МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ
БИШКЕКСКИЙ ТЕХНИЧЕСКИЙ КОЛЛЕДЖ

«Согласовано»
Директор _____«_____»
_____
_____
«___»_____ 2023 г.

Рассмотрено на
Заседании методсовета
Протокол №___от «____»____2023 г
Председатель _____.
_____

УЧЕБНЫЕ МАТЕРИАЛЫ

“ЭВМ и ПУ”

Специальность 000000 “Техническое обслуживание средств вычислительной техники и компьютерных сетей”

Трудоемкость – 5 **кредитов (150 ч)**
Из них:
Аудиторная работа –76 ч.
Самостоятельная работа - 74 ч.

Составитель: Эрмат к.Зарина преподаватель _____

Бишкек-2023

Оглавление

1. Глоссарий

Государственный образовательный стандарт – совокупность норм и правил, определяющая образовательный минимум содержания образовательных программ, базовые требования к качеству подготовки выпускников, предельно допустимую учебную нагрузку обучающихся;

Квалификация – формальный результат процесса оценки и признания, компетентным органом достижения индивидом результатов обучения по установленным стандартам;

Компетенция– заранее заданное социальное требование (норма) к образовательной подготовке ученика (обучаемого), необходимой для его эффективной продуктивной деятельности в определенной сфере;

Кредит – условная мера трудоемкости основной профессиональной образовательной программы;

Модуль– часть учебной дисциплины, имеющая определенную логическую завершенность по отношению к установленным целям и результатам обучения, воспитания;

Основная профессиональная образовательная программа– совокупность учебно-методической документации, регламентирующей цели, ожидаемые результаты, содержание и организацию реализации образовательного процесса по соответствующей специальности;

Профессиональный стандарт - основополагающий документ, определяющий в рамках конкретного вида профессиональной деятельности требования к ее содержанию и качеству и описывающий качественный уровень квалификации сотрудника, которому тот обязан соответствовать, чтобы по праву занимать свое место в штате любой организации, вне зависимости от рода ее деятельности;

Результаты обучения – компетенции, приобретенные в результате обучения по основной образовательной программе/модулю;

Учебный модуль - это логическое и приемлемое разделение труда в рамках профессии, работы или сферы труда, с четким началом и концом (закрытая и автономная единица как в отношении применения, так и оценки). Он представляет собой результат обучения и включает учебный материал, методы обучения и другие материалы, в том числе средства обучения и соответствующую техническую информацию;

Цикл дисциплин–часть образовательной программы или совокупность учебных дисциплин, имеющая определенную логическую завершенность по отношению к установленным целям и результатам обучения, воспитания.

2. Список сокращений

УМ – Учебный модуль;

ПЦК – Предметно-цикловая комиссия;

ТО- Техническое обслуживание ;

СРС - Самостоятельная работа студента;

ЭВМ- Электронно-вычислительная машина;

ПУ- Периферийные устройства

3. Рабочая учебная программа (силлабус)

1.Информация об учебном модуле

Наименование учебного модуля – «ЭВМ и ПУ»

Трудоемкость учебного модуля – 5 кредитов (150 часов)

Расписание учебного модуля -5 семестр – 4 часа в неделю аудиторных

2.Информация о преподавателе

Зарина Эрмат кызы преподаватель _____-

3.Литература

Основная литература:

1. В.В. Косулин. – Казань: Казан. гос. энерг. ун-т, 2016. – 494 с.

Дополнительная литература:

Н. Н. Горнец, А. Г. Рощин. — М. : Издательский центр «Академия», 2012. — 240 с.

4**.Пререквизиты учебного модуля:**

Изучение данной дисциплины базируется на формировании системы теоретических знаний, практических умений в области Электронно-вычислительной системы и периферийных устройств.

5.Постреквизиты учебного модуля:

«Метод доступа к среде передачи, коммуникационные устройства, среда передачи, протокол, спецификация».

6.Описание учебного модуля

Цели обучения учебного модуля: обучение студентов принципам построения, назначения, функционирования и практического использования компьютера и эффективного управления процессами обработки данных в современных ЭВМ.

Результаты обучения учебного модуля: Знать назначение и функции компонентов сетей ЭВМ;

Сфера применения: взаимосвязь компьютерной техники как модель взаимодействия открытых систем, метод доступа к среде передачи;

Методы преподавания учебного модуля: лекция, практические занятия.

Методы изучения учебного модуля: выполнение практических заданий; самостоятельная работа студентов.

7.Тематический план по учебному модулю:

Темы	Всего	Аудиторная работа	Самостоятельная работа студента СРС
	часы	часы	часы
Раздел 1. Введение. Становление и эволюция вычислительной техники (ЭВМ)	4	2	2
Раздел 2. Представление числовой информации в ЭВМ	4	2	2
Раздел 3. Конструктивные узлы вычислительных машин. Счетчики	4	2	2
Раздел 4. Фон-неймановская вычислительная машина	4	4	2
Раздел 5. Организации шин вычислительной машины	6	6	2
Раздел 6. Принципы построения арифметико-логических устройств	6	4	2
Раздел 7. Системы памяти. Организация основной памяти	6	4	2
Раздел 8. Устройства основной памяти	8	4	2
Раздел 9. Принципы организации работы процессоров	8	4	4
Раздел 10. Периферийные устройства	6	4	2
Итого	**150 (5 кредитов)**	**76**	**74**

8. Политика курса:

Соблюдать общее этические нормы поведения в общественных местах; соблюдать правила поведения студентов в колледже; не пропускать занятий без уважительной причины; все пропуски занятий должны быть отработаны, включая пропуски по уважительным причинам (способ отработки определяет преподаватель); не опаздывать на урок более, чем на 5 минут; запрещается пользоваться телефоном во время урока (исключение составляет: использование телефона в качестве калькулятора; обучение в онлайн-режиме).

9. Оценивание:

- оценка «***5 (отлично)***» выставляется за глубокое и полное овладение содержанием учебного материала, в котором студент легко ориентируется; понятийным аппаратом, умение связать теорию с практикой, решать практические задачи, высказывать и обосновывать свои суждения. Отличная оценка предполагает грамотное, логическое изложение ответа (как в устной, так и в письменной форме), качественное внешнее оформление;

- оценка «***4(хорошо)***» выставляется студенту за полное освоение учебного материала, владение понятийным аппаратом, осознанное применение знаний для решения практических задач, грамотное изложение ответа, но в содержании и форме ответа имеются отдельные неточности;

- оценка «***3(удовлетворительно)***» выставляется студенту, который обнаружил знание и понимание основных положений учебного материала, но излагает его неполно, непоследовательно, допускает неточности в определении понятий, в применении знаний для решения практических задач, не умеет доказательно обосновывать свои суждения;

- оценка «***2(неудовлетворительно)***» выставляется студенту, имеющему разрозненные, бессистемные знания, не умеющему выделить главное и второстепенное, допускающему ошибки в определениях, понятиях, искажающему их смысл, беспорядочно и неуверенно излагающему материал (Приложение 2).

1. Курс лекций (учебные материалы) по учебному модулю

Расширенный тематический план по учебному модулю (календарный)

№ тем.	Наименование разделов и тем	Количество часов на тему				№ занятий	СРС задания по занятиям (урокам)
		Всего	аудиторные	практич.	СРС		
	Раздел 1. Введение. Становление и эволюция вычислительной техники (ЭВМ)						
1.1	Становление и эволюция вычислительной техники	6	2		4	2(лк) 4(пр)	**Урок 1 - Домашнее задание**: Проработка конспекта пройденной темы.
1.2	Структуры вычислительных машин		1			1(лк)	**Урок 2 - Домашнее задание:** Проработка конспекта пройденной темы.
	Раздел 2. Представление числовой информации в ЭВМ						
2.1	Компоненты вычислительной сети и способы их взаимодействия**.**	5	1		4	1(лк) 4(пр)	**Урок 3 - Домашнее задание:** Ответить на котрольные вопросы
2.2	Топологии подключения. Логическая и физическая структуризация сети.	2	2			2(лк)	**Урок 4 - Домашнее задание:** Ответить на котрольные вопросы
	Раздел 3. Конструктивные узлы вычислительных машин. Счетчики						
3.1	Стандартизация компьютерных сетей.	10	2		2	2 (лк) 2(пр)	**Урок 5 - Домашнее задание:** Проработка конспекта пройденной темы
3.2	Эталонная модель взаимосвязи открытых систем.		2				**Урок 6 - Домашнее задание:** Проработка конспекта пройденной
	Раздел 4. Фон-неймановская вычислительная машина						
4.1	Стеки коммуникационных	5	1		4	1(лк)	**Урок 7 - Домашнее задание:**

	протоколов.					4(пр)	Проработка конспекта пройденной темы.
4.2	Линии связи.	1	1			1(лк)	**Урок 8 - Домашнее задание:** Проработка конспекта пройденной темы.
4.3	Методы и каналы передачи данных		1			1(лк)	**Урок 9 - Домашнее задание:** Проработка конспекта пройденной темы.
	Раздел 5. Организации шин вычислительной машины						
5.1	Стандарты и технологии локальных сетей	6	2		4	6 (лк)	**Урок 10 - Домашнее задание**: Ответить на котрольные вопросы
5.2	Локальная сеть Ethernet.		1			1(лк)	**Урок 11 - Домашнее задание:** Проработка конспекта пройденной темы.
5.3	Метод доступа к передающей среде и формат пакета сети Ethernet.		2			2(лк)	
5.4	Высокоскоростные варианты сети Ethernet		2			2(лк)	**Урок 12 - Домашнее задание**: Ответить на котрольные вопросы
5.5	Сеть Token Ring		2			2(лк)	**Урок 13 - Домашнее задание:** Проработка конспекта пройденной темы.
5.6	Распределенный волоконно-оптический интерфейс передачи данных FDDI		1			1(лк)	
	Раздел 6. Принципы построения арифметико-логических устройств						
6.1	Использование мостов в локальных сетях.	8	2		4	2 (лк) 4(пр)	**Урок 14 - Домашнее задание**: Проработка конспекта пройденной темы. **Урок 15 - Домашнее задание**: Ответить на вопросы заданные на рабочем листе.
6.2	Сетевые устройства	1	1			1(лк)	**Урок 16 - Домашнее задание**:

							Проработка конспекта пройденной темы.
	Раздел 7. Системы памяти. Организация основной памяти						
7.1	Использование мостов в локальных сетях.	3	1		2	1(лк) 2(пр)	**Урок 17 - Домашнее задание**: Проработка конспекта пройденной темы.
7.2	Управление сетью	1	1			1(лк)	**Урок 18 - Домашнее задание:** Ответить письменно: Ответить на вопросы заданные на рабочем листе
	Раздел 8. Устройства основной памяти						
8.1	Методика расчета конфигурации сети Ethernet	8		4	4	25(лк)	**Урок 25 - Домашнее задание:** Ответить письменно: Ответить на вопросы заданные на рабочем листе.
	Раздел 9. Принципы организации работы процессоров						
9.1	Эволюция компьютерных и телекоммуникационных технологий.	2		2			
	Раздел 10. Периферийные устройства						
10.1	От первых локальных сетей до современных сетевых технологий.	2		2			
	Итого:	150	Л (30)	Пр /Лаб (76)	СРС (24)		

Раздел 1. Введение. Становление и эволюция вычислительной техники (ЭВМ)

Тема 1. Становление и эволюция вычислительной техники
Занятие 1 (лекция)

Ключевые вопросы: - признаки классификации сетей, особенности каждого типа.
Предыдущие знания обучающихся: Основы построения и архитектуры ЭВМ;
принципы построения, параметры и характеристики цифровых и аналоговых элементов ЭВМ
Результаты обучения: С помощью выбора методами элементной базы для построения различных архитектур вычислительных средств показать на практике.
Информация по теме:

В основе современных информационных технологий лежат аппаратные средства компьютерной техники. Поэтому вполне оправдано повышенное внимание, уделяемое изучению вычислительных машин и вычислительных сетей в рамках высшего образования. В рамках федерального государственного образовательного стандарта высшего образования по направлению подготовки 09.03.01 Информатика и вычислительная техника дисциплина «ЭВМ и периферийные устройства» содержит следующие аспекты: основные характеристики ЭВМ различных классов; организация процессора; организация памяти ЭВМ; выполнение команд; организация прерываний; периферийные устройства и т.д.

В результате изучения дисциплины «ЭВМ и периферийные устройства» формируются следующие компетенции или их составляющие: – разработка технических заданий на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием;

– способность участвовать в настройке и наладке программно- аппаратных комплексов;

– способность сопрягать аппаратные и программные средства в составе информационных и автоматизированных систем;

– способность подключать и настраивать модули ЭВМ и периферийного оборудования.

Целью данного издания является изложение теоретических основ, необходимых для освоения практических навыков в работе с ЭВМ и переферийными устройствами (ПУ), а также непосредственное получение этих практических навыков и освоение вышеназванных компетенций.

В результате освоения данной дисциплины студент должен:

знать:

– основы построения и архитектуры ЭВМ;

– принципы построения, параметры и характеристики цифровых и аналоговых элементов ЭВМ;

– современные технические средства взаимодействия с ЭВМ.

уметь:

– выбирать, комплексировать и эксплуатировать программно- аппаратные средства в создаваемых вычислительных и информационных системах и сетевых структурах;

– ставить и решать схемотехнические задачи, связанные с выбором системы элементов при заданных требованиях к параметрам (временным, мощностным, габаритным, надежностным);

– тестировать, испытывать и использовать аппаратные средства вычислительных и информационных систем;

владеть:

– методами выбора элементной базы для построения различных архитектур вычислительных средств.

Материал, изложенный в данном учебном пособии, ставит своей целью привить обучающимся вышеуказанные навыки.

Основные термины

Вычислительная машина – это комплекс технических и программных средств, предназначенный для автоматизации подготовки и решения задач пользователей.

Вычислительную систему определим как совокупность взаимосвязанных и взаимодействующих процессоров или вычислительных машин, периферийного оборудования и программного обеспечения, предназначенную для подготовки и решения задач пользователей.

Формально отличие вычислительной системы (ВС) от вычисли- тельной машины (ВМ) выражается в количестве вычислителей. Множественность вычислителей позволяет реализовать в ВС параллель- ную обработку. С другой стороны, современные вычислительные машины с одним процессором также обладают определенными средствами распараллеливания вычислительного процесса. Иными словами, грань между ВМ и ВС часто бывает весьма расплывчатой, что дает основание там, где это целесообразно, рассматривать ВМ как одну из реализаций ВС. И напротив, вычисли-тельные системы часто строятся из традиционных ВМ и процессоров, поэтому многие из положений, относящихся к ВМ, могут быть распространены и на ВС.

Архитектура вычислительной машины – это логическое построение вычислительной машины, т.е. то, какой машина представляется программисту. Подобную трактовку называют «узкой», и охватывает она перечень и формат команд, формы представления данных, механизмы ввода/вывода, способы адресации памяти и т.п. Такой подход не включает в рассмотрение вопросы физического построения вычислительных средств: состав устройств, число регистров процессора, ёмкость памяти, наличие специального блока для обработки вещественных чисел, тактовая частота центрального процессора и т.д. Этот круг вопросов принято определять понятием организация или *структурная организация*.

Уровни детализации структуры вычислительной машины

Вычислительная машина как законченный объект может рассматриваться на различных уровнях детализации. Таких уровней может быть достаточно много, однако сложившаяся практика ограничивает их число четырьмя (рис. 1.1).

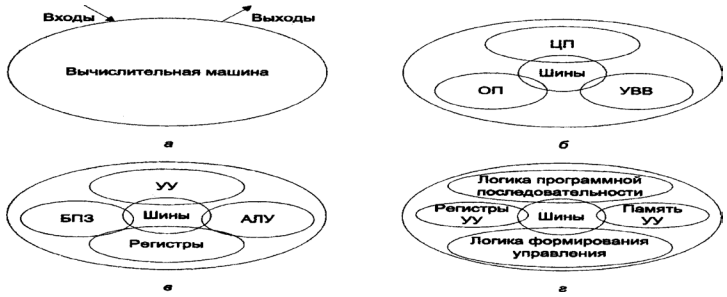


Рис. 1.1. Уровни детализации вычислительной машины:

а) уровень «черного ящика»; *б*) уровень общей архитектуры; *в*) уровень архитектуры центрального процессора; г) уровень архитектуры устройства управления

На *первом уровне* вычислительная машина рассматривается как устройство, способное хранить и обрабатывать информацию, а также обмениваться данными с внешним миром (рис. 1.1, *а*). ВМ представляется

«черным ящиком», который может быть подключен к коммуникационной сети и к которому, в свою очередь, могут подсоединяться устройства.

Уровень общей архитектуры (рис. 1.1, *б*) представляет ВМ в виде четырех составляющих: центрального процессора (ЦП), основной памяти (ОП), устройства ввода/вывода (УВВ) и системы шин.

На *третьем уровне* детализируется каждое из второго уровня. Для взят центральный процессор (рис. 1.1, *в*). В простейшем в нем можно выделить:

– арифметико-логическое устройство (АЛУ), обеспечивающее обработку целых чисел;

– блок обработки чисел в формате с плавающей запятой (БПЗ);

– регистры процессора, использующиеся для краткосрочного хранения команд, данных и адресов;

– устройство управления (УУ), обеспечивающее совместное функционирование устройств ВМ;

– внутренние шины.

На *четвертом уровне* детализируются элементы третьего уровня. На рис. 1.1, *г* раскрыта структура устройства управления. УУ представлено в виде четырех составляющих:

– логики программной последовательности – электронных схем, обеспечивающих выполнение команд программы в последовательности, предписываемой программой;

– регистров и дешифраторов устройства управления;

– управляющей памяти;

– логики формирования управления, генерирующей все необходимые управляющие сигналы.

Эволюция средств вычислительной техники

При описании эволюции средств вычислительной техники (ВТ) обычно используют один из двух подходов: хронологический или технологический. В первом случае это хронология событий, существенно повлиявших на становление ВТ, во втором – технологический подход, когда развитие вычислительной техники рассматривается в терминах архитектурных решений и технологий.

В качестве узловых моментов, определяющих появление нового поколения ВТ, обычно выбираются революционные идеи или технологические прорывы, кардинально изменяющие дальнейшее развитие средств автоматизации вычислений. Одной из таких идей принято считать концепцию вычислительной машины с хранимой в памяти программой, сформулированную Джоном фон Нейманом. Взяв ее за точку отсчета, историю развития ВТ можно представить в виде трех этапов:

1. Донеймановского периода.
2. Эры вычислительных машин и систем с фон-неймановской архитектурой.
3. Постнеймановской эпохи – эпохи параллельных и распределенных вычислений, где наряду с традиционным подходом все большую роль начинают играть отличные от фон-неймановских принципы организации вычислительного процесса.

Значительно большее распространение, однако, получила привязка поколений к смене технологий. Принято говорить о «механической» эре (нулевое поколение) и последовавших за ней пяти поколениях ВС. Первые четыре поколения традиционно связывают с элементной базой вычислительных систем: электронные: лампы, полупроводниковые приборы, интегральные схемы малой степени интеграции (ИМС), большие (БИС), сверхбольшие (СБИС) и ультрабольшие (УБИС) интегральные микросхемы. Пятое поколение в общепринятой интерпретации ассоциируют с интеллектуальными возможностями ВС. Работы по созданию ВС пятого поколения велись в рамках четырех достаточно независимых программ, осуществлявшихся учеными США, Японии, стран Западной Европы и стран Совета экономической взаимопомощи.

Ввиду того, что ни одна из программ не привела к ожидаемым результатам, разговоры о ВС пятого поколения понемногу утихают. Трактовка пятого поколения явно выпадает из «технологического» принципа. С другой стороны, причисление всех ВС на базе сверхбольших интегральных схем (СБИС) к четвертому поколению не отражает принципиальных изменений в архитектуре ВС, произошедших за последние годы.

Концепция машины с хранимой в памяти программой

В основе архитектуры современных ВМ лежит представление алгоритма решения задачи в виде программы последовательных вычислений. Согласно стандарту ISO 2382/1-84, программа для ВМ – это

«упорядоченная последовательность команд, подлежащая обработке».

Вычислительная машина, где определенным образом закодиро- ванные команды программы хранятся в памяти, называется *вычислительной машины с хранимой в памяти программой*. Идея создания такой ВМ принадлежит создателям вычислителя ENIAC Эккерту, Мочли и фон Нейману. Относительно авторства существует несколько версий, но поскольку в законченном виде идея впервые была изложена в 1945 г. в статье фон Неймана, именно его фамилия фигурирует в обозначении архитектуры подобных машин, составляющих подавляющую часть современного парка ВМ и ВС.

Сущность фон-неймановской концепции вычислительной машины можно свести к четырем принципам:

- двоичного кодирования;
- программного управления;
- однородности памяти;
- адресности.
-

Принцип двоичного кодирования

Согласно этому принципу, вся информация (данные, команды), кодируются двоичными цифрами 0 и 1. Каждый тип информации представляется двоичной последовательностью и имеет свой *формат*. Последовательность битов в формате, имеющая определенный смысл, называется *полем*. В числовой информации обычно выделяют поле знака и поле значащих разрядов. В формате команды можно выделить два поля (рис. 1.2): *поле кода операции* (КОп) и *поле адресов* (адресную часть – АЧ).

Код операции (КОп)	Адресная часть (АЧ)

Рис. 1.2. Структура команды

Код операции представляет собой указание, какая операция должна быть выполнена, и задается с помощью *r*-разрядной двоичной комбинации.

Вид *адресной части* и число составляющих ее адресов зависят от типа команды:

– в командах преобразования данных АЧ содержит адреса объектов обработки (операндов) и результата;

– в командах изменения порядка вычислений – адрес следующей команды программы;

– в командах ввода/вывода – номер устройства ввода/ вывода. Адресная часть также представляется двоичной последовательностью, длину которой обозначим через *p*. Таким образом, команда в вычислительной машине имеет вид $(r + p)$-разрядной двоичной комбинации.

Принцип программного управления

Все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последова- тельности управляющих слов – команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, то есть в порядке их положения в программе. При необходимости, с помощью специальных команд эта последовательность может быть изменена. Решение об изменении порядка выполнения команд программы принимается либо на основании анализа результатов предшествующих вычислений, либо безусловно.

Принцип однородности памяти

Команды и данные хранятся в одной и той же памяти и внешне в памяти неразличимы. Распознать их можно только по способу использования. Это позволяет производить над командами те же операции, что и над числами, и, соответственно, открывает ряд возможностей. Так, циклически изменяя адресную часть команды, можно обеспечить обращение к последовательным элементам массива данных. Такой прием носит название модификации команд и с позиций современного программирования не приветствуется. Более полезным является другое следствие принципа однородности, когда команды одной программы могут быть получены как результат исполнения другой программы. Эта возможность лежит в основе *трансляции* – перевода текста программы с языка высокого уровня на язык конкретной ВМ.

Концепция вычислительной машины, изложенная в статье фон Неймана, предполагает единую память для хранения команд и данных. Такой подход был принят в вычислительных машинах, создававшихся в Принстонском университете, из-за чего и получил название *принстонской архитектуры*. Практически одновременно в Гарвардском университете предложили иную модель, в которой ВМ имела отдельную память команд и отдельную память данных. Этот вид архитектуры называют *гарвардской архитектурой*. Долгие годы преобладающей была и остается принстонская архитектура, хотя она порождает проблемы пропускной способности тракта «процессор-память». В последнее время в связи с широким использованием кэш-памяти разработчики ВМ все чаще обращаются к гарвардской архитектуре.

Принцип адресности

Структурно основная память состоит из пронумерованных ячеек, причем процессору в произвольный момент доступна любая ячейка. Двоичные коды команд и данных разделяются на единицы информации, называемые словами, и хранятся в ячейках памяти, а для доступа к ним используются номера соответствующих ячеек – адреса.

Принципы действия компьютера. Фон-неймановская архитектура

Основное отличие персонального компьютера от больших машин, или так называемых мейнфреймов, состоит в том, что он позволяет одновременно использовать его ресурсы только одному пользователю. Он может выполнять одновременно несколько программ: обработки, вывода результатов, загрузки, поиска информации в сети и т.д. Кроме того, многие персональные машины используются в качестве серверов в сети, и их ресурсами (т.е. аппаратными и программными средствами) могут пользоваться несколько пользователей одновременно.

Структура самого компьютера за все время существования машин изменилась незначительно. Она по-прежнему строится на основе модели фон Неймана – ее основная память состоит из отдельных ячеек с последовательными номерами (или «адресами»), в которых могут храниться как коды отдельных команд (программа), так и данных. Однако технологический прогресс привел к объединению нескольких узлов и устройств в одной микросхеме.

Типичная фон-неймановская ВМ (рис. 1.3) содержит: память, устройство управления, арифметико-логическое устройство и устройство ввода/вывода.

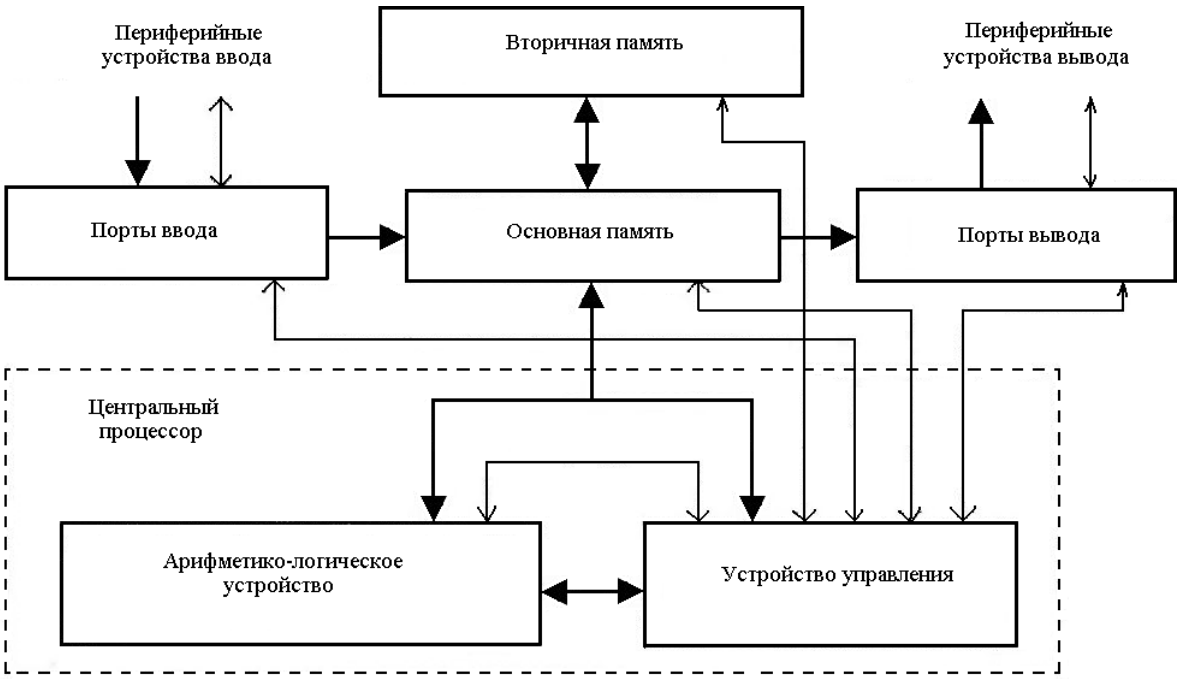


Рис. 1.3. Структура фон-неймановской вычислительной машины

В любой ВМ имеются средства для ввода программ и данных к ним. Информация поступает из подсоединенных к ЭВМ периферийных устройств (ПУ) ввода. Результаты вычислений выводятся на периферийные устройства вывода. Связь и взаимодействие ВМ и ПУ обеспечивают порты ввода и порты вывода. Термином *порт* обозначают аппаратуру сопряжения периферийного устройства с ВМ и управления им. Совокупность портов ввода и вывода называют *устройством ввода/вывода* (УВВ) или *модулем ввода/вывода* ВМ (МВБ).

Введенная информация сначала запоминается в основной памяти, а затем переносится во вторичную память, для длительного хранения. Чтобы программа могла выполняться, команды и данные должны располагаться в основной памяти (ОП), организованной таким образом, что каждое двоичное слово хранится в отдельной ячейке, идентифицируемой адресом, причем соседние ячейки памяти имеют следующие по порядку адреса. Доступ к

любым ячейкам запоминающего устройства (ЗУ) основной памяти может производиться в произвольной последовательности. Такой вид памяти известен как *память с произвольным доступом*. Основная память современных вычислительных машин в основном состоит из полупроводниковых оперативных запоминающих устройств (ОЗУ), обеспечивающих как считывание, так и запись информации. Для таких запоминающих характерна энергозависимость – хранимая информация теряется при отключении электропитания. Если необходимо, чтобы часть основной памяти была энергонезависимой, в состав основной памяти включают постоянные запоминающие устройства (ПЗУ), также обеспечивающие произвольный доступ. Хранящаяся в ПЗУ информация может только считываться (но не записываться).

Размер ячейки основной памяти обычно принимается равным 8 двоичным разрядам – байту. Для хранения больших чисел используются 2, 4 или 8 байтов, размещаемых в ячейках с последовательными адресами. В этом случае за адрес числа часто принимается адрес его младшего байта. Так, при хранении 32-разрядного числа в ячейках с адресами 200, 201, 202 и 203 адресом числа будет 200. Такой прием называют *адресацией по младшему байту* (little endian addressing). Возможен и противо- положный подход – по меньшему из адресов располагается старший байт. Этот способ известен как *адресация по старшему байту* (big endian addressing). Адресация младшему байту характерна для микропроцессоров фирмы Intel и мини-ЭВМ фирмы DEC, а по старшему байту – для микропроцессоров фирмы Motorola и универсальных ЭВМ фирмы IBM. В принципе выбор порядка записи байтов существенен лишь при пересылке данных между вычислительными машинами с различными формами их адресации или при манипуляциях с отдельными байтами числа. В большинстве вычислительных машин предусмотрены специальные инструкции для перехода от одного способа к другому.

Для долговременного хранения больших программ и массивов данных в ВМ обычно имеется дополнительная память, известная как *вторичная*. Вторичная память энергонезависима и чаще всего реализуется на базе магнитных дисков. Информация в ней хранится в виде специальных программно поддерживаемых объектов – файлов.

Устройство управления (УУ) – часть вычислительной машины, организующая автоматическое выполнение программ (путем реализации функций управления) и обеспечивающая функционирование ВМ как единой системы. Устройство управления вычислительной машины следует рассматривать как совокупность элементов, между которыми происходит пересылка информации, в ходе которой эта информация может подвергаться определенным видам обработки. Пересылка информации между любыми элементами ВМ инициируется своим сигналом управления (СУ), т.е. управление вычислительным процессом сводится к выдаче нужного набора сигналов управления в нужной временной последовательности. *Цепи СУ показаны на рис. 1.3 полутоновыми линиями*. Основной функцией УУ является формирование управляющих сигналов, отвечающих за извлечение команд из памяти в порядке, определяемом программой, и последующее исполнение этих команд. Кроме того, УУ формирует СУ для синхронизации и координации внутренних и внешних устройств ВМ.

Еще одной неотъемлемой частью ВМ является *арифметико- логическое устройство* (АЛУ). АЛУ обеспечивает арифметическую и логическую обработку двух входных переменных, в результате которой формируется выходная переменная. Функции АЛУ обычно сводятся к простым арифметическим и логическим операциям, а также операциям сдвига. Помимо результата операции АЛУ формирует ряд *признаков результата* (флагов), характеризующих полученный результат и события, произошедшие в процессе его получения (равенство нулю, знак, четность, перенос, переполнение и т.д.). Флаги могут анализироваться в

УУ с целью принятия решения о дальнейшей последовательности выполнения команд программы.

УУ и АЛУ тесно взаимосвязаны и их обычно рассматривают как единое устройство, известное как *центральный процессор* (ЦП) или просто процессор. Помимо УУ и АЛУ, в процессор входит набор *регистров общего назначения* (РОН), служащих для промежуточного хранения информации в процессе ее обработки.

1.2 Структуры вычислительных машин

В настоящее время примерно одинаковое распространение получили два способа построения вычислительных машин: *с непосредственными связями* и *на основе шины*.

Типичным представителем первого способа может служить классическая фон-неймановская вычислительная машина (рис. 1.3). В ней между взаимодействующими устройствами (процессор, память, устройство ввода/вывода) имеются непосредственные связи. Особенности связей (число линий в шинах, пропускная способность и т.п.) определяются видом информации, характером и интенсивностью обмена Достоинством архитектуры с непосредственными связями можно считать возможность развязки «узких мест» путем улучшения структуры

и характеристик только определенных связей, что экономически может быть наиболее выгодным решением. У фон-неймановских ВМ таким «узким местом» является канал пересылки данных между ЦП и памятью. Кроме того, ВМ с непосредственными связями плохо поддаются реконфигурации.

В варианте *с общей шиной* все устройства вычислительной машины подключены к магистральной шине, служащей единственным трактом для потоков команд, данных и управления (рис. 1.4). Наличие общей шины существенно упрощает реализацию ВМ, позволяет легко менять состав и конфигурацию машины. Благодаря этим свойствам *шинная архитектура* получила широкое распространение в мини-и микроЭВМ. Вместе с тем, именно с шиной связан и основной недостаток архитектуры: в каждый момент передавать информацию по шине может только одно устройство. Основную нагрузку на шину создают обмены между процессором и памятью, связанные с извлечением из памяти команд и данных и записью в память результатов вычислений. На операции ввода/вывода остается лишь часть пропускной способности шины. Практика показывает, что даже при достаточно быстрой шине для 90 % приложений этих остаточных ресурсов обычно не хватает, особенно в случае ввода или вывода больших массивов данных.

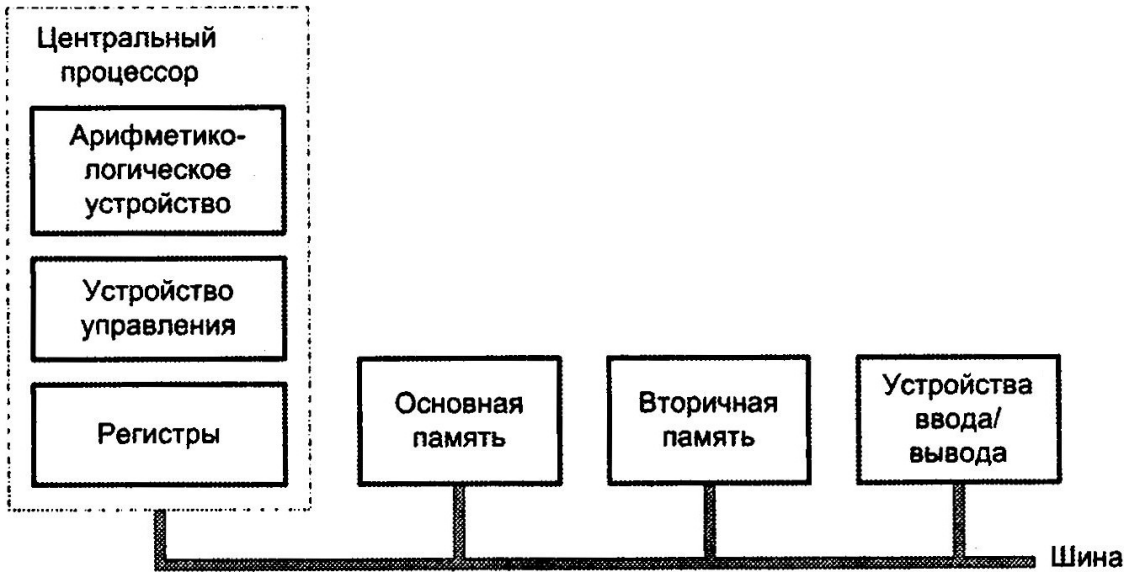


Рис. 1.4. Структура вычислительной машины на базе общей шины

При сохранении фон-неймановской концепции последовательного выполнения команд программы шинная архитектура в чистом ее виде оказывается недостаточно эффективной. Более распространена архитектура с иерархией шин, где помимо магистральной шины имеется

еще несколько дополнительных шин. Они могут обеспечивать непосредственную связь между устройствами с наиболее интенсивным обменом, например процессором и кэшпамятью. Другой вариант использования дополнительных шин – объединение однотипных устройств ввода/вывода с последующим выходом с дополнительной шины на магистральную. Все эти меры позволяют снизить нагрузку на общую шину и более эффективно расходовать ее пропускную способность.

В современных ВМ общая шина состоит из нескольких «подшин» – шины адреса, шины данных и шины управления. В персональных машинах для экономии места на системной плате шины адреса и данных иногда выполняют в виде одной разделяемой во времени шины; тогда адрес и данные по ней передаются только поочередно.

Базовым элементом компьютера является триггер. На его основе выполняются другие узлы компьютера.

Триггер – электронная схема, которая может находиться в одном из двух устойчивых состояний «0» и «1». Внешними сигналами можно переводить триггер из одного состояния в другое.

Регистр – это несколько определенным образом соединенных триггеров, т.е. можно записать двоичное слово в регистр, прочитать его, сдвинуть, инвертировать.

Счетчик позволяет определить число поступивших на него сигналов.

Он также строится на основе триггеров.

Логическая схема реализует определенную логическую функцию, т.е. формирует выходной сигнал при определенных комбинациях сигналов на ее входах.

Содержимое счетчика команд (СчК) процессора передается по адресной шине на регистр адреса (РгА) основной памяти (рис. 1.5). В момент включения компьютера в счетчике команд всегда находится один и тот же начальный адрес. Таким образом, запрашивается содержимое ячейки памяти с этим начальным адресом, принадлежащим BIOS. Как правило, эта ячейка содержит код команды безусловного перехода, служащей для изменения содержимого счетчика команд. Этот код передается на регистр команд (РгК) процессора по шине данных. Содержимое ячейки памяти поступает на регистр команд РгК, поскольку запрос к памяти произведен из счетчика команд; это *обязательное требование* для любого компьютера традиционной архитектуры.

Регистр команд процессора РгК, в свою очередь, состоит из нескольких регистров: регистра кода операции (РгКОП) и регистров адресов процессора (РгАП). Часть слова, попавшая в регистр кода операции, передается в блок управления (БУ), вырабатывающий последовательность управляющих сигналов.

Когда выполняется команда безусловного перехода, вторая адресная часть слова, попавшая в один из регистров адреса процессора, под управлением сигналов с БУ передается вновь на счетчик команд. Эта команда одноадресная, т.е. ее адресная часть содержит только один адрес. На этом и завершается ее выполнение. Блок управления формирует сигнал об окончании выполнения команды, а содержимое СчК вновь передается на РгА памяти, т.е. происходит запрос следующей команды.

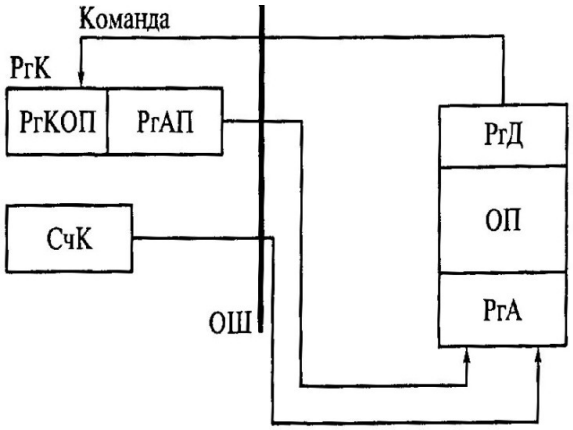


Рис. 1.5. Передача команд из оперативной памяти в центральный процессор

Таким образом, процедура обращения к памяти повторяется. Содержимое ячейки памяти, к которой произведено повторное обращение, рассматривается в качестве новой команды, т.е. вновь загружается на РгК процессора. Обычно вторая команда служит для начала загрузки ОЗУ с магнитного диска; она уже не является командой безусловного перехода. При ее выполнении под давлением кода операции (часть команды, попавшая на РгКОП) вырабатываются иные управляющие сигналы, а содержимое первого регистра РгАП, представляющего собой часть РгК, передается на адресный регистр памяти и рассматривается в качестве адреса первого операнда.

Для ОЗУ безразлично, откуда пришел запрос – из счетчика команд или адресного регистра, поэтому в регистре данных (РгД) памяти слово формируется так же, как и раньше. Однако в процессоре оно помещается на первый регистр данных АЛУ, поскольку запрос этого слова поступил из адресного регистра РгАП процессора. Затем блок управления формирует аналогичные сигналы для передачи на регистр адреса РгА основной памяти содержимого второго регистра адреса РгАП процессора; в результате содержимое ячейки памяти с адресом, находящимся в регистре адресов процессора РгАП, поступает на второй регистр данных арифметического устройства.

Затем блок управления вырабатывает сигналы в зависимости от кода операции в регистре кода операции РгКОП, подает их в АЛУ, которое выполняет соответствующую операцию, а ее результат помещает в выходной регистр-аккумулятор. После этого содержимое регистра- аккумулятора передается в ячейку памяти, адрес которой обычно находится в первом регистре адресов процессора РгАП, т.е. выполняется еще одно обращение к ОП. Информация из регистра-аккумулятора передается на шину данных, а адрес ячейки из регистра адресов процессора РгАП – на адресную шину. В зависимости от конструкции машины, числа адресов в выполняемой команде (адресности) и других особенностей, содержимое регистра-аккумулятора может сохраняться в нем, передаваться в ячейку ОП по адресу, находящемуся в первом или втором РгАП.

После сохранения содержимого регистра-аккумулятора к счетчику команд (СчК) добавляется длина текущей команды в байтах (часто говорят

«единица»), чтобы обратиться к следующей ячейке памяти, и начинается новый цикл выполнения очередной команды.

Таким образом, выполнение программы происходит последо- вательно: каждый раз в машине реализуется лишь одна команда, попадающая в регистр команд из ОП. Чтобы увеличить производи- тельность компьютера, нужно либо повысить скорость выполнения команды, либо выполнять несколько последовательных команд одновременно. Повышение скорости выполнения команды связано с улучшением технических характеристик и увеличением быстродействия всех компонентов, входящих в компьютер – ЦП, ОП, шин интерфейсов, устройств ввода-вывода. Но увеличение скорости выполнения команды принципиально ограничено – скорость распространения сигналов в машине не может превышать скорость света, а длина пути определяется числом вентилей и применяемой технологией. Второй путь, заключающийся в параллельном выполнении нескольких команд, наиболее перспективен.

Структуры вычислительных систем

Понятие «вычислительная система» предполагает наличие множества процессоров или законченных вычислительных машин, при объединении которых используется один из двух подходов.

В вычислительных системах с общей памятью (рис. 1.6) имеется общая основная память, совместно используемая всеми процессорами

системы. Связь процессоров с памятью обеспечивается с помощью коммуникационной сети, чаще всего вырождающейся в общую шину. Таким образом, структура вычислительной системы с общей памятью аналогична архитектуре с общей шиной, в силу чего ей свойственны те же недостатки. Применительно к вычислительным системам данная схема имеет дополнительное достоинство: обмен информацией между процессорами не связан с дополнительными операциями и обеспечивается за счет доступа к общим областям памяти.

Альтернативный вариант организации – распределенная система, где общая память вообще отсутствует, а каждый процессор обладает собственной локальной памятью (рис. 1.7). Часто такие системы объединяют отдельные ВМ. Обмен информацией между составляющими системы обеспечивается с помощью коммуникационной сети посредством обмена сообщениями.

Рис. 1.6. Структура вычислительной системы с общей памятью

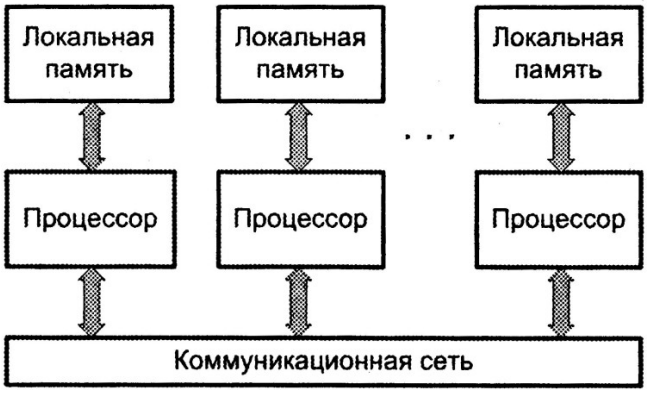


Рис. 1.7. Структура распределенной вычислительной системы

Подобное построение вычислительных систем снимает ограничения, свойственные для общей шины, но приводит к дополнительным издержкам на пересылку сообщений между процессорами или машинами.

Перспективы совершенствования архитектуры вычислительных машин и вычислительных систем

Совершенствование архитектуры вычислительных машин и систем началось с момента появления первых ВМ и не прекращается по сей день. Каждое изменение в архитектуре направлено на абсолютное повышение производительности или на более эффективное решение задач опреде- ленного класса. Эволюцию архитектур определяют самые различные факторы, главные из которых показаны на рис. 1.8.

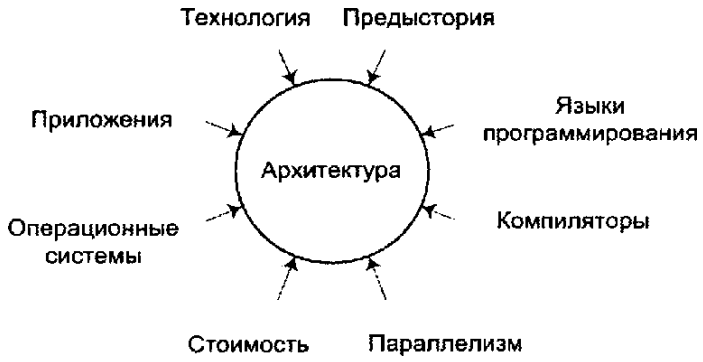


Рис. 1.8. Факторы, определяющие развитие архитектуры вычислительных систем

Надежность, производительность, быстродействие и его показатели

Отказы компьютера могут оказать критическое влияние на работу информационно-справочной системы или системы управления. Например, задержка в выдаче управляющего воздействия в результате отказа компьютера приводит к потере управления, разрушению технологических установок, срыву выполнения задания, т.е. к значительным потерям, а в некоторых случаях и к экологическим катастрофам. Современные технологии не позволяют полностью избавиться от отказов в аппаратуре и ошибок в программах даже при значительных затратах, но ряд мер технического и организационного характера способны снизить их интенсивность.

Надежность – это свойство компьютера сохранять свою работоспособность, т.е. выполнять возложенные на него функции. Однако в работе компьютера возможны ошибки, которые подразделяют на систематические, возникающие в результате отказов, и случайные, возникающие в результате сбоев.

Отказ – это утрата возможности выполнения требуемой функции.

Сбой – кратковременное нарушение правильной работы аппаратуры.

Производительность

Способность машины выполнять некоторый объем работы по обработке данных за единицу времени называют *производительностью.* На нее оказывает влияние множество факторов: характер выполняемых задач, архитектура и параметры процессора, характеристики основной и внешней памяти, наличие дополнительных устройств обработки, быстродействие соединительных шин, способ подключения различных устройств и т.п. Оценка производительности проводится по стандартным методикам, служащим только для непосредственного сравнения различных компьютеров между собой.

Для оценки производительности часто используют понятие *времени прохождения задачи* (*времени ответа*, *времени выполнения*) – интервал от момента поступления задачи на выполнение до момента представления результатов ее решения пользователю. Это время включает в себя работу ЦП, обращения к оперативной памяти и дискам, операции ввода-вывода, накладные расходы, связанные с работой ОП, и т.п. В мультипрограммном режиме, когда процессор компьютера выполняет несколько программ, время прохождения задачи не будет постоянным.

Быстродействие и его показатели

Способность компьютера выполнять определенное число операций за единицу времени называют *быстродействием.* В большинстве случаев его можно определить по тактовой частоте генератора, которая стала важнейшей характеристикой быстродействия. При этом оценивается быстродействие только процессора. Время выполнения программы в ЦП зависит от трех параметров: длительности такта синхронизации (или тактовой частоты), числа тактов синхронизации, необходимого для выполнения каждой команды, и общего числа команд в программе. Для оценки быстродействия персональных компьютеров особенно часто используют частоту.

Кроме того, быстродействие можно оценить по следующим показателям:

– длительность выполнения операций определенного типа (обычно число наиболее коротких арифметических операций, выполняемых за секунду; это так называемое *пиковое быстродействие* процессора);

– средняя длительность выполнения операции из некоторого стандартного набора операций; это *номинальное быстродействие*;

– средняя длительность выполнения представительной задачи; при этом если в затратах времени учитывается только время обработки, то такую задачу принято называть ядром, а если учитывается и время на ввод-вывод, то эталонной задачей. Время на организацию вычисли- тельного процесса не учитывается. Это *системная производительность*.

Быстродействие компьютера, имеющего традиционную архитектуру и призванного решать задачи с большим числом логических операций, принято оценивать числом MIPS (миллион инструкций в секунду). Применение MIPS в качестве универсального показателя наталкивается на ряд трудностей:

– каждый компьютер обладает структурой, ориентированной на обработку слов определенного формата и разрядности, т.е. инструкции в разных компьютерах определяют различный объем работы;

– этот показатель не учитывает сложность выполняемой команды. Поэтому при наличии в компьютере дополнительного сопроцессора, призванного выполнять операции с плавающей точкой, этот показатель снижается. При отсутствии математического сопроцессора операции над числами с плавающей точкой реализуются посредством подпрограмм, состоящих из нескольких достаточно простых команд целочисленной арифметики, и показатель MIPS имеет высокое значение. Сложная команда сопроцессора выполняется достаточно долго (хотя

и значительно быстрее, чем соответствующая подпрограмма), поэтому показатель MIPS снижается.

Контрольные вопросы

1. По каким признакам можно разграничить понятия «вычислительная машина» и «вычислительная система»?
2. В чем состоит различие между «узкой» и «широкой» трактовкой понятия «архитектура вычислительной машины»?
3. Какой уровень детализации вычислительной машины позволяет определить, можно ли данную ВМ причислить к фон-неймановским?
4. Какие закономерности в эволюции вычислительных машин породили появление нового научного направления – «Теория эволюции компьютеров»?
5. По каким признакам выделяют поколения вычислительных машин?
6. Поясните определяющие идеи для каждого из этапов эволюции вычислительной техники.
7. Какой из принципов фон-неймановской концепции вычисли- тельной машины можно рассматривать в качестве наиболее существенного?
8. Оцените достоинства и недостатки архитектур вычислительных машин с непосредственными связями и общей шиной.
9. Сформулируйте основные тенденции развития интегральной схемотехники.
10. Охарактеризуйте основные направления в дальнейшем развитии архитектуры вычислительных машин и систем.

Раздел 2. ПРЕДСТАВЛЕНИЕ ЧИСЛОВОЙ ИНФОРМАЦИИ В ЭВМ

Тема 2. СИСТЕМЫ СЧИСЛЕНИЯ

Занятие (лекция)

В компьютере может храниться и обрабатываться информация различного характера: числа, адреса, команды, различные символы, графические изображения и т.д. Любая информация в компьютере представляется в числовой форме, при этом используются различные системы счисления.

Под *системой счисления* понимается способ представления чисел с помощью символов (цифр), имеющих определенное количественное значение. В любой системе счисления числа представляются в виде последовательности цифр.

В *непозиционных системах счисления* количественный эквивалент каждой цифры не зависит от ее положения в записи числа. Примером непозиционной системы, которая сохранилась до наших дней, может служить римская система счисления. В основе римской системы счисления лежат знаки I для числа 1, V для числа 5, X для 10, L для 50, а для обозначения чисел 100, 500 и 1000 стали применять первые буквы соответствующих латинских слов (Centum – сто, Demimille – половина тысячи, Mille – тысяча).

В *позиционных системах счисления* количественное значение цифры зависит от ее места в числе. В позиционных системах счисления каждая цифра имеет вес. Обычно вес старшей цифры по отношению к весу соседней младшей цифры больше в количество раз,

равное основанию системы счисления. При этом для целых чисел вес младшего разряда в любой системе счисления равен единице.

Основные достоинства любой позиционной системы счисления – простота выполнения арифметических операций и ограниченное количество цифр, необходимых для записи любых чисел. Примером позиционной системы счисления служит десятичная система счисления, которой мы широко пользуемся.

В позиционной системе счисления любое вещественное число в развернутой форме может быть представлено в следующем виде:

где A – само число; q – основание системы счисления; a_i – цифры, принадлежащие алфавиту данной системы счисления; n – число целых разрядов числа; m – число дробных разрядов числа.

Так, десятичное число $A_{10} = 4718{,}63$ в развернутой форме запишется так:

$$A_{10} = 4 \cdot 10^3 + 7 \cdot 10^2 + 1 \cdot 10^1 + 8 \cdot 10^0 + 6 \cdot 10^{-1} + 3 \cdot 10^{-2}.$$

В современных компьютерах используются позиционные системы счисления с основаниями 2, 8, 10 и 16. В табл. 2.1 приведены возможные способы изображения первых 16 чисел во всех четырех системах счисления.

Таблица 2.1

Системы счисления

Двоичные числа D_2	Восьмеричные числа D_8	Десятичные числа D_{10}	Шестнадцатеричные числа D_{16}
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F

При хранении и обработке информации внутри компьютера используется двоичная система счисления. Это объясняется необходимостью физического представления только двух цифр (0 и 1), простотой выполнения арифметических операций и возможностью осуществления любых преобразований информации с помощью двоичных логических элементов.

Шестнадцатеричная (и реже восьмеричная) система счисления используется для более компактного представления информации (по сравнению с двоичной системой) при вводе и выводе больших массивов двоичных данных. Это связано с простотой перехода от двоичной системы счисления к шестнадцатеричной (восьмеричной) и наоборот.

2.1. Формы представления чисел

В общем случае числа имеют знак (плюс или минус). Кроме того, число может включать в себя целую и дробную части. Специальные формы представления чисел позволяют кодировать знаки чисел и указывать положение точки (запятой), разделяющей целую и дробную части числа.

Для кодирования знака числа отводится специальный разряд, называемый *знаковым.* Под него обычно отводится старший разряд числа. Для положительных чисел в нем записывается цифра 0, для отрицательных – 1.

Для указания положения точки используют одну из двух форм: форму с фиксированной или форму с плавающей точкой.

Форма представления чисел с фиксированной точкой

Эта форма, называемая также естественной, предполагает, что все числа в компьютере могут быть только целыми или только дробными. В этом случае положение точки является стандартным для данного компьютера и не требует специального указания. Эта форма является простой, но приводит к некоторому усложнению программирования.

Если в компьютере для всех чисел положение точки зафиксировано справа от младшего цифрового разряда, то числа принимают только целые значения.

На рис. 2.1 представлено (n + 1)-разрядное целое число. Один разряд занимает знак, остальные n разрядов используются как цифровые. Веса цифровых разрядов показаны в верхней части рисунка. В этом случае в компьютере могут быть представлены числа, модуль которых находится в диапазоне

$2^0 \le A \le 2^n - 1$.

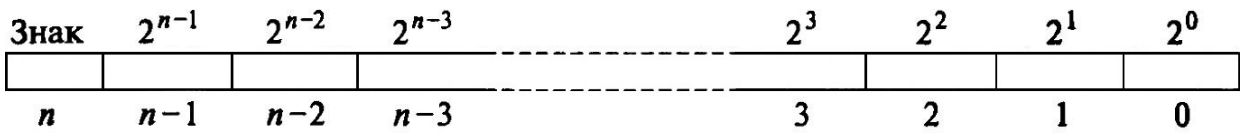


Рис. 2.1. Целое число в формате с фиксированной точкой

При этом точность представления чисел равна единице, так как числа могут быть только целыми.

Если точка зафиксирована слева от старшего цифрового разряда, то все числа могут быть только дробными. Формат дробного числа с фиксированной точкой (ФТ) представлен на рис. 2.2.

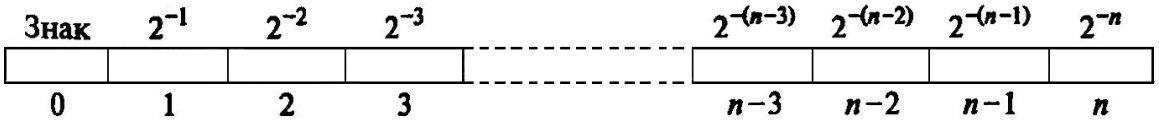


Рис. 2.2. Дробное число в формате с фиксированной точкой

Дробные числа с фиксированной точкой, имеющие n цифровых разрядов, представляются с точностью T^n (величина единицы младшего разряда дроби) в диапазоне

$$2^{-n} \leq |A| \leq 1 - 2^{-n}.$$

В формате с фиксированной точкой могут представляться числа без знака. В этом случае все разряды являются цифровыми.

В современных микропроцессорах используется представление данных в форме целых чисел с фиксированной точкой. Форма дробных чисел с фиксированной точкой применяется для представления мантиссы числа в форме с плавающей точкой. Некоторые из целочисленных форматов микропроцессоров фирмы Intel представлены на рис. 2.3.

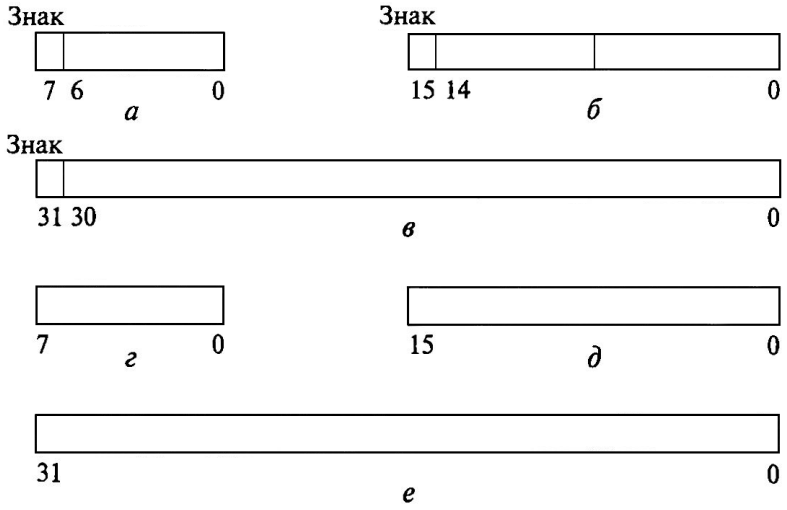


Рис. 2.3. Целочисленные форматы микропроцессоров фирмы Intel:

а − байт (целое со знаком); *б* − слово (целое со знаком); *в* − двойное слово (целое со знаком); *г* − байт (целое без знака); *д* − слово (целое без знака); *е* − двойное слово (целое без знака)

Стандартными форматами являются *байт*, *слово* и *двойное слово*.

При обработке мультимедийной информации используются не только отдельные целые числа, но и группы целых чисел, которые обрабатываются одновременно. При этом несколько малоразрядных чисел упаковываются в 64-разрядное слово. Упакованными могут быть восемь байтов, четыре слова или два двойных слова.

2.1.1. Форма представления чисел с плавающей точкой

Эта форма (нормальная или полулогарифмическая) позволяет представлять в компьютере любые (целые, дробные или смешанные) числа. Число в форме с плавающей точкой записывается в виде двух частей: мантиссы и порядка. Мантисса включает в себя значащие разряды числа, а порядок указывает положение точки. При этом мантисса записывается как дробное число с фиксированной точкой, а порядок – как целое число с фиксированной точкой.

Знак мантиссы является знаком всего числа, а знак порядка определяет, содержит ли число целую часть.

Значение числа с плавающей точкой определяется следующим образом:

$A = mq^p$,

где m – мантисса числа; q – основание системы счисления; p – порядок числа.

Так, в десятичной системе счисления число $A_{10} = -123{,}456$ в форме с плавающей точкой может быть записано следующим образом:

$m_A = 0{,}123456$, $p_A = 3$,

$m_A = 0{,}0123456$, $p_A = 4$,

$m_A = 0{,}00123456$, $p_A = 5$ и.т.д.

При заданных значениях мантиссы и порядка для определения значения числа нужно точку (запятую) в мантиссе перенести на количество разрядов, равное величине порядка, вправо для положительных порядков и влево для отрицательных. Например:

$m_A = 0{,}87412456$, $p_A = 5$ $A = 87412{,}456$;

$m_B = 0{,}12437696$, $p_A = 2$ $B = 0{,}0012437696$.

Число с плавающей точкой может быть нормализованным и ненормализованным. Число не нормализовано, если старшая цифра мантиссы равна нулю, т.е. $m_q = 0$. Нормализованное число с плавающей точкой позволяет сохранить большее количество значащих цифр, поэтому в памяти числа хранятся в нормализованном виде. Нормализация выполняется путем сдвига мантиссы влево до тех пор, пока старший разряд мантиссы не станет равным единице. Так как значение мантиссы при этом увеличивается, для сохранения величины числа при сдвиге на каждый разряд значение порядка уменьшается на единицу.

Нормализованное число с плавающей точкой представляется с точностью 2^{-n}, где n – разрядность мантиссы. Диапазон чисел с плавающей точкой составляет:

$$\frac{1}{2} \cdot 2^{-2^k - 1} \le |A| \le 1 \cdot 2^{-n} \cdot 2^{2^k - 1},$$

где k – разрядность порядка.

Для увеличения диапазона чисел с плавающей точкой (за счет некоторого уменьшения точности) двоичная мантисса может рассматриваться как шестнадцатеричное число. В этом случае каждая двоичная тетрада представляет одну шестнадцатеричную цифру, поэтому нормализация будет нарушена лишь тогда, когда четыре старших разряда мантиссы будут равны нулю. Например, если старшие разряды мантиссы имеют вид 0, 000101 ..., то мантисса считается нормализованной, так как тетрада 0001 представляет шестнадцатеричную цифру 1. Появление незначащих нулей в мантиссе приводит к потере точности. Вместе с тем существенно увеличивается диапазон представления чисел:

$$\frac{1}{16} \cdot 16^{-2^k} \le |A| \le 1 \cdot \left(2^4 - 1\right) \cdot 16^{2^k - 1}$$

Порядок числа может быть положительным или отрицательным. Для упрощения операций над порядками часто используют смещенный порядок путем увеличения действительного порядка на величину 2^k, где k – число разрядов порядка. При этом смещенный порядок всегда является положительным числом и поэтому его знак не указывается.

Пример формата *n*-разрядного числа с плавающей точкой и смещенным порядком представлен на рис. 2.4. Для знака числа отводится старший разряд, *k* разрядов занимает смещенный порядок, а остальные $n - k - 1$ выделяются под мантиссу. Диапазон и точность представляемых в форме с плавающей точкой чисел зависят от формата.

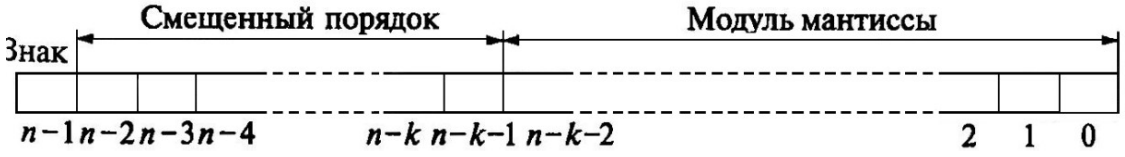


Рис. 2.4. Формат числа с плавающей точкой

Рекомендуемые стандартом основные форматы чисел с плавающей точкой представлены на рис. 2.5. Одинарный формат занимает 32 разряда, двойной – 64. Обычно для повышения точности используют способ скрытой единицы.

Суть способа заключается в том, что в нормализованном числе старший разряд мантиссы всегда равен единице, поэтому его можно не записывать, а подразумевать. Освободившийся разряд используется для записи дополнительного разряда мантиссы. Перед выполнением арифметических операций подразумеваемый разряд восстанавливается.

Рис. 2.5. Основные форматы чисел с плавающей точкой:
а – одинарный; *б* – двойной

В одинарном формате под смещенный порядок отводится восемь разрядов, и под мантиссу – 24 (с учетом скрытой единицы). При этом диапазон представления чисел составляет: $10^{-38} \le |A| \le 10^{38}$. В двойном формате смещенный порядок занимает 11 разрядов, мантисса – 53, а диапазон представления чисел составляет: $10^{-308} \le |A| \le 10^{308}$.

Как и целые числа, числа с плавающей точкой могут быть записаны в упакованном формате. В микропроцессорах фирмы Intel четыре числа одинарной точности (по 32 разряда) упаковываются в группу длиной 128 разрядов. В такую же группу упаковываются два числа двойной точности (по 64 разряда).

2.1.2. Форматы двоично-десятичных чисел

Такие числа могут быть представлены в двух форматах: зонном и упакованном. В этих форматах каждая десятичная цифра и знак числа заменяются двоичной тетрадой в соответствии с используемым двоично- десятичным кодом. Количество цифр в числе может быть произвольным.

В *зонном* формате под каждую десятичную цифру отводится байт (рис. 2.6). В старшей тетраде записывается код зоны (например, код 1111), а в младшей – код цифры. В младшем байте вместо кода зоны записывается знак числа.

В *упакованном* формате байт содержит две цифры (рис. 2.7). Младшая тетрада последнего байта содержит знак числа.

Байт		Байт		…	Байт		Байт	
Зона	Цифра	Зона	Цифра		Зона	Цифра	Знак	Цифра

а

Байт		Байт		Байт		Байт		Байт	
Зона	1	Зона	2	Зона	9	Зона	8	Минус	7
1111	0001	1111	0010	1111	1001	1111	1000	1101	0111

б

Рис. 2.6. Зонный формат двоично-десятичных чисел:
а – структура формата; *б* – пример записи числа – 12987

Байт		Байт		…	Байт		Байт	
Цифра	Цифра	Цифра	Цифра		Цифра	Цифра	Цифра	Знак

а

Байт		Байт		Байт		Байт	
0	2	3	7	4	6	5	Плюс
0000	0010	0011	0111	0100	0110	0101	1100

б

Рис. 2.7. Упакованный формат двоично-десятичных чисел:
а – структура формата; *б* – пример записи числа +237465

Пример 2.1. Запишите число –12,25 в формате с плавающей точкой.

Решение:

1. Переводим абсолютную величину числа в двоичную систему счисления: $-12{,}25_{10} = 12{,}25_{10} = 1100{,}010_2$.

2. Нормализуем до 1:

$1100{,}0102 = 1{,}1000102 \cdot 2^3$.

мантисса

Степень двойки есть абсолютный порядок.

3. Вычисляем смещенный порядок. Смещенный порядок = абсолютный порядок равен + 127:

3 + 127 = 130.

13010 = 100000102.

4. Если знак числа равен 1, то число отрицательное, если 0 – положительное.

5. Заполняем 32-разрядную сетку:

Знак	Смещенный порядок								Мантисса без первой 1																							

Пример 2.2. Значение переменной A представлено в формате с плавающей точкой в шестнадцатеричной системе $A = 430F0000_{16}$. Найдите значение числа A в десятичной системе счисления.

Решение:

Переходим от шестнадцатеричной системы счисления к двоичной системе счисления. Число занимает в памяти машины 32 бита.

$430F0000_{16} = 0100\ 0011\ 0000\ 1111\ 0000\ 0000\ 0000\ 0000_2$.

Находим абсолютный порядок:

$10000110_2 = 134_{10}$.

134 – 127 = 7 – абсолютный порядок.

Записываем мантиссу – первая единица будет всегда + 7 разрядов в поле «Мантисса без первой 1».

$+1.00011110000_2 \cdot 2^7 = +10001111 = 71$.

Ответ: 71_{10}.

Машинные коды

Для упрощения арифметических операций числа записываются в специальной форме, называемой *машинными кодами.* Эти коды позволяют:

- свести операцию вычитания к операции сложения;
- автоматически получать знак суммы (разности);
- выявлять переполнение разрядной сетки.

Существуют следующие коды:

- прямой (ПК);
- обратный (ОК);
- дополнительный (ДК).

1. Что такое система счисления?
2. Что такое основание системы счисления?
3. Как зависит разрядность чисел от величины основания системы счисления?
4. Почему информация в компьютере представляется в двоичной системе счисления?
5. Какие системы счисления используются для представления информации в компьютере?
6. Для чего применяются двоично-десятичные коды?
7. Чем отличаются формы представления чисел с фиксированной и плавающей точками?
8. Как кодируются знаки чисел?
9. Можно ли различить форматы целых и дробных чисел с фиксированной точкой?
10. Какие элементы формата чисел с плавающей точкой вы знаете?
11. Чем отличается нормализованное число с плавающей точкой от ненормализованного числа?
12. Для чего применяются смещенные порядки?
13. Какой прием используется для расширения диапазона чисел с плавающей точкой?
14. В чем заключается способ скрытой единицы?
15. Охарактеризуйте форматы двоично-десятичных чисел.
16. Как кодируются знаки двоично-десятичных чисел?
17. Для чего используются машинные коды чисел?
18. Чем отличаются дополнительный и обратный коды отрицательных чисел?
19. Как из дополнительного кода отрицательного числа получить прямой код числа?
20. Какую позицию запятой в формате с фиксированной запятой можно считать общепринятой?
21. Чем в формате с фиксированной запятой заполняются избыточные старшие разряды?
22. Какое минимальное количество полей должен содержать формат с плавающей запятой?
23. Как в формате с плавающей запятой решается проблема работы с порядками, имеющими разные знаки?
24. От чего зависят точность и диапазон представления чисел в формате с плавающей запятой?

Ссылки на литературные источники, приведенные в рабочей программе дисциплины: пп. 1, 2 основной литературы, интернет-источники.

Вопросы для закрепления:

1. Что такое **Вычислительная сеть**?
2. Чем является любая вычислительная сеть?
3. Что такое **Прозрачность, Интегрируемость**?.

Домашнее задание:

Проработка конспекта пройденной темы.

Литература:

1. Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 c

Раздел 3. Конструктивные узлы вычислительных машин. Счётчики

1. Составные части компьютера
2. Логические элементы
3. Триггеры
4. Контрольные вопросы

Современные компьютеры представляют собой технические системы, отличающиеся сложной структурой, большим числом электронных элементов и электромеханических деталей, а также сложностью выполняемых функций. При множестве электронных элементов, используемых в компьютере, число их типов сравнительно невелико – все устройства имеют регулярную структуру, т.е. состоят из большого числа типовых схем.

Преобразование информации в компьютере выполняется при помощи электронных схем, имеющих различную сложность. По функциональной сложности принято делить электронные схемы компьютера на элементы, узлы и устройства.

Элемент – это простейшая часть компьютера, выполняющая операции над двоичными цифрами (битами). Основные элементы могут быть логическими или элементами памяти. Логические элементы выполняют двоичные (бинарные) операции, на основе которых осуществляются практически все преобразования информации. В качестве логических элементов используются элементы И, ИЛИ, НЕ, И-НЕ, ИЛИ- НЕ, И-ИЛИ-НЕ и т.д. Элементы памяти чаще всего представляют собой триггеры различных типов: *RS*-, *JK*-, *D*- или *T*-триггеры. Кроме логических элементов и элементов памяти в состав компьютера входят вспомогательные элементы, усиливающие или формирующие сигналы стандартной формы.

Узлы состоят из элементов и выполняют операции над байтами или словами, состоящими из нескольких байтов. К типовым узлам компьютера относятся регистры, счетчики, сумматоры, дешифраторы, селекторы, мультиплексоры и др. Несколько узлов могут объединяться в функциональные блоки (например, блок сумматора может включать в себя собственно сумматор и регистр сумматора).

Устройства компьютера строятся из элементов и узлов и выполняют определенный набор однотипных операций. К устройствам относятся запоминающие устройства, арифметико-логическое устройство, центральное устройство управления, устройства ввода и вывода. Устройства компьютера конструктивно выполняют отдельно или

несколько устройств объединяют в один конструктивный блок (например, системный блок может объединять устройство управления, арифметико- логическое устройство и устройства памяти).

В зависимости от состава узлы могут быть комбинационного (комбинационные схемы – КС) или накапливающего типа (автоматы с памятью, последовательные схемы).

Узлы комбинационного типа состоят из логических элементов. Их главная особенность заключается в том, что выходной сигнал (Y) зависит только от комбинации входных сигналов (X) в данный момент времени, при этом каждой комбинации сигналов на входе соответствует выходной сигнал. Выходной сигнал может измениться только при получении другого входного сигнала. При неоднократном повторении одного и того же входного сигнала значение выходного также повторяется, поэтому комбинационная схема фактически осуществляет перекодировку входных сигналов в выходные. Несмотря на кажущуюся примитивность логики работы комбинационных схем, все основные преобразования информации в компьютере выполняются с их помощью. Это объясняется тем, что в основе преобразования данных в компьютере заложено выполнение логических операций.

Автоматы с памятью состоят из логических элементов и элементов памяти. Информация, записанная в памяти автомата, называется *состоянием автомата* (Q). Выходной сигнал автомата в общем случае зависит от сигнала на входе и состояния автомата, поэтому при одном и том же входном сигнале автомат может выдавать различные выходные сигналы. При работе автомата в его памяти накапливается обобщенная информация о всех входных сигналах, поступивших к данному моменту времени, поэтому состояние автомата и выходной сигнал зависят от всей предыстории входных сигналов. Наличие памяти позволяет автомату выполнять не только отдельные операции, но и последовательности взаимосвязанных операций, т.е. заданные алгоритмы обработки данных. Компьютер в целом представляет собой сложный автомат с памятью большой емкости.

4.1. Логические элементы

Логические элементы выполняют преобразования информации, которые описываются логическими функциями. Логическая функция и ее аргументы могут принимать только два значения: «0» и «1». Зависимость значений логической функции от значений ее аргументов можно задать с помощью таблицы истинности.

Элементы И, ИЛИ и НЕ (рис. 4.1) составляют функционально полную систему, т.е. из них можно составить любую комбинационную (логическую) схему. Логические элементы, выполненные в виде интегральных схем, обычно реализуют логические функции И-НЕ (функция Шеффера) или ИЛИ-НЕ (стрелка Пирса). Каждый из этих элементов обладает свойством функциональной полноты. Обозначения элементов И-НЕ и ИЛИ-НЕ на функциональных схемах, а также примеры их реализации приведены на рис. 4.1.

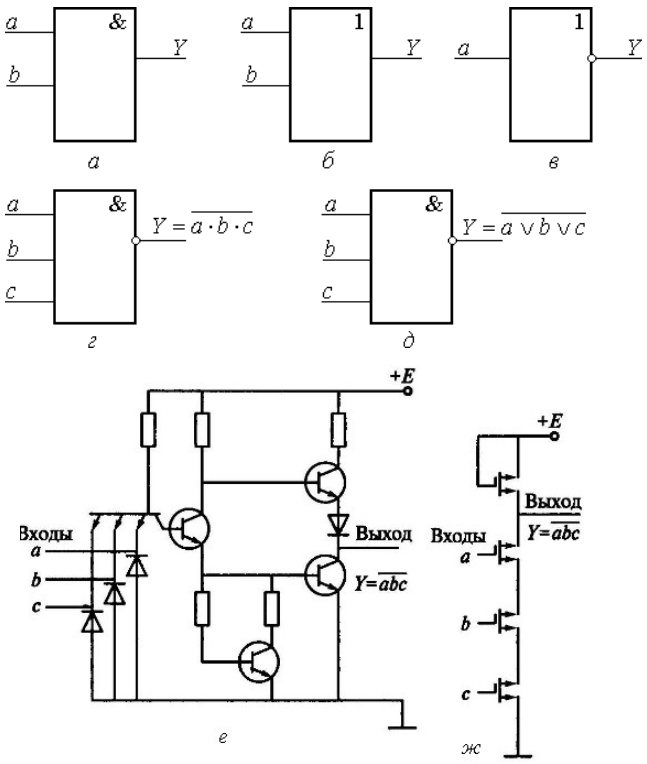


Рис. 4.1. Логические элементы:

а – И; *б* – ИЛИ; *в* – НЕ; *г* – трехвходовой И-НЕ; *д* – трехвходовой ИЛИ-НЕ;

е – принципиальная электрическая схема элемента И-НЕ на ТТЛ;

ж – принципиальная электрическая схема элемента И-НЕ на КМОП

Элементы И на два входа часто используют в качестве ключей (вентилей), которые управляют передачей данных между двумя схемами (рис. 4.2).

При передаче одноразрядных данных (рис. 4.2, *а*) один вход элемента И является информационным, а второй – управляющим. На информационный вход поступают одноразрядные данные *D* от источника *A*. Если на управляющем входе сигнал «Передать» равен «0», то на приемник *B* поступает сигнал «0» независимо от значения данных *D.* Если сигнал

«Передать» равен «1», то сигнал на выходе элемента И совпадает с информационным сигналом и на вход приемника поступают данные *D*.

При передаче многоразрядных данных (рис. 4.2, *б*) каждый разряд данных проходит через отдельный элемент И, а сигнал «Передать» подается одновременно на управляющие входы всех ключей.

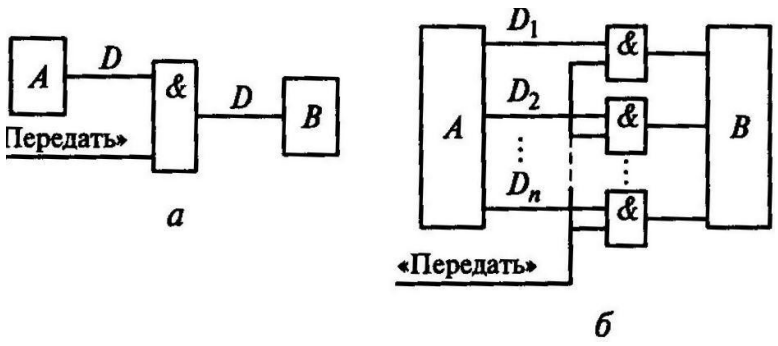


Рис. 4.2. Схемы передачи данных:

а – одноразрядных; *б* – многоразрядных

4.2. Триггеры

4.2.1. Общие сведения о триггерах

Для хранения информации в компьютере могут использоваться различные типы элементов памяти. В зависимости от способа хранения информации элементы памяти могут быть статическими, позволяющими хранить двоичную информацию сколь угодно долго, и динамическими, хранящими информацию в течение ограниченного отрезка времени. В качестве статических элементов памяти в настоящее время применяют триггеры.

Основу триггера составляет бистабильная ячейка, имеющая два устойчивых состояния. Бистабильные ячейки могут быть построены на двух логических элементах И-НЕ или ИЛИ-НЕ, соединенных перекрестными связями (рис. 4.3).

Существование двух устойчивых состояний бистабильной ячейки объясняется наличием в ее схеме обратных связей, позволяющих сигналу с выхода элемента поступать на его же вход через второй элемент. Так, если на рис. 4.3, *а* сигнал на верхнем выходе равен «1» и на оба входа подается сигнал «0», то сигнал «1» с выхода элемента 1 поступает на вход элемента 2 и формирует на его выходе сигнал «0». Этот сигнал поступает на вход элемента 1 и поддерживает такое состояние схемы, делая его устойчивым. В этом состоянии на выходе элемента 1 сигнал равен «1», а на выходе элемента 2 сигнал равен «0».

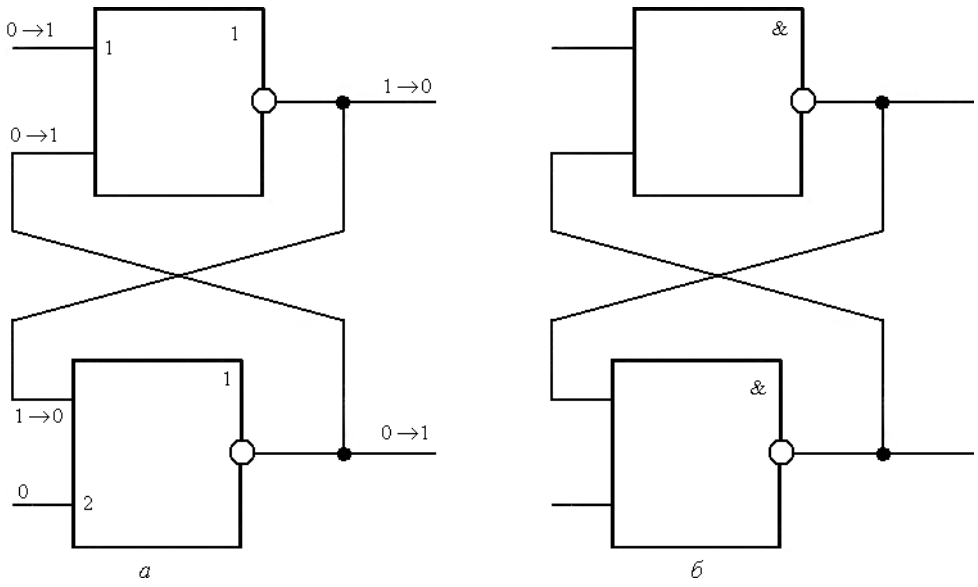


Рис. 4.3. Бистабильная ячейка:

а – на элементах ИЛИ-НЕ; *б* – на элементах И-НЕ

Для изменения состояния схемы необходимо подать на верхний вход элемента 1 сигнал «1». При этом на выходе элемента 1 сигнал становится равным «0». Тогда на выходе элемента 2 формируется сигнал «1», который вместе с единичным входным сигналом устанавливает выходной сигнал элемента 1 равным «0». Такое состояние схемы также является устойчивым и после того, как сигнал на входе элемента 1 станет равным

«0». В этом состоянии на выходе элемента 1 сигнал равен «0», а на выходе элемента 2 – «1». При поступлении сигнала «1» на вход элемента 2 происходит возвращение схемы в начальное состояние. Таким образом, схема имеет два устойчивых состояния, которые можно устанавливать подачей сигнала «1» на вход элемента 1 или 2. Аналогично работает бистабильная ячейка на элементах И-НЕ (рис. 4.3, *б*).

Триггер – это цифровая электронная схема с двумя устойчивыми состояниями, которые устанавливаются при подаче соответствующей комбинации входных сигналов и сохраняются.

Кроме бистабильной ячейки в состав триггера входит схема Управления (рис. 4.4). *Схема управления* – это комбинационная схема, при помощи которой осуществляется запись информации в триггер (изменение состояний триггера). Конкретный вид схемы Управления зависит от типа триггера.

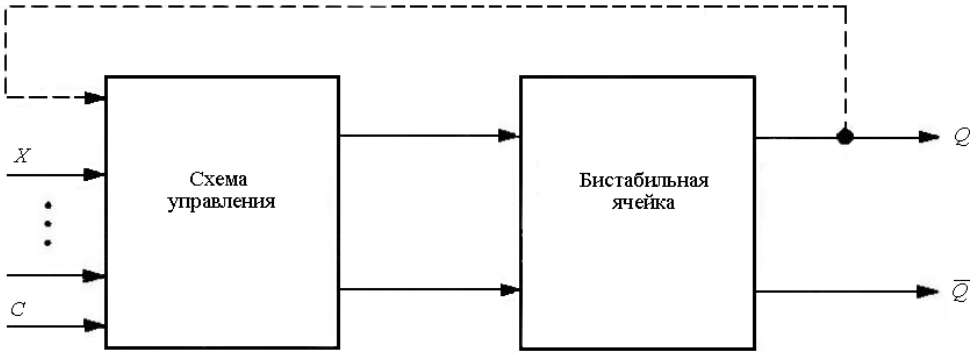


Рис. 4.4. Общая структура триггера

Триггер имеет два выхода: прямой и инверсный *(Q* и *Q).* Сигналы на выходах триггера всегда имеют различные значения. Если на прямом выходе сигнал равен «1», то на инверсном – «0» и наоборот. Состояние триггера определяется значением сигнала на прямом выходе *(Q).*

Если сигнал на прямом выходе равен «1», то триггер находится в состоянии «1». Можно также сказать, что *состояние триггера* – это информация, записанная в триггере. Таким образом, если триггер находится в состоянии «1», то в нем записана единица.

Триггеры могут быть асинхронными или синхронными. В *асинхронных* триггерах используются только основные или информационные входы. Изменение состояния асинхронного триггера может происходить в произвольные моменты времени, определяемые изменениями сигналов на информационных входах.

В *синхронных* триггерах кроме информационных входов имеется вход синхронизации. На этот вход подается сигнал синхронизации *C*, который выполняет функции сигнала, разрешающего переключение триггера из одного состояния в другое. Если сигнал синхронизации *C* равен «0», то состояние синхронного триггера не изменяется при любой комбинации сигналов на информационных входах. Для переключения синхронного триггера необходимо подать на информационные входы определенную, зависящую от типа триггера, комбинацию сигналов и, кроме того, установить значение сигнала *C*, равное «1».

Логика переключения триггера определяется его типом и зависит от числа и назначения входов. Наиболее часто в цифровой технике используют *RS*-, *JK*-, *D*– и *T*-триггеры, а также комбинированные триггеры. Буквами *R, S, J*, *K, D* и *T* обозначаются информационные входы триггеров (*X*).

4.2.2. Асинхронный *RS*-триггер

Он имеет два информационных входа R и S. Вход S (*set*) используется для установки триггера в состояние «1», а вход R (*reset*) – в состояние «1», поэтому *RS*-триггер называют *триггером с установочными входами*. Работу триггера описывает таблица переходов (табл. 4.1).

Таблица 4.1

Переходы *RS*-триггера

Входы		Состояния	
R	S	0	1
0	0	0	1
0	1	1	1
1	0	0	0
1	1	–	–

Входами служат значения входных сигналов R и S, а так же значения состояний триггера в текущий момент времени (Q_t). В таблице переходов приведены значения состояний триггера в следующий момент времени (Q_{t+1}). Переходы триггера из одного состояния в другое происходят, если на вход R или S подается сигнал «1».

При $R = 0$ и $S = 0$ состояние триггера не меняется. Такой режим называется *режимом хранения*. В случае если $R = 0$ и $S = 1$ триггер переходит в состояние «1» независимо от того, в каком состоянии он находился до изменения входных сигналов. При $R = 1$ и $S = 0$ триггер переходит в состояние «0». Таким образом, для записи «1» в *RS*-триггер необходимо подать на его входы сигналы $R = 0$ и $S = 1$, для записи «0» – сигналы $R = 1$ и $S = 0$. Комбинация сигналов R=1 и S =1 является запрещенной, состояние триггера при этом не определено.

Таблица переходов триггера может быть интерпретирована как таблица истинности комбинационной схемы, в которой значения сигналов на входах R_t, S_t и значение текущего состояния Q_t можно рассматривать как логические переменные, а $Q_t + 1$ – как логическую функцию (табл. 4.2).

Таблица 4.2

Таблица переходов *RS*-триггера

Входы		Текущее состояние	Следующее состояние
R	*S*	Q_t	Q_t+1
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	Запрещенная комбинация
1	1	1	

RS-триггер может быть построен на различных логических элементах. Функциональная схема асинхронного *RS*-триггера, построенного на элементах И-НЕ и ИЛИ-НЕ, а также его графическое обозначение показаны на рис. 4.5.

Асинхронный *RS*-триггер представляет собой бистабильную ячейку, поэтому он используется как основа при построении всех триггеров.

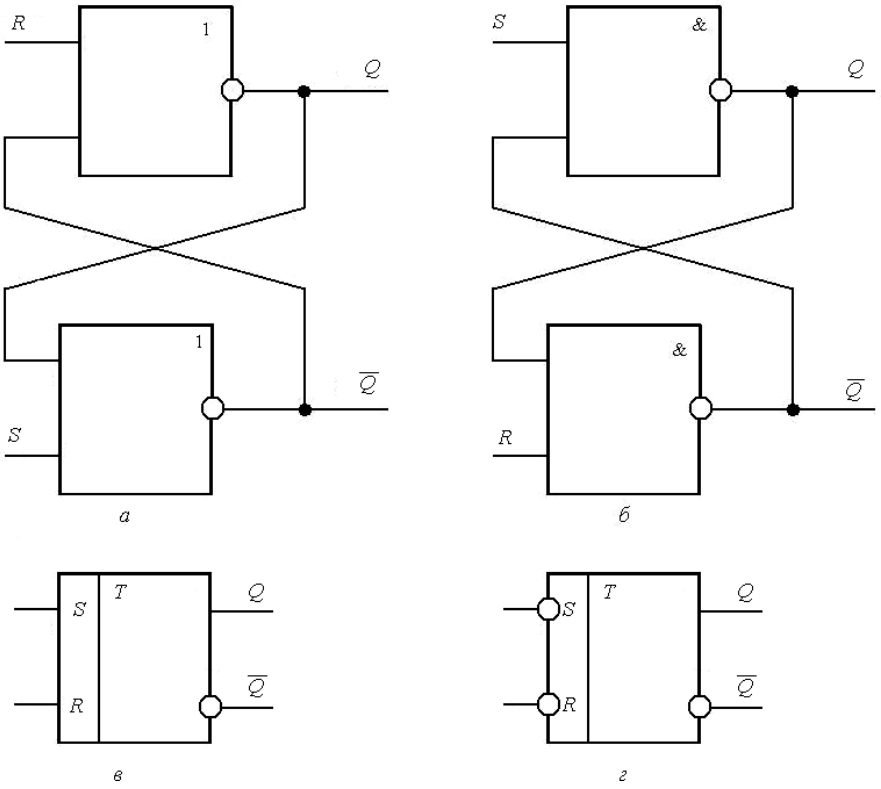


Рис. 4.5. Асинхронный *RS*-триггер:

а – на элементах ИЛИ-НЕ; *б* – на элементах И-НЕ;

в – условное графическое обозначение *RS*-триггера с прямыми входами;

г – условное графическое обозначение *RS*-триггера с инверсными входами

4.2.3. Синхронный *RS*-триггер

Этот триггер имеет дополнительно вход C, на который поступает синхросигнал. Информационные сигналы R и S могут изменять состояние триггера только при значении синхросигнала $C = 1$. Таблица переходов синхронного *RS*-триггера состоит из двух частей. Первая часть таблицы описывает переходы триггера при $C = 1$ и совпадает с таблицей переходов асинхронного триггера (табл. 4.1). Когда $C = 0$, триггер не меняет своего состояния при любой комбинации сигналов на информационных входах и логика его переходов может быть описана табл. 4.3.

При $C = 0$ разрешенными являются любые комбинации входных сигналов, в том числе $R = 1$, $S = 1$.

На рис. 4.6 приведены функциональные схемы синхронных *RS*-триггеров, реализованных на элементах И-НЕ и И-ИЛИ-НЕ, и их условное графическое обозначение. Кроме основных входов R и S там показаны дополнительные входы R_1 и S_1, которые являются асинхронными. При подаче сигналов на них состояние триггера может изменяться независимо от значения сигнала C. В каждый момент времени можно управлять переходами триггера только с помощью синхронных или только с помощью асинхронных входов.

Таблица 4.3

Переходы синхронного *RS*-триггера

Входы			Состояния	
R	S	C	0	1
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	1	1
1	1	0	0	0
1	1	1	Запрещенная комбинация	

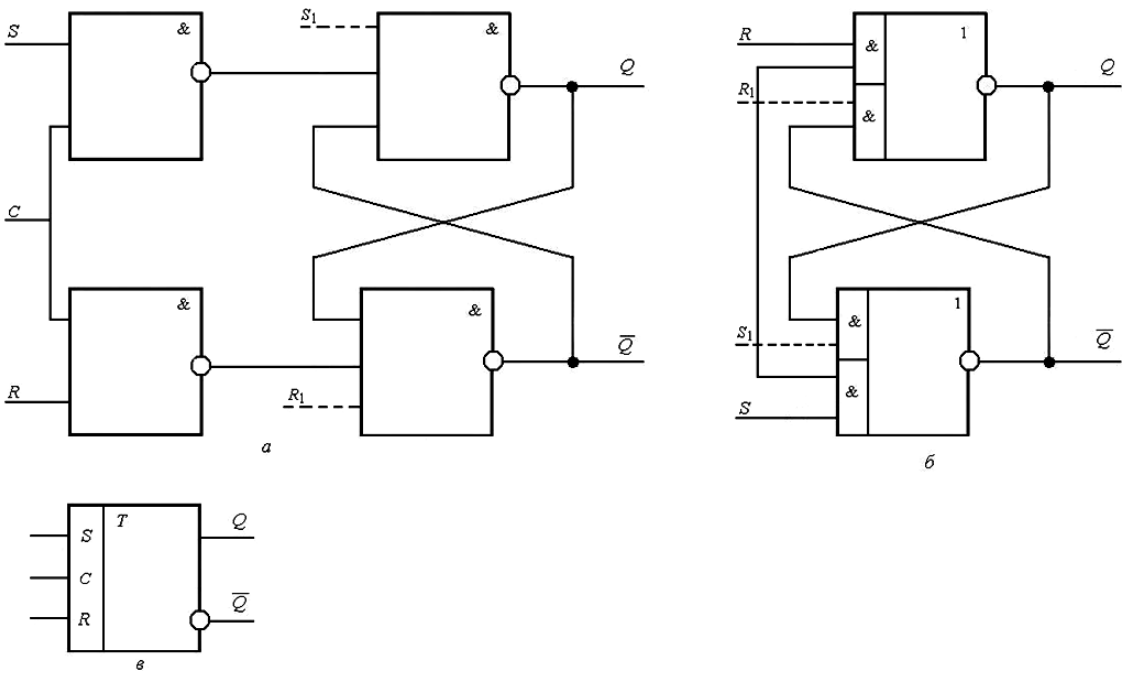


Рис. 4.6. Синхронный *RS*-триггер:

а – на элементах И-НЕ; *б* – на элементах И-ИЛИ-НЕ;

в – условное графическое обозначение

4.2.4. Двухтактный *RS*-триггер

Триггеры используются в различных узлах ЭВМ, между которыми осуществляется передача информации. Устойчивая работа цепочки триггеров происходит только в том случае, когда запись новой информации в триггер производится после считывания прежней и передачи ее в следующий по цепочке триггер.

Это возможно при использовании двух серий синхроимпульсов, сдвинутых относительно друг друга на полпериода. Такой принцип управления и синхронизации применяется в двухтактных триггерах. Двухтактные триггеры используются в сдвигающих регистрах, а также в качестве элементов памяти в цифровых автоматах с памятью для устранения эффекта гонок.

Простейшая схема двухтактного *RS*-триггера может быть построена из двух однотактных, причем синхросигналы на входы *C* первого и второго триггеров должны подаваться в противофазе. Это делается с помощью инвертора (рис. 4.7, *а*).

В основном поле условного графического обозначения двухтактного триггера записываются две буквы *T* (рис. 4.7, *б*). Особенности переключения двухтактного триггера из одного состояния в другое поясняются временной диаграммой (рис. 4.7, *в*).

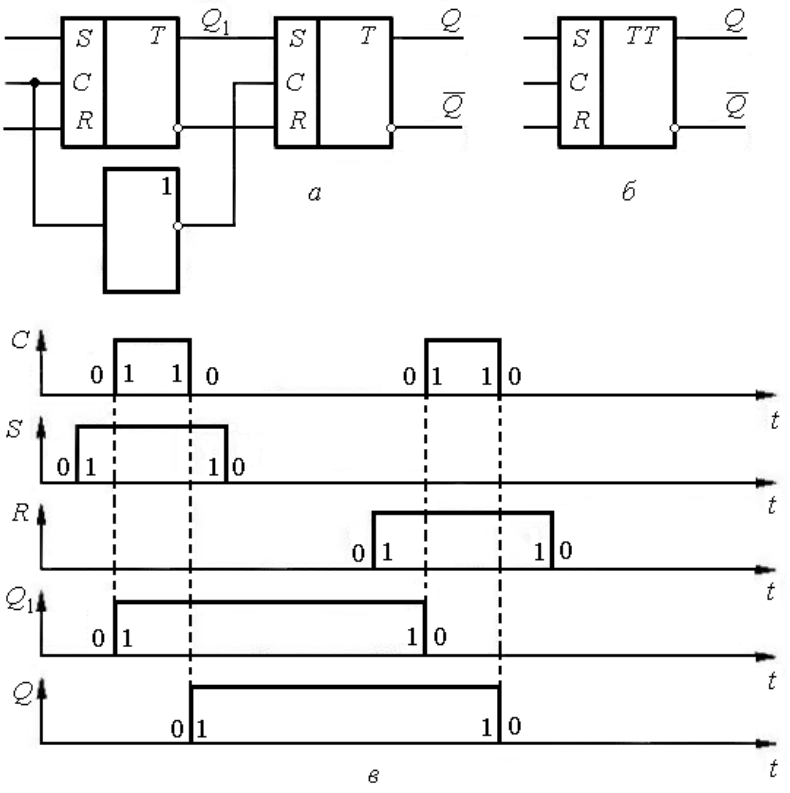


Рис. 4.7. Двухтактный *RS*-триггер:

а – схема; *б* – условное графическое обозначение;

в – временная диаграмма работы

Пусть оба триггера находятся в состоянии «0» и на входы триггера поступают сигналы $S = 1$ и $R = 0$ (запись в триггер сигнала «1»). При поступлении на вход *RS*-триггера сигнала $C = 1$ входная информация по переднему фронту сигнала *C* запоминается в первом однотактном триггере (он переходит в состояние «1»). Второй однотактный триггер хранит информацию о предыдущем состоянии, так как на его входе C = 0.

По окончании действия синхросигнала (по заднему фронту), т.е. при $C = 0$, первый триггер переходит в режим хранения, а информация с выходов первого триггера передается на вход второго триггера. Так как на входе второго триггера сигнал $C = 1$, он также переходит в состояние «1». В результате к началу следующего такта на выходе двухтактного *RS*-триггера появится сигнал состояния, соответствующего входной информации. Аналогичным образом производится запись в двухтактный триггер сигнала нуля. Для установки *RS*-триггера в «0» или

«1» независимо от присутствия сигнала на входе *C* в схему вводят прямые или инверсные входы *R* и *S* асинхронной установки (рис. 4.8, *а*) и отображают их на условном графическом обозначении (рис. 4.8, *б*).

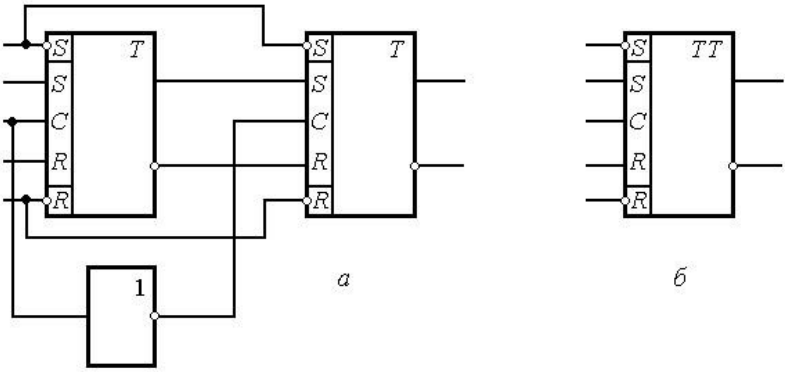


Рис. 4.8. Двухтактный *RS*-триггер с дополнительными входами *R* и *S*:
а – схема; *б* – условное графическое обозначение

4.2.5. Асинхронный и синхронный *D*-триггеры

В вычислительной технике широко применяют *D*-триггеры, которые реализуют функцию временной задержки входного сигнала. Также *D*-триггеры имеют один информационный вход. Логика работы асинхронного *D*-триггера описывается таблицей переходов (табл. 4.4).

В асинхронном *D*-триггере состояние (выходной сигнал) Q_{t+1} повторяет значение входного сигнала D_t, поэтому асинхронный *D*-триггер по существу не является элементом памяти и рассматривается только как основа для построения синхронного *D*-триггера.

Таблица 4.4

Переходы асинхронного *D*-триггера

Вход	Состояния	
D	0	1
0	0	0
1	1	1

Функциональная схема и условное графическое обозначение синхронного *D*-триггера, построенного на основе синхронного *RS*-триггера, показаны на рис. 4.9. Для преобразования *RS*-триггера в *D*-триггер сигнал *D* подается на вход *S* непосредственно, а на вход *R* – через инвертор. Если при *C* = 1 на вход *D* подать сигнал «1», то триггер перейдет в состояние «1», а при подаче сигнала *D* = 0 в триггер будет записан «0». Таким образом, для записи в *D*-триггер единицы на вход *D*

нужно подать сигнал «1», а для записи нуля – сигнал «0» (так как триггер синхронный, на вход *C* необходимо в обоих случаях подавать сигнал «1»). Это делает *D*-триггер удобным для использования в схемах статической памяти, так как для записи достаточно иметь одну линию на разряд данных. При этом сигнал *C* является общим для всех разрядов записываемых данных.

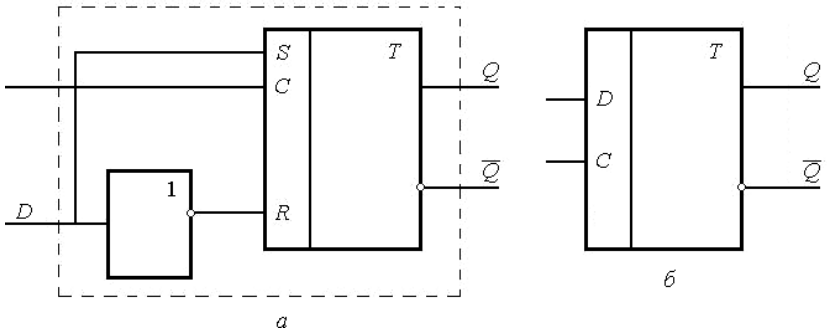


Рис. 4.9. Синхронный *D*-триггер:

а – схема; *б* – условное графическое обозначение

Логику работы синхронного *D*-триггера описывает табл. 4.5. Эту логику можно охарактеризовать выражением «что надо записать в *D*-триггер, то и подается на его вход».

Таблица 4.5

Переходы синхронного *D*-триггера

Входы		Состояния	
D	*C*	0	1
0	0	0	1
0	1	0	0
1	0	0	1
1	1	1	1

Наличие входа синхронизации позволяет записывать новые данные в триггер только в определенные моменты времени (при *C* = 1). В промежутках между ними данные в триггере сохраняются без изменения. При чтении данных из триггера его состояние также не меняется.

4.2.6. *T*-триггер

Этот триггер имеет один информационный вход. Логику работы асинхронного *T*-триггера характеризует таблица переходов (табл. 4.6).

Таблица 4.6

Переходы асинхронного *T*-триггера

Входы	Состояния	
T	0	1
0	0	1
1	1	0

При $T = 1$ асинхронный *T*-триггер меняет свое состояние на противоположное, а при $T = 0$ состояние триггера не изменяется.

Так как *T*-триггер суммирует (или подсчитывает) по модулю два числа единиц, поступающих на его информационный вход, то *T*-триггер называют также *триггером со счетным входом*. Логику работы синхронного *T*-триггера описывает табл. 4.7.

Таблица 4.7

Переходы синхронного *T*-триггера

Входы		Состояния	
C	*T*	0	1
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

При $C = 0$ триггер не изменяет своего состояния, а при $C = 1$ работает как асинхронный *T*-триггер.

Функциональная схема *T*-триггера может быть построена на основе синхронного *RS*-триггера (однотактного или двухтактного). Схемы асинхронного и синхронного *T*-триггеров показаны на рис. 4.10 и 4.11 соответственно.

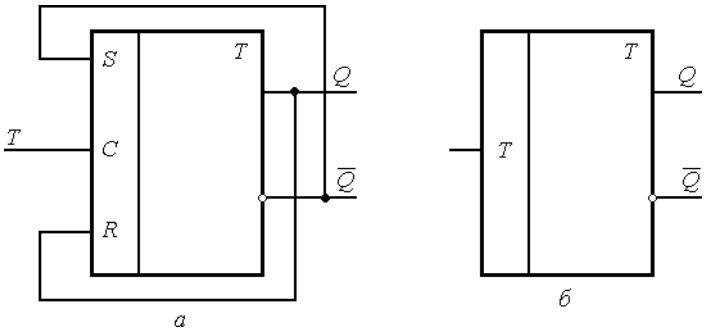


Рис. 4.10. Асинхронный *T*-триггер:

а – схема; *б* – условное графическое обозначение

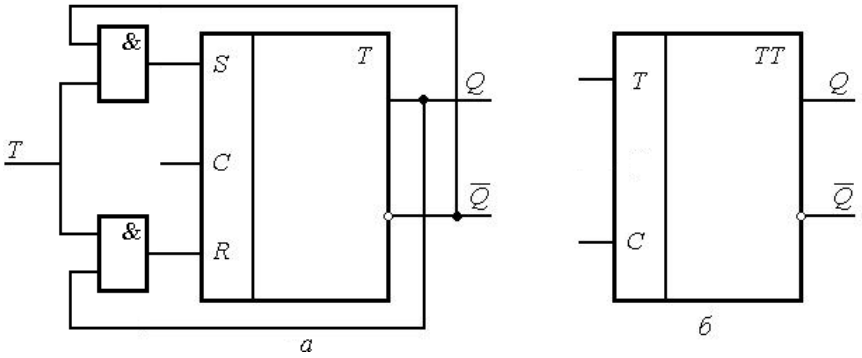


Рис. 4.11. Синхронный N-триггер:

а – схема; *б* – условное графическое обозначение

Поскольку на этих схемах сигнал с выхода триггера поступает на его же вход, триггер должен во время переключения сохранять состояние и одновременно воспринять новую информацию. Для устойчивой работы в этом случае целесообразно использовать двухтактные триггеры.

4.2.7. *JK*-триггер

Такие триггеры называют *универсальными.* Универсальность схемы *JK*-триггера состоит в том, что простой коммутацией входов и выходов можно получать схемы других типов триггеров.

JK -триггер имеет два информационных входа. Вход *J* используется для установки триггера в состояние «1», а вход *K* – в состояние «0», т.е. входы *J* и *K* аналогичны входам S и *R RS*-триггера. Отличие *JK*-триггер от *RS*-триггера заключается в том, что на входы *J* и *K* могут одновременно поступать сигналы «1». В этом случае *JK*-триггер изменяет свое состояние. Таким образом, он работает так же, как *RS*-

триггер, за исключением комбинации сигналов $J = 1$; $K = 1$, при которой он работает как T-триггер. При $C = 1$ переходы JK-триггера описывает табл. 4.8.

Таблица 4.8

Переходы JK-триггера

Входы		Состояния	
J	*K*	0	1
0	0	0	1
0	1	0	0
1	0	1	1
1	1	1	0

Функциональная схема двухтактного *JK*-триггера и его условное графическое изображение показаны на рис. 4.12. Этот триггер представляет собой комбинацию *RS*- и *T*-триггеров, что согласуется с логикой его работы. Примеры построения других типов триггеров на основе *JK*-триггера представлены на рис. 4.13. Следует отметить, что триггер любого типа можно преобразовать в любой другой триггер.

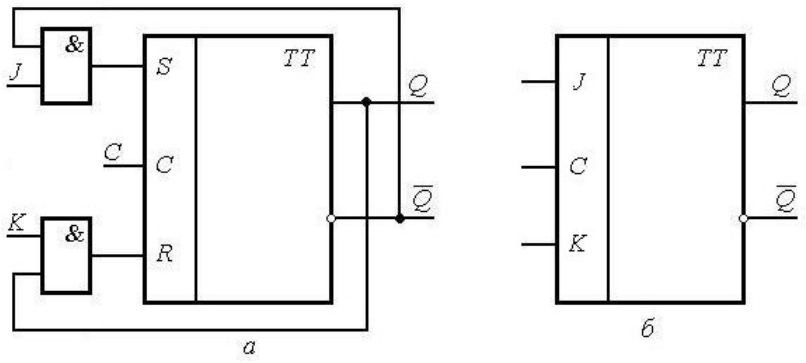


Рис. 4.12. Двухтактный JK-триггер:

а – схема; *б* – условное графическое обозначение

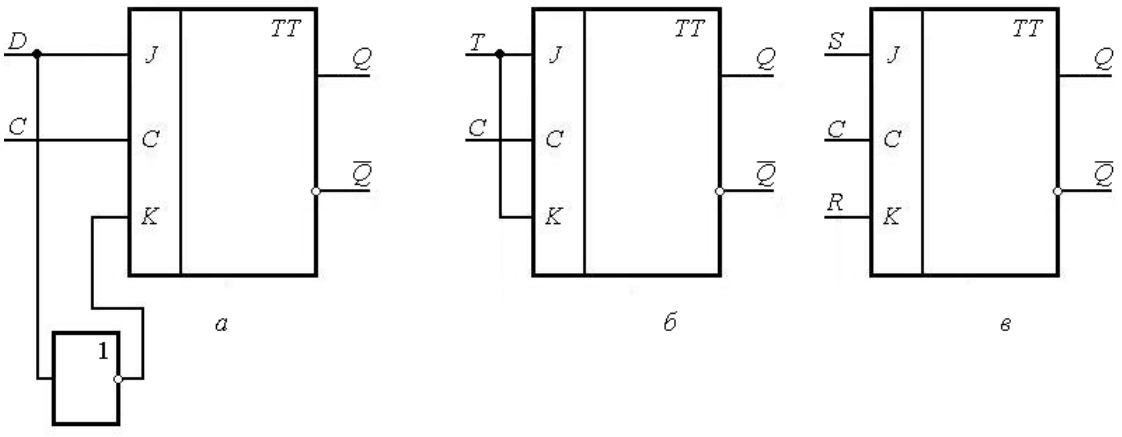


Рис. 4.13. Схемы преобразования *JK*-триггера:
а – в *D*-триггер; *б* – в *T*-триггер; *в* – *RS*-триггер

Регистры предназначены для приема, временного хранения и выдачи данных. Чаще всего данные сохраняются в регистре на время выполнения одной или нескольких машинных команд. Кроме хранения данных регистры могут выполнять и другие операции (сдвиг данных, логические операции). Основу регистра составляют триггеры, число которых равно разрядности хранимых данных.

Наиболее простую схему имеет регистр с параллельной записью данных. На рис. 6.1 приведена схема регистра, построенного на синхронных *D*-триггерах с дополнительными асинхронными входами *S* и *R*. Записываемые данные (D_0 ... D_n – 1) подаются на информационные входы триггеров. Запись производится при поступлении сигнала «Прием», подаваемого в определенный момент времени на входы синхронизации всех триггеров регистра. Для установки регистра в состояние «0» используется сигнал «Уст. 0», который подается на дополнительные входы триггеров *R*.

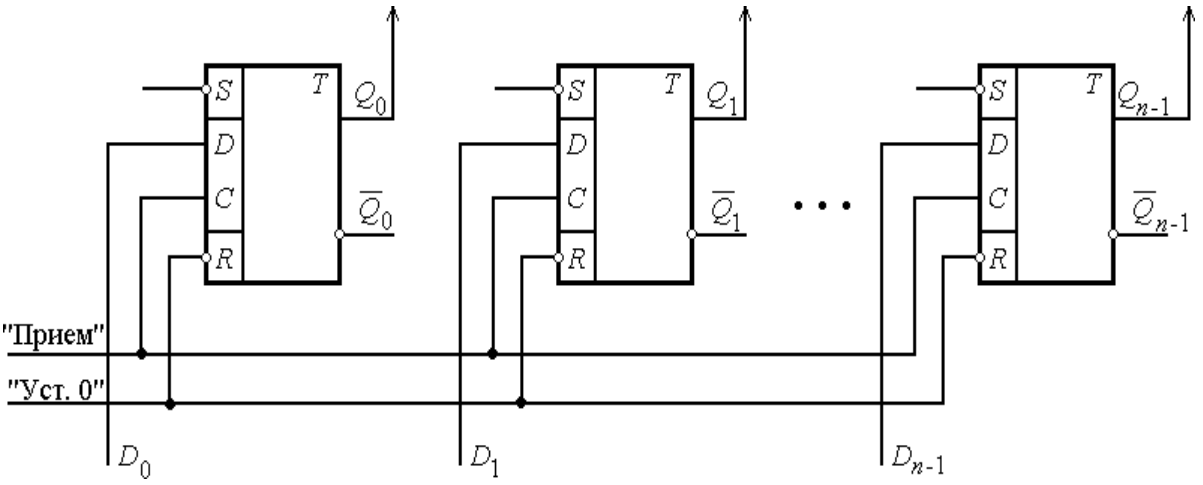


Рис. 6.1. Регистр с параллельной записью

Данные из регистра с параллельной записью и входом установки в состояние «0» могут быть выданы в прямом или обратном (инверсном) коде (рис. 6.2, *а*). Условное графическое обозначение регистра показано на рис. 6.2, *б*.

Для выдачи данных из регистра в прямом или обратном коде подается соответствующий управляющий сигнал.

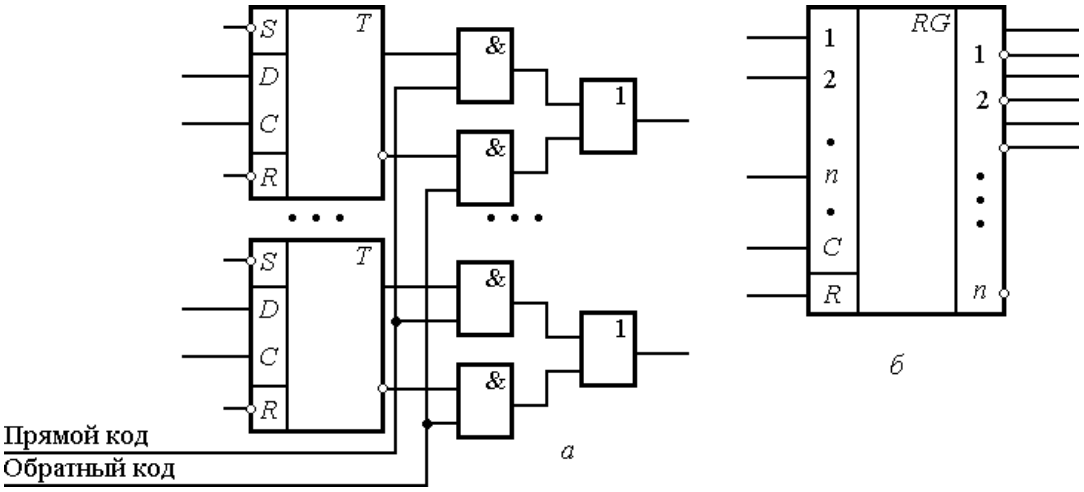


Рис. 6.2. Регистр:

а – схема выдачи данных из регистра; *б* – условное графическое обозначение

Данные обычно выдаются из одного регистра и одновременно принимаются в другой (рис. 6.3). При этом используются регистры, построенные на синхронных *RS*-триггерах, а передача данных осуществляется в парафазном коде. В этом случае прием данных в регистр Рг1 производится при поступлении сигнала ПрРг1, а запись данных из Рг1 в регистр Рг2 – при поступлении сигнала ПрРг2.

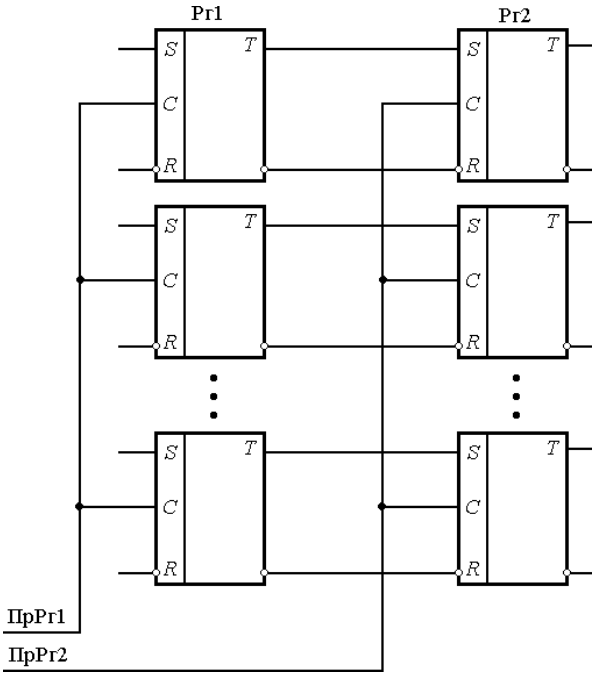


Рис. 6.3. Схема передачи данных из регистра в регистр

Передача данных между регистрами может выполняться как прямо, т.е. без изменения номеров разрядов, так и со сдвигом влево или вправо на один или несколько разрядов.

Для сдвига информации в пределах одного регистра используются сдвигающие регистры (рис. 6.4). Сложность сдвига заключается в том, что каждый разряд регистра должен одновременно принять новую и передать старую информацию. Для устойчивой работы сдвигающего регистра в каждом разряде используют два однотактных триггера или один двухтактный, т.е. сдвигающий регистр фактически состоит из двух регистров.

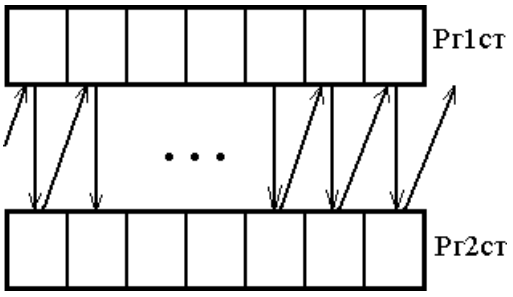


Рис. 6.4. Схема сдвига данных на один разряд вправо

Сдвигающий регистр представляет собой сдвоенный регистр и состоит из регистров первой (Рг1ст) и второй (Рг2ст) ступеней, между которыми производится обмен данными. Перед сдвигом в регистрах первой и второй ступеней записана одна и та же информация. При сдвиге данные сначала передаются из Рг2ст в Рг1ст со сдвигом вправо, затем возвращаются в Рг2ст без сдвига.

Схема регистра сдвига на один разряд вправо, выполненного на однотактных *RS*-триггерах, приведена на рис. 6.5. Сдвиг производится при поступлении сигнала «Сдвиг». Каждый сигнал «Сдвиг» вызывает сдвиг данных на один разряд. При этом в освободившийся разряд записывается «0», а данные, выходящие из регистра, теряются.

Аналогичным образом можно построить регистр сдвига вправо, регистр со сдвигом данных на несколько разрядов в одном такте или реверсивный сдвигающий регистр, позволяющий сдвигать данные влево или вправо в зависимости от управляющего сигнала.

На основе сдвигающего регистра можно построить преобразователь последовательного кода в параллельный, и наоборот (рис. 6.6).

Преобразователь выполнен на основе регистра со сдвигом данных вправо на один разряд. Регистр сдвига состоит из двухтактных *JK*-триггеров с дополнительными асинхронными входами *S* и *R*.

Для преобразования параллельного кода в последовательный регистр устанавливается в нулевое состояние сигналом «Уст. 0». Затем данные в параллельном коде поступают на дополнительные входы регистра *S*. При подаче сигнала «Сдвиг» данные в последовательном коде выдаются на последовательный выход регистра. Для преобразования последователь- ного кода в параллельный данные подаются на последовательный вход одновременно с поступлением сигналов «Сдвиг». После заполнения регистра данные могут быть выданы в параллельном коде с выходов D_1, D_2, ..., D_{n-1}.

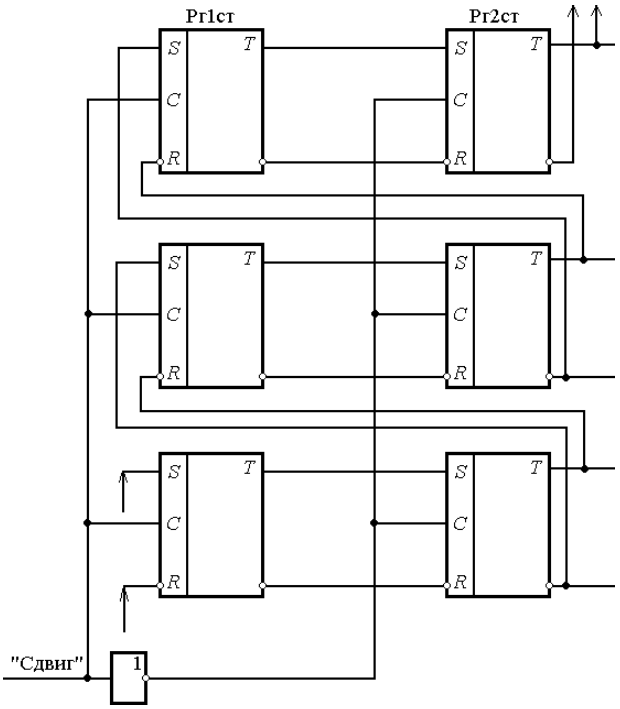


Рис. 6.5. Регистр сдвига вправо

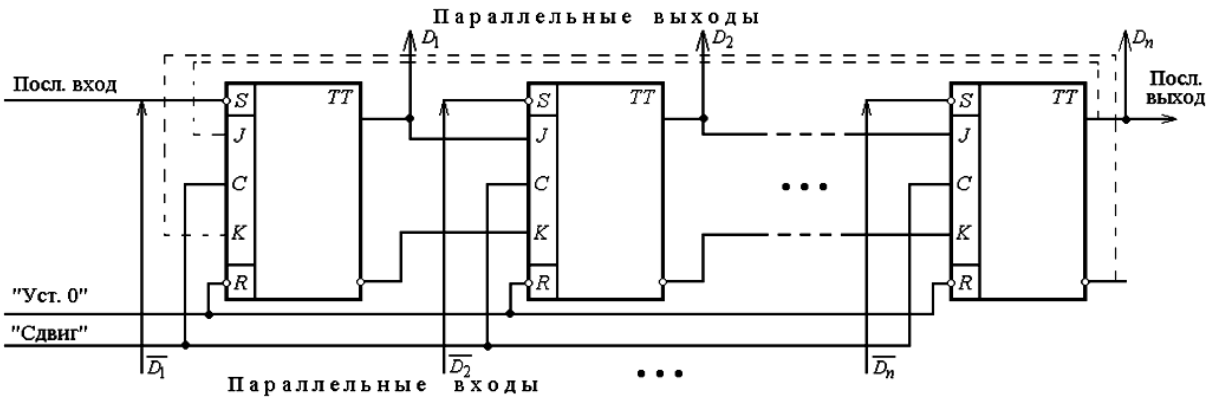


Рис. 6.6. Преобразователь параллельного кода в последовательный

Подача сигналов с выходов последнего триггера на входы *J* и *K* первого триггера позволяет получить кольцевой регистр сдвига, в котором разряды данных последовательно проходят через все триггеры, а состояние регистра периодически повторяется.

6.1. Счетчики

Счетчики предназначены для подсчета числа входных сигналов. Обычно входными сигналами являются импульсы напряжения. Результат счета – это число, представленное в двоичной или двоично- десятичной системе счисления. В двоичном счетчике число разрядов зависит от модуля счета *M* (коэффициента пересчета), который определяет число состояний счетчика. При поступлении на вход счетчика *M* импульсов счетчик возвращается в исходное состояние, после этого состояния счетчика повторяются. Модуль счета на единицу больше максимального числа $K_{\text{сч}}$, которое может зафиксировать счетчик.

В двоичных счетчиках с естественным порядком счета $M = 2^n$, где n – разрядность счетчика. Тогда $K_{\text{сч}} = 2^n - 1$. Такие счетчики имеют наиболее простую схему и строятся на *T*– или *JK*-триггерах.

В зависимости от направления счета счетчики могут быть суммирующими, вычитающими и реверсивными. *Суммирующие* счетчики при поступлении каждого входного импульса увеличивают показания на единицу, *вычитающие* – уменьшают. *Реверсивные* счетчики могут работать как в режиме суммирования, так и в режиме вычитания. Схемы трехразрядных двоичных счетчиков показаны на рис. 6.7, 6.8. Счетчики построены на двухтактных *T*-триггерах с дополнительными входами для установки счетчика в начальное состояние.

В суммирующем счетчике начальное (нулевое) состояние счетчика задается управляющим сигналом «Уст 0» (рис 6.7, *а*). Входные импульсы $U_{\text{вх}}(t)$ поступают на вход триггера младшего разряда *T*. Этот триггер изменяет состояние по спаду каждого входного импульса, остальные триггеры меняют состояние по спаду сигнала на прямом выходе триггера соседнего младшего разряда. Таким образом, второй триггер меняет состояние после каждого второго импульса, третий – после каждого четвертого и т.д. При подаче на вход счетчика последовательности импульсов счетчик меняет состояния в соответствии с временной диаграммой (рис. 6.7, *б*). Так как триггер с выходом Q_1, фиксирует младший разряд результата счета, то состояния счетчика меняются в последовательности, соответствующей логике работы суммирующего счетчика по модулю 8:

$Q_3Q_2Q_1$ □ 000 □ 001 □ 010 □ 011 □ 100 □ 101 □ 110 □ □ 111 □ 000 …

Условное графическое обозначение суммирующего счетчика показано на рис. 6.7, *в*.

В вычитающих счетчиках на входы старших разрядов подаются сигналы с инверсных выходов триггеров младших разрядов. В начальном состоянии счетчика все триггеры установлены в состояние «1» сигналом

«Уст. 1» (рис. 6.8, *а*). При поступлении входных импульсов первый триггер меняет состояние по спаду каждого входного импульса, а все остальные – по спаду сигнала на инверсном выходе триггера соседнего младшего разряда. Последовательность изменения состояний счетчика показана на временной диаграмме (рис. 6.8, *б*) и соответствует логике работы вычитающего счетчика по модулю 8:

$Q_3Q_2Q_1$ □ 111 □ 110 □ 101 □ 100 □ 011 □ 010 □ 001 □ □ 000 □ 111 …

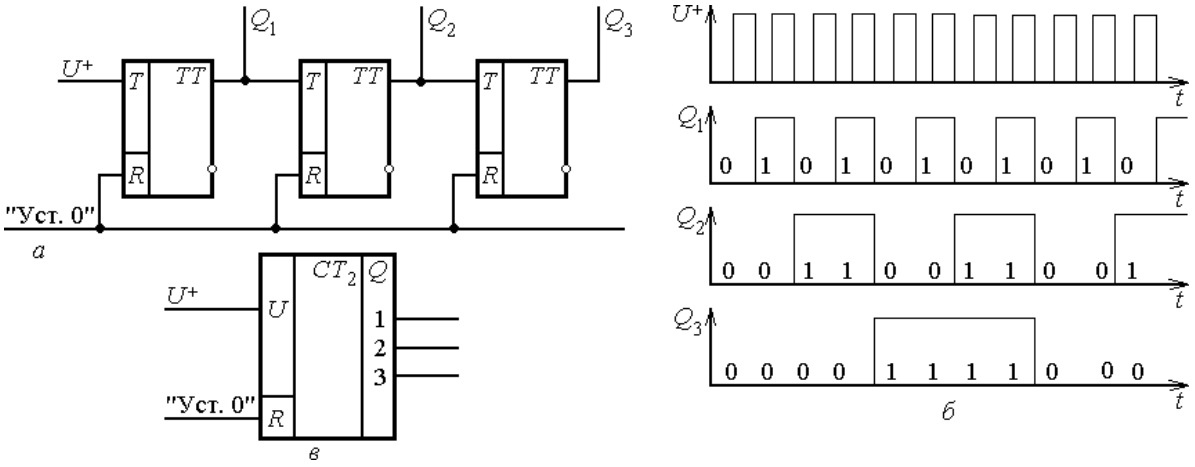


Рис. 6.7. Суммирующий счетчик:

а – схема; *б* – временная диаграмма; *в* – условное графическое обозначение

В рассмотренных счетчиках входной сигнал (в худшем случае) должен последовательно пройти через все триггеры. Такие счетчики называются *счетчиками с последовательными переносами*. Для повышения быстродействия используют схемы со сквозными или параллельными переносами.

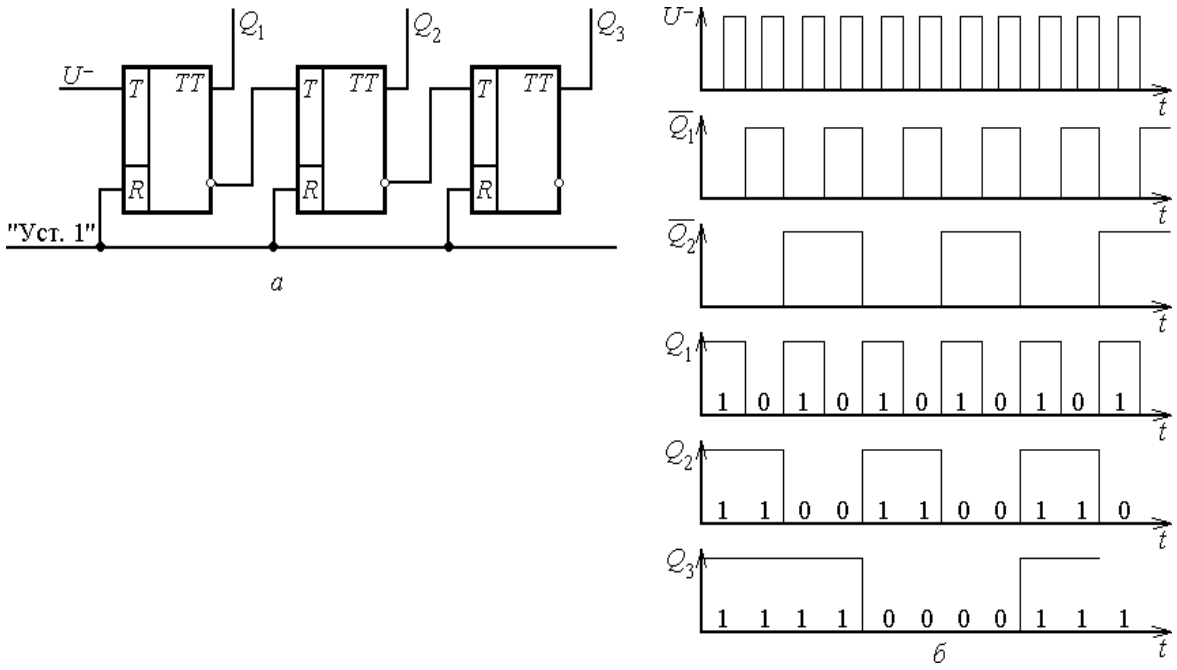


Рис. 6.8. Вычитающий счетчик:

а – схема; *б* – временная диаграмма работы

В счетчиках со сквозными переносами сигнал переноса последовательно проходит мимо каждого триггера, находящегося в состоянии «1», через вентиль на два входа. В таком счетчике время установления состояния пропорционально числу вентилей, т.е. разрядности счетчика. Схема суммирующего счетчика со сквозными переносами показана на рис. 6.9.

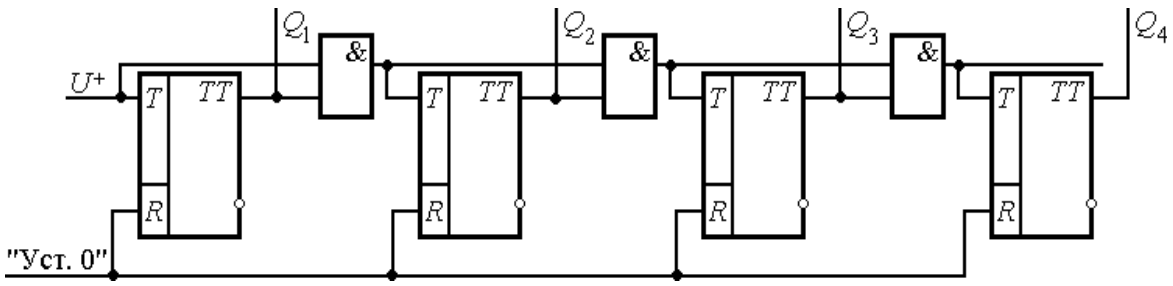


Рис. 6.9. Счетчик со сквозными переносами

Счетчики с параллельными переносами используют вентили с большим числом входов. Это позволяет сигналам переноса проходить мимо группы триггеров, находящихся в состоянии «1». Схема сумми- рующего счетчика с параллельными переносами показана на рис. 6.10.

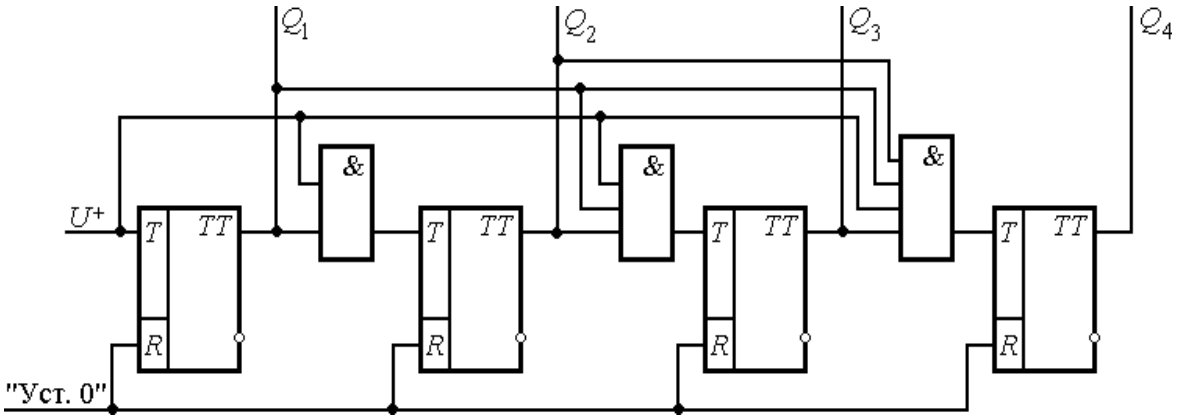


Рис. 6.10. Счетчик с параллельными переносами

Занятие 3(лекция)

Информация по теме:

Работы по стандартизации вычислительных сетей ведутся большим количеством организаций. В зависимости от их статуса различаются и виды стандартов: стандарты отдельных фирм, стандарты специальных комитетов и объединений, национальные стандарты, международные стандарты.

К организациям наиболее активно и успешно занимающихся разработкой стандартов в области вычислительных сетей относятся:

Международная организация по стандартизации (International Organization for Standardization (ISO);

Международный союз электросвязи (International Telecommunications Union, ITU) – организация, являющаяся специализированным органом ООН. Наиболее важную роль играет постоянно действующий в ней Международный комитет по телефонии и телеграфии (Consultative Committee on International Telegraphy and Telephony, CCITT). В 1993 году в результате реорганизации ITU он сменил название на сектор телекоммуникационной стандартизации ITU (ITU Telecommunications Standardization Sector – ITU-T).

Институт инженеров по электротехнике и радиоэлектронике – Institute of Electrical and Electronics Engineers, IEEE – национальная организация США, определяющая сетевые стандарты.

Ассоциация электронной промышленности (Electronic Industries Association, EIA) – промышленно-торговая группа производителей электронного и сетевого оборудования, национальная коммерческая ассоциация США.

Американский институт национальных стандартов (American National Standards Institut, ANSI) – представляет США в ISO.

В широком смысле открытой системой может называться любая система (компьютер, вычислительная сеть, аппаратные или программные продукты), которая построена в соответствии с открытыми спецификациями. Термин «спецификация» означает формализованное описание аппаратных или программных компонент, способ их функционирования, взаимодействия с другими компонентами, условий эксплуатации, ограничений и особых характеристик. Не всякая спецификация является стандартом. Открытые специфика-

ции – опубликованные, общедоступные, соответствующие стандарту и принятые после всестороннего обсуждения.

Для обеспечения обмена данными между компьютерными сетями ISO совместно с CCITT разработала многоуровневый комплект протоколов, известный как эталонная модель взаимосвязи открытых систем (Open System Interconnection – OSI) (1977 г). Одна из основных ее идей – обеспечение относительно легкого и простого обмена информацией при использовании изготовленных разными фирмами аппаратных и программных средств, соответствующих стандартам OSI. Модель OSI касается только одного аспекта открытости – открытости взаимодействия устройств, связанных в вычислительную сеть. Ярким примером открытой системы является Internet.

Домашнее задание:

Проработка конспекта пройденной темы.

Литература:

2. Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 c

Раздел 4. Фон-неймановская вычислительная машина

1. Функциональная организация фон-неймановской вычислительной машины.
2. Микрооперации и микропрограммы.
3. Контрольные вопросы.

Занятие 4 (лекция)

Представим фон-неймановскую вычислительную машину в виде гипотетической машины с аккумуляторной архитектурой (рис. 11.1). Пусть машина обладает следующими характеристиками:

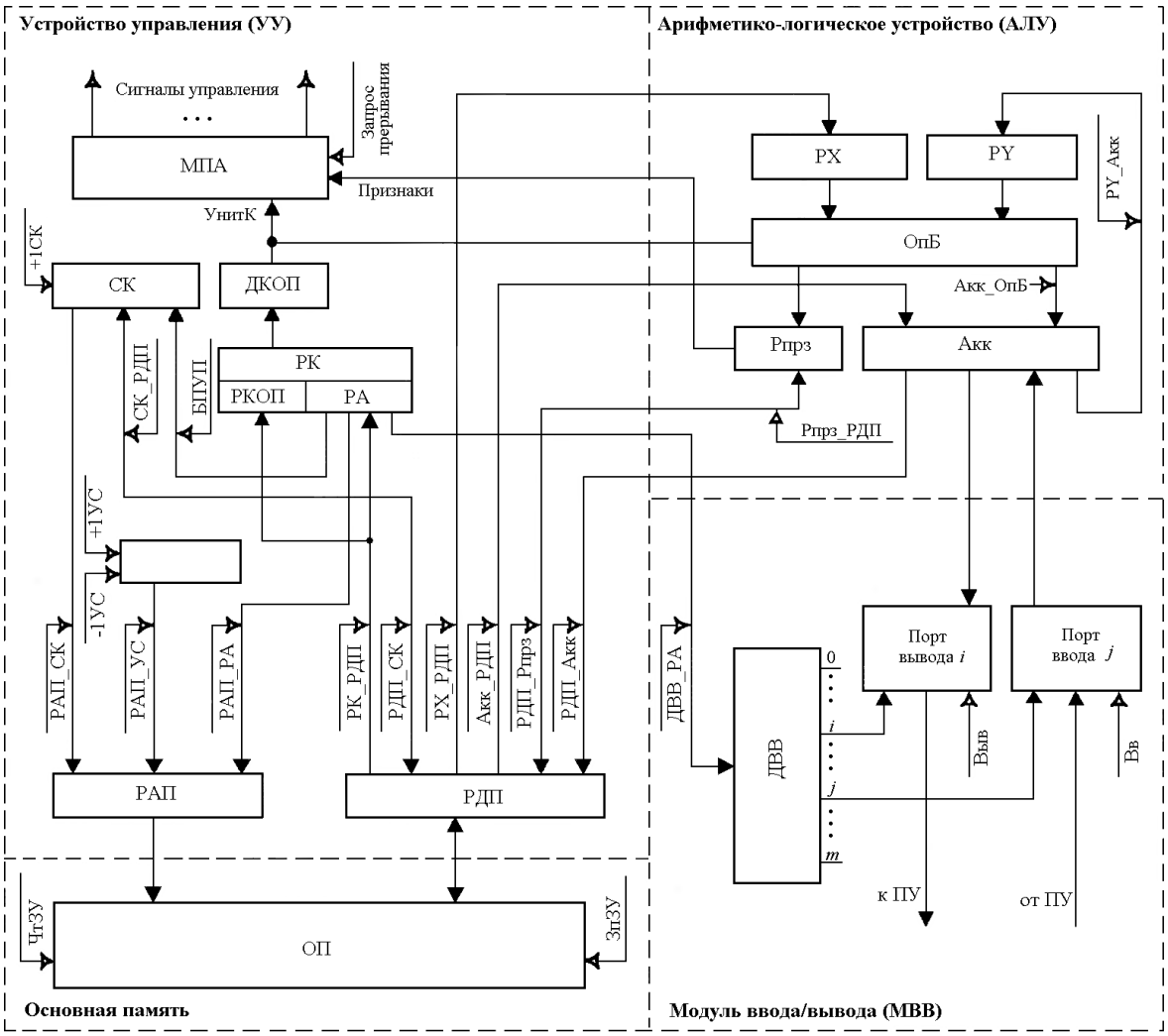


Рис. 11.1. Функциональная схема
гипотетической фон-неймановской вычислительной машины

Одноадресные команды. Адресная часть команды содержит только один адрес. При выполнении операций с двумя операндами предполагается, что другой операнд находится в специальном регистре АЛУ – аккумуляторе, а результат также остается в аккумуляторе.

Единство форматов. Длина команд и данных совпадает с разрядностью ячеек памяти, то есть любая команда или операнд занимают только одну ячейку памяти. Таким образом, адрес очередной команды в памяти может быть получен путем прибавления единицы к адресу текущей команды, а для извлечения из памяти любой команды или любого операнда достаточно одного обращения к памяти.

Список команд, выполняемых гипотетической вычислительной машиной приведен в табл. 11.1.

На функциональной схеме (рис. 11.1) показаны типовые узлы каждого из основных устройств вычислительной машины, а также сигналы, инициирующие выполнение отдельных операций по пересылке информации и ее обработке, необходимых для функционирования машины.

Таблица 11.1

Команды гипотетической вычислительной машины

Мнемони-ческое обозначение	КОП	АЧ (адресная часть)	Описание
1	2	3	4
LDA	1	ADR	Загрузка в аккумулятор содержимого ячейки основной памяти (ОП) с адресом ADR
STA	2	ADR	Запись содержимого аккумулятора в ячейку ОП с адресом ADR
ADD	3	ADR	Сложение содержимого аккумулятора и ячейки ОП, имеющей адрес ADR. Результат остается в аккумуляторе
SUB	4	ADR	Вычитание из содержимого аккумулятора числа, хранящегося в ячейке ОП, имеющей адрес ADR. Результат остается в аккумуляторе
INP	5	IPRT	Ввод в аккумулятор информации с периферий-ного устройства, подключенного к порту ввода с номером IPRT
OUT	6	OPRT	Вывод содержимого аккумулятора на перифе-рийное устройство, подключенное к порту вывода с номером OPRT
JMP	7	ADR	Безусловный переход к команде, хранящейся по адресу ADR
BRZ	8	ADR	Переход к команде, хранящейся по адресу ADR, при условии, что результат предыдущей арифме- тической операции равен 0, иначе естественный порядок вычислений не нарушается
	9÷E		Прочие возможные команды
HLT	F		Останов вычислений

11.1.1. Устройство управления

Устройства управления (УУ) – это часть вычислительной машины, которая организует автоматическое выполнение программ и функциони- рование вычислительной машины как единой системы. В машине, изображенной на рис. 11.1, оно содержит следующие узлы.

Счетчик команд (СК) – неотъемлемый элемент устройства управления любой вычислительной машины, построенной в соответствии с фон-неймановским принципом программного управления. Согласно этому принципу соседние команды программы располагаются в ячейках памяти со следующими по порядку адресами и выполняются преимущественно в той же очередности, в какой они размещены в памяти вычислительной машины. Таким образом, адрес очередной команды может быть получен путем увеличения адреса ячейки, из которой была считана текущая команда, на длину выполняемой команды, представленную числом занимаемых ею ячеек. Реализацию такого режима и призван обеспечивать счетчик команд – двоичный счетчик, в котором хранится и модифицируется адрес очередной команды программы. Перед началом вычислений в счетчик команд заносится адрес ячейки основной памяти, где хранится команда, которая должна быть выполнена первой. В процессе выполнения каждой команды путем увеличения содержимого счетчика команд на длину выполняемой команды в счетчике формируется адрес следующей подлежащей выполнению команды. В рассматриваемой вычислительной машине любая команда занимает одну ячейку, поэтому содержимое счетчика увеличивается на единицу, что обеспечивается подачей сигнала управления +1СК. По завершении текущей команды адрес следующей команды программы всегда берется из счетчика команд. Для изменения естественного порядка вычислений (перехода в иную точку программы) достаточно занести в счетчик адрес точки перехода.

В ряде вычислительных машин счетчик команд реализуется в виде обычного регистра, а увеличение его содержимого производится внешней схемой (схемой инкремента/декремента).

Счетчик команд определяет лишь местоположение команды в памяти, но не держит информации о том, что это за команда. Чтобы приступить к выполнению команды, ее необходимо извлечь из памяти и разместить в *регистре команды* (РК). Этот этап носит название выборки команды. Только с момента загрузки команды в регистр команды становится «видимой» для процессора. В регистре команды команда хранится в течение всего времени ее выполнения. Любая команда

содержит два поля: поле кода операции и поле адресной части. Учитывая это обстоятельство, регистр команды иногда рассматривают как совокупность двух регистров – *регистра кода операции* (РКОп) и *регистра адреса* (РА), в которых хранятся соответствующие составляющие команды.

Если команда занимает несколько последовательных ячеек, то код операции всегда находится в том слове команды, которое извлекается из памяти первым. Это позволяет по коду операции определить, требуются ли считывание из памяти и загрузка в регистр команды остальных слов команды. Собственно выполнение команды начинается только после занесения в регистр команды ее полного кода.

Указатель стека (УС) – это регистр, где хранится адрес вершины стека. В реальных вычислительных машинах стек реализуется в виде участка основной памяти, обычно расположенного в области наибольших адресов. Заполнение стека происходит в сторону уменьшения адресов, при этом вершина стека – это ячейка, куда была произведена последняя по времени запись. Для хранения адреса такой ячейки и предназначен указатель стека. При выполнении операции **push** (занесение в стек) содержимое указателя стека с помощью сигнала **–1УС** сначала уменьшается на единицу, после чего используется в качестве адреса, по которому производится запись. Соответствующая ячейка становится новой вершиной стека. Считывание из стека (операция **pop**) происходит из ячейки, на которую указывает текущий адрес в указателе стека, после чего содержимое указателя стека сигналом +1УС увеличивается на единицу. Таким образом, вершина стека опускается, а считанное слово считается удаленным из стека. Хотя физически считанное слово и осталось в ячейке памяти, при следующей записи в стек оно будет заменено новой информацией.

Регистр адреса памяти (РАП) предназначен для хранения исполни-

тельного адреса ячейки основной памяти вплоть до завершения операции (считывание или запись) с этой ячейкой. Наличие регистра адреса памяти позволяет компенсировать различия в быстродействии основной памяти и прочих устройств машины.

Регистр данных памяти (РДП) призван компенсировать разницу

в быстродействии запоминающих устройств и устройств, выступающих в роли источников и потребителей хранимой информации. В регистр дан- ных памяти при чтении заносится содержимое ячейки основной памяти, а при записи – помещается информация, подлежащая сохранению в ячейке основной памяти. Собственно момент считывания и записи в ячейку опре- деляется сигналами **ЧтЗУ** и **ЗпЗУ** соответственно.

Дешифратор кода операции (ДКОП) преобразует код операции в форму, требуемую для работы микропрограммного автомата (МПА). Информация после декодирования определяет последующие действия микропрограммного автомата, ее вид зависит от организации микропрограммного автомата. В рассматриваемой вычислительной машине код операции преобразуется в унитарный код **УнитК**, в котором каждой команде (каждому коду операции) соответствует отдельный бит. Часто код операции преобразуется в адрес первой команды микро- программы, реализующей указанную в команде операцию. С этих позиций дешифратор кода операции правильнее было бы назвать не дешифратором, а преобразователем кодов.

Микропрограммный автомат (МПА) правомочно считать централь- ным узлом устройства управления. Именно микропрограммный автомат формирует последовательность сигналов управления, в соответствии с которыми производятся все действия, необходимые для выборки из памяти и выполнения команд. Исходной информацией для микропрог- раммного автомата служат: декодированный код операции, состояние признаков (флагов), характеризующих результат предшествующих вычислений, а также внешние запросы на прерывание текущей программы.

11.1.2. Арифметико-логическое устройство

Арифметико-логическое устройство (АЛУ) предназначено для арифметической и логической обработки данных. В машине, изображенной на рис. 11.1, оно содержит следующие узлы.

Операционный блок (ОпБ) представляет собой ту часть АЛУ, которая, собственно, и выполняет арифметические и логические операции над поданными на вход операндами. Выбор конкретной операции (из возможного списка операций для данного операционного блока) определяется кодом операции команды. В нашей вычислительной машины код операции поступает непосредственно из регистра команды. В реальных машинах код операции зачастую преобразуется в микропрограммном автомате в иную форму и уже из микропрог- раммного автомата поступает в АЛУ. Операционные блоки современных АЛУ строятся как комбинационные схемы, то есть они не обладают внутренней памятью и до момента сохранения результата операнды должны присутствовать на входе блока.

Регистры операндов **РХ**и **РУ**обеспечивают сохранение операндов

на входе операционного блока вплоть до получения результата операции и его записи в аккумулятор.

Регистр признаков (Рпрз) предназначен для фиксации и хранения признаков (флагов), характеризующих результат последней выполненной арифметической или логической операции. Такие признаки могут информировать о равенстве результата нулю, о знаке результата, о возникновении переноса из старшего разряда, переполнении разрядной сетки и т.д. Содержимое регистра признаков обычно используется устройством управления для реализации условных переходов по результатам операций АЛУ. Под каждый из возможных признаков отводится один разряд регистра признаков.

Формирование признаков осуществляется блоком формирования состояний регистра признаков, который может входить в состав операционного блока либо реализуется в виде внешней схемы, располагаемой между операционным блоком и регистром признаков.

Аккумулятор (Акк) – это регистр, на который возлагаются самые разнообразные функции. Так, в него предварительно загружается один из операндов, участвующих в арифметической или логической операции. В аккумуляторе может храниться результат предыдущей команды и в него же заносится результат очередной операции. Через аккумулятор зачастую производятся операции ввода и вывода.

В вычислительных машинах с регистровой архитектурой его можно рассматривать как один из регистров общего назначения.

11.1.3. Основная память

Вне зависимости от типа используемых микросхем *основная память* (ОП) представляет собой массив запоминающих элементов (ЗЭ), организован- ных в виде ячеек, способных хранить некую единицу информации, обычно один байт. Каждая ячейка имеет уникальный адрес. Ячейки основной па- мяти организованы в виде матрицы, а выбор ячейки осуществляется путем подачи разрешающих сигналов на соответствующие строку и столбец этой матрицы. Выбор обеспечивается дешифратором адреса памяти, преобра- зующим поступивший из регистра адреса памяти (РАП) адрес ячейки в разре- шающие сигналы, подаваемые в горизонтальную и вертикальную линии, на пересечении которых расположена адресуемая ячейка. Поскольку для совре- менных вычислительных машин характерна значительная емкость основной памяти приходится использовать несколько микросхем запоминающих устройств (ЗУ). В этих условиях процесс обращения к ячейке состоит из выбора нужной микросхемы (на основании старших разрядов адреса) и выбора ячейки внутри микросхемы (определяется младшими разрядами адреса). Первая часть процедуры производится внешними схемами, а вто- рая – внутри микросхем запоминающего устройства.

11.1.4. Модуль ввода/вывода

Задачей *модуля ввода/вывода* (МВВ) является обеспечение подключения к вычислительной машине различных периферийных устройств (ПУ) и обмена информацией с ними. В рассматриваемом варианте модуль ввода/вывода состоит из дешифратора номера порта ввода/вывода, множества портов ввода и множества портов вывода.

Порты ввода и порты вывода. Портом называют схему, ответст- венную за передачу информации из периферийного устройства ввода в аккумулятор АЛУ (порт ввода) или из аккумулятора на периферийное устройство вывода (порт вывода). Схема обеспечивает электрическое и логическое сопряжение вычислительной машины с подключенным к нему периферийным устройством.

Дешифратор номера порта ввода/вывода. В модуле ввода/вывода рассматриваемой вычислительной машины предполагается, что каждое периферийное устройство подключается к своему порту. Каждый порт имеет уникальный номер, который указывается в адресной части команд ввода/вывода. Дешифратор номера порта ввода/ вывода (ДВВ) обеспечивает преобразование номера порта в сигнал, разрешающий операцию ввода или вывода на соответствующем порте. Непосредственно ввод (вывод) происходит при поступлении из микропрограммного автомата (МПА) сигнала **Вв** или **Выв**.

11.2. Микрооперации и микропрограммы

Для пояснения логики функционирования ВМ ее целесообразно представить в виде совокупности узлов, связанных между собой коммуникационной сетью (рис. 11.2).

Функционирование вычислительной машины можно описать как последовательность пересылок информация между ее узлами и элементарных действий, выполняемых в узлах. Понятие узла трактуется весьма широко: от регистра до АЛУ или основной памяти. Также широко следует понимать и термин «элементарное действие». Это может быть установка регистра в некоторое состояние или выполнение операции в АЛУ. Любое элементарное действие производится при поступлении из микропрограммного автомата УУ соответствующего сигнала управления (СУ). Возможная частота формирования сигналов на выходе автомата определяется синхронизирующими импульсами (ТИ), поступающими от генератора тактовых импульсов (ГТИ). Элементарные пересылки или преобразования информации, выполняемые в течение одного такта сигналов синхронизации, называются *микрооперациями*.

В течение одного такта могут одновременно выполняться несколько микроопераций. Совокупность сигналов управления, порождающих микрооперации, выполняемые и одном такте, *называют микрокомандой*. Относительно сложные действия, осуществляемые вычислительной машиной в процессе ее работы, реализуются как последовательность микроопераций и могут быть заданы последовательностью микрокоманд, называемой *микропрограммой*. Реализует микропрограмму (вырабатывает управляющие сигналы, задаваемые ее микрокомандами) *микропрограммный автомат* (МПА).

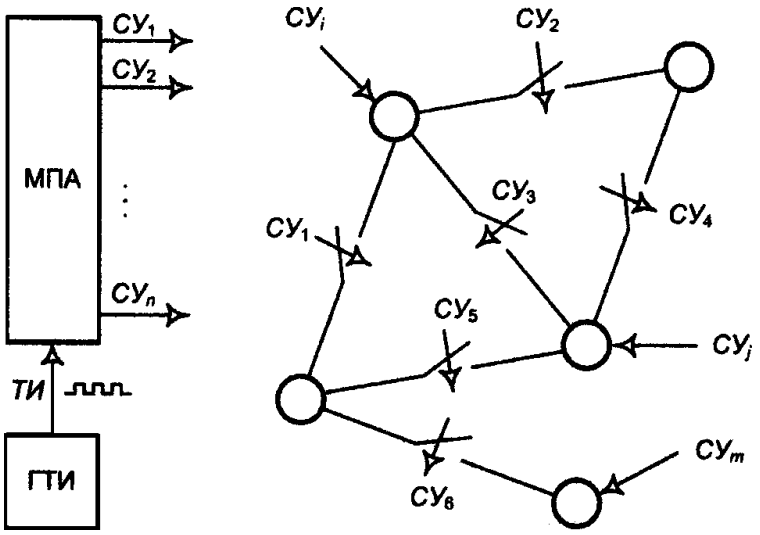


Рис. 11.2. Вычислительная машина с позиций микроопераций
и сигналов управления

11.2.1. Способы записи микропрограмм

Для записи микропрограмм в компактной форме используются граф-схемы алгоритмов и языки микропрограммирования.

Граф-схемы алгоритмов. Граф-схема алгоритма (ГСА) имеет вид ориентированного графа. При построении графа оперируют пятью типами вершин (рис. 11.3).

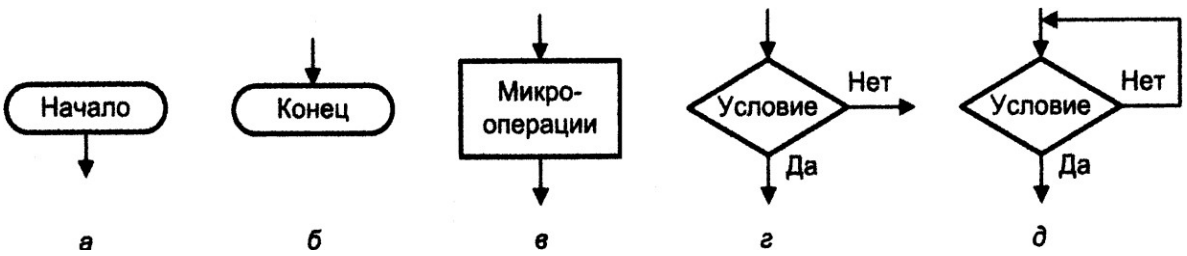


Рис. 11.3. Разновидности вершин граф-схемы алгоритма:

а – начальная; *б* – конечная; *в* – операторная; *г* – условная; *д* – ждущая

Начальная вершина (рис. 11.3, *а*) определяет начало микро- программы и не имеет входов. Конечная вершина (рис. 11.3, *б*) указывает конец микропрограммы, поэтому имеет только вход. В операторную вершину (рис. 11.3, *в*) вписывают микрооперации, выполняемые в течение одного машинного такта. С вершиной связаны один вход и один выход. Условная вершина (рис. 11.3, *г*) используется для ветвления вычисли- тельного процесса. Она имеет один вход и два выхода, соответствующие положительному («Да») и отрицательному («Нет») исходам проверки условия, записанного в вершине. С помощью ждущей вершины (рис. 11.3, *д*) можно описывать ожидание в работе устройств. В этом случае выход «Да» соответствует снятию причины, вызвавшей ожидание. Граф-схемы алгоритмов составляются в соответствии со следующими правилами.

1. ГСА должна содержать одну начальную, одну конечную и конечное множество операторных и условных вершин.

2. Каждый выход вершины ГСА соединяется только с одним входом.

3. Входы и выходы различных вершин соединяются дугами, направленными от выхода к входу.

4. Для любой вершины ГСА существует, по крайней мере, один путь из этой вершины к конечной вершине, проходящий через операторные и условные вершины в направлении соединяющих их дуг.

5. В каждой операторной вершине записываются микрооперации y, соответствующие одной микрокоманде Y. Сами микрокоманды указываются возле операторных вершин.

6. В каждой условной вершине записывается один из элементов множества логических условий x.

7. Начальной вершине ставится в соответствие фиктивный оператор y_0, а конечной – фиктивный оператор y_k.

На рис. 11.4 показан пример микропрограммы, записанной на языке ГСА.

Языки микропрограммирования (ЯМП) обеспечивают описание функционирования ВМ в терминах микроопераций/

Если средства языка ориентированы на запись микропрограммы без привязки к конкретным средствам для их реализации, то такой ЯМП называют *языком функционального микропрограммирования*, а соответст- вующие микропрограммы – *функциональными микропрограммами*. Подобные микропрограммы служат исходной формой для описания функционирования ВМ.

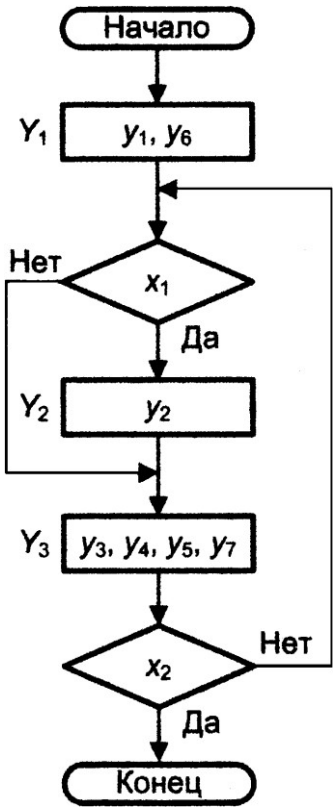


Рис. 11.4. Пример граф-схемы микропрограммы

В примере микрокоманда Y_1 инициирует микрооперации y_1 и y_6, микрокоманда Y_2 – микрооперацию y_2, а Y_3 – микрооперации y_3, y_4, y_5 и y_7.

В случае, когда средства языка нацелены на описание микро- программ, привязанных к конкретной реализующей их структуре, ЯМП называют *языком структурно-функционального микропрограммирования*.

11.2.2. Совместимость микроопераций

Совместимостью называется свойство совокупности микроопераций, гарантирующее возможность их параллельного выполнения. Различают функциональную и структурную совместимости. Пусть S_1, S_2, S_3, S_4, – подмножества слов из множества S. Тогда микрооперации $S_1 \square \square \square S_2 \square$ и $S_3 \square \square \square S_4 \square$ называются *функционально совместимыми*, если $S_1 \square S_2 \square \square$, то есть если микрооперации присваивают значения разным словам. В функциональных микропрограммах, описывающих алгоритмы выполнения операций

без учета структуры вычислительной машины, одновременно могут выполняться только функционально совместимые микрооперации.

Структура ВМ может внести дополнительные ограничения на возможность параллельного выполнения микроопераций. Микрооперации называются *структурно несовместимыми*, если из-за ограничений, обусловленных структурой ВМ, они не могут выполняться параллельно. Обычно структурная несовместимость связана с использованием микрооперациями одного и того же оборудования.

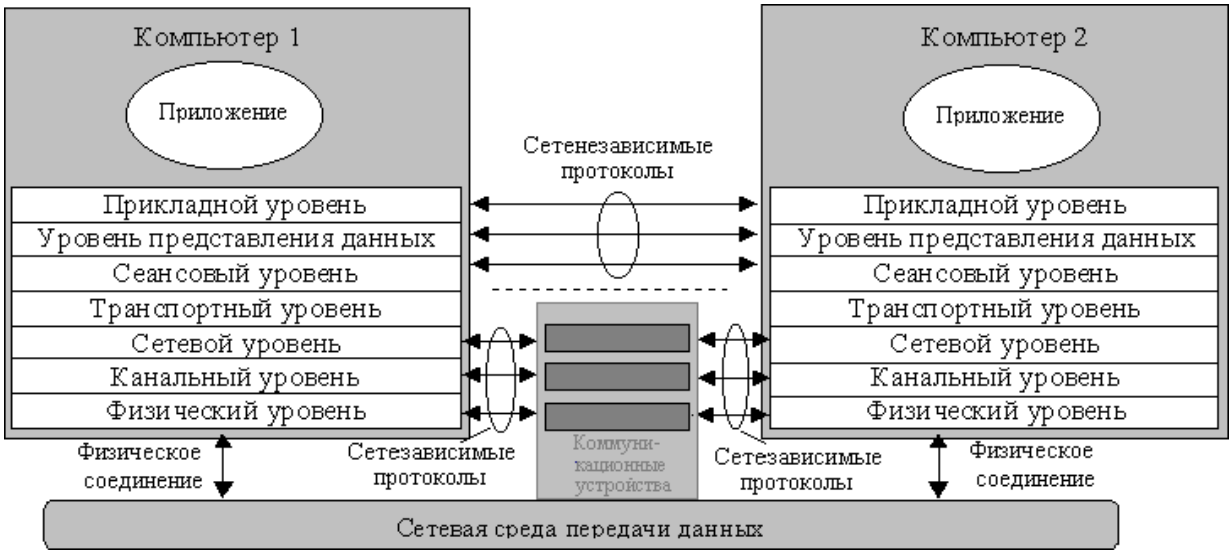
Контрольные вопросы

1. Какую функцию выполняет счетчик команд и какой должна быть его разрядность?
2. Какое из полей регистра команд должно быть заполнено в первую очередь?
3. Какой адрес должен быть занесен в указатель стека при его инициализации?
4. Какими средствами компенсируется различие в быстродействии процессора и основной памяти?
5. На основании какой информации микропрограммный автомат формирует сигналы управления?
6. Можно ли считать наличие регистров операндов обязательным условием работы любого операционного блока?
7. Каким образом используется информация, хранящаяся в регистре признаков?
8. С каким понятием можно ассоциировать сигнал управления?
9. В чем состоит различие между микрокомандой и микрооперацией?
10. Какие существуют способы записи микропрограмм?
11. Перечислите основные правила составления граф-схемы алгоритма.
12. Что подразумевает понятие «совместимость микроопераций»?

Информация по теме:

Все задачи, которые необходимо решить для организации взаимодействия между объектами информационной системы, разделены на семь отдельных процедур или уровней, представленных на рисунке. Каждый уровень выполняет определенную логическую функцию и обеспечивает определенный набор услуг для расположенного над ним уровня.

Отдельные уровни модели OSI удобно рас- сматривать как группы программ, предназначенных для выполнения конкретных функций. При обращении некоторого сетевого приложе- ния к прикладному уровню на основании запроса программное обеспечение формирует сообщение стандартного формата, состоящее из заголовка и поля данных. После формирования сообщения при- кладной уровень направляет его вниз. Протокол каждого уровня на основании информации, полученной из заголовка



вышележащего уровня, выполняет требуемые действия и добавляет к сообщению собственную служебную информацию. Когда сообщение по сети поступает на станцию-адресат, оно принимается физическим уровнем и перемещается вверх. Последовательно каждый уровень анализирует и обрабатывает заголовок своего уровня, выполняя соответствующие данному уровню функции, а затем его удаляет.

Три нижних уровня – физический, канальный, сетевой - являются сетезависимыми, то есть протоколы этих уровней тесно связаны с технической реализацией сети и используемым коммуникационным оборудованием.

Три верхних уровня – прикладной, представительский и сеансовый – ориентированы на приложения и мало зависят от технических особенностей построения сети. На протоколы этих уровней не влияют изменения в топологии сети, замена оборудования или переход на другую сетевую технологию.

Транспортный уровень является промежуточным, он скрывает все детали функционирования нижних уровней от верхних, что позволяет разрабатывать приложения, независящие от технических средств непосредственной транспортировки сообщений.

В стандартах ISO для обозначения единиц данных, с которыми имеют протоколы разных уровней, используется общее название протокольный блок данных (*Protocol Data Unit, PDU*). Для обозначения блоков данных определенных уровней используются специальные названия: кадр (frame), пакет (packet), дейтаграмма (datagram), сегмент (segment).

Домашнее задание:

Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 с

Раздел 4. Организация шин вычислительной машины

1. Типы шин.
2. Иерархия шин.
3. Контрольные вопросы.
4. Методы повышения эффективности шин.
5. Контрольные вопросы.
6. Шины <<большого >> интерфейса.
7. Шины << малого>> интерфейса.
8. Контрольные вопросы.

Занятие 5(лекция)

Совокупность трактов, объединяющих между собой основные устройства ВМ и организующих связь ВМ с внешним миром, образует *систему шин* вычислительной машины. Система шин обеспечивает обмен информацией между:

– центральным процессором и памятью;

– центральным процессором и модулями ввода/вывода;

– памятью и модулями ввода/вывода.

Большинство машин содержат несколько различных шин, каждая из которых оптимизирована под определенный вид коммуникаций. Часть шин скрыта внутри интегральных микросхем или доступна только в пределах печатной платы. Некоторые шины имеют доступные извне точки для того, чтобы к ним легко можно было подключить дополнительные устройства, причем большинство таких шин не просто доступны, но и отвечают определенным стандартам, что позволяет подсоединять к шине устройства различных производителей.

Шина характеризуется:

– совокупностью сигнальных линий;

– физическими, механическими и электрическими характеристиками;

– используемыми сигналами арбитража, состояния, управления и синхронизации;

– правилами взаимодействия подключенных к шине устройств (протоколом шины).

Шину образует набор линий связи. Линия связи – это физическая среда, обеспечивающая передачу сигналов, чаще всего представляющих двоичные цифры 1 и 0. По линии может пересылаться развернутая во времени последовательность таких сигналов. При совместном использовании несколько линий могут обеспечить одновременную (параллельную) передачу двоичных чисел. Физически линиями связи могут быть провода, полоски проводящего материала на монтажной плате, оптоволокно, радио и инфракрасные каналы.

Операции на шине называют *транзакциями.* Основные виды транзакций – *транзакции чтения* и *транзакции записи.* Если в обмене участвует устройство ввода/вывода, можно говорить о *транзакциях ввода* и *вывода,* по сути эквивалентных транзакциям чтения и записи соответственно. Шинная транзакция включает в себя две составляющих: посылку адреса и прием (или посылку) данных.

Когда два устройства обмениваются информацией по шине, одно из них должно инициировать обмен и управлять им. Такого рода устройства называются *ведущими* (bus master). Ведущий не обязательно использует данные сам. Он, например, может захватить управление шиной в интересах другого устройства. Устройства, не обладающие возможностями инициирования транзакции, называются *ведомыми* (bus slave). К шине может быть подключено несколько потенциальных ведущих, но в любой момент времени активным может быть только один из них: если несколько устройств передают информацию одновременно, их сигналы перекрываются и искажаются. Для предотвращения одновременной активности нескольких ведущих в любой шине предусматривается процедура допуска к управлению шиной только одного из претендентов (*арбитраж*). В то же время некоторые шины допускают широковещательный режим записи, когда информация одного ведущего передается сразу нескольким ведомым (здесь арбитраж не требуется). Сигнал, направленный одним устройством, доступен всем остальным устройствам, подключенным к шине.

13.1. Типы шин

Важным параметром, определяющим характеристики шины, является ее целевое назначение. По этому параметру можно выделить:

– шины «процессор-память»;

– шины ввода/вывода;

– системные шины.

13.1.1. Шины «процессор-память»

Шины «процессор-память» обеспечивают непосредственную связь центрального процессора (ЦП) вычислительной машины с основной памятью (ОП). Часто эта задача возлагается на системную шину, однако в плане эффективности значительно выгоднее, если обмен между ЦП и ОП ведется по независимой шине. В современных микропроцессорах шину, связывающую ЦП и ОП, называют *передней* или *первичной шиной* и обозначают аббревиатурой FSB (Front-Side Bus). Интенсивный обмен информацией между процессором и основной памятью требует, чтобы полоса пропускания шины, то есть количество информации, проходящей по шине в единицу времени, была наибольшей. К группе шин «процессор- память» можно отнести также шину, связывающую процессор с кэш-памятью второго уровня (L2), известную как *тыльная* или

вторичная шина – BSB (Back-Side Bus). BSB позволяет вести обмен с большей скоростью, чем FSB, что отвечает возможностям более скоростной кэш-памяти. Архитектура с распределением функций связи ЦП с памятью между шинами FSB и BSB известна как архитектура DIB (Dual Independent Bus – двойная независимая шина). Наличие двух шин позволяет одновременно обращаться к ОП и L2, тем самым увеличивая общую производительность ВМ. Перечисленные варианты связи процессора с памятью иллюстрирует рис. 13.1.

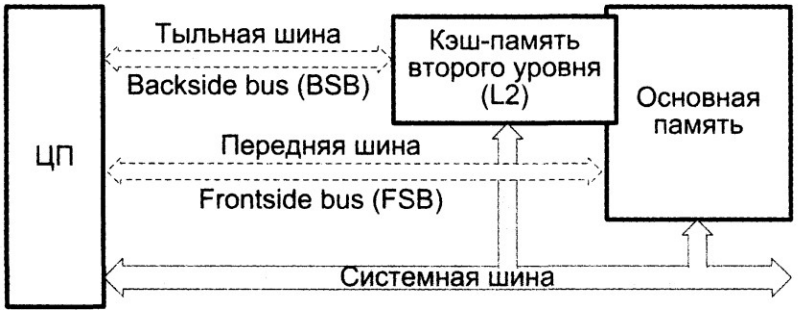


Рис. 13.1. Шины «процессор-память»

Для обеспечения максимальной пропускной способности шина «процессор-память» всегда проектируются с учетом особенностей организации системы памяти, а длина шины делается по возможности минимальной.

13.1.2. Шина ввода/вывода

Шина ввода/вывода служит для соединения процессора (памяти) с устройствами ввода/вывода (УВВ). Учитывая разнообразие таких устройств, шины ввода/вывода унифицируются и стандартизируются. Связи с большинством УВВ (но не с видеосистемами) не требуют от шины высокой пропускной способности. При проектировании шин ввода/вывода в учет берутся стоимость конструктива и соединительных разъемов. Такие шины содержат меньше линий по сравнению с вариантом «процессор-память», но длина линий может быть весьма большой.

13.1.3. Системная шина

С целью снижения стоимости некоторые ВМ имеют общую шину для памяти и устройств ввода/вывода. Такая шина часто называется системной. *Системная шина* служит для физического и логического

объединения всех устройств ВМ. Поскольку основные устройства машины, как правило, размещаются на общей монтажной плате, системную шину часто называют объединительной шиной (backplane bus).

Системная шина обычно состоит из нескольких сотен линий, которые можно подразделить на три функциональных группы (рис. 13.2): шину данных, шину адреса и шину управления. К последней обычно относят также линии для подачи питающего напряжения на подключаемые к системной шине устройства.

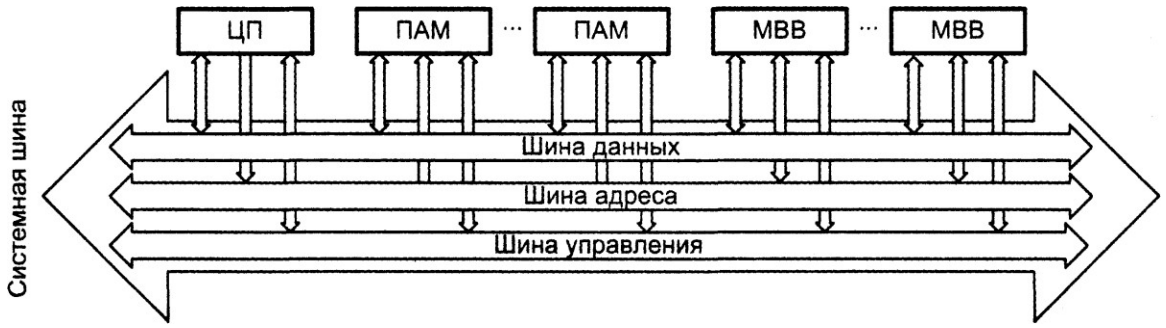


Рис. 13.2. Структура системной шины:

ЦП – центральный процессор; ПАМ – память; МВВ – модуль ввода/вывода

Любая транзакция на системной шине начинается с выставления ведущим устройством адресной информации. Адрес позволяет выбрать ведомое устройство и установить соединение между ним и ведущим. Для передачи адреса используется часть сигнальных линий шины, совокупность которых часто *называют шиной адреса* (ША).

На ША могут выдаваться адреса ячеек памяти, номера регистров ЦП, адреса портов ввода/вывода и т.п. Для идентификации того к какому устройству идет обращение (ОП, регистр, порт ввода/вывода), используются специальные управляющие линии. Однако эта же информация может косвенно содержаться в самом адресе.

Разнообразной может быть и структура адреса. Так, в адресе старшие биты могут указывать на один из модулей основной памяти, в то время как младшие биты определяют ячейку внутри этого модуля.

В некоторых шинах предусмотрены адреса специального вида, обеспечивающие одновременный выбор определенной группы ведомых либо всех ведомых сразу (*broadcast*). Эта возможность обычно практикуется в транзакциях записи (от ведущего к ведомым), однако существует также специальный вид транзакции чтения (одновременно от нескольких ведомых общему ведущему). Английское название такой транзакции чтения *broadcall* можно перевести как *«широковещательный опрос».*

Информация, возвращаемая ведущему, представляет собой результат побитового логического сложения данных, поступивших от всех адресуемых ведомых.

Число сигнальных линий, выделенных для передачи адреса (*ширина шины адреса*), определяет максимально возможный размер адресного пространства. Это одна из базовых характеристик шины, поскольку от нее зависит потенциальная емкость адресуемой памяти и число обслуживаемых портов ввода/вывода. В современных микропроцессорах шина адреса состоит из 36 сигнальных линий, при этом адресное пространство составляет 64 Гбайт. В перспективных разработках предусматривается расширение ША до 40 линий.

Совокупность линий, служащих для пересылки данных между модулями системы, называют *шиной данных* (ШД). Важнейшие характеристики ШД – ширина и пропускная способность.

Ширина шины данных определяется количеством битов информации, которое может быть передано по шине за одну транзакцию (*цикл шины*). Цикл шины следует отличать от периода тактовых импульсов – одна транзакция на шине может занимать несколько тактовых периодов. Ширина шины данных оставляет 32, 64 или 128 битов. В любом случае ширину ШД выбирают кратной целому числу байтов, причем это число, как правило, представляет собой целую степень числа 2.

Элемент данных, задействующий всю ширину ШД, принято называть *словом*, хотя в архитектуре некоторых ВМ понятие «слово» трактуется по-другому, то есть слово может иметь разрядность, не совпадающую с шириной ШД.

В большинстве шин используются адреса, позволяющие указать отдельный байт слова. Это свойство оказывается полезным, когда желательно изменить в памяти лишь часть полного слова.

При передаче по ШД части слова пересылка обычно производится по тем же сигнальным линиям, что и в случае пересылки полного слова, однако в ряде шин «урезанное» слово передается по младшим линиям ШД. Последний вариант может оказаться более удобным при последующем расширении шины данных, поскольку в этом случае сохраняется преемственность со «старой» шиной.

Ширина шины данных существенно влияет на производительность ВМ. Так, если ширина шины данных вдвое меньше длины команды, ЦП в течение каждого цикла команды вынужден осуществлять доступ к памяти дважды.

Если адрес и данные в системной шине передаются по независимым (выделенным) сигнальным линиям, то ширина ША и ШД обычно выбирается независимо. Наиболее частые комбинации: 16–8,16–16, 20–8, 20–16, 24–32, 32–32 и 36–64.

В некоторых ВМ адрес и данные пересылаются по одним и тем же линиям, но в разных тактах цикла шины. Этот прием называется *временным мультиплексированием*. В этом случае линии адреса и данных объединены в единую *мультиплексируемую шину адреса/данных*. Такая шина функционирует в режиме разделения времени, поскольку цикл шины разбит на временной интервал для передачи адреса и временной интервал для передачи данных. Структура мультиплексируемой шины адреса/данных показана на рис. 13.3.

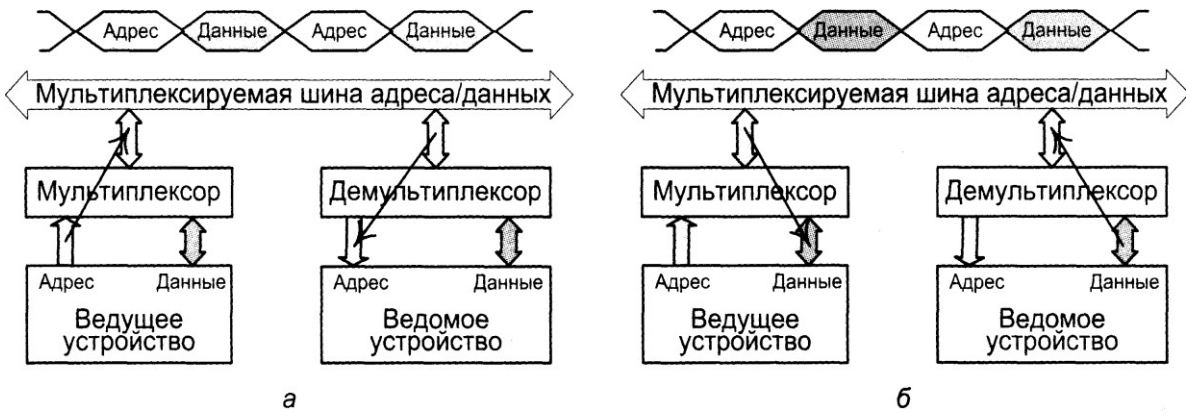


Рис. 13.3. Мультиплексирование адреса и данных:

а – передача адреса; *б* – передача данных

Мультиплексирование адресов и данных предполагает наличие мультиплексора на одном конце тракта пересылки информации и демультиплексора на его другом конце. Мультиплексоры и демультиплексоры играют роль коммутирующих устройств. Они обеспечивают перенаправление информации с одного из своих входов на общий выход (мультиплексор) либо с общего входа на один из выходов (демультиплексор).

Мультиплексирование позволяет сократить общее число линий, но требует усложнения логики связи с шиной. Кроме того, оно ведет к потенциальному снижению производительности, поскольку исключает возможность параллельной передачи адресов и данных, что можно было бы использовать в транзакциях записи, одновременно выставляя на ША адрес, а на ШД – записываемое слово.

Совокупность линий, по которым передается управляющая информация, а также информация о состоянии участвующих в транзакции устройств, называется *шиной управления* (ШУ). Сигнальные линии, входящие в ШУ, можно условно разделить на несколько групп.

Первую группу образуют линии, по которым пересылаются *сигналы управления транзакциями*, то есть сигналы, определяющие:

– тип выполняемой транзакции (чтение или запись);

– количество байтов, передаваемых по шине данных, и если пересылается часть слова, то номера байтов;

– какой тип адреса выдан на шину адреса;

– какой протокол передачи должен быть применен.

На перечисленные цели обычно отводится от двух до восьми сигнальных линий.

Ко второй группе относят линии передачи *информации состояния* (*статуса*). В эту группу входят от одной до четырёх линий, по которым ведомое устройство может информировать ведущего о своем состоянии или передать код возникшей ошибки.

Третья группа – *линии арбитража*. Арбитраж необходим для выбора одного из нескольких ведущих, одновременно претендующих на доступ к шине. Число линий арбитража в разных шинах варьируется от 3 до 11.

Четвертую группу образуют *линии прерывания*. По этим линиям передаются запросы на обслуживание, посылаемые от ведомых устройств к ведущему. Под собственно запросы обычно отводятся одна или две линии, однако при одновременном возникновении запросов от нескольких ведомых возникает проблема арбитража, для чего могут понадобиться дополнительные линии, если только с этой целью не используются линии третьей группы.

Пятая группа – линии для организации *последовательных локальных сетей.* Наличие от 1 до 4 таких линий стало общепринятой практикой в современных шинах. Обусловлено это тем, что последовательная передача данных протекает значительно медленнее, чем параллельная, и сети значительно выгоднее строить без загрузки быстрых линий основных шин адреса и данных. Кроме того, шины этой группы могут быть использованы как полноценный, хотя и медленный, избыточный тракт для замены ША и ШД в случае их отказа. Иногда шины пятой группы назначаются для реализации специальных функций, таких, например, как обработка прерываний или сортировка приоритетов задач.

В некоторых ШУ имеется **шестая группа** сигнальных линий – от 4 до 5 *линий позиционного кода*, подсоединяемых к специальным выводам разъема. С помощью перемычек на этих выводах можно задать уникальный позиционный код разъема на материнской плате или вставленной в этот разъем дочерней платы. Такой код может быть использован для индивидуальной инициализации каждой отдельной платы при включении или перезапуске системы.

Седьмая группа – это группа линий *тактирования и синхронизации*. При проектировании шины этим линиям уделяется особое внимание. В состав группы, в зависимости от протокола шины (синхронный или асинхронный), входят от двух до шести линий.

Восьмая группа – это линии для подвода питающего напряжения и линии заземления.

Функционирование системной шины можно описать следующим образом. Если один из модулей хочет передать данные в другой, он должен выполнить два действия: получить в свое распоряжение шину и передать по ней данные. Если какой-то модуль хочет получить данные от другого модуля, он должен получить доступ к шине и с помощью соответст- вующих линий управления и адреса передать в другой модуль запрос. Далее он должен ожидать, пока модуль, получивший запрос, пошлет данные.

Физически системная шина представляет собой совокупность параллельных электрических проводников. Этими проводниками служат металлические полоски на печатной плате. Шина подводится ко всем модулям, и каждый из них подсоединяется ко всем или некоторым ее линиям. Если вычислительная машина конструктивно выполнена на нескольких платах, то все линии шины выводятся на разъемы, которые затем объединяются проводниками на общем шасси.

13.2. Иерархия шин

Если к шине подключено большое число устройств, ее пропускная способность падает, поскольку слишком частая передача прав управления шиной от одного устройства к другому приводит к ощутимым задержкам. По этой причине во многих ВМ предпочтение отдается использованию нескольких шин, образующих определенную иерархию.

13.2.1. Вычислительная машина с одной шиной

В системах соединений с одной шиной имеется одна системная шина, обеспечивающая обмен информацией между всеми устройствами ВМ (рис. 13.4, *а*).

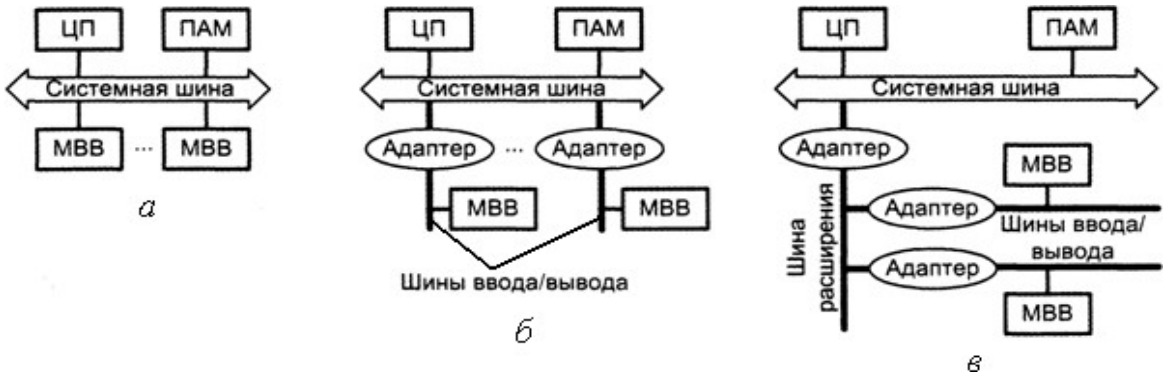


Рис. 13.4. Система соединений:

а – с одной шиной; *б* – с двумя видами шин; *в* – с тремя видами шин

Для такого подхода характерны простота и низкая стоимость. Однако одношинная организация не в состоянии обеспечить высокие интенсивность и скорость транзакций, в силу чего становится

«узким местом» вычислительной машины.

13.2.2. Вычислительная машина с двумя видами шин

Хотя контроллеры модулей ввода/вывода (МВВ) могут быть подсоединены непосредственно к системной шине, больший эффект достигается применением одной или нескольких шин ввода/вывода (рис. 7.6, *б*). МВВ подключаются к шинам ввода/вывода, которые берут на себя основной трафик, не связанный с выходом на процессор или память. *Адаптеры шин* обеспечивают буферизацию данных при их пересылке между системной шиной и контроллерами МВВ. Это позволяет ВМ поддерживать работу множества модулей ввода/вывода и одновре- менно «развязать» обмен информацией по тракту процессор-память и обмен информацией с МВВ.

Подобная схема существенно снижает нагрузку на скоростную шину

«процессор-память» и способствует повышению общей производитель- ности вычислительной машины.

13.2.3. Вычислительная машина с тремя видами шин

Для подключения быстродействующих периферийных устройств в систему шин может быть добавлена высокоскоростная шина расширения (рис. 13.4, *в*).

Шины ввода/вывода подключаются к шине расширения, а уже с нее через адаптер к системной шине, что еще более снижает нагрузку на последнюю. Такую организацию шин называют *архитектурой с «пристройкой»* (mezzanine architecture).

Контрольные вопросы

1. Перечислите основные виды структур взаимосвязей вычисли- тельной машины.
2. Какие параметры включает в себя полная характеристика шины?
3. Что такое транзакция, из каких этапов она состоит?
4. В чем заключается основное различие между ведущими и ведомыми устройствами?
5. Что такое широковещательный режим записи?
6. Какие шины в составе ВМ образуют иерархию шин?
7. Что такое «широковещательный опрос» и как он реализуется?
8. Какая характеристика вычислительной машины зависит от ширины адресной шины?
9. В чем заключаются основные преимущества и недостатки мультиплексирования адреса и данных?
10. Какие группы сигнальных линий образуют шину управления?

Информация по теме:

Наиболее популярными являются стеки: TCP/IP, IPX/SPX, NetBIOS/SMB, DECnet, SNA, OSI. Все стеки, кроме SNA, на фи- зическом и канальном уровнях используют одни и те же хорошо стандартизированные протоколы Ethernet, Token Ring, FDDI и позволяют использовать во всех сетях одну и ту же аппаратуру. На верхних уровнях все стеки работают по своим собственным протоко- лам, которые часто не соответствуют рекомендуемому моделью OSI разбиению на уров- ни. В таблице показано соответствие популярных стеков протоколов модели OSI.

модель OSI	IBM/ Microsoft	TCP/IP	Стек OSI
Прикладной	SMB	telnet FTP SMPT	X.400 X.500 FTAM
Представительский		SNPT WWW	Представительный протокол модели OSI
Сеансовый	NetBIOS	TCP	Сеансовый протокол модели OSI
Транспортный			Транспортный протокол модели OSI
Сетевой		IP RIP OSPF	ES-ES IS-IS
Канальный	Ethernet, Token Ring, FDDI, Fast Ethernet, SLIP, X.25, ATM, PPP, 100VG-AnyLAN		
Физический	Коаксиальный кабель, экранированная и неэкранированная витая пара, волоконно-оптический кабель, радиоволны		

Стек TCP/IP – один из самых распространенных стеков транспортных протоколов вычислительных сетей. Основными протоколами стека являются, давшие ему название, протоколы IP и TCP. Они относятся соответственно к сетевому и транспортному уровням. IP обеспечивает продвижение пакета по составной сети, TCP гарантирует надежность его доставки. TCP/IP вобрал в себя популярные протоколы прикладного уровня – протокол пересылки файлов FTP, протокол эмуляции терминала telnet, почтовый протокол SMPT. Особенностью технологии TCP/IP является гибкая система адресации, позволяющая дос- таточно просто включать в интерсеть сети других технологий. Однако мощные функцио- нальные возможности стека TCP/IP требуют для своей реализации высоких вычислитель- ных затрат и предъявляют высокие требования к администрированию IP-сетей.

Ссылки на литературные источники, приведенные в рабочей программе дисциплины: пп. 1, 2 основной литературы, интернет-источники, периодические издания.

Домашнее задание:
Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 c

Раздел 6. Принципы построения арифметико-логических устройств

1. Структуры операционных систем.
2. Вспомогательные системы счисления, используемые в операционных устройствах.
3. Контрольные вопросы.

Занятие 6 (лекция)

В классической фон-неймановской вычислительной машине арифметическая и логическая обработка данных возлагается на арифметико- логическое устройство (АЛУ). В реальных вычислительных машинах АЛУ обычно реализуется не как единое устройство, а в виде комплекса операционных устройств (ОПУ), каждое из которых ориентировано на определенные операции и определенные типы данных. В общем случае можно выделить:

– ОПУ для арифметической обработки чисел в форме с фиксиро- ванной запятой;

– ОПУ для арифметической обработки чисел в форме с плавающей запятой;

– ОПУ для логической обработки данных;

– ОПУ десятичной арифметики.

Специализированные операционные устройства для логической обработки данных и десятичной арифметики в современных вычисли- тельных машинах встречаются достаточно редко, поскольку подобную обработку можно достаточно эффективно организовать на базе ОПУ для чисел с фиксированной запятой. Таким образом, АЛУ образуют два вида операционных устройств: для обработки чисел в форме с фиксированной запятой (ФЗ) и обработки чисел в форме с плавающей запятой (ПЗ). В свою очередь, эти ОПУ также могут представлять собой совокупность операционных устройств, специализированных под определенную арифметическую операцию, например ОПУ сложения и вычитания, ОПУ умножения, ОПУ деления и т.п. В минимальном варианте АЛУ должно содержать аппаратуру для реализации лишь

основных логических операций, операций сдвигов, а также сложения и вычитания чисел в форме с фиксированной запятой. Опираясь на этот набор, можно программным способом обеспечить выполнение остальных арифметических и логических операций как для чисел с фиксированной запятой, так и для других форм представления информации. Подобный вариант, однако, не позволяет добиться высокой скорости вычислений, поэтому по мере расширения технологических возможностей доля аппаратных средств в АЛУ постоянно возрастает.

На рис. 17.1 показана динамика изменения соотношения между аппаратной и программной реализацией функций АЛУ по мере развития элементной базы вычислительной техники.

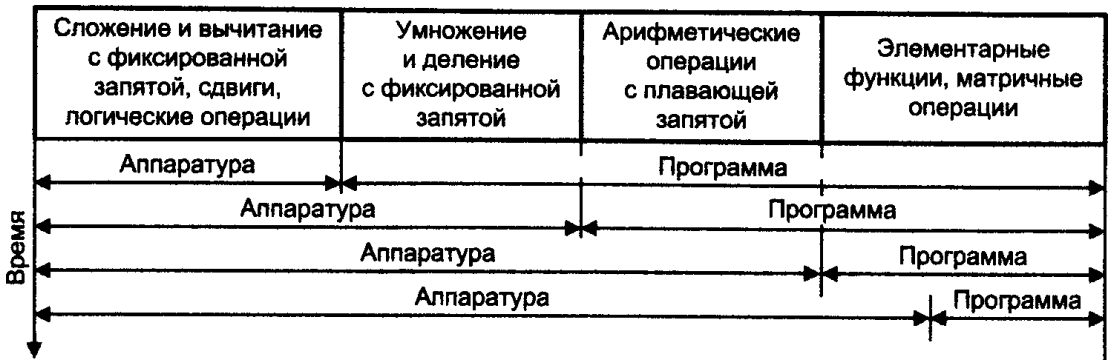


Рис. 17.1. Динамика изменения соотношения между аппаратной и программной реализациями функций АЛУ

17.1. Структуры операционных устройств

Набор элементов, на основе которых строятся различные операционные устройства, называется *структурным базисом*. Структурный базис ОПУ включает в себя:

– регистры, обеспечивающие кратковременное хранение слов данных;
– управляемые шины, предназначенные для передачи слов данных;
– комбинационные схемы, реализующие вычисление логических условий и выполнение микроопераций по сигналам от устройства управления.

17.1.1. Операционные устройства с жесткой структурой

В операционных устройствах с жесткой структурой комбинационные схемы жестко распределены между всеми регистрами. Каждому регистру придается свой набор комбинационных схем, позволяющих реализовать определенные микрооперации. Пример операционного устройства с жесткой структурой, обеспечивающего выполнение операций типа

«сложение», приведен на рис. 17.2. В состав ОПУ входят три регистра со своими логическими схемами:

– регистр первого слагаемого **РСл1**и схема **ЛРСл1**;
– регистр второго слагаемого **РСл2**и схема **ЛРСл2**;
– регистр суммы **РСм**и схема комбинационного сумматора **См**.

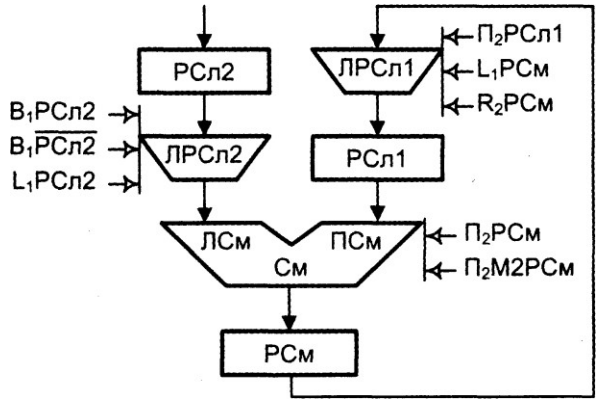


Рис. 17.2. Операционное устройство с жесткой логикой

Логическая схема **ЛРСл1** обеспечивает передачу результата из регистра **РСм**в регистр **РСл1**:

– прямым кодом **РСл1:= РСм**(по сигналу управления **П2РСл1**);

– со сдвигом на один разряд влево **РСл1:= L1** (**РСм • 0**) (по сигналу управления **L1РСм**);

– со сдвигом на два разряда вправо **РСл 1:= R2**(**s • s • РСм**)
(по сигналу управления **R1РСм**).

Логическая схема второго слагаемого **ЛРСл2** реализует микрооперации передачи второго слагаемого из **РСл2** на левый вход сумматора:

– прямым кодом **ЛСм:= РСл2**(по сигналу управления **B1РСл2**);

– инверсным кодом $ЛСм := \overline{РСл2}$ (по сигналу управления $\overline{B_1РСл2}$);

– со сдвигом на один разряд влево **ЛСм:= L1**(**РСл2 • 0**)
(по сигналу управления **L1РСл2**).

Комбинационный сумматор **С**м предназначен для суммирования (обычного или по модулю 2) операндов, поступивших на его левый (**ЛСм**) и правый (**ПСм**) входы. Результат суммирования заносится в регистр **РСм**: **РСм:= ЛСм + ПСм** (по сигналу управления **П2РСм**) или **РСм:= ЛСм □ ПСм** (по сигналу управления **П2М2РСм**).

Аппаратные затраты на ОПУ с жесткой структурой $С_Ж$ можно оценить по выражению:

$$C_Ж \approx nC_T N \approx 3\sum_{i=1}^{N} n_i \sum_{j=1}^{K} k_{ij} \sum_{i=1}^{N} \sum_{j=1}^{K} n_i C_j k_{ij}, \qquad (17.1)$$

где N – количество внутренних слов ОПУ; $n_1, \ldots, n_N$ – длины слов; $n = \frac{n_1 + \ldots + n_N}{N}$ – средняя длина слова; k_{ij} – количество микроопераций типа $j = 1, 2, \ldots, K$ (сложение, сдвиг, передача и т.п.), используемых для вычисления слов с номерами $i = 1,2,\ldots, N$; C_T – цена триггера; C_j – цена одноразрядной схемы для реализации микрооперации j-го типа.

В приведенном выражении первое слагаемое определяет затраты на хранение n-разрядных слов, второе – на связи регистров с комбина- ционными схемами, а третье – суммарную стоимость комбинационных схем, реализующих микрооперации K типов над N словами.

Затраты времени на выполнение операций типа «сложение» в ОПУ с жесткой структурой равны:

$$T_{Ж} = t_{В} + t_{C} + t_{П}, \qquad (17.2)$$

где $t_{В}$ – длительность микрооперации выдачи операндов из регистров; t_C – продолжительность микрооперации «сложение»; $t_{П}$ – длительность микрооперации приема результата в регистр.

Достоинством операционного устройства с жесткой структурой является высокое быстродействие, недостатком – малая регулярность структуры, что затрудняет реализацию таких ОПУ в виде больших интегральных схем.

17.1.2. Операционные устройства с магистральной структурой

В операционном устройстве с магистральной структурой все внутренние регистры объединены в отдельный узел регистров общего назначения (РОН), а все комбинационные схемы – в операционный блок (ОПБ). Операционный блок и узел регистров сообщаются между собой с помощью магистралей – отсюда и название «магистральное ОПУ».

Пример магистрального операционного устройства представлен на рис. 17.3. В состав узла РОН здесь входят N регистров общего назначения, подключаемых к магистралям **A** и **B** через мультиплексоры **MUX A** и **MUX B**. Каждый из мультиплексоров является управляемым коммутатором, соединяющим выход одного из РОН с соответствующей магистралью. Номер подключаемого регистра определяется адресом a или b, подаваемым на адресные входы мультиплексора из устройства управления.

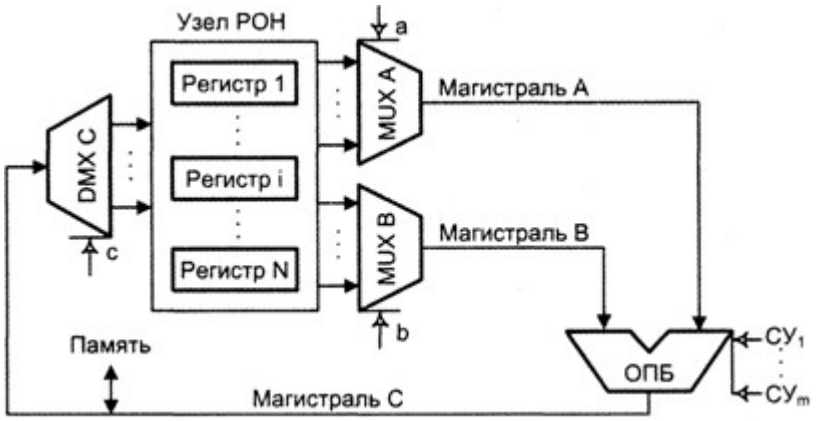


Рис. 17.3. Магистральное операционное устройство

По магистралям **A** и **B** операнды поступают на входы операционного блока, к которым подключается комбинационная схема, реализующая требуемую микрооперацию (по сигналу управления из УУ). Таким образом, любая микрооперация операционного блока может быть выполнена над содержимым любых регистров операционного устройства. Результат микрооперации по магистрали *C* через демультиплексор **DMX C** заносится в конкретный регистр узла РОН. Демультиплексор представляет собой управляемый коммутатор, имеющий один информационный вход и *N* информационных выходов. Вход подключается к выходу с заданным адресом **c**. Адрес c поступает на адресные входы **DMX C**из УУ.

Используя обозначения, введенные в предыдущем разделе, выражение для оценки аппаратных затрат на магистральное ОПУ можно записать в следующем виде:

$$C_{\text{M}} \square nC_{\text{T}}N \square 3n(N \square K) \square n \square \sum_{j \square 1}^{K} C_j, \qquad (17.3)$$

где первое слагаемое определяет затраты на *N* регистров, второе – затраты на связи узла РОН и ОПБ, а третье – суммарную цену ОПБ.

Из сопоставления выражений для затрат (3.1) и (3.3) следует, что магистральная структура экономичнее жесткой структуры, если

$$3(N \square K \square M) \square \sum_{i \square 1}^{N} \sum_{j \square 1}^{K} C_j K_{ij} \square \sum_{j \square 1}^{K} C_j, \qquad (3.4)$$

$$N \quad K$$

где $M \square \square \square K_{ij}$ – количество микроопераций, реализуемых

$$i \square 1 j \square 1$$

операционным устройством с жесткой структурой.

С учетом последнего неравенства можно сформулировать следующее сильное условие экономичности магистральных структур:

$$M \square N \square K. \qquad (17.5)$$

Затраты времени на сложение в магистральных ОПУ больше, чем в ОПУ с жесткой структурой:

$$T_{\text{М}} \square t_{\text{В}} \square t_{\text{С}} \square t_{\text{П}} \square t_{\text{MUX}} \square t_{\text{DMX}} \square t_{\text{Ж}} \square t_{\text{MUX}} \square t_{\text{DMX}}, \qquad (17.6)$$

где t_{MUX} – задержка на подключение операндов из РОН к ОПБ; t_{DMX} – задержка на подключение результата к РОН.

Основным достоинством магистральных ОПУ является высокая универсальность и регулярность структуры, что облегчает их реализацию на кристаллах ИС. Магистральная структура ОПУ в современных ВМ является превалирующей.

Магистральные операционные устройства классифицируют по виду и количеству магистралей, организации узла регистров общего назначения, типу операционного блока.

Магистрали могут быть однонаправленными и двунаправленными, соответственно обеспечивающими передачу данных в одном или двух различных направлениях. Типичным режимом работы магистрали является разделение времени, при котором в различные моменты времени магистраль используется для передачи функционально разнотипных данных.

По функциональному назначению выделяют:

– *магистрали внешних связей,* соединяющих ОПУ с памятью и каналами ввода/ вывода ВМ;

– *внутренние магистрали* ОПУ, отвечающие за связь между узлом РОН и операционным блоком.

Количество магистралей внешних связей зависит от архитектуры конкретной ВМ и обычно не превышает двух для внешних связей и трех – для внутренних.

Структура *трехмагистрального операционного устройства* (3М) представлена на рис. 17.4, *а,* а соответствующая ему микропрограмма выполнения операции типа «сложение» – на рис. 17.4, *б*.

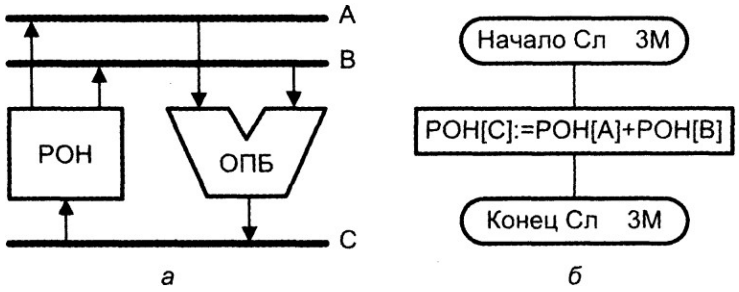


Рис. 17.4. Трехмагистральное операционное устройство:

а – структура; *б* – микропрограмма сложения

Данный вариант характеризуется наибольшим быстродействием: выборка операндов из РОН, выполнение микрооперации суммирования и запись результата в РОН производятся за один такт. Основной недостаток трехмагистральной организации – относительно большая площадь, занимаемая магистралями на кристалле интегральной микросхемы.

Двухмагистральное операционное устройство (2М) при меньшей площади, покрываемой магистралями, требует введения как минимум одного буферного регистра (БР), предназначенного для временного хранения одного из операндов (рис. 17.4, *а*), при этом операция сложения будет выполняться уже за два такта (рис. 17.5, *б*):

– Такт 1: загрузка буферного регистра одним из операндов.

– Такт 2: выполнение микрооперации в операционном блоке над содержимым буферного регистра и одного из РОН; запись результата в РОН.

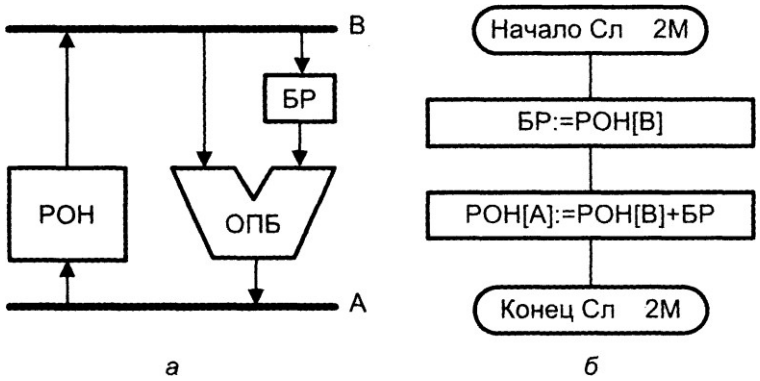


Рис. 17.5. Двухмагистральное операционное устройство:

а – структура; *б* – микропрограмма сложения

Одномагистральное операционное устройство минимизирует расходы площади ИМС (рис. 17.6, *а*).

В одномагистральных операционных устройствах (1М), вместе с тем, возникает необходимость введения не менее двух буферных регистров БР1, БР2, и длительность операции возрастает до трех тактов (рис. 17.6, *б*):

– такт 1: загрузка БР1 одним из операндов;

– такт 2: загрузка БР2 вторым операндом;

– такт 3: выполнение микрооперации в операционном блоке над содержимым БР1 и БР2; запись результата в один из РОН.

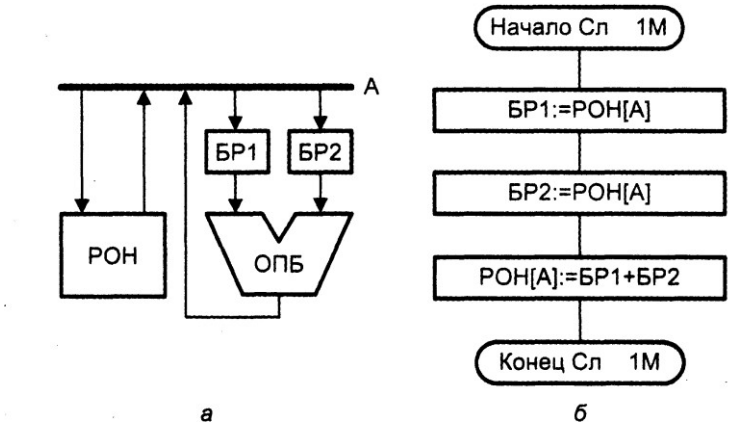


Рис. 17.6. Одномагистральное операционное устройство:

а – структура; *б* – микропрограмма сложения

Организация узла РОН магистрального операционного устройства. Количество регистров в узле РОН магистрального операционного устройства обычно превышает тот минимум, который необходим для реализации универсальной системы операций. Избыток регистров используется:

– для хранения составных частей адреса (индекса, базы);

– в качестве буферной, сверхоперативной памяти для повышения производительности ВМ за счет уменьшения требуемых пересылок между основной памятью и ОПУ.

Количество регистров колеблется в среднем от 8 до 16, иногда может достигать 32÷64. В процессорах с сокращенным набором команд количество РОН доходит до нескольких сотен.

Организация узла РОН может обеспечивать одноканальный или двухканальный доступ как ко входу (записи), так и к выходу (считыванию). В первом случае ко входу узла подключается один демультиплексор, а к выходу – один мультиплексор. Во втором случае доступ осуществляется с помощью двух демультиплексоров и (или) двух мультиплексоров. Двухканальный доступ повышает быстродействие операционного устройства, так как позволяет обратиться параллельно к двум регистрам.

Организация операционного блока магистрального операционного устройства. Тип операционного блока (ОПБ) определяется способом обработки данных. Различают ОПБ последовательного и параллельного типа.

В *последовательном операционном блоке* (рис. 17.7) операции выполняются разряд за разрядом.

Бит переноса, возникающий при обработке i-го разряда операндов, подается на вход ОПБ и учитывается при обработке $(i + 1)$-го разряда операндов. Результат поразрядно заносится в выходной регистр, предыдущее содержимое которого перед этим сдвигается на одну позицию. Таким образом, после n циклов в выходном регистре формируется слово результата, где каждый разряд занимает предназначенную для него позицию.

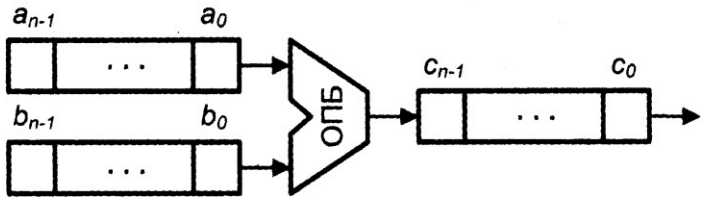


Рис. 17.7. Последовательный операционный блок

При *параллельной организации операционного блока* (рис. 17.8) все разряды операндов обрабатываются одновременно. Внутренние переносы обеспечиваются схемой ОПБ.

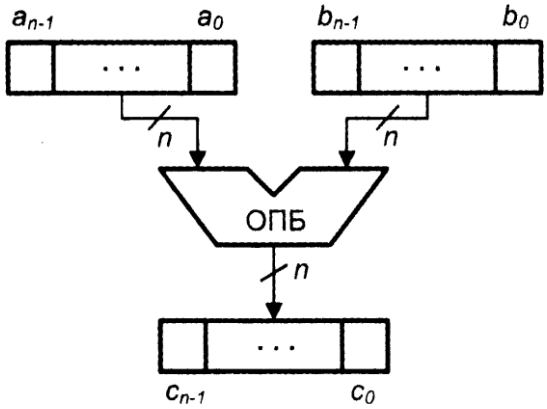


Рис. 17.8. Параллельный операционный блок

Реализация эффективной системы переносов в рамках «длинного» слова сопряжена с определенными аппаратурными издержками, поэтому на практике часто используют параллельно-последовательную схему ОПБ.

В ней слово разбивается на группы по 2, 4 или 8 разрядов, обработка всех разрядов внутри группы осуществляется параллельно, а сами группы обрабатываются последовательно.

Обобщенная схема ОПБ приведена на рис. 17.9. В нее входят: дешифратор микрокоманды **ДшМК**, формирователи кодов **ФК1** и **ФК2**, многофункциональный сумматор **См**, сдвигатель **Сдв** и формирователь признаков результата (ФПР). Набор микроопераций, реализуемых ОПБ, выбран таким образом, чтобы обеспечить выполнение основных арифметических и логических операций, предусмотренных системой команд вычислительной машины.

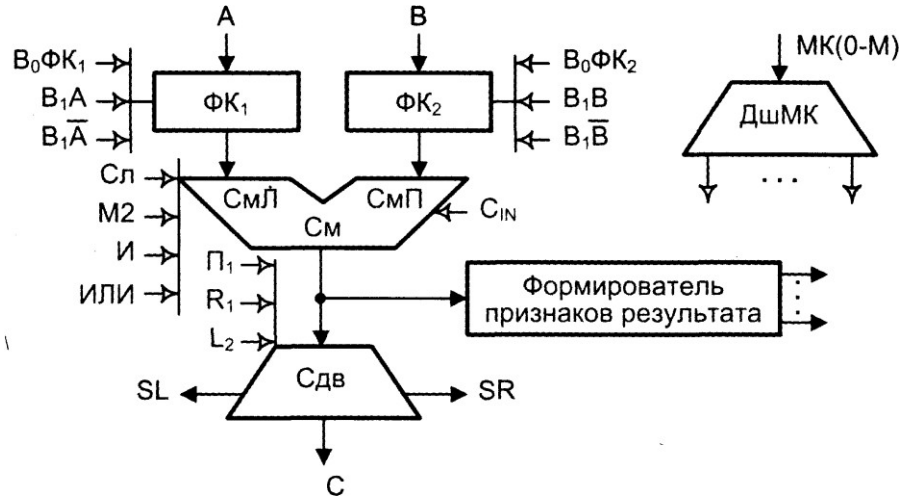


Рис. 17.9. Обобщенная схема операционного блока

Дешифратор микрокоманды вырабатывает внутренние сигналы управления для элементов ОПБ. Он введен в схему с целью минимизации количества связей, требуемых для передачи сигналов управления из УУ.

Формирователи кодов **ФК1** и **ФК2** служат для формирования прямых и инверсных кодов операндов, поступающих по магистралям **А**и **В**.

Многофункциональный сумматор выполняет микрооперации арифметического сложения (с учетом переноса C_{IN}), сложения по модулю два, логического сложения и логического умножения кодов на левом и правом входах.

Формирователь признаков результата на основе анализа кода на выходе **См** вырабатывает значения осведомительных сигналов (признаков результата), передаваемых в УУ машины. Осведомительными сигналами могут быть: признак знака **S**, признак переполнения **V**, признак нулевого значения результата **Z**и т.п.

Вспомогательные системы счисления, используемые в операционных устройствах

Хотя стандартная двоичная система в фон-неймановских ВМ является основной, при построении операционных устройств определенную пользу может принести временный переход к иным системам счисления. Прежде всего, это касается операционных устройств для умножения и деления, где использование промежуточных систем счисления позволяет существенно сократить время выполнения данных операций. Естественно, что по завершении операции ее результат представляется в стандартной двоичной системе. В роли вспомогательных обычно выступают *избыточные системы счисления* и *системы счисления с основанием, кратным целой степени числа* 2, либо их сочетание.

Избыточные системы счисления

В отличие от обычной позиционной системы, где цифра числа может принимать значения в диапазоне от 0 до q-1 *(q* – основание системы счисления), в избыточных системах цифра может иметь более чем q значений, например иметь знак. Если в стандартной двоичной системе счисления значение цифры ограничено множеством {0, 1}, то в знакоцифровой системе с тем же основанием $q = 2$ возможное значение цифры определяется множеством {–1, 0, 1}.

Число в избыточной системе может быть записано несколькими способами (по этой причине такие системы и называют избыточными). Например, пятиразрядное двоичное представление числа одиннадцать 01011(8 + 2 + 1)в избыточной системе с основанием 2 может иметь

три представления: 01011 (8+2+1), 10101 ($16\bar{4}\bar{1}$) и 01101 ($8+4-\bar{1}$).

Промежуточный переход к избыточной системе счисления с последующим возвратом к стандартной двоичной системе позволяет создавать эффективные ОПУ, особенно для тех арифметических операций, реализация которых носит итеративный характер. Так, стандартное умножение на двоичное число 11111111 предполагает до 8 сложений (по числу разрядов множителя), в то время как при записи этого числа

в избыточной системе с $q = 2$ (100000001) можно обойтись лишь одним вычитанием и одним сложением.

Системы счисления с основанием, кратным целой степени 2. В основе операционных устройств умножения и деления лежит итеративное выполнение операций сложения и сдвига. Количество итераций в общем случае равно разрядности операндов. Сокращение числа итераций – наиболее очевидный путь ускорения работы соответствующих ОПУ.

Такое сокращение возможно за счет перехода к системам счисления с большим основанием, причем наиболее удобны в этом смысле системы с основанием *q,* кратным целой степени числа 2, в частности: $q = 4$ (Radix-4), $q = 8$ (Radix-8), $q = 16$ (Radix-16). Одна цифра в таких системах эквивалентна 2, 3 и 4 двоичным цифрам соответственно. Это позволяет, например, при выполнении операции умножения за один раз анализировать не один разряд множителя, а сразу 2, 3 или 4 и, соответственно, сократить общее число итераций, требуемых для анализа всех разрядов множителя.

Избыточные системы счисления с основанием, кратным целой степени 2. В ОПУ умножения и деления широкое распространение получили знакоцифровые системы счисления у которых основание кратно степени числа 2, а именно 2, 4, 8 и 16. В соответствии со значением q их обычно принято обозначать как Radix-2 $(q = 2)$, Radix-4 $(q = 4)$, Radix-8 $(q = 8)$ и Radix-16 $(q = 16)$. В таких системах счисления используются цифры в диапазоне от $-(q-1)$ до $+(q-1)$. Так, Radix-8 предполагает использования цифр из множества {–7, –6, –5, –4, –3, –2, –1, 0, 1, 2, 3, 4, 5, 6, 7}. Сочетание избыточности и большого основания системы счисления дает дополнительный эффект в сокращении числа итераций при выполнении операций умножения и деления, а значит, и в сокращении времени выполнения этих операций.

Естественно, что переход к вспомогательным системам счисления порождает определенное усложнение аппаратных средств ОПУ, но чаще всего это окупается, выигрышем в быстродействии.

Контрольные вопросы

1. Охарактеризуйте состав операционных устройств, входящих в АЛУ. Из каких соображений и каким образом он может изменяться?
2. Поясните понятие «операционные устройства с жесткой структурой». В чем заключается жесткость их структуры? Каковы их достоинства и недостатки?
3. Чем обусловлено название операционных устройств с магист- ральной структурой? Сравните магистральные структуры с жесткими структурами, выделяя достоинства, недостатки и область применения.
4. Дайте развернутую характеристику классификации операционных устройств с магистральной структурой. Поясните достоинства и недостатки «минимального» и «максимального» вариантов.
5. Поясните функциональный состав параллельного операционного блока магистрального ОПУ. Каким образом можно минимизировать количество внешних связей этого блока? Ответ сопроводите конкретным примером.

Информация по теме:

Линия связи в общем случае состоит из физической среды, по которой передаются электрические информационные сигналы, аппаратуры передачи данных и промежуточной аппаратуры. Синонимом термина линия связи является термин канал связи.

В зависимости от среды передачи данных линии связи разделяют на проводные, кабельные (медные, волоконно-оптические), радиоканалы наземной и спутниковой связи.

В компьютерных сетях применяются типы кабеля: на основе скрученных пар медных проводов, коаксиальные с медной жилой и волоконно-оптические. Основные характеристики кабельных линий связи: амплитудно-частотная характеристика, полоса пропускания, затухание, помехоустойчивость, перекрестные наводки на ближнем конце линии, пропускная способность, достоверность передачи данных, удельная стоимость.

Новый стандарт EIA/TIA-568A коаксиальные кабели не описывает, как морально устаревшие.

Медный неэкранированный кабель UTP делится на 5 категорий (Category 1 - Category 5).

Основным стандартом неэкранированной витой пары STP является фирменный стандарт IBM, в котором кабели делятся не на категории, а на типы: Type 1, Type 2, . . . , Type 9.

В зависимости от распределения показателя преломления и от величины диаметра сердечника различают: многомодовое волокно со ступенчатым изменением показателя преломления, многомодовое волокно с плавным изменением показателя преломления, одномодовое волокно.

В спутниковых системах используются антенны СВЧ-диапазона частот для приема радиосигналов от наземных станций и ретрансляции этих сигналов обратно на наземные станции.

LMDS (Local Multipoint Distribution System) - это стандарт сотовых сетей беспроводной передачи информации для фиксированных абонентов.

Радиоканалы WiMAX (Worldwide Interoperability for Microwave Access) в отличие от традиционных технологий радиодоступа работают и на отраженном сигнале, вне прямой видимости базовой станции.

Радиоканалы MMDS (Multichannel Multipoint Distribution System). Эти системы способна обслуживать территорию в радиусе 50—60 км, при этом прямая видимость передатчика оператора является не обязательной.

Стандартом беспроводной связи для локальных сетей является технология Wi-Fi. Wi-Fi обеспечивает подключение в двух режимах: точка-точка (для подключения двух ПК) и инфраструктурное соединение (для подключения несколько ПК к одной точке доступа). Радиоканалы Bluetooht – это технология передачи данных на короткие расстояния (не более 10 м) и может быть использована для создания домашних сетей.

Ссылки на литературные источники, приведенные в рабочей программе дисциплины: пп. 1, 2 основной литературы, интернет-источники, периодические издания.

Домашнее задание:
Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 с

Раздел 7. Системы памяти. Организация основной памяти.

Занятие 7 (лекция)

Запоминающие устройства характеризуются следующими параметрами:

– месторасположение;
– ёмкость;
– единица пересылки;
– метод доступа;
– быстродействие;
– физический тип;
– физические особенности;
– стоимость.

По месту расположения запоминающие устройства разделяют на: процессорные, внутренние и внешние.

Процессорные виды памяти (регистры, кэш-память первого уровня) обычно размещают на общем кристалле с центральным процессором и являются наиболее скоростными, а регистры общего назначения вообще считаются частью центрального процессора.

Внутренние ЗУ образуют запоминающие устройства, расположенные на системной плате. К внутренней памяти относят основную память, а также кэш-память второго и последующих уровней (кэш-память второго уровня может также размещаться на кристалле процессора).

Внешние ЗУ к ядру вычислительной машины подключаются аналогично устройствам ввода/вывода являются и являются наиболее медленными запоминающими устройствами (магнитные и оптические диски, магнитные ленты). Однако они характеризуются большой емкостью.

Ёмкость запоминающего устройства характеризуют числом битов либо байтов, которое может храниться в запоминающем устройстве. На практике применяются более крупные единицы: килобайт, мегабайт, гигабайт, терабайт. В вычислительной технике, ориентированной на двоичную систему счисления, они соответствуют значениям целой степени числа 2, т.е. 2^{10}, 2^{20}, 2^{30}, 2^{40}.

Для основной памяти (ОП) *единица пересылки* определяется шириной шины данных, то есть количеством битов, передаваемых по линиям шины параллельно. Обычно единица пересылки равна длине

слова, но не обязательно. Применительно к внешней памяти данные часто передаются единицами, превышающими размер слова, и такие единицы называются блоками.

Различают четыре основных *метода доступа к данным*: последовательный, прямой, произвольный и ассоциативный. Для внутренней памяти характерны произвольный и ассоциативный.

Последовательный доступ. Запоминающие устройства с последовательным доступом ориентированы на хранение информации в виде последовательности блоков данных, называемых записями. Для доступа к нужному элементу (слову или байту) необходимо прочитать все предшествующие ему данные. Время доступа зависит от положения требуемой записи в последовательности записей на носителе информации и позиции элемента внутри данной записи. Примером может служить ЗУ на магнитной ленте.

Прямой доступ. Каждая запись имеет уникальный адрес, отражающий ее физическое размещение на носителе информации. Обращение осуществляется как адресный доступ к началу записи, с последующим последовательным доступом к определенной единице информации внутри записи. В результате время доступа к определенной позиции является величиной переменной. Такой режим характерен для магнитных дисков.

Произвольный доступ. Каждая ячейка памяти имеет уникальный физический адрес. Обращение к любой ячейке занимает одно и то же время и может производиться в произвольной очередности. Примером могут служить запоминающие устройства основной памяти.

Ассоциативный доступ. Этот вид доступа позволяет выполнять поиск ячеек, содержащих такую информацию, в которой значение отдельных битов совпадает с состоянием одноименных битов в заданном образце. Сравнение осуществляется параллельно для всех ячеек памяти, независимо от ее емкости. По ассоциативному принципу построены некоторые блоки кэш-памяти.

Для количественной оценки *быстродействия* используются четыре параметра:

1. Время выборки данных. Оно соответствует интервалу времени между началом операции считывания и выдачей считанных данных из запоминающего устройства.
2. Время хранения данных – интервал времени, в течение которого запоминающее устройство в заданном режиме сохраняет данные без регенерации.

3. Цикл обращения к ЗУ или период обращения ($T_Ц$). Это минимальный интервал времени между двумя последовательными доступами к данным запоминающего устройства. Период обращения включает в себя время доступа плюс некоторое дополнительное время. Дополнительное время может требоваться для затухания сигналов на линиях.

4. Скорость передачи данных – количество данных, считываемых (записываемых) запоминающим устройством в единицу времени.

Физический тип запоминающего устройства подразумевает материал, из которого изготовлено ЗУ, например, ЗУ внутренней памяти современных вычислительных машин базируются на полупроводниковой технологии.

В зависимости от примененной технологии следует учитывать и ряд *физических особенностей* ЗУ. Так, для полупроводниковой технологии приходится учитывать фактор энергозависимости. В энергозависимой памяти информация может быть искажена или потеряна при отключении источника питания, в то время как в энергонезависимых ЗУ записанная информация сохраняется и при отключении питающего напряжения. Полупроводниковая память может быть как энергозависимой, так и нет, в зависимости от ее типа. Помимо энергозависимости нужно учитывать, приводит ли считывание информации к ее разрушению.

Стоимость ЗУ принято оценивать отношением общей стоимости ЗУ к его емкости в битах, т.е. стоимостью хранения одного бита информации.

22.1. Иерархия запоминающих устройств

Иерархическая память состоит из запоминающих устройств различных типов (рис. 22.1), которые, в зависимости от характеристик, относят к определенному уровню иерархии. Более высокий уровень меньше по емкости, быстрее и имеет большую стоимость в пересчете на бит, чем более низкий уровень. Уровни иерархии взаимосвязаны: все данные на одном уровне могут быть также найдены на более низком уровне, и все данные на этом более низком уровне могут быть найдены на следующем нижележащем уровне и т.д.

Три верхних уровня иерархии образуют *внутреннюю память* вычислительной машины, а все нижние уровни – это *внешняя* или *вторичная* память. По мере движения вниз по иерархической структуре:

– уменьшается соотношение «стоимость/бит»;

– возрастает ёмкость;

– растет время выборки;

– уменьшается частота обращения к памяти со стороны центрального процессора.

Из свойства локальности программы (программа занимает как правило определенную область памяти и расположена в ней компактно) вытекает, что программу разумно представить в виде последовательно обрабатываемых фрагментов – компактных групп команд и обрабатываемых ими данных. Помещая такие фрагменты в более быструю память, можно существенно снизить общие задержки на обращение, поскольку команды и исходные данные, будучи один раз переданы из медленного ЗУ в быстрое, затем могут использоваться многократно, и среднее время доступа к ним в этом случае определяется уже более быстрым ЗУ. Это позволяет хранить большие программы и массивы данных на медленных, емких, но дешевых ЗУ, а в процессе обработки активно использовать сравнительно небольшую быструю память, увеличение емкости которой сопряжено с высокими затратами.

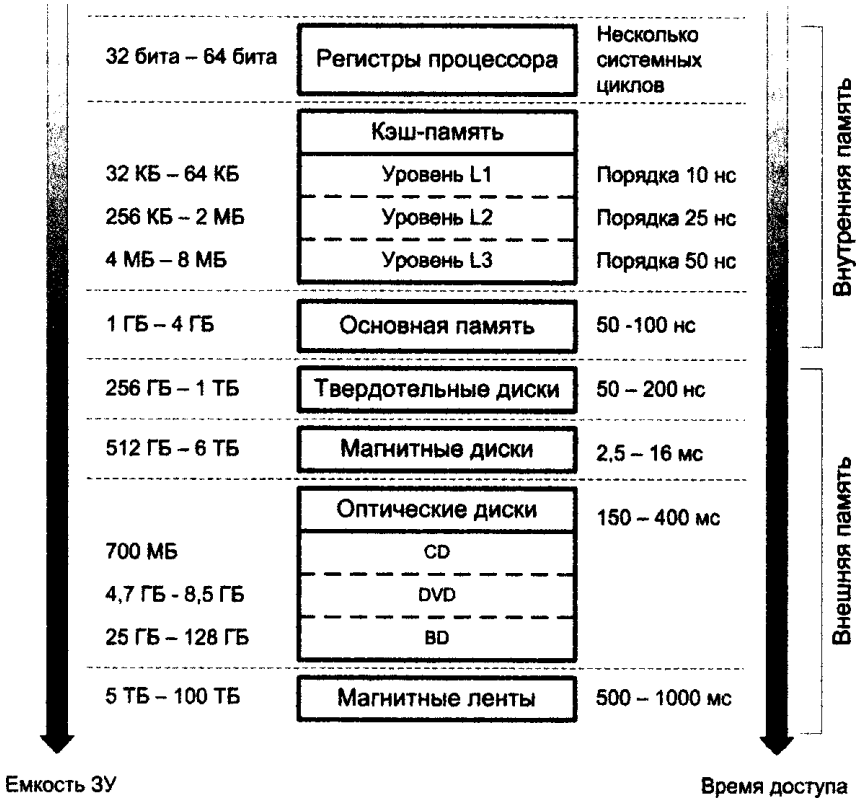


Рис. 22.1. Иерархия запоминающих устройств

При доступе к командам и исходным данным, например для их считывания, сначала производится поиск в памяти верхнего уровня. Факт обнаружения нужной информации называют *попаданием* (hit), в противном случае говорят о *промахе* (miss). При промахе производится

поиск в ЗУ следующего более низкого уровня, где также возможны попадание или промах. После обнаружения необходимой информации выполняется пересылка блока, содержащего искомую информацию с нижних уровней на верхние.

При оценке эффективности подобной организации памяти обычно используют следующие характеристики:

– *коэффициент попаданий* (hit rate) – отношение числа обращений к памяти, при которых произошло попадание, к общему числу обращений к ЗУ данного уровня иерархии;

– *коэффициент промахов* (miss rate) – отношение числа обращений к памяти, при которых имел место промах, к общему числу обращений к ЗУ данного уровня иерархии;

– *время обращения при попадании* (hit time) – время, необходимое для поиска нужной информации в памяти верхнего уровня (включая выяснение, является ли обращение попаданием), плюс время на фактическое считывание данных;

– *потери на промах* (miss penalty) – время, требуемое для замены блока в памяти более высокого уровня на блок с нужными данными, расположенный в ЗУ следующего (более низкого) уровня. Потери на промах включают в себя:

– *время доступа* (access time) – время обращения к первому слову блока при промахе; время доступа обусловлено задержкой памяти более низкого уровня;

– *время пересылки* (transfer time) – дополнительное время для, пересылки оставшихся слов блока; время пересылки связано с полосой пропускания канала между ЗУ двух смежных уровней.

Самый быстрый, но и минимальный по емкости тип памяти – это внутренние регистры ЦП, которые иногда объединяют понятием *сверхоперативное запоминающее устройство* – СОЗУ или *регистровый файл.* Как правило, количество регистров невелико, хотя в архитектурах с сокращенным набором команд их число может доходить до нескольких сотен. Основная память (ОП), значительно большей ёмкости, располагается ниже. Между регистрами ЦП и основной памятью часто размещают кэш-память, которая по ёмкости ощутимо проигрывает ОП, но существенно превосходит последнюю по быстродействию, уступая в то же время СОЗУ. Все виды внутренней памяти реализуются на основе полупроводниковых технологий и в основном являются энергозависимыми.

Долговременное хранение больших объемов данных обеспечивается внешними ЗУ, среди которых наиболее распространены запоминающие

устройства на основе магнитных и оптических дисков, а также магнитоленточные ЗУ. В последнее время все большую популярность получают твердотельные диски на базе флэш-памяти.

Еще один уровень иерархии может быть добавлен между основной памятью и магнитными дисками. Этот уровень носит название дисковой кэш-памяти и реализуется в виде самостоятельного ЗУ, включаемого в состав магнитного диска. Дисковая кэш-память существенно улучшает производительность при обмене информацией между дисками и основной памятью.

22.2. Основная память

Основная память (ОП) представляет собой единственный вид памяти, к которой ЦП может обращаться непосредственно (исключение составляют лишь регистры центрального процессора). Информация, хранящаяся на внешних ЗУ, становится доступной процессору только после того, как будет переписана в основную память.

Основную память образуют запоминающие устройства с произвольным доступом. Такие ЗУ образованы как массив ячеек, а «произвольный доступ» означает, что обращение к любой ячейке занимает одно и то же время и может производиться в произвольной последовательности. Каждая ячейка содержит фиксированное число запоминающих элементов и имеет уникальный адрес, позволяющий отличать ее от других ячеек.

Основная память может включать в себя два типа устройств; *оперативные запоминающие устройства* (ОЗУ) и *постоянные запоминающие устройства* (ПЗУ).

Преимущественную долю основной памяти образует ОЗУ, допускающее как запись, так и считывание информации, причем обе операции выполняются однотипно, практически с одной и той же скоростью. В англоязычной литературе ОЗУ соответствует аббревиатура RAM – *Random Access Memory,* то есть «память с произвольным доступом». Микросхема ОЗУ должна быть постоянно подключена к источнику питания и поэтому может использоваться только как временная память.

Вторую группу полупроводниковых ЗУ основной памяти образуют энергонезависимые микросхемы ПЗУ (ROM – *Read-Only Memory).* ПЗУ обеспечивает считывание информации, но либо вообще не допускает ее изменения, либо процесс такого изменения (запись) сильно отличается от считывания и требует значительно большего времени.

22.2.1. Блочная организация основной памяти

Емкость основной памяти современных вычислительных машин слишком велика, чтобы ее можно было реализовать на базе единственной интегральной микросхемы (ИМС). Необходимость объединения нескольких ИМС ЗУ возникает также, когда разрядность ячеек в микросхеме ЗУ меньше разрядности слов ВМ.

Увеличение разрядности ЗУ реализуется за счет объединения адресных входов объединяемых ИМС ЗУ. Информационные входы и выходы микросхем являются входами и выходами модуля ЗУ увеличенной разрядности (рис. 22.2). Полученную совокупность микросхем называют *модулем памяти.* Модулем можно считать и единственную микросхему, если она уже имеет нужную разрядность. Один или несколько модулей образуют *банк памяти*.

Для получения требуемой емкости ЗУ нужно определенным образом объединить несколько банков памяти меньшей ёмкости. В общем случае основная память вычислительной машины практически всегда имеет блочную структуру, то есть содержит несколько банков.

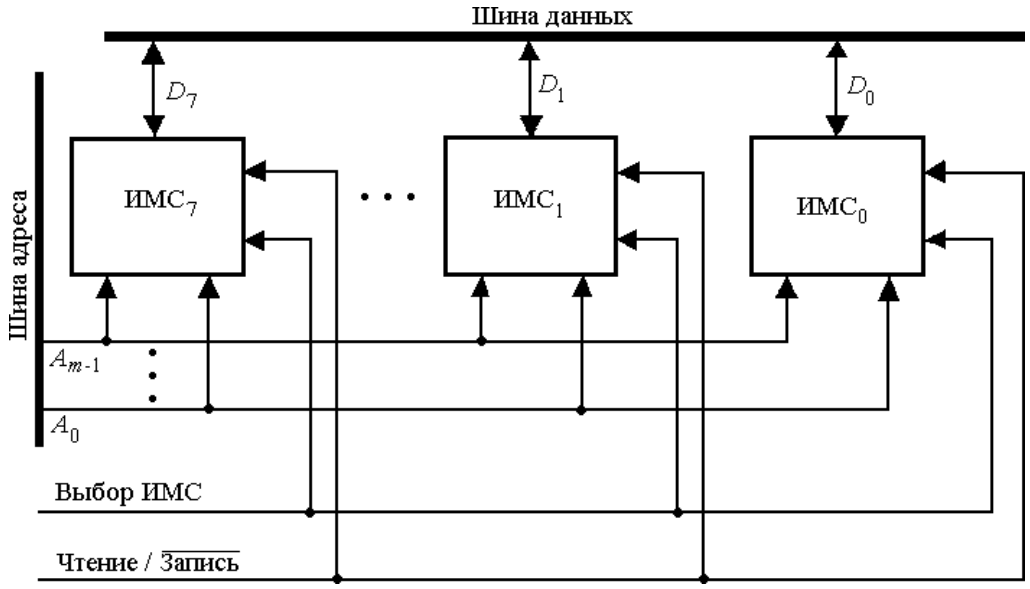


Рис. 22.2. Увеличение разрядности памяти

При использовании блочной памяти, состоящей из B банков, адрес ячейки A преобразуется в пару (b, w), где b – номер банка, w – адрес ячейки внутри банка. Известны три схемы распределения разрядов адреса A между b и w:

– блочная (номер банка b определяет старшие разряды адреса);

– циклическая ($b = A$ mod B; $w = A$ div B);

– блочно-циклическая (комбинация двух предыдущих схем).

Рассмотрим основные структуры блочной ОП на примере памяти емкостью 512 слов (2^9), построенной из четырех банков по 128 слов в каждом.

Блочная структура памяти показана на рис. 22.3.

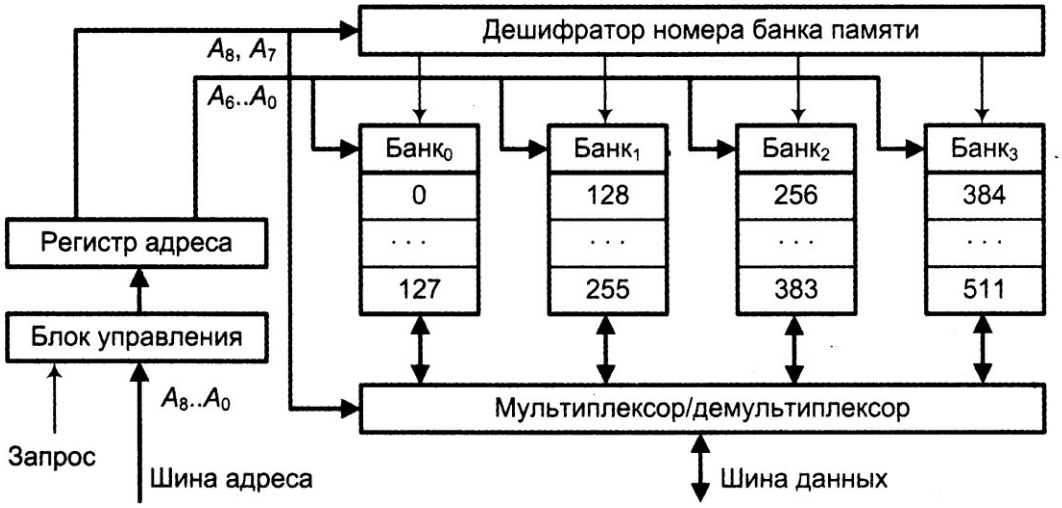


Рис. 22.3. Структура основной памяти на основе блочной схемы

Адресное пространство памяти разбито на группы последовательных адресов, и каждая такая группа обеспечивается отдельным банком памяти. Для обращения к ОП используется 9-разрядный адрес, семь младших разрядов которого ($A_6 \div A_0$) поступают параллельно на все банки памяти и выбирают в каждом из них одну ячейку. Два старших разряда адреса (A_8, A_7) содержат номер банка. Выбор банка обеспечивается либо с помощью дешифратора номера банка памяти, либо путем мультиплексирования информации (на рис. 22.3 показаны оба варианта). В функциональном отношении такая ОП может рассматриваться как единое ЗУ, ёмкость которого равна суммарной емкости составляющих, а быстродействие – быстродействию отдельного банка.

Блочное построение памяти дает еще одно преимущество – одновременный доступ ко многим банкам памяти, что существенно повышает скорость доступа. Для этого используется методика, которая называется *расслоение памяти*. В ее основе лежит так называемое *чередование адресов* (address interleaving), заключающееся в изменении системы распределения адресов между банками памяти.

Прием чередования адресов базируется на свойстве локальности по обращению, согласно которому последовательный доступ в память обычно производится к ячейкам, имеющим смежные адреса.

Иными словами, если в данный момент выполняется обращение к ячейке с адресом 5, то следующее обращение, вероятнее всего, будет к ячейке с адресом 6, затем 7 и т. д. Чередование адресов обеспечивается за счет циклического разбиения адреса. В нашем примере (рис. 22.4) для выбора банка используются два младших разряда адреса *(*A_1, A_0), а для выбора ячейки в банке – 7 старших разрядов ($A_8 \div A_2$).

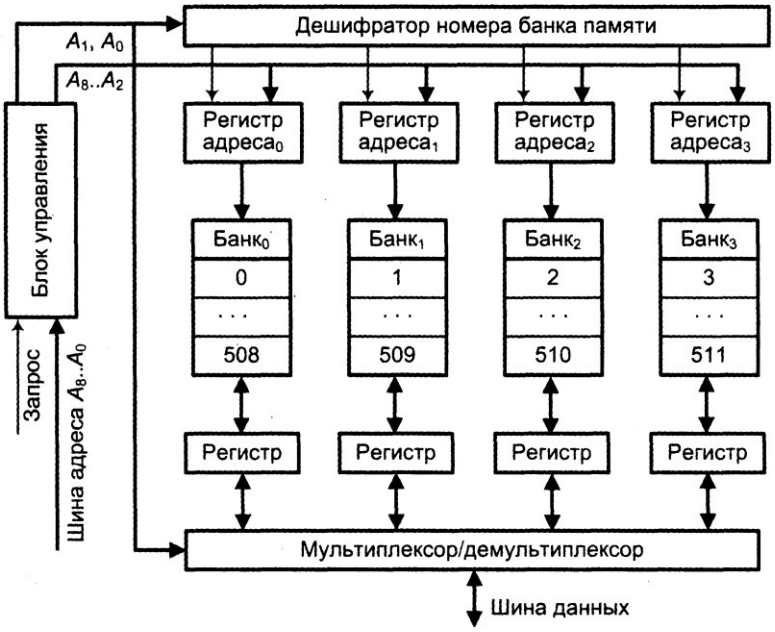


Рис. 22.4. Блочная память с чередование адресов по циклической схеме

Так как в каждом такте на шине адреса может присутствовать адрес только одной ячейки, параллельное обращение к нескольким банкам невозможно, однако оно может быть организовано со сдвигом на один такт. Адрес ячейки запоминается в индивидуальном регистре адреса, и дальнейшие операции по доступу к ячейке в каждом банке протекают независимо. При большом количестве банков среднее время доступа к ОП сокращается почти в B раз *(*B – количество банков), но при условии, что ячейки, к которым производится последовательное обращение, относятся к разным банкам. Если же запросы к одному и тому же банку следуют друг за другом, каждый следующий запрос должен ожидать завершения обслуживания предыдущего. Такая ситуация называется *конфликтом по доступу*. При частом возникновении конфликтов по доступу метод становится неэффективным.

В блочно-циклической схеме (рис. 22.5) расслоения памяти каждый

банк состоит из нескольких модулей, адресуемых по круговой схеме. Адреса между банками распределены по блочной схеме. Таким образом,

адрес ячейки разбивается на три части. Старшие биты определяют номер банка, следующая группа разрядов адреса указывает на ячейку в модуле, а младшие биты адреса выбирают модуль в банке.

Рис. 22.5. Блочно-циклическая схема расслоения памяти

Традиционные способы расслоения памяти хорошо работают в рамках одной задачи, для которой характерно свойство локальности. В многопроцессорных системах с общей памятью, где запросы на доступ к памяти достаточно независимы, не исключен иной подход, который можно рассматривать как развитие идеи расслоения памяти. Для этого в систему включают несколько контроллеров памяти, что позволяет отдельным банкам работать автономно. Эффективность данного приема зависит от частоты независимых обращений к разным банкам. Лучшего результата можно ожидать при большом числе банков, что уменьшает вероятность последовательных обращений к одному и тому же банку памяти.

22.2.2. Синхронные и асинхронные запоминающие устройства

По способу синхронизации ЗУ подразделяются на синхронные и асинхронные.

В микросхемах, где реализован *синхронный принцип,* процессы чтения и записи (если это ОЗУ) выполняются одновременно с тактовыми сигналами контроллера памяти.

Асинхронный принцип предполагает, что момент начала очередного действия определяется моментом завершения предшествующей операции. В *асинхронных ЗУ* цикл чтения начинается только при поступлении запроса от контроллера памяти, и если память не успевает выдать данные в текущем такте, контроллер может считать их только в следующем такте, поскольку очередной шаг контроллера начинается с приходом очередного тактового импульса. В последнее время асинхронная схема активно вытесняется синхронной.

22.2.3. Организация микросхем памяти

Интегральные микросхемы (ИМС) памяти представляют собой массив запоминающих элементов (ЗЭ). Запоминающий элемент способен хранить 1 бит информации. Для запоминающего элемента любой полупроводниковой памяти характерны следующие свойства:

– два стабильных состояния, представляющие двоичные 0 и 1;

– в запоминающий элемент (хотя бы однажды) может быть произведена запись информации, посредством перевода его в одно из двух возможных состояний;

– для определения текущего состояния запоминающего элемента его содержимое может быть считано.

Каждый запоминающий элемент в микросхеме памяти имеет определенный адрес, на основании которого осуществляется доступ к данному запоминающему элементу. На физическую организацию массива однобитовых запоминающих элементов накладывается логическая организация памяти, то есть разрядность микросхемы *n*. Разрядность микросхемы определяет количество запоминающих элементов, имеющих один и тот же адрес (такая совокупность запоминающих элементов называется *ячейкой*).

Массив запоминающих элементов организован в виде совокупности из *n* матриц. Таким образом, *i*-й разряд всех ячеек хранится в *i*-й матрице массива.

При матричной организации ИМС памяти (рис. 22.6) реализуется координатный принцип адресации ячеек: адрес ячейки, поступающий по шине адреса вычислительной машины, пропускается через логику выбора, где он разделяется на две составляющих: адрес строки и адрес столбца. Адреса строки и столбца запоминаются соответственно в регистре адреса строки и регистре адреса столбца микросхемы. Регистры соединены каждый со своим дешифратором. Выходы дешифраторов образуют систему горизонтальных и вертикальных линий, к которым подсоединены запоминающие элементы матрицы, при этом набор запоминающих элементов, образующий ячейку, расположен на пересечении одной горизонтальной и одной вертикальной линии.

Запоминающие элементы, объединенные общим «горизонтальным» проводом, принято называть *строкой* (row). Запоминающие элементы, подключенные к общему «вертикальному» проводу, называют *столбцом* (column). Кроме того, к каждому запоминающему элементы необходимо подключить линию, по которой будет передаваться считанная и записываемая информация.

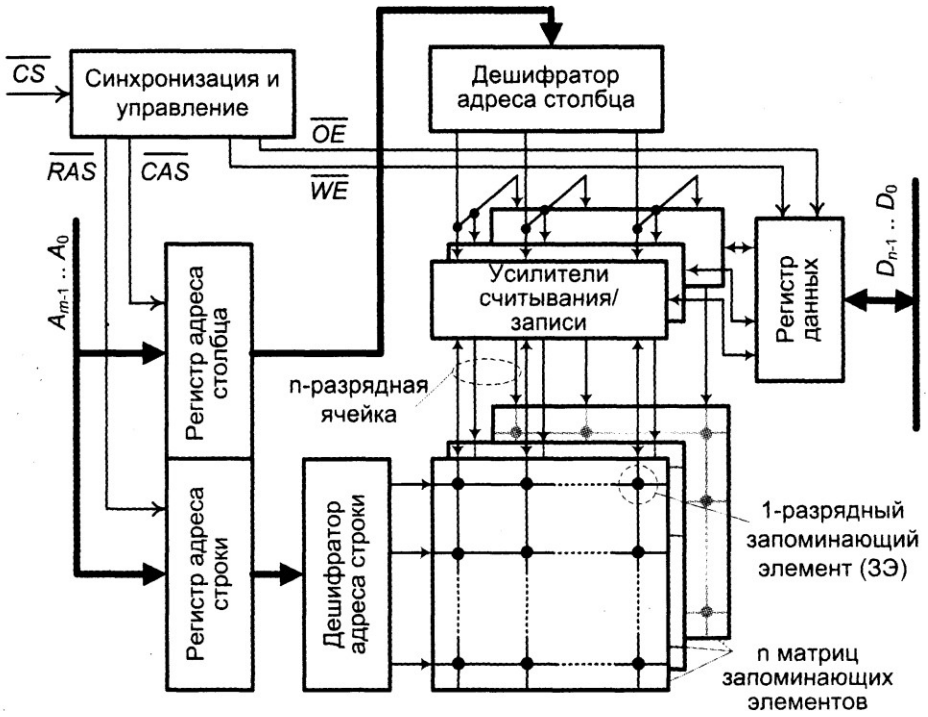


Рис. 22.6. Структура микросхемы памяти

Совокупность запоминающих элементов и логических схем, связанных с выбором строк и столбцов, называют *ядром* микросхемы памяти. Помимо ядра, в ИМС имеется еще интерфейсная логика, обеспечивающая взаимодействие ядра с внешним миром. В ее задачи, в частности, входят коммутация нужного столбца на выход при считывании и на вход – при записи.

Для уменьшения числа контактов ИМС адреса строки и столбца в большинстве микросхем подаются в микросхему через одни и те же контакты последовательно во времени (мультиплексируются) и запоминаются соответственно в регистре адреса строки и регистре адреса столбца микросхемы. Мультиплексирование обычно реализуется внешней по отношению к ИМС схемой.

Для синхронизации обработки адресной информации внутри ИМС адрес строки сопровождается сигналом RAS (Row Address Strobe – строб строки), а адрес столбца – сигналом CAS (Column Address Strobe – строб столбца). Чтобы стробирование было надежным, эти сигналы подаются с задержкой, достаточной для завершения переходных процессов на шине адреса и в адресных цепях микросхемы.

Сигнал выбора микросхемы CS (Chip Select) разрешает работу ИМС и используется для выбора определенной микросхемы в системах, состоящих из нескольких ИМС. Вход WE (Write Enable – разрешение записи) определяет вид выполняемой операции (считывание или запись).

Записываемая информация, поступающая по шине данных, первоначально заносится во входной регистр данных, а затем – в выбранную ячейку. При выполнении операции чтения информация из ячейки до ее выдачи на шину данных буферизируется в выходном регистре данных. Обычно роль входного и выходного выполняет один и тот же регистр. Усилители считывания/записи (УСЗ) служат для электрического согласования сигналов на линиях данных и внутренних сигналов ИМС. Обычно число УСЗ равно числу запоминающих элементов в строке матрицы, и все они при обращении к памяти подключаются к выбранной горизонтальной линии. Каждая группа УСЗ, образующая ячейку, подключена к одному из столбцов матрицы, то есть выбор нужной ячейки в строке обеспечивается активизацией одной из вертикальных линий. На все время, пока ИМС памяти не использует шину данных, информационные выходы микросхемы переводятся в третье (высо- коимпедансное) состояние, что эквивалентно отключению микросхемы от шины данных. Управление переключением в третье состояние обеспечивается сигналом OE (Output Enable – разрешение выдачи выходных сигналов). Этот сигнал активизируется при выполнении операции чтения.

Для большинства перечисленных выше управляющих сигналов активным обычно считается их низкий уровень.

Управление операциями с основной памятью осуществляется контроллером памяти, который может входить в состав центрального процессора либо реализуется в виде внешнего по отношению к памяти устройства. В последних типах ИМС памяти часть функций контроллера возлагается на микросхему памяти. Хотя работа ИМС памяти может быть организована как по синхронной, так и по асинхронной схеме, контроллер памяти – устройство синхронное, то есть срабатывающее исключительно по тактовым импульсам. По этой причине операции с памятью принято описывать с привязкой к тактам. В общем случае на каждую такую операцию требуется, как минимум, пять тактов, которые используются следующим образом:

1. Указание типа операции (чтение или запись) и установка адреса строки.
2. Формирование сигнала RAS.
3. Установка адреса столбца.
4. Формирование сигнала CAS.
5. Возврат сигналов RAS и CAS в неактивное состояние.

«Классическая» процедура доступа к памяти – чтение из ИМС с мультиплексированием адресов строк и столбцов (рис. 22.7) выглядит

следующим образом. Сначала на входе WE устанавливается уровень, соответствующий операции чтения, а на адресные контакты ИМС подается адрес строки, сопровождаемый сигналом RAS. По заднему фронту этого сигнала адрес запоминается в регистре адреса строки микросхемы, после чего дешифрируется. После стабилизации процессов, вызванных сигналом RAS, выбранная строка подключается к УСЗ. Далее на вход ИМС подается адрес столбца, который по заднему фронту сигнала CAS заносится в регистр адреса столбца. Одновременно подготавливается выходной регистр данных, куда после стабилизации сигнала CAS загружается информация с выбранных УСЗ.

Рис. 22.7. Временная диаграмма классической процедуры чтения

Быстродействие ИМС памяти характеризуется следующими параметрами:

– t_{RAS} – минимальное время от перепада сигнала RAS (с высокого уровня к низкому) до момента появления и стабилизации считанных данных на выходе ИМС. Это соответствует *времени выборки данных*;

– t_{RC} – минимальное время от начала доступа к одной строке микросхемы памяти до начала доступа к следующей строке. Это соответствует *циклу обращения к ЗУ*;

– t_{RAS} – минимальное время от перепада сигнала CAS с высокого уровня к низкому до момента появления и стабилизации считанных данных на выходе ИМС;

– T_{PC} – минимальное время от начала доступа к одному столбцу микросхемы памяти до начала доступа к следующему столбцу.

Наибольшее распространение получили следующие фундаментальные подходы к ускорению операции чтения/записи:

– последовательный;
– регистровый;
– быстрый постраничный;

– пакетный;
– конвейерный;
– удвоенной скорости.

Последовательный режим. Адрес и управляющие сигналы подаются на микросхему до поступления синхроимпульса. В момент прихода синхроимпульса вся входная информация запоминается во внутренних регистрах – по его переднему фронту, и начинается цикл чтения. Через некоторое время, но в пределах того же цикла, данные появляются на внешней шине, причем момент этот определяется только моментом прихода синхронизирующего импульса и скоростью внутренних цепей микросхемы.

Регистровый режим отличается от последовательного режима наличием регистра на выходе микросхемы. Считанные данные заносятся в промежуточный выходной регистр и хранятся там до появления отрицательного фронта (спада) синхроимпульса и в этот момент передаются на шину. Изменяя ширину импульса синхронизации, можно менять время появления данных на шине, что оказывается весьма полезным при проектировании специализированных ВМ. По быстродействию микросхемы с регистровым режимом идентичны ИМС с последовательным режимом.

Быстрый постраничный режим. Под страницей понимается строка матрицы ЗЭ. Последовательность элементов страницы соответствует ячейкам памяти с последовательно возрастающими адресами. Это позволяет при доступе к ячейкам со смежными адресами (согласно принципу локальности такая ситуация наиболее вероятна) существенно ускорить доступ ко второй и последующим ячейкам. Для доступа к очередной ячейке достаточно подавать на ИМС лишь адрес нового столбца, сопровождая его сигналом CAS. *Обращение к первой ячейке в последовательности производится стандартным образом* – поочередным заданием адреса строки и адреса столбца, то есть здесь время доступа уменьшить практически невозможно. Рассмотренный режим называется *режимом постраничного доступа* или просто *постраничным режимом* (Page Mode). В интегральных микросхемах DRAM распространена модификация режима, известная как *быстрый постраничный режим* (FPM – Fast Page Mode). Особенность модификации – в способе занесения новой информации в регистр адреса столбца. Полный адрес (строки и столбца) передается только при первом обращении к строке. Активизация регистра адреса столбца производится не по сигналу CAS, а по заднему фронту сигнала RAS. Сигнал RAS

остается активным на протяжении всего страничного цикла и позволяет заносить в регистр адреса столбца новую информацию не по спадающему фронту CAS, а как только адрес на входе ИМС стабилизируется, то есть практически по переднему фронту сигнала CAS. Временная диаграмма операции чтения для рассматриваемого режима показана на рис. 22.8.

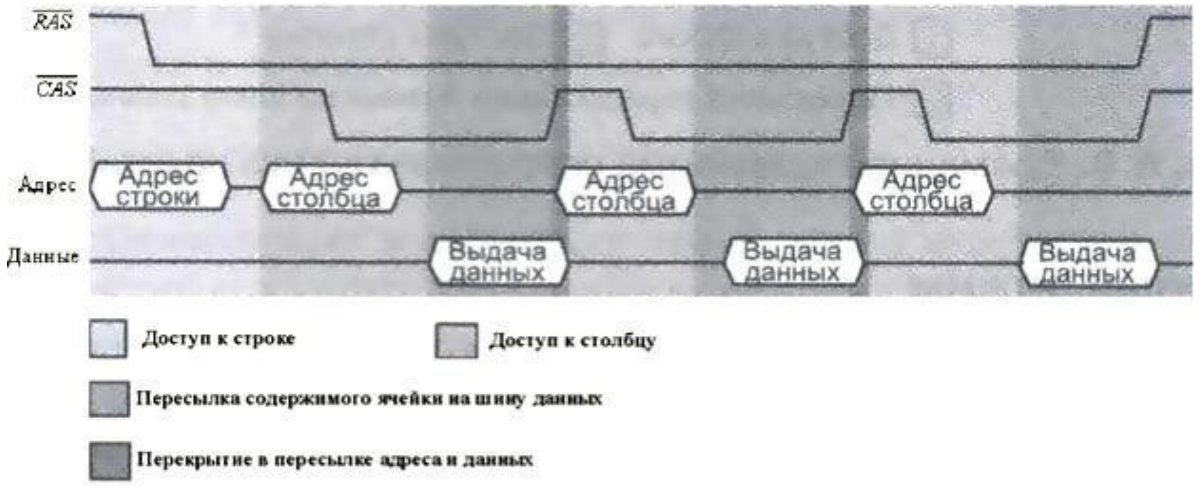


Рис. 22.8. Временная диаграмма операции чтения в быстром постраничном режиме

В целом потери времени сокращаются на два такта, но реальный выигрыш наблюдается лишь при передаче блоков данных, хранящихся в одной и той же строке микросхемы. Если же программа часто обращается к разным областям памяти, переходя с одной строки ИМС на другую, преимущества метода теряются.

Пакетный режим – режим, при котором на запрос по конкретному адресу память возвращает пакет данных, хранящихся не только по этому адресу, но и по нескольким последующим адресам.

Разрядность ячейки памяти современных ВМ обычно равна одному байту. Если ширина шины данных равна четырем байтам, то одно обращение к памяти требует последовательного доступа к четырем смежным ячейкам – пакету. С учетом этого обстоятельства, в ИМС памяти часто используется модификация страничного режима, носящая название *группового* или *пакетного режима.* При его реализации адрес столбца заносится в ИМС только для первой ячейки пакета, а переход к очередному столбцу производится уже внутри микросхемы. Это позволяет для каждого пакета исключить три из четырех операций занесения в ИМС адреса столбца и тем самым еще более сократить среднее время доступа (рис. 22.9).

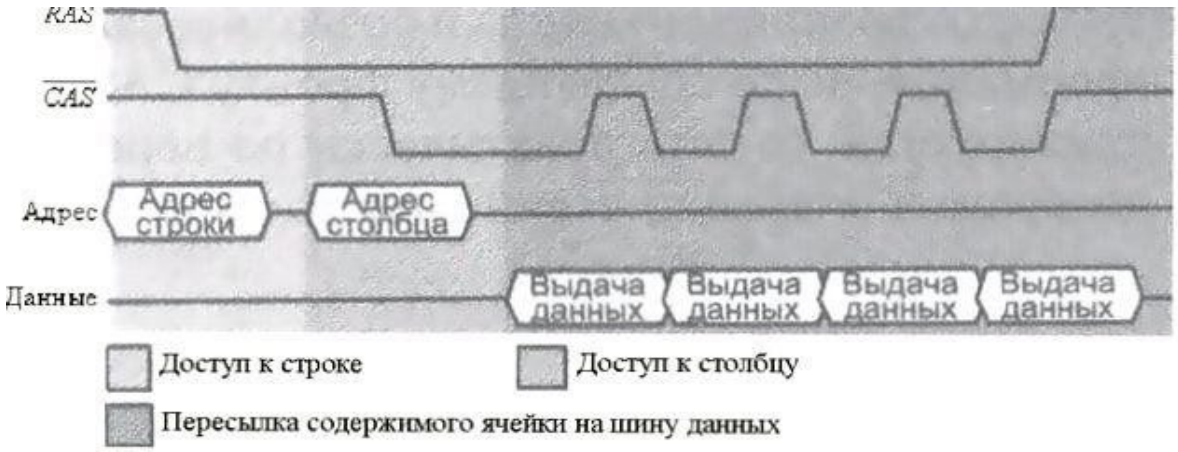


Рис. 22.9. Временная диаграмма операции чтения в пакетном режиме

В *конвейерном режиме* процесс разбивается на два этапа. Пока данные из предыдущего цикла чтения передаются на внешнюю шину, происходит запрос на следующую операцию чтения. Таким образом, два цикла чтения перекрываются во времени. Из-за усложнения схемы передачи данных на внешнюю шину время считывания увеличивается на один такт, и данные поступают на выход только в следующем такте, но такое запаздывание наблюдается лишь при первом чтении в последовательности операций считывания из памяти. Все последующие данные поступают на выход друг за другом, хотя и с запаздыванием на один такт относительно запроса на чтение. Так как циклы чтения перекрываются, микросхемы с конвейерным режимом могут использоваться при частотах шины, вдвое превышающих допустимую для ИМС с последовательным режимом чтения. На рис. 22.10. показана временная диаграмма операции чтения, в которой конвейерный режим сочетается с пакетным.

Режим удвоенной скорости. Важным этапом в дальнейшем развитии технологии микросхем синхронной памяти стал режим DDR (Double Data Rate) – удвоенная скорость передачи данных. Сущность метода заключается в передаче данных по обоим фронтам импульса синхронизации, то есть дважды за период. Таким образом, пропускная способность увеличивается в те же два раза.

Помимо упомянутых, используются и другие приемы повышения быстродействия ИМС памяти, такие как включение в состав микросхемы вспомогательной кэш-памяти и независимые тракты данных, позволяющие одновременно производить обмен с шиной данных и обращение к матрице ЗЭ и т.д.

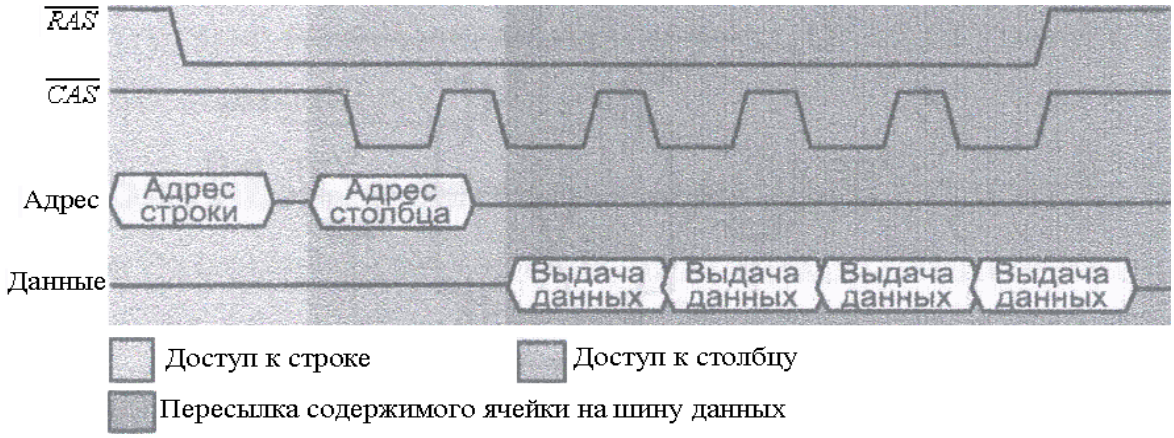


Рис. 22.10. Временная диаграмма операции чтения в пакетном режиме с конвейеризацией

Контрольные вопросы

1. Какие операции определяет понятие «обращение к ЗУ»?
2. Какие единицы измерения используются для указания емкости запоминающих устройств?
3. В чем отличие между временем доступа и периодом обращения к запоминающему устройству?
4. Чем вызвана необходимость построения системы памяти по иерархическому принципу?
5. Что включает в себя понятие «локальность по обращению»?
6. Благодаря чему среднее время доступа в иерархической системе памяти определяется быстродействующими видами ЗУ?
7. Что в иерархической системе памяти определяют термины «промах» и «попадание»?
8. На какие вопросы необходимо ответить, чтобы охарактеризовать определенный уровень иерархической памяти?
9. Какие виды запоминающих устройств может содержать основная память?
10. Охарактеризуйте возможные варианты построения блочной памяти.
11. Какие возможности по сокращению времени доступа к информации предоставляет блочная организация памяти?
12. Чем обусловлена эффективность расслоения памяти?
13. Какая топология запоминающих элементов лежит в основе организации полу-проводниковых ЗУ?
14. Какое минимальное количество линий должен содержать столбец ИМС памяти?
15. Поясните назначение управляющих сигналов в микросхеме памяти.
16. Чем отличаются страничный, быстрый страничный и пакетный режимы доступа к памяти?

Информация по теме:

Так как аппаратура передачи и приема данных работает с данными в дискретном виде (т.е. единицам и нулям данных соответствуют дискретные электрические сигналы), то при их передаче через аналоговый канал требуется преобразование дискретных данных в аналоговые (модуляция).

При приеме таких аналоговых данных необходимо обратное преобразование – демодуляция. Модуляция/демодуляция – процессы преобразования цифровой информации в аналоговые сигналы и наоборот. При модуляции информация представляется синусоидальным сигналом той частоты, которую хорошо передает канал передачи данных. К способам модуляции относятся: амплитудная модуляция; частотная модуляция; фазовая модуляция.

При передаче дискретных сигналов через цифровой канал передачи данных используется кодирование: потенциальное; импульсное. Потенциальное или импульсное кодирование применяется на каналах высокого качества, а модуляция на основе синусоидальных сигналов предпочтительнее в тех случаях, когда канал вносит сильные искажения в передаваемые сигналы.

При обмене данными между узлами вычислительных сетей используются три метода передачи данных: симплексная (однонаправленная) передача (телевидение, радио); полудуплексная (прием/передача информации осуществляется поочередно); дуплексная (двунаправленная), каждый узел одновременно передает и принимает.

коды включают в себя также коды проверки ошибок и другую информацию.

Существует три принципиально различные схемы коммутации в вычислительных сетях: коммутация каналов; коммутация пакетов; коммутация сообщений.

При цифровом кодировании дискретной информации применяют потенциальные и импульсные коды.

В потенциальных кодах для представления логических единиц и нулей используется только значение потенциала сигнала, а его перепады, формирующие законченные импульсы, во внимание не принимаются. Импульсные коды позволяют представить двоичные данные либо импульсами определенной полярности, либо частью импульса - перепадом потенциала определенного направления.

Для обнаружения искажений наиболее популярны методы, основанные на циклических избыточных кодах (CRC), которые выявляют многократные ошибки. Для восстановления кадров используется метод повторной передачи на основе квитанций. Этот метод работает по алгоритму с простоями источника, а также по алгоритму скользящего окна. Для повышения полезной скорости передачи данных в сетях применяется динамическая компрессия данных на основе различных алгоритмов. Коэффициент сжатия зависит от типа данных и применяемого алгоритма и может колебаться в пределах от 1:2 до 1:8.

Ссылки на литературные источники, приведенные в рабочей программе дисциплины: пп. 1, 2 основной и дополнительной литературы, интернет-источники.

Домашнее задание:
Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 с

Раздел 8. Устройства основной памяти.

1. Оперативные запоминающие устройства.
2. Постоянные запоминающие устройства.
3. Контрольные вопросы.

Занятие (лекция)

Большинство из применяемых в настоящее время типов микросхем оперативной памяти не в состоянии сохранять данные без внешнего источника энергии, то есть являются энергозависимыми. Несмотря на это они обладают рядом достоинств по сравнению с энергонезависимыми типами ОЗУ: большей емкостью, низким энергопотреблением, более высоким быстродействием и невысокой себестоимостью хранения единицы информации.

Энергозависимые ОЗУ можно подразделить на две основные подгруппы: динамическую память (DRAM – Dynamic Random Access Memory) и статическую память (SRAM – Static Random Access Memory).

Статические оперативные запоминающие устройства

В *статических ОЗУ* запоминающий элемент может хранить записанную информацию неограниченно долго (при наличии питающего напряжения). Роль запоминающего элемента в статическом ОЗУ исполняет триггер. Такой триггер представляет собой схему с двумя устойчивыми состояниями, обычно состоящую из шести транзисторов (рис. 23.1).

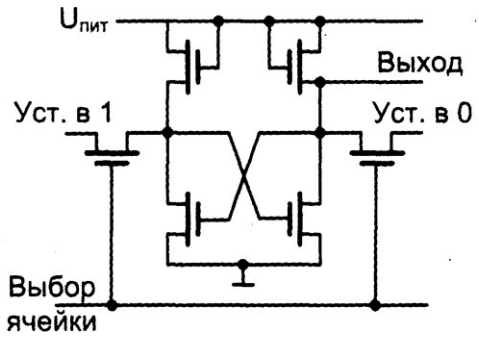


Рис. 23.1. Запоминающий элемент статического ОЗУ

Статические ОЗУ на настоящий момент – наиболее быстрый, правда, и наиболее дорогостоящий вид оперативной памяти. Основная сфера применения статических ОЗУ – кэш-память. Статические ОЗУ построены по синхронной схеме. В рамках группы *синхронных статических ОЗУ* выделяют ИМС типа SSRAM и более совершенные PB SRAM.

Как и в любой синхронной памяти, все события в SSRAM происходят с поступлением внешних тактовых импульсов. Отличительная особенность SSRAM – регистры, где фиксируется входная информация. В варианте с пакетным конвейерным доступом (PB SRAM – Pipelined Burst SRAM) реализована внутренняя конвейеризация, за счет которой скорость обмена пакетами данных возрастает примерно вдвое. Память данного типа хорошо работает при повышенных частотах системной шины.

Важным моментом, характеризующим SRAM, является технология записи. Известны два варианта записи: *стандартная* и *запаздывающая.* В стандартном режиме адрес и данные выставляются на соответствующие шины в одном и том же такте. В режиме запаздывающей записи данные для нее передаются в следующем такте после выбора адреса нужной ячейки, что напоминает режим конвейерного чтения, когда данные появляются на шине в следующем такте. Оба рассматриваемых варианта позволяют производить запись данных с частотой системной шины. Различия сказываются только при переключении между операциями чтения и записи.

В развитие идеи записи с запаздыванием компания IDT (Integrated Device Technology) предложила технологию, получившую название ZBT SRAM (Zero Bus Turnaround) – нулевое время переключения шины. Идея ее состоит в том, чтобы запись с запаздыванием производить с таким же интервалом, какой требуется для чтения. Так, если SRAM с конвейерным чтением требует три тактовых периода для чтения данных из ячейки, то данные для записи нужно передавать с таким же промедлением относительно адреса. В результате перекрывающиеся циклы чтения и записи идут один за другим, позволяя выполнять операции чтения/записи в каждом такте без каких-либо задержек.

Динамические оперативные запоминающие устройства

Запоминающий элемент *динамического ОЗУ* способен хранить информацию только в течение достаточно короткого промежутка времени, после которого информацию нужно восстанавливать заново, иначе она будет потеряна. Такой запоминающий элемент состоит из одного конденсатора и запирающего транзистора (рис. 23.2).

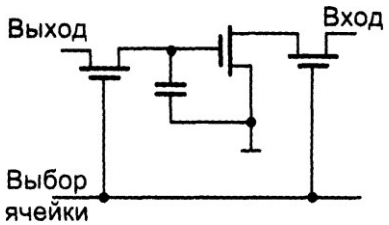


Рис. 23.2. Запоминающий элемент динамического ОЗУ

Наличие или отсутствие заряда в конденсаторе интерпретируется как

1 или 0, соответственно. Простота схемы позволяет достичь высокой плотности размещения запоминающих элементов и в итоге снизить стоимость. С другой стороны, считать значение, хранящееся в ячейке DRAM, невозможно, не изменив эту информацию, поэтому после операции считывания заряд нужно восстановить. Такое восстановление производится путем занесения считанной информации в специальный буфер, с последующей перезаписью из буфера на то же место, благодаря чему происходит перезаряд конденсаторов. Кроме того, накапливаемый на конденсаторе заряд из-за паразитных утечек со временем теряется. Среднее время утечки заряда ЗЭ динамической памяти составляет сотни или даже десятки миллисекунд, поэтому заряд необходимо успеть восстановить в течение данного отрезка времени, иначе хранящаяся информация будет утеряна. Периодическое восстановление заряда путем перезаписи информации называется *регенерацией* и осуществляется каждые 2÷8 мс. Операции разрядки-перезарядки могут занимать до 5 % от общего времени работы с памятью, снижая скорость работы DRAM. В среднем быстродействие DRAM в 8−16 раз ниже, чем у статической памяти.

В различных типах ИМС динамической памяти нашли применение три основных метода регенерации:

– одним сигналом RAS (ROR – RAS Only Refresh);

– сигналом CAS, предваряющим сигнал RAS (CBR – CAS Before RAS);

– автоматическая регенерация (SR – Self Refresh).

Регенерация одним RAS использовалась еще в первых микросхемах DRAM. На шину адреса выдается адрес регенерируемой строки, сопровождаемый сигналом RAS. При этом выбирается строка ячеек, и хранящиеся там данные поступают на внутренние цепи микросхемы, после чего записываются обратно. Так как сигнал CAS не появляется, цикл чтения/записи не начинается. В следующий раз на шину адреса подается адрес следующей строки и т.д., пока не восстановятся все ячейки, после чего цикл повторяется. К недостаткам метода можно отнести занятость шины адреса в момент регенерации, из-за чего доступ к другим устройствам ВМ блокирован.

Особенность метода CBR в том, что если в обычном цикле чтения/записи сигнал RAS всегда предшествует сигналу CAS, то при появлении первым сигнала CAS начинается специальный цикл регенерации. В этом случае адрес строки не передается, а микросхема использует свой внутренний счетчик, содержимое которого увеличивается на единицу при каждом очередном CBR-цикле. Режим позволяет регенерировать память, не занимая шину адреса, т.е. более эффективен.

Автоматическая регенерация памяти связана с энергосбережением, когда система переходит в режим «сна» и тактовый генератор перестает работать. При отсутствии внешних сигналов RAS и CAS обновление содержимого памяти методами ROR или CBR невозможно, и микросхема производит регенерацию самостоятельно, запуская собственный генератор, который тактирует внутренние цепи регенерации.

Область применения статической и динамической памяти определяется скоростью и стоимостью. Главным преимуществом SRAM является более высокое быстродействие, однако из-за малой емкости микросхем и высокой стоимости применение статической памяти, как правило, ограничено относительно небольшой по емкости кэш-памятью.

Динамической памяти в вычислительной машине значительно больше, чем статической, поскольку именно DRAM используется в качестве основной памяти вычислительной машины. Как и SRAM, динамическая память состоит из ядра (массива ЗЭ) и интерфейсной логики (буферных регистров, усилителей чтения данных, схемы регенерации и др.). Практически у всех видов DRAM ядро организовано практически одинаково. Главные различия связаны с интерфейсной логикой, причем различия эти обусловлены также и областью применения микросхем – помимо основной памяти ВМ, ИМС динамической памяти входят, например, в состав видеоадаптеров. Чтобы оценить различия между видами DRAM, предварительно остановимся на «классическом» алгоритме работы с динамической памятью. Для этого воспользуемся рис. 22.6.

Адрес ячейки DRAM передается в микросхему за два шага – вначале адрес столбца, а затем строки. Для указания, какая именно часть адреса передается в определенный момент, служат два вспомогательных сигнала RAS и CAS. При обращении к ячейке памяти на шину адреса выставляется адрес строки. После стабилизации процессов на шине подается сигнал RAS, и адрес записывается во внутренний регистр микросхемы памяти. Затем на шину адреса выставляется адрес столбца и выдается сигнал CAS. В зависимости от состояния линии WE производится чтение данных из ячейки или их запись в ячейку (перед записью данные должны быть помещены на шину данных). Интервал между установкой адреса и выдачей сигнала RAS (или CAS) оговаривается техническими характеристиками микросхемы, но обычно адрес выставляется в одном такте системной шины, а управляющий сигнал – в следующем. Таким образом, для чтения или записи одной ячейки динамического ОЗУ требуется пять тактов, в которых происходит соответственно: выдача адреса строки, выдача сигнала RAS, выдача адреса столбца, выдача сигнала CAS, выполнение операции чтения/записи (в статической памяти процедура занимает лишь от двух до трех тактов).

Следует также помнить о необходимости регенерации данных. Но наряду с естественным разрядом конденсатора ЗЭ со временем к потере заряда приводит также считывание данных из DRAM, поэтому после каждой операции чтения данные должны быть восстановлены. Это достигается за счет повторной записи тех же данных сразу после чтения. При считывании информации из одной ячейки фактически выдаются данные сразу всей выбранной строки, но используются только те, которые находятся в интересующем столбце, а все остальные игнорируются. Таким образом, операция чтения из одной ячейки приводит к разрушению данных всей строки и их нужно восстанавливать. Регенерация данных после чтения выполняется автоматически интерфейсной логикой микросхемы, и происходит это сразу же после считывания строки.

Асинхронные динамические ОЗУ. Работа таких микросхем не привязана жестко к тактовым импульсам. В течение достаточно длительного периода времени основная память ВМ строилась на базе микросхем ОЗУ асинхронного типа. На этом этапе микросхемы постоянно совершенствовались, главным образом, за счет перехода к более эффективным режимам чтения. На смену «классическим» DRAM с последовательным режимом чтения пришли микросхемы с быстрым постраничным режимом (FPM DRAM). Достаточно быстро их сменили микросхемы типа EDO DRAM с усовершенствованным страничным (гиперстраничным) режимом. Линия асинхронных динамических ОЗУ фактически завершилась микросхемами типа BEDO DRAM, в которых сочетались пакетный и конвейерный режим чтения. Несмотря на достаточно высокие скоростные характеристики BEDO и этим микросхемам свойственен общий недостаток асинхронной схемы – дополнительные затраты времени на взаимодействие микросхем памяти и контроллера.

Синхронные динамические ОЗУ. В синхронных DRAM обмен информацией синхронизируется внешними тактовыми сигналами и происходит в строго определенные моменты времени, что позволяет взять все от пропускной способности шины «процессор-память» и избежать циклов ожидания. Адресная и управляющая информация фиксируется в микросхеме памяти, после чего ответная реакция микросхемы произойдет через четко определенное число тактовых импульсов, и это время процессор может использовать для других действий, не связанных с обращением к памяти.

Для обозначения микросхем «обычных» синхронных динамических ОЗУ используется аббревиатура SDRAM. В настоящее время наиболее распространенным типом динамической памяти персональных ВМ являются микросхемы, реализующие технологию DDR SDRAM (SDRAM с удвоенной скоростью передачи данных). Основная особенность технологии в том, что передача данных синхронизируется как передним, так и задним фронтом тактового импульса. Таким образом, при сохранении тактовой частоты шины памяти запрошенные данные передаются вдвое быстрее (рис. 23.3) – по два пакета за один тактовый период (память работает в пакетном режиме). Как следствие, получается двукратное увеличение пропускной способности шины памяти.

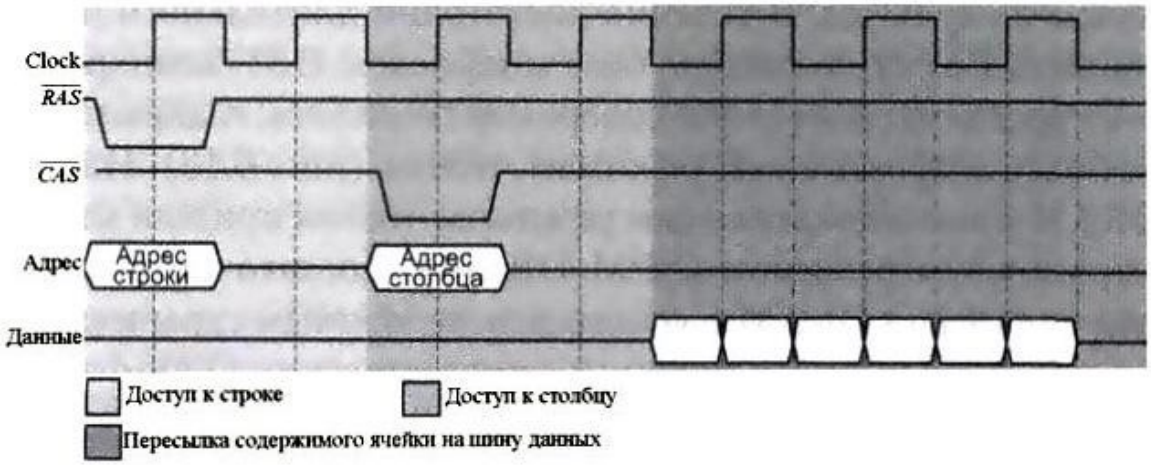


Рис. 23.3. Иллюстрация идей синхронизации по обоим фронтам тактового импульса

С момента появления технологии сменилось несколько поколений DDR (DDR, DDR2, DDR3). С каждым следующим поколением увеличивается рабочая частота микросхем и снижается потребляемая мощность за счет снижения рабочего напряжения.

Обозначение микросхем типа DDR имеет вид *DDRx-yyyy*, где *x* – поколение DDR, а *yyyy* – эффективная частота DDR, которая вдвое больше реальной максимальной тактовой частоты. Например, микросхемы DDR2-800 могут работать на частоте шины памяти 400 МГц. Следует иметь в виду, что указанное число отражает лишь максимальное значение частоты, на которой может работать микросхема, но «автоматической» подстройки частоты не происходит. Тактовая частота шины памяти задается контроллером памяти, внешним по отношению к микросхеме памяти.

Приводимые в обозначении числа определяют лишь теоретический максимум, который никогда не может быть достигнут, поскольку предполагается, что микросхема в каждом тактовом периоде пересылает в контроллер памяти исключительно Данные. Однако некоторые периоды не могут быть использованы для передачи Данных, поскольку в эти периоды контроллер и память обмениваются служебной информацией, обеспечивающей их взаимодействие.

В упрощенном варианте структуру DDR SDRAM можно представить в виде, показанном на рис. 23.4.

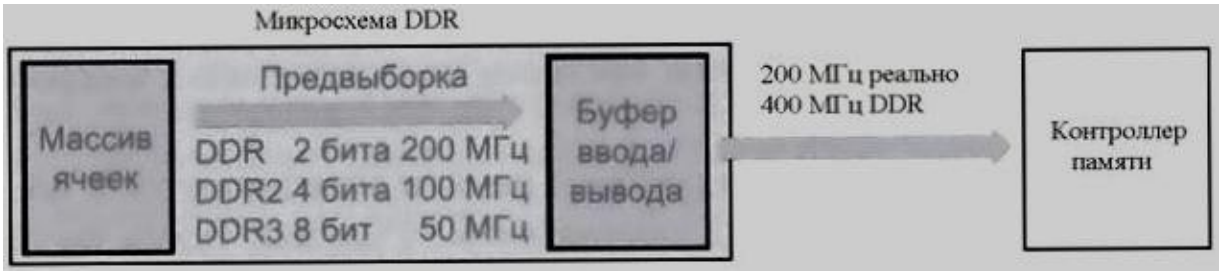


Рис. 23.4. Структура DDR SDRAM с иллюстрацией идеи
n-битовой предвыборки

Чтобы обеспечить передачу данных дважды за такт, используется специальная архитектура с предвыборкой, в которой присутствует буфер ввода/вывода (буфер предвыборки). В каждом тактовом периоде шины из массива ячеек в этот буфер передаются два бита данных. Это называется 2*n*-битовой предвыборкой. В DDR2 внутренняя шина между массивом ячеек и буфером увеличена до 4 битов, а в DDR3 – до 8 битов.

Многоканальная динамическая память

В условиях непрерывно возрастающей скорости и производи- тельности процессоров память все в большей степени становится «узким местом» вычислительной машины. Это происходит из-за того, что процессор работает быстрее основной памяти и вынужден ожидать поступления данных из основной памяти. Многоканальная архитектура – это технология, теоретически позволяющая увеличить пропускную способность тракта передачи данных между памятью и *контроллером памяти*.

Двухканальная архитектура требует использования двухканального контроллера (или двух независимых одноканальных) и наличия двух или более модулей памяти, построенных на микросхемах типа DDR, DDR2,

либо DDR3. Модули памяти устанавливаются в разъемы материнской платы, каждый из которых обозначен своим цветом. Отдельные каналы позволяют каждому модулю независимо обращаться к контроллеру, тем самым увеличивается полоса пропускания. Модули не обязательно должны быть идентичными. При различных модулях общая скорость памяти будет определяться скоростью более медленного модуля.

В последних разработках компании Intel (процессоры Core 17) и AMD (Sao Paulo) заложено использование трехканальной памяти на микросхемах типа DDR3. Идея такой памяти аналогична двухканальной, но число каналов между ОП и контроллером памяти увеличено до трех, с соответствующим увеличением теоретической пропускной способности. Перспективные разработки AMD ориентированы на четырехканальную память на микросхемах DDR3.

Оперативные запоминающие устройства для видеоадаптеров

Использование памяти в видеоадаптерах имеет свою специфику, и для реализации дополнительных требований прибегают к несколько иным типам микросхем. Так, при создании динамичных изображений часто достаточно просто изменить расположение уже хранящейся в видеопамяти информации. Вместо того чтобы многократно пересылать по шине одни и те же данные, лишь несколько изменив их расположение, выгоднее заставить микросхему памяти переместить уже хранящиеся в ней данные из одной области ядра в другую. На ИМС памяти можно также возложить операции по изменению цвета точек изображения.

В современных графических картах ведущих производителей применяются микросхемы типа VRAM. Также широкое распространение получили микросхемы динамической памяти, в основе которых лежит идеология DDR: GDDR2, GDDR3, GDDR4, GDDR5.

Микросхемы VRAM. ОЗУ типа VRAM (Video RAM) отличается высокой производительностью и предназначено для мощных графических систем. При разработке ставилась задача обеспечить постоянный поток данных при обновлении изображения на экране. Для типовых значений разрешения и частоты обновления изображения интенсивность потока данных приближается к 200 Мбит/с. В таких условиях процессору трудно получить доступ к видеопамяти для чтения или записи. Чтобы разрешить эту проблему, в микросхеме сделаны существенные архитектурные изменения, позволяющие обособить обмен между процессором и ядром VRAM (для чтения/записи информации) и операции по выдаче информации на схему формирования видеосигнала.

GDDR (Graphics Double Data Rate) – семейство микросхем видеопамяти с идеологией DDR. Эволюцию семейства отражает ряд: GDDR2, GDDR3, GDDR4, GDDR5. От обычных микросхем типа DDR это семейство отличают некоторые особенности, обусловленные спецификой применения. Так, в основе GDDR3 лежит та же технологическая база, что и в DDR2, однако в видеоварианте снижены требования к потребляемой мощности и рассеиваемому теплу. Это и ряд других усовершенствований привели к повышению производительности модулей памяти и упрощению системы охлаждения. В свою очередь, в GDDR4 реализована 8-битовая предвыборка взамен 4-битовой в GDDR3. GDDR5 отличается от своего предшественника (GDDR4) тем, что в микросхеме реализована технология DDR3 (а не DDR2). Это изменение позволило использовать графические карты с GDDR5 в приложениях, особо критичных к полосе пропускания.

Постоянные запоминающие устройства

Постоянные запоминающие устройства хранят информацию при отсутствии питающего напряжения. Микросхемы ПЗУ также построены по принципу матричной структуры накопителя, где в узлах расположены перемычки в виде проводников, полупроводниковых диодов или транзисторов, одним концом подключенные к адресной линии, а другим – к разрядной линии считывания. В такой матрице наличие перемычки может означать 1, а ее отсутствие – 0. В некоторых типах ПЗУ элемент, расположенный на перемычке, исполняет роль конденсатора. Тогда заряженное состояние конденсатора означает 1, а разряженное – 0.

Основным режимом работы ПЗУ является считывание информации, которое мало отличается от аналогичной операции в ОЗУ как по организации, так и по длительности. Именно это обстоятельство подчеркивает общепризнанное название постоянных запоминающих устройств – ROM (Read-Only Memory – память только для чтения). В то же время запись в ПЗУ по сравнению с чтением обычно сложнее и связана с большими затратами времени и энергии. Занесение информации в ПЗУ называют программированием или «прошивкой». Полупроводниковые микросхемы ПЗУ по возможностям и способу программирования разделяют на:

– программируемые при изготовлении;

– однократно программируемые после изготовления;

– многократно программируемые.

С появлением многократно программируемых ПЗУ популярность первых двух групп существенно снизилась. Процедура программирования

таких ПЗУ обычно предполагает два этапа: сначала производится стирание содержимого всех или части ячеек, а затем производите запись новой информации.

Группу *ПЗУ, программируемых при изготовлении*, образуют так называемые масочные устройства. Занесение информации в масочные ПЗУ составляет часть производственного процесса и заключается в подключении или не подключении запоминающего элемента к разрядной линии считывания. В зависимости от этого из ЗЭ будет всегда извлекаться

1 или 0. В роли перемычки выступает транзистор, расположенный на пересечении адресной и разрядной линий. Какие именно ЗЭ должны быть подключены к выходной линии, определяет маска, «закрывающая» определенные участки кристалла.

Создание масок для ROM оправдано при производстве большого числа копий. Если требуется относительно небольшое количество микросхем с данной информацией, разумной альтернативой являются *однократно программируемые ПЗУ*. Такие микросхемы обозначают аббревиатурой PROM (Programmable ROM – *программируемые ПЗУ).* Информация в них может быть записана только однократно. Занесение информации в PROM производится электрически, путем пережигания отдельных перемычек, и может быть выполнено поставщиком или потребителем спустя какое-то время после изготовления микросхемы. Для записи информации требуется специальное оборудование – программаторы. Основными недостатками данного вида ПЗУ были большой процент брака и необходимость специальной термической тренировки после программирования, без которой надежность хранения данных была невысокой.

23.2.1. Многократно программируемые ПЗУ

Процедура программирования таких ПЗУ обычно предполагает два этапа: сначала производится стирание содержимого всех или части ячеек, а затем производится запись новой информации.

В этом классе постоянных запоминающих устройств выделяют несколько групп.

– EPROM (Erasable Programmable ROM – стираемые программируемые ПЗУ;

– EEPROM (Electrically Erasable Programmable ROM – электрически стираемые программируемые ПЗУ);

– флэш-память;

– PCM (Phase Change Memory) – фазовая память.

Микросхемы EPROM. В EPROM запись информации производится электрическими сигналами, так же, как в PROM, однако перед операцией записи содержимое всех ячеек должно быть приведено к одинаковому состоянию (стерто) путем воздействия на микросхему ультрафиолетовым облучением. Процесс стирания может выполняться многократно. Каждое стирание занимает порядка 20 мин. Данные хранятся в виде зарядов плавающих затворов МОП-транзисторов, играющих роль конденсаторов с очень малой утечкой заряда. Заряженный ЗЭ соответствует логическому нулю, а разряженный – логической единице. Цикл программирования занимает нескольких сотен миллисекунд. Время считывания близко к показателям ROM и DRAM.

Микросхемы EEPROM. Более привлекательным вариантом многократно программируемой памяти является электрически стираемая программируемая постоянная память EEPROM. Стирание и запись информации в эту память производятся по байтам, причем стирание – не отдельный процесс, а лишь этап, происходящий автоматически при записи. Операция записи занимает существенно больше времени, чем считывание – несколько сотен микросекунд на байт. В микросхеме используется тот же принцип хранения информации, что и в EPROM. Программирование EPROM не требует специального программатора и реализуется средствами самой микросхемы.

Выпускаются два варианта микросхем: с последовательным и параллельным доступом. На долю последних приходится 90 % всех выпускаемых ИМС. В EEPROM с доступом по последовательному каналу адреса, данные и управляющие команды передаются по одному проводу и синхронизируются импульсами на тактовом входе. Преимуществом такого подхода является малые габариты и минимальное число линий ввода/вывода, недостатком – большое время доступа.

В целом, EEPROM дороже, чем EPROM, а микросхемы имеют менее плотную упаковку ячеек, т.е. меньшую ёмкость.

Флэш-память. Флэш-память – это энергонезависимая перепрограмми- руемая полупроводниковая память. Флэш-память во многом похожа на EEPROM, но использует особую технологию построения запоминаю- щего элемента. Логически запоминающий элемент представлен элементом ИЛИ-НЕ (NOR), либо И-НЕ (NAND).

Запоминающим элементом флэш-памяти служит полевой транзистор, у которого под управляющим затвором располагается так называемый плавающий затвор, который может хранить заряд в виде электронов (рис. 23.5), причем этот заряд, даже при отсутствии питающего напряжения, может сохраняться длительное время (несколько лет).

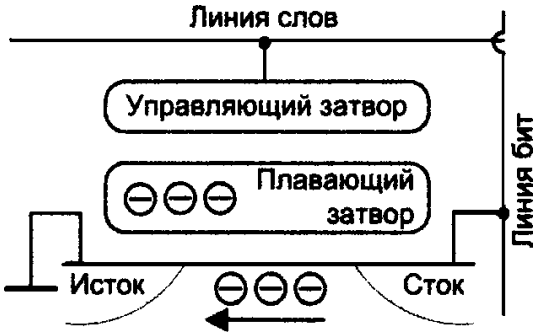


Рис. 23.5. Запоминающий элемент флэш-памяти

Состояние запоминающего элемента зависит от количества электронов в плавающем затворе, которое при чтении может быть измерено косвенно через величину порогового напряжения из ЗЭ. Если в плавающем затворе менее 5000 электронов, считается, что в ЗЭ записана логическая единица. Если заряд более 30 000 электронов – логический ноль.

Записи информации в ЗЭ предшествует стирание заряда с плавающего затвора (запись в ЗЭ логической 1). Для этого на исток подается высокое положительное, а на управляющий затвор – высокое отрицательное напряжение. В результате электроны с плавающего затвора

«переходят» на исток (рис. 23.6, *а*).

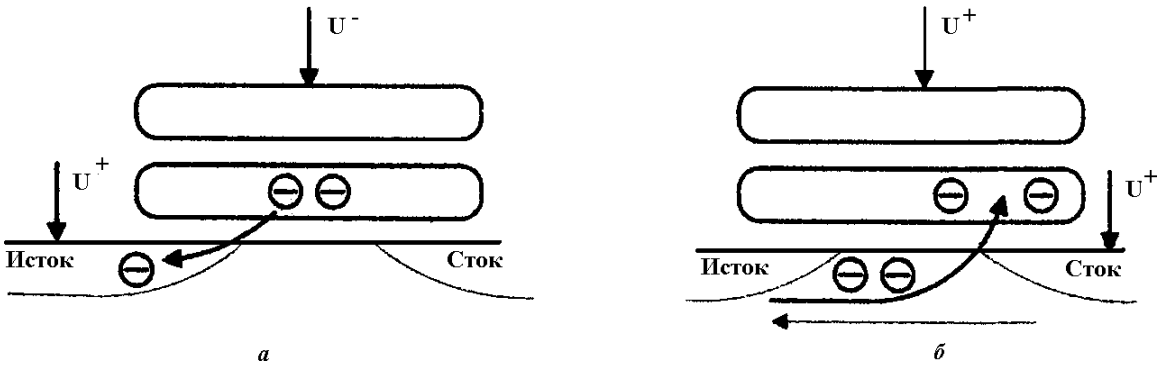


Рис. 23.6. Стирание и программирование (запись) информации:

а – стирание информации; *б* – программирование

Программирование заключается в записи логических нулей, т.е. заряде плавающего затвора тех запоминающих элементов, где должен храниться логический 0. Это обеспечивается подачей на сток высокого положительного напряжения, а на управляющий затвор вдвое большего положительного напряжения. В этих условиях электроны из канала между истоком и стоком переходят на плавающий затвор (рис. 23.6, *б*).

Полностью содержимое флэш-памяти может быть очищено за одну или несколько секунд, что значительно быстрее, чем у EEPROM. Программирование (запись) байта занимает время порядка 10 мкс, а время доступа при чтении составляет 35÷200 нс.

Флэш-память с NOR ячейками позволяет работать с отдельными байтами, аналогично EEPROM. Во флэш-памяти типа NAND ячейки группируются в небольшие блоки, и в качестве наименьшей единицы информации при записи выступает блок. В силу этих различий флэш- память с ячейками NOR выгодно использовать в случае произвольного доступа, например для хранения BIOS в персональных компьютерах. Память с ячейками NAND более подходит при работе с большими массивами информации, например в твердотельных дисках.

Независимо от типа ячеек флэш-память имеет ограничение по количеству циклов перезаписи (в лучшем случае – несколько миллионов), что не позволяет использовать ее как энергонезависимое ОЗУ. Кроме того, флэш-память уступает микросхемам ОЗУ по быстродействию. *Фазовая память PCM*. Носителем информации служат микроскопи- ческие частицы халькогенидного стекла, которое может находиться в одном из двух состояний. Эти состояния (фазы) различаются по оптическим и электрическим характеристикам. Под воздействием тепла материал может переходить из одного состояния в другое. В оптических запоминающих устройствах такой нагрев достигается с помощью луча лазера. В случае PCM нагрев достигается пропусканием через материал электрического тока, причем состояние, в которое переходит нагреваемый участок, зависит от величины тока, приложенного напряжения и длительности нагревания. В обычном (холодном) состоянии материал представляет собой аморфную стеклообразную структуру с высоким электрическим сопротивлением. При воздействии высокой температуры (до 600 градусов по Цельсию) материал кристаллизуется и будет обладать очень низким сопротивлением. Высокое сопротивление в аморфной фазе используется для представления двоичного «0», а низкое сопротивление в кристаллической фазе – «1». На практике фазовый переход может осуществляться за 16 нс. В целом, PCM обеспечивает стирание содержимого ячеек почти в 10 раз быстрее флэш-памяти, а темп перезаписи превышает аналогичный показатель для флэш-памяти в семь раз.

PCM выгодно использовать там, где требуется быстрая запись информации, поскольку технология позволяет изменять значение отдельных битов без предварительного стирания целого блока ячеек. Кроме того, запоминающие элементы PCM обладают высоким

быстродействием. Еще одно преимущество PCM перед флэш-памятью – количество циклов перезаписи без деградации микросхемы достигает 100 миллионов.

Энергонезависимые оперативные запоминающие устройства

Под понятие *энергонезависимое ОЗУ* (NVRAM – Non-Volatile RAM) подпадает несколько типов памяти. От перепрограммируемых постоянных ЗУ их отличает отсутствие этапа стирания, предваряющего запись новой информации, поэтому вместо термина «программирование» для них употребляют стандартный термин «запись».

Микросхемы BBSRAM. К рассматриваемой группе относятся обычные статические ОЗУ со встроенным литиевым аккумулятором и усиленной защитой от искажения информации в момент включения и отключения внешнего питания. Для их обозначения применяют аббревиатуру BBSRAM (Battery-Back SRAM).

Микросхемы NVRAM. Разработана компанией Simtec. Особенность ее в том, что в одном корпусе объединены статическое ОЗУ и перепрог- раммируемая постоянная память типа EEPROM. При включении питания данные копируются из EEPROM в SRAM, а при выключении – автоматически перезаписываются из SRAM в EEPROM. Благодаря такому приему данный вид памяти можно считать энергонезависимым.

Микросхемы FRAM. FRAM (Ferroelectric RAM – ферроэлектрическая память) разработана компанией Ramtron. По быстродействию данное ЗУ несколько уступает динамическим ОЗУ и пока рассматривается лишь как альтернатива флэш-памяти. Причисление FRAM к оперативным ЗУ обусловлено отсутствием перед записью явно выраженного цикла стирания информации.

Запоминающий элемент FRAM похож на ЗЭ динамического ОЗУ, то есть состоит из конденсатора и транзистора. Отличие заключено в диэлектрических свойствах материала между обкладками конденсатора. В FRAM этот материал обладает большой диэлектрической постоянной и может быть поляризован с помощью электрического поля. Поляризация сохраняется вплоть до ее изменения противоположно направленным электрическим полем, что и обеспечивает энергонезависимость данного вида памяти. Данные считываются за счет воздействия на конденсатор электрического поля. Величина возникающего при этом тока зависит от того, изменяет ли приложенное поле направление поляризации на противоположное или нет, что может быть зафиксировано усилителями

считывания. В процессе считывания содержимое ЗЭ разрушается и должно быть восстановлено путем повторной записи, то есть, как и DRAM, данный тип ЗУ требует регенерации. Количество циклов перезаписи для FRAM обычно составляет 10 млрд.

Главное достоинство данной технологии в значительно более высокой скорости записи по сравнению с EEPROM. В то же время относительная простота ЗЭ позволяет добиться высокой плотности размещения элементов на кристалле, сопоставимой с DRAM.

Контрольные вопросы

1. Чем обусловлена необходимость регенерации содержимого динамических ОЗУ?
2. Охарактеризуйте основные сферы применения статических и динамических ОЗУ.
3. Какое влияние на асинхронный режим работы памяти оказывает синхронный характер работы контроллера памяти?
4. Какой вид ПЗУ обладает наиболее высокой скоростью перепрограммирования?
5. Какими методами обеспечивается энергозависимость ОЗУ?
6. В чем состоит различие между режимами стандартной и запаздывающей записи в статических ОЗУ?
7. В чем проявляется специфика ОЗУ, предназначенных для видеосистем?

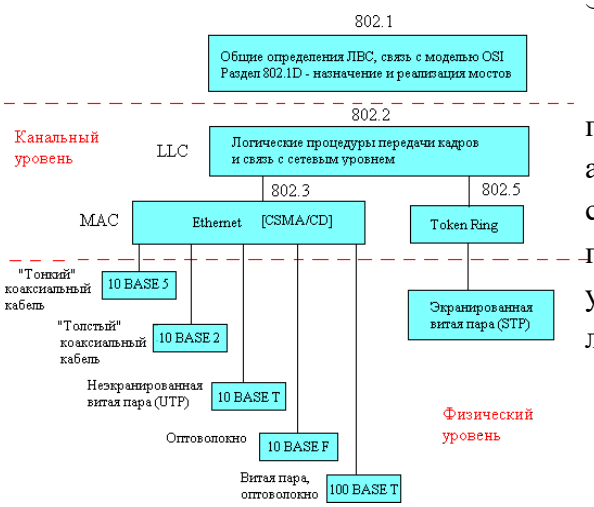
Информация по теме:

В 1980 году в институте IEEE был организован комитет 802 по стандартизации локальных сетей. Результатом его работы стало семейство стандартов IEEE 802.x, содержащих рекомендации по проектированию нижних уровней локальных сетей.

Стандарты IEEE 802 имеют достаточно четкую структуру, приведенную на рисунке.

Специфика локальных сетей нашла свое отражение в разделении канального уровня на два подуровня: логической переда-чи данных (Logical Link Control – LLC),управления доступом к среде (MediaAccess Control,MAC).

Уровень MAC поя-вился из-за существованияв ЛВС разделяемой средыпередачи данных. Этот уровень обеспечивает корректное совместное использование общей среды, предоставляя ее в соответствии с определенным алгоритмом в распоряжении той или иной станции сети. После того как доступ к среде получен, ею может пользоваться более высокий уровень – LLC, организующий передачу логических единиц данных с различным уровнем



качества транспортных услуг. В современных ЛВС получили распространение несколько протоколов уровня MAC, реализующих различные алгоритмы доступа к пере- дающей среде. Эти протоколы полностью определяют специфику отдельных технологий.

Стандарты 802.3, 802.4, 802.5 и 802.12 описывают технологии локальных сетей, полученные в результате улучшений фирменных технологий, легших в их основу. К други известным стандартам относятся:

802.6 – Metropolitan Area Network , MAN – сети мегаполисов;

802.8 – Fiber Optic Technical Advisory Group – техническая консультационная груп-па по волоконно-оптическому кабелю;

802.9 – Integrated Voice and data Networks – интегрированная среда передачи голосаи данных;

802.10 – Network Security – сетевая безопасность;

802.11 – Wireless Networks – беспроводные сети.

Ссылки на литературные источники, приведенные в рабочей программе дисциплины: пп. 1, 2 основной литературы, интернет-источники.

Домашнее задание:
Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 с

Раздел 9. Принципы организации работы процессоров.

1. Назначение и структура процессора.
2. Система команд. Форматы команд и способы адресации.
3. Система прерываний и приостановок, состояние процессора.
4. Режимы работы процессора
5. Контрольные вопросы.

Занятие (лекция)

Современные процессоры строятся из одной или нескольких интегральных схем. Как правило, в персональном компьютере ЦП представляет собой одну микросхему, в которой используются все возможности современной полупроводниковой технологии. Такую микросхему принято называть *микропроцессором*. В больших компьютерах процессоры могут быть выполнены из нескольких интегральных схем.

В компьютерах используются процессоры нескольких типов. Наиболее распространены центральные, специализированные (например, процессоры для цифровой обработки сигналов или обработки графической информации), процессоры ввода-вывода, передачи данных и т.п.

Центральный процессор служит для обработки данных и управления всей компьютерной системой в целом. Он позволяет обрабатывать данные с фиксированной и плавающей точками, поля переменной длины, а иногда и десятичные данные, назначать приоритеты доступа и управлять различными видами памяти.

Процессор представляет собой совокупность АЛУ и устройства управления. В состав ЦП входят арифметическое устройство, счетчик команд, регистр команд, регистры данных, блок микропрограммного или аппаратного управления, регистры общего назначения и ряд узлов, предназначенных для связи с оперативной памятью и другими устройствами компьютера, а также для ускорения выполнения операций. Помимо этого в состав ЦП входят часы астрономического времени, таймер и ряд других «системных» средств.

В одном кристалле с процессором размещается и кэш-память. Для понимания принципов работы компьютера нужно разделять понятия

«исполнение» и «выполняемые функции», поэтому процессором по-прежнему будем называть совокупность АЛУ и устройства управления.

Система команд. Форматы команд и способы адресации

Для получения результата вычислений компьютер выполняет *машинную программу*, т.е. последовательность команд, в виде которой записан алгоритм обработки. *Команда* компьютера представляет собой двоичный код, определяющий выполняемую операцию и необходимые для

этого данные, или *операнды*. Большинство команд содержит *адрес*, т.е. номер регистра или ячейки памяти, где сохраняется результат выполнения операции. Для выполнения программы нужно также знать адрес следующей исполняемой команды.

Обычно различают команды:

– арифметические операции над числами с фиксированной точкой;

– логические операции;

– арифметические операции над числами с плавающей точкой;

– операции ввода/вывода;

– команды управления (например, управления циклами, условные и безусловные переходы) и т.д.

Каждый тип компьютера обладает собственной *системой команд*, т.е. в нем существует аппаратура или память микропрограмм, призванная вырабатывать управляющие сигналы с целью реализации командных операций. Для исполнения конкретной программы необходим компьютер, способный выполнять команды, составляющие эту программу. При разработке новых компьютеров стремятся сохранить их преемственность; для этого компьютеры изготавливают программно совместимыми, т.е. имеющими одинаковые системы команд. Однако во многих случаях систему команд «расширяют», т.е. добавляют дополнительные операции, сохраняя при этом формат команд. Эго значит, что новая машина может выполнять все программы, составленные для прежних компьютеров, но программы, в которых используются дополнительные команды, не могут выполняться компьютерами старых моделей. Такую совместимость называют *обратной*.

Команда представляет собой слово, содержащее код выполняемой операции и адреса операндов. Код команды включает в себя несколько полей. Команда состоит из операционной и адресной частей. В *операционной* части размещается код операции, а в *адресной* – адреса операндов, т.е. информация о местонахождении обрабатываемых данных и получаемого результата.

Формат команды – это структура полей ее кода с указанием номеров разрядов, определяющих границы полей. В универсальных ВМ код операции в команде составляет 8 разрядов, а число различных операций – не более 256. Остальные разряды отводятся под адреса операндов. Команды представляют собой слово размером 16, 32 или 48 разрядов.

В различных машинах число адресных полей в команде может составлять от одного до четырех. В четырехадресной команд первое поле – код операции. Он предназначен для кодирования выполняемой операции.

На основе данного поля формируются управляющие сигналы для выполнения соответствующих действий. Затем располагаются четыре адресных поля: $A1$ – адрес первого операнда, $A2$ – адрес второго операнда, $A3$ – адрес ячейки памяти, отведенной для записи результата операции, $A4$ – адрес ячейки, где находится следующая команда. Но такая четырех- адресная команда занимает слишком много места в памяти компьютера, поэтому они в настоящее время не находят применения.

Наибольшее распространение имеют:

– трёхадресные команды (рис. 9.3) характерны для компьютеров с сокращенным набором команд. Первый и второй адреса такой команды указывают месторасположение операндов, в третий адрес (ячейку памяти) заносится результат операции. Для определения адреса следующей выполняемой команд служит счетчик команд (СК), к содержимому которого после выполнения любой команды добавляется ее длина в байтах. Для перехода к выполнению команды, которая занимает не следующую по порядку ячейку памяти, в машине предусматривают специальные команды переходов. Трехадресные команды используются в RISC-компьютерах; в них операнды размещают в регистрах общего назначения, число которых может достигать 256.

– двухадресные команды (рис. 9.4); оба операнда находятся в регистрах, а результат выполнения операции также записывается в регистр. Такую команду принято называть командой RR-типа (регистр- регистр). Если один операнд находится в регистре, а второй в ячейке памяти, адрес которой индексируется, то такая команда относится к RX-типу. Команда, второй операнд которой находится в ячейке памяти без индексации, а первый в регистре, носит название RS-типа.

– одноадресные команды (рис. 9.5) содержат только один адрес, а поскольку обычно в арифметической операции используется два операнда, то второй уже находится в одном из регистров процессора. Результат операции всегда сохраняется в выходном регистре процессора, который называют *аккумулятором*. Для сложения двух чисел нужно выполнить три одноадресных команды: поместить первый операнд из памяти в регистр процессора, передать второй операнд из памяти и сложить его с первым, а затем сохранить результат в памяти. Однако на практике при выполнении программы никогда не приходится использовать все три команды. В регистре процессора сохраняется один из операндов, полученный при выполнении предыдущей операции (т.е. первая команда не нужна), а результат сложения будет использован следующей командой (т.е. его не нужно сохранять в памяти).

В действительности в адресных полях находятся не сами адреса, а информация, позволяющая их определить. Способ определения исполнительного адреса $A_{\text{исп}}$ (т.е. адреса ячейки памяти, где находится операнд или команда) по адресу в коде команды A_k называется *способом адресации*.

Для указания способа адресации в некоторых системах команд используют специальное поле – способ адресации (рис. 9.2); это позволяет выполнять одну и ту же команду с любым предусмотренным способом адресации, что увеличивает длину команды. В настоящее время применяются следующие способы адресации:

– прямая (или абсолютная) адресация (рис. 10.2);
– косвенная адресация (рис. 10.3);
– регистровая адресация (рис. 10.4);
– косвенная регистровая адресация (рис. 10.5);
– адресация со смещением (рис. 10.6);
– относительная адресация (рис. 10.7);
– базовая регистровая адресация (рис. 10.8);
– индексная адресация (рис. 10.9);
– страничная адресация (рис. 10.10)
– блочная адресация (п. 10.1.11).

Все виды адресации в компьютере IBM PC определяют место расположения и способ нахождения адреса второго операнда; первый операнд всегда находится в одном из регистров. Для компьютеров IBM PC многочисленные способы адресации достались «по наследству» от ранних моделей микропроцессоров, когда разрядность была ограничена.

Выбору реализуемой компьютером системы команд должно уделяться самое серьезное внимание. Каждая аппаратно реализуемая операция, входящая в систему команд, выполняется быстрее, чем аналогичная операция, не входящая в систему команд, так как она реализуемая в виде подпрограммы. На первый взгляд из этого следует, что увеличение аппаратно реализуемых операций, т.е. расширение системы команд, может привести к повышению быстродействия машины. Этотвывод положен в основу концепции архитектуры компьютера срасширенным набором команд – CISC-архитектура. Однако расширение системы команд может приводить и к обратным результатам. Именно это обстоятельство послужило толчком для разработки машин с сокращенным набором команд – RISC-архитектура.

Система прерываний и приостановок, состояние процессора

Во время выполнения какой-либо программы компьютером могут возникнуть события, требующие его немедленной реакции – необходимость незамедлительного решения другой задачи, переполнение разрядной сетки, программный или аппаратный сбой, окончание предусмотренного интервала времени и т.д. Все это события требуют переключения компьютера на другую программу. Переход к другой программе осуществляется посредством *системы прерываний*.

Прерывание программы – это процесс переключения процессора с одной программы на другую по внешнему сигналу с сохранением информации для последующего возобновления прерванной программы (сохранение контекста). При возникновении события, приводящего к прерыванию, формируется сигнал, называемый *запросом прерывания*. Существует несколько источников запросов прерывания: это схемы контроля процессора, система питания, память, периферийные устройства и т.д. При наличии нескольких источников запросов прерывания устанавливается определенный порядок их обслуживания путем назначения приоритетов. Запросы прерываний направляются на различные разряды специального регистра (регистра запросов прерываний), опрос которого производится в строго определенной последовательности при завершении очередной команды. Номер разряда этого регистра не только определяет приоритет запроса прерываний, но и позволяет найти соответствующую данному запросу программу обслуживания прерывания. Поступивший запрос может прервать только менее приоритетную программу.

Существует также векторная система прерываний. В ней информация о месте возникновения запроса передается от источника прерываний в виде адреса ячейки памяти, содержимое которой определяет конкретную программу обслуживания. Помимо адреса перехода к программе обслуживания эта ячейка хранит дополнительную управляющую информацию. Содержимое этой ячейки (или нескольких последовательных ячеек) принято называть *вектором прерываний*.

Еще один очень важный процесс – *приостановки*, при которых средства управления, работающие автономно от процессора, задерживают его работу на время цикла памяти, когда память занята приемом или выдачей информации для другого устройства. Во время приостановок, называемых также занятием цикла памяти, процессор никаких действий

не выполняет, его состояние не меняется, но выполнение очередной команды задерживается до освобождения памяти. Возможности ограничены непосредственной передачей данных между ОП и процессором, когда память или шина используются несколькими устройствами.

Работой компьютера управляют команды программы, но конкретные выполняемые действия зависят от его текущего состояния. При работе машины в специальном регистре процессора постоянно находится *слово состояния программы*, которое и характеризует его состояние. Это слово содержит информацию, необходимую для возобновления программы при прерываниях: указания о разрешенных прерываниях, адрес текущей выполняемой команды, различные признаки, ключи защиты и маски.

Режимы работы процессора

При наличии единственного процессора компьютер в каждый момент времени может выполнять лишь одну команду программы. Однако в память компьютера может быть введено несколько программ. При *однопрограммном* режиме работы переход к выполнению следующей программы осуществляется только после полного завершения предыдущей. Компьютер, способный работать исключительно в однопрограммном режиме, обладает наиболее простой структурой. Именно по этой причине все первые персональные компьютеры были однопрограммными.

Однако выполнение программы требует не только ресурсов процессора. Нужно, чтобы в оперативной памяти присутствовали; как данные, так и программа, а обработка не задерживалась из-за отсутствия в ОП необходимой информации или занятости областей памяти результатами предыдущей обработки. Ввод и вывод информации осуществляются соответствующими устройствами, например клавиатурой, принтером и т.п. При вводе и вывод. ресурсы процессора (например, арифметическое устройство) практически не используются и, следовательно, могут быть доступными для другой программы.

Для этого в память компьютера должно быть введено несколько программ. Они могут находиться в состоянии *обработки* (активное состояние, при котором программа обрабатывается в процессоре), *готовности к обработке* или *ожидания* некоторого события (завершения операции ввода-вывода или освобождение ресурса).

Когда для текущей программы не хватает данных, процесс переключается на выполнение другой готовой для обработки программы, а текущая программа откладывается. Такой режим называют *мультипрограммным*; он предполагает наличие нескольких независимых программ (или фрагментов программы), принятых на обслуживание. При этом фрагменты программы считают независимыми, если каждый из них может быть выполнен без результатов обработки других. При мультипрограммном режиме обслуживание, т.е. обработка, ввод или вывод любой программы, может быть начато независимо от завершения других фрагменты. Все программы (или запросы на выполнение) находятся в очередях к соответствующим устройствам: процессору, внешней памяти, устройствам ввода или вывода.

Переключение программ из состояния готовности в состояние обработки требует дополнительного времени для процессора на выполнение управляющей программы. Кроме того, необходимы дополнительные средства для сохранения промежуточных результатов, ведения очередей и т.п. Но за счет мультипрограммирования удается значительно повысить загрузку процессора, а следовательно, и пропускную способность компьютера. Скорость работы процессора остается прежней, но за счет его равномерной и более плотной загрузки повышается пропускная способность компьютера.

При мультипрограммном режиме возникают дополнительные трудности в случае организации вывода на коллективно используемое периферийное устройство, например на единственный принтер, который печатает результаты работы всех программ. Так, прежде чем начать печатать результаты решения очередной задачи, необходимо полностью закончить печать результатов предыдущей. По этой причине организуют так называемый системные вывод, т.е. результаты обработки заносят в буферные области памяти, отведенные для каждой задачи, а вывод результатов на коллективно используемый принтер выполняют из этих областей только после его освобождения.

Таким образом, выполняемые в мультипрограммном режиме программы конкурируют за получение времени процессора, доступа к памяти, устройствам ввода и вывода. Распределение ресурсов системы между выполняемыми программами осуществляется управляющими программами ОС, которые переключают программы и предоставляют им необходимые ресурсы.

Режимы обработки программ

Известно несколько режимов обработки программ. В системах общего назначения наиболее часто используют *пакетный режим*. Пакет заданий (совокупность независимых друг от друга программ) вводят в память компьютера. Каждое задание снабжается описанием, содержащим приоритет задания, необходимую емкость памяти и требуемые устройства ввода-вывода. Задания, имеющие одинаковый приоритет, попадают в одну очередь. Задача из очереди с наибольшим приоритетом, находящаяся в состоянии готовности, переходит в активное состояние и начинает обрабатываться. Этот процесс будет прерван, если необходимые для решения задачи ресурсы недоступны, а также при появлении готовых к выполнению более приоритетных задач. Однако пакетный режим решения задач обладает и рядом недостатков:

– время ожидания результатов решения задачи заранее не определено; помимо ресурсов компьютера оно зависит от числа входящих в пакет программ и их приоритетов, приоритета самой задачи и других факторов.

– невозможно оперативно вносить изменения в программу;

– любое изменение связано с необходимостью повторного ввода нового пакета.

Все эго привело к появлению компьютеров, работающих в режиме *разделения времени*. Компьютер, работающий в этом режиме, должен предоставлять каждому пользователю терминал, посредством которого пользователь может передать запрос на обработку своей задачи и получить результаты ее решения. Процессор выделяет каждому запросу для его обработки некоторый интервал времени. Если длительность этого интервала недостаточна для завершения обработки запроса, то такая программа прерывается (независимо от степени готовности решения) и опять направляется в очередь, а ресурсы процессора предоставляются следующей программе. Длительность этих интервалов обслуживания выбирается так, чтобы сделать приемлемыми время ожидания ответа на запросы и затраты времени на переключение программ.

Помимо выбора длительности интервала необходимо распределить их между отдельными программами. Такое распределение устанавливается в соответствии с принятой дисциплиной обслуживания. В простейшем случае все вновь поступающие запросы пользователей ставятся в конец общей очереди, а для обслуживания выбирается программа из ее начала. Если за выделенный интервал времени программа успевает завершиться,

то результат выдается пользователю. Если же по истечении этого интервала выполнение программы не завершено, она направляется в конец очереди, а ресурсы компьютера предоставляются следующей программе из очереди.

Для уменьшения потерь времени на переключение программ и первоочередное предоставление ресурсов более коротким программам используют дисциплину обслуживания с несколькими очередями, для которых выделяемые интервалы времени переменны. Поступающая на обработку программа ставится в конец очереди с наибольшим приоритетом. Если эта очередь пустая, то происходит выбор задачи из следующей очереди, а если программа за выделенный интервал времени не завершена, она переходит в конец следующей очереди. Задачи, стоящие в последней предусмотренной очереди, выполняются до конца без прерывания. Иногда приоритет делается зависящим от времени ожидания. Тогда программа может переходить из одной очереди в другую, а приоритет становится динамическим.

Компьютеры, предназначенные для систем цифровой обработки сигналов, работают в режиме *реального времени*. Существует множество задач фильтрации сигналов, спектрального анализа и синтеза, распознавания речи, управления производственными процессами и ряда других, решение которых должно быть получено не позже определенного момента времени. Данные для них поступают и выдаются по каналам связи в цифровом или аналоговом виде. Компьютер работает в реальном масштабе времени, если он формирует решение не позже заранее установленного срока. Быстродействие такого компьютера зависит от управляемого процесса; оно может быть и очень большим, и сравнительно невысоким.

Для поддержания режима реального времени в компьютере должен быть таймер, позволяющий измерять интервалы времени между различными событиями. Однако таймер устанавливается не только в компьютерах реального времени. Он необходим и для компьютеров, работающих в пакетном режиме и режиме разделения времени. В компьютерах, работающих в пакетном режиме, переключение программ производится не только тогда, когда программа не может выполняться из-за нехватки ресурсов, но и по истечении некоторого интервала времени. В режиме разделения времени необходимо измерять предоставляемый каждой программе интервал. Кроме того, обычно ведется учет времени, использованного каждой программой.

Наличие нескольких выполняемых программ приводит к необходи- мости организации защиты памяти, предотвращающей воздействие одной программы на другую. Каждая программа получает в свое распоряжение определенную область памяти, а вторжение в «чужую» область приводит к искажению информации, принадлежащей другой программе. Для предотвращения этого в компьютерах предусматривают средства защиты памяти.

Контрольные вопросы

1. Что такое микропроцессор? Для чего он служит?
2. Из каких основных узлов состоит процессор?
3. Что представляет из себя команда процессора и из каких частей она состоит?
4. На какие типы делятся команды?
5. Что такое система команд процессора?
6. Какой формат имеет команда процессора?
7. Сколько полей под адреса отводится в команде?
8. Какие способы адресации операндов используются командах? Кратко охарактеризуйте каждый тип.
9. Что такое прерывание и что такое система прерываний процессора?
10. Какие типы прерываний бывают? Кратко охарактеризуйте каждый тип.
11. Что такое слово состояния программы и для чего оно служит?
12. В каких режимах может работать процессор?
13. В каких режимах может обрабатываться программа?
14. Проведите сравнительный анализ режимов обработки программ.

Информация по теме:

Ethernet – самый распространенный в настоящее время стандарт локальных сетей. Ethernet версии 1 появилась в 1980 г. В 1982 году была опубликована спецификация на Ethernet версии 2.0. Обе версии используются до сих пор, причем между ними сущест- вуют различия и по интерфейсу, и по уровню сигналов. На базе Ethernet версии 2 институ- том IEEE был разработан стандарт IEEE 802.3. В 1995 году был принят стандарт Fast Ethernet, который во многом не является самостоятельным стандартом. Его описание 802.3u является дополнительным разделом к основному стандарту. Аналогично, принятый 1998 году стандарт Gigabit Ethernet описан в разделе 802.z основного документа. В таблице приведены данные «классических» технологий сети Ethernet.

Ethernet	Thick Wire Ethernet	Thin Wire Ethernet	UTP Ethernet	Fiber Optic Ethernet	Broadband Ethernet
IEEE 802.3	10BASE-5	10BASE-2	10BASE-T	10BASE-F	10BROAD36
Скорость пере- дачи данных, Мбит/с	10	10	10	10	10

Метод передачи сигналов	Одно-Полосной	Одно-полосной	Одно-полосной	Одно-полосной	Широко-полосной
Длина сегмента кабеля,м	500	185	100	2000	1800
Максим. расстояние между узлами сети (при использовании повторителей), м	2500	925	500	2500	3600
Максимальное число станций на сегменте	100	30	1024	1024	100
Максим. число повторителей Между любыми станциями сети	4	4	4	4	4
Тип кабеля	коаксиальный, «толстый» RG-8 или RG-11	коаксиальный, «тонкий» RG-58	Неэкранированная витая пара категории 3, 4, 5	Многомодовый волоконно-оптический кабель	75 Омный коаксиальный, «толстый»

Все конфигурации используют одинаковый метод доступа к среде передачи данных и единый формат пакета.

Домашнее задание:
Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 с

Лекция 36. Периферийные устройства.

1. Клавиатура и мышь.
2. Дисплеи
3. Принтеры
4. Контрольные вопросы.

Занятие (лекция)

Клавиатуры предназначены для ввода буквенно-цифровой или текстовой информации в память компьютера, а мыши – для указания какой-либо определенной точки на экране, например некоторого окна. В переносных компьютерах вместо мыши используют различные трекболы, шаровые манипуляторы, джойстики и другие подобные устройства, служащие для управления специальной меткой – маркером, перемещаемым по экрану. Для ввода текста помимо клавиатуры могут использоваться также устройства автоматического распознавания печатных и рукописных символов, устройства ввода с промежуточного носителя (например, с магнитной ленты или дискеты) и т.п.

36.1.1. Клавиатура

Она состоит из совокупности ключей, замыкаемых при нажатии клавиш, а также схем управления для формирования кода символа, исключения неоднозначности кодирования из-за «дребезга» контактов и выполнения других управляющих функций. Традиционная клавиатура содержит более 100 клавиш. Число клавиш всегда меньше числа символов в алфавите, поэтому на клавиатуре располагают специальные управляющие клавиши, изменяющие коды остальных.

В настоящее время существует несколько разновидностей ключей, но наиболее распространен ключ (рис. 36.1), состоящий из клавиши 1, возвратной пружины 2, плунжера 3, корпуса 4 и собственно контактов 5. При нажатии на клавишу контакты замыкаются, что приводит к изменению электрического сопротивления между ними. В некоторых клавиатурах вместо механических контактов используют подвижную и неподвижную пластины, при нажатии на клавишу между которыми происходит изменение емкости, или полупроводник, в котором возникает разность потенциалов под действием магнитного поля, создаваемого закрепленным на плунжере подвижным магнитом.

Клавиатура представляет собой прямоугольную матрицу, состоящую из ключей, расположенных на пересечении столбцов и строк (рис. 36.2). Генератор тактовых импульсов, счетчик и дешифратор служат для

последовательного опроса состояния ключей в столбцах клавиатуры. Сигнал с дешифратора поступает на очередной столбец клавиатуры и соответствующий адресный вход ПЗУ (вход *X*). Если в данном столбце находится нажатая клавиша, то этот сигнал проходит и на второй адресный вход *Y*. В ячейках ПЗУ записаны коды символов, таким образом, содержимое ячейки с адресом *XY*, т.е. код нужного символа, выдается на выходной регистр. Поскольку число клавиш на клавиатуре меньше числа символов, то в ПЗУ записаны не полные коды символов, а лишь младшие разряды. Старшие разряды определяются содержимым специального регистра (СРг), состояние которого фиксируется при нажатии клавиш переключения регистров, например клавиши Shift. Для исключения влияния «дребезга» контактов выдача символа из регистра осуществляется с задержкой на время завершения переходного процесса.

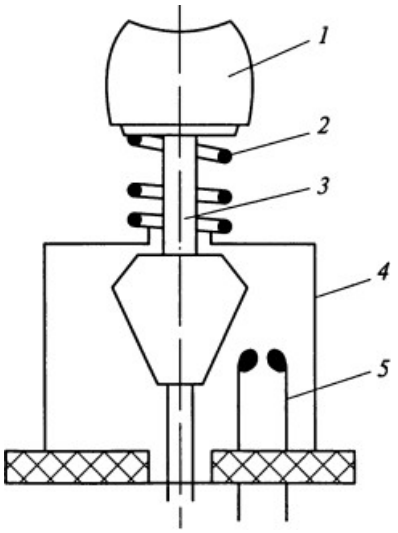


Рис. 36.1. Устройство ключа клавиатуры:

1 – клавиша; 2 – возвратная пружина; 3 – плунжер; 4 – корпус; 5 – контакты

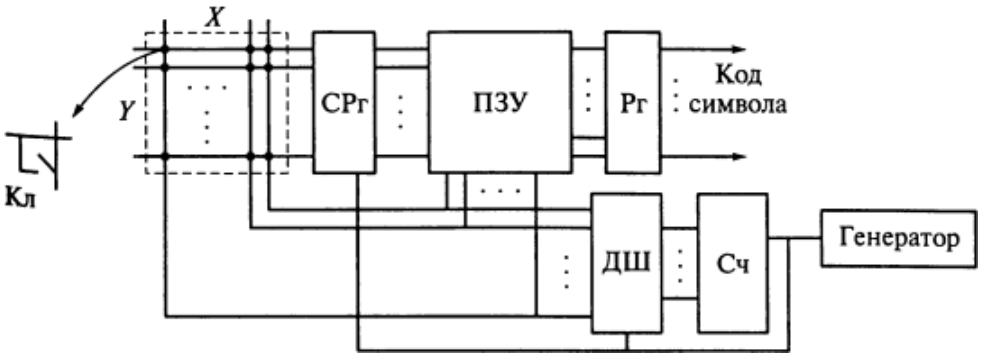


Рис. 36.2. Схема управления клавиатурой

Для управления клавиатурой можно использовать и микро- процессоры. В этом случае ее шины *XY* подключаются к портам ввода и вывода, а для передачи в компьютер сформированного программным путем кода символа используется второй порт вывода. При микропроцессорном управлении достаточно просто избежать влияния

«дребезга» контактов и фиксировать ошибочные состояния, когда одновременно оказываются нажатыми несколько клавиш. Для этих целей программа определения кода символа выполняется несколько раз и производится сравнение полученных кодов. Передача кода символа в выходной порт осуществляется только в случае многократного совпадения полученных кодов.

36.1.2. Мышь

В настоящее время наиболее распространенной является оптическая мышь, в которой использован полупроводниковый красный лазерный излучатель. Луч света от этого излучателя направляется на поверхность, по которой осуществляется перемещение мыши, и, отражаясь от нее, попадает в объектив монохромной КМОП-камеры, которая фотографирует

небольшой (около 1 мм^2) участок поверхности. *Фотографирование* – это

процесс разбиения участка поверхности на небольшие квадратики, число которых может быть 16×16, и вычисления усредненного значения яркости для каждого из них. Фотографирование производится сенсором, расположенным за объективом камеры. Конструктивно оптический сенсор и объектив выполнены в виде одной интегральной схемы, на нижней стороне которой располагается объектив. Каждому квадратному участку присваивается значение яркости от 0 до 63, где 0 соответствует белому, а 63 – черному участкам. Таким образом, получается мозаичное цифровое описание участка поверхности.

Это цифровое описание участка изображения передается от сенсорного датчика в сигнальный процессор, где запоминается и сравнивается с предыдущим описанием. Если описания полностью совпадают, то мышь не перемещалась; если совпадения не выявлено вовсе, то мышь перемещалась слишком быстро. Если полного совпадения нет, но выявлена некоторая общая часть изображения, то сигнальный процессор определяет величину и направление перемещения мыши и передает новые координаты мыши в компьютер. Поверхность фотографируется с очень большой скоростью – около 1500 снимков в секунду, что позволяет перемещать мышь довольно быстро со скоростью до 25 см/с. Оптическая мышь практически не загрязняется и позволяет работать достаточно быстро, но не на любой поверхности.

36.2. Дисплеи

Устройства оперативного отображения информации – дисплеи – долгое время строились исключительно на основе электронно-лучевых трубок (ЭЛТ). Экран такого дисплея представляет собой прямоугольную сетку, в узлах которой расположены отдельные элементарные индикаторы (ЭИ), изменяющие свои оптические свойства под воздействием управляющих сигналов. На экране ЭЛТ такая сетка может создаваться в процессе формирования изображения. Включение и выключение ЭИ для формирования видимого изображения производится в последова- тельности, называемой *опросом*.

В настоящее время используют *растровый метод*, при котором последовательность опроса всегда постоянна и не зависит от характера изображения. Луч (три луча – красный, зеленый и синий, или RGB) равномерно перемещается по экрану слева направо, прочерчивая строку растра. Это прямой ход луча. Затем выполняется обратный ход луча, по окончании которого он занимает положение в начале следующей строки, т.е. происходит перемещение луча сверху вниз. После выполнения последнего прямого хода, когда луч достиг крайнего нижнего положения, осуществляется его возврат в начало. Этот полный цикл называется *кадром*.

Во время прямого хода интенсивность луча можно менять, что, в свою очередь, ведет к изменению интенсивности излучения строго определенной точки экрана. Но точка на экране светится недолго, и нужно поддерживать ее состояние, т.е. многократно повторять весь процесс, чтобы изображение стало видимым. *Частота повторения процесса*, или *частота кадров*, должна быть не меньше критической частоты мерцаний, в современных дисплеях она составляет 80÷160 Гц.

Многократное повторение процесса отображения информации на экране ЭЛТ требует запоминания этой информации в буферной памяти. При использовании растрового метода формирования изображений объем буфера определяется произведением числа адресуемых точек на экране и числа бит, необходимых для записи параметров (например, цвета, яркости) каждой точки. Число адресуемых точек (разрешающая способность экрана) современного дисплея, составляет от 768×1024 до 1200×1600, а число бит для записи параметров – до 64. Таким образом, необходима буферная память объемом 64÷128 Мбайт.

Эта буферная видеопамять должна хранить описания каждого графического примитива (элементарного графического изображения) в *растровой форме*, т.е. в виде отдельных точек, а не векторной, когда этот

примитив обрабатывается в машине. Кроме того, графический дисплей позволяет производить процедуры над графическим примитивом: перемещать, поворачивать, масштабировать, стирать и т.д. Эти процедуры реализуются над примитивом, представленным в *векторной форме*. По этой причине в структуре видеокарты должен быть предусмотрен блок (рис. 36.3), выполняющий функции векторного графического процессора (ВГП) и ОЗУ для хранения описания дисплейного файла (ОЗУ ДФ). Преобразованный в растровую форму посредством растрового графического процессора (РГП) файл сохраняется в памяти видеоконтроллера (ВК). Оператор взаимодействует с таким дисплеем посредством специального блока интерактивного взаимодействия (БИВ) с помощью клавиатуры, мыши и других средств.

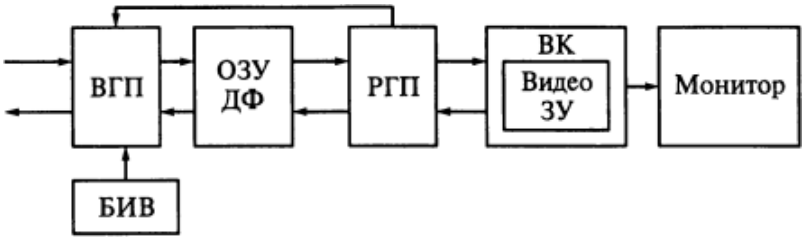


Рис. 36.3. Структура видеокарты:

ВГП – видеографический процессор; ОЗУ ДФ – ОЗУ для хранения дисплейного файла; РГП – растровый графический процессор; ВК – видеоконтроллер;

БИВ – блок интерактивного взаимодействия

Помимо формирования описания РГП совместно с ВК выполняет функцию кадрирования, т.е. формирует окно изображения. Процедура кадрирования может осуществляется программными и аппаратными средствами. Для аппаратного выполнения этой процедуры в видео- контроллере предусматривают специальные счетчики размера окна.

Графические дисплеи имеют «жесткую» временную диаграмму. Это значит, что время экспозиции точки растра всегда постоянно и определяется частотой регенерации кадра, числом строк в кадре и растровых элементов в строке. Эти параметры оказывают существенное влияние на преобразования, выполняемые растровым графическим процессором.

Недостатки дисплеев на ЭЛТ – большие габаритные размеры и масса, высоковольтное питание, сравнительно большое энергопотребление.

В ЖК-мониторах сохраняется необходимость векторно-растрового преобразования. Принцип действия любого ЖК-монитора основан на способности некоторых типов жидкокристаллических веществ поворачивать плоскость поляризации проходящего через них света.

Такая панель представляет собой две стеклянные пластины с нанесенными на них прозрачными электродами. Между стеклянными пластинами находятся ЖК, а сверху и снизу они покрыты поляризующими пленками, или поляроидами (рис. 36.4). Снизу эта конструкция подсвечивается ртутной флуоресцентной лампой с холодным катодом. Пусть обе пленки имеют горизонтальную поляризацию, а ЖК расположены так, что не меняют направление поляризации света, когда электрическое поле отсутствует. Тогда свет от лампы, пройдя сквозь первую поляризующую пластину, получит горизонтальную поляризацию, без изменений пройдет сквозь слой ЖК, а затем и через второй поляризатор. Такая панель будет казаться прозрачной.

Если же изменить положение ЖК (для этого нужно создать электрическое поле) так, чтобы они поворачивали плоскость поляризации проходящего через них света на 90°, то свет не пройдет сквозь вторую поляризующую пластину и панель станет непрозрачной. (Свет будет полностью поглощен вторым поляризатором, так как направление его поляризации будет перпендикулярно направлению поляризации второй пластины.) Поворачивая жидкие кристаллы на промежуточные углы, можно плавно регулировать прозрачность панели. Для получения цветного изображения каждый пиксель панели разбивается на три субпиксела и на него накладывается цветоделительная маска, окрашивающая проходящий через каждый субпиксель свет в один из основных цветов: красный, синий и зеленый.

Рис. 36.4. Структура жидкокристаллической панели

ЖК-мониторы, в основу которых положена модуляция внешнего света, обладают низкой контрастностью. Это вызвано тем, что поляризаторы всегда пропускают пусть даже небольшую часть света, т.е. черный цвет не может иметь нулевую интенсивность свечения. Кроме того, из-за большой толщины панели угол обзора оставался крайне

небольшим, а в силу большой длины электродов, подходящих к каждой точке, такие панели обладали весьма большим временем реакции. По этим причинам такие панели в настоящее время уже не используются, а вместо них выпускаются активно-матричные – каждый пиксел имеет собственный управляющий транзистор и снабжен поддерживающим напряжение конденсатором. Такие панели называют *активными матрицами*, или *TFT-панелями*, так как в них используют тонкопленочные транзисторы (толщиной менее 0,1 мкм) для обеспечения прозрачности матрицы.

В ЖК-мониторе используется постоянная подсветка элемента экрана. Она позволяет избежать мерцаний. Управляющее напряжение приложено к каждой ячейке в течение кадра, поэтому не требуется высокая частота регенерации. Однако длительное свечение точек изображения ухудшает воспроизведение быстродвижущихся объектов. По этой причине далеко не все ЖК-мониторы пригодны для динамичных игр и просмотра видеофильмов.

Размеры современных ЖК-мониторов составляют 15, 17, 19 и 26 дюймов, а стандартное разрешение 1280×1024 при размере пикселя в 0,264 мм. Они обладают достаточно высокой яркостью, а углы обзора по горизонтали и вертикали составляют порядка 150÷170°. Время переключения пикселя лежит в пределах от 16 до 25 мс. Благодаря компактности конструкции в ЖК-мониторах можно предусмотреть возможность поворота экрана на 90°, что удобно при работе с текстовой информацией. Кроме того, для ЖК-мониторов не требуется высокого напряжения, а потребление электроэнергии у них сравнительно невелико.

36.3. Принтеры

Основными требованиями, предъявляемыми к современным принтерам – высокое качество печати, использование различных шрифтов, возможность печати графики, получения многоцветных изображений и нескольких копий, высокое быстродействие и низкая стоимость. Указанные требования могут противоречить друг другу, что приводит к большому разнообразию конструкций.

Все принтеры принято классифицировать по следующим признакам: способу регистрации изображения (лазерные, струйные, ударные и т.д.), способу формирования изображения (знакопечатающие и матричные), числу одновременно формируемых символов (последовательные, построчные и страничные), в соответствии с размерами бумаги, на которой формируется изображение.

В настоящее время дома и в офисах наиболее широко применяются лазерные и струйные принтеры. Принтеры, основанные на термографи- ческом принципе формирования изображения, применяются при печати чеков, билетов и т.п. В них используется специальная термобумага, меняющая цвет при тепловом воздействии на нее.

Рассмотрим принципы лазерной и струйной печати. Лазерные печатающие устройства обладают высокой скоростью печати, достигающей 20 ÷ 25 страниц в минуту, и обеспечивают высокое разрешение в 1200 точка/дюйм и выше. Обычно лазерные принтеры предназначаются для рабочих групп, но существуют модели и для индивидуального пользования. Если первые лазерные устройства обеспечивали исключительно монохромную печать, то в настоящее время существуют устройства, позволяющие получать цветные отпечатки. Лазерный способ регистрации изображения основан на явлении местного разрушения электростатического заряда, созданного в слое полупровод- никового материала под действием света.

Структура одного из первых лазерных принтеров показана на рис. 36.5, *а*. Вращающийся барабан 4, покрытый слоем фотопроводника, заряжается вследствие образования на его поверхности слоя положительных ионов под действием ионизатора 6. Ионизатор заряжает всю поверхность барабана. При попадании на эту поверхность луча света 7 происходит точечное разрушение заряда.

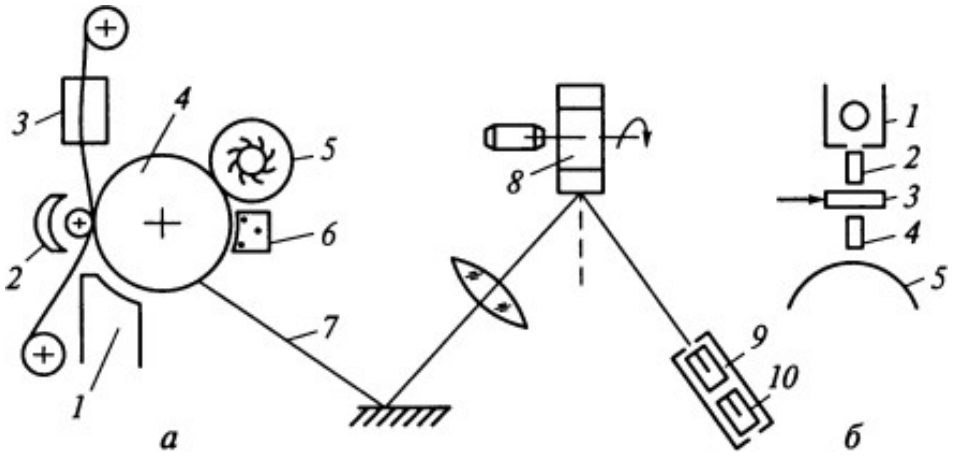


Рис. 36.5. Лазерный принтер:

а – формирование видимого изображения с помощью лазерного луча:
1 – краситель; 2 – бумага; 3 – устройство термосилового закрепления;
4 – барабан; 5 – устройство очистки барабана; 6 – ионизатор; 7 – луч;
8 – зеркало; 9 – модулятор; 10 – лазер;

б – формирование видимого изображения газоразрядной лампой:

1 – газоразрядная лампа; 2 – система световодов; 3 – панель ЖК-затворов;
4 – фокусирующая матрица; 5 – барабан

Таким образом, если избирательно осветить поверхность барабана, то на ней можно создать скрытое электростатическое изображение, формируемое лазерным лучом лазера 10 и модулируемое модулятором 9. Перемещение луча по поверхности барабана осуществляется многогранным зеркалом 8. Скрытое электростатическое изображение проявляется путем осаждения на положительно заряженные участки поверхности отрицательно заряженных частиц красителя 1. Затем проявленное изображение переносится на бумагу 2, где краситель фиксируется путем термосилового закрепления 3. Поверхность барабана 4 очищается для печати следующей страницы устройством очистки барабана 5.

В настоящее время более широкое распространение получил способ формирования скрытого изображения на поверхности барабана с помощью газоразрядной лампы (рис. 36.5, *б*). Свет от такой газоразрядной лампы 1 расщепляется на отдельные лучи с помощью системы оптоволоконных световодов 2, и каждый луч модулируется индивидуальным ЖК-затвором. Совокупность таких затворов выполнена в виде единой панели 3. Затем оптическая фокусирующая матрица 4 формирует отдельные пучки света, образуя горизонтальные ряды точек строки изображения. Развертка по вертикали создается вращением барабана 5. Все остальные действия по переносу изображения на бумагу, закреплению изображения и подготовке поверхности барабана для печати следующей страницы выполняются так же, как описано выше. Для получения цветного отпечатка процедура печати выполняется трижды; при этом используются три красителя красного, зеленого и синего цветов. В последних моделях цветных лазерных принтеров все три красителя наносят за один оборот барабана, что значительно повышает скорость цветной печати.

Струйные принтеры обладают значительно меньшим быстродействием, но качество цветной печати у них очень высокое. Так, разрешение современных струйных принтеров составляет порядка 4800×1200 точка/дюйм, а время получения одного отпечатка – около 30 с. Обычно такие принтеры позволяют печатать как на стандартных листах бумаги формата А4, так и на формате 10×15 см, характерном для фотоснимков. Высокое качество печати и такой формат привели к тому, что струйные принтеры часто называют фотопринтерами. Печать производится посредством специальных сопел, выбрасывающих капли красителя размером около 2÷3 пикалитров. Число сопел может достигать 512 на каждый цвет и в конечном итоге определяет качество получаемого отпечатка. Выброс струи производится в момент, когда сопло 2 (рис. 36.6) находится напротив печатаемой точки изображения. Импульсное давление в сопле

создается пьезоэлектрическим элементом 1, а жидкий краситель может поступать из специального резервуара или представлять собой твердый краситель, размягчаемый под действием тока. Перемещение сопла или наличие нескольких сопел позволяет синтезировать контур в виде мозаики от дельных точек. Кроме того, наличие нескольких сопел даст возможность получать много цветные изображения.

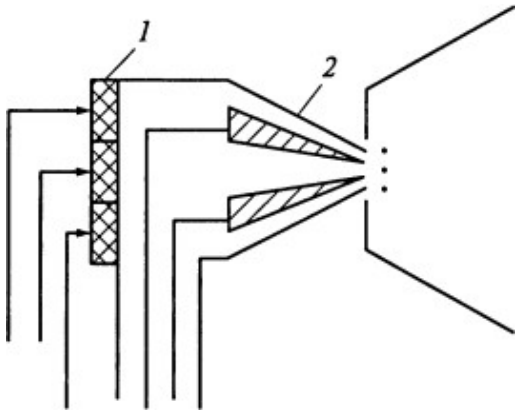


Рис. 36.6. Устройство сопла струйного принтера: 1 – пьезоэлектрический элемент; 2 – сопло

Струйный метод печати используют не только в принтерах, но и в специальных широкоформатных устройствах регистрации графической информации для получения различных плакатов, рекламы и т.п.

Контрольные вопросы

1. Как формируется код символа в клавиатуре? Как он передается на память компьютера?
2. Каково назначение мыши? На чем основана работа оптической мыши? Поясните принцип ее работы.
3. В чем принцип работы дисплея на базе ЭЛТ? Что такое частота регенерации изображения и чему она равна в современных дисплеях?
4. Как работает ЖК-дисплей? Что такое разрешающая способность? Какой разрешающей способностью обладают современные дисплеи?
5. Какие принтеры в настоящее время наиболее распространены? Какая разрешающая способность для них характерна?
6. Как работает струйный принтер? Какими достоинствами и недостатками он обладает?
7. Как работает лазерный принтер? Перечислите все этапы формирования изображения.

Информация по теме:

Спецификация Ethernet и IEEE 802.3 определяют несколько конфигураций подключения оборудования. Они различаются типом кабеля, топологией подключения устройств и т.п. Во всех конфигурациях используется один метод доступа станций к среде – множественный доступ с контролем несущей и обнаружением коллизий – Carrier Sense Multiple Access/Collision Detection (CSMA/CD).

Во время работы станция постоянно проверяет среду передачи. Передающая среда может быть свободна, если ни одна другая станция не передает данные, или занята. Любая станция, обнаружив, что среда свободна, может начать передачу своих данных. Поэтому возможна ситуация, когда одновременно несколько станций начнут передавать данные – коллизия. При этом происходит смешение и искажение сигналов. Четкое распознавание коллизий всеми станциями сети является необходимым условием корректной работы сети Ethernet. Если какая-либо станция не распознает коллизию и решит, что переданный ею кадр верен, то кадр будет утерян или значительно искажен. Для надежного распознавания коллизий должно выполняться соотношение: T_{min} □ PDV, где T_{min} – время передачи кадра минимальной длины, а PDV – время, за которое сигнал коллизии успевает распространиться до самого дальнего узла сети – время двойного оборота.

Минимальная длина пакета в IEEE 802.3 и Ethernet - 64 байта, со- ответственно длина поля данных - 46 байтов. IEEE 802.3 позволяет ис- пользовать символы-заполнители (PAD) в поле данных, если требует- ся передать сообщение короче, чем 46 байтов. В Ethernet пакет с длиной поля данных менее 46 байтов просто не будет обрабатываться. На рисунке показана структура пакета в сети Ethernet.

Назначение полей:

Преамбула служит для синхронизации работы приемника и передатчика.

АП - Адрес Приемника адрес станции, которой направляется пакет.

АИ - Адрес Источника - адрес передающей станции.

Поле Типа Пакета определяет тип Ethernet пакета.

Поле Длины Пакета определяет в IEEE 802.3 количество байт данных в поле данных. Количество символов-заполнителей (PAD) не входит в это число.

Поле Данных содержит данные и символы-заполнители в IEEE 802.3, данные - в Ethernet.
ПОО Поле Обнаружения Ошибок служит для определения достоверности полученной информации.

IEEE 802.3 определяет, что после поля адреса источника следует двухбайтное поле длины пакета, а в Ethernet - поле типа пакета.

Домашнее задание:
Проработка конспекта пройденной темы.

Литература:Чекмарев Ю.В. Локальные вычислительные сети : учебное пособие / Чекмарев Ю.В.. — Саратов : Профобразование, 2017. — 200 с