMP r	10111IIII	+	1	4	1	Сравнение регистра и аккумулятора
CMP M	10111110	+	1	7	2	Сравнение памяти и аккумулятора

Таблица 5.3 (продолжение)

Мнемокод	Код	Флажк и	Число			Содержание
			бай- тов	так- тов	цик- лов	
1	2	3	4	5	6	7
Команды арифметических и логических операций						
CPI b_2	FE	+	2	7	2	Сравнение непосредственных данных и аккумулятора
CMA	2F	—	1	4	1	Инвертирование аккумулятора
STC	37	C	1	4	1	Установка флажка переноса
CMC	3F	C	1	4	1	Инвертирование флажка переноса
RLC	07	C	1	4	1	Циклический сдвиг аккумулятора влево
RRC	0F	C	1	4	1	Циклический сдвиг аккумулятора вправо
RAL	17	C	1	4	1	Циклический сдвиг аккумулятора влево через разряд переноса
RAR	1F	C	1	4	1	Циклический сдвиг аккумулятора вправо через разряд переноса
Команды управления						
JMP b_3b_2	C3	—	3	10	3	Безусловный переход
J _{усл} b_3b_2	11УУУУ01	—	3	10	3	Условный переход
CALL b_3b_2	CD	—	3	18	5	Безусловный вызов подпрограммы
C _{усл} b_3b_2	11УУУ100	—	3	2/5	9/18	Условный вызов подпрограммы
RET	C9	—	3	10	3	Возврат
R _{усл}	11УУУ100	—	3	17/11	5/3	Возврат при условии
RST n	11nnnn111	—	1	11	3	Повторный запуск
SPHL	E9	—	1	6	1	Пересылка пары регистров H в SP

Таблица 5.3 (окончание)

Мнемокод	Код	Флажки	Число			Содержание
			бай- тов	так- тов	цик- лов	
1	2	3	4	5	6	7
Специальные команды						
PUSH r_p	11РП0101	—	1	11	3	Пересылка пары регистров в стек
PUSH PSW	F5	—	1	11	3	Пересылка аккумулятора и регистра флажков в стек
POP r_p	11РП0001	—	1	10	3	Загрузка регистровой пары из стека
POP PSW	F1	+	1	10	3	Загрузка аккумулятора и регистра флажков из стека
XTHL	E3	—	1	18	5	Обмен между регистровой парой H и стеком
PCHL	F9	—	1	5	1	Пересылка регистровой пары H в PC
IN port	DB	—	2	10	3	Ввод
OUT port	D3	—	2	10	3	Вывод
EI	FB	—	1	4	1	Разрешение прерывания
DI	F3	—	1	4	1	Запрещение прерывания
HLT	76	—	1	7	2	Останов
NOP	00	—	1	4	1	Нет операции
RIM	20	—	1	4	1	Чтение маски прерывания
SIM	30	—	1	4	1	Запись маски прерывания

Во второй графе коды первого байта команды $b1$ даются в двоичном восьмиразрядном представлении, если требуется указать в них адреса операндов, или в двухразрядном шестнадцатиричном представлении в иных случаях. Разряды обобщенных адресов регистров — источников данных выражены буквами ИИИ, регистров — приемников данных — буквами ППП, пар регистров — буквами ПР. Подставляя вместо буквенных символов определенные адреса, получим коды конкретных вариантов команды (например, из обоб-

шенной формы "пересылка из регистра в регистр" конкретный вариант "пересылка из регистра В в регистр D"). Коды условий, при выполнении которых осуществляется указанная в команде операция, обозначены буквами УУУ, расшифровка которых имеет вид табл. 5.4.

Таблица 5.4

УУУ	Мнемокод	Условия
000	NZ	Неравенство нулю
001	Z	Равенство нулю
010	NC	Отсутствие переноса
011	C	Наличие переноса
100	PO	Нечетность
101	PE	Четность
110	P	Плюс
111	M	Минус

Включая конкретные условия в мнемкоды команд, получаем их варианты. Например, команда условного перехода из обобщенной формы $J_{\text{усл}} b_3b_2$ переводится в вариант JNZ b_3b_2 — переход к команде с адресом b_3b_2 , если признак результата говорит о том, что результат не равен нулю. Признаки формируются в регистре флажков, формат которого представляется в виде:

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 выполнении команды текущее содержимое программного счетчика PC загружается в стек, а в PC формируется код с нулевым старшим байтом и младшим байтом вида 000nnn000.

Операция сравнения производится вычитанием операндов с установкой признака результата ($Z = 1$ — равные операнды, $S = 0$ — содержимое аккумулятора больше второго операнда, $S = 1$ — меньше).

В третьей графе прочерк означает, что выполнение команды не сопровождается выработкой флажков-признаков, знак плюс говорит об установке всех признаков, знак плюс в скобках — об установке всех признаков, кроме признака наличия или отсутствия переноса C , а символ C означает, что вырабатывается только признак наличия или отсутствия переноса.

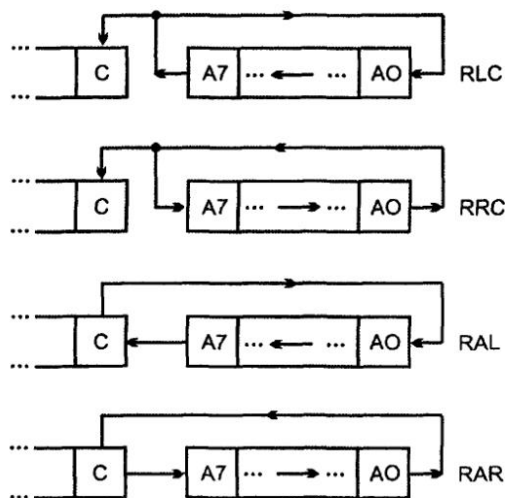


Рис. 5.11. Схемы, поясняющие выполнение сдвигов микропроцессором

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).

Команды RIM и SIM подробно рассмотрены выше. Два возможных значения чисел тактов и циклов приведены для команд, выполнение которых зависит от состояния признаков — флажков.

В табл. 5.5 на примере микропроцессоров фирмы Intel приведены сравнительные параметры двух микропроцессоров, "возраст" которых составляет около двадцати лет, и двух современных, появившихся в 1997—99 гг. Первые сохраняют до сих пор свое значение как средство построения простых систем управления техническими объектами и технологическими процессами, вторые — как средство построения новейших компьютеров.

Таблица 5.5

Параметр	Low End Microprocessors		High End Microprocessors	
	8085A	8086	Pentium II	Pentium III
Год выпуска	1977	1978	1997	1999
Разрядность	8	16	64	64
Адресное пространство	64K	1M	64M	1Г
Частота тактовых импульсов, МГц	3	8	333	600
Среднее число тактов на операцию	≈10	≈10	0,5	—

Таблица 5.5 (окончание)

Параметр	Low End Microprocessors		High End Microprocessors	
	8085A	8086	Pentium II	Pentium III
Число выводов корпуса	40	40	242	242
Число транзисторов, тыс.	6,2	29	7500	28000

Для МП 8085А укажем также следующие данные, необходимые для практической работы с ним:

напряжение питания, В	$5 \pm 10\%$
ток потребления, мА	≤ 170
ток входа, мкА	≤ 10
емкость входа, пФ	≤ 10
ток выхода при низком уровне выходного напряжения, мА	≤ 2
ток выхода при высоком уровне выходного напряжения, мА	$\leq 0,4$
максимальная емкость нагрузки, пФ	150

По мере развития микропроцессорной техники происходит естественный процесс специализации МП соответственно областям их применения. Важнейший класс проблемно ориентированных МП — процессоры цифровой обработки сигналов, которые находят применение в современных системах связи, обработки графических изображений, медицине и многих других областях. Сведения о таких МП, в частности, можно почерпнуть в работе [17].

Пример выполнения команды

Выполнение команды реализуется в МПС через работу ее шин. 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 Н 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 (см. табл. 5.3). Пара регистров 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 ячейки памяти 2000...2002.

Однобайтная команда MOV A, M пересылки из памяти в аккумулятор является вариантом команды MOV r, M с кодом 01ППП110. 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 которой 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.

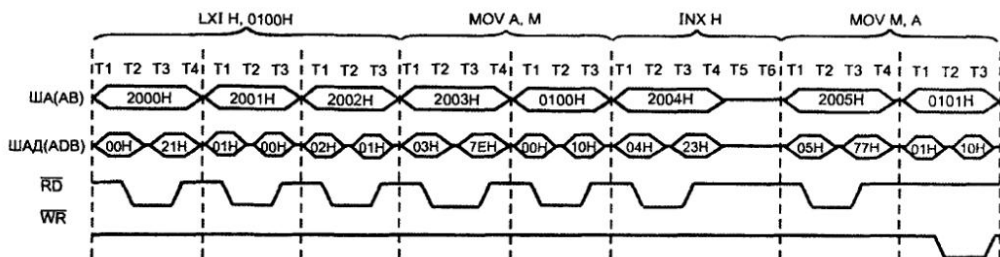


Рис. 5.12. Реализация программы сигналами шин микропроцессорной системы

§ 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.

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 следующей таблицей (числа в первой строке измеряются в единицах K).

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 $4K \times 8$. 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 $(8K + 4K - 1)$).

Сигнал разрешения работы дешифратора $E = \bar{E}_1 \bar{E}_2 \bar{E}_3$. Двенадцать младших разрядов адреса выбирают ячейку в субмодуле. Старшие разряды адреса декодируются для формирования сигналов выбора кристалла \overline{CS} . Стробящим сигналом \overline{RD} определяется интервал выполнения операции чтения.

Одним из условий разрешения работы дешифратора является низкий уровень сигнала IO/M.

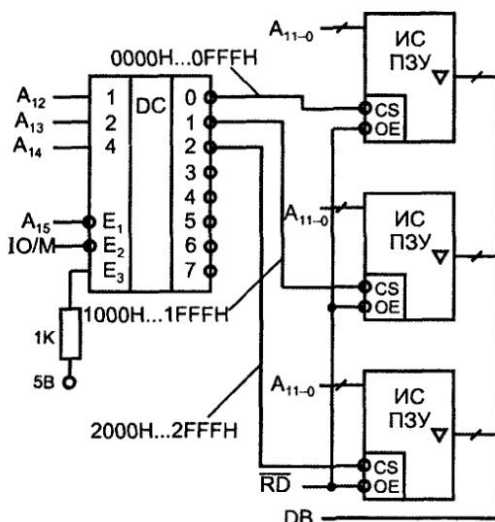


Рис. 5.13. Пример адресации модуля памяти

Пример 2

Адресация модуля памяти, составленного из субмодулей с организацией $2K \times 8$ 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 $8000H \dots BFFFH$, 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 $A_{13} \dots A_{11}$ 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.

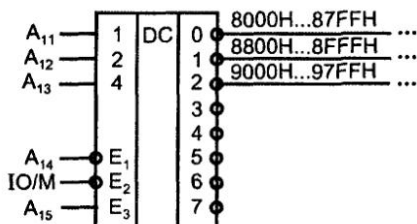
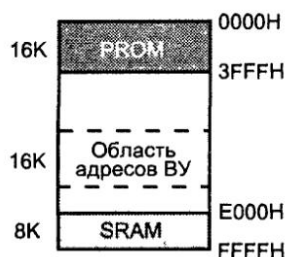


Рис. 5.14. Вариант адресации модуля памяти

Пример 3

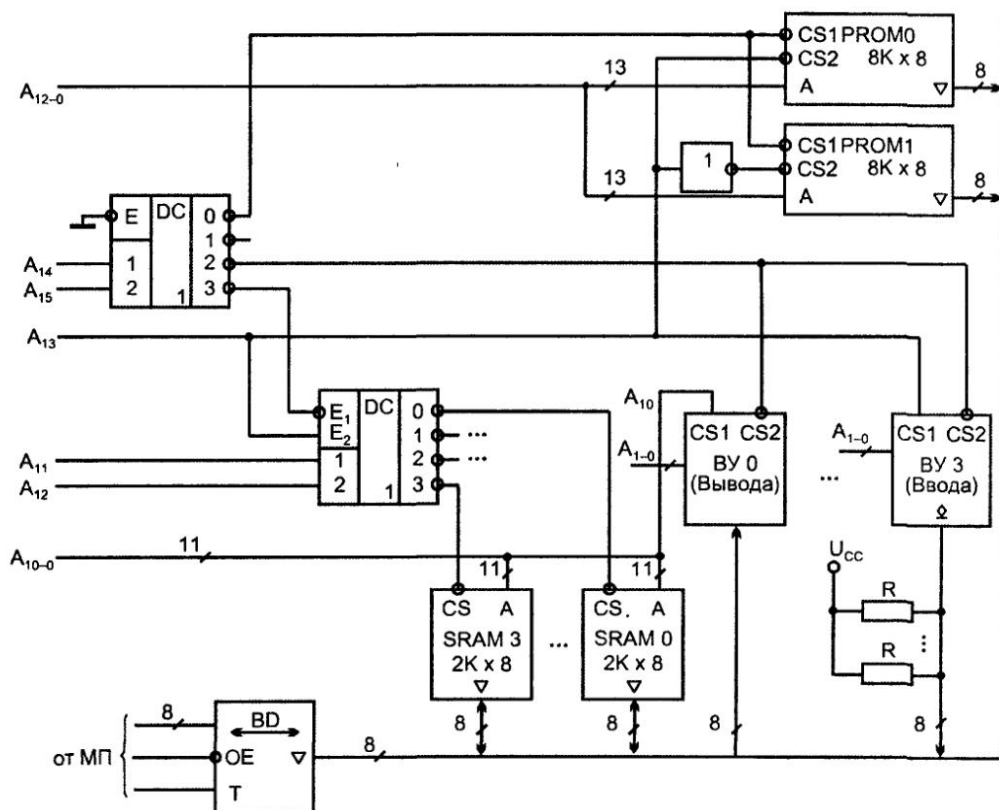
Рассмотрим пример размещения в АП адресов ПЗУ, ОЗУ и ВУ, т. е. совмещенный ввод/вывод. Для памяти используем абсолютную адресацию, а для

ВУ — линейную селекцию. Пусть для ПЗУ отведено 16К адресов в начале АП, адреса ВУ занимают третью четверть АП, а адреса ОЗУ занимают последние 8К 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.



а

Рис. 5.15. Пример распределения адресного пространства между модулями памяти и внешними устройствами (а) и схема адресации модулей памяти и внешних устройств (б)



б

Пусть ПЗУ строится на микросхемах с организацией $8K \times 8$, а ОЗУ на микросхемах $2K \times 8$. 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.

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
ВУ0	1	0	X	X	X	X	X	X	X	0	0	0	0	1	d	d
ВУ1	1	0	X	X	X	X	X	X	X	0	0	0	1	0	d	d
ВУ2	1	0	X	X	X	X	X	X	X	0	0	1	0	0	d	d
ВУ3	1	0	X	X	X	X	X	X	X	0	1	0	0	0	d	d
ВУ4	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 A₁₃ разрешает работу микросхемы PROM0 в первой половине первой четверти АП при нулевом состоянии и работу микросхемы PROM1 — при единичном. Линии A₂...A₆ использованы для линейной селекции внешних устройств, а линии A₁₂ и A₁₁ 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).

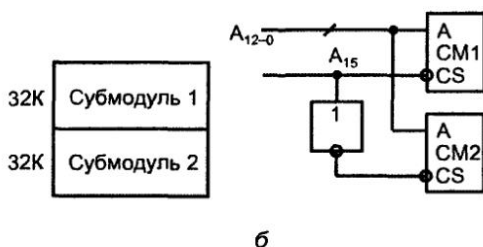
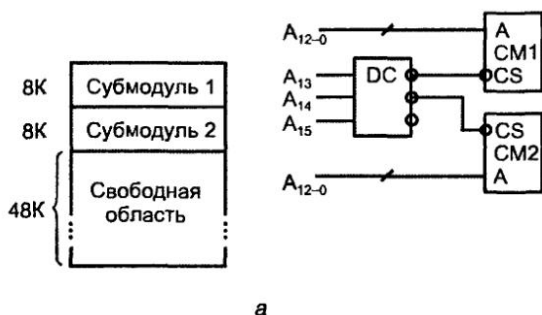


Рис. 5.16. Примеры реализации абсолютной (а) и неабсолютной (б) адресации субмодулей памяти

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 $0XXdd\dots d$ принадлежат субмодулю СМ1, а это соответствует верхним 32К АП. Все адреса вида $1XXdd\dots d$ 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 CM1 и CM2 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 CM1, вторая — адресами CM2, 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 потребуются 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.

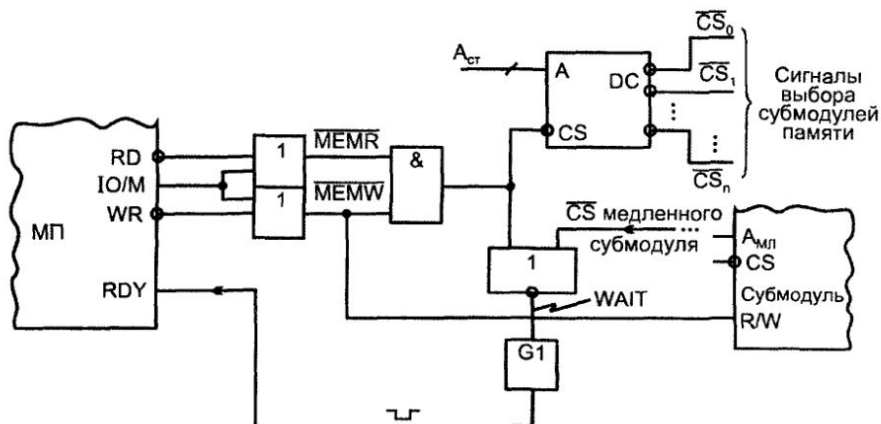


Рис. 5.17. Пример схемы выработки сигналов управления для тактируемого статического ЗУ

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.

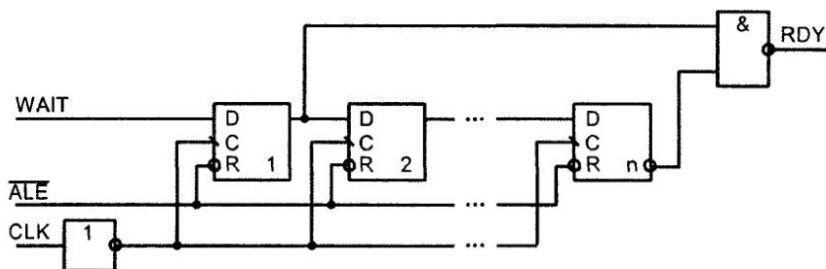


Рис. 5.18. Пример схемы генерации сигнала неготовности при работе процессора с памятью малого быстродействия

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. § 1.6).

Согласование временных диаграмм МП и ЗУ

Имея уточненную по результатам анализа нагрузок схему, можно рассмотреть задачу согласования временных диаграмм МП (точнее, системной шины) и 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.

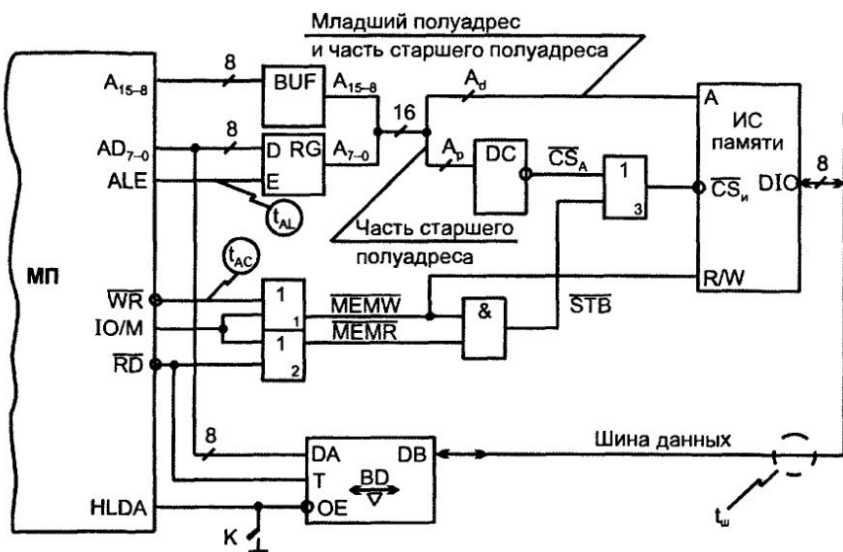


Рис. 5.19. Схема трактов передачи сигналов при управлении памятью

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:

- часть старшего полуадреса поступает на ЗУ через время t_{BUF} задержки буфера (другая часть старшего полуадреса поступает на дешифратор выработки сигналов \overline{CS}_A);
- 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;
- через время t_A после поступления адреса ЗУ вырабатывает выходные данные (t_A — характеристика ЗУ по ТУ);
- 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 ноль в соответствующих разрядах). Поэтому в дальнейшем задержку $t_{Ш}$ учитывать не будем.

Таким образом, на основании сказанного должно соблюдаться соотношение¹:

$$t_{AL} + t_{RG} + t_A + t_{BD} \leq t_{AD}.$$

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

¹ 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.

ожидания. Иначе требуется ввести столько тактов ожидания, сколько требуется для удовлетворения неравенства.

Из полученного неравенства следует условие, предъявляемое к параметру t_A :

$$t_A \leq t_{AD}(N) - (t_{AL} + t_{RG} + t_{BD}).$$

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}_I 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 представления адреса и stroba чтения). По истечении времени задержки элемента ИЛИ на входе \overline{CS}_I 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:

$$t_{AC} + 2t_{или} + t_{и} + t_{CS} + t_{BD} \leq t_{AD},$$

на основании которого предъявляется требование к величине t_{CS} :

$$t_{CS} \leq t_{AD} - (t_{AC} + 2t_{или} + t_{и} + t_{BD}).$$

Разность времен появления сигналов \overline{CS}_I 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:

$$t_{SU(A-CS)CX} \geq t_{SU(A-CS)TY},$$

где $t_{SU(A-CS)TY}$ — параметр памяти.

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 stroba 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}_I будет снят после окончания сигнала \overline{RD} через время, равное суммарной задержке эле-

ментов ИЛИ с номером два, И и ИЛИ с номером три. На основании рисунка можно записать соотношение:

$$t_{\text{DIS}}(\text{CS}) \leq t_{\text{RA}} - (2t_{\text{ИЛИ}} + t_{\text{И}}).$$

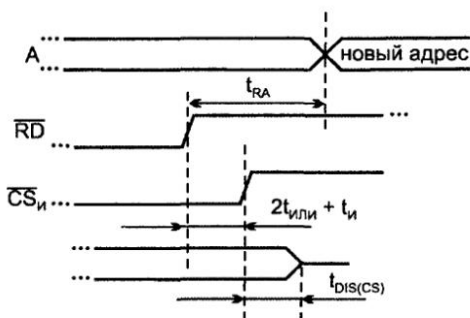


Рис. 5.20. Временные диаграммы сигналов для завершения цикла чтения из памяти

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 § 4.8*. 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).

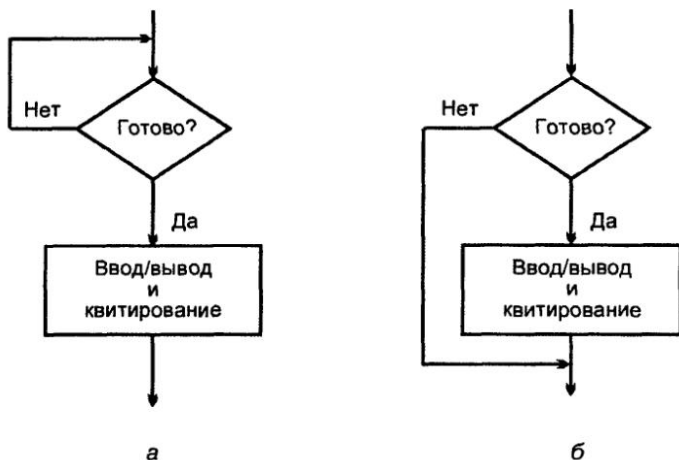


Рис. 5.21. Алгоритмы условного программного обмена с занятием цикла (а) и совмещенного (б)

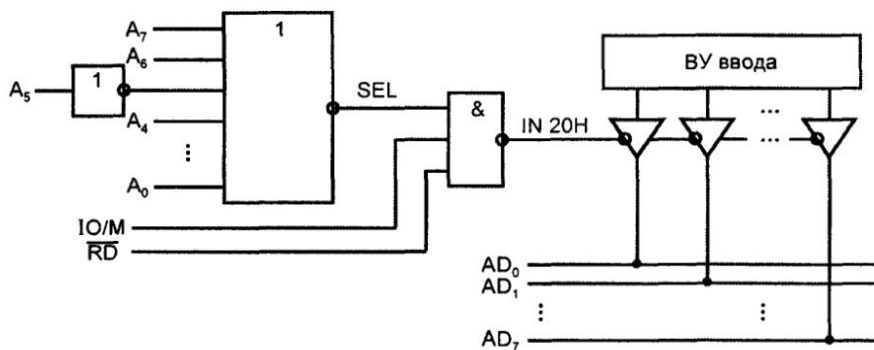
Потери времени на ожидание готовности исключаются при обмене по прерываниям, когда инициатором обмена является само ВУ при его готовности к обмену. *Общие вопросы обмена по прерываниям уже отмечались в § 5.2*, дополнительные сведения будут даны в дальнейшем при рассмотрении контроллеров прерывания.

Несмотря на отмеченные недостатки, программный обмен широко применяется вследствие его эффективности с точки зрения аппаратных затрат.

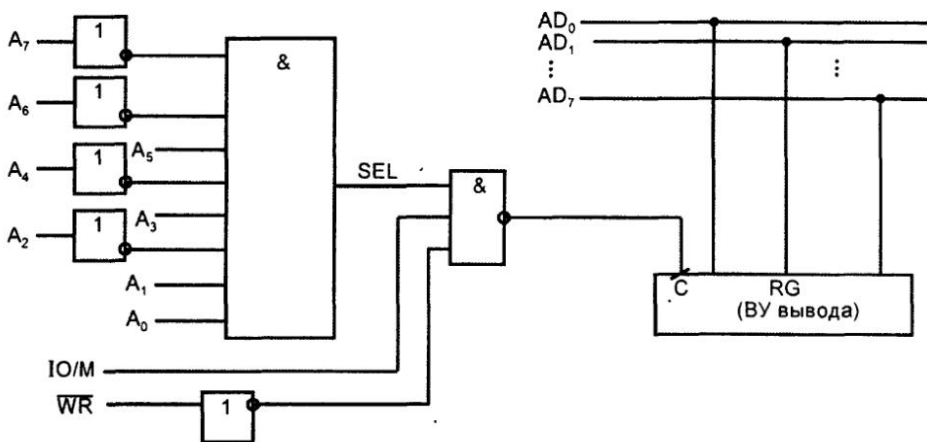
При вводе данных микропроцессор вырабатывает адрес ВУ A_{7-0} , сигнал $IO/M = 1$ и строб чтения \overline{RD} . В МПС с четверкой сигналов управления чтением/записью и вводом/выводом при вводе формируются адрес ВУ и строб чтения \overline{IOR} .

При использовании первого из указанных набора 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 $\overline{A_7A_6A_5A_4A_3A_2A_1A_0}$ или, что то же самое, при подаче на входы элемента ИЛИ-НЕ набора $A_7A_6\overline{A_5}A_4A_3A_2A_1A_0$. 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.



а



б

Рис. 5.22. Схемы реализации команд ввода (а) и вывода (б) при безусловном программном обмене

На рис. 5.22, б показана реализация команды OUT 2BH. В этом случае безусловный вывод осуществляется при поступлении от микропроцессора адреса $A_{7-0} = 00101100$, сигналов $IO/M = 1$ и $\overline{WR} = 0$. Поступая на конъюнктор с инверторами в соответствующих входных линиях, адрес ВУ формирует

единичный сигнал выбора \overline{BY} SEL, что в совокупности с единичными сигналами на двух других входах элемента И-НЕ дает отрицательный перепад напряжения на его выходе. 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.

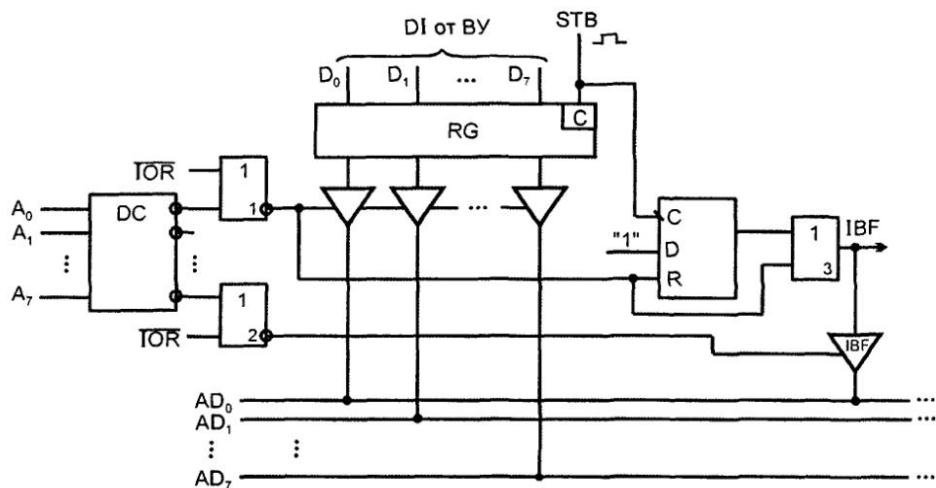
Вариант построения порта ввода с квитированием и "четверкой" управляющих сигналов в интерфейсе (рис. 5.23, а) предусматривает наличие регистра-защелки RG. Под блоком DC понимается схема декодирования адреса (не обязательно состоящая из одного дешифратора). ВУ готовит данные на линиях D_{7-0} и оповещает об их наличии сигналом STB, загружающим регистр, и своим задним фронтом, устанавливающим триггер, создавая этим $IBF = 1$ (IBF , Input Buffer Full). Этим фиксируется готовность ко вводу со стороны порта.

Микропроцессор начинает обращение к порту чтением 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 ИЛИ₂ 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 ИЛИ₁, 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.

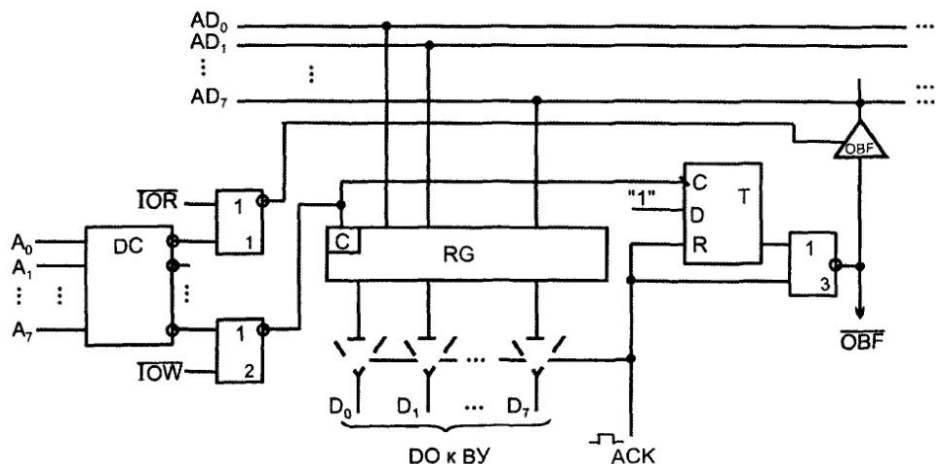
В программе описанный процесс ввода отображается следующей процедурой:

```
IA:   IN OFFH      ; Ввод слова состояния
      ANI 01H      ; Выделение бита
```

JZ IA ; Если IBF = 0, то ждать
IN 00H ; Иначе ввод данных



а



б

Рис. 5.23. Схемы реализации команд ввода (а) и вывода (б) при условном программном обмене

Условный вывод данных иллюстрируется схемой (рис. 5.23, б). Сначала вводится слово состояния с линий AD_{7-0} , на одной из которых действует сиг-

нал готовности \overline{OBF} (Output Buffer Full). Данные будут выводиться процессором в RG , готовность к выводу выражается в том, что данные регистра уже приняты VU . 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 (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 $\overline{OBF} = 1$ идет обмен, иначе — переход к циклам ожидания. Появление готовности вызывает запись в RG данных и установку триггера, дающего $\overline{OBF} = 0$, как сигнал для VU к использованию выводимых данных, после чего VU дает сигнал ACK , сбрасывающий триггер и выставяющий сигнал готовности порта к выводу.

Литература к главе: [2], [5], [7], [13], [17], [37], [45], [50], [57].

Глава 6

Интерфейсные БИС/СБИС микропроцессорных комплектов

§ 6.1. Интерфейсы микропроцессорных систем

С задачей обмена информацией между модулями МПС или другими блоками связано понятие стандартного интерфейса, т. е. совокупности средств, обеспечивающих совместимость модулей или иных блоков.

Аспектами стандартизации интерфейса являются функциональная, электрическая и механическая совместимости.

Функциональная совместимость модулей требует выработки определенных управляющих сигналов, генерируемых обменивающимися модулями, имеющих заданное смысловое значение и временное положение.

Электрическая совместимость обеспечивается определенными уровнями сигналов, их 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 интерфейсы и др.), способу передачи информации (параллельные, последовательные и др.), режиму передачи данных (дуплексный, полудуплексный и симплексный), способу обмена (асинхронные и 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 БИС, называют *системным (внутренним)*.

Интерфейс (шина) Microbus

Интерфейс Microbus был разработан в конце 70-х годов для построения систем на основе 8-разрядных микропроцессоров Intel 8080, Motorola 6800 и др.

Он является системным, однопроцессорным, магистральным, параллельным, асинхронным интерфейсом с полудуплексной (двусторонней поочередной) передачей данных. Интерфейс получил широкое распространение при объединении в систему не более 10 подключаемых к магистрали ИС, расположенных в непосредственной близости друг от друга. Для этого интерфейса разработан ряд интерфейсных БИС (комплектов K580, K589 и др.).

В функциональном аспекте интерфейс задается набором линий (сигналов), обеспечивающих обмен информацией между модулями, и временными параметрами (длительностями сигналов и их взаимным расположением во времени).

Интерфейс Microbus имеет 36 линий, в числе которых 16-разрядная шина адреса, 8-разрядная шина данных и следующие линии для управляющих сигналов: \overline{MEMR} , \overline{MEMW} , \overline{IOR} , \overline{IOW} , RDY, INT, \overline{INTA} , HOLD, \overline{HLDA} , CLK, RESET, \overline{BUSEN} .

Эти сигналы рассматривались при описании микропроцессора Intel 8085A и не нуждаются в дополнительных пояснениях. Исключение составляет сигнал \overline{BUSEN} . Этот сигнал поступает от контроллера прямого доступа к памяти при захвате им шин МПС и для подстраховки блокирует выходы шин микропроцессора с тремя состояниями (типа TC). При построении систем может и не использоваться.

В интерфейсе адресные пространства памяти и ВУ разделены, выполняются протоколы адресного (программного) обмена, обмена по прерываниям и прямого доступа к памяти.

В сведениях об интерфейсе приводятся также временные характеристики сигналов для циклов адресного обмена и др.

Интерфейс И-41

Позднее был разработан интерфейс фирмы Intel Multibus и на его основе отечественный интерфейс И-41. Этот интерфейс является многомашинным, системным, магистральным, параллельным, полудуплексным. 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 (одна из ее линий передает признак двухбайтной передачи), 16-разрядная шина данных и две линии контроля каждого байта на четность, 8-разрядная шина управления адресным (программным) обменом, 9-разрядная шина прерываний, 7-разрядная шина управления интерфейсом, 10-разрядная вспомогательная шина и шина источников питания. На интерфейсе И-41 заданы протоколы:

□ адресного обмена (с возможным запретом обращения);

- ☐ арбитража запросов задатчиков на управление магистралью и смены задатчика;
- ☐ обработки прерываний;
- ☐ аварии в системе электропитания.

Интерфейс МПИ

Интерфейс МПИ (на основе Q-bus) — магистральный, параллельный, полудуплексный, 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 (интерфейс "с общей шиной") и может составлять 64 К (16-разрядный адрес) или 16 М (24-разрядный адрес). Формат данных — байт или два байта. Для адресации ВУ отводится 8К в конце АП.

Мультиплексирование адресов и данных снижает пропускную способность интерфейса, но значительно уменьшает число линий связи, упрощая и удешевляя шину.

С ростом разрядности и быстродействия процессоров изменялись и соответствующие характеристики интерфейсов.

Появление ПЭВМ IBM PC/AT ассоциируется с применением интерфейса (шины) ISA, 32-разрядных процессоров 80386 и т. д. — с шиной EISA (Extended ISA) или MCA (микроканал). На уровне локальных шин сейчас широко применяется шина PCI (фирмы Intel), известна шина VL-bus и др.

Тактовая частота современных системных шин составляет 66...133 МГц.

Уже в первые годы развития техники интерфейсов фирма Intel разработала ряд БИС, предназначенных для реализации системных шин. В маркировке этих микросхем первыми были цифры 82, после которых стояли еще две цифры, обозначающие конкретный тип интерфейсной схемы. Простейшими микросхемами были шинные формирователи и порты (буферные регистры), более сложные операции обслуживались адаптерами и контроллерами. В ходе последующего развития интерфейсные схемы (схемы системной поддержки) претерпели ряд изменений, связанных с совершенствованием схемотехнологии ИС. Сейчас уровень интеграции ИС позволяет на одном кристалле объединить целый ряд устройств, которые ранее выполнялись в виде отдельных микросхем. Микросхемы с набором различных интерфейсных устройств, тем не менее, в структурном плане до сих пор базируются на "простых" ИС типа 82XX. Например, о современном периферийном контроллере 82C206 сказано: содержит две ИС 8259, две ИС 8237, одну ИС 8254 и др., где перечисленные ИС представляют собою давно разработанные

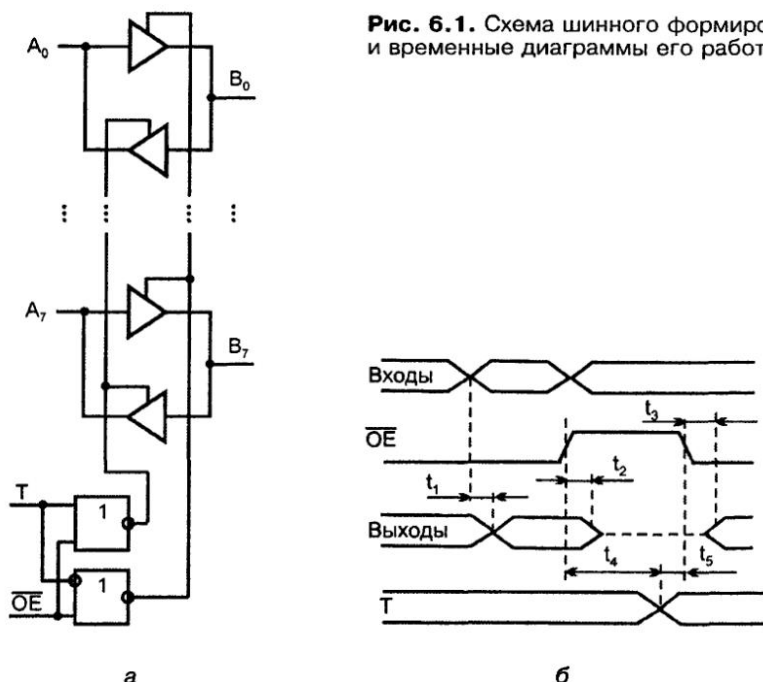
структуры типа 82XX. Более того, даже в библиотеках схемных решений новейших СБИС программируемой логики структуры традиционных интерфейсных схем используются в качестве макрофункций. Таким образом, рассматриваемые ниже адаптеры и контроллеры имеют как бы три лица: отдельных микросхем, частей более сложных кристаллов и макрофункций библиотек СБИС программируемой логики.

§ 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 шиной. Они усиливают сигналы по мощности при работе на шину, отключают источник информации от шины, когда он не участвует в обмене, формируют при необходимости требуемые уровни сигналов логической 1 или 0. Двухнаправленные ШФ позволяют в зависимости от сигнала управления передавать сигналы в шину или, напротив, принимать их с шины и передавать приемнику данных.

Рис. 6.1. Схема шинного формирователя K580BA86 (а) и временные диаграммы его работы (б)



Различные ШФ отличаются не только разрядностью, но и передачей сигналов в прямом или инвертированном виде (ШФИ), а также прямыми или инверсными сигналами разрешения работы. Отличаются они и электрическими характеристиками.

В серии КР580 имеются ШФ ВА86 и ШФИ ВА87 — аналоги микросхем 8286 и 8287, схема первого показана на рис. 6.1, а.

Шина А (линии A_{0-7}) принимает данные от МП или передает их ему, шина В (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 Т (Transmit). Функционирование ШФ подчиняется условиям, указанным в табл. 6.1.

Таблица 6.1

\overline{OE}	Т	Режим
1	0	Нет передачи
1	1	Нет передачи
0	1	Передача от А к В
0	0	Передача от В к А

Так как шина А связана с МП, а шина В — с магистралью, для них предусмотрена разная нагрузочная способность: выходы В обеспечивают токи 32 мА и -5 мА (при высоком и низком уровнях выходного напряжения соответственно), выходы А обеспечивают токи 16 мА и -1 мА.

Уровни выходного напряжения $\geq 2,4$ В и $\leq 0,5$ В, требуемые уровни входного напряжения $\geq 2,0$ В и $\leq 0,8$ В.

На временных диаграммах (см. рис. 6.1, б) показаны задержки сигналов при их распространении через открытые ШФ и относительно изменений управляющих сигналов. Первая задержка для ШФ составляет 30 нс, для ШФИ 22 нс, задержка t_2 перехода выходов в состояние "отключено" не превышает 18 нс, задержка t_3 переходов от состояния "отключено" к активным состояниям не более 30 нс. Времена выдержки t_4 и предустановки t_5 сигнала относительно моментов изменения сигнала \overline{OE} составляют соответственно не менее t_2 и не менее 30 нс.

ШФ выполняются на элементах ТТЛШ. Приведенные временные параметры даны для максимальных нагрузочных токов и емкостей 300 пФ (для выходов В) и 100 пФ (для выходов А).

Шинные формирователи широко представлены в сериях цифровых элементов. Кроме рассмотренных выше, можно указать ШФ серий К589, К555, КР1533, КР1554 и др.

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;
- задержки при питании 4,5 В: распространения сигнала ≤ 6 нс, выхода из ТС в активное состояние $\leq 6,5$ нс, перехода из активного состояния в ТС $\leq 8,5$ нс.

Буферные регистры

Буферные регистры служат для подключения к магистрали внешнего устройства. В отличие от ШФ, буферные регистры 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 МПК К580 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. Численно эти задержки не превышают 30, 18 и 30 нс

соответственно. Задержка t_4 от момента изменения stroba до изменения выхода схемы не более 45 нс. Время t_5 предустановки сигнала на входе относительно спада stroba не лимитировано, время выдержки входного сигнала относительно спада stroba $t_6 \geq 25$ нс.

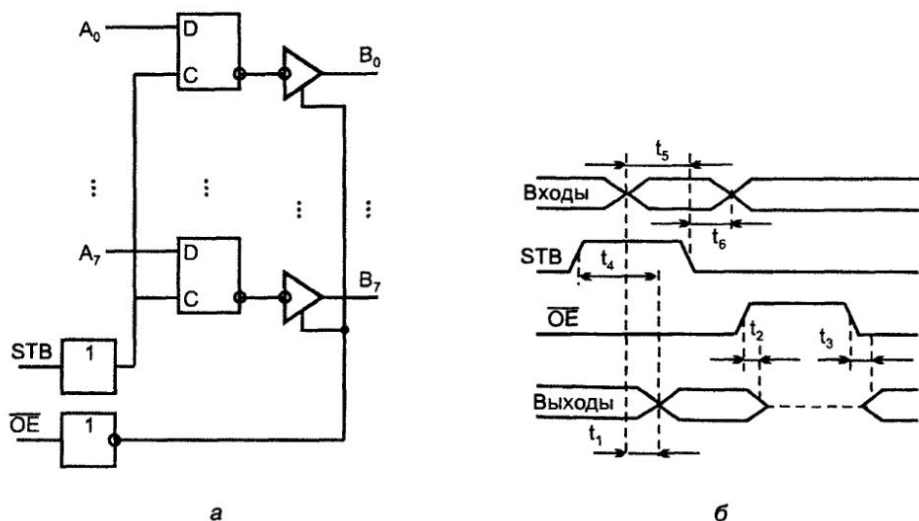


Рис. 6.2. Схема буферного регистра K580IP82 (а) и временные диаграммы его работы (б)

Буферный регистр IP83 отличается от порта IP82 тем, что инвертирует передаваемые данные. Его параметры совпадают с параметрами порта IP82, отличия имеются только в задержках t_1 и t_4 , которые для порта IP83 равны максимально 22 и 40 нс.

Примером часто применяемого порта может служить также многоцелевой регистр K589IP12, описание которого дано, в частности, в [23].

Шинные формирователи и буферные регистры связывают, как правило, выходы МП с внешней средой, поскольку нагрузочная способность МП недостаточна.

Буферные регистры широко представлены в сериях ИС, в частности, тех, которые указаны выше для ШФ.

В серии КР1533 буферные регистры обеспечивают выходные токи 15...70 нс 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 периферийные адаптеры

Шинные формирователи и порты осуществляют лишь непосредственную или буферизованную во времени передачу данных между МП и шиной данных. Более сложные операции выполняются периферийными адаптерами. Программируемость адаптеров обеспечивает им широкую область применения вследствие изменяемости процедур обмена без изменений в схеме (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 адаптера Intel 8255A, имеющего отечественный аналог К580ВВ55А. Эти БИС представляют собою однокристалльные устройства параллельного ввода/вывода и обеспечивают двунаправленный обмен с квитированием или без него при программном обмене или обмене по прерываниям. С их помощью ВУ, работающие с параллельными кодами, связываются с магистралью системы.

Параллельный периферийный адаптер (ППА, PPI) типа 55А (рис. 6.3) имеет три двунаправленных 8-разрядных порта РА, РВ и РС, причем порт РС разделен на два четырехразрядных канала: старший РС_H и младший РС_L. Обмен информацией между каналами А, В, С и шиной данных МПС производится через буфер данных BD в соответствии с сигналами управления.

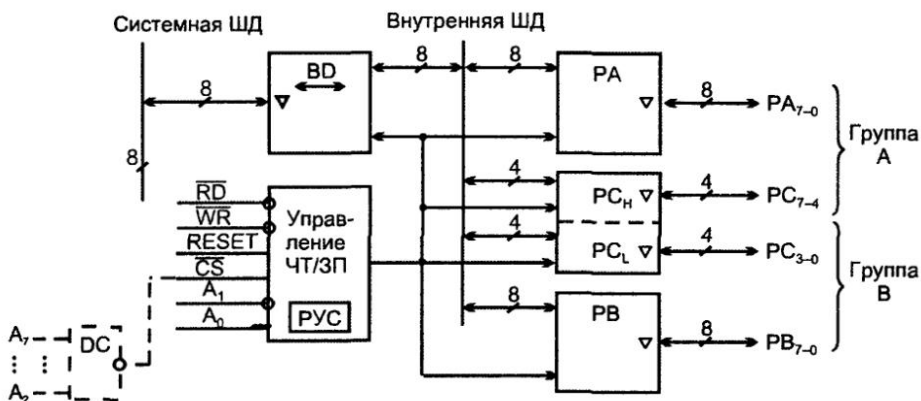


Рис. 6.3. Структура параллельного периферийного адаптера

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. Адресуемых объектов 5: три порта (А, В и С), регистр управляющего слова РУС и команда установки/сброса битов порта С 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 А → ШД
0	1	0	1	0	Порт В → ШД
1	0	0	1	0	Порт С → ШД
1	1	0	1	0	Запрещенная комбинация
0	0	1	0	0	ШД → Порт А
0	1	1	0	0	ШД → Порт В
1	0	1	0	0	ШД → Порт С
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 чтение по этому адресу запрещено, 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 слов 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 А может работать в любом из трех режимов, порт В только в двух (0 и 1), а режим порта С зависит от режимов портов А и В.

Порт С имеет особенности, в отличие от портов А и В, которые оперируют со словами в целом, разряды порта С могут программироваться и использо-

ваться поодиночке. В частности, любой из восьми разрядов порта С может быть установлен или сброшен программным способом. Это нужно для передачи сигналов квитирования при обмене через порты А и В в режимах 1 и 2. При работе порта в режиме 1 для него требуются три линии под сигналы управления, в режиме 2 — пять.

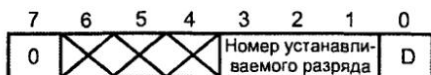
Режимы работы портов:

- ☐ режим 0 — однонаправленный ввод/вывод без квитирования, в этом режиме могут работать порты А и В, а также свободные (не занятые передачей служебных сигналов для портов А и В) линии порта С;
- ☐ 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.



а



б

Рис. 6.4. Форматы управляющих слов параллельного периферийного адаптера

Формат управляющего слова УС1 показан на рис. 6.4, а. Разряд 7 содержит единицу, что является признаком управляющего слова УС1. Разряды 6–3 определяют режим и вид портов А и свободных от служебных сигналов линий порта С_н (старшей половины порта), а разряды 2–0 — то же для порта В и младшей половины порта С (С_л).

Режим порта А выбирается по условиям: 00 — режим 0, 01 — режим 1, 1X — режим 2. Порт В 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. 6.4, 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.

Режим 0

В режиме 0 осуществляется прямой однонаправленный ввод-вывод данных без сигналов их сопровождения. Каждый из 4-х 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").

Режим 1

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{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. 6.5, а.

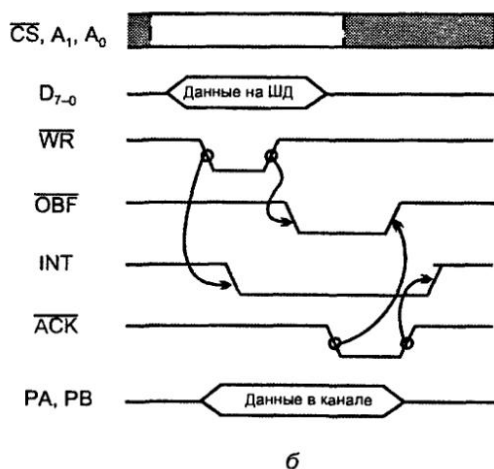
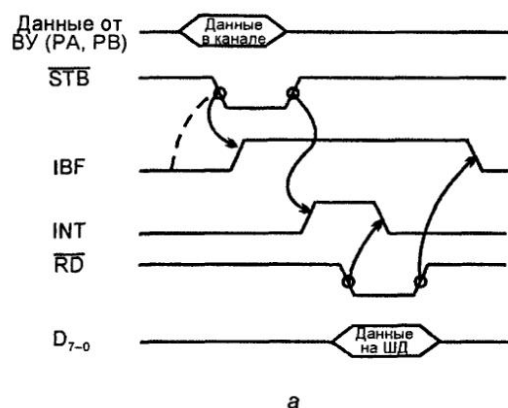


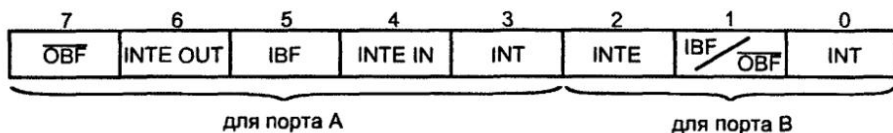
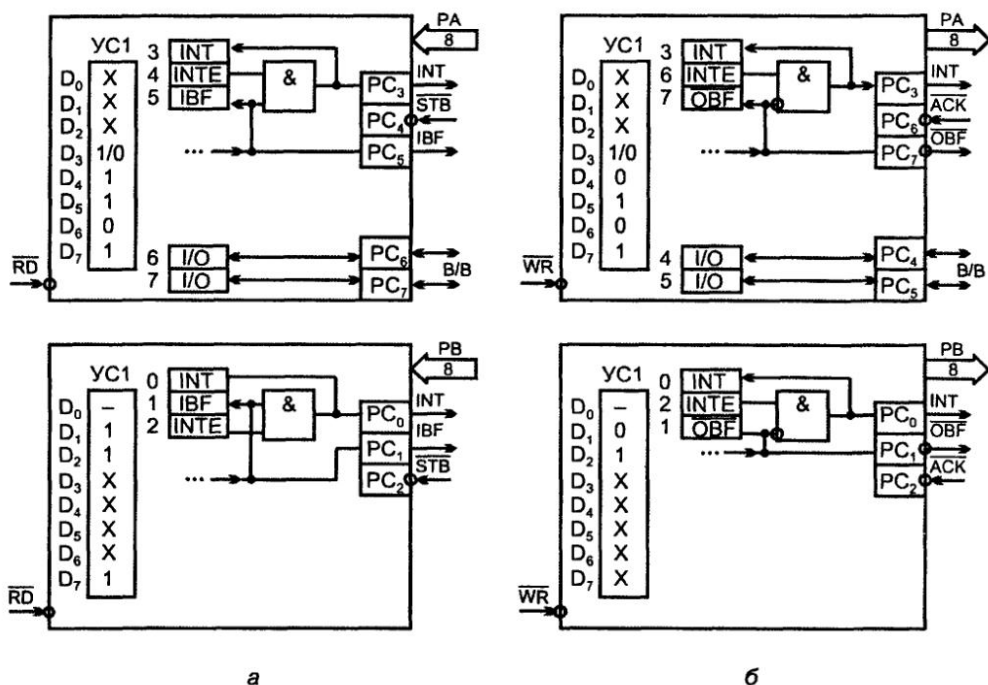
Рис. 6.5. Временные диаграммы процессов ввода (а) и вывода (б) в режиме 1 параллельного периферийного адаптера

Имея данные для ввода в порт, ВУ при условии $IBF = 0$ вырабатывает сигнал готовности информации STB . Передний фронт этого сигнала устанавливает сигнал IBF , запрещающий внешнему устройству ввод следующего слова до освобождения адаптера (того порта, который имеется в виду). К моменту окончания 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{RD} . Передний фронт \overline{RD} отмечает начало считывания слова микропроцессором

и снимает запрос на прерывание INT. Пока прерывания не разрешены, осуществляется хранение данных в адаптере. Задний фронт RD отмечает завершение считывания слова микропроцессором и снимает сигнал IBF, допуская новую запись слова со стороны ВУ.

При выводе используются следующие управляющие сигналы:

- $\overline{\text{OBF}}$ (Output Buffer Full) — выходной буфер полон, строб вывода новых данных;
- ACK (Acknowledge) — подтверждение приема внешним устройством;
- INT — запрос прерывания.



в

Рис. 6.6. Структура порта С, формирование слова состояния для режимов 1 параллельного периферийного адаптера (а, б) и формат слова состояния (в)

Временные диаграммы вывода в режиме 1 показаны на рис. 6.5, б. При выводе выполняется команда **OUT port**, и процессор устанавливает адрес порта и данные на ШД. При разрешенных прерываниях далее вырабатывается сигнал **WR**, загружающий данные с ШД в буфер адаптера и сбрасывающий запрос прерывания **INT**. После окончания записи в адаптер формируется сигнал **ОВФ**, указывающий на готовность данных для ВУ. Приняв данные, ВУ выдает сигнал подтверждения приема **АСК**, снимающий **ОВФ**, а по окончании сигнала **АСК** 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 и формирование в адаптере слова состояния для ввода и вывода в режиме 1 показано на рис. 6.6, а, б.

К битам **INT** и **IBF** предусмотрен программный доступ через чтение порта С. В состав слова состояния входят также флажки **INTE**, управляемые командой сброса/установки битов порта С. Генерация запроса прерывания **INT** и установка связанного с ним одноименного флажка готовности в слове состояния возможны только при установленном флажке **INTE**. Это означает возможность маскирования запросов прерывания, позволяющего запрещать или разрешать работу ВУ.

Формат слова состояния показан на рис. 6.6, в. 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.

Свободные линии порта С могут быть использованы для простого ввода/вывода.

Режим 2

Особенности функциональной схемы порта А допускают его применение для двунаправленной передачи между ШД и ВУ. 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, некоторые из которых могут совмещаться во времени. Используются сигналы управления: **STB**, **IBF**, **ОВФ**, **АСК**, **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 UC1 или UC2, 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 нужны значения.

Подробное описание рассмотренного ППА имеется в работах [23], [37], [41], [45] и др.

Улучшенный вариант адаптера BB55A отличается от предшественника BB55 работой с расширенным стробом записи, свойственным, в частности, и микропроцессору K1821BM85A.

Для связи с периферийными устройствами, удаленными от МПС (на расстояние не более 15 м), применяется интерфейс ИРПР (интерфейс радиальный параллельный), осуществляющий однонаправленные асинхронные передачи по 8- или 16-разрядной шине (в базовом варианте). Логические требования интерфейса ИРПР могут быть выполнены при использовании адаптера BB55/55A.

Пример применения ППА

На рис. 6.7 дан пример использования ППА в схеме подключения аналого-цифрового преобразователя (АЦП) и цифро-аналогового преобразователя (ЦАП) к МПС, выполняющей задачу цифрового управления некоторым аналоговым объектом.

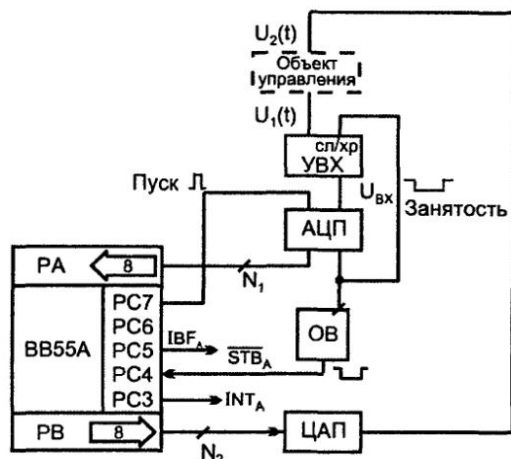


Рис. 6.7. Пример схемы использования параллельного периферийного адаптера для подключения АЦП и ЦАП к шинам микропроцессорной системы

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 PA. Строб загружает данные (код N_1) в порт A адаптера, при этом, как известно, формируются сигналы IBF_A и INT_A . Сигнал INT_A является запросом процессору на ввод байта из порта A.

Процессор выполняет подпрограмму обслуживания запроса и вводит код N_1 командой $IN\ port$.

Выработанный процессором N_2 выводится через порт B 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 A работает как порт ввода с квитированием, а порт B как порт прямого вывода, что задает следующие значения битам УС1: $D_7 = 1$ (признак УС1), $D_6D_5 = 01$ (режим 1 порта A), $D_4 = 1$ (ввод для порта A), $D_3 = 0$ (эта линия выделена для вывода сигнала "Пуск"), $D_2 = 0$ (режим 0 для порта B), $D_1 = 0$ (вывод для порта B), $D_0 = 0$ (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 A, 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 A адрес $F0H$, порту B — $F1H$, порту C — $F2H$ 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 установки/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. Программируемые связные адаптеры

При увеличении расстояний, на которые передаются данные, параллельные связи становятся неприемлемо сложными и дорогими. В этом случае применяют преобразование параллельных данных в последовательные для их передачи по одной сигнальной линии. 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 применении обычных телефонных сетей для связи удаленных объектов, что широко распространено в практике.

Тракт передачи последовательных данных в общем случае включает в себя источник и приемник данных, программируемые связные адаптеры (ПСА) и модемы (рис. 6.8, а). Такой тракт соответствует взаимодействию процессора с ВУ, оперирующими параллельными кодами, но находящимися на большом расстоянии от процессора.

ПСА преобразуют данные из параллельной формы в последовательную или наоборот и выполняют также некоторые другие функции.

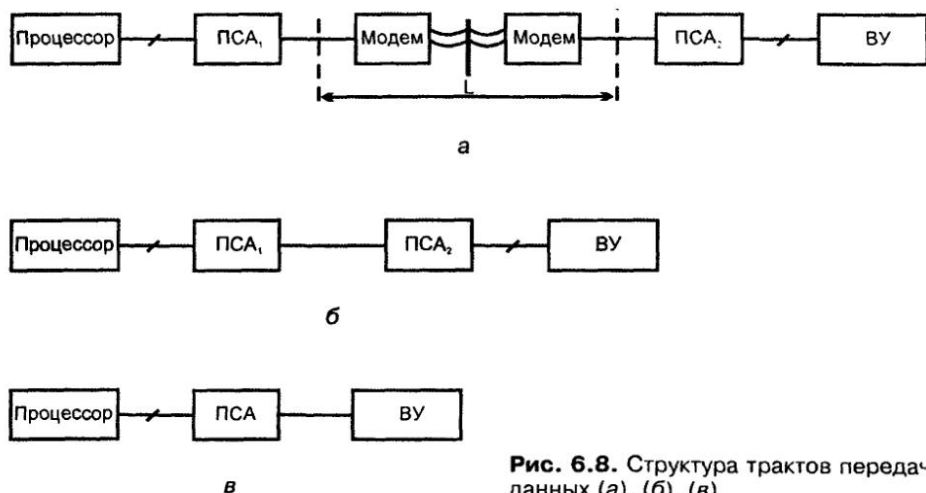


Рис. 6.8. Структура трактов передачи данных (а), (б), (в)

Модемы (модуляторы-демодуляторы) преобразуют двоичные импульсные сигналы (последовательности нулей и единиц) в некоторый аналоговый модулированный сигнал, приспособленный к передаче по узкополосным телефонным линиям. Узкополосность телефонных линий (полоса пропускания около 3 кГц) ограничивает их *бодовую скорость*.

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 ноль и единица, битовая скорость передачи, измеряемая в битах/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.

Показанный на рис. 6.8, а тракт передачи является наиболее полным. 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. 6.8, б), где ПСА₁ преобразует параллельные данные в последовательные, а ПСА₂ — последовательные в параллельные. Если требуется взаимодействие процессора с относительно недалеко расположенным ВУ, оперирующим с последовательными кодами, тракт передачи будет иметь вид (рис. 6.8, 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 7 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 придется 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.

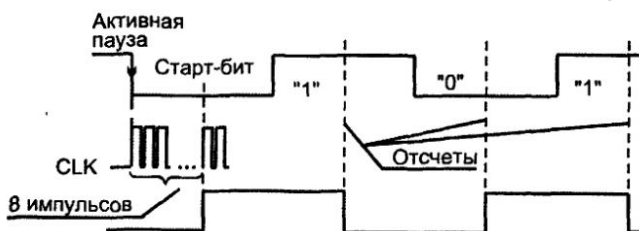
При синхронной передаче символы следуют один за другим слитно, поэтому можно говорить о передаче массива символов — текста. Если очередной символ не готов, передача не останавливается, передатчик посылает в линию специальные символы синхронизации, до тех пор пока не сможет передать следующий символ данных. Синхронный обмен повышает скорость передачи данных.

Скорость передачи оценивается числом передаваемых в секунду битов. Скорость асинхронной передачи обычно соответствует типовому периферийному оборудованию.

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 битов (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. 6.9, а).



а



б

Рис. 6.9. 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), 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 наименее влияют

на величину считываемого уровня. Это требование достигается следующим образом. *Передачик 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).

Частота 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. 6.9, 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. Оба указанных триггера (флажка) — разряды внутреннего регистра состояния ПСА.

Принятый символ поступает в регистр хранения, находящийся в буфере ШД, 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. Может в это время пойти третий символ, тогда второй 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 обеспечивает считывание слова за интервал, меньший, чем интервал вдвигания 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 возобновится 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.

В адаптере имеются регистры, хранящие назначенные для данного приемника коды синхронизации (регистры PCC_1 и PCC_2).

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).

Структура ПСА

На рис. 6.10 показана структура ПСА (PCI, Programmable Communication Interface) типа 8251A фирмы Intel, аналогом которого является отечественный ПСА K580BB51A. 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 используется 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.

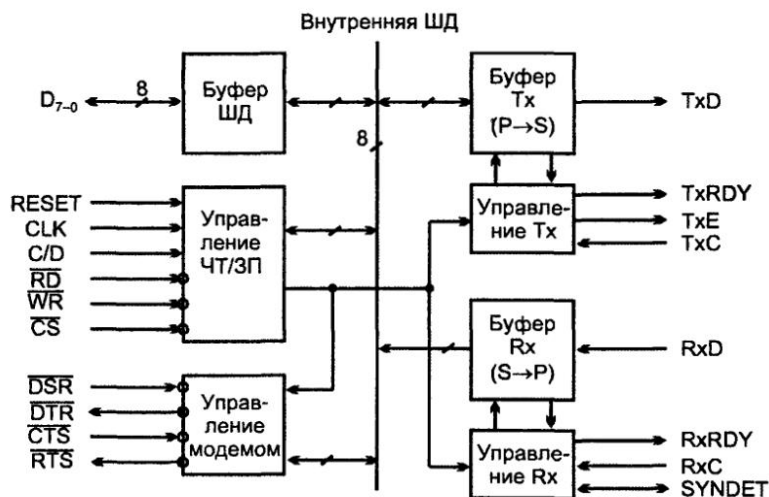


Рис. 6.10. Структура программируемого связного адаптера

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

Выводы и сигналы ПСА имеют следующее назначение:

- ❑ **RESET** — установка адаптера в исходное состояние, после него адаптер находится в бездействии до записи нового набора управляющих слов для определения задаваемых ему функций. В состояние бездействия адаптер вводится также программой по команде сброса;
- ❑ **CLK** — вход тактовой частоты для внутреннего тактирования процессора. Внешние входы и выходы адаптера не привязаны к тактам сигнала CLK, но частота этого сигнала должна быть выше битовой частоты передачи данных не менее чем в 30 раз;
- ❑ **RD**, **WR** и **CS** — сигналы, смысл которых уже 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. Вме-

сте 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 управления другими устройствами. Для управления модемом (терминалом) имеются две пары сигналов квитирования;

- \overline{DSR} (Data Set Ready) — запрос готовности 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 процессору;
- \overline{DTR} (Data Terminal Ready) — этот сигнал является реакцией на запрос \overline{DSR} . Активируется соответствующим битом командного слова, если процессором разрешен обмен с модемом. Связан с разрешением модему посылки 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.

Передачик ПСА

Буфер передатчика адаптера (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 необходимым образом поток битов на вывод TxD по отрицательным фронтам импульсов частоты TxС. Передача начинается после ее разрешения и при условии $\overline{\text{CTS}} = 0$. Вывод TxD принимает высокий уровень напряжения после сброса, запретов по условиям TxEN или $\overline{\text{CTS}}$ либо при условии "передатчик пуст", связанном с сигналом TxЕ (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 stroba записи $\overline{\text{WR}}$, когда символ данных загружается из процессора;
- TxЕ — 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 режима передачи и оповещения процессора о моменте переключения линии передачи 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;
- TxС и RxС — сигналы синхронизации передатчика и приемника, задающие скорость следования последовательных битов. При синхронных передачах базовая скорость равна частоте TxС (RxС), при асинхронных она является частью частоты TxС (RxС) (это 1, или 1/16 или 1/64 от TxС или RxС). Очень часто частоты TxС и RxС 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. Вывод RxD 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)

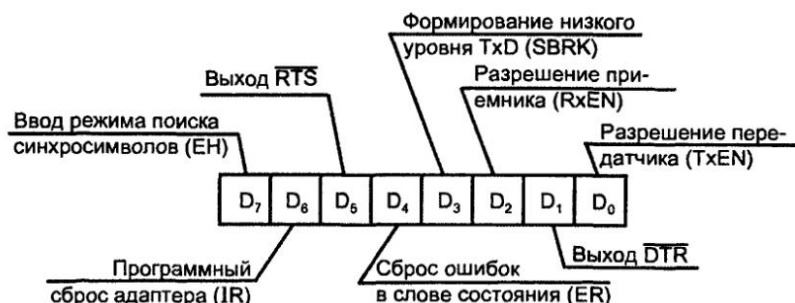
Программирование адаптера

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. 6.11. Как видно, трактовка разрядов D₇ и D₆ слова MI зависит от режима.

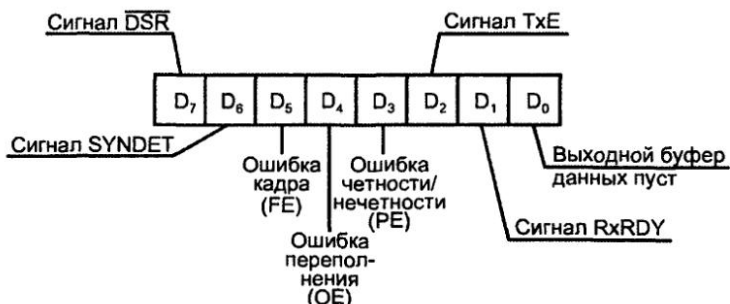
При синхронном обмене и внутренней синхронизации после инструкции режима MI в адаптер вводятся один или два синхросимвола, для хранения которых в схеме управления приемником имеются два специальных регистра



Рис. 6.11. Формат управляющего слова режима программируемого связного адаптера



а



б

Рис. 6.12. Форматы командного слова (а) и слова состояния адаптера (б)

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. 6.12, а. Правильная загрузка нескольких регистров без индивидуальных адресов у них обеспечивается жестким порядком записи управляющих слов в программируемый адаптер.

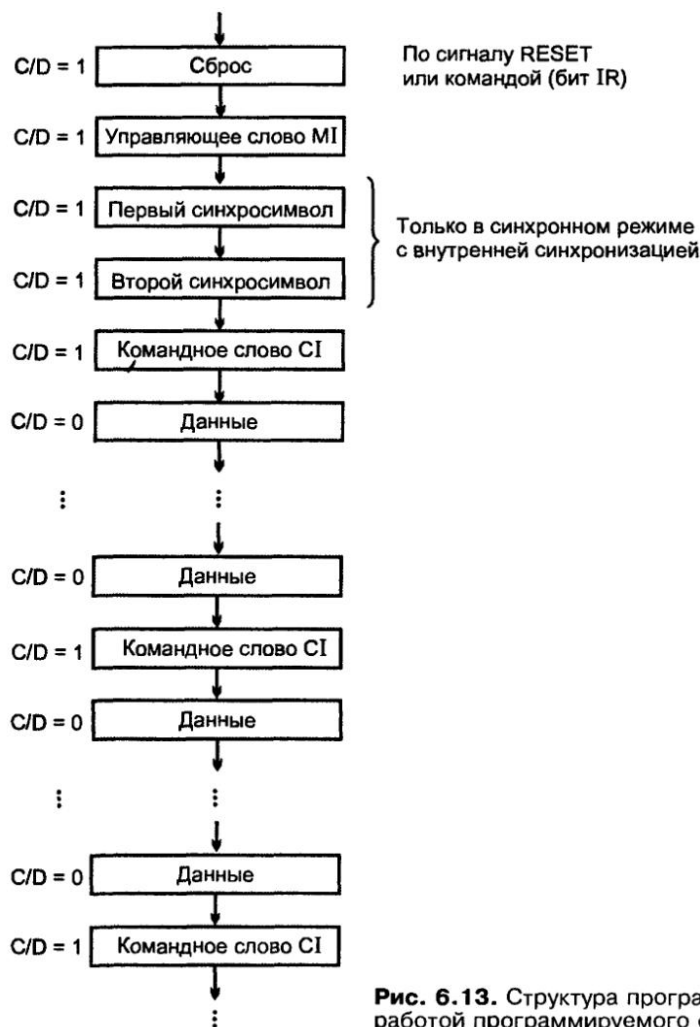


Рис. 6.13. Структура программного блока управления работой программируемого связного адаптера

Формат слова состояния адаптера представлен на рис. 6.12, б Структура программного блока, управляющего работой адаптера, приведена на рис. 6.13.

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 $\overline{CTS} = 0$, 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.

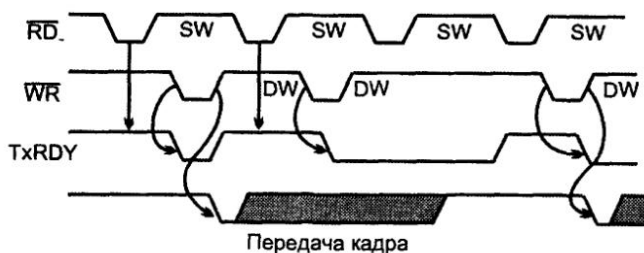


Рис. 6.14. Временные диаграммы программного условного обмена с помощью программируемого связанного адаптера

При программном условном обмене (рис. 6.14) 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.

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. 6.15) выдает на выходе низкий логический уровень, разрешающий работу адаптера, в ответ на одну-единственную комбинацию входных сигналов A_{7-1} . На вход CLK поданы синхриимпульсы 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 4 выхода с разными частотами. С помощью ключа К можно изменять скорость передачи-приема данных. Остальные соединения понятны без дополнительных пояснений.

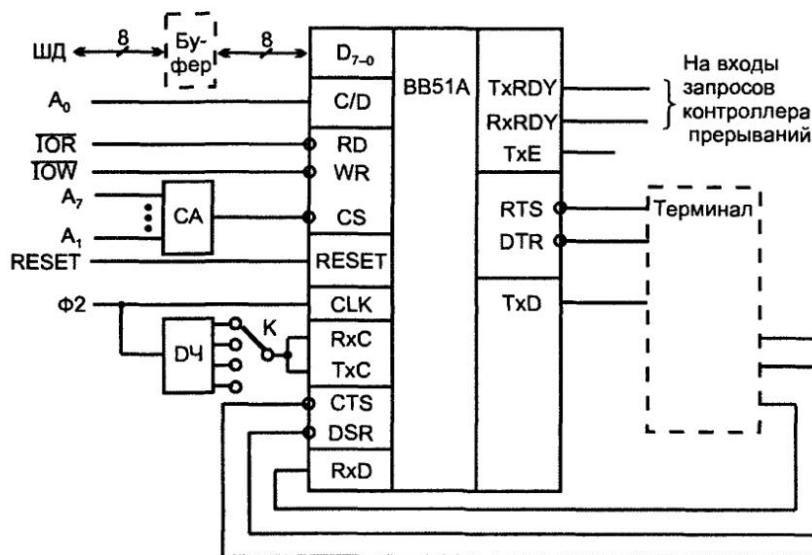


Рис. 6.15. Схема подключения программируемого связного адаптера к микропроцессору и терминалу

Последовательные порты персональных ЭВМ строятся на основе адаптеров типа UART, например, типа 16550 и 16550A. Эти адаптеры во многом подобны адаптеру BB51A, но имеют 16-символьные буферы FIFO, предназначенные для приема и передачи данных. Конструктивно они обычно входят в одну БИС с другими схемами.

§ 6.5. Программируемые контроллеры прерываний

При работе микропроцессорной системы в ней или внешней среде происходят события, требующие немедленной реакции, что обеспечивается прерыванием выполняемых программ и переходом к обслуживанию запросов прерывания. *Типы и характер запросов прерывания освещены в § 5.3.*

Аппаратно прерывания обслуживаются специализированными ИС, простейшими из которых являются блоки приоритетного прерывания (Intel 8214, K589ИК14 и др.). Эти блоки решают несложные задачи обработки нескольких векторных прерываний при фиксированных приоритетах запросов.

Более сложные задачи решаются программируемыми контроллерами прерываний (ПКП), в частности ИС Intel 8259A, K1810ВН59.

Эти контроллеры непосредственно реализуют прерывания с обработкой 8 запросов. С помощью нескольких ПКП легко организуются устройства обработки до 64 запросов. Контроллеры обеспечивают различные виды прерываний.

Система прерываний должна выдать команду перехода к той подпрограмме обслуживания, которая соответствует признанному запросу.

Контроллер ВН59 обеспечивает различные виды прерываний.

Вложенные прерывания с фиксированными приоритетами входов

Имеются 8 входов запроса прерывания $IR_7...IR_0$ (от английского Interrupt Request). Высший приоритет имеет вход IR_0 , низший — у входа 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. Возможность вложенных прерываний обеспечивается введением команды EI (Enable Interrupt) в подпрограммы обслуживания прерываний. Прерывания с фиксированными приоритетами реализуются просто, но запросы неравноправные 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 запросами.

Прерывания с круговым (циклическим) приоритетом

В этом случае у каждого входа тоже есть свой приоритет, но после обслуживания он изменяется в круговом порядке так, что обслуженный вход получает низший приоритет. Такая дисциплина обслуживания характерна для си-

туации с источниками, не имеющими преимуществ друг перед другом. Запрашивающее обслуживания устройство будет ждать в худшем случае до того, как 7 других источников будут обслужены по одному разу. Работу с круговым приоритетом можно иллюстрировать примером (рис. 6.16), в котором регистр запросов вначале содержит 6-й и 4-й запросы, т. е. наивысший приоритет имеет 4-й запрос, который и будет обслужен. После обслуживания приоритетность входов изменяется как бы вращением кольца, причем номер 7 с низшим приоритетом становится на 4-ю 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.

	IR ₇	IR ₆	IR ₅	IR ₄	IR ₃	IR ₂	IR ₁	IR ₀	Входы запросов
До обслуживания	0	1	0	1	0	0	0	0	Наличие запросов
	7	6	5	4	3	2	1	0	Уровни приоритета
	IR ₇	IR ₆	IR ₅	IR ₄	IR ₃	IR ₂	IR ₁	IR ₀	Входы запросов
После обслуживания	0	1	0	0	0	0	0	0	Наличие запросов
	2	1	0	7	6	5	4	3	Уровни приоритета

Рис. 6.16. Пример обслуживания запросов прерывания с круговым приоритетом

Кроме рассмотренного, имеется также режим адресуемого циклического приоритета, при котором дно кольца после обслуживания ставится в любое положение, определяемое программно, а не устанавливается автоматически на позицию обслуженного запроса.

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 показано на рис. 6.17. 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.

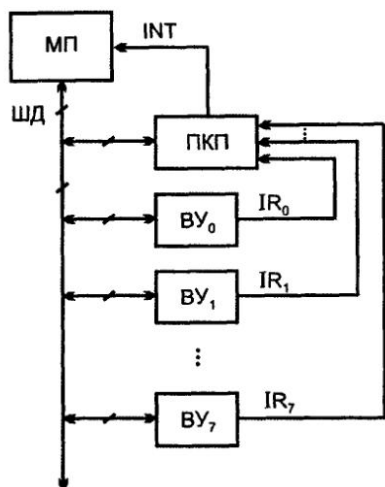


Рис. 6.17. Схема включения контроллера прерываний в микропроцессорную систему

Структура ПКП

Структура ПКП Intel 8259A представлена на рис. 6.18. В английской терминологии ПКП называют PIC, т. е. Programmable Interrupt Controller.

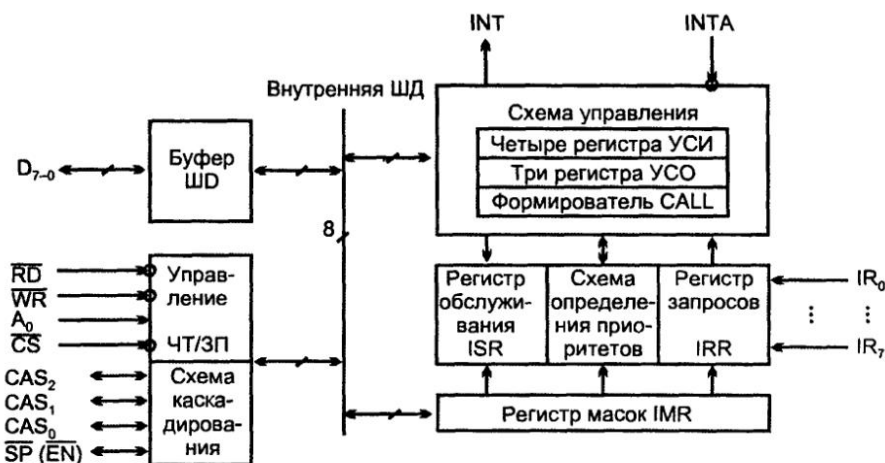


Рис. 6.18. Структура программируемого контроллера прерываний

Запросы прерываний от ВУ поступают на регистр запросов IRR, 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, так что маскирование может быть осуществлено 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{INTA} принятый запрос переходит в регистр обслуживания ISR (Interrupt Servicing Register) 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.

Можно также обслуживать прерывания по результатам опроса источников запросов, когда сигнал INT не используется, и процессор сам производит поочередный опрос входов, начиная со старшего по приоритету. Обнаружение запроса ведет к его обслуживанию с переходом на соответствующую подпрограмму.

Буфер ШД восьмиразрядный, двунаправленный, с третьим состоянием. При программировании контроллера через него передаются управляющие слова, и считывается состояние регистров, а также код запроса, выработавшего сигнал INT. При обслуживании прерывания по сигналу \overline{INTA} через буфер ШД в шину данных системы выдается трехбайтная команда вызова подпрограммы CALL.

Смысл сигналов \overline{RD} , \overline{WR} и \overline{CS} , ясен (совпадает со смыслом этих сигналов в описанных выше устройствах). Сигнал \overline{INTA} поступает от процессора в виде трех последовательных импульсов, для выдачи контроллером кода команды CALL, младшего байта адреса начала подпрограммы и старшего байта этого адреса. Первый импульс \overline{INTA} сбрасывает запрос в соответствующем бите IRR.

Сигналы $IR_0 \dots IR_7$ — входы запросов прерывания (Interrupt Requests), A_0 — младший разряд адреса, показывает, к какому регистру управляющих слов (УСИ или УСО) обращается процессор. Сигналы CAS_{2-0} связаны с работой контроллера в групповой схеме, образуют выходную шину для ведущего контроллера и входную для ведомых. Сигнал \overline{SP} (\overline{EN}) двухфункциональный, как \overline{SP} он определяет, является ли контроллер ведущим или ведомым в групповой схеме, как \overline{EN} используется в так называемом буферизованном режиме для разрешения выхода на шину системы, т. е. для управления выходными буферами участников обмена.

Программирование контроллера

Перед работой контроллер программируют засылкой в него управляющих слов инициализации УСИ (ICW, Initialization Control Words) и управляющих слов операций УСО (OCW, Operation Control Words).

Управляющие слова УСИ приводят контроллер в исходное состояние последовательностью из 2...4 слов (байтов), записываемых в контроллер строками \overline{WR} .

Для понимания формата управляющего слова УСИ1 рассмотрим порядок формирования адреса подпрограммы обслуживания какого-либо запроса. По второму импульсу \overline{INTA} должен сформироваться младший байт начального адреса подпрограммы. Интервал между начальными адресами подпрограмм принимается равным 4 или 8, поэтому младший байт адреса подпрограммы 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

При этом три первых разряда программируются и участвуют в задании расположения области подпрограмм обслуживания в адресном пространстве системы, а пять младших фиксированы и автоматически вводятся контроллером.

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

В этом случае программируются всего два старших разряда.

Формат управляющего слова УСИ1 показан на рис. 6.19 (для примера взят интервал 4). Разряды 7...5 содержат программируемую часть младшего байта адреса подпрограммы, единичное значение 4-го разряда вместе с условием $A_0 = 0$ служит признаком слова УСИ1, разряд 3 задает способ восприятия входных запросов $IR_7... IR_0$ (восприятие фронта или восприятие уровня, т. е. Level Triggered/Edge Triggered Interrupt Mode) 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 ли при программировании загрузка управляющего слова УСИ4, т. е. вводится ли нет буферизованный режим.



Рис. 6.19. Форматы управляющих слов инициализации программируемого контроллера прерываний

Формат УСИ2 для систем с процессорами типа 8080 и 8085A имеет вид, показанный также на рис. 6.19. УСИ2 содержит старший байт начального адреса области памяти для подпрограмм обслуживания прерываний. "Скачки" через 4 или 8 байтов для получения начальных адресов отдельных подпрограмм формируются самим контроллером.

Если несколько контроллеров работают совместно в групповой структуре, то загружается и УСИ3. Ведущему контроллеру это слово сообщает, какие его входы подключены к ведомым контроллерам, а каждому из ведомых — к какому входу ведущего подключен его выход запроса прерывания INT. Таким образом, формат УСИ3 отражает физическую схему соединения контроллеров.

Для буферизованного режима вводится УСИ4, в котором бит 4 отражает наличие или отсутствие так называемого специального режима вложенных прерываний, бит 3 устанавливает буферизованный режим ($BUF = 1$), бит 2 определяет характер контроллера в групповой структуре (ведущий или ведомый, т. е. Master/Slave), т. к. сигнал \overline{SP} уже не может быть использован для этой цели, поскольку вывод этого сигнала теперь используется под сигнал \overline{EN} , бит 1 зада-

ет обычный или автоматический конец прерываний, а значение бита 0 зависит от того, с каким МП работает система (для процессоров 8080 и 8085A этот бит должен быть нулевым).

Последовательность инициализации контроллера показана на рис. 6.20. В результате инициализации сбрасываются схемы приема запросов, управляемые фронтами, и регистры. Входу IR_7 присваивается низший приоритет, снимается специальное маскирование и чтение состояния устанавливается на IRR , если $IC4 = 0$, то все функции, задаваемые УСИ4, обнуляются.

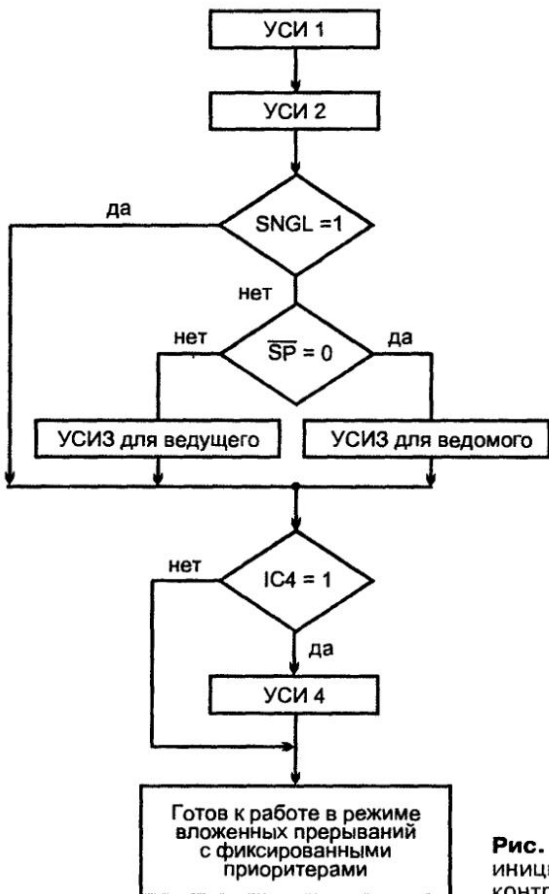


Рис. 6.20. Последовательность инициализации программируемого контроллера прерываний

После инициализации контроллер может работать в базовом режиме. Для выбора других режимов в контроллер загружаются управляющие слова операций УСО.

Слово УСО1 загружается после инициализации, с его помощью в любое время можно 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 операций, связанных с концом обслуживания прерывания и установкой "дна" приоритетного кольца. Каждая подпрограмма обслуживания прерывания должна сообщать контроллеру о своем завершении, передавая ему одно из УСО2, в котором задан характер конца обслуживания прерывания из числа следующих:

КП — конец прерываний, т. е. неадресуемый конец прерываний, заключающийся в сбросе бита ISR с максимальным приоритетом, который был обслужен;

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;

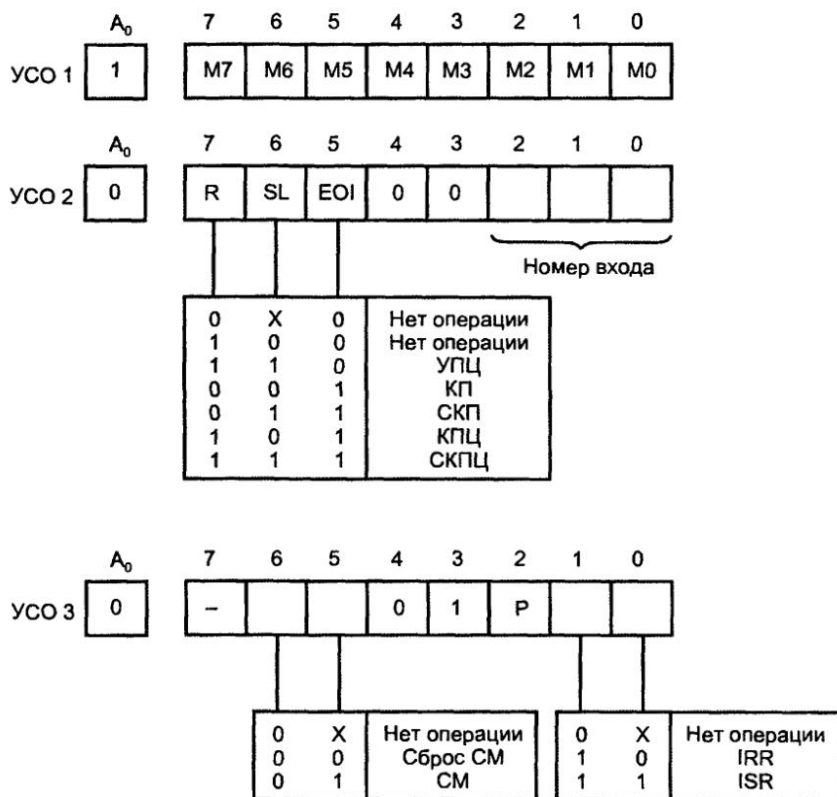


Рис. 6.21. Форматы управляющих слов операций программируемого контроллера прерываний

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).

Форматы управляющих слов УСО2 показаны на рис. 6.21.

Команды типа УСО3 (признак $A_0 = 0$, $D_3 = 1$, $D_4 = 0$) применяются в режиме чтения и при установке/снятии режима специального маскирования. Формат УСО3 показан на рис. 6.21.

Чтение состояния заключается в чтении регистров контроллера или кода старшего из поступивших запросов. Для чтения доступны IMR, IRR и ISR. Чтение регистра масок не требует предварительной загрузки УСО3. Если поданы сигналы $\overline{RD} = 0$ и $A_0 = 1$, то контроллер в любое время выдает на ШД содержимое IMR. 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 об источнике прерывания. Режим опроса инициируется выдачей в контроллер УСО3 с единичным значением бита P (Polling). Следующий цикл чтения при $A_0 = 0$ интерпретируется как подтверждение прерываний, и контроллер выставляет на ШД сведения об источнике с высшим приоритетом. Если запросов нет, то контроллер формирует слово ответа с нулевым значением бита P. При наличии запросов $P = 1$, а в разрядах $D_2...D_0$ записан код входа с высшим приоритетом. Режим поллинга применяется, в частности, если для нескольких запросов действует одна и та же подпрограмма обслуживания.

Режим специального маскирования имеет следующий смысл. В контроллере обслуживаемый запрос блокирует обработку запросов с меньшим приоритетом, даже если он временно замаскирован. С помощью УСО3 можно изменить ситуацию и разрешить обработку прерываний с меньшими приоритетами при сохранении маскирования данного бита ISR, т. е. установить режим, в котором каждый бит регистра ISR запрещает только собственный уровень, разрешая все остальные.

Каскадное включение контроллеров

Каскадное включение контроллеров расширяет число обрабатываемых запросов. Принцип такого расширения (рис. 6.22) — подключение к ведущему контроллеру ведомых (не более восьми). На рисунке показано подключение одного ведомого контроллера (остальные 7 могут быть подключены аналогичным способом). Подключение одного ведомого контроллера дает схему с 15 входами, максимально можно получить по данной методике схему с 64 входами.

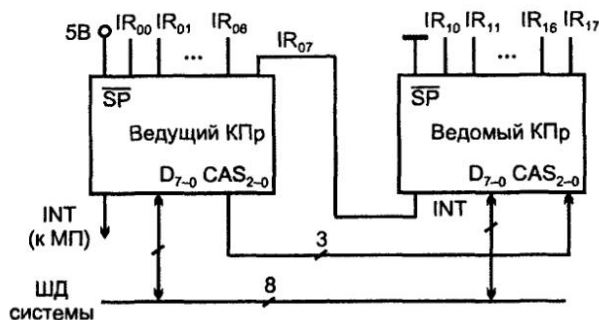


Рис. 6.22. Схема каскадного включения контроллеров прерываний

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 УСОЗ). 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 передачи между ВУ и памятью требуют вначале принять данные от источника в процессор, а затем выдать их из процессора приемнику, т. е. реализуются за два командных цикла. При ПДП данные не проходят через процессор, и передача слова производится за один цикл. ПДП особенно удобен при передачах блоков данных в высоком темпе, например при обме-

не 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 реализации ПДП разработаны специальные аппаратные средства, выпускаются БИС КПДП, способные благодаря программированию обслуживать ПДП с учетом конкретных требований различных систем.

Взаимодействие блоков микропроцессорной системы при ПДП показано на рис. 6.23. Микропроцессор выполняет операцию программирования КПДП, настраивая его на определенный режим работы, и может читать состояние контроллера. Соответствующие связи показаны штриховой линией. При осуществлении ПДП микропроцессор отключен, а контроллер вырабатывает сигналы управления обменом для ВУ и ОЗУ. Тракт передачи данных связывает ВУ с ОЗУ непосредственно.



Рис. 6.23. Схема взаимодействия блоков микропроцессорной системы при прямом доступе к памяти

Возможны *два вида ПДП* — с блочными или одиночными передачами. В первом работа процессора останавливается на все время передачи блока данных, во втором передачи слов в режиме ПДП перемежаются с выполнением программы, и для передач ПДП выделяются отдельные такты машинных циклов, в которых процессор не использует системные шины. Каждый командный цикл начинается с машинного цикла М1 — выборки команды. В этом машинном цикле есть такт декодирования принятой процессором команды, в котором системные шины не используются. На это время системные шины можно отдать для ПДП и передать одно слово. Производительность системы может возрасти из-за параллелизма процессов обмена и обработки данных, 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.

При непрерывной передаче массива данных 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, поэтому содержимое рабочих регистров МП не нарушается и на входение в режим ПДП не требуется затрат времени (нет передачи в стек на хранение содержимого рабочих регистров МП). ПДП предоставляется по завершении текущего машинного цикла.

Структура и функции КПДП

Примером КПДП может служить БИС Intel 8237A (K580BT57), основные блоки которой показаны на рис. 6.24.

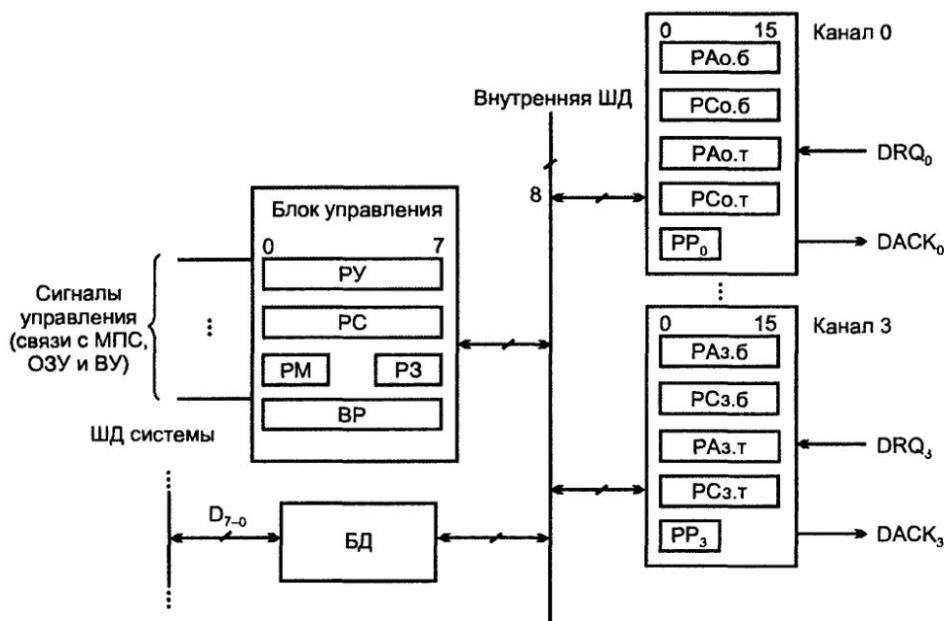


Рис. 6.24. Структура контроллера прямого доступа к памяти

Действия, выполняемые КПДП при блочных передачах, состоят в следующем:

- ❑ прием сведений об области памяти, отведенной для блока данных, подлежащих передаче (начальный адрес и размер блока);
- ❑ трансляция запроса на ПДП, исходящего от ВУ, в запрос ПДП для процессора с учетом маскирования и приоритетности запросов, поступающих на КПДП. Прием сигнала подтверждения ПДП, свидетельствующего о том, что процессор отключился от системных шин;
- ❑ генерация адресов для ЗУ и сигналов управления для ЗУ и ВУ;
- ❑ фиксация завершенности ПДП;

□ снятие запроса ПДП с соответствующего входа процессора и возвращение управления основной программе.

Возможности КПДП позволяют организовать 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 4 независимых канала и возможность каскадирования схем до любого числа каналов.

В каждом из каналов контроллера размещено по пять регистров, а именно: два регистра адреса (базовый $RA_{i,6}$ и текущий $RA_{i,T}$, где i — номер канала), два регистра счета слов (базовый $PC_{i,6}$ и текущий $PC_{i,T}$) и регистр режима RP_i . Адресные регистры и регистры счета слов шестнадцатиразрядные, следовательно, начальный адрес блока данных может располагаться в любом месте адресного пространства емкостью 64 К, а максимальный размер блока также 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 с ПДП.

Базовые регистры адреса и счета слов позволяют реализовать *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. 6.25, а.

В регистре режима два младших бита используются для загрузки слова в тот или иной канал, а шесть остальных определяют режимы, указанные на рисунке. В режиме "Контроль" нет передач между ВУ и памятью, но сигналы доступа к данным формируются. Это позволяет выполнять по отношению к информации операции контроля. В режиме обмена по требованию передачи выполняются до выработки контроллером признака конца счета или поступления внешнего сигнала EOP (End of Process) или до перехода сигнала DRQ в пассивное состояние, т. е. до истощения ВУ в смысле исчерпания его данных. При этом возобновление данных в

БУ 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.

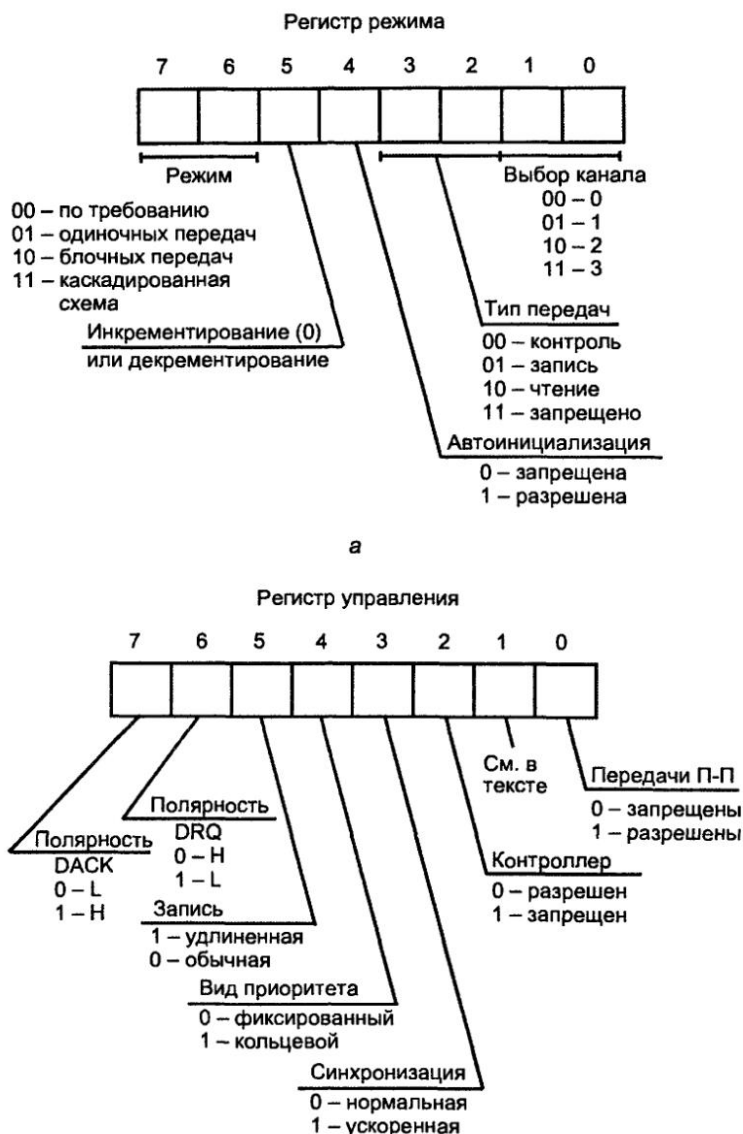


Рис. 6.25. Форматы регистров режима (а) и управления (б) контроллера прямого доступа к памяти

Восьмиразрядный *регистр управления* РУ программируется процессором, сбрасывается по сигналу RESET или командой Master Clear. Формат слова управления показан на рис. 6.25, 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 и при каждом чтении состояния. Четыре старших бита устанавливаются, если соответствующие каналы запрашивают обслуживание. Таким образом, РС позволяет определить, какие каналы закончили ПДП и какие требуют его.

Четырехразрядный *регистр масок* РМ имеет биты, соответствующие четырём 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, как и в микропроцессоре Intel 8085A, применено мультиплексирование шин для передачи старшего байта адреса через шину данных

Запрос незамаскированного канала порождает запрос прерывания HRQ для процессора, реализующего один из четырех вариантов ПДП, указанных 7 и 6 битами регистра режима.

Подробная схема взаимодействия блоков в микропроцессорной системе с ПДП дана на рис. 6.26.

Выводы и сигналы контроллера

Выводы и сигналы контроллера имеют следующий смысл:

D7-0	двунаправленные линии системной ШД с тремя состояниями;
RESET	сброс внутренних регистров контроллера и внешних выходных управляющих сигналов. Сброс регистра режима PP деактивирует 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;
A3-0	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;
A7-4	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 быстроедействие основной памяти недостаточно для работы в синхронизме с ВУ, и тогда он удлиняет циклы обращения к памяти;
\overline{MEMW} \overline{MEMR}	сигналы записи и чтения памяти в режиме ПДП. В циклах чтения и записи контроллер вырабатывает парные сигналы \overline{IOW} и \overline{MEMR} или \overline{IOR} и \overline{MEMW} ;
HRQ	(Hold Request) запрос захвата шин, выдается контроллером микропроцессору на его вход HOLD при наличии запроса от ВУ;
HLDA	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	сигнал выдачи адреса, разрешающий запись старшего байта адреса, вырабатываемого контроллером, по ШД во внешний регистр-защелку, хранящий адрес до конца цикла;

DRQ_i ($i = 0...3$)	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;
$DACK_i$ ($i = 0...3$)	подтверждение (разрешение) ПДП. Информировать ВУ о предоставлении ему очередного цикла ПДП. Устанавливается и сбрасывается для передачи каждого байта данных. Вырабатывается согласно приоритетам входов DRQ_i ;
\overline{EOP}	двухнаправленный, 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.

Работа контроллера поясняется временной диаграммой (рис. 6.27). В исходном состоянии S_{01} запрограммированный контроллер ожидает запросы ВУ. По линиям DRQ ВУ сообщают о своей готовности к обмену. Получив сигнал DRQ , контроллер транслирует его в запрос для МП по его входу $HOLD$ выдачей сигнала HRQ , сигнализирующего о необходимости отключения МП от системной шины. В состоянии S_{02} контроллер ожидает отключения МП. Получив сигнал $HLDA$, контроллер определяет канал с высшим приоритетом и начинает сам управлять шинами. Собственно обмен начинается с состояния S_1 , в котором подготавливается адрес ячейки памяти и вырабатывается сигнал AEN , блокирующий реакцию других ВУ на выставленный контроллером адрес. На шине адресов выставляется младший байт адреса $A_7...A_0$, а на шине данных — старший $A_{15}...A_8$. Старший байт сопровождается сигналом $ADSTB$, записывающим его во внешний регистр для последующей непрерывной выдачи на соответствующие линии шины адресов.

Без вмешательства процессора передается блок данных. В состоянии S_2 вырабатываются сигналы чтения ВУ \overline{IOR} или памяти \overline{MEMR} в зависимости от направления обмена.

В состоянии S_3 происходит запись в ВУ или память. В этом состоянии контроллер может находиться произвольное число тактов в зависимости от готовности памяти (при ожидании). При записи для управления памятью выдается сигнал \overline{MEMR} (для ВУ — \overline{IOW}).

В состоянии S_4 во время передачи последнего байта выдается сигнал конца счета и проводится подготовка к следующему циклу.

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 ЕОР, прекращающего ПДП-обслуживание.

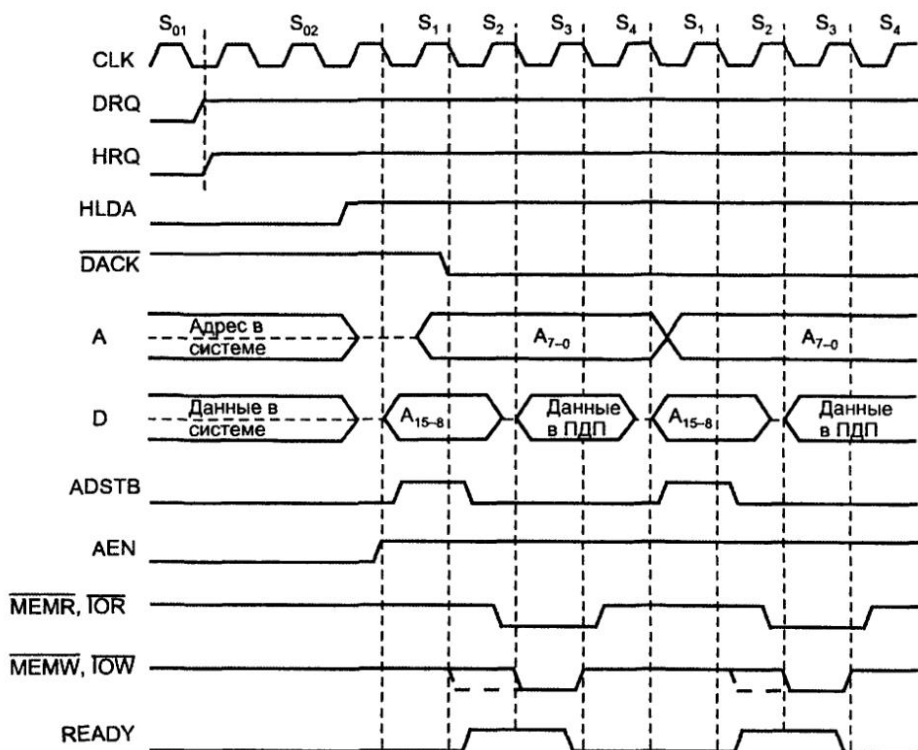


Рис. 6.27. Временные диаграммы работы контроллера прямого доступа к памяти

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. 6.28 приведен пример двухуровневой схемы с двумя контроллерами во втором ярусе, что позволяет получить 8 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 ис-

пользуется, 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.

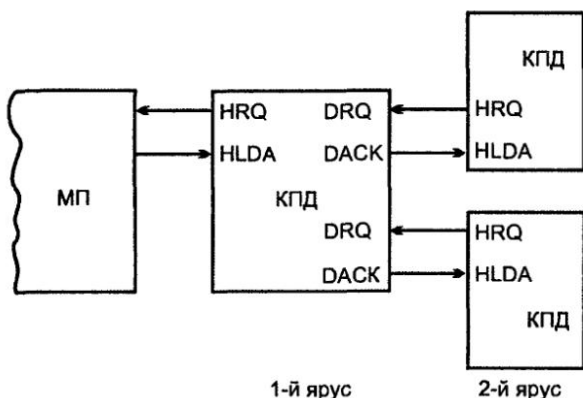


Рис. 6.28. Пример схемы наращивания числа каналов прямого доступа к памяти

§ 6.7. Программируемые интервальные таймеры

Программируемые интервальные таймеры (ПИТ, PIT) выполняют операции, связанные с временами, частотами и интервалами. ПИТ ВИС4 серий К1821 и К1860 (аналог микросхемы Intel 8254), входящий также в состав современных интегрированных периферийных СБИС и библиотек для СБИС программируемой логики, — трехканальный, содержит три 16-разрядных счетчика с независимыми режимами работы при изменении входной частоты от нулевой до 10 МГц (для разных модификаций максимальные частоты 5; 8 и 10 МГц). Таймеры могут работать в шести режимах в двоичной или двоично-десятичной системах счисления.

Структура ИС ВИС4

Структура ИС ВИС4 показана на рис. 6.29, а. Двухнаправленный буфер данных БД с тремя состояниями выхода связывает ПИТ с шиной данных системы. Блок управления чтением-записью принимает от шин МПС сигналы RD (IOR) или WR (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 CS разрешает или запрещает работу ПИТ. Сигналы младших линий адреса A₁ и A₀ выби-

рают конкретный счетчик CTR_i комбинациями 00, 01 и 10, или регистр управляющего слова (комбинацией 11).

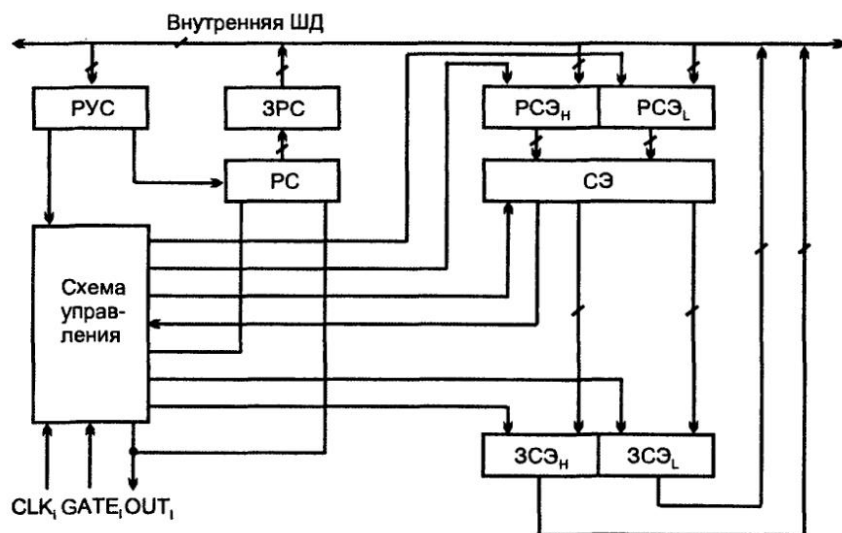
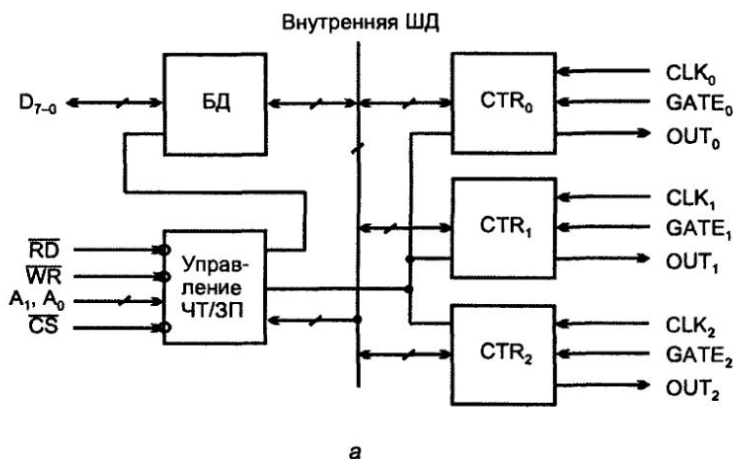
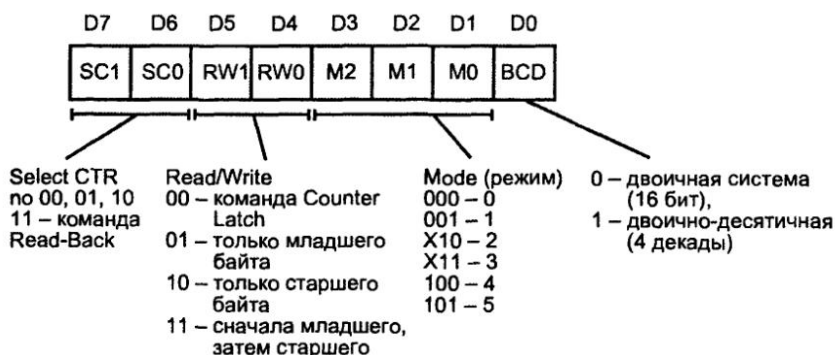


Рис. 6.29. Структура интервального таймера (а) и одного из его каналов (б)

Внутренняя структура счетчика CTR_i приведена на рис. 6.29, б. Регистр управляющего слова ПУС хранит загруженные в таймер сведения о назначенных режимах работы счетчиков. Регистр состояния РС вместе с защел-

кой 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 8-разрядными 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 8-разрядным шинам. Состояние СЭ может читаться только через защелки.

СЭ имеют на входах 8-разрядные 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.



а



б

Рис. 6.30. Форматы управляющего слова (а) и слова состояния (б) интервального таймера

Работа ПИТ

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. 6.30, 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 ниже при рассмотрении операции чтения.

После управляющего слова в счетчик записывается начальное число в формате, определенном управляющим словом. Номер счетчика выбирается сигналами A_1A_0 .

Процедуры инициализации

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

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 30, б. Он позволяет наблюдать уровень выходного сигнала, наличие в счетчике числа или нуля, запрограммированные режимы работы счетчиков.

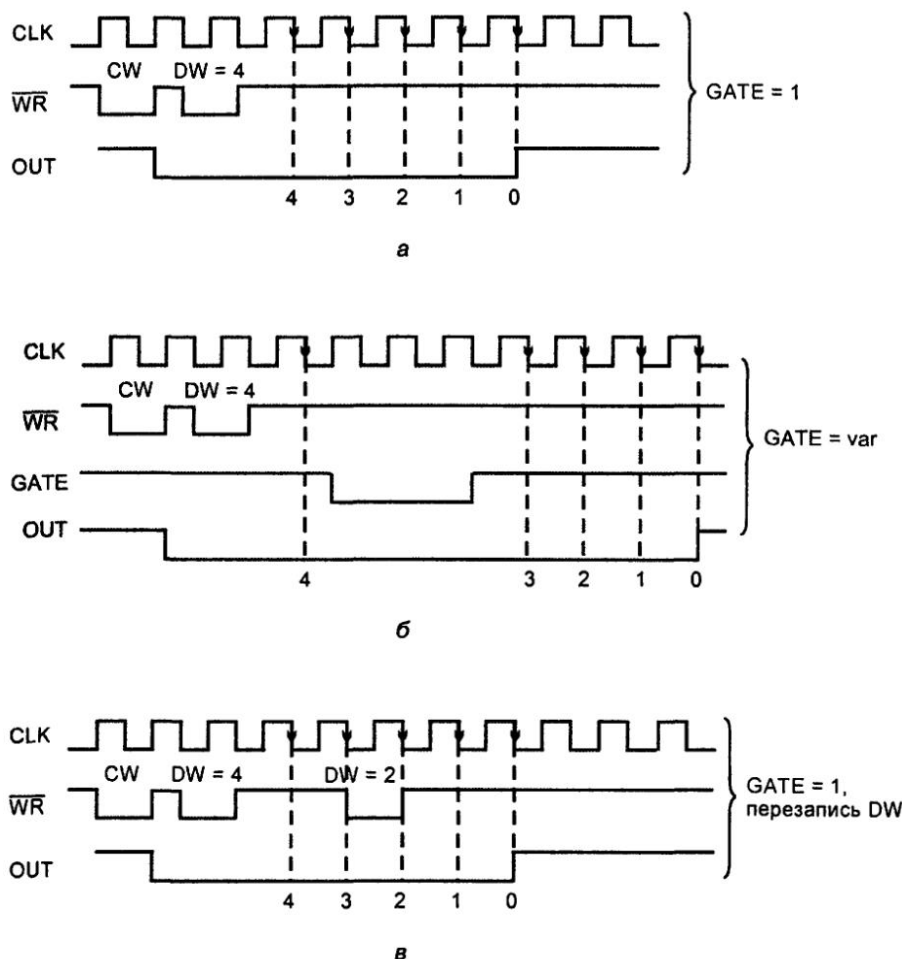


Рис. 6.31. Временные диаграммы режима 0 интервального таймера (а, б, в)

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. Рис. 6.31 показывает процессы в счетчике для режима 0 при постоянно разрешенном счете (а), наличии интервалов запрещения счета (б), когда $GATE = var$ и при поступлении нового начального числа в процессе счета, т. е. перезаписи начального числа DW, Data Word (в).

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. 6.32.

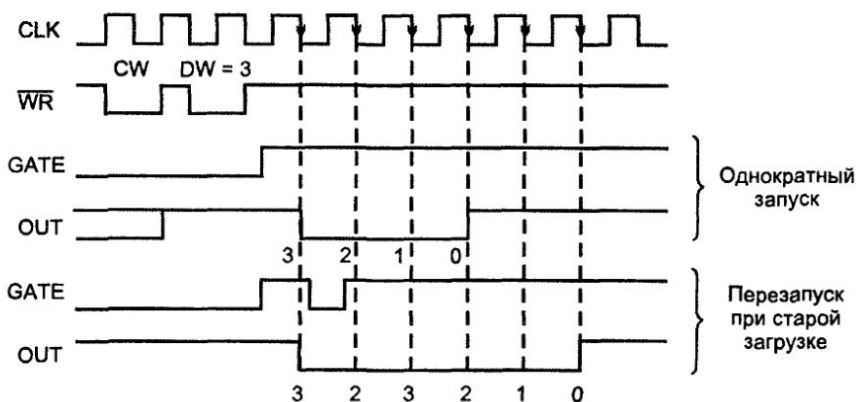


Рис. 6.32. Временные диаграммы режима 1 интервального таймера

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. 6.33).

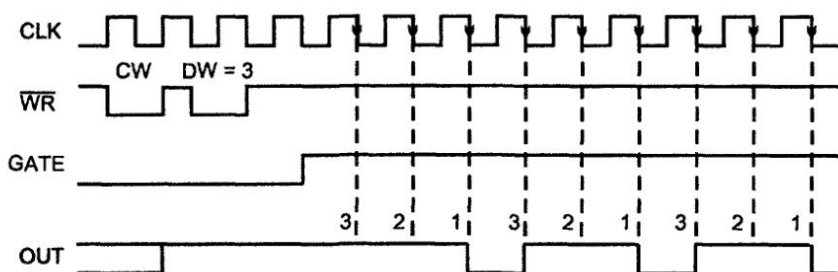


Рис. 6.33. Временные диаграммы режима 2 интервального таймера

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.

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 нечетном 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. 6.34).

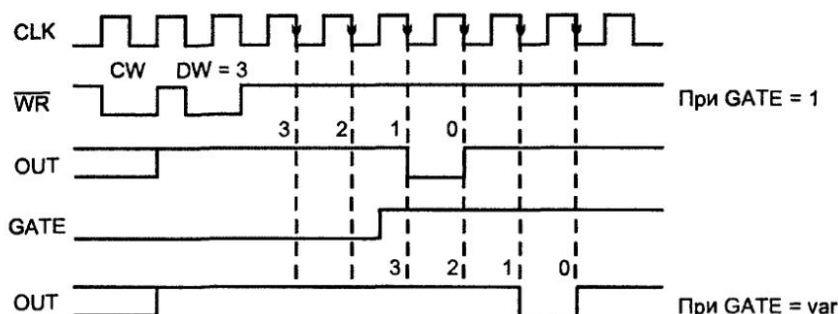


Рис. 6.34. Временные диаграммы режима 4 интервального таймера

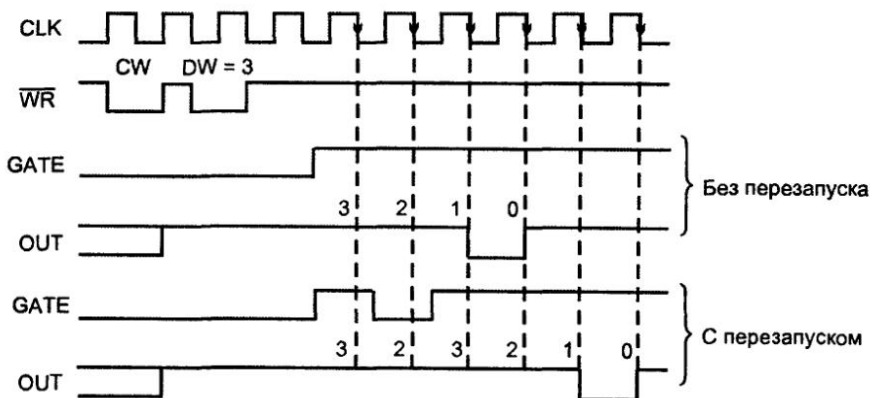


Рис. 6.35. Временные диаграммы режима 5 интервального таймера

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 .

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.

Литература к главе: [2], [5], [11], [13], [23], [37], [41], [45].

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Глава 7

Программируемые логические матрицы, программируемая матричная логика, базовые матричные кристаллы

§ 7.1. Вводные замечания

В цифровые системы обработки информации входят процессор, память, периферийные устройства и интерфейсные схемы. Процессор является стандартным устройством — он не изготавливается для конкретной системы по специальному заказу, а решает требуемую задачу путем последовательного выполнения определенных команд из присущей ему системы команд.

Память также реализуется стандартными микросхемами — ее функции остаются одними и теми же для разных систем.

Стандартные БИС/СБИС лидируют по уровню интеграции, т. к. высокая стоимость проектирования оптимизированных по плотности БИС/СБИС, достигающая сотен миллионов долларов, оказывается в данном случае приемлемой, поскольку раскладывается на большое число производимых микросхем.

Наряду со стандартными в системе присутствуют и некоторые нестандартные части, специфичные для данной разработки. Это относится к схемам управления блоками, обеспечения их взаимодействия и др. Реализация нестандартной части системы исторически была связана с применением микросхем малого и среднего уровней интеграции. Применение МИС и СИС сопровождается резким ростом числа корпусов ИС, усложнением монтажа, снижением надежности системы и ее быстродействия. 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 базо-

вые матричные кристаллы БМК, называемые также вентиляными матрицами BM (GA, Gate Array).

PLA и PAL в английской терминологии объединяются также термином PLD, Programmable Logic Devices.

Развитие БИС/СБИС с программируемой и репрограммируемой структурой оказалось настолько перспективным направлением, что привело к созданию новых эффективных средств разработки цифровых систем, таких как CPLD (Complex PLD), FPGA (Field Programmable GA) и SPGA (System Programmable GA).

В рамках современных БИС/СБИС с программируемой и репрограммируемой структурой стала решаться и задача создания целой системы на одном кристалле.

§ 7.2. Программируемые логические матрицы и программируемая матричная логика (ПЛМ и ПМЛ)

Программируемые логические матрицы

Программируемые логические матрицы появились в середине 70-х годов. Основой их служит последовательность программируемых матриц элементов И и ИЛИ. В структуру входят также блоки входных и выходных буферных каскадов (БВх и БВых).

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 ОЕ, а иногда выполняют и более сложные действия.

Основными параметрами ПЛМ (рис. 7.1) являются число входов t , число термов ℓ и число выходов n .

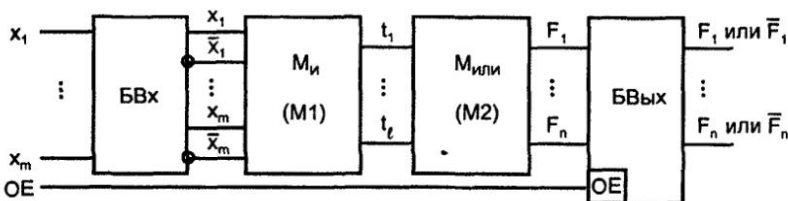


Рис. 7.1. Базовая структура ПЛМ

Переменные $x_1 \dots x_m$ 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 (двухуровневую логику). *ПЛМ способна реализовать систему n логических функций от m аргументов, содержащую не более ℓ термов.* Воспроизводимые функции являются комбинациями 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 как на основе биполярной технологии, так и на МОП-транзисторах. В матрицах имеются системы горизонтальных и вертикальных связей, в узлах пересечения которых при программировании создаются или ликвидируются элементы связи.

На рис. 7.2, а в упрощенном виде (без буферных элементов) показана схемотехника биполярной ПЛМ К556РТ1 с программированием пережиганием перемычек. Показан фрагмент для воспроизведения системы функций

$$F_1 = \bar{x}_1 \bar{x}_2 x_3 \vee x_2 \bar{x}_3 \vee x_1 \bar{x}_4 = t_1 \vee t_2 \vee t_3;$$

$$F_2 = \bar{x}_1 \bar{x}_2 x_3 \vee x_1 \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_4 \vee x_2 \bar{x}_3 x_4 = t_1 \vee t_4 \vee t_5 \vee t_6;$$

$$F_3 = x_1 \bar{x}_4 \vee x_1 \bar{x}_2 = t_1 \vee t_7$$

размерностью 4, 7, 3. Параметрами микросхемы К556РТ1 являются 16, 48, 8. Элементами связей в матрице И служат диоды, соединяющие горизонтальные и вертикальные шины, как показано на рис. 7.2, 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 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. Так выполняется операция И, в данном случае вырабатывается терм $\bar{x}_1\bar{x}_2x_3$.

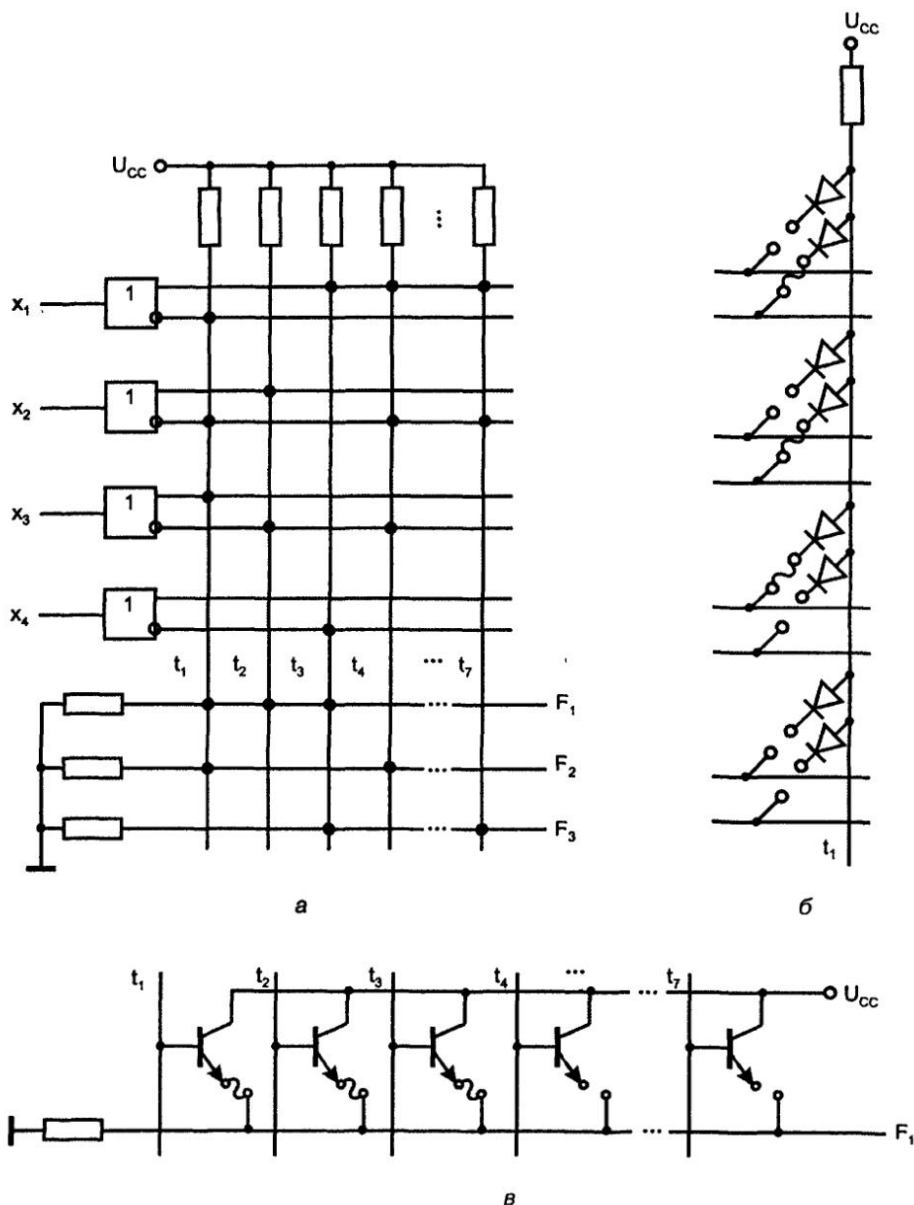


Рис. 7.2. Схемотехника ПЛМ, реализованной в биполярной технологии (а), и элементы связей в матрицах И (б) и ИЛИ (в)

Элементами связи в матрице ИЛИ служат транзисторы (рис. 7.2, в), включенные по схеме эмиттерного повторителя относительно линий термов и образующие схему ИЛИ относительно выхода (горизонтальной линии). На рис. 7.2, в показана выработка функции F_1 . Работа схемы ИЛИ, реализованной в виде параллельного соединения эмиттерных повторителей, рассмотрена ранее в § 1.2.

При изображении запрограммированных матриц наличие элементов связей (целые перемиčky) отмечается точкой в соответствующем узле.

В схемах на МОП-транзисторах в качестве базовой логической ячейки используют инвертирующие (ИЛИ-НЕ, И-НЕ). Соответственно этому меняются и операции, реализуемые в первой и второй матрицах ПЛМ. В частности, в схемотехнике п-МОП базовой ячейкой обычно служит ячейка ИЛИ-НЕ, а структура ПЛМ имеет вид (рис. 7.3). 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:

$$F_1 = \overline{t_1 \vee t_2 \vee t_3}.$$

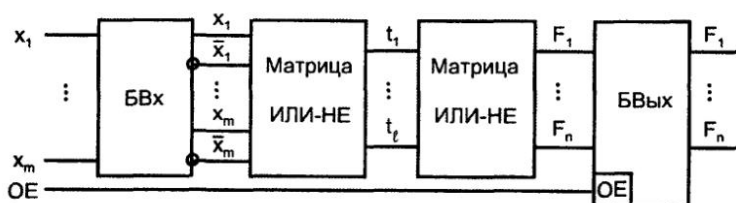


Рис. 7.3. Схемотехника ПЛМ, реализованной на МОП-транзисторах

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.

Подготовка задачи к решению с помощью ПЛМ

Имея в виду подбор ПЛМ минимальной сложности, следует уменьшить по возможности число термов в данной системе функций. Содержанием мини-

мизации функций будет поиск кратчайших дизъюнктивных форм. Вести поиск минимальных по числу термов представлений задачи следует до уровня, когда число термов становится равным ℓ — параметру 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 устройств (программаторов) и сведения для них о данной ПЛМ 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. 7.1).

Таблица 7.1

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	.

Таблица 7.1 (окончание)

x_1	x_2	x_3	x_4	F_1	F_2	F_3
Н	Н	—	Н	.	А	.
—	Н	Л	Н	.	А	.
Н	Л	—	—	.	.	А

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.

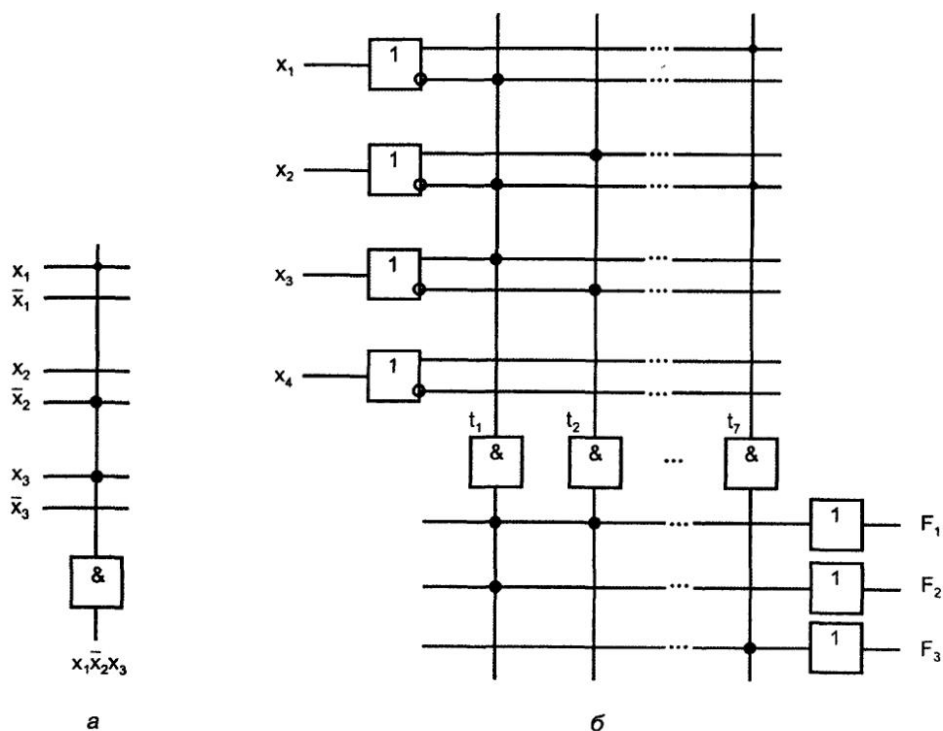


Рис. 7.4. Упрощенное изображение схемы многовходового логического элемента (а) и ПЛМ (б)

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 $x_1\bar{x}_2x_3$ показан на рис. 7.4, а. Схема рис. 7.2, а в новом упрощенном изображении имеет вид, приведенный на рис. 7.4, б.

Воспроизведение скобочных форм переключательных функций

С помощью ПЛМ можно воспроизводить не только дизъюнктивные нормальные формы переключательных функций, но и скобочные формы. 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. 7.5).

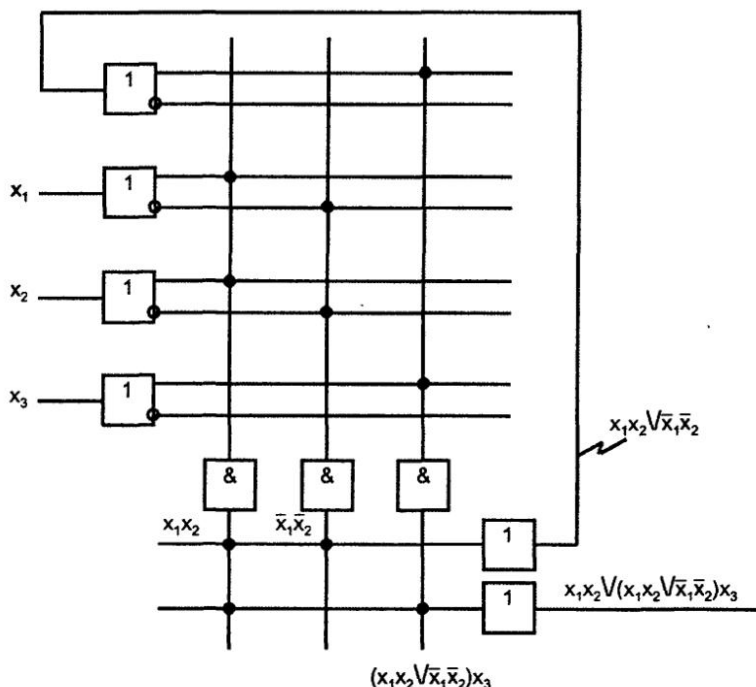


Рис. 7.5. Схема включения ПЛМ при воспроизведении скобочных форм переключательных функций

Наращивание (расширение) ПЛМ

Если размерность задачи превосходит возможности имеющихся ПЛМ, приходится их наращивать. Когда число функций в системе N превосходит число выходов ПЛМ, несколько ПЛМ включаются параллельно по входам (рис. 7.6). На выходах каждой из ПЛМ воспроизводится часть функций. Общее число ПЛМ определяется как $\lceil N/n \rceil$. Так как число термов предполагается достаточным ($\ell_{\text{сист}} < \ell$), все ПЛМ могут быть запрограммированы на одни и те же термы.

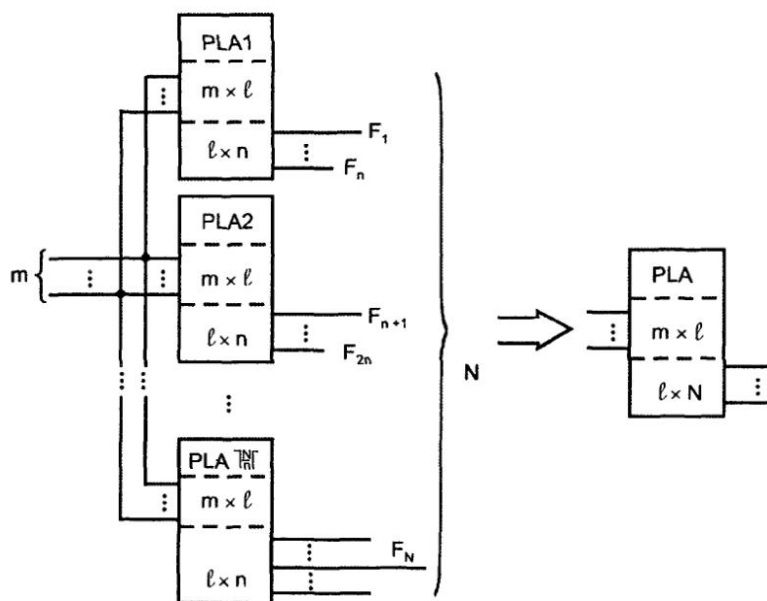


Рис. 7.6. Схема расширения ПЛМ по числу выходов

Если число термов системы $\ell_{\text{сист}}$ превышает число термов ПЛМ ($\ell_{\text{сист}} > \ell$), то к одной 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. 7.7).

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.

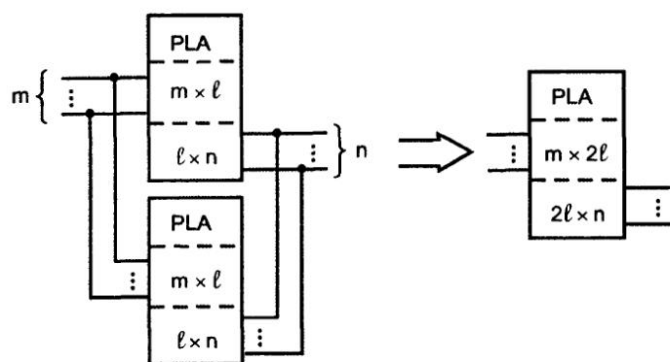


Рис. 7.7. Схема расширения ПЛМ по числу термов

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. 7.8).

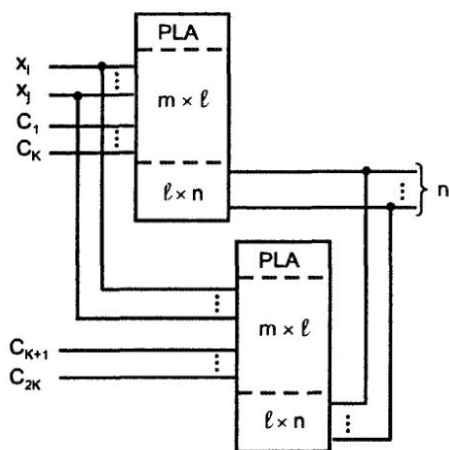


Рис. 7.8. Схема первого варианта расширения ПЛМ по числу входов

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. 7.9. 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).

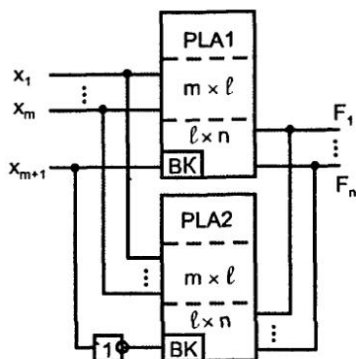


Рис. 7.9. Схема расширения числа входов ПЛМ на единицу

Простая эвристическая методика расширения ПЛМ по входам, не претендующая на оптимальность, предложена в работе [31] и описана также в [37].

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.

Программируемая матричная логика

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. 7.10) 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 придется по четыре конъюнктора.

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).

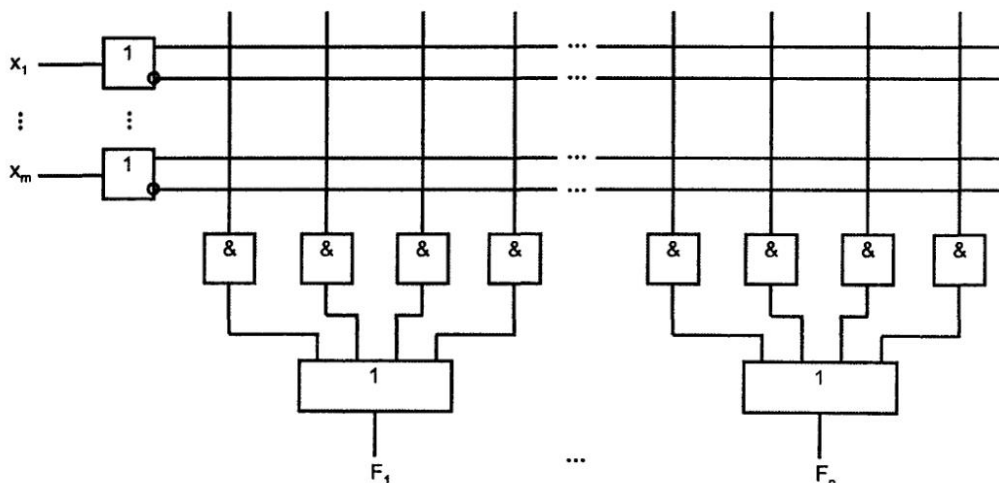


Рис. 7.10. Базовая структура ПМЛ

Функциональные разновидности ПЛМ и ПМЛ

Рассмотренные структуры ПЛМ и ПМЛ — базовые, с которых началось развитие этих направлений. В дальнейшем происходило обогащение функциональных возможностей ПЛМ и ПМЛ с помощью ряда приемов, в первую очередь следующих.

Схемы с программируемым выходным буфером

В этих схемах обеспечивается возможность получения выходных функций в прямом или инверсном виде. В такой схеме (рис. 7.11) выработанные матрицами функции $F_1^* \dots F_n^*$ проходят через выходной буфер, разрядные схемы которого выполнены как сумматоры по модулю 2.

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.

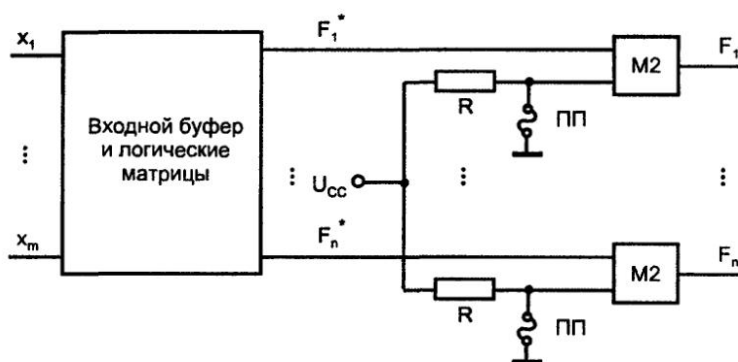


Рис. 7.11. Схема программируемого выходного буфера

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:

$$F_1 = \bar{x}_3 \bar{x}_2 \bar{x}_0 \vee \bar{x}_3 x_2 x_0 \vee x_3 \bar{x}_1 x_0 \vee x_3 x_1 \bar{x}_0;$$

$$F_2 = \bar{x}_3 \bar{x}_2 x_0 \vee \bar{x}_3 x_2 \bar{x}_1 \vee x_3 \bar{x}_1 \bar{x}_0 \vee x_3 x_1 x_0.$$

Карты Карно для этих функций (рис. 7.12) показывают контуры, соответствующие 8 различным термам системы:

$$t_1 = \bar{x}_3 \bar{x}_2 \bar{x}_0, \quad t_2 = \bar{x}_3 x_2 x_0, \quad t_3 = x_3 \bar{x}_1 x_0, \quad t_4 = x_3 x_1 \bar{x}_0;$$

$$t_5 = \bar{x}_3 \bar{x}_2 x_0, \quad t_6 = \bar{x}_3 x_2 \bar{x}_1, \quad t_7 = x_3 \bar{x}_1 \bar{x}_0, \quad t_8 = x_3 x_1 x_0.$$

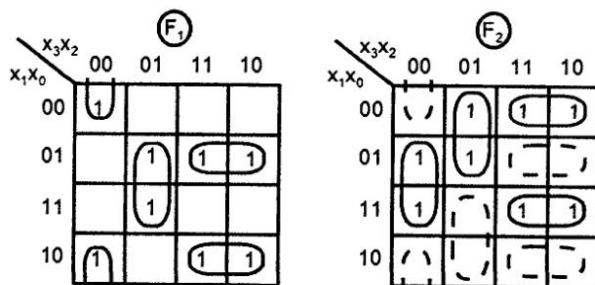


Рис. 7.12. Карты Карно для примера воспроизведения функций в ПЛМ с программируемым выходным буфером

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:

$$F_1 = \bar{x}_3\bar{x}_2\bar{x}_0 \vee \bar{x}_3x_2x_0 \vee x_3\bar{x}_1x_0 \vee x_3x_1\bar{x}_0 = t_1 \vee t_2 \vee t_3 \vee t_4;$$

$$\bar{F}_2 = \bar{x}_3\bar{x}_2\bar{x}_0 \vee \bar{x}_3x_2x_1 \vee x_3\bar{x}_1x_0 \vee x_3x_1\bar{x}_0 = t_1 \vee t_5 \vee t_3 \vee t_4,$$

в которой всего пять различных термов. Возврат от функции \bar{F}_2 к функции F_2 осуществляется пережиганием перемычки в линии выхода F_2 .

Схемы с двунаправленными выводами

Используя элементы с тремя состояниями выхода, можно построить схему, в которой некоторые выводы можно приспособлять 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. 7.13). Выход элемента одновременно связан с матрицей И как вход.

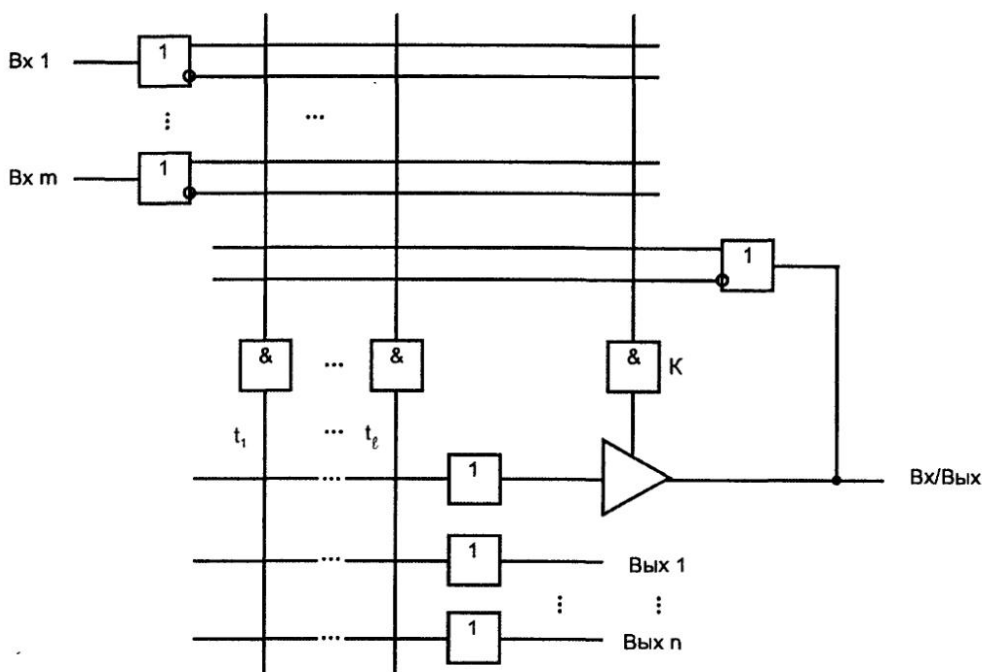


Рис. 7.13. Схема с двунаправленным буфером

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.
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 единственный 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 $m + n + p$.

Схемы с памятью

Эти схемы позволяют строить автоматы наиболее удобным способом, т. к. помимо комбинационной части содержат на кристалле триггеры (регистры), обычно типа D (рис. 7.14). ПЛМ с памятью характеризуется четырьмя параметрами. Кроме трех обычных параметров, она имеет и параметр r — число элементов памяти (разрядов регистра). Структура рис. 7.14 совпадает с канонической 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.

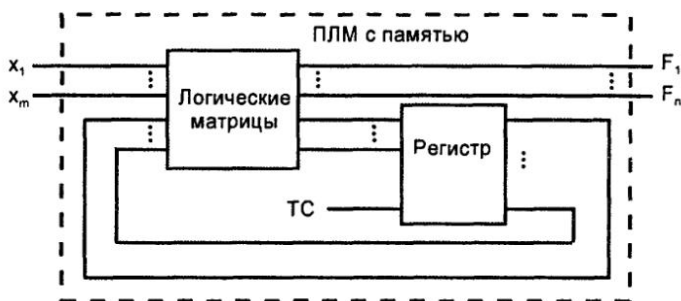


Рис. 7.14. Структура ПЛМ с памятью

ПМЛ с разделяемыми конъюнкторами

Наряду с модификациями схем, рассмотренными выше, существуют и специфические модификации, относящиеся только к ПМЛ. К ним относится вариант с так называемыми разделяемыми конъюнкторами. Прием "разделения конъюнкторов" 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.

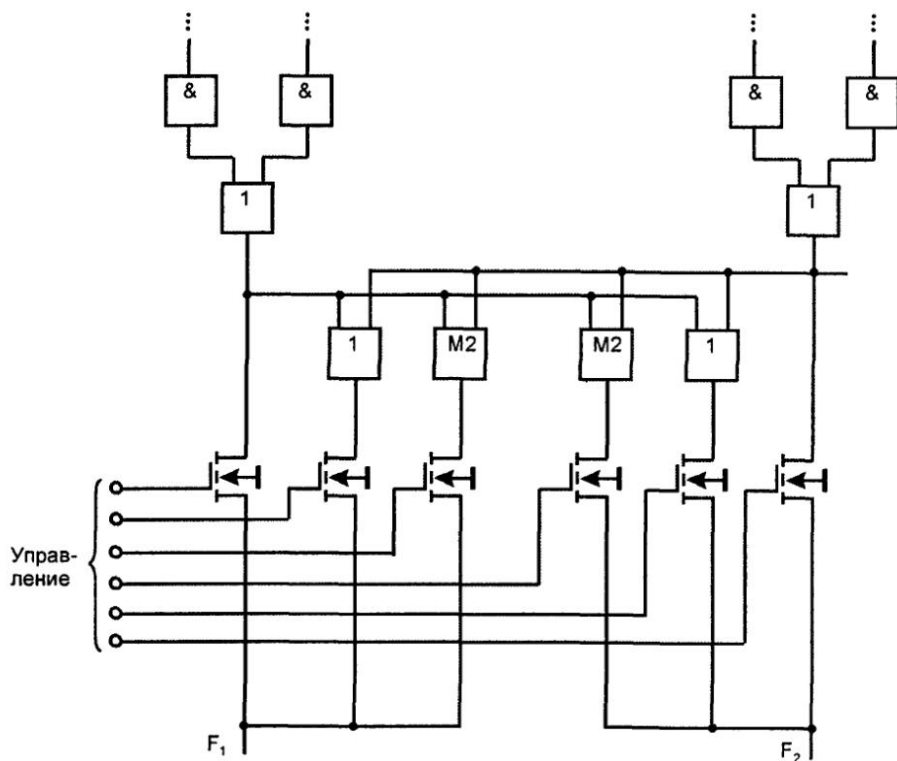


Рис. 7.15. Пример реализации разделения термов в ПМЛ

Вариант схмотехнической реализации разделяемости конъюнкторов показан на рис. 7.15. 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.

ПМЛ серии К1556

Первая отечественная ПМЛ появилась в серии КР1556 (микросхема ХЛ8, за которой последовали ИС ХП4, ХП6, ХП8). Микросхема ХЛ8 — ПМЛ с двунаправленными выводами (входами-выходами), структура которой показана на рис. 7.16. 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.

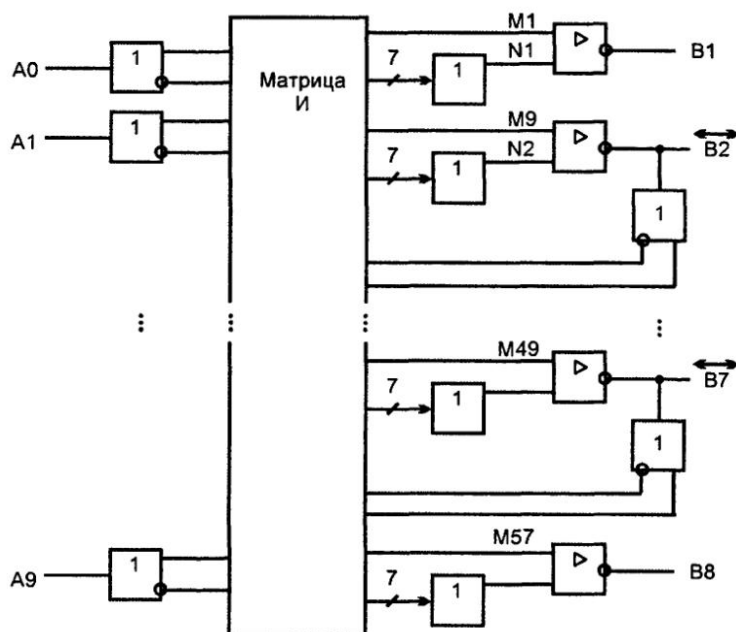


Рис. 7.16. Структура ПМЛ КР1556ХЛ8

Выходные буферы ПМЛ получают разрешение или запрещение работы от матрицы И, как было рассмотрено в предыдущем параграфе. Набор элементов ИЛИ состоит из 8 элементов с семью входами, т. е. на каждый элемент ИЛИ приходится по 7 конъюнкторов с числом входов от 10 до 16. Исходя из сказанного, можно оценить и размерность матрицы И, содержащей 2048 узлов (64×32).

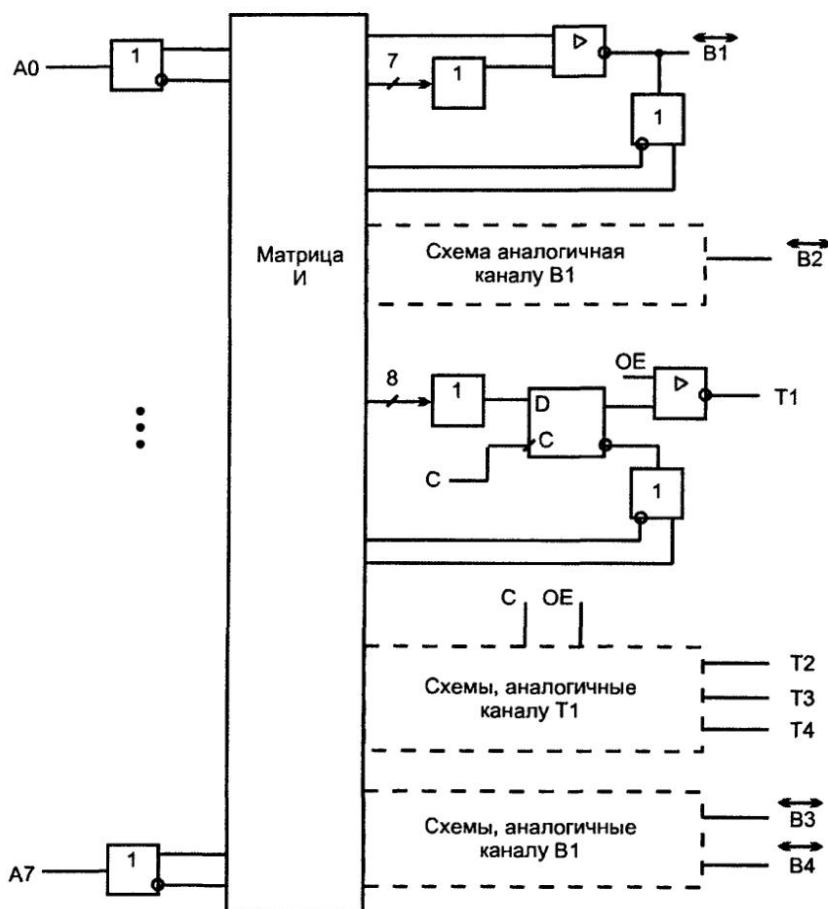


Рис. 7.17. Структура ПМЛ КР1556ХП4

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. 7.17) имеет первый уровень логики, на котором образуются термы входных переменных, второй уровень — матрица ИЛИ, состоящая из 8 дизъюнкторов (четырех 7-входовых и четырех 8-входовых). Выходные

усилители выполнены по схеме с тремя состояниями. Четыре D-триггера имеют управление от положительного фронта внешнего синхросигнала С. Сигнал ОЕ управляет буферами, подключенными к выходам триггеров.

Число входов у ПМЛ типа ХП — восемь, 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 4-х 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, 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:

$$Q_{1H} = \overline{LE} \overline{Q_1} Q_0 \vee \overline{LE} Q_1 \overline{Q_0} \vee LE A_1,$$

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:

$$Q_{2H} = \overline{LE} \overline{Q_2} Q_1 Q_0 \vee \overline{LE} Q_2 \overline{Q_1} \vee \overline{LE} Q_2 \overline{Q_0} \vee LE A_2;$$

$$Q_{3H} = \overline{LE} \overline{Q_3} Q_2 Q_1 Q_0 \vee \overline{LE} Q_3 \overline{Q_2} \vee \overline{LE} Q_3 \overline{Q_1} \vee \overline{LE} Q_3 \overline{Q_0} \vee LE A_3,$$

где первые слагаемые отображают процесс переключения разряда, последнее — 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). *Подробнее пример проектирования на основе ПМЛ рассмотрен в § 9.2.*

При проектировании устройств на основе ПЛМ и ПМЛ пользуются подсистемами автоматизации проектирования, т. к. ручная подготовка задачи может оказаться неприемлемо громоздкой. 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.

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 конъюнктов. Разные выходы имеют от 8 до 16 конъюнктов. Выходные величины вырабатываются не просто дизъюнктами, а более сложными схемами, называемыми макроэлементами (макроячейками).

Схема макроэлемента PAL 22V10 содержит триггер типа D с цепями тактирования, асинхронного сброса AR и синхронной установки SP¹ (рис. 7.18). Мультиплексор "4—1" работает на выходной буфер, мультиплексор "2—1" передает сигналы обратной связи в матрицу И. Цепи с плавкими перемычками программируют мультиплексоры. На вход мультиплексора "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.

¹ Сигналы AR и SP вырабатываются специальными терминами матрицы И.

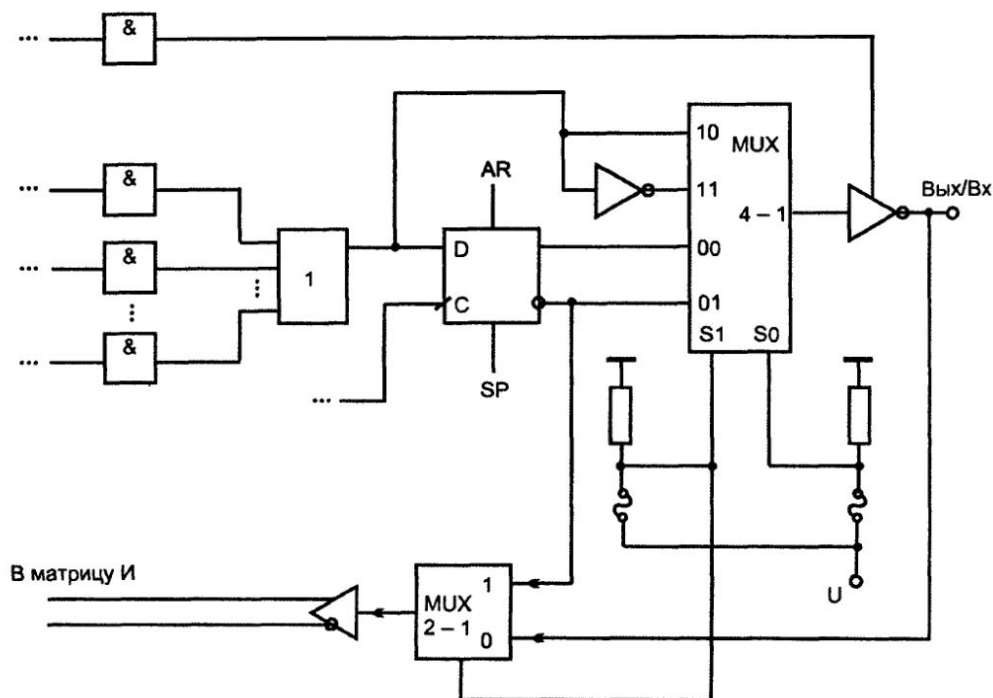


Рис. 7.18. Схема макроэлемента PAL 22V10

Пример более сложной структуры PLD

На рис. 7.19 показана структура (матрица И и один из 12 макроэлементов) БИС, 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 \bar{F}), 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 макроэлемента 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 разделение термов, позволяя данному макроэлементу не только исполь-

зовать 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.

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 ноль. В первом случае, проходя через элемент M2, функция инвертируется, во втором — не изменяется. Выход элемента M2 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 M2 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 M2 приводит к сложению величины Q с единицей по модулю 2, т. е. к подаче на вход триггера через мультиплексор величины \bar{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 M2.

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 вентилях. Двенадцать макроэлементов за счет комбинирования своих термов с термами соседних макроэлементов позволяют получать логические функции от 4, 8, 12 либо 16 термов. Время распространения сигналов через матрицу составляет приблизительно 25 нс.

§ 7.3. Базовые матричные кристаллы (вентильные матрицы с масочным программированием)

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, кроме того, позволила выполнить и нетиповые части машины на БИС.

Формулировку "позволила выполнить" в данном случае следует понимать с учетом экономических факторов.

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.

Выход из создавшихся трудностей был найден на путях разработки БИС/СБИС, 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 элементов (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 3 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 80-х годов. Применяются схемотехнологии КМОП, ТТЛШ, ЭСЛ и др. В настоящее время уровень интеграции БМК достиг миллионов вентилях 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. Учитывается и число 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 потребо-

ваться до 500...600 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 элементов, регулярно повторяющийся на определенной площади кристалла. Этот набор может состоять из некоммутированных элементов, а также из частично коммутированных. Базовые ячейки внутренней области БМК именуются матричными базовыми ячейками (МБЯ), ячейки периферийной зоны — периферийными базовыми ячейками (ПБЯ). Применяются два способа организации ячеек БМК:

- из элементов МБЯ может быть сформирован один логический элемент, а для реализации более сложных функций используются несколько ячеек;
- из элементов МБЯ может быть сформирован любой функциональный узел, а состав элементов ячейки определяется схемой самого сложного узла.

Функциональная ячейка (ФЯ) — функционально законченная схема, реализуемая путем соединения элементов 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 изготавливаемого устройства, как принято говорить, должна быть покрыта элементами библиотеки.

Эквивалентный вентиль (ЭВ) — группа элементов БМК, соответствующая возможности реализации логической функции вентиля (обычно это двух-

входовой 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. 7.20. Первоначальной и, в известной мере, классической является структура канального БМК (рис. 7.21, а). Во внутренней (центральной) области такого БМК расположена матрица базовых ячеек 1 и каналы для трассировки 2.

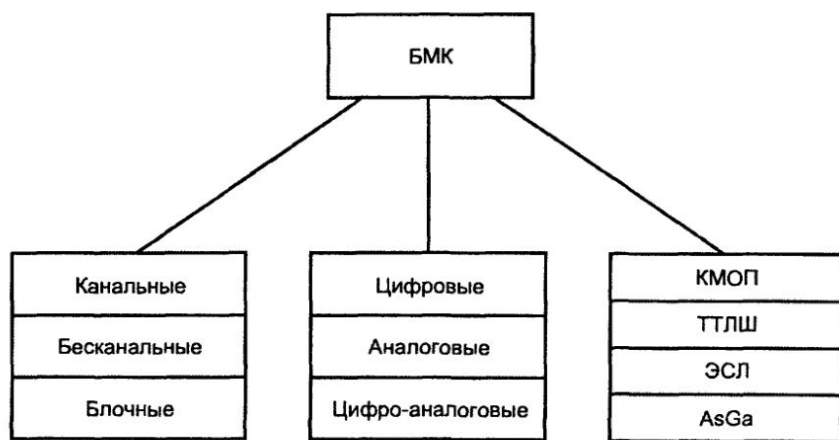


Рис. 7.20. Классификация базовых матричных кристаллов

Каналы могут быть вертикальными и горизонтальными как на рис. 7.21, а, либо только вертикальными (рис. 7.21, б). Канальные БМК могут иметь большие возможности по созданию связей, но имеют низкую плотность упаковки из-за значительных затрат площади кристалла на области межсоединений.

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. Внутренняя область такого БМК содержит плотно упакованные ряды базовых ячеек и не имеет фиксированных каналов для

трассировки межсоединений (рис. 7.21, 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.

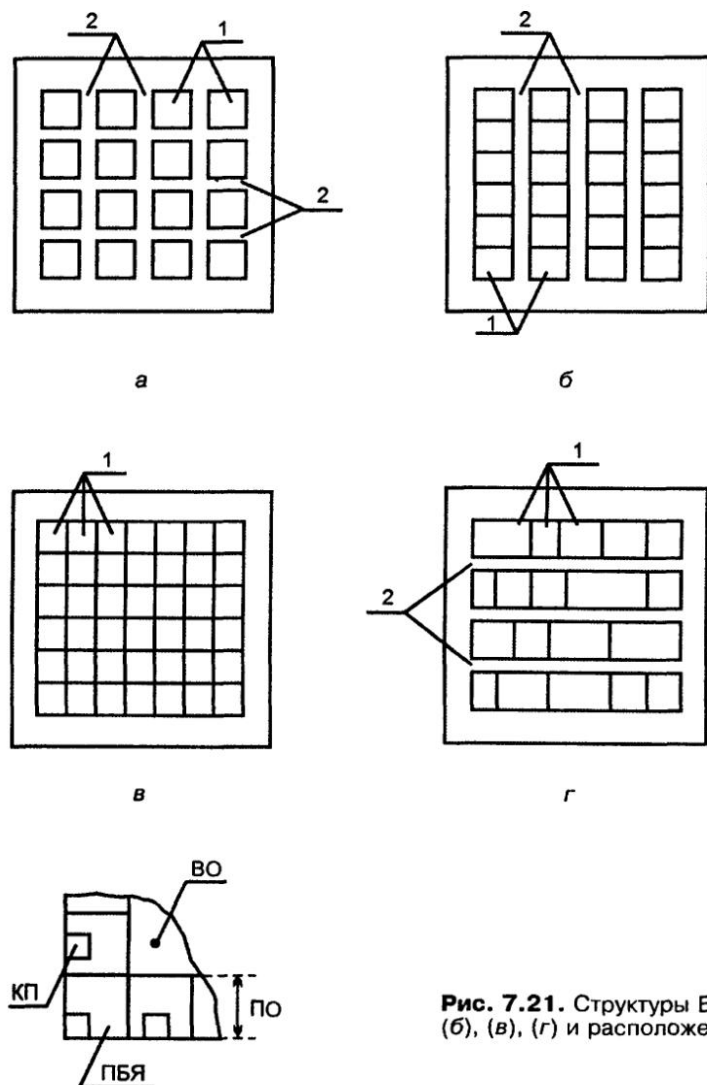


Рис. 7.21. Структуры БМК различных типов (а), (б), (в), (г) и расположение областей БМК (д)

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.

Бесканальные БМК реализуются в вариантах "*море вентиляй*" и "*море транзисторов*". Первый содержит массив законченных логических элементов, второй — массив транзисторов.

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. 7.21, з). 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. 7.21, 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. 7.22). На периферии блоков изготавливаются внутренние буферные каскады для формирования достаточно мощных сигналов, обеспечивающих передачу сигналов по межблочным связям, имеющим относительно большую длину.

Тип обрабатываемых сигналов (цифровые, аналоговые) влияет на качество и состав схемных элементов базовых ячеек. В связи с этим БМК подразделяют-

ся 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.

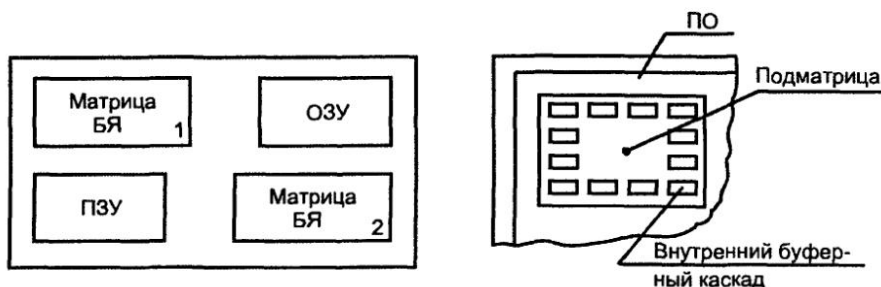


Рис. 7.22. Блочная структура БМК

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. На основе схемотехнологии ТТЛШ выполнялись БМК среднего быстродействия.

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 изготавливать 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. 7.23 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 R_0 , 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.

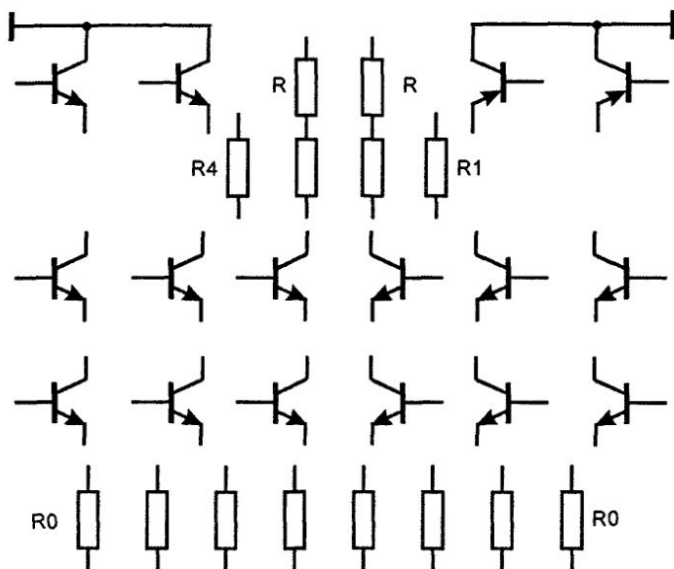


Рис. 7.23. Компонентный состав базовой ячейки БМК типа ЭСЛ

На рис. 7.24 представлен один из вариантов БЯ БМК типа КМОП. Схемными элементами таких БЯ служат только транзисторы с р- 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. 7.24 показаны топология и электрическая схема ячейки с 4 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 проводимо-

сти каналами всегда используются попарно, поэтому пары транзисторов могут иметь общий затвор.

Усложнение ячейки достигается объединением простых ячеек в группу.

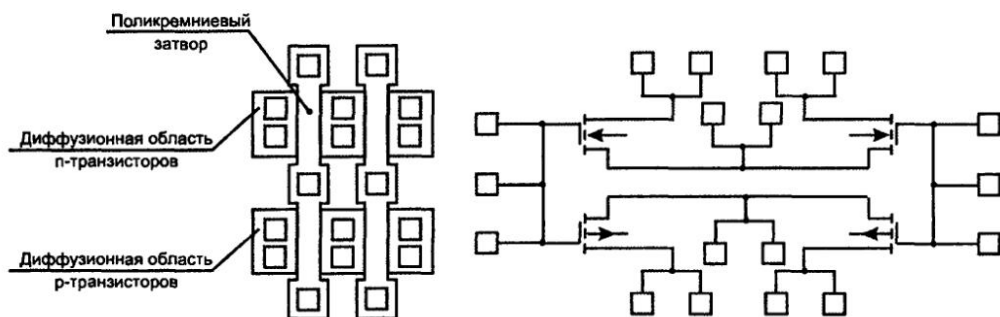


Рис. 7.24. Вариант базовой ячейки БМК типа КМОП

Параметры БМК

Параметры БМК можно разделить на 4 группы:

- ❑ функциональные возможности (число эквивалентных вентилях, 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. 7.2 приведены основные параметры некоторых отечественных БМК, представляющих разные схемотехнологические типы.

Таблица 7.2

Схема	Тип БМК	Эквивалентных вентилях, тыс.	Контактных площадок	Задержка вентиля (нс) или тактовая частота (МГц)	Мощность вентиля мВт/вент
K1520XM6	Канальный ЭСЛ	10	208	0,35 нс	2
ТРАП-50	Канальный ТТЛШ	50	256	2,5 нс	0,05

Таблица 7.2 (окончание)

Схема	Тип БМК	Эквивалент- ных вентиляй, тыс.	Контакт- ных пло- щадок	Задержка вентиля (нс) или тактовая частота (МГц)	Мощность вентиля мВт/вент
Из серии "Ряд"	КМОП	22	132	50 МГц	—
ТИТУЛ-30	Море вентиляй, КМОП	30	202	2,0 нс	Статика 10^{-4} Динамика 0,015 *

* f — частота переключения вентиля, МГц.

На уровне мировой техники изготавливаются БМК с миллионами эквивалентных вентиляй, обладающих задержками 0,1...0,2 нс.

Литература к главе: [4], [6], [31], [37], [41], [43], [49].

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Глава 8

Современные и перспективные БИС/СБИС со сложными программируемыми и репрограммируемыми структурами (FPGA, CPLD, FLEX, SOC и др.)

§ 8.1. Общие сведения

Микросхемы ПЛМ, ПМЛ и БМК, рассмотренные в гл. 7, положили начало двум основным ветвям дальнейшего развития логических схем с программируемой и репрограммируемой структурами. Продолжением линии ПМЛ стали БИС/СБИС CPLD (Complex Programmable Logic Devices), а линии БМК — FPGA (Field Programmable Gate Arrays). Стремление объединить достоинства обеих линий привело к созданию БИС/СБИС смешанной (комбинированной) архитектуры, для которых еще не выработано общепринятое название (фирма Altera пользуется названием FLEX (Flexible Logic Element MatriX) — гибкие). Рост уровня интеграции дал возможность размещать на кристалле схемы, сложность которых соответствует целым системам. Эти схемы именуются SOC (Systems On Chip).

Сказанное иллюстрируется рис. 8.1, где под MPGA понимаются Mask Programmable GAs (вентильные матрицы с масочным программированием или БМК), а остальные термины уже объяснены.

Новизна темы, которой посвящена эта глава, сопровождается отсутствием установившейся терминологии, особенно в русскоязычной литературе, где иногда одни и те же термины обозначают разные вещи. Для некоторых терминов русские аналоги еще не определились. Ввиду сказанного ниже используются преимущественно английские термины и аббревиатуры, что, кстати говоря, характерно также для справочной литературы и документации САПР.

Общее название, объединяющее совокупность ИС, рассматриваемых в этой главе, т. е. "БИС/СБИС с программируемой (либо репрограммируемой) структурой" слишком громоздко, поэтому для краткости будем пользоваться обозначением "СБИС ПЛ" (СБИС программируемой логики), в котором не

разделяются понятия БИС и СБИС (для нас это разделение несущественно) и не отражается однократность или многократность программируемости микросхемы.

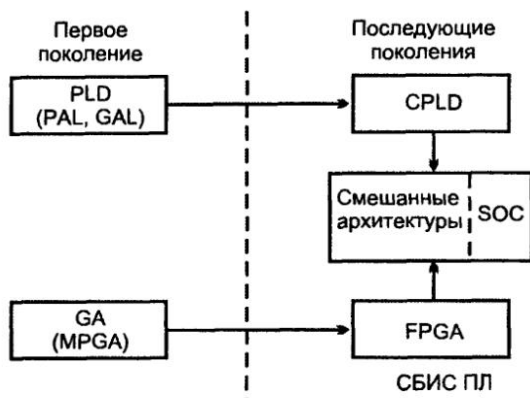


Рис. 8.1. Взаимосвязь поколений СБИС программируемой логики

В разработке СБИС ПЛ участвуют уже десятки фирм, ведущими среди них являются Xilinx, Altera, Actel, Atmel, AMD (Vantis), Lattice (все США) и некоторые другие. Перечисленные фирмы достаточно полно представляют спектр продукции в области СБИС ПЛ, хотя и не исчерпывают ее. Последующее изложение темы ориентировано в основном на разработки фирм Xilinx, Altera и Actel.

Сфера применения СБИС ПЛ чрезвычайно широка, на них могут строиться не только крупные блоки систем, но и системы в целом, включая память и процессоры. Области применения СБИС ПЛ уточняются в дальнейшем, предварительно отметим важность таких применений, как *отработка прототипов систем* при их проектировании, даже если конечная реализация систем рассчитана на другие средства, и *создание малотиражных изделий* быстрыми и эффективными способами.

СБИС ПЛ классифицируются по нескольким признакам.

Классификация по конструктивно-технологическому типу программируемых элементов

Классификация СБИС ПЛ по конструктивно-технологическому типу показана на рис. 8.2. Программируемость, т. е. реализуемость конкретного проекта на стандартной СБИС, 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 схемы, формируемой на кристалле. Число программируемых двухполюсников (программируемых точек связи ПТС) в СБИС ПЛ зависит от ее сложности и может достигать до нескольких миллионов. Для современных СБИС ПЛ характерны следующие виды программируемых ключей:

- ☐ переключки типа antifuse (русский термин отсутствует);
- ☐ ЛИЗМОП транзисторы с двойным затвором (см. рис. 4.16, б и текст к нему);
- ☐ ключевые транзисторы, управляемые триггерами памяти конфигурации ("теневым" ЗУ).

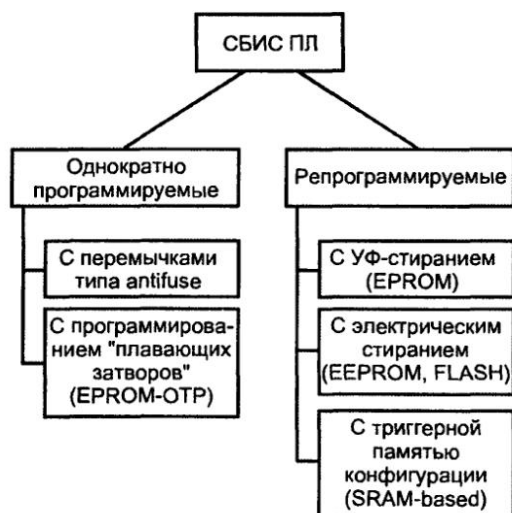


Рис. 8.2. Классификация СБИС ПЛ по типу программируемых элементов

Программирование с помощью *переключек типа antifuse* является однократным. Высококачественные переключки фирмы Actel (рис. 8.3) компактны, имеют очень малые токи в первоначальном (непроводящем) состоянии (около одного фемтоампера, $1 \text{ фА} = 10^{-15} \text{ А}$). Переключка образована трехслойным диэлектриком с чередованием слоев "оксид-нитрид-оксид". Соответственно чередованию слоев Oxid-Nitrid-Oxid переключки также называют переключками типа ONO.

Программирующий импульс напряжения пробивает переключку и создает проводящий канал из поликремния между электродами (один электрод поликремниевый, другой — диффузионная область n^+). Величина тока, создаваемого импульсом программирования, влияет на диаметр проводящего ка-

нала, что позволяет управлять параметрами проводящей перемычки (ток 5 мА создает перемычку со средним значением сопротивления 600 Ом, ток 15 мА — 100 Ом). Размер ℓ зависит от топологической нормы применяемой технологии (близок к ней). Паразитная емкость перемычки для топологической нормы 1 мкм менее 10 фФ. Параметры обоих состояний перемычки должны сохраняться около 40 лет.

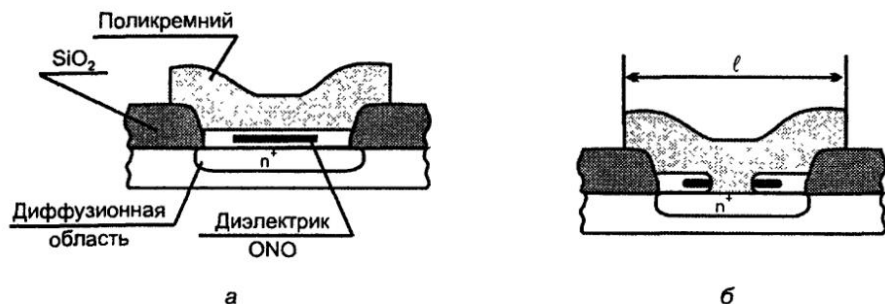


Рис. 8.3. Программируемые перемычки типа ONO до (а) и после (б) программирования

Элементы EPROM и EEPROM (Flash) на ЛИЗМОП транзисторах с плавающим затвором используются в схемах программируемой памяти и рассмотрены в гл. 4. Точно так же используются они и в СБИС ПЛ. Из элементов с УФ-стиранием выделился вариант вообще без возможности стирания данных — вариант EPROM-ОТР (ОТР, One Time Programmable). Если в обычных EPROM стирание данных производится облучением кристалла через прозрачное окошко в корпусе, то в схемах ОТР дорогостоящий корпус с окошком заменен на дешевый без окошка, т. е. возможность стирания исключается.

Не повторяя подробностей, напомним основные свойства элементов EPROM и EEPROM.

Репрограммируемые СБИС ПЛ на основе схемотехники EPROM требуют длительного (около часа) стирания старой конфигурации под воздействием ультрафиолетового излучения с извлечением СБИС из устройства и ограничением числа программирований из-за деградации свойств материалов под действием УФ-излучения. Память конфигурации с EEPROM, стираемая электрическими сигналами, для обновления не требует извлечения микросхемы из устройства, допускает достаточно большое число циклов стирания ($10^4 \dots 10^6$), стирание старой и запись новой информации занимают время порядка миллисекунд. В то же время площадь запоминающего элемента с электрическим стиранием вначале была приблизительно вдвое больше, чем площадь элемента с УФ-стиранием. В последнее время схемотехника EEPROM совершенствуется и все больше вытесняет схемотехнику EPROM.

Транзисторный ключ, управляемый триггером памяти конфигурации, показан на рис. 8.4. Ключевой транзистор T2 замыкает или размыкает участок *ab* 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.

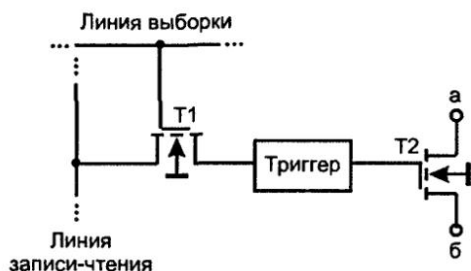


Рис. 8.4. Схема ключевого транзистора, управляемого триггером памяти конфигурации

Загрузка соответствующих данных в память конфигурации программирует СБИС ПЛ. Быстрый процесс оперативного программирования может производиться неограниченное число раз. В СБИС ПЛ 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 времени порядка десятков и даже сотен миллисекунд, если речь не идет о специальных СБИС ПЛ с так называемым динамическим оперативным репрограммированием.

Триггеры памяти конфигурации распределены по всему кристаллу СБИС вперемешку 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*.

Репрограммирование СБИС ПЛ с триггерной памятью конфигурации производится в том же режиме, что и рабочий режим, путем записи кодовой последовательности в цепочку триггеров. Стирание информации как специфический процесс воздействия на запоминающие элементы, требующий относительно длительных операций, вообще устранено. Несмотря на повы-

шенную сложность запоминающего элемента и его энергозависимость в силу указанных и других (рассматриваемых ниже) достоинств, СБИС ПЛ с триггерной памятью конфигурации занимают важнейшее место в новых вариантах FPGA и CPLD.

Говоря об общих свойствах СБИС ПЛ, следует отметить, что благодаря регулярной структуре они *реализуются с уровнем интеграции, близким к максимальному*. Так как для средств программирования межсоединений требуются затраты дополнительной площади кристалла, СБИС ПЛ по уровню интеграции уступают БМК, но в последнее время все более успешно их догоняют.

В отличие от БМК, СБИС ПЛ выпускаются как полностью готовые, в них реализованы уже не только логические элементы, триггеры и т. п., но и межсоединения. Потребитель СБИС ПЛ не обращается к их изготовителю для выполнения каких-либо завершающих операций, т. к. программирование выполняет самостоятельно. *Это дает основания отнести 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 СБИС ПЛ большое внимание уделяется вопросам *понижения потребляемой мощности*. С ростом сложности СБИС потребляемая мощность становится наиболее критическим фактором. Так как мощность пропорциональна квадрату напряжения питания U_{cc} , его снижение дает значительный эффект. 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 и даже 1,5 В.

Поскольку передача сигналов низкого уровня по внешним связям не приемлема из-за малой помехоустойчивости таких сигналов, часто в СБИС используются два напряжения питания: повышенное для схем ввода/вывода данных и меньшее для питания основных логических схем и накопителей памяти. Соответственно этому для разных областей кристалла могут изготавливаться транзисторы с разными пороговыми напряжениями.

Для быстродействующих низковольтных схем разрабатывается глубоко субмикронная технология КМОП и возрождается старая идея построения схем типа "кремний на диэлектрике" (SOI, Silicon On Insulator), в которых устранены многие паразитные схемные элементы. В схемах особо низковольтной логики достигается очень малая мощность элементов, например, 4,3 нВт/вентиль/МГц. При этом плотность упаковки такова, что достижима схемная сложность до 40 млн вентилях (около 100 млн транзисторов).

В схемах СБИС ПЛ нередко используется иерархия режимов понижения мощности.

В активных режимах часто при программировании 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).

Для реализации режима с Турбо-битом используются специальные схемы выявления фактов изменения входных сигналов. Каждый вход снабжается несложной схемой, содержащей элементы ИЛИ, M2 и элемент задержки. Любое изменение входного сигнала выявляется и вызывает подачу на схему нормального питания, необходимого для быстрого протекания в ней 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.

Схема выявления перепадов входных сигналов увеличивает задержку на пути "вход-выход" СБИС на 30...40%. 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.

Режим *Standby Power* (мощности в режиме ожидания) 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* 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.

Эффективность 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.

§ 8.2. Программируемые пользователем вентильные матрицы (FPGA)

Программируемые пользователем вентильные матрицы (ППВМ или FPGA) 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 или IOB, Input/Output Blocks). Таким образом, архитектуру ППВМ можно представить рисунком, подобным рис. 7.21, а, д, если вместо наименования "базовая ячейка" иметь в виду наименование КЛБ, а вместо "периферийной ячейки" — БВВ.

К наиболее известным FPGA относятся БИС/СБИС семейств XC2000, XC3000, XC4000, XC5000 и Spartan фирмы Xilinx, которая в 1985 г. впервые выпустила FPGA с триггерной памятью конфигурации. Среди FPGA с перемычками типа antifuse следует отметить семейства ACT1, 1200XL, ACT3, 3200DX фирмы Actel, используемые, в частности, в космической аппаратуре США.

Свойства и возможности FPGA зависят в первую очередь от характера их КЛБ и системы межсоединений.

Логические блоки FPGA

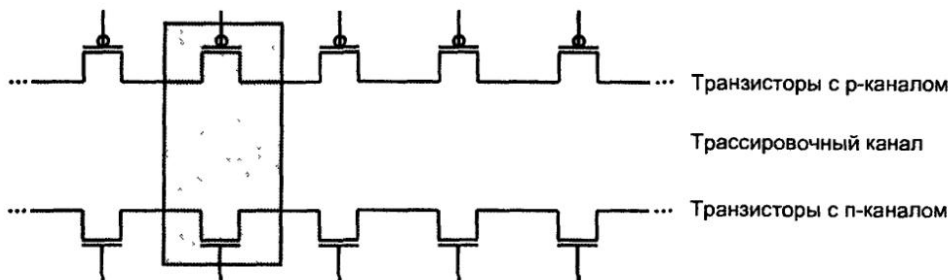
В качестве КЛБ (далее для краткости просто ЛБ — логические блоки) используются:

- ☐ транзисторные пары, 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).

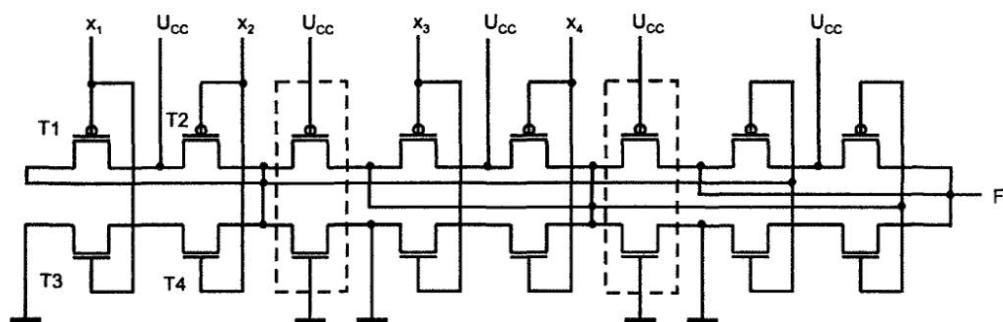
Первое свойство связано с тем, насколько "мелкими" будут те части, из которых можно "собирать" нужные схемы, второе — с тем, насколько велики логические возможности ЛБ.

Примером наиболее *мелкозернистого* может служить ЛБ фирмы Crosspoint Solutions (рис. 8.5, а). Блок содержит цепочки транзисторов с р- и n-каналами (на рисунке использованы американские обозначения транзисторов, более простые, чем отечественные). ЛБ — пара из транзисторов разного типа проводимости (выделенный прямоугольник). Между цепочками транзисторов имеются трассировочные каналы, в которых могут быть реализованы необходимые межсоединения элементов.

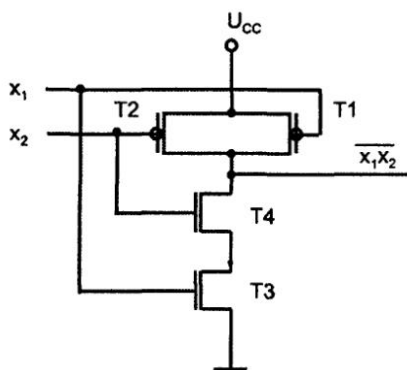


а

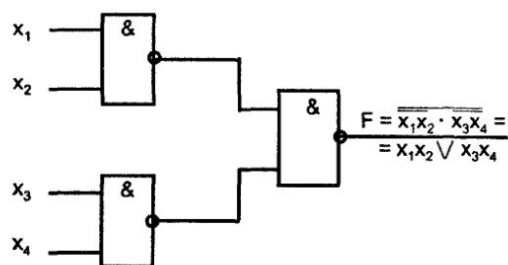
Рис. 8.5. Схема мелкозернистых логических блоков (а)



б



в



г

Рис. 8.5. (окончание) Реализация межсоединений для воспроизведения функции x_1x_2/x_3x_4 (б) и пояснения к этой реализации (в, г)

На рис. 8.5, б показан пример межсоединений, дающих реализацию функции $F = x_1x_2/x_3x_4$. 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. 8.5, в, т. е. ячейки И-НЕ обычного для схемотехники КМОП типа. Эти ячейки соединены между собой как показано на рис. 8.5, 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 используемые в семействе микросхем АСТ фирмы Actel. На рисунке (см. рис. 2.15, а) был показан ЛБ семейства АСТ1, состоящий из трех мультиплексоров "2—1" и элемента ИЛИ, для которого воспроизводимую функцию можно представить следующим образом:

$$F = (\overline{S_0} \vee \overline{S_1})(\overline{S_A} A_0 \vee S_A A_1) \vee (S_0 \vee S_1)(S_B B_0 \vee S_B B_1).$$

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 ко входам переменные и константы соответственно (рис. 8.6), где $S_0 = c$; $S_1 = S_A = B_0 = 0$, $A_0 = A_1 = 1$, $B_1 = a$, $S_B = b$, получим функцию $F = ab \vee \overline{c}$.

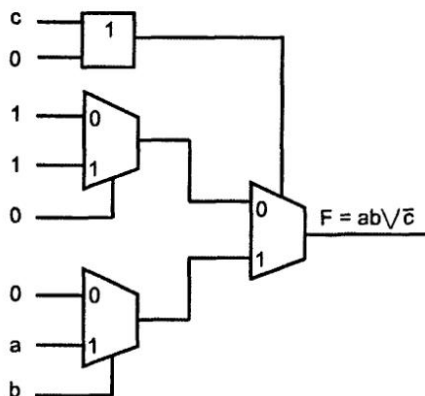


Рис. 8.6. Пример реализации функции $F = ab \vee \overline{c}$ с помощью мультиплексорного логического блока

Крупнозернистый блок семейства XC4000E (рис. 8.7) в качестве основы имеет три табличных функциональных логических преобразователя G, F и H, а также ряд программируемых мультиплексоров (отмечены номерами 1...12 или надписями у выходов) и два триггера.

В FPGA с триггерной памятью реконфигурации, как правило, применяют крупнозернистые блоки. В таких блоках реализуются более сложные функции, что ведет к упрощению программируемой части межсоединений. В то же время труднее полностью использовать логические элементы блоков, что ведет к потерям площади кристалла и быстродействия. Иными словами, меньшая зернистость, можно выиграть в одном и проиграть в другом.

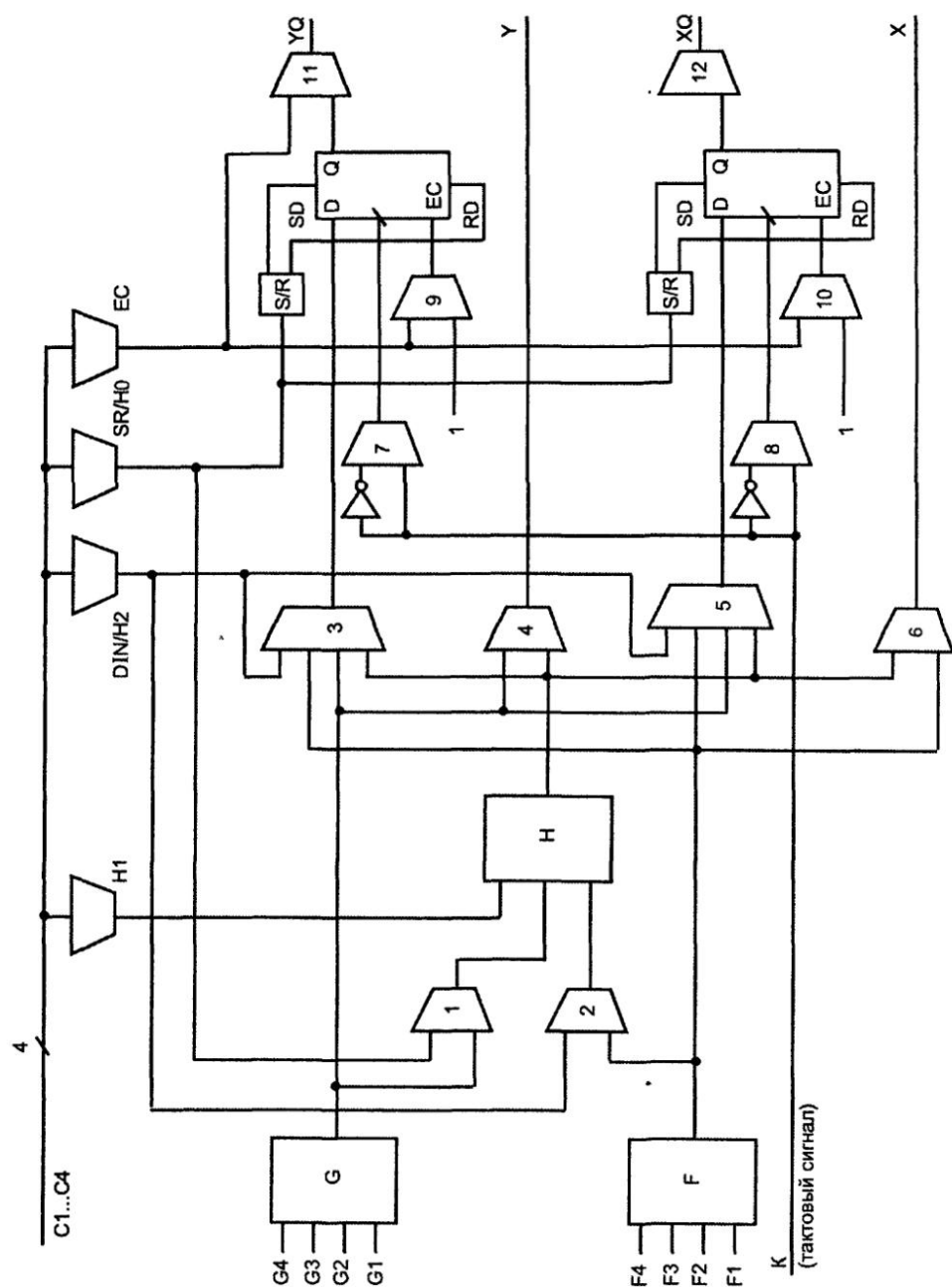


Рис. 8.8. Схема логического блока FPGA XC4000E

Табличные преобразователи представляют собой ППЗУ, для которых аргументы логической функции служат адресом (см. § 4.5). Воспроизводятся любые функции числа аргументов n при организации памяти $2^n \times 1$. Число воспроизводимых функций, т. е. число возможных вариантов программирования ЗУ, составляет 2^{2^n} . Логические преобразователи G и F (блоки памяти с организацией 16×1) воспроизводят функции 4-х аргументов. Их выходные сигналы могут непосредственно передаваться на выходы Y и X при соответствующем программировании мультиплексоров 4 и 6, либо использоваться иным образом. Через мультиплексоры 1 и 2 выходы преобразователей G и F могут быть поданы на входы преобразователя H, если мультиплексоры запрограммированы на передачу сигналов от нижних входов. Кроме того, преобразователь H может использоваться как третий независимый генератор функций со входами H0, H1 и H2, если мультиплексоры 1 и 2 запрограммированы иначе. Входной сигнал DIN может добавляться как дополнительный аргумент и при подаче на преобразователь H выходов преобразователей G и F.

При подаче выходных сигналов преобразователей G и F на вход преобразователя H он воспроизводит функции большего, чем 4 числа аргументов (от 5 до 9, причем для 5 аргументов воспроизводятся любые функции, а для 6...9 лишь некоторые).

В зависимости от программирования мультиплексоров 3 и 5, триггеры принимают данные от логических преобразователей или внешнего входа DIN. Сигналы K тактирования триггеров поступают от общего входа через мультиплексоры 7 и 8, программирование которых позволяет индивидуально изменять полярность фронта, тактирующего триггеры. Сигнал разрешения тактирования EC также поступает от общего входа, но, благодаря мультиплексорам 9 и 10, можно либо использовать сигнал разрешения, либо постоянно разрешить тактирование. Триггеры имеют асинхронные входы установки и сброса (SD — Set Direct и RD — Reset Direct), один из которых через программируемый селектор S/R может быть подключен к выходу коммутатора SR, который, в свою очередь, может программироваться для подключения к любому из внешних выводов ЛБ C1...C4. Это же возможно и для других выходов коммутаторов верхней строки рис. 8.7.

В специальных режимах блоки G и F функционируют как обычные ОЗУ, способные хранить 32 бита данных. Возможна реализация двухпортовых ОЗУ, буферов FIFO и т. д. Память распределена по всему кристаллу.

Блоки ввода/вывода FPGA

Характерные черты блока ввода/вывода рассмотрим на примере семейств XC4000, XC4000E (рис. 8.8). Блок имеет два канала — для ввода сигналов и для вывода. В каждом канале сигналы могут передаваться прямым путем

или фиксироваться в триггерах в зависимости от программирования мультиплексоров 7 и 4. При переводе буфера 1 в третье состояние выходной контакт не должен оставаться разомкнутым, т. к. на "плавающем" высоком входе элементов типа КМОП может накапливаться любой заряд, что может имитировать ввод в схему непредусмотренных сигналов.

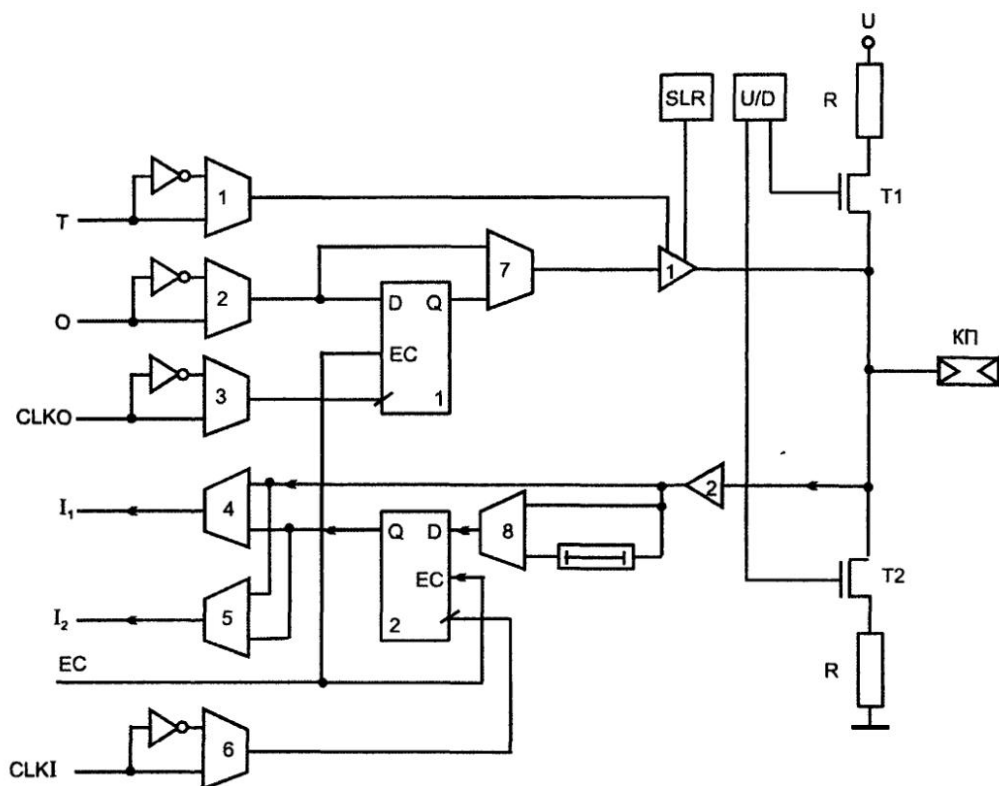


Рис. 8.8. Схема блока ввода/вывода FPGA семейства XC4000E

Благодаря резисторам R потенциал разомкнутой контактной площадки КП либо "подтягивается" к высокому уровню, либо привязывается к нулевой точке. Выбор между этими вариантами программируется элементами памяти конфигурации, имеющимися в схеме U/D (Up/Down). Выходной буфер 1 имеет регулировку крутизны фронта (линия SLR, Slew Rate). Скорости нарастания выходного сигнала можно придать одно из двух значений (быстрая и медленная), для чего имеется программируемый элемент памяти в схеме SLR. Пологие фронты снижают уровень помех, возникающих при работе схемы, и желательны везде, где это приемлемо по соображениям быстродействия. При включении питания во всех буферах устанавливается режим пологих фронтов.

Если внешний вывод работает в режиме входа (буфер 1 в третьем состоянии, буфер 2 активен), то внешний сигнал может подаваться в микросхему либо напрямую, либо через триггер, либо в обоих вариантах одновременно. В последнем случае блок ввода/вывода может демультиплексировать внешние сигналы (например, для шин адресов/данных сохранять адрес в триггере и передавать данные по прямому входу). Синхросигналы триггеров различны для входного (CLKI) и выходного (CLKO) триггера. Их полярности, как и полярность выходного сигнала О (Output), могут программироваться соответствующими мультиплексорами.

Сигнал на входе триггера 2 можно специально задерживать на несколько наносекунд программированием мультиплексора 8. Это сделано для такого подбора временного положения сигнала относительно тактирующего импульса, при котором обеспечивается совместимость с шиной PCI.

Системы межсоединений FPGA

Системы межсоединений (системы коммутации), как и логические блоки, реализуются в широком диапазоне архитектурных и технологических решений. Линии связей в FPGA обычно сегментированы, т. е. составлены из проводящих сегментов (участков, не содержащих ключей) различной длины, соединяемых друг с другом программируемым элементом связи (ключом). Малое количество сегментов ведет к недостаточно эффективному использованию логических блоков, слишком большое — 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 предсказуемость последних, облегчающая проектирование.

Наличие ключей и схем для их программирования усложняют 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. Ключ в линии связи имеет схему замещения в виде RC-звена. В последовательной цепочке RC-звеньев *задержка зависит от числа звеньев квадратично*, поэтому цепи с большим числом ключей в них особенно нежелательны. Может оказаться целесообразным разбиение длинной линии на несколько коротких с помощью промежуточных буферных каскадов.

В разработке FPGA с однократным программированием переключками antifuse ведущую 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. В зависимости от положения входа ЛБ, ближайший канал может находиться выше или ниже данного ряда ЛБ. Выход ЛБ имеет свой вертикальный сегмент, пересекающий несколько каналов (рис. 8.9).

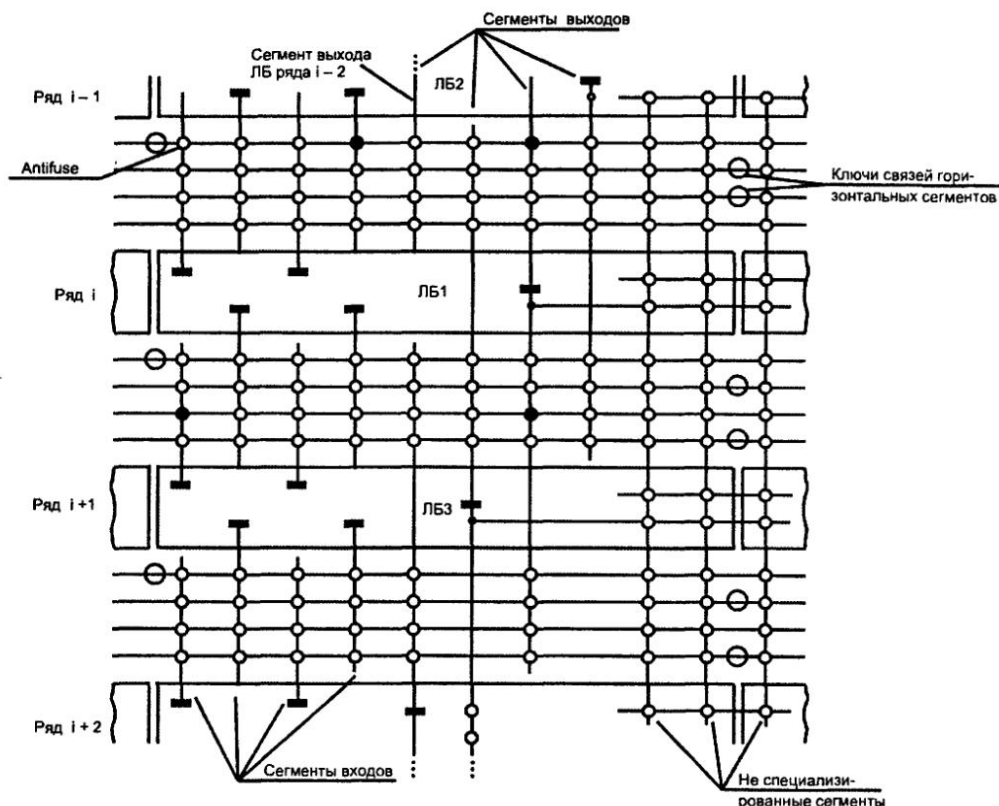


Рис. 8.9. Система коммутации FPGA фирмы Actel (для упрощения рисунка пересечения горизонтальных и вертикальных линий без их соединения показаны просто светлыми кружками, хотя точнее было бы показать их в виде \oplus)

В каждом пересечении сегментов предусмотрена программируемая переключка, 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. 8.9 показаны ЛБ с 4 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.

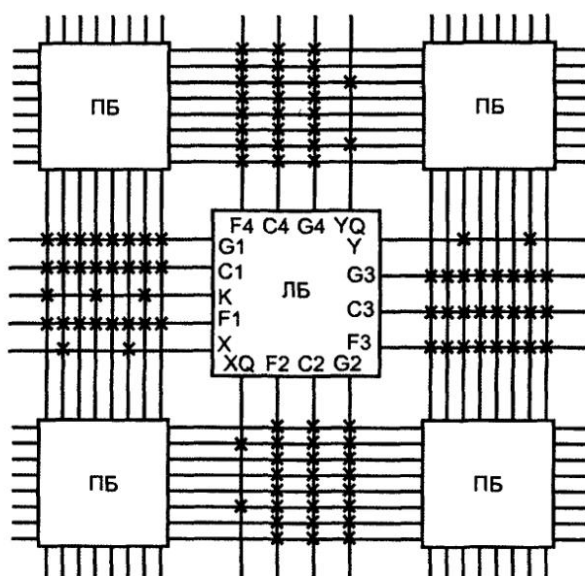
На рис. 8.9 для примера показаны зачерненными кружками те перемычки, которые должны быть запрограммированы для подачи сигнала с выхода ЛБ1 на входы блоков ЛБ2 и ЛБ3.

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.

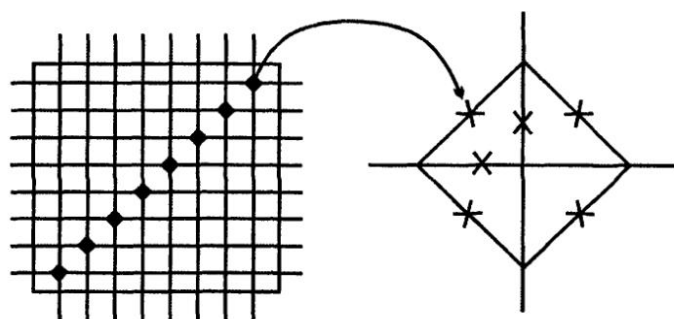
Система межсоединений FPGA фирмы Xilinx — иерархическая, включающая в себя связи общего назначения (General-Purpose Interconnects), длинные линии (Long Lines) прямые связи (Direct Interconnects) линии тактирования (Clock Lines).

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 XC4000E показаны рис. 8.10, а. В этой системе переключательные блоки (переключательные матрицы) расположены на пересечении горизонтальных и вертикальных трассировочных каналов, каждый из которых имеет восемь линий.



а



б

Рис. 8.10. Схема связей общего назначения с линиями одинарной длины (а) и схема переключательного блока (б) 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. 8.10, *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. 8.10, *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. 8.10, *б*), показана отдельно справа. 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.

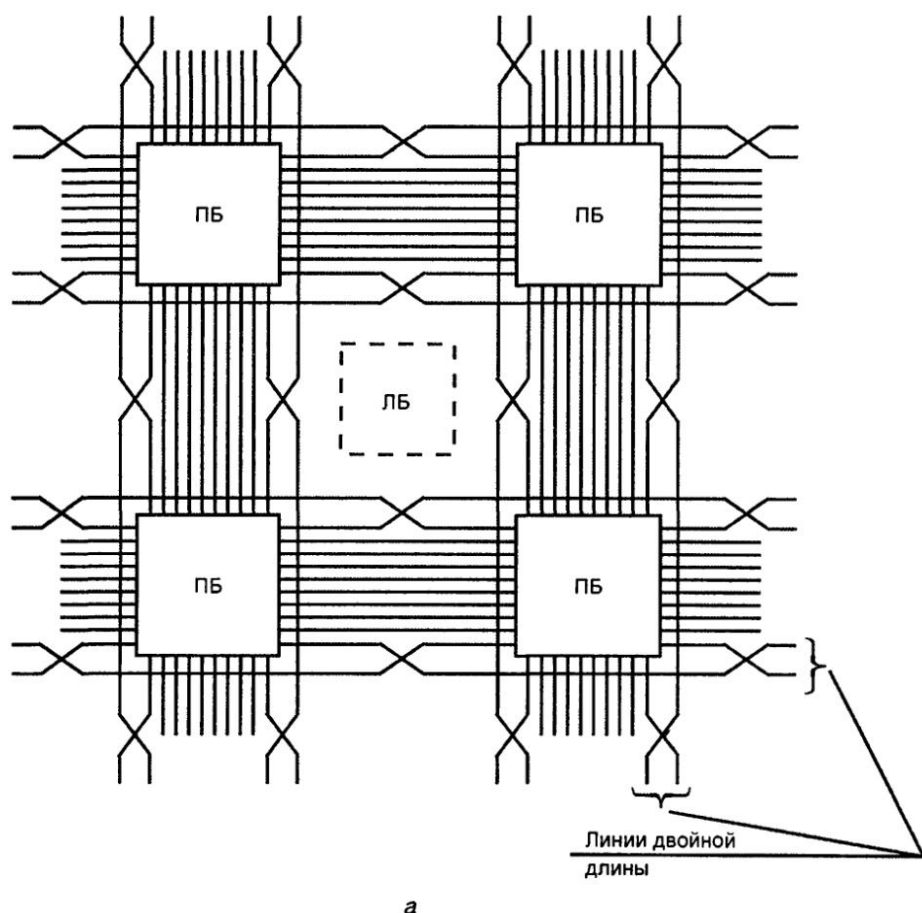
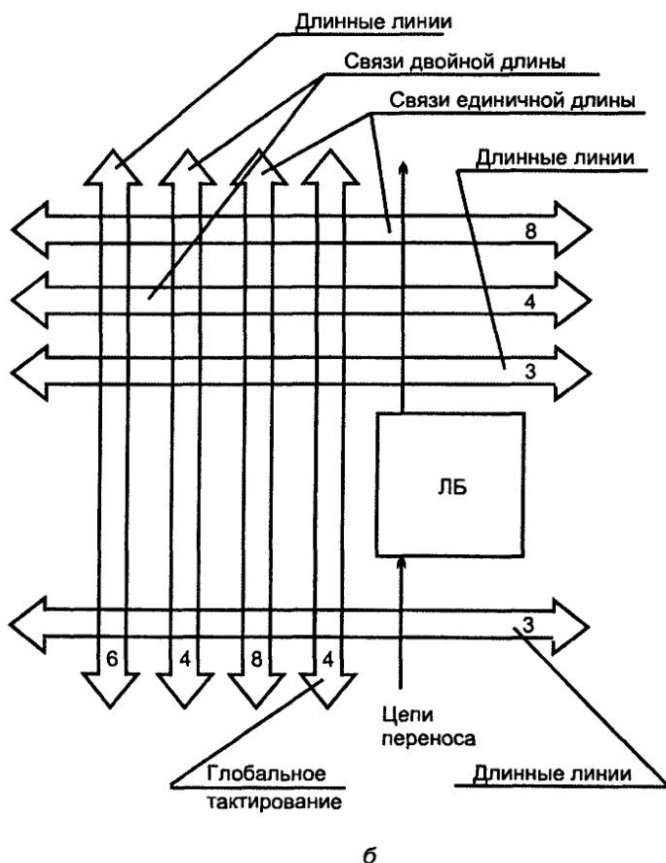


Рис. 8.11. Схема связей общего назначения с линиями двойной длины FPGA XC4000E (*а*)



б

Рис. 8.11. (окончение) Общие ресурсы связей этой микросхемы (б)

Для ускорения и упрощения дальних передач приняты специальные меры. Наряду со связями общего назначения типа рис. 8.10 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. 8.11, а.

Для передач на большие расстояния с очень малой задержкой или для передач на разные приемники с малой расфазировкой сигналов служат длинные линии (здесь термин "длинные линии" имеет прямой смысл, и его не следует путать с аналогичным термином, употребляемым при согласовании волновых сопротивлений). Длинные линии пересекают кристалл вдоль или поперек по всей его длине или ширине (на рисунках не показаны).

В микросхемах семейства XC4000E различают 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. 8.11, б).

Основные параметры FPGA фирмы Xilinx приведены ниже (см. § 8.5, табл. 8.1).

Области применения FPGA и других СБИС ПЛ

Вначале развитие FPGA связывалось с переносом концепции БМК в область малотиражной аппаратуры, но в дальнейшем в связи с появлением репрограммируемости стало ясным, что это нечто большее. Это, в частности, видно из приведенных ниже примеров.

Рассмотренные области применения не являются исключительной прерогативой FPGA, иногда возможно применение других СБИС ПЛ для решения перечисленных задач.

Построение реконфигурируемых систем

В различной аппаратуре встречаются ситуации, в которых те или иные блоки работают поочередно. Например, средства кодирования и декодирования при записи и чтении данных, использующие помехоустойчивые коды. Обе функции никогда не выполняются одновременно. Поэтому не обязательно иметь два устройства (кодер и декодер), а можно иметь одну FPGA с двумя разными конфигурациями, хранимыми в ПЗУ. То есть одна и та же аппаратная часть может выполнять различные преобразования после соответствующей перестройки.

Задачи логической эмуляции

При отладке устройств традиционно пользовались изготовлением прототипа 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 FPGA в задачах логической эмуляции дает сочетание достоинств обоих классических подходов. Система из FPGA 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 FPGA сложнее, чем создание программной модели. Поэтому программные модели не зачеркиваются появлением репрограммируемых FPGA. Следует также помнить, что полные свойства окончательно изготовленного устройства логическая эмуляция на FPGA 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 FPGA 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 FPGA 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 FPGA. 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. FPGA с динамическим реконфигурированием обозначаются DRFPGA (Direct Reconfigurable FPGA).

Сама DRFPGA может иметь практически любое число настроек, рост их количества ограничивается лишь емкостью памяти для их хранения. Устройства с DRFPGA уже используются практически и дают ожидаемый положительный эффект. В таких устройствах требуется быстрая смена настроек. Обычная настройка с введением в FPGA последовательного потока битов или байт-последовательного потока занимает достаточно большое время. В DRFPGA задача решается иначе. В самой системе уже имеется набор загруженных настроек, быстро сменяющих друг друга соответственно требованиям реализуемого алгоритма. Проблемы построения систем на FPGA с динамической реконфигурацией активно исследуются и отражены в литературе (см., например, [54]).

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.

Обогащение цифровой элементной базы

Одним из применений FPGA можно считать обогащение элементной базы цифровых устройств новыми микросхемами типа FPIC (FPID) — Field Programmable Interconnect Circuits (Devices).

Микросхемы FPIC содержат программируемые соединения и блоки ввода/вывода, но не имеют логических блоков. Они предназначены для произвольного соединения своих внешних выводов согласно программированию. Для окончательно изготавливаемых продуктов это не является необходимым, но при отработке прототипов и в системах с динамически реконфигурируемой структурой такие микросхемы бесспорно полезны. Соединяя СБИС ПЛ через FPIC, можно легко изменять их межсоединения, чего не обеспечивают технологии с жесткой трассировкой (печатные платы и др.). Область применения FPIC более узка, чем у таких СБИС ПЛ как FPGA и CPLD, соответственно тиражность их производства ниже и стоимость выше.

§ 8.3. Сложные программируемые логические схемы (CPLD) и СБИС программируемой логики смешанной архитектуры (FLEX и др.)

Сложные программируемые логические ИС (СПЛИС) архитектурно произошли от структур PLD (PAL, GAL) и называются CPLD (Complex Programmable Logic Devices).

Для русского эквивалента этого названия примем СПЛИС, хотя в ряде работ встречается наименование ПЛИС. Следовать этому нежелательно, т. к. многие авторы трактуют термин ПЛИС как наименование всех ИС программируемой логики вообще. Приемлемым вариантом названия для CPLD является и СПЛУ — сложные программируемые логические устройства, что соответствует переводу термина CPLD на русский язык.

Архитектурно CPLD состоят из центральной коммутационной матрицы, множества функциональных блоков ФБ (именуемых также макроячейками, макроэлементами и др.) и блоков ввода/вывода на периферии кристалла. Архитектура CPLD показана на рис. 8.12, где через ПМС обозначена программируемая матрица соединений (PIA, Programmable Interconnect Array).

не будут задействованы, так что система коммутации с единой матрицей в целом требует довольно большого числа ключей.

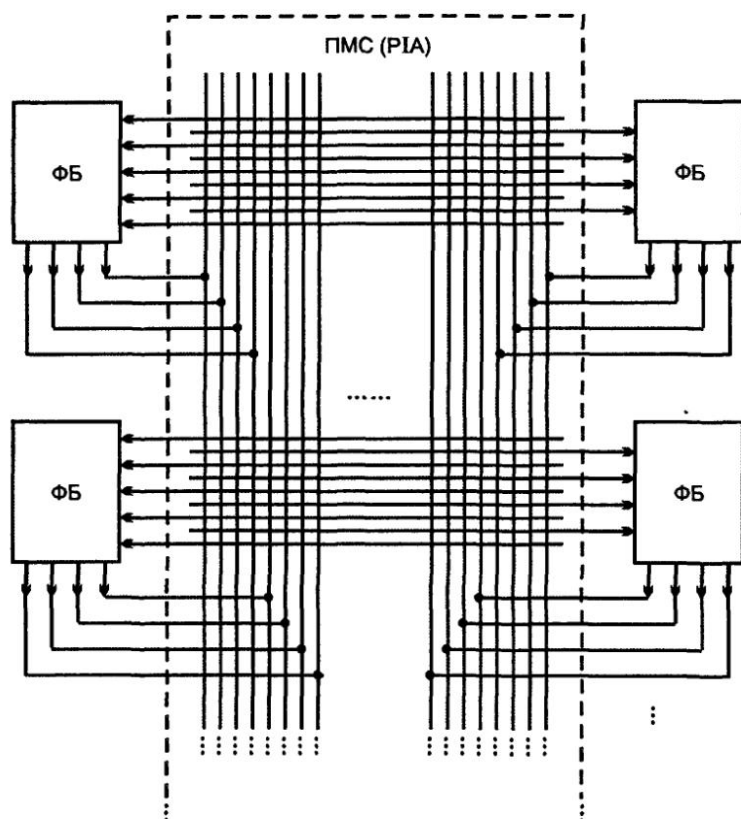


Рис. 8.13. Схема коммутации функциональных блоков CPLD с помощью программируемой матрицы соединений

Типичная ПМС (рис. 8.13) позволяет соединять выход любого ФБ со входами других. Входы ФБ связаны с горизонтальными линиями, пересекающими все вертикальные линии. Любой вход может быть подключен к любому выходу программированием точек связи между вертикальными и горизонтальными линиями. Иначе говоря, ПМС обеспечивает полную коммутируемость блоков.

Внутри самих ФБ может существовать локальная система коммутации, подобная глобальной.

CPLD и СБИС ПЛ смешанной архитектуры производятся многими фирмами. К ведущим фирмам относятся Altera (семейства MAX, FLEX, APEX и др.), Atmel (семейство ATF 1500 и др.), Vantis (ранее была известна как фирма AMD, семейство MACH), Xilinx (семейство XC9500), Philips, Cypress Semicond. и т. д.

В рамках каждого семейства потребителю предлагается ряд представителей, различных по сложности, стоимости и другим параметрам, с целью гибкого обслуживания разнообразных потребностей. Общее число разновидностей CPLD оказывается очень большим. Для более подробного рассмотрения выделим характерные CPLD из продукции ведущих фирм.

Классическими представителями CPLD являются микросхемы MAX 7000, фирмы Altera, имеющие память конфигурации типа и EEPROM. Для "старых" CPLD небольшой сложности триггерная память конфигурации вообще не была характерной. В дальнейшем с освоением глубоко субмикронной технологии и многослойных металлизаций положение изменилось, триггерная память конфигурации появилась и в CPLD, но, строго говоря, одновременно изменилась и их архитектура, так что соответствующие СБИС правильнее относить к СБИС ПЛ смешанной архитектуры (FLEX, APEX), рассматриваемым ниже.

CPLD MAX 7000

На рис. 8.14 показан фрагмент CPLD MAX 7000S, дающий достаточно полное представление о ней, т. к. структура CPLD в целом составляет повторением изображенного на фрагменте яруса из логических блоков ЛБ и блоков ввода/вывода БВВ то или иное число раз (на рис. 8.14 повторяющиеся ярусы должны располагаться сверху вниз) в зависимости от числа ЛБ у данной CPLD, т. е. от ее сложности. На рисунке выделены логические блоки ЛБ, содержащие по 16 макроячеек МЯ, получающих термы от локальных программируемых матриц И (ЛПМИ), программируемая матрица соединений ПМС и блоки ввода/вывода БВВ.

Микросхемы семейства MAX 7000 имеют маркировку EPM7XXX (усовершенствованные варианты отмечаются дополнительной буквой в конце), где в трех последних позициях размещается число МЯ у данной микросхемы (от 032 у младшего представителя до 256 у старшего).

Как и во всех СБИС ПЛ, логические операции производятся в ЛБ, которые соединяются в единую схему с помощью ПМС. Каждый ЛБ содержит 16 макроячеек, так что у младшего представителя семейства 2 логических блока, у старшего — 16.

ПМС обеспечивает возможность подачи на любой вход ЛБ сигнала от любого источника (выходов ЛБ или контактов ввода/вывода), причем она организована так, что на пути сигнала нет программируемых ключей, и задержки сигналов малы. Подача сигнала из ПМС в ЛБ (рис. 8.15) происходит через конъюнктор, открытый по второму входу единичным логическим сигналом с помощью программируемого транзистора, который не находится в цепи передачи сигнала. На вход ЛБ можно передать сигнал с любой вертикальной линии ПМС. Вертикальные линии непрерывны и идут по всей длине между двумя столбцами из ЛБ.

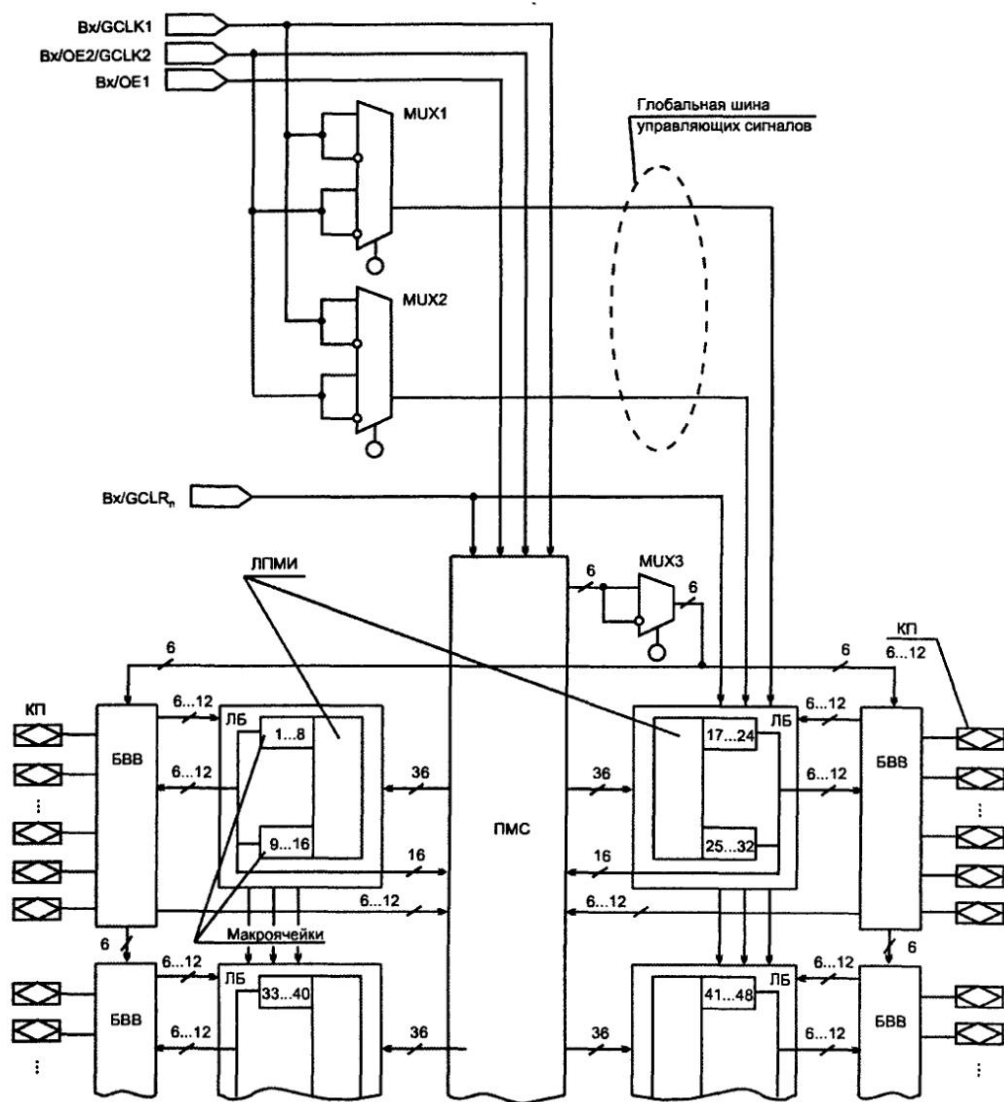


Рис. 8.14. Фрагмент структуры CPLD MAX 7000S

Каждый ЛБ непосредственно связан со своим блоком ввода/вывода, имеющим от 6 до 12 контактов (КП — контактная площадка). Как видно, не все макроячейки могут иметь внешний вывод. Часть из них может быть использована только для подачи сигнала обратной связи в ПМС, что является естественным при построении ряда узлов (счетчиков и др.).

ПМС получает следующие сигналы: 16 сигналов обратной связи от каждого ЛБ, от 6 до 12 сигналов от БВВ, четыре сигнала (вверху на рис. 8.15)

от специализированных входов. Разное число контактов у БВВ позволяет выбирать более экономичный вариант там, где требования интерфейса это допускают.

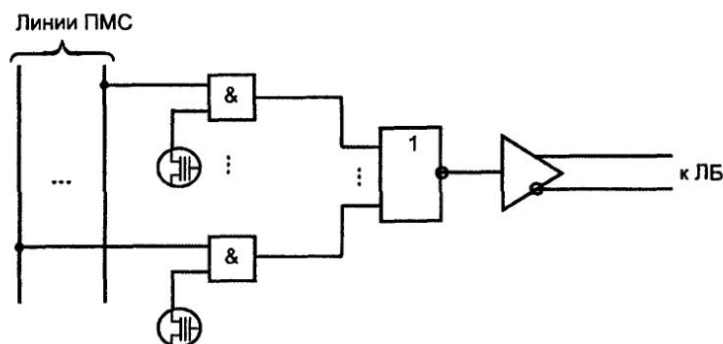


Рис. 8.15. Схема передачи сигналов из программируемой матрицы соединений в логические блоки

К специализированным сигналам относятся так называемые глобальные сигналы тактирования $GCLK1$ и $GCLK2$ и сброса $GCLR$, а также сигналы разрешения выходов $OЕ$. Здесь и в других случаях термин "глобальный" означает "единый для всех одноименных блоков СБИС". При необходимости перечисленные линии могут быть использованы как простые входы ПМС.

Из ПМС поступает по 36 сигналов для каждого ЛБ и еще 6 сигналов, которые передаются в прямом или инверсном виде через мультиплексор $MUX3$ для глобальной шины разрешения выходов блока ввода/вывода.

Логический блок обеспечивает построение как комбинационных цепей, так и схем с элементами памяти. Одна из *макроячеек ЛБ* показана на рис. 8.16. Из матрицы элементов И в матрицу распределения термов МРТ поступает 5 основных термов (на рисунке слева). МРТ дает возможность использовать эти термы для сборки по ИЛИ с последующей подачей результата на элемент сложения по модулю 2 для образования комбинационной функции, а также для управления триггером по входам сброса (CLR_n), установки (PR_n). Терм t может быть использован для тактирования триггера или разрешения тактирования в зависимости от программирования мультиплексора $MUX2$.

Триггер может тактироваться от глобального сигнала $GCLK$ с минимальной задержкой поступления сигнала синхронизации от общего входа микросхемы, что типично при реализации синхронных автоматов. Тактирование от глобального сигнала может сопровождаться индивидуальным управлением от сигнала разрешения тактирования ENA , что характерно для построения аperiodических автоматов, и, наконец, возможно тактирование локальным сигналом от терма t , что соответствует асинхронным схемам.

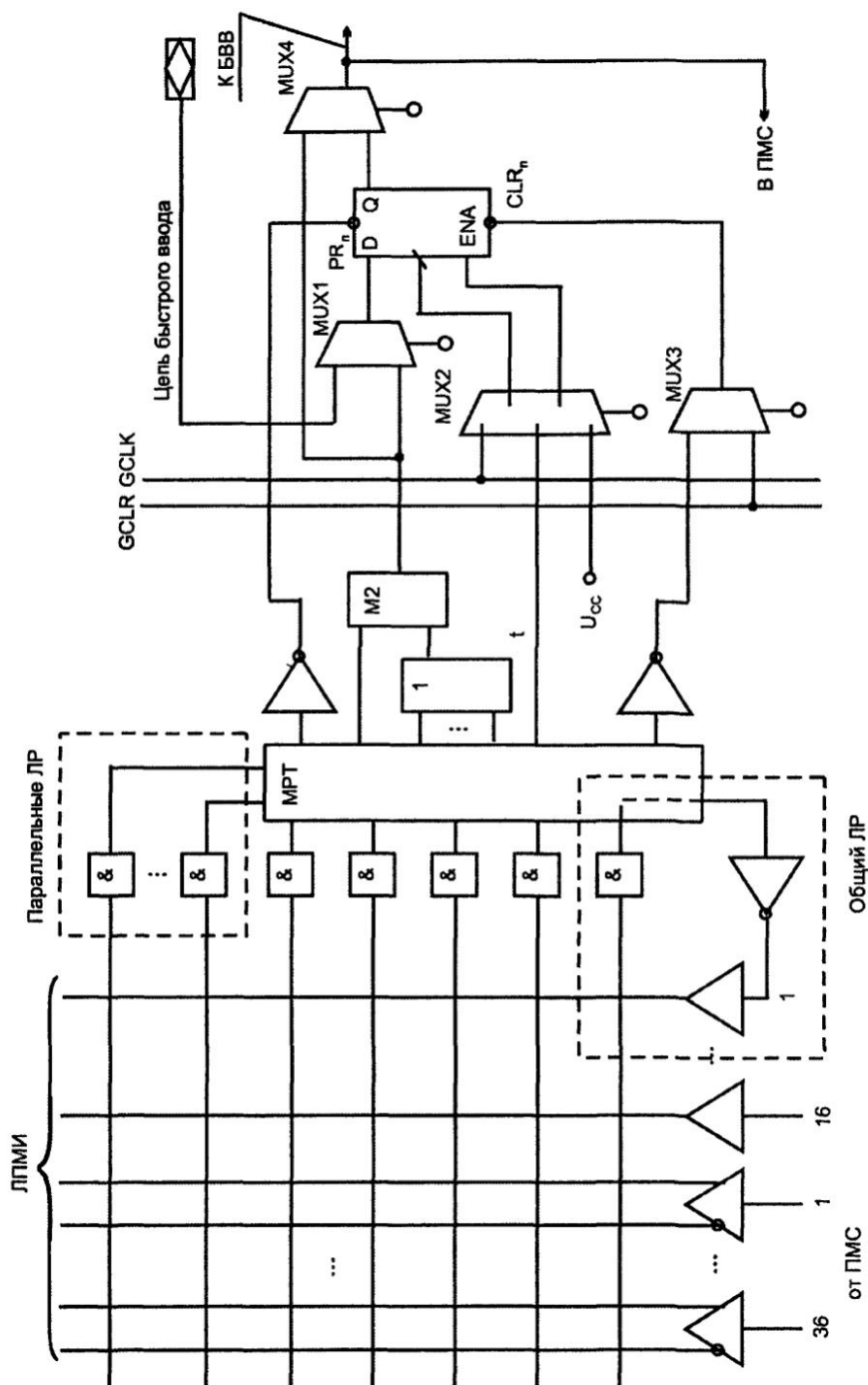


Рис. 8.16. Схема макроячейки CPLD MAX 7000S

Логический расширитель, называемый параллельным, образуется за счет коммутации термов предшествующих макроячеек, которые передаются в последующие (рис. 8.17), так что макроячейки связываются в цепочку. Можно образовать цепочку, длина которой не превышает восьми звеньев. Понятно, что вследствие последовательного включения вентилях ИЛИ в выработку результата вносится дополнительная задержка.

Блок ввода/вывода (рис. 8.18) дает возможность гибкого управления разрешением выходного буфера. ПМС формирует шесть глобальных сигналов разрешения выхода ОЕ. Для некоторых представителей семейства MAX 7000 имеется возможность программирования выхода как выхода с открытым коллектором (ОК), кроме того, может программироваться и скорость изменения выходных сигналов с целью, указанной ранее (эта скорость связана с уровнем создаваемых помех).

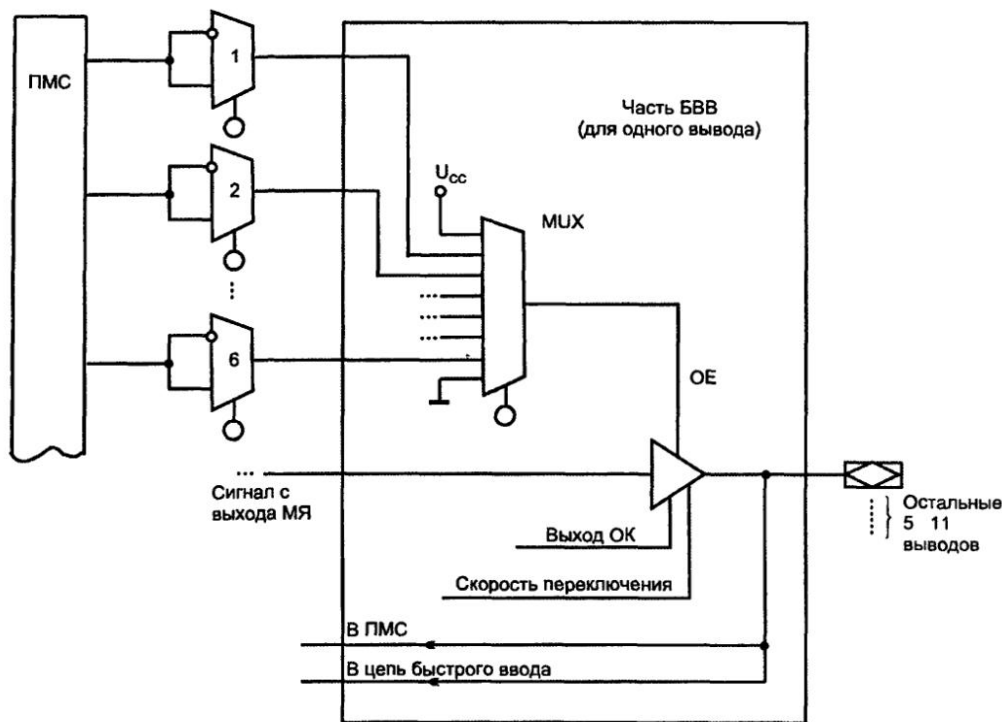


Рис. 8.18. Схема блока ввода/вывода CPLD MAX 7000S

После семейства MAX 7000, принадлежащего ко второму поколению CPLD, фирма Altera выпустила семейство третьего поколения MAX 9000. Между обоими семействами много общего, но третьему поколению присущи и но-

вые свойства: интерфейс JTAG, программирование в системе ISP без применения повышенных напряжений и др. *Сведения о параметрах микросхем MAX 7000 и MAX 9000 приведены в табл. 8.2 § 8.5.*

Развитие архитектур СБИС ПЛ идет по пути создания комбинированных структур, сочетающих достоинства FPGA и CPLD. Так, например, фирма Altera выпустила семейство FLEX 8000 (Flexible Logic Element MatriX) и позднее FLEX 10K с триггерной памятью конфигурации.

По архитектуре микросхемы типа FLEX занимают промежуточное положение между классическими вариантами CPLD и FPGA. Сохранив ряд качеств предшествующих разработок CPLD, микросхемы FLEX в то же время имеют логические элементы табличного типа (LUT), их логические блоки расположены в виде матрицы, причем трассировочные каналы проходят горизонтально и вертикально между ЛБ, что характерно для FPGA. Одновременно с этим трассы в каналах не сегментированы, а непрерывны, что типично для CPLD и дает хорошую предсказуемость и малые величины задержек.

Микросхемы семейства FLEX 10K

Фрагмент структуры микросхем FLEX 10K, по которому можно судить о микросхеме в целом (остальные части являются повторением фрагмента), показан на рис. 8.19. Фрагмент содержит логические блоки, имеющие локальные ПМС и составленные из логических элементов ЛЭ (LE, Logic Element) табличного типа, строки и столбцы глобальной программируемой матрицы соединений ГПМС и элементы ввода/вывода ЭВВ, подсоединенные к концам строк и столбцов ГПМС. Кроме того, микросхема содержит реконфигурируемые модули памяти РМП (ЕАВ). Впервые такая встроенная память появилась в семействе FLEX 10K, до этого ресурсы памяти состояли в тех элементах, которые используются в логических преобразователях табличного типа.

Память может быть организована в вариантах 2048×1 , 1024×2 , 512×4 и 256×8 и ориентирована на реализацию буферов FIFO или (только для FLEX10KE) двухпортовых ОЗУ. Отдельные РМП могут объединяться для создания более емких блоков памяти.

В микросхемах FLEX реализовано двухуровневое разбиение средств логического преобразования данных. Наименьшей структурной единицей, выполняющей логические операции, является логический элемент (ЛЭ, LE). Компактная группа из восьми ЛЭ образует логический блок ЛБ (LAB, Logic Array Block). Логические блоки выступают как самостоятельные структурные единицы на следующем уровне иерархии. Все они расположены по строкам и столбцам, которым соответствуют строки и столбцы глобальной программируемой матрицы соединений ГПМС.

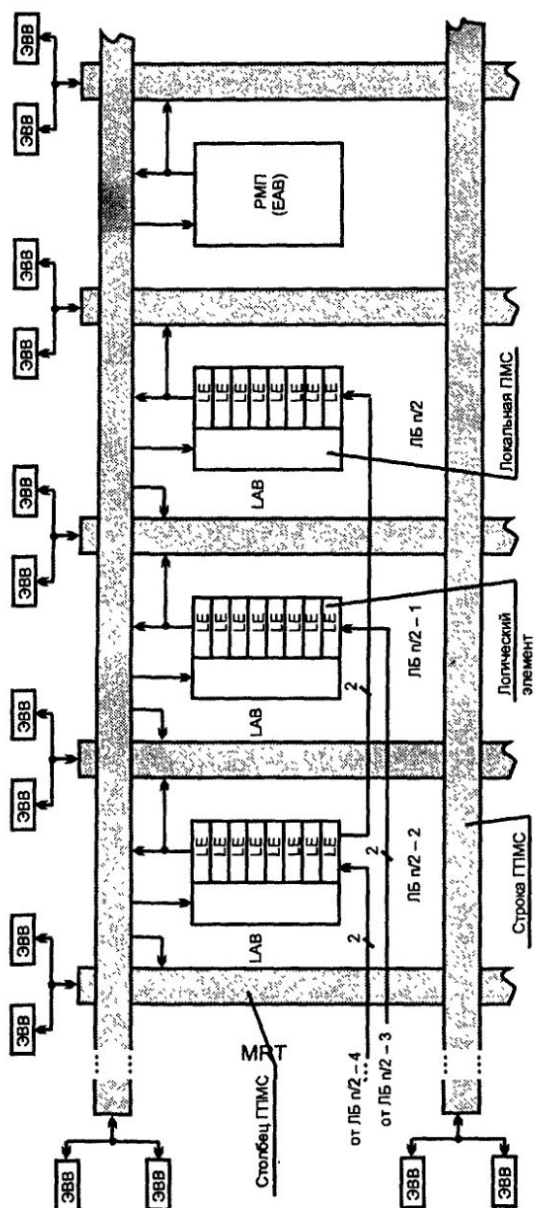


Рис. 8.19. Фрагмент структуры СБИС FLEX 10K

Логический элемент

Логический элемент (рис. 8.20, а) имеет четырехвходовой табличный функциональный преобразователь типа LUT (память емкостью 16 бит), схему переноса, схему каскадирования, программируемый триггер, схему управления сбросом/установкой триггера и несколько программируемых мультиплексоров. Функциональный преобразователь ФП-4 может также работать в режиме воспроизведения двух функций трех переменных (16 бит памяти в этом случае разбиваются на два блока по 8 бит), одна из которых — функция переноса, необходимая для таких схем, как счетчики или сумматоры с последовательными переносами или другие схемы с функциями переноса. В микросхемах FLEX удалось реализовать цепи переноса высокого быстродействия (с задержкой 1 нс на каскад), поэтому в библиотеке функциональных блоков немало рекомендованных структур с последовательным переносом, отличающихся схемной простотой.

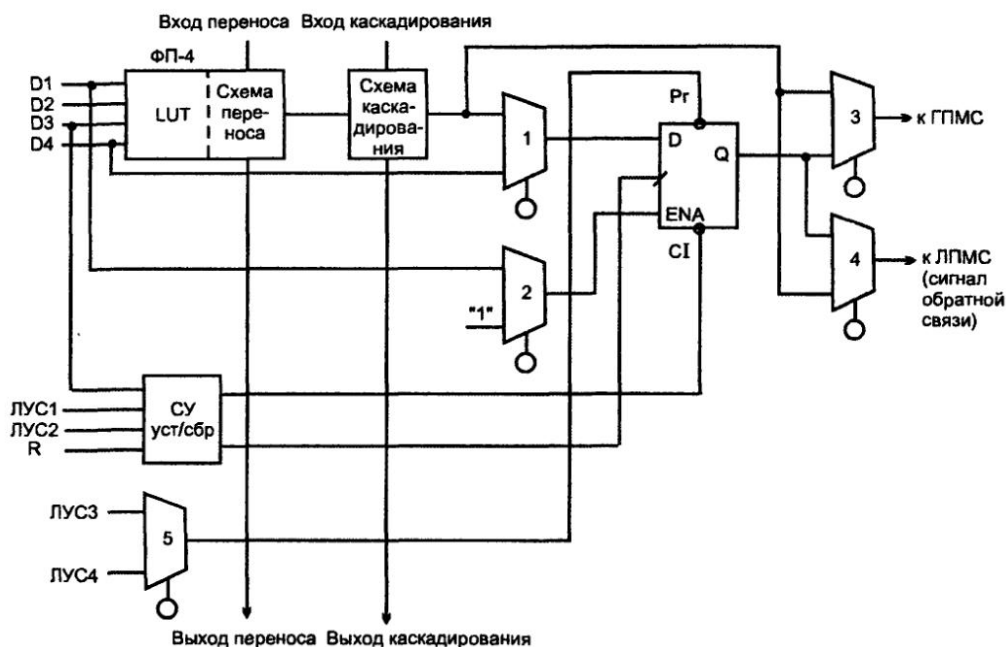
Функциональный преобразователь реализует функции с малыми задержками. Синхронный триггер может функционировать как триггер D, T, RS, JK. Входные сигналы асинхронных сброса и установки вырабатываются схемой управления, в которую поступают два локальных управляющих сигнала ЛУС1, ЛУС2, сигнал общего сброса СБИС и входная переменная D3. В схеме управления установкой/сбросом (СУ уст/сбр) имеются программируемые мультиплексоры, благодаря которым можно задать один из шести режимов воздействия на триггер. Все режимы асинхронные — это операции сброса, установки или загрузки в разных вариантах.

Триггер может быть использован совместно с комбинационной частью логического элемента или независимо, как отдельный элемент, если на его вход через мультиплексор 1 поступает сигнал со входа D4.

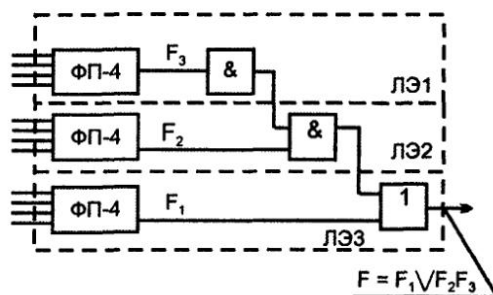
Выходные сигналы ЛЭ через мультиплексоры 3, 4 могут подаваться в глобальную и локальную программируемые матрицы соединений в программируемом комбинационном или регистровом варианте.

Тактирование триггера возможно от любого из двух локальных управляющих сигналов ЛУС3 и ЛУС4.

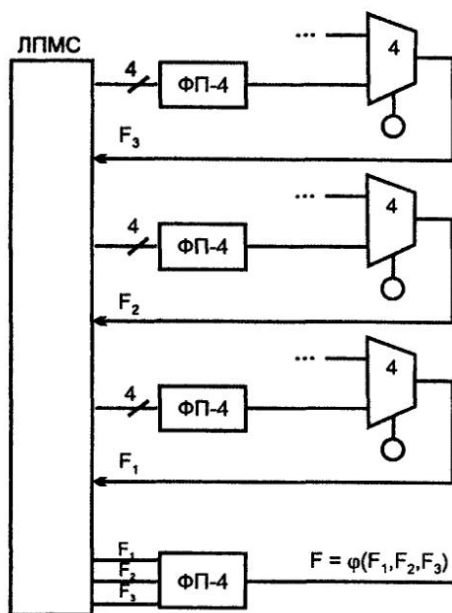
Функции более чем четырех аргументов могут быть получены двумя способами в виде композиций из функций 4-х переменных. Первый способ (рис. 8.20, б) предполагает использование схем каскадирования, имеющихся в каждом ЛЭ. Схему каскадирования можно настраивать на любую логическую операцию, кроме сложения по модулю 2 и равнозначности. С помощью схем каскадирования отдельные функции 4-х переменных объединяются в функцию большего числа аргументов. Второй способ, более обычный для СБИС ПЛ, использует схему с обратными связями (рис. 8.20, в), результатом при этом является получение "функции от функций". Вначале логическими элементами вырабатываются функции, зависящие от не более чем 4-х аргументов, а затем эти функции играют роль аргументов для ЛЭ, формирующего окончательный результат.



а



б



в

Рис. 8.20. Схема логического элемента СБИС FLEX 10K (а) и схемы воспроизведения функций многих переменных (б, в)

Возможности обоих способов соответствуют возможностям декомпозиции воспроизводимых функций.

В СБИС семейства FLEX 10K впервые были включены встроенные реконфигурируемые матрицы памяти РМП (EABs, Embedded Array Blocks) с общей емкостью от приблизительно 6 Кбит до более чем 20 Кбит статической памяти. Такие блоки расположены в центре каждой строки матрицы логических блоков. В каждом блоке имеется 2048 программируемых битов памяти. Возможности блока не ограничиваются его использованием как блока памяти. Блок может быть и функциональным преобразователем табличного типа для получения сложных функций (в области цифровой обработки сигналов, арифметико-логических операций, интерфейсных функций и т. п.). Например, в одном блоке можно реализовать множительное устройство 4×4 , способное работать на частотах до 50 МГц. Если же строить подобное устройство без применения EABs, то потребовалось бы занять 8 логических блоков LAB и максимальная рабочая частота составила бы приблизительно 20 МГц.

Подробности устройства глобальной системы коммутации и работу элементов ввода/вывода рассматривать не будем. В них много сходства с работой схем аналогичного назначения, рассмотренных ранее. *Основные параметры СБИС семейства FLEX 10K даны в табл. 8.3 § 8.5.*

§ 8.4. СБИС программируемой логики типа "система на кристалле"

Уменьшение топологических норм проектирования и ряд технологических усовершенствований довели уровень интеграции современных микросхем СБИС ПЛ до величин в несколько миллионов эквивалентных вентилей, а быстродействие до тактовых частот в 500...600 и более МГц. На таких кристаллах можно разместить целую систему (процессорную часть, память, интерфейсные схемы и др.).

Определение СБИС как "система на кристалле" возникло вследствие двух факторов. Во-первых, из-за высокого уровня интеграции, позволяющего разместить на кристалле схему высокой сложности (систему). При этом разные по функционированию блоки реализуются одними и теми же аппаратными средствами благодаря их программируемости. Такие СБИС обозначаются в англоязычной литературе термином *genepic*. Во-вторых, из-за того, что *СБИС приобретает специализированные области, выделенные на кристалле для определенных функций — аппаратные ядра (Hardcores)*. Системы разного назначения разделяются, тем не менее, на типовые части, что и ставит вопрос о целесообразности введения в СБИС ПЛ наряду с программируемой логикой специализированных областей с заранее определенными функциями.

Введение специализированных аппаратных ядер, имея ряд позитивных следствий, сужает в то же время круг потребителей СБИС, поскольку в сравне-

нии с полностью программируемыми схемами (типа generic) уменьшается их универсальность.

Реализация сложных функций специализированными аппаратными ядрами значительно уменьшает площадь кристалла в сравнении с их реализациями на конфигурируемых логических блоках. Для некоторых аппаратных ядер площадь снижается на порядок, для других меньше. Например, умножитель 8×8 , построенный по модифицированному алгоритму Бута и реализованный методами заказного проектирования, разместился на площади в 5 раз меньшей, чем такой же, реализованный на реконфигурируемых логических блоках, обычных для взятой FPGA.

Таким образом, введение специализированных аппаратных ядер в FPGA и CPLD — процесс противоречивый по результатам. Он сокращает площадь кристалла при реализации сложных функций и ведет к достижению максимального быстродействия, но и таит в себе нежелательные последствия для изготовителя СБИС, т. к. может ощутимо сузить рынок их сбыта, а это ведет к росту цен и потере в какой-то мере конкурентоспособности продукции.

Что же будет преобладать? Здесь ключевой вопрос — какие именно специализированные аппаратные ядра будут выбраны для реализации.

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 уже 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. Многие FPGA уже давно основываются на SRAM-ячейках (обычно на каждый конфигурируемый ЛБ тратится 16...32 бит ОЗУ), и эти ячейки могут быть применены не только для конфигурирования ЛБ, но и организуются в простые ОЗУ, которые могут далее объединяться в более емкие регистровые файлы. Однако такой вариант не дает максимального быстродействия и существенно снижает количество доступной пользователю логики кристалла, т. к. каждый 16...32 бита памяти "выводят из строя" целый ЛБ, т. е. по эквивалентной сложности 10...20 логических вентилей.

В среднем блок ОЗУ с заказным проектированием емкостью 256...512 бит может быть реализован на площади в приблизительно 1/10 от той, которая затрачивается на подобный блок, составленный из распределенных на кристалле ячеек памяти конфигурации. Времена доступа также уменьшаются в 1,5...4 раза.

Области ОЗУ — первые и, безусловно, главные специализированные аппаратные ядра. Других не так уж много. Это умножители, используемые в некоторых СБИС ПЛ, а также схемы интерфейса JTAG. Ядра интерфейса JTAG успешно внедрились во многие СБИС ПЛ, поскольку они выполняют важные функции, нужные очень многим, занимают очень небольшую площадь на кристалле и позволяют достичь высокого быстродействия.

Самыми сложными из практически известных ядер являются контроллеры шины PCI, также необходимые в очень многих приложениях и требующие максимального быстродействия.

Семейство СБИС типа APEX 20K/KE

Перспективы существенного расширения перечня реализованных специализированных аппаратных ядер явно ограничены. Для реализации систем на кристалле фирма Altera выпустила семейство СБИС типа APEX 20K/KE, построенных по архитектуре, названной Multicore. В них комбинируются табличные методы реализации функций и реализации их двухуровневыми структурами, т. е. сочетаются характерные признаки FPGA и CPLD. Имеется встроенная память и гибкая система интерфейсов (рис. 8.21).

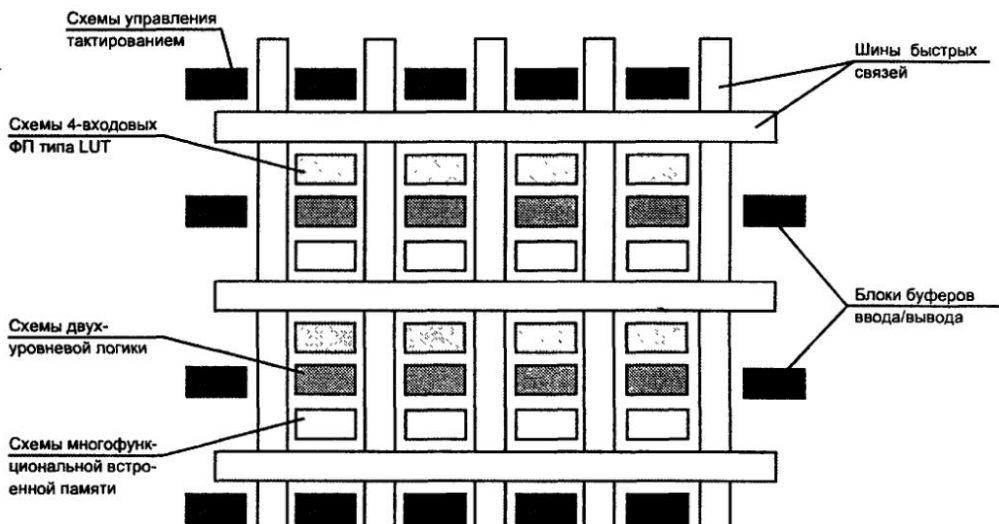


Рис. 8.21. Структура СБИС семейства APEX 20K

Эти СБИС отличались наибольшим среди промышленных СБИС ПЛ объемом встроенной памяти (от 58 до 540 Кбит для разных микросхем). Сложность — от 263 К до 2,67 М системных вентилей. Понятие "системных вентилей" поясняется в § 8.5.

Сочетание средств логической обработки из арсеналов FPGA и CPLD облегчает обработку разнотипных функций (трактов передачи данных, автоматов с памятью). При реализации автоматов задержки выработки функций возбуждения триггеров составляют 4,8 нс, что соответствует работе автомата на частотах около 200 МГц. Ядро ОЗУ состоит из блоков ESB (Embedded System Block) по 2 Кбит, число блоков от 26 (у младших представителей семейства) до 264 (у старших). Блоки могут работать независимо (с варианта-

ми организации 128×16 , 256×8 , 512×4 , 1024×2 , 2048×1) или соединять с другими для образования более емкой памяти. Вместе со схемами близлежащих логических блоков блоки памяти могут образовывать стандартные SRAM, буферы FIFO, двухпортовую память, а в некоторых микросхемах возможна организация ассоциативной памяти CAM (Content-Addressable Memory). Имеются обширные возможности выбора сигналов интерфейса — в семействе "К" это 2,5 В I/O; 3,3 В PCI; LVCMOS (низковольтные схемы КМОП) и LVTTTL (низковольтные TTL). В семействе "KE" выбор сигналов интерфейса значительно шире.

Семейство СБИС типа Virtex

В качестве "истинной программируемой системы на кристалле" фирма Xilinx представляет СБИС ПЛ Virtex. Как и APEX 20K, это кристаллы с мегавентильным уровнем интеграции и большими емкостями встроенной памяти. 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. Понятие "системная частота" поясняется в § 8.5. Достижимый процент использования вентилях оценивается как 90%. Число пользовательских выводов лежит в диапазоне 180...804. Линии ввода/вывода программируются на ряд стандартов интерфейсных сигналов (GTL+, LVTTTL, SSTL3-1 и др.).

Файл конфигурации (для кристаллов с 100 тыс. эквивалентных вентилях) имеет объем 2 Мбита и загружается за менее чем 3 мс, что является малым временем, способствующим применению микросхем в системах с реконфигурацией аппаратных средств.

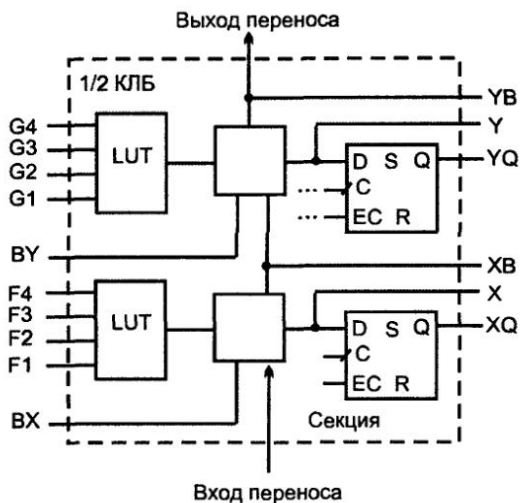
Система межсоединений сохраняет многое из свойственного предыдущим семействам СБИС ПЛ фирмы Xilinx, но имеет и своеобразие. В дополнение к прежним ресурсам межсоединений созданы новые диагональные связи с хорошей предсказуемостью задержек.

Логические ячейки ЛЯ содержат 4-входовые преобразователи типа LUT, схемы переноса и управления СПУ и D-триггеры с общим (глобальным) тактированием и входами сброса/установки (рис. 8.22, а). Для упрощения программного обеспечения средств проектирования ЛЯ соединяются парами в секции (Slice). Две секции составляют конфигурируемый логический блок КЛБ (CLB). Имеются схемы PLL (см. § 3.6), обеспечивающие коррекцию временных соотношений для тактовых импульсов.

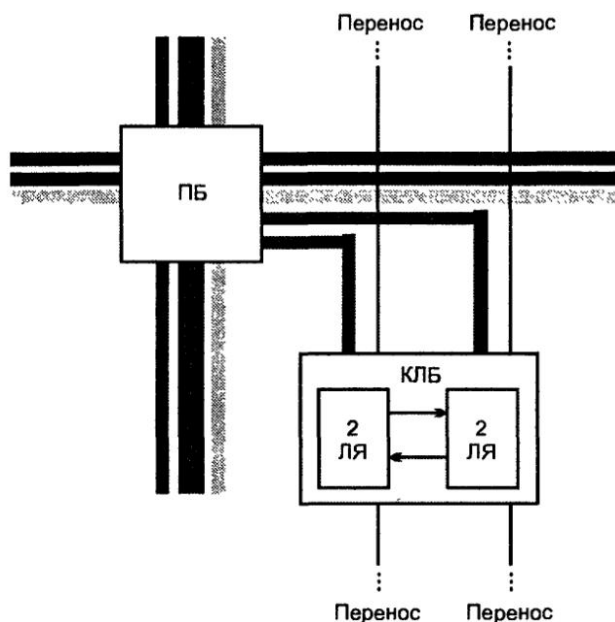
Логические ячейки имеют высокоскоростные схемы переноса для построения каскадных структур. Благодаря им такие схемы, как 32-разрядный счетчик или АЛУ работают на частотах до 100 МГц.

Основа LUT — статическое ЗУ емкостью 16 бит с задержкой выработки функций 1,2 нс. Память может использоваться и по прямому назначению с

возможностью объединения в более крупные блоки. Можно объединять два элемента LUT для воспроизведения функций пяти переменных.



а



б

Рис. 8.22. Схема секции (а) и схема связей КЛБ с системой межсоединений (б)

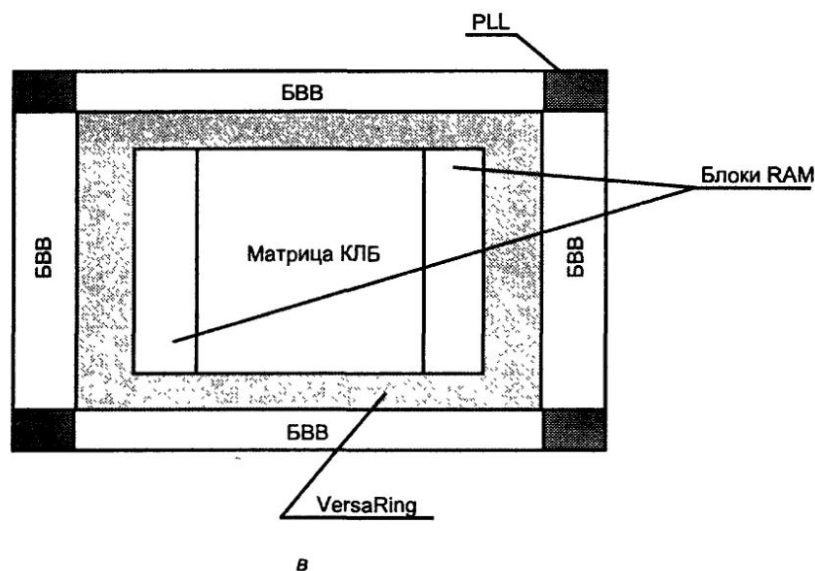


Рис. 8.22. (окончание) Общий план (а) СБИС семейства Virtex

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.

В области памяти содержится набор 2-портовых SRAM по 4 Кбит каждый с несколькими вариантами организации ($4K \times 1$, $2K \times 2$, $1K \times 4$, 512×8 , 256×16). Память синхронная, время цикла 10 нс. Предусмотрены конфигурации памяти типа видеобуфера, буферов FIFO.

Внутренняя область состоит из матрицы блоков (Versa Block), содержащих переключательные блоки ПБ и КЛБ (рис. 8.22, б). Внутри КЛБ созданы быстрые пути трассировки. Между КЛБ образованы связи общего назначения и некоторое количество прямых связей (см. § 8.2). Линии обратных связей обладают малыми задержками, поэтому воспроизведение сложных функций с помощью нескольких LUTs не ведет к большим потерям быстродействия. Прямые связи обеспечивают межсоединения двух смежных КЛБ в горизонтальном направлении, разгружая в известной степени систему межсоединений общего назначения.

С каждым Versa-блоком связан ПБ (GRM, General Routing Matrix), коммутирующий горизонтальные и вертикальные связи. В системе связей есть не только линии одинарной длины, передающие сигналы от одной GRM к другой смежной, но и буферизированные линии передач на расстояние в 6 GRM. По всей длине и ширине кристалла проходят длинные линии.

На периферии кристалла расположены блоки ввода/вывода с буферизированием 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 перекоммутировать связи с внешними выводами, так что при модификации схемы во внутренней области связи с монтажом печатной платы могут не изменяться.

Параметры микросхем семейства Virtex приведены в табл. 8.4 (см. § 8.5). Микросхемы типа "система на кристалле" выпускаются и рядом других фирм: Ultra 39K (Cypress Semicond.), QL (Quick Logic), ispLSI500V (Lattice Semicond.), ProASIC500K (Actel), PSD (WSI) и др.

§ 8.5. Параметры и популярные семейства СБИС программируемой логики

СБИС ПЛ 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, гибкая логика со смешанной архитектурой, схемы типа "система на одном кристалле"), в частности, наличие или отсутствие специализированных областей встроенной памяти.

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 ISP, совместимости со стандартными интерфейсами (в частности, шиной PCI), 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 пояснений.

Уровень интеграции (сложность)

Уровень интеграции СБИС ПЛ оценивается числом эквивалентных вентилях (обычно это вентили 2И-НЕ), размещенных на кристалле. Объективная

оценка сложности не так проста, как может показаться на первый взгляд. Нельзя просто подсчитать число эквивалентных вентилях в СБИС ПЛ, поскольку их в ней может даже и не быть, а имеющиеся блоки могут не разбиваться на такие вентили.

Рекламируемые изготовителями оценки сложности кристаллов до некоторых пор получались по разным методикам, что вносило путаницу в вопросы сравнительной оценки разных СБИС ПЛ. Затем консорциум компаний под названием PREP (Programmable Electronics Performance Corporation) предложил набор эталонных схем для проверки возможностей СБИС ПЛ. Для измерения сложности кристалла каждая схема реализуется в нем в максимально возможном числе раз, что дает оценку сложности СБИС ПЛ числом "помещающихся" в ней эталонных схем. В качестве эталонных схем было выбрано более 10 типовых функциональных узлов (регистры, дешифраторы и т. п.). Для эталонных схем можно определить число эквивалентных вентилях, необходимых для их построения. Таким путем можно выйти и на оценку уровня интеграции СБИС ПЛ числом эквивалентных вентилях. При этом методика PREP четко оговаривает условия измерений, использует усреднение данных по разным эталонным узлам и приводит к более или менее объективной оценке сложности СБИС.

СБИС ПЛ первых поколений имели достаточно однородную структуру и для них указывались два числа эквивалентных вентилях — *полное* (total) и *пользовательское* (usable), т. к. не вся схема кристалла может быть использована системотехником для реализации его проекта. В дальнейшем положение осложнилось, например, в СБИС появились специализированные области памяти и другие структурные неоднородности. Для таких СБИС приводятся несколько чисел для оценки сложности. Под числом *логических вентилях* понимают оценку сложности без учета памяти. Под числом *системных вентилях* понимают оценку, учитывающую и область памяти, сложность которой также пересчитана в число эквивалентных вентилях. Число системных вентилях обычно задается диапазоном, т. к. конкретные числа существенно зависят от типа реализуемого проекта. Поэтому при сравнении СБИС ПЛ по сложности следует обращать внимание на тот смысл, который вкладывается в данном конкретном случае в приводимый показатель "число эквивалентных вентилях". К сожалению, во многих книгах такие цифры приводятся без комментариев.

Быстродействие СБИС

Быстродействие СБИС характеризуется либо 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 счетного триггера (*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.

На рис. 8.23 представлены параметры сложности и быстродействия семейств СБИС ПЛ.

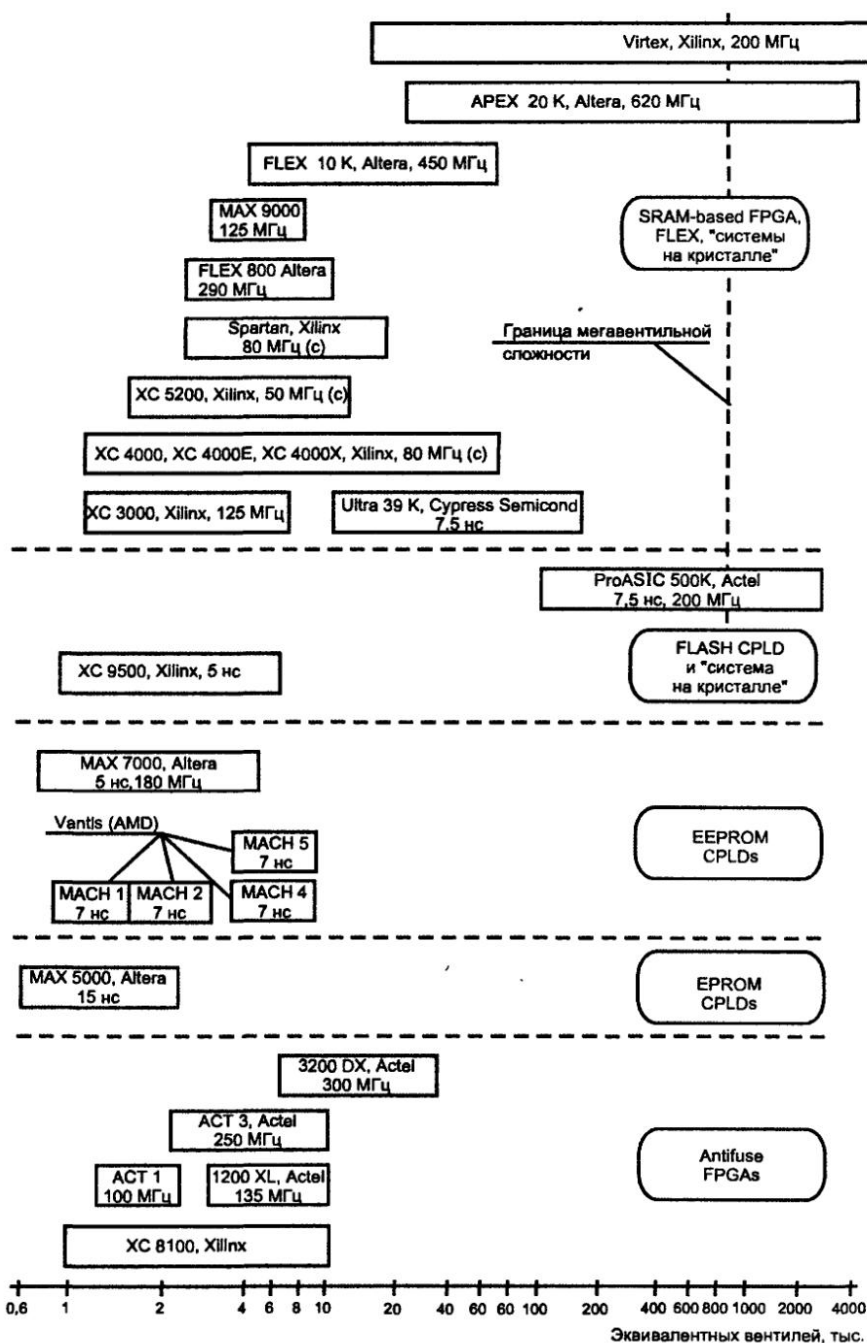


Рис. 8.23. Параметры сложности и быстродействие СБИС ПЛ

Замечание

Ввиду расхождений в методике определения отдельных параметров разными производителями и неполноты сведений в технических описаниях некоторых микросхем приводимые данные следует считать ориентировочными.

В качестве параметров быстродействия даны максимальные частоты работы, причем системные частоты отмечены буквой "с" в скобках. Сложность оценивается полным числом вентилях, для микросхем со встроенной памятью приводятся числа, соответствующие системным вентилям. Границы сложностей для семейства даются в логарифмической шкале.

Нельзя не отметить, что параметры СБИС ПЛ довольно быстро изменяются, т. к. новые их поколения изготавливаются по все более совершенной технологии. Поэтому сведения, подобные приведенным на рис. 8.23, нуждаются в постоянных дополнениях. Тем не менее, такие сведения полезны, т. к. дают общее представление о развитии и возможностях микросхем программируемой логики.

Среди представленных популярных СБИС ПЛ широко известна продукция фирмы Xilinx, которая в 1985 г. выпустила первую в мире FPGA с триггерной памятью конфигурации и с тех пор постоянно развивает архитектуру и схемотехнику СБИС ПЛ.

После первого поколения FPGA XC2000 (на рис. 8.23 не показано) появились схемы второго поколения XC3000 и третьего — XC4000. Продукция фирмы обогатилась также семействами FPGA, ориентированными на определенные области применения. Так, например, семейство XC6200 ориентировано на реализацию сопроцессоров. Это можно возлагать и на FPGA общего назначения, однако их архитектуры создают при этом определенные ограничения, т. к. на интерфейс с центральным процессором тратится значительная часть ресурсов программируемой логики, да и сама архитектура FPGA не особенно приспособлена к потоковым алгоритмам, типичным для сопроцессоров. Этим недостатком лишена архитектура семейства XC6200, предусматривающая встроенный интерфейс с процессором, мелкозернистые ЛБ и др.

FPGA семейства XC8100 не являются репрограммируемыми, их однократное программирование ведется с помощью перемычек технологии MicroVia, позволяющей располагать пробиваемые перемычки между двумя металлизированными слоями, т. е. не в плоскости размещения логических элементов, что значительно сокращает площадь кристалла, имеющего архитектуру "море вентилях". Тем самым совмещаются хорошие трассировочные ресурсы и небольшая площадь кристалла. Семейство является недорогой альтернативой схемам БМК.

Семейство репрограммируемых FPGA XC5200 в сравнении со схемами семейства XC4000 отличается, в частности, меньшей стоимостью.

Новые технологические достижения позволили выпустить в последние годы семейства FPGA с улучшенной архитектурой, являющиеся продолжением развития линии XC4000, а именно XC4000E и XC4000X. Появилась также и серия Spartan.

Семейства XC4000E и XC4000X позволяют реализовать на их основе сложные проекты систем высокого быстродействия, в их составе 20 представителей. Семейство Spartan ориентировано на удовлетворение полного комплекса требований, характерного для построения систем эквивалентной сложностью до 40 тыс. вентиляей.

В новых семействах имеются низковольтные версии (с буквой L в маркировке).

Микросхемы семейства XC9500 относятся к классу CPLD.

Интересно отметить, что при запуске в производство большой серии устройств (свыше 5—10 тысяч) можно заказать у самой фирмы Xilinx реализацию на основе стандартных БМК, что при этих условиях может значительно снижать стоимость продукции.

Другим примером фирмы-производителя классических FPGA является фирма Actel, работавшая с программируемыми элементами antifuse, а недавно выпустившая микросхему с элементами памяти конфигурации типа Flash (семейство Pro ASIC500K).

Высокое качество, компактность и надежность перемычек типа ONO позволили размещать на кристалле большое их число (уже в первых FPGA было до 700 000 перемычек). Это означало возможность увеличения числа сегментов в системе межсоединений, и, тем самым, обеспечения хороших трассировочных ресурсов кристалла. Автоматическое размещение и трассировка проектов осуществимы даже при 90...100% использовании логических элементов и фиксации позиций задействованных выводов.

Фирмой Actel производятся FPGA ACT1, 1200XL (замена семейства ACT2), ACT3, 3200DX с однократным программированием и микросхемы с Флэш-памятью конфигурации, к которым относится семейство Pro ASIC500K, начальная серия которого насчитывает 7 представителей. Семейство рассчитано, в частности, на применения с интенсивным интерфейсом (имеет большое число конфигурируемых блоков ввода/вывода) и отличается потреблением малой мощности. Гибкость возможных применений микросхем семейства обусловлена мелкозернистостью логических ячеек, которые можно рассматривать как трехходовые программируемые вентили с программируемостью каждого входа на прямую или инверсную передачу входного сигнала. В состав ячейки входят элементы 2И-НЕ, мультиплексоры и флэш-ключи, которые конфигурируются как популярные логические элементы или триггеры типа D.

Микросхемы семейства Pro ASIC500K выполнены по технологии с минимальными размерами 0,25 мкм с четырьмя слоями металлизации. Задержка сигнала в логической ячейке и локальной связи не превышает 0,5 нс. Блоки памяти работают на частотах до 133 МГц, цикл FIFO составляет 7,5 нс.

СБИС ПЛ фирмы Altera принадлежат в основном к CPLD и схемам гибкой логики FLEX со смешанной архитектурой. Первыми CPLD были несложные микросхемы семейства Classic, заменявшие собою схемы из 10...20 корпусов СИС при задержках по цепи "вход-выход" не более 10 нс, наличии Турбо-бита и режима засекречивания проекта.

Микросхемы семейства MAX 5000 значительно сложнее, заменяют устройства из нескольких десятков корпусов СИС, архитектура их заложила основы архи-

текстур следующих поколений и близка к архитектуре семейства MAX 7000, подробно рассмотренной в § 8.3.

Семейство MAX9000 продолжило линию развития своих предшественников, но обладает и рядом новых свойств (ISP, интерфейс JTAG, регистровая буферизация входных сигналов, работа в системах со смешанным питанием 5 и 3,3 В, наличие двух линий обратной связи в макроячейке, что позволяет реализовать в ней одновременно комбинационную и регистровую схему и др.).

Семейства типа FLEX имеют смешанную архитектуру. Первое их поколение — схемы FLEX 8000, второе — FLEX 10K, которое рассмотрено в § 8.3.

СБИС ПЛ типа "система на кристалле" представлены уже в продукции нескольких фирм, в том числе фирм Altera (APEX 20K), Xilinx (Virtex). Сюда же можно отнести микросхемы ProASIC500K фирмы Actel и Ultra 39K фирмы Cypress Semiconductor.

Параметры наиболее популярных СБИС ПЛ приведены в табл. 8.1—8.4.

Таблица 8.1. Основные параметры FPGA фирмы Xilinx

Схема	Максимальное число вентилей (без ЗУ), тысяч	Максимальная емкость памяти (без логики), бит	Типичный диапазон числа вентилей (логики + ЗУ), тысяч*	Число КЛБ в матрице	Число триггеров	Число пользовательских входов-выходов
XC4000E/EX/XL						
XC4003E	3	3200	2—5	100 = 10 x 10	360	80
XC4005E/XL	5	6272	3—9	196 = 14 x 14	616	112
XC4006E	6	8192	4—12	256 = 16 x 16	768	128
XC4008E	8	10368	6—15	324 = 18 x 18	936	144
XC4010E/XL	10	12800	7—20	400 = 20 x 20	1120	160
XC4013E/XL	13	18432	10—30	576 = 24 x 24	1536	192
XC4020E/XL	20	25088	13—40	784 = 28 x 28	2016	224
XC4025E	25	32768	15—45	1024 = 32 x 32	2560	256
XC4028EX/XL	28	32768	18—50	1024 = 32 x 32	2560	256
XC4036EX/XL	36	41472	22—65	1296 = 36 x 36	3168	288
XC4044XL	44	51200	27—80	1600 = 40 x 40	3840	320
XC4052XL	52	61952	33—100	1936 = 44 x 44	4576	352
XC4062XL	62	73728	40—130	2304 = 48 x 48	5376	384
XC4085XL	85	100352	55—180	3136 = 56 x 56	7168	448

Таблица 8.1. Основные параметры FPGA фирмы Xilinx (окончание)

Схема	Максимальное число вентилей (без ЗУ), тысяч	Максимальная емкость памяти (без логики), бит	Типичный диапазон числа вентилей (логика + ЗУ), тысяч*	Число КЛБ в матрице	Число триггеров	Число пользовательских входов-выходов
Spartan						
XCS05(XL)	5	—	2—5	100 = 10 x 10	360	80
XCS10(XL)	10	—	3—10	196 = 14 x 14	616	112
XCS20(XL)	20	—	7—20	400 = 20 x 20	1120	160
XCS30(XL)	30	—	10—40	576 = 24 x 24	1536	192
XCS40(XL)	40	—	13—40	784 = 28 x 28	2016	224

* При использовании ресурсов схемы для реализации как логики, так и памяти, предполагается, что в качестве ЗУ работают 20...30% КЛБ.

Таблица 8.2. Основные параметры СБИС семейства MAX фирмы Altera

Схема	Макро-ячеек	Вентилей, тысяч	Триггеров	Число пользовательских входов-выходов	Частота f_{CNT} , МГц
MAX 7000					
EPM7032	32	0,6	—	36	178
EPM7064	64	1,25	—	68	178
EPM7096	96	1,8	—	76	151
EPM7128	128	2,5	—	100	151
EPM7160	160	3,2	—	104	151
EPM7192	192	3,75	—	124	125
EPM7256	256	5	—	164	125
MAX 9000					
EPM9320	320	6	484	168	125
EPM9400	400	8	580	159	125
EPM9480	480	10	676	175	118
EPM9560	560	12	772	216	118

Обе серии FPGA способны работать на системной частоте свыше 80 МГц. Низковольтные версии работают при напряжениях питания 3,0...3,6 В (XC4000E и XC4000X с буквами L) или 3,3 В (Spartan). Данные семейства Spartan справедливы как для схем с обычным питанием (без добавления букв XL, т. е. просто XCS05 и т. д.), так и для схем с добавлением XL в маркировке.

Таблица 8.3. Основные параметры СБИС семейства FLEX 10K фирмы Altera

Схема	Частота $f_{\text{СНТ}}$, МГц	Вентилей, тысяч	Триггеров	Емкость ОЗУ, бит	Число пользовательских входов-выходов
EPF10K10	450	10	720	6144	134
EPF10K20	450	20	1344	12288	189
EPF10K30	450	30	1968	12288	246
EPF10K40	450	40	2576	16384	183
EPF10K50	450	50	3184	20480	310
EPF10K70	450	70	4096	18432	358
EPF10K100	450	100	5392	24576	406
EPF10K130	450	130	7126	32762	470

Таблица 8.4. Основные параметры СБИС типа "система на кристалле" фирмы Altera (APEX) и Xilinx (Virtex)

Схема	Максимальное число системных вентилей, тысяч	Емкость встроенного ОЗУ, Кбит	Емкость распределенного ОЗУ, Кбит	Число пользовательских входов-выходов
APEX 20K				
EP20K100(E)	236	53	—	252
EP20K160(E)	404	82	—	320
EP20K200(E)	526	106	—	382
EP20K300(E)	728	147	—	420
EP20K400(E)	1052	213	—	502
EP20K600(E)	1573	311	—	620
EP20K1000(E)	2670	540	—	780

Таблица 8.4. Основные параметры СБИС типа "система на кристалле" фирмы Altera (APEX) и Xilinx (Virtex) (окончание)

Схема	Максимальное число системных вентиляей, тысяч	Емкость встроенного ОЗУ, Кбит	Емкость распределенного ОЗУ, Кбит	Число пользовательских входов-выходов
Virtex*				
XCV50	58	38	24	180
XCV100	109	41	38	180
XCV150	165	49	55	260
XCV200	237	57	75	284
XCV300	323	64	98	316
XCV400	468	82	153	404
XCV600	661	98	211	512
XCV800	888	114	300	512
XCV1000	1124	130	392	512
XCV1000E	1569	329	392	660
XCV1600E	2189	589 *	497	724
XCV2000E	2542	655	614	804
XCV2600E	3264	749	812	804
XCV3200E	4074	851	1038	804

* Для семейства XCVXXHE показаны лишь микросхемы наивысшей сложности, имеющие напряжение питания 1,8 В. Кроме них имеются шесть микросхем с напряжением питания 2,5 В, сложность которых ниже.

§ 8.6. Интерфейс JTAG. Периферийное сканирование. Программирование в системе (ISP). Конфигурирование СБИС ПЛ

Интерфейс JTAG и периферийное сканирование

Термином JTAG обозначают совокупность средств и операций, позволяющих проводить тестирование БИС/СБИС без физического доступа к каждому ее выводу. Аббревиатура JTAG возникла по наименованию разработчика — объединенной группы по тестам Joint Test Action Group. Термином "периферийное сканирование" (ПС) или по-английски Boundary Scan Testing (BST) называют тестирование по JTAG стандарту (IEEE Std 1149.1). Такое тестирование возможно только для микросхем, внутри которых име-

ется набор специальных элементов — ячеек периферийного сканирования (ячеек ПС), в английской терминологии BSC (Boundary Scan Cells) и схем управления их работой.

Ячейки BSC размещены между каждым внешним выводом микросхемы и схемами кристалла, образующими само проверяемое устройство. Все большее число современных микросхем снабжается интерфейсом JTAG, т. е. возможностями периферийного сканирования.

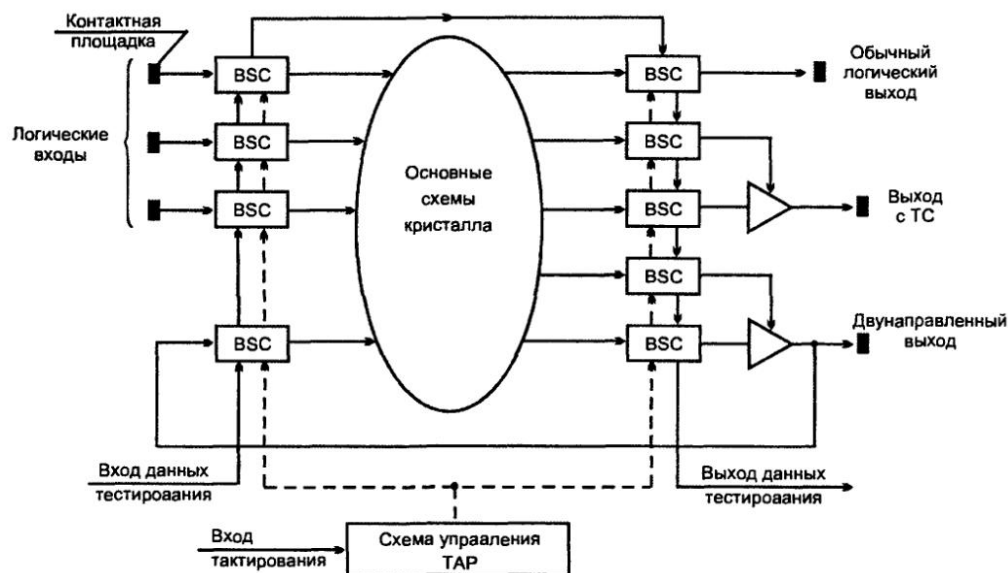
Основная концепция периферийного сканирования иллюстрируется рис. 8.24, а. Ячейки сканирования BSC могут работать в разных режимах. В рабочем режиме они просто пропускают сигналы через себя слева направо и не изменяют функционирования устройства. При этом для выходов обычного логического типа нужна одна BSC, для выходов с третьим состоянием — две (вторая для выработки сигнала управления буфером), для двунаправленных выводов — три, что видно из рис. 8.24, а. Входные сигналы проходят через ячейки BSC прямо к соответствующим точкам основных схем кристалла.

В режиме тестирования пропуск сигналов через ячейки прекращается, а сами они, соединяясь последовательно, образуют сдвигающий регистр, обладающий также некоторыми дополнительными функциями. В такой сдвигающий регистр со входа данных тестирования может быть введен тестовый код для подачи на входные точки основной схемы кристалла. Результат, который выработает основная схема, загружается в ячейки BSC на ее выходах и затем выводится последовательно для сравнения с ожидаемым правильным результатом вне устройства.

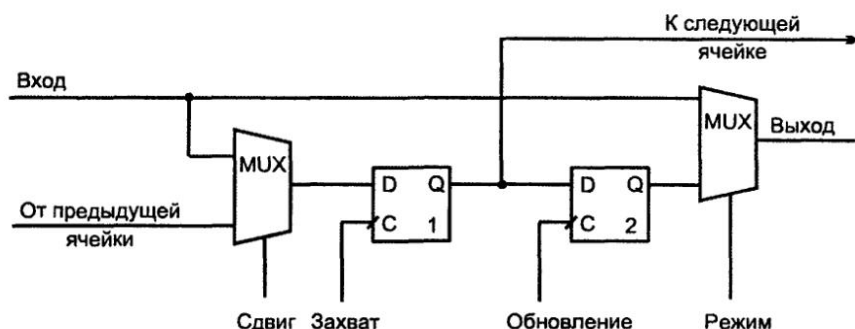
Схема BSC (рис. 8.24, б) содержит два мультиплексора и два D-триггера. В зависимости от адресного входа "Режим" выходного мультиплексора, ячейка либо свободно пропускает сигнал со входа на выход, либо передает на выход состояние второго триггера. Адресный сигнал входного мультиплексора "Сдвиг" управляет подачей на первый триггер входного сигнала (от логических входов микросхемы) или же сигнала от предыдущей ячейки. Таким образом, по синхросигналу для первых триггеров и при передаче через входной мультиплексор сигнала "Вход" осуществляется параллельная загрузка этих триггеров во всех ячейках. При передаче через входной мультиплексор сигнала от предыдущей ячейки тактовый сигнал производит сдвиг на один разряд в регистре, образованном последовательным соединением ячеек.

По синхросигналу "Обновление" текущее содержимое регистра, составленного из цепочки первых триггеров, переписывается в статический регистр, составленный из вторых триггеров. Сдвиги в регистре на триггерах 1 не будут влиять на содержимое регистра на триггерах 2.

Периферийное сканирование позволяет проверять работу самих микросхем, монтажные межсоединения микросхем между собой на печатной плате, считывать сигналы на выводах микросхемы во время ее работы или управлять этими сигналами.



а



б

Рис. 8.24. Структура аппаратных средств интерфейса JTAG (а) и схема ячейки периферийного сканирования (б)

Проверка работы самих микросхем состоит в задании для них входного воздействия и наблюдения за полученным результатом, проверка исправности монтажа микросхем на плате осуществляется, например, при взаимодействии двух микросхем, имеющих JTAG интерфейс. В этом случае тестирующая информация вводится в выходные ячейки одной микросхемы, а затем

переписывается во входные ячейки другой. При исправности всех межсоединений принятая информация идентична введенной.

Порт тестирования ПТ (TAP, Test Access Port) — это 4 (иногда 5) специально выделенных выводов микросхемы. Функциональное назначение этих линий:

TDI (вход тестовых данных) — вход последовательных данных периферийного сканирования. Команды и данные вдвигаются в микросхему с этого вывода по переднему фронту сигнала TCK;

TDO (выход тестовых данных) — выход последовательных данных. Команды и данные выдвигаются из микросхемы с этого вывода по заднему фронту сигнала TCK;

TCK (вход тестового тактирования) — тактирует работу встроенного автомата управления периферийным сканированием. Максимальная частота сканирования периферийных ячеек равна 8 МГц;

TMS (вход управления тестированием) — обеспечивает выбор режима тестирования.

В некоторых случаях к перечисленным сигналам добавляется сигнал $\overline{\text{TRST}}$ для инициализации порта тестирования, что необязательно, т. к. инициализация возможна путем подачи соответствующей последовательности сигналов на вход TMS.

Работа средств обеспечения интерфейса JTAG подчиняется сигналам автомата управления, встроенного в микросхему. Состояния автомата определяются сигналами TDI и TMS порта тестирования. Определенное сочетание сигналов TMS и TCK обеспечивает ввод команды для автомата и ее исполнение.

Порядок операций при периферийном сканировании: загрузка кода команды, загрузка данных, исполнение команды, считывание результата. Подробнее о работе средств управления интерфейсом JTAG говорится, например, в работе [30].

С помощью расширения возможностей интерфейса JTAG можно производить также реконфигурацию микросхем непосредственно в системе, без извлечения микросхем из устройства.

Программирование в системе

Реконфигурация (программирование) в системе — одно из важнейших достоинств СБИС ПЛ, позволяющее легко производить изменения в логике их работы. Потребности в изменениях возникают как для устранения не выявленных при первоначальном тестировании ошибок, так и при модернизации системы (Upgrade). Свойство программируемости непосредственно в системе обозначается аббревиатурой ISP (In System Programmable).

Создание хорошо приспособленной к программированию в системе БИС/СБИС предъявляет определенные требования к ее архитектуре и программному обеспечению средств проектирования. Задача изменения функционирования БИС/СБИС легче решается при использовании в ней мелкозернистых логических блоков (см. § 8.2), наличии схем разделения

термов в CPLD, управлений инверсиями, гибких вариантов установок и тактирования триггеров и т. д.

Возможности программирования в системе растут, если при проектировании часть функциональных возможностей СБИС ПЛ оставлять свободной, имея в виду упрощение изменений проекта в будущем. При этом рекомендуется иметь *запас по скорости, функциональным возможностям и ресурсам межсоединений*. Если первоначальным вариантом проекта заняты более 90% емкости микросхемы и скорость ее работы близка к пределу, то для облегчения последующих изменений целесообразно подумать о применении большей по уровню интеграции или более быстродействующей микросхемы, поскольку ожидать успешного "апгрейда" без запасов ресурсов рискованно.

Следует иметь в виду, что при реконфигурации в системе должно сохраняться назначение внешних выводов, поскольку иначе потребуются изменить монтаж печатных плат.

Требования к числу допустимых для микросхемы циклов репрограммирования

Эти требования зависят от решаемых задач. Для отработки прототипа, работа которого в дальнейшем будет неизменной, достаточным может быть число циклов репрограммирования порядка нескольких десятков, что обеспечивается даже схемами с УФ-стиранием данных. Для многих других применений число допустимых циклов репрограммирования должно быть существенно большим или даже практически неограниченным.

Настройка микросхем программируемой логики

Настройка на требуемый алгоритм функционирования производится с помощью программаторов (например, для PLD) либо непосредственно в системе, что рассмотрено выше. Последнее характерно для СБИС ПЛ с триггерной памятью конфигурации.

Средства конфигурирования СБИС ПЛ с триггерной "теневой" памятью позволяют загружать ее от внешней памяти различными способами. Данные для конфигурирования могут поступать от разных источников (компьютера, ПЗУ, других СБИС ПЛ), форма их представления может быть последовательной или параллельной, роль конфигурируемой СБИС может быть активной или пассивной. В результате возникают *несколько возможных режимов конфигурирования*.

СБИС ПЛ, имеющие интерфейс JTAG, могут конфигурироваться с его использованием (расширенный интерфейс JTAG).

Для иллюстрации общих положений, касающихся конфигурирования СБИС ПЛ, рассмотрим несколько подробнее режимы конфигурирования на примере FPGA семейства XC4000 [30].

Микросхемы семейства XC4000 используют по несколько сотен битов конфигурации на каждый логический блок и его межсоединения. Каждый бит

задает необходимое состояние элементу памяти в схеме логического блока, программируемого мультиплексора, программируемого ключа связи и т. д. Данные конфигурации вырабатываются средствами системы автоматизации проектирования.

Микросхемы имеют три вывода для задания одного из шести режимов конфигурирования. Три режима называются *активными*, в них СБИС ПЛ сама вырабатывает сигналы тактирования и управления для процесса обмена с источником данных конфигурации. В активном последовательном режиме данные конфигурации поступают от последовательной памяти PROM, специально разработанной фирмой Xilinx для этой цели. В активных параллельных режимах данные конфигурации поступают от ПЗУ с восьмиразрядной (байтовой) организацией, а затем в самой FPGA преобразуются в последовательный битовый поток. Активных параллельных режимов два — в одном массив хранимых в ПЗУ данных начинается с нулевого адреса, в другом — с адреса 3FFFF.

В режимах, называемых *пассивными*, сигналы тактирования и управления обменом вырабатываются вне FPGA источником данных. Наиболее популярен режим "пассивный последовательный", в котором поступающие на FPGA данные вводятся в нее по фронтам внешнего тактирующего сигнала CCLK (Configuration Clock).

Два режима называются *периферийными*. В этом режиме FPGA играет роль внешнего устройства микропроцессорной системы, принимая восьмиразрядные данные с шины и регулируя скорость обмена сигналами квитирования RDY/BUSY. Самой FPGA присваивается адрес, и в системе имеется селектор адреса, подключенный к системной шине. Периферийный режим называется асинхронным, если внутренний генератор формирует сигналы тактирования для преобразования байтового потока в битовый, и синхронным, если для этой цели применяются тактирующие сигналы от внешнего источника.

Несколько FPGA с различными конфигурациями можно связывать в последовательную цепочку и использовать единый битовый поток для конфигурирования всей цепочки.

Среди СБИС ПЛ имеются такие, в которых *реализованы одновременно триггерная и энергонезависимая память конфигурации*. В этом случае можно производить конфигурирование (инициализацию) СБИС ПЛ без дополнительных внешних источников путем автоматической загрузки триггерной памяти из энергонезависимой.

Инициализация СБИС ПЛ осуществляется автоматически при включении питания.

Литература к главе: [3], [30], [34], [38], [39], [46], [47], [48], [51], [52], [53], [54], [55], [59].

Глава 9

Методика и средства проектирования цифровых устройств

§ 9.1. Общие сведения

Проектирование — разработка технической документации, позволяющей изготовить заданное устройство в заданных условиях.

Сущность процесса проектирования удачно отражена в работе [10] 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 — функции 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 $\Phi_1 \dots \Phi_K$, 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.*

Синтез неоднозначен. Выбор наилучшего варианта осуществляется по результатам анализа, когда проверяется правильность работы и некоторые показатели, характеризующие устройство.

Декомпозиция функций блоков выполняется до тех пор, пока не получатся типовые функции, каждая из которых может быть реализована той или иной микросхемой.

Процесс проектирования — многошаговый и итерационный, с возвратами назад и пересмотром ранее принятых решений.

К этому наглядному описанию процесса проектирования следует добавить лишь, отражая возможности современной элементной базы, что декомпозиция заканчивается при получении типовых функций, соответствующих тем или иным микросхемам или элементам функциональных библиотек программируемых БИС/СБИС.

Характер проектирования существенно зависит от вида применяемой элементной базы.

Классификация цифровых ИС с точки зрения методов проектирования

Классификация цифровых ИС по признакам, связанным с методами их проектирования, приведена на рис. 9.1.

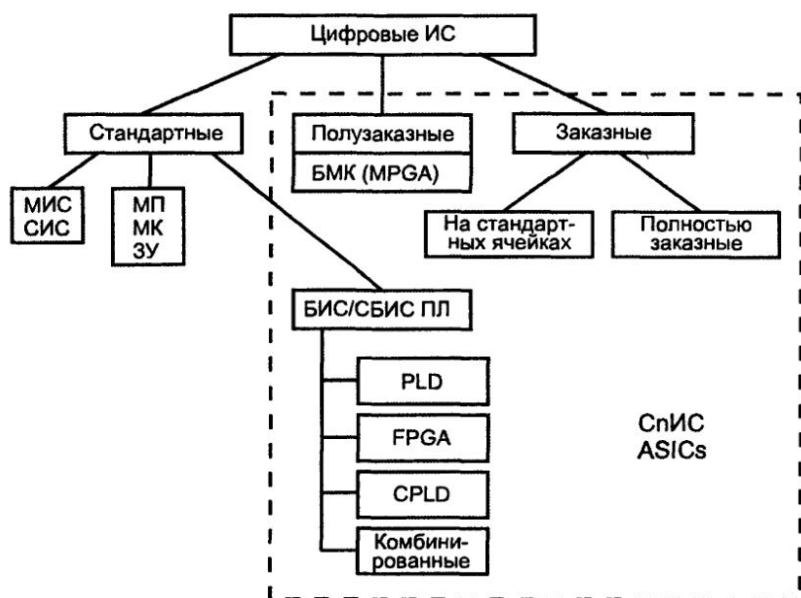


Рис. 9.1. Классификация цифровых ИС по признаку методов проектирования

К *стандартным* микросхемам отнесены схемы малой и средней степени интеграции МИС и СИС. Эти микросхемы производятся массовыми тиражами и реализуют стандартные элементы и узлы, функционирование которых никак не определяется конкретными потребителями. К стандартным схемам высокого уровня интеграции (БИС и СБИС) относятся микропроцессоры МП, микроконтроллеры МК и запоминающие устройства ЗУ, остающиеся неизменными после изготовления независимо от устройств и систем, в которых они используются. Стандартные ИС имеют обширный рынок, что благоприятно для снижения их стоимости.

К *специализированным* ИС (СпИС) относятся все, структура которых в отличие от структур стандартных ИС массового производства каким-либо способом приспособляется к конкретным требованиям того или иного проекта. В английской терминологии СпИС именуются ASICs (Application Specific Integrated Circuits). Среди СпИС различают классы *полузаказных* и *заказных*. Разновидностями заказных микросхем являются полностью заказные и спроектированные методом "на стандартных ячейках".

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.

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 базовые матричные кристаллы БМК (в английской терминологии MPGA, Mask Programmable Gate Arrays). В этом случае имеется стандартный полуфабрикат, который доводится до готового изделия с помощью индивидуальных межсоединений. Реализация требует изготовления лишь малого числа фотошаблонов. Стоимость и длительность проектирования в сравнении с полностью заказными схемами сокращаются в 3...4 раза, но результат еще дальше от оптимального, поскольку в матричных БИС (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 вначале из набора библиотечных элементов подбираются необходимые функциональные блоки, а затем решаются задачи их размещения и трассировки.

САПР для проектирования по методу стандартных ячеек более сложны, чем для проектирования на основе БМК, которому 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.

Особое место в классификации занимают БИС/СБИС ПЛ. С одной стороны, они относятся к СпИС, т. к. в конечном счете приспособляются к требованиям конкретного проекта. В то же время этот процесс (конфигурация схемы) не затрагивает изготовителя, для которого схемы являются стандартным продуктом со всеми вытекающими из этого выгодами.

По мнению специалистов, высказанному в журнале *Electronic Design*, "программируемая аппаратура произведет в ближайшие годы такой же переворот, как микрокомпьютеры в начале 70-х гг."

Области применения СпИС различных типов

Все типы СпИС имеют свои области применения. Каждому типу свойственно определенное соотношение таких параметров как сложность (достижимый уровень интеграции), быстродействие, стоимость. На выбор типа СпИС для реализации проекта влияет совокупность свойств. Основные соображения можно пояснить с позиций экономики, обратившись к формуле стоимости ИС, изготавливаемой уже освоенным технологическим процессом:

$$C_{\text{ис}} = C_{\text{изг}} + C_{\text{пр}}/N,$$

где $C_{\text{изг}}$ — 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; $C_{\text{пр}}$ — 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. Для дорогостоящих вариантов проектирования БИС/СБИС производство становится рентабельным только при большом объеме их продаж.

Затраты $C_{\text{пр}}$ и $C_{\text{изг}}$ — находятся во взаимосвязи. Рост затрат на проектирование, как правило, ведет к снижению $C_{\text{изг}}$, 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 тиражности их производства N и сложности.

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. 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 долларов, а ЗУ емкостью в 1 Мбит — в 395 млн долларов.

Диаграмма областей целесообразного применения разных типов СпИС в зависимости от их сложности и тиражности, приведена на рис. 9.2.

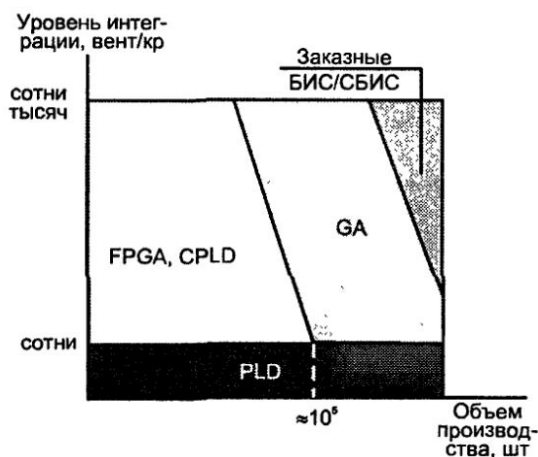


Рис. 9.2. Диаграмма областей целесообразного применения различных типов специализированных БИС/СБИС

Интересно отметить, что ведущая компьютерная фирма IBM использует методологию проектирования смешанного типа БМК/СЯ (Gate-Array/Standard Cell Intermix), размещая в одной СБИС области, выполненные по методу стандартных ячеек СЯ и по методу БМК. Более плотные и быстродейст-

вующие схемы типа СЯ используются в критических трактах обработки сигналов, остальная площадь занимается транзисторами БМК.

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. *Соответствующий материал изложен в гл. 1, 2 и 3.*

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 гл. 4, 5 и 6.*

Львиной долей инженерных разработок аппаратуры в условиях современной России по-видимому явится *использование программируемой логики* для создания требуемых устройств и/или их отладки. Этому вопросу следует уделить дополнительное внимание.

Проектирование на основе схем программируемой логики высокой сложности выполняется только с помощью систем автоматизированного проектирования САПР. Укрупненная *структура алгоритмов проектирования* показана на рис. 9.3.

Проектирование на концептуальном уровне возлагается на проектировщика и слабо связано с автоматизацией. На этом уровне по существу определяется требуемое функционирование устройства, множества входных и выходных сигналов, их характер и взаимосвязь, разбиение проекта на части и т. д. Результаты концептуального синтеза вводятся в САПР, которая производит компиляцию проекта, т. е. синтезирует устройство в базисе библиотеки своих моделей. Полученный проект требует тщательной проверки, поэтому за этапом синтеза следует этап анализа, проводимого моделированием и теоретической верификацией. Моделирование имеет несколько уровней с разной степенью отображения свойств реального объекта. Оно может быть функ-

циональным, проверяющим правильность логической структуры устройства, временным, учитывающим задержки сигналов в схемах устройства без учета окончательной топологии трассировки, и т. д. В результате моделирования могут выявиться ошибки, требующие исправления, что придает процессу проектирования итеративный характер с возвратами к прежним этапам и введением в проект нужных коррекций.



Рис. 9.3. Укрупненная структура алгоритмов автоматизированного проектирования цифровых устройств на основе микросхем программируемой логики

Далее производится конфигурирование микросхемы программируемой логики, после чего возможна реальная проверка работы устройства — физическое моделирование проекта. При успешном завершении физического моделирования устройство готово к установке в систему.

§ 9.2. Пример "ручного" проектирования цифрового устройства с использованием программируемой матричной логики (ПМЛ)

Пусть требуется спроектировать восьмиразрядный двоично-кодированный счетчик с управляемым модулем счета, величина которого задается восьмиразрядным входным кодом N . Такой счетчик считает входные сигналы от нулевого до $N - 1$, а при появлении N -го сигнала автоматически сбрасывается в ноль, после чего цикл повторяется снова. Счетчик должен иметь сигналы асинхронного сброса R , разрешения счета V (работать при $V = 1$), выдачи параллельного кода OE и тактирования CLK . Внешняя организация счетчика приведена на рис. 9.4, а.

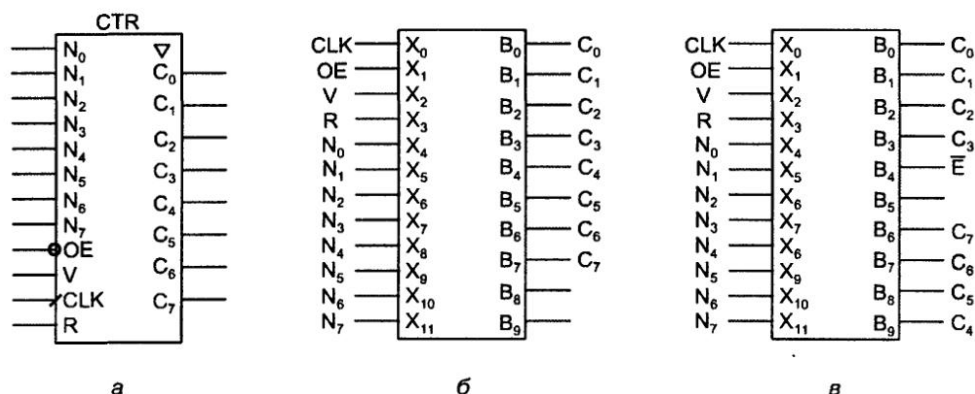


Рис. 9.4. Внешняя организация проектируемого счетчика (а) и варианты распределения сигналов по контактам реализующей его микросхемы (б, в)

Средством реализации счетчика выбрана PLD 85C22V10 (рис. 9.5, а). Это быстродействующая PLD с десятью макроэлементами, имеющая тактовую частоту до 71,4 МГц при использовании обратных связей и 100 МГц при их отсутствии. В макроэлементы поступают от 5 до 16 термов. Кроме того, матрица И вырабатывает отдельные термы для сигналов разрешения буферов OE . Общими для всей PLD являются термы выработки сигналов асинхронного сброса AR и синхронной установки SP . Наличие уровня логической единицы на выводе SP приведет к тому, что очередной тактовый импульс переведет все триггеры в состояние 1. При появлении единичного сигнала AR все триггеры немедленно сбросятся.

Микросхема имеет 12 специализированных входов и 10 двунаправленных выводов.

В схеме макроэлемента (рис. 9.5, б) вывод может быть запрограммирован как вход или как комбинационный либо регистровый выход с возможностью инвертирования выходных сигналов в зависимости от программирования мультиплексора "4—1" с помощью транзисторов ЛИЗМОП, обеспечивающих сигналы S_0 и S_1 . Программируется также тип обратной связи с помощью мультиплексора "2—1", имеющего сложную схему управления, вырабатывающую окончательное значение адресующего сигнала как сумму по модулю два величин S_1 и S_2 .

При программировании на режим комбинационного выхода сигнал обратной связи посылается в матрицу И с контакта Vx/Vyx . При программировании регистрового выхода можно использовать сигнал обратной связи с выхода триггера или с контакта Vx/Vyx .

Возможные конфигурации макроэлемента МЭ даны в табл. 9.1, где приняты обозначения: R — регистровый, C — комбинационный, P — снимаемый с контакта Vx/Vyx (от английского Pin), H — H-активный, L — L-активный.

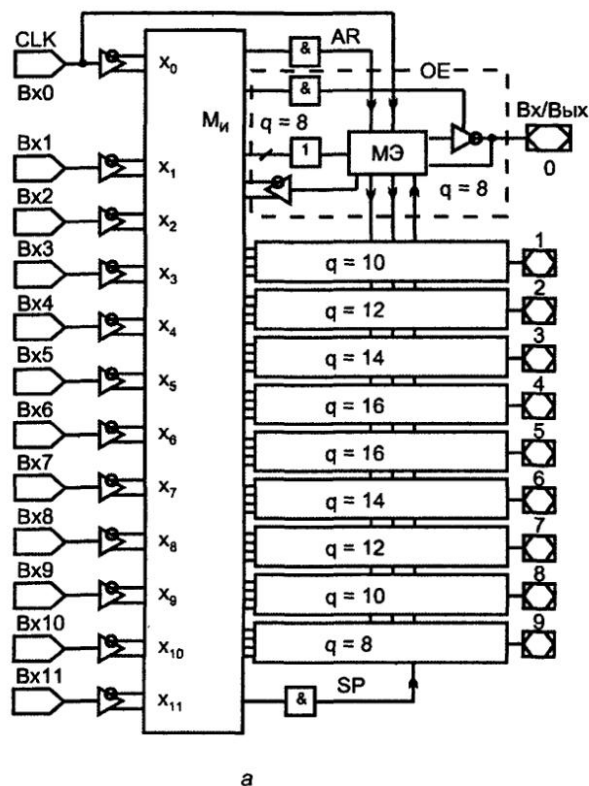


Рис. 9.5. Структура PLD 85C22V10 (а) и схема макроэлемента этой PLD (б)

Таблица 9.1

S_2	S_1	S_0	Выход (тип/полярность)	Обратная связь
0	0	0	R/L	R
0	0	1	R/H	R
0	1	0	C/L	P
0	1	1	C/H	P
1	0	0	R/L	P
1	0	1	R/H	P
1	1	0	C/L	R
1	1	1	C/H	R

Возможен выбор полярности тактирующего сигнала индивидуально для каждого МЭ.

Управление третьим состоянием буфера задается термами, получаемыми от матрицы И.

На рис. 9.5, а раскрыта схема только для одного выхода. Схемы для остальных выходов аналогичны показанной во всем, кроме числа термов, поступающих в МЭ из матрицы И. Число термов q для каждого выхода указано в прямоугольнике условного обозначения.

Первый этап проектирования

На первом этапе проектирования выясняется достаточность числа входов и выходов выбранной PLD для реализации проекта. Для этого производят предварительное распределение сигналов между контактами микросхемы. В данном случае, очевидно, что выходы счетчика должны быть определены на любых 8 из 10 контактов типа Вх/Вых. Наличие двух свободных контактов типа Вх/Вых создает возможность варьирования распределением оставшихся сигналов, т. к. их 12, а свободных контактов 14. Предпочтителен вариант, в котором контакты типа Вх/Вых остаются свободными, поскольку в случае необходимости реализации функций с числом термов больше 16 потребуется дополнительная логика и можно будет воспользоваться ресурсами свободных МЭ.

Входной сигнал CLK однозначно закрепляется за контактом X_0 , т. к. только с него подаются синхросигналы на триггеры МЭ. Таким образом, предварительный вариант распределения сигналов по контактам PLD приобретет вид (см. рис. 9.4, б).

Второй этап проектирования

На втором этапе определяются функции возбуждения для всех 8 триггеров счетчика.

Счетчик должен сбрасываться при совпадении его содержимого с входным кодом N . Согласно логике работы счетчика, функции возбуждения триггеров могут быть представлены в виде:

$$D_i = \begin{cases} 0, & \text{если } Q_k = N_k \text{ и } V = 1; \\ Q_i, & \text{если хотя бы один } Q_j = 0 \text{ или } V = 0; \\ \bar{Q}_i, & \text{если все } Q_j = 1, V = 1 \text{ и хотя бы один } Q_k \neq N_k, \end{cases}$$

где $i, j, k = 0 \dots 7$ — номера разрядов счетчика, причем $j < i$, а k — любое; D_i — вход i -го триггера; Q_i — его выход.

Отсюда:

$$D_i = [Q_0 Q_1 \dots Q_{i-1} \bar{Q}_i \vee (\overline{Q_0 Q_1 \dots Q_{i-1}} \vee \bar{V}) Q_i] \bar{E} V,$$

где $E = (\overline{N_0 \oplus Q_0})(\overline{N_1 \oplus Q_1}) \dots (\overline{N_7 \oplus Q_7})$ — признак равенства содержимого счетчика и входного управляющего слова.

Преобразовав выражение для D_i в ДНФ, получим:

$$D_i = Q_0 Q_1 \dots Q_{i-1} \bar{Q}_i \bar{E} V \vee \underbrace{\bar{Q}_0 Q_1 \bar{E} \dots \bar{Q}_{i-1} Q_i \bar{E}}_{i \text{ слагаемых}} \vee Q_i V,$$

где $i = 1 \dots 7$ и $D_0 = Q_0 \bar{V} \vee \bar{Q}_0 V \bar{E}$.

Как видно, число термов в функциях возбуждения зависит от номера разряда. Максимальное число 9 требуется для старшего разряда. Для следующих нужны 8, 7, 6 и т. д. термов.

Следовательно, функцию возбуждения старшего разряда нельзя получить с выходов 0 и 9, макроэлементы которых имеют по 8 термов.

Третий этап проектирования

На третьем этапе функции с большим числом термов, превышающим возможности выходных каналов PLD, которые не могут быть воспроизведены в ДНФ, разбиваются на подфункции. Реализовав подфункции на отдельных выходах, эти подфункции через цепи обратных связей вводят в матрицу И в качестве аргументов для формирования функции в целом (см. § 7.2). В нашем примере функций с неприемлемым для реализации в ДНФ числом термов не оказалось. Заметим, что это благоприятное обстоятельство возникло из-за того, что в преобразованном выражении для D_i не оказалось

функции E и присутствует только \bar{E} . Действительно, реализация операции $a \oplus b = \bar{a}b \vee a\bar{b}$ требует двух термов, а в функцию E входит произведение 8 таких сумм с независимыми переменными, что дает 256 термов. В то же время реализация инверсного значения \bar{E} имеет вид $\bar{E} = (N_0 \oplus Q_0) \vee (N_1 \oplus Q_1) \vee \dots \vee (N_7 \oplus Q_7)$ и требует всего 16 термов, что реализуемо в форме ДНФ на двух макроэлементах PLD.

Для завершения разработки узла определим термы для сигналов сброса/установки триггеров и термы управления выходными буферами. Входы сброса триггеров AR соединим со входом R , а всем сигналам управления буферами придадим значения OE_1 . Выходной буфер канала выработки \bar{E} должен быть постоянно включен и для него $OE_{\bar{E}} = 1$. Сигнал установки триггеров не используется, и его значение делаем нулевым.

Как отмечалось в § 7.2, нулевой терм получают подключением к его входам одновременно прямого и инверсного значений любого сигнала, а единичный — отключением всех входов.

Единичное значение выходной функции (дизъюнкции термов) получают установкой любого из ее термов в 1.

Замечание

В логических элементах (И, ИЛИ и др.) при одновременном присутствии на их входах прямого и инверсного значений одной и той же переменной может возникать статический риск при ее изменении (см. § 2.1). Это часто нежелательно. Поэтому для подачи прямого и инверсного значения сигнала используют незадействованные (неперекрывающиеся) входы матрицы И.

Таким образом, получаем:

$$\begin{aligned}\bar{E} &= N_0 \bar{Q}_0 \vee \bar{N}_0 Q_0 \vee N_1 \bar{Q}_1 \vee \bar{N}_1 Q_1 \vee \dots \vee N_7 \bar{Q}_7 \vee \bar{N}_7 Q_7; \\ D_7 &= Q_0 Q_1 \dots Q_6 \bar{Q}_7 \bar{E} \vee \bar{Q}_0 \bar{Q}_7 \bar{E} \vee \bar{Q}_1 Q_7 \bar{E} \vee \dots \vee \bar{Q}_6 Q_7 \bar{E} \vee Q_7 \bar{V}; \\ D_6 &= Q_0 Q_1 \dots Q_5 \bar{Q}_6 \bar{E} \vee \bar{Q}_0 Q_6 \bar{E} \vee \bar{Q}_1 Q_6 \bar{E} \vee \dots \vee \bar{Q}_5 Q_6 \bar{E} \vee Q_6 \bar{V}; \\ &\vdots \\ D_1 &= Q_0 \bar{Q}_1 \bar{E} \vee \bar{Q}_0 Q_1 \bar{E} \vee Q_1 \bar{V}; \\ D_0 &= \bar{Q}_0 \bar{E} \vee Q_0 \bar{V}; \\ AR &= R; \\ SP &= R \bar{R}; \\ C &= CLK; \\ OE_7 &= OE; \\ &\vdots \\ OE_0 &= OE; \\ OE_{\bar{E}} &= 1.\end{aligned}$$

Четвертый этап

На четвертом этапе перераспределяют входные и выходные сигналы PLD соответственно количеству термов и наличию обратных связей у разных МЭ. Для нашего примера можно принять распределение сигналов, показанное на рис. 9.4, в. Контакт B_5 с наибольшим количеством термов оставлен неиспользуемым, чем уменьшается объем работы при программировании PLD и сохраняется свободным больший ресурс микросхемы. Это может быть полезно при модификации схемы.

Последний этап проектирования

На последнем этапе результаты проектирования представляют в виде таблицы программирования (прошивки) PLD. Для спроектированного счетчика в соответствии с приведенными выше выражениями получим таблицу, форма которой, в сущности, аналогична описанной в § 7.2, а ее заголовок и начало имеют следующий вид:

Таблица 9.2

Термы	Входы матрицы И														Выходы БИС					
	Внешние							От обратных связей												
	CLK	\overline{OE}	V	R	N_0	...	N_7	Q_0	...	\bar{E}	-	Q_7	Q_8	...	C_0	...	\bar{E}	-	C_7	...
	X_0	X_1	X_2	X_3	X_4	...	X_{11}	OC_0	...	OC_4	OC_5	OC_6	OC_7	...	B_0	...	B_4	B_5	B_7	...
								$R(H)$		P			$R(H)$		$\frac{R(H)}{H}$		$\frac{C}{H}$	-	$\frac{R(H)}{H}$	
AR					H															
SP					H, L															
OE_0		L																		
$\overline{Q_0} \bar{V} \bar{E}$			H					L		H					A					
$Q_0 \bar{V}$			L					H							A					
$N_0 \overline{Q_0}$					H			L								A				
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Строки таблицы соответствуют термам, столбцы — переменным, из которых составляются термы. В каждом столбце указан тип цепи счетчика (CLK, OE и

т. д.), тип цепи БИС (X_0 , X_1 и т. д.) для соответствующих входов счетчика. Для цепей, с которых снимаются сигналы обратных связей, указывается их тип: регистровая или комбинационная, Н-активная или Л-активная, причем при подаче обратной связи с контакта БИС она отмечается буквой Р (от Pin). Для сигналов, используемых как выход и как обратная связь, характеристика дана дробью, числитель которой относится к типу обратной связи, а знаменатель — к типу выхода. Суммарное число термов у микросхемы равно 132 — это $(8 + 10 + 12 + 14 + 16) \times 2 = 120$ термов, поступающих в элементы ИЛИ всех МЭ, плюс 10 термов управления третьим состоянием буферов плюс терм сигнала AR и терм сигнала SP. Следовательно, в табл. 9.2 будет 132 строки.

Приняв, как и ранее, в качестве признака вхождения в терм в прямом или инверсном виде буквы Н и Л соответственно, а признака отсутствия данной переменной в терме — отсутствие какой-либо записи, можно указать данные для программирования связей в БИС. Для примера это показано для нескольких термов.

Терм SP формируется как нулевой путем подачи в него прямого и инверсного значений одной и той же переменной R. Другим вариантом формирования нулевого терма является подключение многих парафазных сигналов к этому терму, например, всех входов матрицы И.

Символ А означает присутствие данного терма в данной выходной функции, а отсутствию терма соответствует отсутствие какой-либо записи. Против каждого выхода микросхемы приводится такое количество символов А, которое равно количеству термов у данного выхода. Для выполнения этого условия функцию дополняют "пустыми" термами.

На основании информации табл. 9.1 и табл. 9.2 система программирования обеспечит требуемые для реализации разработанного счетчика межсоединения элементов микросхемы.

§ 9.3. Методика и средства автоматизированного проектирования цифровых устройств

Все современные методики проектирования ЦУ на базе сложных программируемых БИС/СБИС основаны на применении САПР. Правильный выбор САПР — важнейшее условие эффективного проектирования и ускорения выпуска продукции, т. е. сокращения времени до продажи (по-английски, Time-to-Market).

Методы и средства проектирования тесно связаны с выбором САПР и, наоборот, выбор САПР определяет допустимые и целесообразные методы и средства проектирования, так что эти вопросы нельзя рассматривать в отрыве друг от друга.

Средства описания проекта

Применение САПР требует эффективных, наглядных, управляемых и контролируемых средств описания проекта. Описать проектируемое устройство можно разными способами, причем обычно применяют способ, пригодный для описания проекта в целом. Методы описания, применимые исключительно для отдельных фрагментов устройства, относятся к числу редких.

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. Сходство способа описания и внутренней организации и поведения разрабатываемого устройства существенно сокращает время создания проекта, упрощает его тестирование и, как правило, оказывается наиболее наглядным и понятным.

Графическое представление проекта создается в базе допустимых для выбранной САПР библиотечных элементов, например, в базе 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.

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. 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.

В отличие от текстовых, графические способы представления проекта обычно узко специализированы и требуют особых средств для переноса информации о проекте в другую среду, для чего могут быть применены специальные универсальные языки передачи информации о проекте (типа языка EDIF, Electronic Design Interchange Format).

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 язык AHDL (Altera HDL) 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 той или иной CPLD или FPGA.

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.

Разделение устройства на операционный блок и блок управления

Как уже указывалось, возможно как только временное (поведенческое), так и только пространственное (архитектурно-структурное) описание проекта. Однако часто 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.*

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.

Этапы проектных процедур

Порядок разработки ЦУ укрупненно упоминался в § 9.1 (см. рис. 9.3). Более детально маршрут проектирования с использованием САПР рассматривается ниже. Разработка обычно выполняется в следующем порядке:

1. Составление содержательной граф-схемы алгоритма или функциональной блок-схемы устройства. Первая задача — переход от технического задания (ТЗ) к формализованному описанию проектируемого устройства. ТЗ, как правило, является смесью словесного и технического описания, его формализация приводит к выявлению основных блоков устройства (или алгоритма) и определению их связей и/или взаимодействия. В сущности именно в этот момент реализуются начальные действия первого этапа. Формально же первый этап — разбиение задачи на отдельные функционально обособленные подзадачи — этап декомпозиции. 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.

2. 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.

3. 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.

Формы и средства описания автомата разнообразны. *Современная тенденция состоит в переходе от записи логических выражений, ограниченных правилами ТЗ, к графической форме.* Описание в виде граф-схемы переходов (диаграммы состояний) становится одним из самых распространенных вариантов задания автоматов (в английской терминологии State Machines). Графические редакторы для создания автоматов включаются в состав средств задания исходных проектов современных САПР (например, в САПР Foundation фирмы Xilinx разработки фирмы ALDEC).

Редакторы разных фирм — производителей СБИС ПЛ имеют особенности, но для всех них характерны исключительная простота, естественность и дружелюбность интерфейса с пользователем, а также отсутствие жесткой необходимости знания выходного языка редактора. Наиболее совершенные версии программ типа StateCAD Version 3.2 пакета Workview Office фирмы Viewlogic обладают полным набором средств для выполнения всей проектной процедуры разработки УА, позволяющих реализовать следующие операции:

- ❑ рисовать граф переходов, включая наименование состояний, направления, условия и приоритеты условий переходов, формируемые сигналы и способы их образования;
- ❑ проверять корректность составленного графа переходов (повторение имен, неоднозначность перехода, некорректность перехода и т. д.);
- ❑ компилировать проект (формировать выходной текстовый файл) в выбранном языковом базисе;
- ❑ моделировать поведение автомата в интерактивном или компиляционном режиме.

Важное достоинство программы StateCAD Version 3.2 — 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 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 микросхемы ПЛ, 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.

Как уже отмечалось, при использовании графических редакторов от пользователя не требуется обязательное владение выходным языком редактора. Однако в определенных случаях такое владение исключительно полезно. Полезность ориентации в языковых конструкциях проявляется, 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.

4. Компиляция проекта. После составления проекта и всех его частей можно приступать к самому ответственному этапу проектирования — компиляции проекта. Именно здесь проявляются все скрытые ошибки и нестыковки. Компиляция разбивается на ряд последовательных подэтапов: сборка базы данных проекта, контроль соединений, логическая минимизация проекта, формирование загрузочного (конфигурационного) файла и др.

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.

5. 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.

В современных САПР наиболее распространено тестирование путем работы с редакторами временных диаграмм. Эти редакторы делятся на компилирующие и интерпретирующие. В многооконных САПР интерпретирующего типа просто отображаются результаты моделирования для текущего момента модельного времени во всех видах отображения проекта (сигналы в электрических схемах, в топологии), легко изменить ход эксперимента и состав отображаемых сигналов. Достоинством компилирующих систем моделирования является минимизация временных затрат.

Программы для тестирования 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.

6. 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.

Например, в САПР MAX + PLUS II 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.

7. Организация натурных экспериментов. Последним этапом проектирования является этап экспериментальной проверки 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 тяжкие последствия.

Выполнение натурных экспериментов существенно увеличивает вероятность выпуска бездефектной продукции. Средства ускорения работ на этом этапе и возможности его переноса на ранние этапы разработки, т. е. до того момента, когда будет закончено изготовление конечного продукта, известны — 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 средств программируемой логики. Широкий спектр прототипных плат, содержащих микросхемы программируемой логики и дополнительную аппаратуру (прежде всего микросхемы быстродействующих ОЗУ), выпускается и поставляется 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.

Основные сведения о языке VHDL

В заключение параграфа остановимся на некоторых вопросах, относящихся к наиболее известному языку проектирования аппаратных средств VHDL, который будет 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 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 уровня представления схем до уровня системы в целом. В настоящее время он 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 языком, 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 спроектированных 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 (присвоения ($:=$), условия (IF), выбора (CASE), цикла (LOOP), вызова процедуры), так и традиционные типы данных: числовые, логические, символьные, перечислительные и агрегированные (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 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 нижних уровней иерархии должны иметь описание, определяющее их функционирование. Один и тот же объект может иметь несколько архитектурных тел (естественно, что при моделировании поведения системы или при ее синтезе специальные средства конфигурирования (Configuration Declaration) определяют единственный вариант поведения).

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 следуют описание объектов (Entity Declaration), которые будут использованы как компоненты проектируемого устройства, и раздел архитектуры (Architecture Declaration), который может быть представлен в структурном или поведенческом вариантах.

Многие другие термины и понятия здесь и ниже не затрагиваются, поскольку цель более или менее серьезного изучения языка может ставиться лишь в работах достаточно большого объема.

Проблемно-ориентированными и поэтому *наиболее важными средствами и понятиями языка VHDL являются:*

- ☐ средства описания иерархии проекта для описания структуры и/или поведения отдельных объектов проекта;
- ☐ средства задания и описания параллелизма для выполняемых действий и операторов;
- ☐ понятие сигнала для физических объектов, имеющих временное измерение для своих значений и средства для работы с ними.

Иерархическое построение описания системы в языке VHDL является развитием традиционного иерархического подхода и отличается тем, что распространяется не только на описание поведения, но и на описание структуры системы. Архитектурное тело (Architecture Body) — описывает поведение объекта или его структуру. Внутри архитектурного тела может быть и смесь структурного описания с поведенческим. Специальные синтаксические конструкции могут описывать интерфейс структурной компоненты объектов (component ... port), соединение компонентов между собой (port map, generic map), создание фрагмента структуры (for ... generate и if ... generate) или конкретизации конфигурации (for ... use). То, что описанию архитектуры предшествует описание интерфейса объекта (Entity) не является существенным отличием языка VHDL и аналогично (в определенном смысле) описанию прототипа в языке СИ.

Наиболее важным свойством языка VHDL является понятие параллелизма выполнения действий. Параллелизм начинается с введения понятий процес-

са (Process) и охраняемого блока (Block) и распространяется при определенных условиях на такие традиционно последовательные операторы, как вызов процедуры и оператор присвоения. Для управления параллелизмом естественно введение операторов, задающих момент запуска (абсолютных — wait ..., относительных — after), и операторов, задерживающих момент запуска (wait ... until, wait ... for). 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 стиль, когда преобразование потока входных данных в поток выходных данных осуществляется с использованием только последовательных операторов;
- ❑ параллельный стиль, когда описание поведения задано в виде параллельно выполняемых процессов;
- ❑ потоковый стиль, 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 (Мура или Мили).

Третьей важнейшей особенностью языка VHDL является введение физического типа данных. Понятие сигнала (Signal) отражает основные свойства реальных входных и выходных данных проекта. Среди различных свойств сигналов важнейшими представляются временные характеристики таких данных и, прежде всего, наличие у них прошлого, настоящего и будущего состояний. Специфические свойства сигналов потребовали введения понятия *назначение значения сигнала* (\leq), основное отличие которого от понятия *присвоение значения переменной* ($:=$) состоит в задержке изменения состояния сигнала до тех пор, пока не будут подготовлены результаты преобразования во всех одновременно инициированных процессах, и лишь после этого одновременно изменяются значения всех сигналов сразу.

Примеры поведенческих описаний элементов на языке VHDL

Проиллюстрируем поведенческие варианты описаний на простейших примерах. Пусть требуется описать на языке VHDL логический элемент, реализующий функцию $Z = (a \vee b)c$ под наименованием input3_orand1. Начиная с раздела Entity Declaration, описание может иметь следующий вид:

```
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;

```

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 СБИС ПЛ. 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 быстродействие, тогда как компилятор (при не установленных 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, ориентированных на описание 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 требовать определения функций взаимных преобразований. Пример образования схемы под именем `input3_orand1`, приведенный выше, 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 "зашелка", 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).

Следующим типовым фрагментом, играющим очень важную роль в проектировании, являются цифровые автоматы. Основные разновидности поведения автоматов сводятся к автоматам Мили или Мура, хотя расширенные структурные возможности схем ПЛ привели к широкому практическому распространению разновидности автоматов Мили — асинхронным автоматам Мили.

Наличие в языке возможности использования перечислительного типа данных позволяет ввести перечислительные типы данных "состояния", а при желании разработчика и типы данных "входы" и "выходы". Наиболее предпочтительной представляется процессная форма описания поведения автомата. В зависимости от типа используемого автомата, его архитектура может включать от одного до четырех 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. Отдельный процесс может вводиться для описания процедуры тактирования и начальной установки автомата.

В качестве примера автомата рассмотрим автомат Мили. Пусть автомат задан таблицей переходов (табл. 9.3) и таблицей выходов (табл. 9.4), где в клетках таблицы переходов записаны состояния, в которые переходит автомат из исходного состояния при соответствующем входе, а в клетках таблицы выходов — выходные сигналы при тех же условиях. Нетрудно видеть, что приведенный пример соответствует реверсивному счетчику, причем Y1 и Y2 соответствуют выдаче сигналов переноса.

Таблица 9.3

Вход	Исходное состояние			
	S0	S1	S2	S3
X0	S0	S1	S2	S3
X1	S1	S2	S3	S0
X2	S3	S0	S1	S2

Таблица 9.4

Вход	Исходное состояние			
	/S0	S1	S2	S3
X0	Y0	Y0	Y0	Y0
X1	Y0	Y0	Y0	Y1
X2	Y2	Y0	Y0	Y0

Фрагмент VHDL-программы, описывающий такой автомат, имеет вид, приведенный в листинге 1. Предполагается, что перечислительный тип `state` задан списком имен, переменные `y` и `x` объявлены в `Entity` блока, в котором определен данный процесс, и их тип задан списком имен-значений.

Процесс после задания исходного состояния (`s<=s0`) входит в бесконечно повторяющуюся петлю, в начале которой помещен оператор `WAIT`. Примененная конструкция оператора соответствует синхронному автомату, состояние которого изменяется по тактирующему сигналу `p_clk`, причем `p_clk` является глобальной переменной проекта.

Важно обратить внимание на то, что изменения состояний происходят в момент появления нарастающего фронта сигнала `p_clk`, т. к. запускающее событие определено как "появление единицы и наличие переходного процесса на входе `p_clk`".

Синтаксическая конструкция `p_clk'stable` 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 в качестве условия продолжения процесса выражения "`not p_clk'stable`" 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 в операторе `Wait`.

Листинг 1

```
-- описано вне процесса
TYPE state IS (s0,s1,s2);
TYPE input IS (x0,x1);
TYPE output IS (y0,y1,y2);
```

```

SIGNAL x: INPUT;
SIGNAL y_out :OUTPUT;
SIGNAL s: state;

```

PROCESS

```

BEGIN
  s<=s0;
  LOOP
WAIT UNTIL (p_clk='1' AND NOT p_clk'stable);
-- Реализация переходов

```

CASE s IS

```

  WHEN s0=> IF x=x0 THEN s<=s0;
    ELSEIF (x=x1) THEN s<=s1;
    ELSE s<=s2;
    END IF;
  WHEN s1=> IF x=x0 THEN s<=s1;
    ELSEIF x=x1 THEN s<=s2;
    ELSE s<=s0;
    END IF;
  WHEN s2=> IF x=x0 THEN s<=s2;
    ELSEIF x=x1 THEN s<=s3;
    ELSE s<=s1;
    END IF;
  WHEN s3=> IF x=x0 THEN s<=s3;
    ELSEIF x=x1 THEN s<=s0;
    ELSE s<=s2;
    END IF;
END CASE;
-- Формирование выходов
IF (s=s3 AND x=x1) THEN y<=y1;
ELSEIF (s=s0 AND x=x2) THEN y<=y2;
ELSE y<=y0;
END IF;
END LOOP;
END PROCESS;

```

После вычисления нового состояния и выходных сигналов (обратите внимание на то, что сигналы вычисляются на основе состояний, которые были "перед" фронтом тактирующего сигнала, а не вычисленных в текущем цикле) программа переходит в состояние ожидания нового запускающего события.

Наличие определенных стереотипов и у проектировщиков, и у САПР для описания типовых фрагментов ЦУ позволяет упростить написание и понимание описаний на языке VHDL достаточно сложных систем.

§ 9.4. 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. Будем для определенности 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 определяют основные блоки устройства и их взаимодействие. Блочная схема устройства приведена на рис. 9.6. Функциональное назначение блоков следует из их названий. Схема укрупненно отображает следующие процессы.

Внешнее управляющее устройство (процессор) обеспечивает запись байтов в ОЗУ, подавая на него помимо данных также 8-разрядный адресный код и строб записи Write. Преобразователь параллельного кода в последователь-

ный представляет в своей основе сдвигающий регистр, загружаемый параллельно байтом из ОЗУ, и затем по командам из устройства управления выводящий последовательные данные во внешнее устройство — 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.

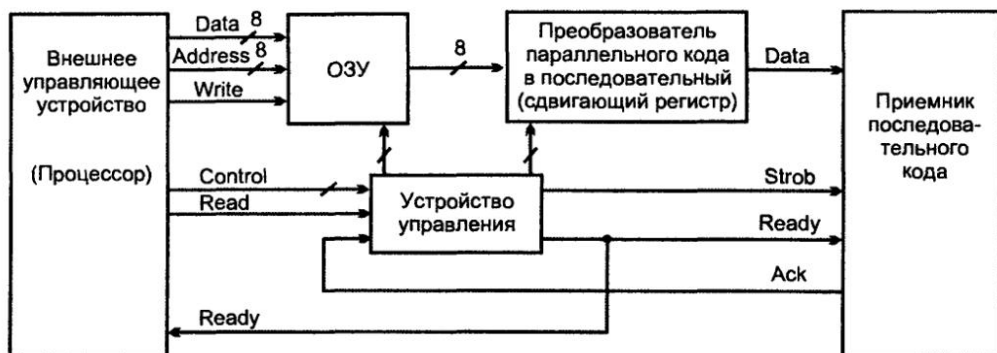


Рис. 9.6. Блок-схема устройства, принятого в качестве примера для проектирования средствами САПР

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 + PLUSII и блоков схемы (рис. 9.6), нетрудно видеть, что для реализации рассматриваемого устройства из состава библиотеки выбранной САПР можно использовать следующий набор библиотечных параметризуемых модулей (LPM):

- блок ОЗУ (LPM_RAM_DQ) с организацией 256×8;
- 8-разрядный 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.

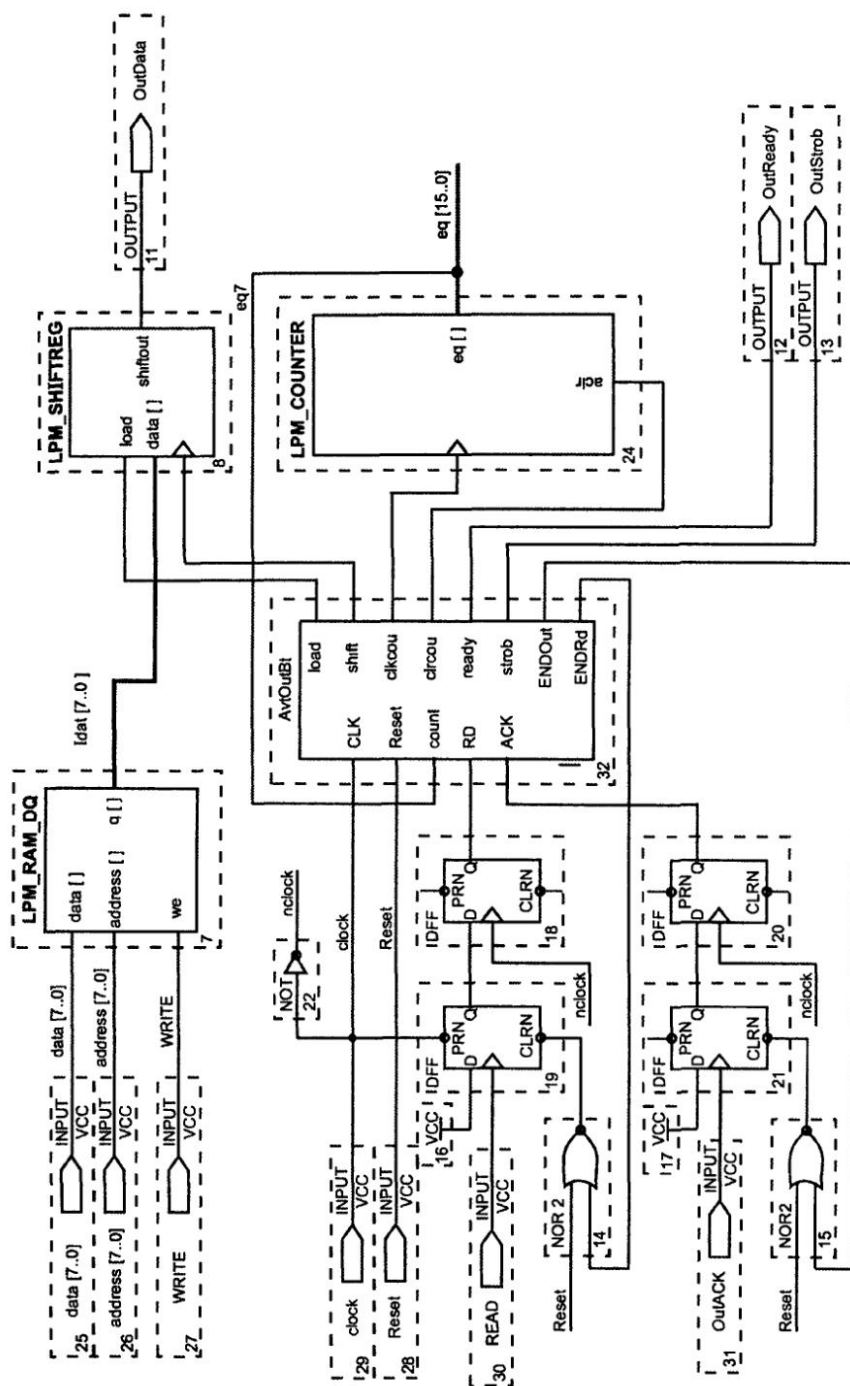


Рис. 9.7. Структурная схема устройства, проектируемого средствами САПР

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. 9.7. Кроме указанных выше 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.

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 отдельными 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. 9.8. Граф-схема переходов при помощи графического редактора программы StateCAD Version 3.2 пакета Workview Office фирмы Viewlogic 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.

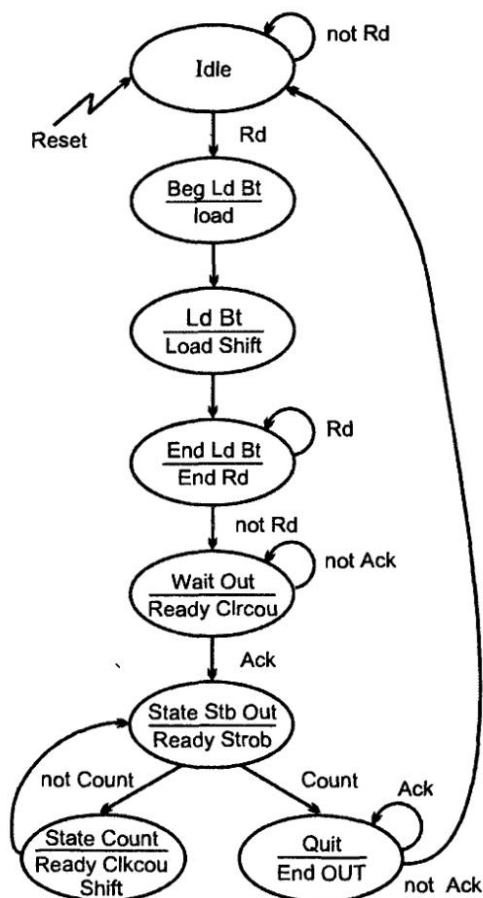


Рис. 9.8. Граф-схема автомата управления для проектируемого средствами САПР устройства

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). Необходимую для правильной работы автомата синхронизацию асинхронных сигналов READ и OutAck выполняют дополнительно введенные в схему цепочки D триггеров.

Пояснения к синтаксису AHDL и VHDL программ устройства управления

Для автомата нашего примера с помощью программы StateCAD Version 3.2 пакета Workview Office фирмы Viewlogic 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 2). Во-вторых, было создано описание этого же автомата на языке высокого уровня VHDL (листинг 3).

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 есть в этом 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*.

```
% 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 запись 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. 9.8. 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.

Второй вариант описания автомата, который мог бы управлять спроектированным устройством, компилировался с помощью программы StateCAD Version 3.2 пакета Workview Office 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.

Листинг 3

```
-- 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 опи-

сание 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 граф-схеме 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 обеспечит 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 (*.gpt), 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.

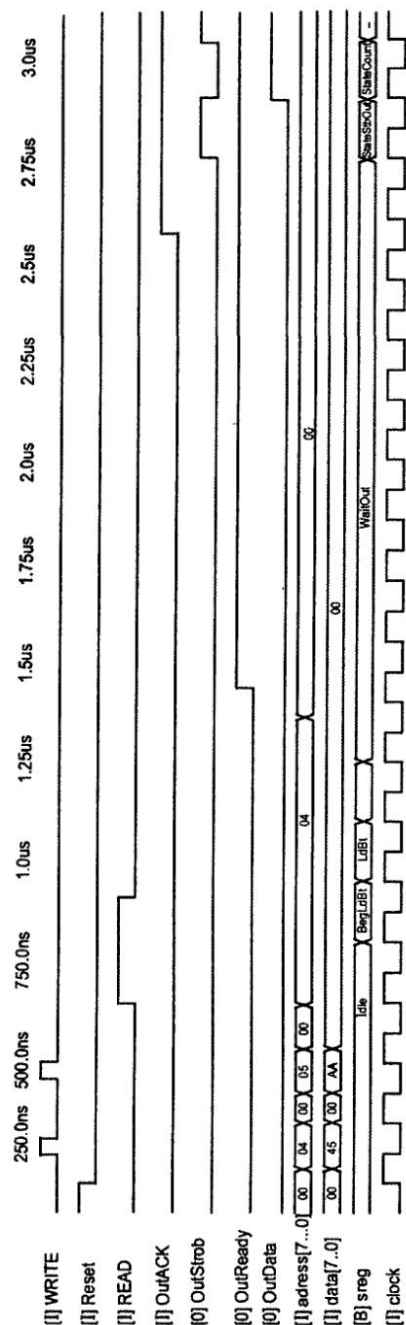
Пятый этап.

Тестирование проекта

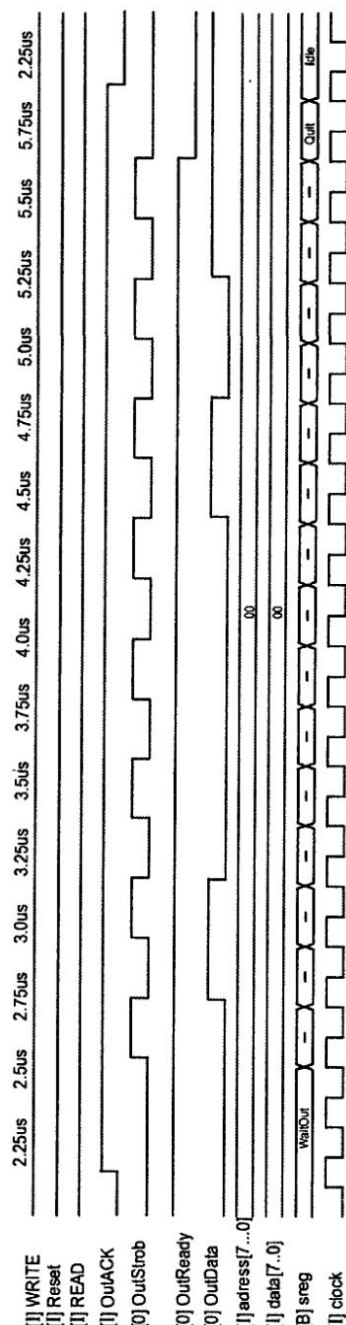
Тестирование проекта также выполнялось средствами САПР MAX + PLUS II. Созданная тестовая последовательность должна была проверять лишь ключевые моменты работы разработанного устройства. Результаты моделирования приведены на рис. 9.9. Дадим пояснения основным фрагментам моделирования.

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 шестнадцатичной 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.
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.



a



b

Рис. 9.9. Временные диаграммы, построенные средствами моделирования для устройства, спроектированного средствами САПР

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.

Литература к главе: [3], [10], [30], [34], [38], [39].

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

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.

А

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.

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 массочного 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 вентиля" 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. выше).

Вес кодовой комбинации — число единиц в разрядах данной комбинации.

Видеопамять — ЗУ с последовательным циклическим доступом к словам и периодом цикла, соответствующим процессу сканирования монитора электронными лучами.

Витая пара — одна из распространенных конструкций линий передачи сигналов, представляющая собою два скрученных провода.

Волновое сопротивление — параметр линии передачи сигналов, трактуемой как "длинная линия".

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 А и В это интервал времени между началом сигнала А и окончанием сигнала В (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 А и В это интервал времени между началом сигнала А и началом сигнала В.

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.

Демультимплексор — 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

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.

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.

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.

Конвейеризация — способ повышения частоты тактирования в тракте обработки данных, для реализации которого комбинационные цепи тракта разбиваются на ступени.

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.

Коэффициент отражения — отношение амплитуды отраженной волны к амплитуде падающей волны в концах длинной линии.

Кратность ошибки — число неверных разрядов в данной кодовой комбинации

Кратчайшая ДНФ — дизъюнктивная нормальная форма представления переключатальной функции, содержащая минимальное число конъюнктивных термов.

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.

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 — однокристалльная 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).

Модуль счета — число состояний, которое может иметь счетчик, т. е. емкость счетчика.

Мультиплексор — схема, передающая на выход одну из нескольких входных величин под управлением адресующего кода.

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 3У — параметр 3У, 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.

П

Параллельный периферийный адаптер (Parallel Peripheral Interface) — устройство, обслуживающее обмен параллельными данными между процессором и внешними устройствами.

Перекрестная помеха — помеха, порождаемая взаимным влиянием близлежащих сигнальных линий.

Периферийное сканирование (Boundary Scan Testing) — тестирование БИС/СБИС по интерфейсу JTAG.

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 изготавливается 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).

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 выводов БИС/СБИС.

Приоритетный шифратор — устройство, вырабатывающее двоичный номер старшего из имеющихся на входах запросов (прерывания, прямого доступа к памяти и др.).

Программируемость в системе (In System Programmable) — свойство БИС/СБИС программируемой логики конфигурироваться непосредственно в системе, т. е. без изъятия из схемы.

Программируемая логическая матрица (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 (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.

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 маскування 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, изготавливаемых с помощью полного набора фотошаблонов, фрагменты которых могут заимствоваться из библиотеки готовых решений.

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.

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.

Синхронизатор одиночных импульсов — схема выработки по команде одиночного импульса, принадлежащего тактовой последовательности системы

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 вентилях 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.

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 таких 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.

Страничная организация памяти — организация памяти, при которой ячейки рассматриваются как состоящий из двух частей, причем старшая часть указывает на страницу (субмодуль), а младшая является адресом слова на данной странице (в данном субмодуле).

Схема ускоренного умножения — в данном контексте схема, реализующая алгоритм умножения "сразу на два разряда".

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.

Tag — 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).

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 — высококачественная репрограммируемая память на элементах типа EEPROM, в которой стирание данных производится электрическими сигналами для всего кристалла либо для отдельных блоков (симметричных или несимметричных).

Функциональная ячейка — типовое схемное решение, входящее в состав библиотеки БМК и реализуемое на основе одной или нескольких базовых ячеек кристалла.

Функции возбуждения триггера — функции, определяющие такие воздействия на триггеры автомата, которые переводят автомат из одного состояния в другое согласно требуемому графу переходов.

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-

HE). Понятие "эквивалентный клапан" используется при оценке сложности (уровня интеграции) БМК и БИС/СБИС программируемой логики.

Энергонезависимость — свойство запоминающего устройства сохранять информацию при отключении питающих напряжений.

Я

Ячейки периферийного сканирования (Boundary Scan Cells) — дополнительные схемы в составе БИС/СБИС, обеспечивающие реализуемость их тестирования по интерфейсу JTAG.

Дополнение. Словарь иностранных сокращений и терминов

AHDL — язык описания аппаратуры фирмы Altera.

ASICs — Application Specific Integrated Circuits — специализированные ИС, изготавливаемые тем или иным способом по индивидуальному техническому заданию (для конкретного проекта).

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.

BSCs — Boundary Scan Cells — см. Ячейки периферийного сканирования.

BST — Boundary Scan Testing — см. Периферийное сканирование.

CDRAM — Cached DRAM — динамическое ОЗУ повышенного быстродействия, достигаемого путем кэширования.

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.

CPLD — Complex PLD — БИС/СБИС программируемой логики, структура которой представляет собою совокупность блоков типа PAL или GAL, объединенных матрицей программируемых соединений. Программируется пользователем.

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).

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 VHDL 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, представляющий собою развитие структуры типа FPM, состоящее в фиксации строки данных в статическом регистре с целью ускорения считывания данных, принадлежащих этой строке.

FIFO — First-In — First-Out — ЗУ с последовательным доступом к данным типа "очередь" (по правилу "первый вошел — первый вышел").

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.

HDL — Hardware Description Language — язык описания аппаратуры.

Hit — сигнал "попадание" в схемах кэш-памяти, свидетельствующий о наличии запрашиваемой единицы информации в этой памяти.

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 — First-Out — ЗУ с последовательным доступом к данным стекового типа (по правилу "последний вошел — первый вышел").

LUT — Look Up Table — см. Табличный функциональный преобразователь.

MAX + PLUS II — пакет программных средств для проектирования БИС/СБИС программируемой логики фирмы Altera.

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. Поочередное обращение к разным банкам позволяет повысить частоту обращений к памяти, т. к. для каж-

дого из банков частота обращений окажется пониженной, и банки получат дополнительное время для подготовки к очередному циклу обращений к ним.

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.

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.

RDRAM — Rambus DRAM — динамическое ОЗУ высокого быстродействия, разработанное фирмой Rambus и отличающееся высоким темпом передачи данных внутри пакета при относительно больших значениях времени доступа к первому слову пакета.

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.

SOC — System On Chip — БИС/СБИС программируемой логики высшего уровня сложности, на 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.

StrataFlash — запоминающее устройство типа Флэш с запоминанием двух битов в одном запоминающем элементе с помощью многоуровневого заряда плавающих затворов ЛИЗМОП транзисторов.

TAP — Test Access Port — см. Порт тестирования.

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.

ХАСТ — пакет программных средств для проектирования БИС/СБИС программируемой логики фирмы Xilinx.

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Пустая страница

Принятые сокращения

А	адрес
АЗУ	ассоциативное ЗУ
АП	автомат с памятью
	адресное пространство
АЦП	аналого-цифровой преобразователь
БВВ (IOB)	блок ввода/вывода
БИС	большая интегральная схема
БИСМ	БИС, реализованная на основе БМК
БМК (GA)	базовый матричный кристалл
БЯ	базовая ячейка
ВУ	внешнее устройство
ГПСП	генератор псевдослучайных последовательностей
ГПСЧ	генератор псевдослучайных чисел
ДНФ (SOP)	дизъюнктивная нормальная форма
ДОЗУ (DRAM)	динамическое ОЗУ
ЗГ	задающий генератор системы синхронизации
ЗУ	запоминающее устройство
ЗУПВ	запоминающее устройство с произвольной выборкой
ЗЭ	запоминающий элемент
ЗЯ	запоминающая ячейка
ИС	интегральная схема
КЛБ (LAB)	конфигурируемый логический блок
КМОП (CMOS)	комплементарная МОП-структура
КПДП (DMA)	контроллер прямого доступа к памяти
КЦ	командный цикл
ЛБ	логический блок
ЛВ	линия выборки (словарная)
ЛЗС	линия записи-считывания (разрядная)
ЛИЗМОП	МОП-структура с лавинной инжекцией заряда

ЛФ	логическая функция
ЛЭ (LE)	логический элемент
М	модуль счета
	информационная емкость ЗУ
МАБИС	матричная БИС
МБ	множительный блок
МБЯ	матричная базовая ячейка
МНОП	структура "металл-нитрид-оксид-полупроводник"
МОП (MOS)	структура "металл-оксид-полупроводник"
МПК	микропроцессорный комплект
МПС	микропроцессорная система
МСБ	множительно-суммирующий блок
МЦ	машинный цикл
МЭТ	многоэмиттерный транзистор
ОЗУ	оперативное ЗУ
ОК	открытый коллектор
ОС	обратная связь
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
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 вентиляционная матрица
ППЗУ (PROM)	программируемое постоянное ЗУ
ПС (BS)	периферийное сканирование

ПСА (PCI)	программируемый связной адаптер
ПТ (TAP)	порт тестирования (для интерфейса JTAG)
ПТС	программируемая точка связи
ПЯ	периферийная ячейка
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
СБИС (VLSI)	сверхбольшая интегральная схема
СБИС ПЛ	СБИС программируемой логики
СДНФ	совершенная дизъюнктивная нормальная форма
СОЗУ (SRAM)	статическое ОЗУ
СпИС (ASIC)	специализированная ИС
СПЛИС (CPLD)	сложная программируемая логическая ИС
ССИ (SSI)	семисегментный индикатор
Т	транзистор
	такт
	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 приемопере- датчик
ФБ	функциональный блок
ЦАП	цифроаналоговый преобразователь
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

1. Автоматное управление асинхронными процессами в ЭВМ и дискретных системах/В. И. Варшавский, М. А. Кишиневский, В. Б. Мараховский и др./ Под ред. В. И. Варшавского. — М.: Наука, 1986. — 398 с.
2. Алексенко А. Г., Шагурин И. И. Микросхемотехника. — М.: Радио и связь, 1990. — 496 с.
3. Антонов А. П., Мелехин В. Ф., Филиппов А. С. Обзор элементной базы фирмы Altera. — СПб.: ЭФО, 1997. — 142 с.
4. Базовые матричные кристаллы и матричные БИС/В. Г. Домрачев, П. П. Мальцев, И. В. Новаченко, С. Н. Пономарев. — М.: Энергоатомиздат, 1992. — 224 с.
5. Балашов Е. П., Григорьев В. Л., Петров Г. А. Микро- и миниЭВМ: Учебное пособие для вузов. — Л.: Энергоатомиздат. Ленинградское отделение, 1984. — 376 с.
6. Баранов С. И., Скляров В. А. Цифровые устройства на программируемых БИС с матричной структурой. — М.: Радио и связь, 1986. — 272 с.
7. Бродин В. Б., Шагурин И. И. Микроконтроллеры: Справочник. — М.: ЭКОМ, 1999. — 395 с.
8. Букреев И. Н., Мансуров Б. М., Горячев В. И. Микроэлектронные схемы цифровых устройств. 3-е изд. — М.: Радио и связь, 1990. — 415 с.
9. 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.
10. Грушин С. И., Душутин И. Д., Мелехин В. Ф. Проектирование аппаратных средств микропроцессорных систем: Учеб. пособие. — Л.: ЛПИ им. Калинина, 1990. — 78 с.
11. Гук М. Ю. Аппаратные средства IBM PC: Энциклопедия. — СПб.: Питер, 1998. — 815 с.
12. Гутников В. С. Интегральная электроника в измерительных устройствах. — Л.: Энергоатомиздат, 1988. — 304 с.
13. Каган Б. М., Сташин В. В. Основы проектирования микропроцессорных устройств автоматики. — М.: Энергоатомиздат, 1983. — 304 с.
14. Киносита К., Асада К., Карацу О. Логическое проектирование СБИС: Пер. с япон. — М.: Мир, 1988. — 309 с.

15. Колосов В. Г., Мелехин В. Ф. Проектирование узлов и систем автоматики и вычислительной техники: Учеб. пособие для вузов. — Л.: Энергоатомиздат, 1983, — 256 с.
16. Кохонен Т. Ассоциативные запоминающие устройства: Пер. с англ. — М.: Мир, 1982, — 384 с.
17. Куприянов М. С., Матюшкин Б. Д. Цифровая обработка сигналов: Процессоры. Алгоритмы. Средства проектирования. — СПб.: Политехника, 1998. — 592 с.
18. Ланцов Л. П., Зворыкин Л. Н., Осипов И. Ф. Цифровые устройства на комплементарных МДП интегральных микросхемах. — М.: Радио и связь, 1983. — 272 с.
19. Лебедев О. Н. Применение микросхем памяти в электронных устройствах: Справочное пособие. — М.: Радио и связь, 1994. — 216 с.
20. 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.
21. 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.
22. Микропроцессоры. В 3-х кн.: Учеб. для вузов/П. В. Нестеров, В. Ф. Шаньгин, В. Л. Горбунов и др./ Под ред. Л. Н. Преснухина. — М.: Высшая школа, 1986. — Кн. 1 — 495 с., Кн. 2 — 383 с., Кн. 3 — 351 с.
23. 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.
24. 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.
25. Мурсаев А. Х., Угрюмов Е. П. Структуры и схемотехника современных интегральных полупроводниковых запоминающих устройств. — СПб.: ГЭТУ, 1997. — 69 с.
26. Основы построения технических средств ЕС ЭВМ на интегральных микросхемах/В. В. Саморуков, В. М. Микитин, В. А. Павлычев и др./ Под ред. Б. Н. Файзулаева. — М.: Радио и связь, 1981. — 288 с.
27. Перспективы развития вычислительной техники: В II 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.

28. Потемкин И. С. Функциональные узлы цифровой автоматики. — М.: Энергоатомиздат, 1988. — 320 с.
29. 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.
30. Программируемые логические ИМС на КМОП-структурах и их применение/П. П. Мальцев, Н. И. Гарбузов, А. П. Шаралов, А. А. Кнышев. — М.: Энергоатомиздат, 1998. — 158 с.
31. Пупырев Е. И. Перестраиваемые автоматы и микропроцессорные системы. — М.: Наука, 1984. — 192 с.
32. Пухальский Г. И., Новосельцева Т. Я. Цифровые устройства: Учебное пособие для вузов. — СПб.: Политехника, 1996. — 885 с.
33. Савельев А. Я. Арифметические и логические основы цифровых автоматов: Учебник. — М.: Высшая школа, 1980. — 255 с.
34. Соловьев В. В., Васильев А. Г. Программируемые логические интегральные схемы и их применение. — Мн.: Беларуская навука, 1998. — 270 с.
35. Схемотехника ЭВМ/А. Т. Воронин, В. И. Зуев, В. И. Кальнин и др./ Под ред. Г. Н. Соловьева. — М.: Высшая школа, 1985. — 392 с.
36. Титце У., Шенк К. Полупроводниковая схемотехника: Справочное руководство: Пер. с нем. — М.: Мир, 1982. — 512 с.
37. Угрюмов Е. П. Проектирование элементов и узлов ЭВМ: Учеб. пособие для вузов. — М.: Высшая школа, 1987. — 318 с.
38. Угрюмов Е. П., Грушвицкий Р. И., Альшевский А. Н. БИС/СБИС с репрограмируемой структурой: Учебное пособие. — СПб.: ГЭТУ, 1997. — 96 с.
39. Угрюмов Е. П., Смирнов А. М., Альшевский А. Н. БИС с программируемой структурой: Учебное пособие. — СПб.: ГЭТУ, 1995. — 64 с.
40. Федоров Р. Ф., Яковлев В. В., Добрис Г. В. Стохастические преобразователи информации. — Л.: Машиностроение. Ленингр. отд-ние, 1978. — 304 с.
41. Хвош С. Т., Варлинский Н. Н., Попов Е. А. Микропроцессоры и микроЭВМ в системах автоматического управления: Справочник. — Л.: Машиностроение, Ленингр. отд-ние, 1987. — 640 с.
42. Хоровиц П., Хилл У. Искусство схемотехники: В 3-х томах: Т. 2: Пер с англ. — 4-е изд. — М.: Мир, 1993. — 371 с.
43. Цифровые интегральные микросхемы: Справочник/П. П. Мальцев, Н. С. Долидзе, М. И. Критенко и др. — М.: Радио и связь, 1994. — 240 с.
44. Шило В. Л. Популярные цифровые микросхемы: Справочник. 2-е изд. — Челябинск: Металлургия, Челябинское отд-ние, 1989. — 352 с.

45. Шелкунов Н. Н., Дианов А. П. Микропроцессорные средства и системы. — М.: Радио и связь, 1989. — 282 с.
46. ACT Family FPGA Data Book, Fctel, 1990.
47. Altera 1998 Data Book, January 1998.
48. Atmel Corporation 8051 Flash Microcontroller Data Book, 1997.
49. Bostock G. Programmable Logic Devices. — N-Y.: McGraw Hill, 1988 — 243 p.
50. Brey B. The 8085A Microprocessor: Software, Programming and Architecture. — Prentice-Hall, Englewood Cliffs. — N.J., 1986. — 220 p.
51. Bursky D. Advanced CPLD Architectures Challenge FPGAs, Gas//Electronic Design. — 1998. — № 22. — pp. 78—86.
52. Bursky D. Embedded Logic And Memory Find A. Home In FPGA//Electronic Design. — 1999. — № 14. — pp. 43—56.
53. Bursky D. High-Density FPGA Family Delivers Megagate Capacity//Electronic Design. — 1997. — № 25. — pp. 67—70.
54. Chang D., Mazek-Sadowska M. Dinamically Reconfigurable FPGA//IEEE Transaction on Computers. — 1999 — № 6 — pp. 565—578.
55. Kresta D., Johnson T. High-Level Design Methodology Comes Into Its Own//Electronic Design. — 1999. — № 12 — pp. 57—60.
56. Oshima Y., Sheu B., Jen S, High-Speed Memory Architectures For Multimedia Applications//IEEE Circuits & Devices. — Vol 13, — № 1. — Jan. 1997. — pp. 8—13.
57. Short K. Microprocessors And Programmed Logic. — 2-nd Ed. — Englewood Cliffs: Prentice-Hall, 1987. — 515 p.
58. 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.
59. The Programmable Logic Data Book. — Xilinx. — 1998.

Примечание

В списке литературы мало современных отечественных работ, что отражает объективное состояние дел в области издания научно-технической литературы в России. В перечне работ на английском языке преобладают фирменные каталоги и журнальные публикации, т. к. иностранные книги в настоящее время практически недоступны для российского читателя.

Предметный указатель

А

- Автомат
 - автономный, 123
 - линейный, 170
 - на триггерах с мультиплексным управлением, 129
 - с кодированием состояний в коде "1 из N", 132
- Мура, 123
 - с памятью, 39
- Адресация
 - абсолютная, 256
 - косвенная, 256
 - неабсолютная, 256
 - непосредственная, 256
 - программируемых перемычек, 411
 - прямая, 256
- Адресное пространство, 255
- Аномалии метастабильные, 117
- Арифметико-логические устройства, 90
- Ассоциативный доступ, 182

Б

- Базовая ячейка БМК, 382
- Базовые матричные кристаллы, 358
- Бесканальные БМК, 385
- Блок регистров, 262
- Блоки
 - ввода/вывода FPGA, 402
 - множительно-суммирующие, 93
 - ускоренного переноса, 91
- Блочные БМК, 385
- Бод, 316
- Буфер
 - FIFO, 181
 - LIFO, 182
- Буферные регистры, 304
- Быстрый страничный доступ, 237

В

- Ввод асинхронных данных, 120
- Вес комбинации, 70
- Видеопамять, 181
- Витая пара, 24
- Внешняя память, 175
- Время
 - выдержки, 104
 - доступа, 178
 - предустановки, 104; 177
 - сохранения, 177
 - удержания, 177

Г

- Генераторы импульсов, 29

Д

- Двоичные дешифраторы, 46
- Демультимплексоры, 55
- Динамические ЗУ, 180; 229
- Дифференциальная передача сигнала, 23
- ДОЗУ типа
 - CDRAM, 243
 - DRDRAM, 242
 - EDORAM, 238
 - MDRAM, 239
 - RDRAM, 241
 - SDRAM, 239
- Дребезг контактов, 117

З

- ЗУ
 - с последовательным доступом, 181
 - типа PROM, 197
 - типов EPROM и EEPROM, 200

И

- Индикаторы семисегментные, 30
- Интерфейс
 - JTAG, 397
 - с общей шиной, 256
 - с отдельной шиной, 256
- Информационная емкость, 176
- ИС
 - на стандартных ячейках, 447
 - полностью заказные, 446
 - полузаказные, 447
 - специализированные, 446
 - стандартные, 446
- Искажения сигналов в
 - несогласованных линиях, 19
- Искусственная энергонезависимость статических ЗУ, 226

К

- Каналы трассировки, 383
- Канальные БМК, 383
- Классификация триггеров, 102
- Код Грея, 126
- Кодовое расстояние, 70
- Командный цикл, 264
- Комбинационные цепи, 39
- Компараторы, 64
 - на больше, 66
 - на равенство, 65
- Компиляция проекта, 462
- Конвейеризация продвижения данных, 239
- Контроль
 - по модулю 2, 69
 - с использованием кодов Хемминга, 73
- Коррекция расфазирования синхросигналов, 137
- Кратность ошибки, 70
- Критерии качества ЦУ, 46
- Круговой приоритет, 330
- Кэш-память, 175
 - полностью ассоциативная, 191
 - с наборно-ассоциативной архитектурой, 192
 - с прямым размещением, 192

Л

- Линейная селекция, 256
- Логические блоки FPGA, 398

М

- Маскирование запросов, 330
- Масочные ЗУ, 195
- Матричные перемножители, 93
- Машинный цикл, 264
- Методика PREP, 437
- Микропроцессор, 249
- Микропроцессорная система (МПС), 250
- Микросхемы типа, 431
- Минимизация логических функций, 44
- МНОП-транзистор, 201
- Модемы, 316
- Модуль памяти, 257
- М-последовательность, 171
- Мультиплексная формула, 55
- Мультиплексоры, 54

Н

- Наращивание
 - (расширение) ПЛМ, 365
 - размерности дешифратора, 49
 - размерности мультиплексоров, 55
 - размерности приоритетного шифратора, 52

О

- Области применения FPGA, 410
- Оперативные ЗУ, 178
- Организация ЗУ, 176
- Основная память, 175
- Ошибка
 - пропуска, 319
 - формата, 319

П

- Память типа StrataFlash, 216
- Параллельный периферийный адаптер, 306

Перекрестные помехи, 17
Периферийное сканирование, 397
ПМЛ с разделяемыми
 конъюнкторами, 372
Полиномиальные счетчики, 170
Порт тестирования, 442
Постоянные ЗУ, 179
Преобразователь параллельного кода в
 последовательный, 148
Приоритетные шифраторы, 51
Программирование
 в системе, 442
 ЗУ, 199
 ПЛМ, 362
Программируемая
 матричная логика, 357
 пользователем память, 181
Программируемые
 логические матрицы, 357
 пользователем вентиляемые
 матрицы FPGA, 397
Программируемый
 интервальный таймер, 348
 контроллер прерываний, 329
 контроллер прямого доступа к
 памяти, 340
 связной адаптер, 315
Программный
 безусловный ввод/вывод, 293
 счетчик, 263
Прототипные платы, 464
Прямой доступ к памяти, 338
Псевдослучайные
 последовательности, 171

Р

Распределитель тактов, 163
Регенерация данных, 180
Регистр
 команд, 263
 флажков, 261
Регистровые
 ЗУ, 175
 файлы, 145
Регистры
 параллельные, 144
 последовательно-
 параллельные, 145

последовательные
 (сдвигающие), 144
универсальные, 148

Режимы

 конфигурирования СБИС ПЛ, 401
 неиспользуемых входов, 34
 работы таймера, 353

С

Свойство самозапуска, 159
Синхронизаторы одиночных
 импульсов, 119
Синхронизация
 в цифровых устройствах, 132
 двухфазная, 142
 однофазная, 138
Система команд МП, 273
Системная частота тактирования, 438
Системные эквивалентные вентили,
 438
Системы
 коммутации CPLD, 419
 межсоединений FPGA, 409
Сложные программируемые
 логические схемы (CPLD), 412
Согласование волновых
 сопротивлений
 параллельное, 21
 последовательное, 21
Специализированные аппаратные
 ядра, 431
Специальное маскирование, 330
Способ описания проекта
 графический, 459
 текстовый, 459
Старт-бит, 318
Статические
 ЗУ, 180
 ЗУ типа БиКМОП, 228
 риски, 39
Стек, 262
Стоп-бит, 318
Структура
 2D, 182
 2DM, 186
 3D, 184

- Структурное уравнение триггера, 106
- Сумматор
 - накапливающий, 89
 - параллельный с параллельным переносом, 82
 - параллельный с последовательным переносом, 81
 - последовательный, 80
 - с условным переносом, 88
- Сумматоры групповой структуры, 85
- Схемы
 - свертки, 71
 - ускоренного умножения, 96
- Счетчик Джонсона, 165
- Счетчики
 - в коде "1 из N", 163
 - в коде Грея, 162
 - двоичные, 152
 - с групповой структурой, 155
 - с произвольным модулем, 157
 - синхронные с параллельным переносом, 154
 - синхронные с последовательным переносом, 156

Т

- Такт, 268
- Тестирование проекта, 463
- Токовые импульсы в цепях питания, 15
- Транзистор типа ЛИЗМОП, 202
- Трассировочная способность БМК, 381
- Трехшинная структура МПС, 250
- Триггер типа
 - D, 103
 - JK, 103
 - RS, 103
 - T, 103
- Триггер-зашелка, 104
- Триггеры, 101
 - двухступенчатые, 104
 - синхронные, 103
 - управляемые уровнем, 103
 - управляемые фронтом, 103
- Турбо-бит, 396

У

- Указатели старшей единицы, 53
- Указатель стека, 262
- Универсальные логические модули на основе мультиплексоров, 56
- Усилители-регенераторы, 232

Ф

- Фильтрация напряжений питания, 17
- Флэш-память, 205
 - с несимметричной блочной структурой, 211
 - с одновременным стиранием всех данных, 209
- файловая, 213
- Формирование импульсов по длительности, 28
- Функциональная ячейка БМК, 382
- Функциональные блоки CPLD, 413
- разновидности ПЛМ и ПМЛ, 368
- Функция
 - генерации, 83
 - прозрачности, 83

Х, Ш

- Характеристическое уравнение триггера, 106
- Шинные формователи, 302

Э

- Эквивалентный вентиль, 382; 431
- Элементы
 - задержки, 25
 - с открытым коллектором, 11
 - с тремя состояниями выхода, 10
- Эмиттерный дот, 14
- Энергонезависимость, 178

Я

- Язык VHDL, 464
- Языки описания аппаратуры, 459
- Ячейки периферийного сканирования, 440