Министерство образования и науки Российской Федерации

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ГОУВПО «АмГУ»)

УТВЕРЖДАЮ

Зав. кафедрой ИиУС
_____ А.В. Бушманов
«___» _____ 2010 г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ «АВТОМАТИЗИРОВАННЫЕ КОМПЛЕКСЫ ИНФОРМАЦИОННЫХ СИСТЕМ»

для направления подготовки 230100.68
«Информатика и вычислительная техника»

Составитель: Шевко Д.Г.

2010

Печатается по решению
редакционно-издательского совета
факультета математики и информатики
Амурского государственного
университета

Шевко Д.Г. Учебно-методический комплекс дисциплины «Автоматизированные комплексы информационных систем» для направления подготовки 230100.68 «Информатика и вычислительная техника». – Благовещенск: Амурский гос. ун-т, 2010. – 181 с.

Учебно-методическое пособие содержит: рабочую программу преподавания дисциплины; изложение курса лекций; методические указания и учебные задания для выполнения курса лабораторных работ; карту обеспеченности дисциплины кадрами профессорско-преподавательского состава

РАБОЧАЯ ПРОГРАММА

по дисциплине «Автоматизированные комплексы информационных систем» для направления подготовки 230100.68 «Информатика и вычислительная техника»

Курс 2 Семестр 3

Лекции 18 час. Экзамен

Практические занятия Зачет 3 семестр

Лабораторные занятия 54 час.

Самостоятельная работа

Всего часов

Составитель: доцент Шевко Д.Г.

Факультет Математики и информатики

Кафедра Информационных и управляющих систем

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

Целью преподавания дисциплины является овладение теоретическими и практическими знаниями при комплексировании вычислительных средств, вычислительных систем и сетей, для получения технического обеспечения автоматизированных систем управления разнообразными по своей структуре процессами.

Задачи изучения дисциплины ориентированы на получение основных сведений о принципах организации и функционирования отдельных устройств ЭВМ в целом, характеристики, возможности и области применения, наиболее распространенных классов и типов ЭВМ, при решении различного класса задач, которыми должен овладеть студент после изучения данной дисциплины.

В результате изучения дисциплины студент должен:

Знать принципы построения и организацию функционирования современных вычислительных машин, систем и сетей, их функциональную и структурную организацию, основы построения элементов, узлов и устройств вычислительных машин, технико-эксплуатационные показатели

средств вычислительной техники, используемых для обработки информации;

Уметь использовать методы анализа и синтеза цифровых схем и микропроцессорных средств при создании АСОИУ, методы и средства разработки и отладки программ на языках низкого уровня;

Иметь опыт выполнения схемотехнических расчетов электронных элементов и устройств ЭВМ, проектирования микропроцессорных контроллеров, анализа и оценки архитектуры вычислительных систем.

2. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Дисциплина «Автоматизированные комплексы информационных систем» для направления подготовки 230100.68 «Информатика и вычислительная техника» введена по решению УМС.

НАИМЕНОВАНИЕ ТЕМ, ИХ СОДЕРЖАНИЕ, ОБЪЕМ

Раздел I. Принципы построения и архитектура ЭВМ (11 час)

Тема 1.1. Основные характеристики ЭВМ (1 час)

1. Основные понятия: структура, архитектура ЭВМ.
2. Основные характеристики ЭВМ: быстродействие, производительность, емкость запоминающего устройства, надежность, точность, достоверность.

Тема 1.2. Классификация средств ЭВТ (1 час)

1. Два типа электронной вычислительной техники: аналоговая и цифровая. Сферы применения ЭВМ.
2. Классификация средств вычислительной техники по быстродействию: суперЭВМ, большие ЭВМ, средние ЭВМ, персональные и профессиональные ЭВМ, встраиваемые микропроцессоры.

Тема 1.5. Математические и логические основы ЭЦВМ (2 час)

1. Построение ЭЦВМ. Структурная схема ЭЦВМ.
2. Этапы решения задачи на ЭВМ.
3. Системы счисления. Арифметические операции в двоичной с.ч. Перевод чисел из одной системы счисления в другую.
4. Представление чисел в ЭВМ.
5. Общие сведения о представлении других видов информации.

Тема 1.6. Элементная база ЭВМ (Логические основы ЭВМ) (1 час)

Тема 1.7. Центральные устройства ЭВМ (2 час)

1. Основная память (ОП). Основные показатели. Классификация устройств ОП. Назначение ОЗУ, ПЗУ. Структурная схема ОЗУ. Основные характеристики ОЗУ.
2. Стековая память. Кэш-память первого, второго уровней. Размещение информации в ОП.
3. Эволюция микропроцессоров(МП). Основные технические характеристики МП. Классификация МП.
4. Структурная схема МП.
5. Полный цикл работы МП при выполнении команды.
6. Система команд МП.
7. Структурная схема базовой модели микропроцессора фирмы Intel.

Тема 1.8. Периферийные устройства ЭВМ (1 час)

1. Видеосистема.
2. Клавиатура.
3. Принтер.
4. Сканер.
5. Манипулятор типа мышь.
6. Накопитель на гибком магнитном диске.
7. Накопитель на жестком магнитном диске.
8. Накопитель на оптическом диске.

Тема 1.9. Архитектура структуры ЭВМ (2 час)

1. Общие сведения о специальных ЭВМ (сЭВМ). Общая архитектура и состав сЭВМ.
2. Структурная схема системной платы ПК.
3. Общие сведения о мини-ЭВМ. Общая архитектура и состав мини-ЭВМ.
4. Общие сведения о ЭВМ общего назначения. Общая архитектура и состав ЭВМ общего назначения.

Тема 1.10. Программное обеспечение ЭВМ (1 час)

1. Структура программного обеспечения ЭВМ.
2. Операционные системы. Структура ДОС ПЭВМ.
3. Системы автоматизации программирования.
4. Режимы работы ЭВМ.

Раздел II. Архитектура вычислительных систем (4 час)

Тема 2.1. Классификация вычислительных систем (1 час)

1. Понятие ВС. Принципы построения ВС. Классификация ВС.

2. Типы ВС.

Тема 2.2. Архитектура вычислительных систем (1 час)

1. Четыре основные архитектуры ВС.
2. Отличительные особенности архитектур ВС.

Тема 2.3. Типовые структуры вычислительных систем (2 час)

- типовые структуры ВС;
- три аспекта совместимости: аппаратный, программный, информационный;
- уровни комплексирования. Классификация уровней программного параллелизма: независимые задания, отдельные части задания, программы и подпрограммы, циклы и итерации, операторы и команды, фазы отдельных команд. Структуры ВС: ОКОД, МКМД;
- организация функционирования ВС. Операционные системы многомашинных ВС. Программное обеспечение многопроцессорных ВС.

Раздел III. Телекоммуникационные вычислительные сети (3 час)

Тема 3.1. Принципы построения ТВС (1 час)

1. Общие сведения.
2. Классификация сетей.
3. Управление взаимодействием прикладных процессов

Тема 3.2. Телекоммуникационные системы (ТКС) (1 час)

1. Функции ТКС.
2. Типы сетей, линий и каналов связи.
3. Коммутируемые и выделенные каналы.
4. Аналоговое и цифровое кодирование цифровых данных.
5. Цифровые сети связи.
6. Спутниковые сети связи.
7. Коммутация в сетях.
8. Маршрутизация пакетов в сетях.
9. Методы защиты от ошибок.

Тема 3.3. Глобальные вычислительные сети и сетевые технологии (1 час)

1. Протоколы обмена данными.
2. Протокол TCP/IP.
3. Системы сетевых коммутаций.

4. Электронная почта стандарта X400.
5. Электронная почта стандарта Internet.
6. Системы телеконференций.
7. Сеть Internet.
8. Отечественные глобальные сети.

ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа №1. Задачи по моделям памяти и структурам программ (6 часов)

Лабораторная работа №2. Задачи по программированию на языке ассемблера (12 часов)

Лабораторная работа №3. Задачи по программированию операций над файлами, каталогами и дисками (12 часов)

Лабораторная работа №4. Задачи по защите программ от копирования и несанкционированного использования (12 часов)

Лабораторная работа №5. Задачи по программированию ввода с клавиатуры (12 часов)

САМОСТОЯТЕЛЬНАЯ РАБОТА СТУДЕНТОВ

В качестве самостоятельной работы по дисциплине «Автоматизированные комплексы информационных систем» студенты готовят рефераты по следующим темам:

1. Основные характеристики, области применения ЭВМ различных классов;
2. Функциональная и структурная организация процессора;
3. Организация памяти ЭВМ;
4. Основные стадии выполнения команды;
5. Организация прерываний в ЭВМ;
6. Организация ввода-вывода;
7. Периферийные устройства;
8. Архитектурные особенности организации ЭВМ различных классов;
9. Параллельные системы;
10. Понятие о многомашинных и многопроцессорных вычислительных системах;

11. Матричные и ассоциативные вычислительные сети;
12. Конвейерные и потоковые вычислительные сети;
13. Сети ЭВМ;
14. Информационно-вычислительные системы и сети.

ВОПРОСЫ К ЗАЧЕТУ

Основные характеристики ЭВМ
Классификация средств ЭВТ
Математические и логические основы ЭЦВМ
Элементная база ЭВМ
Центральные устройства ЭВМ
Периферийные устройства ЭВМ
Архитектура структуры ЭВМ
Программное обеспечение ЭВМ
Классификация вычислительных систем
Архитектура вычислительных систем
Типовые структуры вычислительных систем
Принципы построения ТВС
Телекоммуникационные системы
Глобальные вычислительные сети и сетевые технологии

1. Организация современных ЭВМ. Устройство ЭВМ семейства Intel
2. Организация машинного уровня ЭВМ
3. Структура и форматы команд ЭВМ. Способы адресации
4. Процессор. Центральное управляющее устройство ЭВМ
5. Арифметико-логическое устройство ЭВМ. Регистры ЭВМ
6. Организация памяти ЭВМ. Организация внутренней памяти процессора
7. Способы организации кэш-памяти. Методы обновления строк основной памяти
8. Оперативная память. Методы управления памятью
9. Организация виртуальной памяти. Страничное и сегментное распределение памяти
10. Методы повышения пропускной способности оперативной памяти. Защита памяти
11. Организация подсистем ввода-вывода. Основные различия
12. Ввод-вывод в режиме прерываний
13. Прямой доступ к памяти. Контроллеры ввода-вывода
14. Унификация средств обмена и интерфейсы ЭВМ

15. Стандартные шины ЭВМ. Основные типы и их характеристики
16. Информационно вычислительные системы и сети ЭВМ
17. Перспективы развития современных ЭВМ

Виды контроля

Для проверки эффективности преподавания дисциплины проводится контроль знаний студентов. При этом используются следующие виды контроля:

- *текущий контроль* за аудиторной и самостоятельной работой обучаемых осуществляется во время проведения аудиторных занятий посредством устного опроса, проведения контрольных работ;
- *промежуточный контроль* осуществляется два раза в семестр в виде анализа итоговых отчетов на аттестационные вопросы;
- *итоговый контроль* в виде зачета осуществляется после успешного прохождения студентами текущего и промежуточного контроля и сдачи отчета по самостоятельной работе.

Требования к знаниям студентов, предъявляемые на зачете

Для получения зачета студент должен посещать занятия, проявлять активность в аудитории, обязан выполнить все лабораторные работы, знать теоретический материал в объеме лекционного курса, защитить отчет по самостоятельной работе.

3. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

Перечень обязательной (основной) литературы

1. Цифровая схемотехника: учеб. Пособие: рек. УМО/ Е.П. Угрюмов. – 2-е изд., перераб. И доп.. – СПб.: БХВ – Петербург, 2007. – 782 с. : ил.
2. Вычислительные системы, сети и телекоммуникации: учеб. пособие: рек. Мин. обр. РФ/В. Л. Бройдо, О.П. Ильина. – 3-е изд.. – СПб.: Питер, 2008. – 766 с.: рис.
3. Проектирование систем на микросхемах с программируемой структурой: [учеб. пособие] / Р. Грушвицкий, А.Х. Мурсаев, Е.П. Угрюмов. – 2-е изд.. – СПб.: БХВ – Петербург, 2006. – 736 с.: ил.
4. Материнские платы и чипсеты/ Е.А. Рудометов. – 4-е изд.. – СПб.: Питер, 2007. – 368 с.: рис. табл.. – (Анатомия ПК).

5. Схемотехника электронных систем. Аналоговые и импульсные устройства: [учеб. пособие]/ В. И. Бойко [и др.]. – СПб.: БХВ – Петербург, 2004. – 482 с.: ил.. – Библиогр.: с. 475.
6. Схемотехника электронных систем. Цифровые устройства: [учеб. пособие]/ В. И. Бойко [и др.]. – СПб.: БХВ – Петербург, 2004. – 497 с.
7. Assembler: учеб.: доп. Мин. обр. РФ/ В.И. Юров. – 2-е изд.. – СПб.: Питер, 2008. – 637 с.: ил.. – (Учебник для вузов). – Библиогр.: с. 625.
8. Архитектура ЭВМ и систем: учеб.: рек.Мин. обр. РФ/ В.Л. Бройдо, О.П. Ильина. 2-е изд.. – СПб.: Питер, 2009. – 720 с.. – (Учебник для вузов).

Перечень дополнительной литературы

1. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: Учебник для вузов. – СПб.: Питер, 2006. – 668 с.
2. Assembler. Учебник для вузов / Юров В.И. – СПб.: Питер, 2008. – 637 с.
3. Assembler. Практикум / Юров В.И. – СПб.: Питер, 2008. – 399 с.
4. Архитектура компьютерных систем и сетей: Учеб. пособие / Т.П. Барановская, В.И. Лойко, М.И. Семенов, А.И. Трубилин; Под ред. В.И. Лойко. – М.: Финансы и статистика, 2003. – 256 с.: ил.
5. Бройдо В.Л. Вычислительные системы, сети и телекоммуникации: Учебник для вузов. 2-е изд. – СПб.: Питер, 2008. – 703 с.: ил.
6. Хамахер, Карл. Организация ЭВМ: [учеб. пособие] / К. Хамахер, З. Вранешич, С. Заки. - 5-е изд. - СПб.: Питер, 2003.
7. http://do.rksi.ru/library/courses/arh/

4. УЧЕБНО-МЕТОДИЧЕСКАЯ (ТЕХНОЛОГИЧЕСКАЯ) КАРТА ДИСЦИПЛИНЫ

Номер недели	Номер темы	Лабораторные занятия	Используемые метод. пособия и наглядные	Самостоятельная Работа студентов		Форма контроля
				Содержание	Часы	
1	2	3	4	5	6	7

1	1.1	1	1–8 (осн.), 1–7 (доп.)	Выбор темы самостоятельной работы	2	
2	1.2	1	1–8 (осн.), 1–7 (доп.)			з.л.р.
3	1.5	2	1–8 (осн.), 1–7 (доп.)	Поиск литературы по теме самостоятельной работы	6	
4	1.5	2	1–8 (осн.), 1–7 (доп.)			собесед.
5	1.6	2	1–8 (осн.), 1–7 (доп.)	Работа с литературой и поиск информации в сети Интернет по новейшим достижениям в области Автоматизированных комплексов информационных систем по теме самостоятельной работы	10	
6	1.7	2	1–8 (осн.), 1–7 (доп.)			з.л.р.
7	1.7	3	1–8 (осн.), 1–7 (доп.)			
8	1.8	3	1–8 (осн.), 1–7 (доп.)			к.р.
9	1.9	3	1–8 (осн.), 1–7 (доп.)			
10	1.9	3	1–8 (осн.), 1–7 (доп.)			з.л.р.
11	1.10	4	1–8 (осн.), 1–7 (доп.)			
12	2.1	4	1–8 (осн.), 1–7 (доп.)			собесед.
13	2.2	4	1–8 (осн.), 1–7 (доп.)			
14	2.3	4	1–8 (осн.), 1–7 (доп.)			з.л.р.
15	2.3	5	1–8 (осн.), 1–7 (доп.)	Подготовка отчета	6	
16	3.1	5	1–8 (осн.), 1–7 (доп.)			к.р.
17	3.2	5	1–8 (осн.), 1–7 (доп.)	Защита отчета по самостоятельной работе	2	
18	3.3	5	1–8 (осн.), 1–7 (доп.)			зачет

Условные обозначения:
осн. – основная литература
доп. – дополнительная литература
к.р. – контрольная работа
собесед. – собеседование
з.л.р. – защита лабораторной работы

ЛЕКЦИОННЫЙ КУРС

Раздел I. Принципы построения и архитектура ЭВМ [16]

Тема 1.1. Основные характеристики ЭВМ

Цель изучения темы: усвоить основные понятия и определения; изучить основные характеристики ЭВМ; уметь сравнить ЭВМ по основным характеристикам.

Вопросы.

1. Основные понятия: структура, архитектура ЭВМ.
2. Основные характеристики ЭВМ: быстродействие, производительность, емкость запоминающего устройства, надежность, точность, достоверность.

Вопрос 1.1.1 Основные понятия.

Структура - это совокупность элементов и их связей. Различают структуры технических, программных и аппаратурно-программных средств.

Архитектура ЭВМ - это многоуровневая иерархия аппаратурно-программных средств, из которых строится ЭВМ. Каждый из уровней допускает многовариантное построение и применение. Конкретная реализация уровней определяет особенности структурного построения ЭВМ.

Детализацией архитектурного и структурного построения ЭВМ занимаются различные категории специалистов вычислительной техники. Инженеры - схемотехники проектируют отдельные технические устройства и разрабатывают методы их сопряжения друг с другом. Системные программисты создают программы управления техническими средствами, информационного взаимодействия между уровнями, организации вычислительного процесса. Программисты-прикладники разрабатывают пакеты программ более высокого уровня, которые обеспечивают взаимодействие пользователей с ЭВМ и необходимый сервис при решении ими своих задач.

Вопрос 1.1.2 Основные характеристики ЭВМ

Структуру ЭВМ определяет следующая группа характеристик:

- технические и эксплуатационные характеристики ЭВМ (быстродействие и производительность, показатели надежности, достоверности, точности, емкость оперативной и внешней памяти, габаритные размеры, стоимость технических и программных средств, особенности эксплуатации т.д.);
- характеристики и состав функциональных модулей базовой конфигурации ЭВМ; возможность расширения состава технических и программных средств; возможность изменения структуры;
- состав программного обеспечения ЭВМ и сервисных услуг (операционная система или среда, пакеты прикладных программ, средства автоматизации программирования).

К основным характеристикам ЭВМ относятся:

Быстродействие это число команд, выполняемых ЭВМ за одну секунду.

Сравнение по быстродействию различных типов ЭВМ, не обеспечивает достоверных оценок. Очень часто вместо характеристики быстродействия используют связанную с ней характеристику производительность.

Производительность это объем работ, осуществляемых ЭВМ в единицу времени.

Применяются также относительные характеристики производительности. Фирма Intel для оценки процессоров предложила тест, получивший название индекс iCOMP (Intel Comparative Microprocessor Performance). При его определении учитываются четыре главных аспекта производительности: работа с целыми числами, с плавающей запятой, графикой и видео. Данные имеют 16- и 32-разрядной представление. Каждый из восьми параметров при вычислении участвует со своим весовым коэффициентом, определяемым по усредненному соотношению между этими операциями в реальных задачах. По индексу iCOMP ПМ Pentium 100 имеет значение 810, а Pentium 133-1000.

Емкость запоминающих устройств. Емкость памяти измеряется количеством структурных единиц информации, которое может одновременно находится в памяти. Этот показатель позволяет определить, какой набор программ и данных может быть одновременно размещен в памяти.

Наименьшей структурной единицей информации является **бит**- одна двоичная цифра. Как правило, емкость памяти оценивается в более крупных единицах измерения - байтах (байт равен восьми битам). Следующими единицами измерения служат 1 Кбайт = 210 = 1024 байта, 1 Мбайт = 210 Кбайта = 220 байта, 1 Гбайт =210 Мбайта = 220 Кбайта = 230 байта.

Емкость оперативной памяти (ОЗУ) и емкость внешней памяти (ВЗУ) характеризуются отдельно. Этот показатель очень важен для определения,

какие программные пакеты и их приложения могут одновременно обрабатываться в машине.

Надежность это способность ЭВМ при определенных условиях выполнять требуемые функции в течение заданного периода времени (стандарт ISO (Международная организация стандартов) 2382/14-78).

Высокая надежность ЭВМ закладывается в процессе ее производства. Применеие сверхбольшие интегральные схемы (СБИС) резко сокращают число используемых интегральных схем, а значит, и число их соединений друг с другом. Модульный принцип построения позволяет легко проверять и контролировать работу всех устройств, проводить диагностику и устранение неисправностей.

Точность это возможность различать почти равные значения (стандарт ISO - 2382/2-76).

Точность получения результатов обработки в основном определяется разрядностью ЭВМ, а также используемыми структурными единицами представления информации (байтом, словом, двойным словом).

Достоверность это свойство информации быть правильно воспринятой.

Достоверность характеризуется вероятностью получения безошибочных результатов. Заданный уровень достоверности обеспечивается аппаратурно-программными средствами контроля самой ЭВМ. Возможны методы контроля достоверности путем решения эталонных задач и повторных расчетов. В особо ответственных случаях проводятся контрольные решения на других ЭВМ и сравнение результатов.

Контрольные вопросы

1. Каково понятие структуры? Перечислите типы структурных средств.
2. Каково понятие архитектура ЭВМ?
3. Перечислите категории специалистов ВТ, занимающихся детализацией архитектурного и структурного построения ЭВМ.
4. Перечислите и поясните основные техническим характеристикам ЭВМ.

Тема 1.2. Классификация средств ЭВТ

Цель изучения темы: Изучить типы ЭВМ; усвоить сферы применения ЭВМ различных типов; изучить классификацию средств ЭВТ по быстродействию; научиться определять тип ЭВМ к конкретной сфере применения.

Вопросы.

1. Два типа электронной вычислительной техники: аналоговая и цифровая. Сферы применения ЭВМ.
2. Классификация средств вычислительной техники по быстродействию: суперЭВМ, большие ЭВМ, средние ЭВМ, персональные и профессиональные ЭВМ, встраиваемые микропроцессоры.

Вопрос 1.2.1 Два типа электронной вычислительной техники. Сферы применения ЭВМ.

Традиционную электронную вычислительную технику (ЭВТ) подразделяют на аналоговую и цифровую. В аналоговых вычислительных машинах (АВМ) обрабатываемая информация представляется соответствующими значениями аналоговых величин: тока, напряжения, угла поворота какого-то механизма и т.п. Эти машины обеспечивают приемлемое быстродействие, но не очень высокую точность вычислений (0.001-0.01). АВМ используются в основном в проектных и научно-исследовательских учреждениях в составе различных стендов по отработке сложных образцов техники. По своему назначению их можно рассматривать как специализированные вычислительные машины.

В цифровых вычислительные машинах (ЭВМ) информация кодируется двоичными кодами чисел. ЭВМ обладают универсальными свойствами и являются самой массовой ЭВТ.

Академик В.М.Глушков указывал, на три глобальные сферы деятельности человека, которые требуют использования качественно различных типов ЭВМ.

- Первое направление - применение ЭВМ для автоматизации вычислений
- Вторая сфера применения ЭВМ связана с использованием их в системах управления. Она родилась примерно в 60- годы, когда ЭВМ стали интенсивно внедряться в контуры управления автоматических и автоматизированных систем. Новое применение вычислительных машин потребовало видоизменения их структуры. ЭВМ используемые в управлении, должны были не только обеспечивать вычисления, но и

автоматизировать сбор данных и распределение результатов обработки.

Сопряжение с каналами связи потребовало усложнения режимов работы ЭВМ, сделало их многопрограммными и многопользовательскими. Для исключения взаимных помех между программами пользователей в структуру машин были введены средства разграничения: блоки прерываний и приоритетов, блоки защиты и т.п. Для управления разнообразной периферией стали использоваться специальные процессоры ввода-вывода данных или каналы. Именно тогда и появился дисплей как средство оперативного человеко-машинного взаимодействия пользователя с ЭВМ.

Новой сфере работ в наибольшей степени отвечали мини-ЭВМ. Именно они стали использоваться для управления отраслями, предприятиями, корпорациями. Машины нового типа удовлетворяли следующим требованиям:

- были более дешевыми по сравнению с большими ЭВМ, обеспечивающими централизованную обработку данных;
- были более надежными, особенно при работе в контуре управления; обладали большой гибкостью и адаптируемостью настройки на конкретные условия имели архитектурную прозрачность, т.е. структура и функции ЭВМ были понятны пользователям.
- В настоящее время использование мини-ЭВМ сокращается. На смену им приходят ЭВМ других типов: серверы, обеспечивающие диспетчерские функции в сетях ЭВМ, средние ЭВМ или старшие модели персональных ЭВМ (ПЭВМ).
- Третье направление связано с применением ЭВМ для решения задач искусственного интеллекта. Примеров подобных задач много: задачи робототехники, доказательства теорем, машинного перевода текстов с одного языка на другой, планирования с учетом неполной информации, составления прогнозов, моделирования сложных процессов и явлений и т.д.

Вопрос 1.2.2 Классификация средств вычислительной техники.

В настоящее время выпускает в основном три класса ЭВМ.

- Большие ЭВМ (mainframe), которые представляют собой многопользовательские машины с центральной обработкой, с большими возможностями для работы с базами данных, с различными формами удаленного доступа.
- Средние ЭВМ, предназначенные в первую очередь для работы в финансовых структурах. В этих машинах особое внимание уделяется сохранению и безопасности данных, программной совместимости и т.д. Они могут использоваться в качестве серверов в локальных сетях.

- Персональные ЭВМ.

Кроме перечисленных типов вычислительной техники, необходимо отметить класс вычислительных систем, получивший название "суперЭВМ". Особенно эффективно применение суперЭВМ при решении задач проектирования, в которых натурные эксперименты оказываются дорогостоящими, недоступными или практически неосуществимыми. В этом случае ЭВМ позволяет методами численного моделирования получить результаты вычислительных экспериментов, обеспечивая приемлемое время и точность решения. Дальнейшее развитие суперЭВМ связывается с использованием направления массового параллелизма, при котором одновременно могут работать сотни и даже тысячи процессоров. Образцы таких машин уже выпускаются несколькими фирмами: nCube (гиперкубические ЭВМ), Connection Machine, Mass Par, NCR/Teradata, KSR, IBM RS/6000, MPP и др.

Необходимо отметить и еще один класс наиболее массовых средств ЭВТ - встраиваемые микропроцессоры. Успехи микроэлектроники позволяют создавать миниатюрные вычислительные устройства, вплоть до однокристальных ЭВМ. Эти устройства, универсальные по характеру применения, могут встраиваться в отдельные машины, объекты, системы. Таким образом, можно предложить следующую классификацию средств вычислительной техники, в основу которой положено их разделение по быстродействию.

- СуперЭВМ для решения крупномасштабных вычислительных задач, для обслуживания крупнейших информационных банков данных.
- Большие ЭВМ для комплектования ведомственных, территориальных и региональных вычислительных центров.
- Средние ЭВМ широкого назначения для управления сложными технологическими производственными процессами. ЭВМ этого типа могут использоваться и для управления распределенной обработкой информации в качестве сетевых серверов.
- Персональные и профессиональные ЭВМ, позволяющие удовлетворять индивидуальные потребности пользователей. На базе этого класса ЭВМ строятся автоматизированные рабочие места (АРМ) для специалистов различного уровня.
- Встраиваемые микропроцессоры, осуществляющие автоматизацию управления отдельными устройствами и механизмами.

Высокие скорости вычислений, обеспечиваемые ЭВМ различных классов, позволяют перерабатывать и выдавать все большее количество информации, что, в свою очередь, порождает потребности в создании связей между отдельно используемыми ЭВМ. Поэтому все современные ЭВМ в настоящее время имеют средства подключения к сетям связи и комплексирования в системы.

Контрольные вопросы

1. Укажите два типа ЭВМ. Поясните их особенности.
2. Перечислите глобальные сферы деятельности человека, которые требуют использования качественно различных типов ЭВМ.
3. Какова классификация СВТ в зависимости от быстродействия?

Тема 1.5. Математические и логические основы ЭЦВМ

Цель изучения темы: изучить принцип работы ЭВМ; усвоить арифметические основы ЭВМ; усвоить логические основы ЭВМ; построить схему по заданной логической функции, в заданном базисе.

Вопросы.

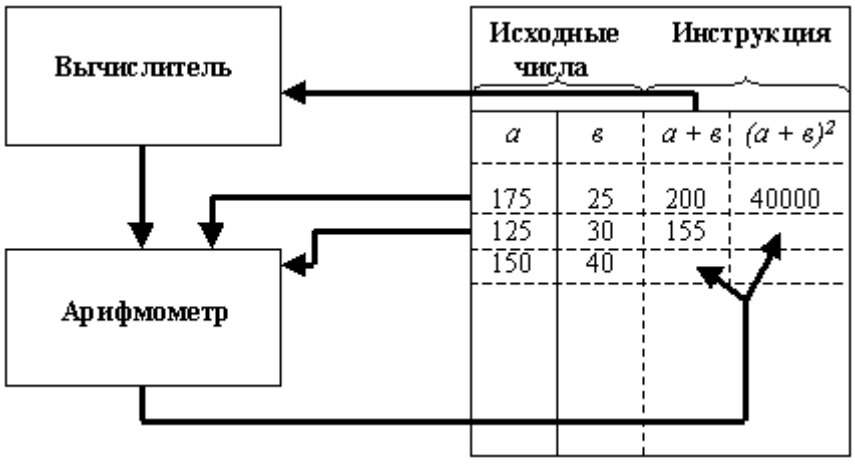
1. Построение ЭЦВМ. Структурная схема ЭЦВМ.
2. Этапы решения задачи на ЭВМ.
3. Системы счисления. Арифметические операции в двоичной с.ч.. Перевод чисел из одной системы счисления в другую.
4. Представление чисел в ЭВМ.
5. Общие сведения о представлении других видов информации.

1.5.1. Построение ЭЦВМ. Структурная схема ЭЦВМ

Принцип действия ЭЦВМ резко отличается от принципа действия АВМ. Если термин "аналоговое вычислительное устройство" относится к устройству, действие которого характеризуется непрерывными процессами, то термин "цифровое вычислительное устройство" относится к устройству, которое работает циклически и оперирует с дискретными величинами.

ЭЦВМ автоматически выполняют всю совокупность операций, предусмотренных численными методами решения математических задач. Их работа аналогична работе группы вычислителей. Работа расчетчика может быть пояснена схемой, приведенной на рис.

Рисунок 1.5.1. Схема ручной обработки информации.



Расчетчик, получая необходимые исходные числа записывает их на листе бумаги. Правила обработки исходной информации записываются в виде букв, символов, знаков, алгебраических операций.

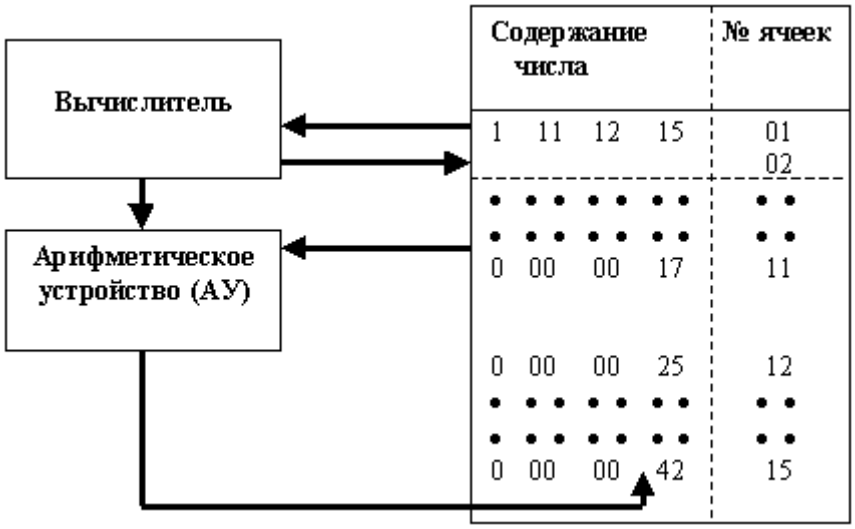
Расчетчик читает числа в графах расчетных листов, вводит их с помощью клавиатуры в счетную машину, выполняет операции, указанные в расчетных листах. Результаты операции расчетчик записывает в определенную, предназначенную для этой цели графу.

Таким образом, вычислительный процесс, осуществляемый вручную, характеризуется:

1. Записью (хранением) цифровых данных и промежуточных результатов в соответствующих графах расчетного листа.
2. Обработкой информации, т.е. выполнением вычислений, осуществляемых на счетной машине.
3. Управлением процессом обработки, т.е. принятием решения о реализации на данном этапе той или иной операции.

То же самое характеризует вычислительный процесс при использовании ЭЦВМ. Связь основных блоков ЭЦВМ наглядно показана на рис.

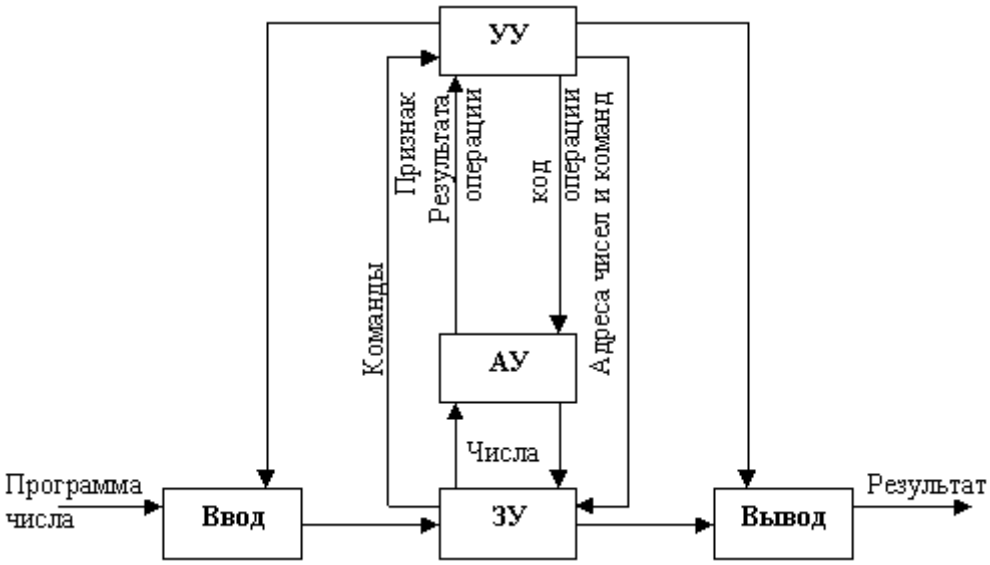
Рисунок 1.5.2. Схема автоматической обработки информации.



При сравнении схемы автоматической обработки информации и схемы ручной обработки информации, видно, что расчетный лист здесь заменяет запоминающее устройство (ЗУ) . Ячейки ЗУ соответствуют квадратикам листа бумаги вычислителя. ЗУ связано двухсторонней связью с арифметическим устройством (АУ). Для координации работ, проводимых на ЭЦВМ, необходимо устройство управления (УУ). Выполнение каждой отдельной команды должно происходить в два этапа: во-первых, команда должна быть извлечена из памяти и помещена в специальное устройство УУ, во-вторых, она должна интерпретирована и расшифрована, чтобы на ее основании могли быть сформированы управляющие сигналы, на АУ и ЗУ с целью выполнения определенных элементарных операций.

С учетом выше изложенного структурная схема ЭЦМ приведена на рис.

Рисунок 1.5.3. Структурная схема ЭЦВМ



Вопрос 1.5.2. Этапы решения задачи на ЭВМ

При решении любой задачи исходная информация, введенная в ЗУ, подвергается обработке по предварительно составленной программе. Поэтому ЭВМ являются программно управляемыми машинами. Составление программы для решения какой либо задачи называется программированием. Процесс решения задачи условно можно разбить на несколько этапов.

Необходимо вычислить численное значение величины:

```
x=(a2+b*c)/d-k,
где a=7240.76; b=385.64; c=64.93; d=479.67; r=17890.72.
```

В рассматриваемом примере само условие поставленной задачи является математической формулировкой. Алгоритм решения задачи:

1. Вычислить a2 = a*a;
2. Найти произведение bc = b*c;
3. Определить значения числителя: a2+b*c;
4. Найти значение дроби, для чего числитель разделить на d: (a2+b*c)/d;
5. Вычесть из дроби значение k, получив таким образом искомое значение x: x=(a2+b*c)/d-k;

Весь ход вычисления делится на ряд последовательных простейших операций, для каждой из которых составляется отдельная команда. Команда представляет собой условный набор цифр (код), определяющий действие машины для выполнения одной операции. Иначе говоря, командой

называется машинное слово, предназначенное для управления работой ЭВМ при выполнении одной определенной операции.

Команды программы записываются в ячейки ЗУ, начиная с номера 0100, а исходные данные - в ячейках, начиная с номера 0200.В рассматриваемом примере распределение памяти может быть представлено в следующем виде:

```
0200) A 0205
0201) B 0206
0202) C 0207 Номера рабочих ячеек
0203) D 0210
0204) K 0211
```

Совокупность операций, выполняемых машиной, называется системой команд. В рассматриваемом примере используется только часть кодов операций из системы команд ЭВМ:

```
01 - сложение;
02 - вычитание;
03 - умножение;
04 - деление;
33 - останов машины.
```

Первая команда должна обеспечить получение A2 умножением A на A. Записывается она следующим образом:

```
03 0200 0200 0205
```

Здесь 03 - код операции умножения;

0200 - первый адрес команды, указывающий номер ячейки, в которой находится первый сомножитель (A);

0200 - второй адрес команды, указывающий номер ячейки, в которой находится второй сомножитель (A);

0205 - второй адрес команды, указывающий номер ячейки, в которую должен быть послан результат выполнения операции (A2).

Переведённая команда читается так: "Умножить число, находящиеся в ячейке 0200, на содержимое ячейки 0200 (т.е. само на себя) и результат операции послать в ячейку 0205".

Вторая команда составляется аналогично:

```
03 0201 0202 0206
```

Читается она так: "Умножить число, находящиеся в ячейке 0201 (B), на содержимое ячейки 0202 (C), и результат операции послать в ячейку 0206 (BC)".

Запишем составляемую программу в виде таблицы, помня, что мы условились команды хранить в ячейках, начиная с номера 0100.

Рисунок 1.5.4.

Номер команды	Код операций	Первый адрес	Второй адрес	Третий адрес	Пояснения
0100	03	0200	0200	0205	A^2=(A*A) 0205
0101	03	0201	0202	0206	BC=(B*C) 0206
0102	01	0205	0206	0207	(A^2+BC) 0207
0103	04	0207	0203	0210	(A^2+BC/D) 0210
0104	02	0210	0204	0211	X=(A^2+BC/D)-K
0105	33	0000	0000	0000	Останов

Ответ на решённую задачу находится в ячейке 0211.

Команда по своей структуре имеет операционную и адресную части. Операционная часть содержит код выполняемой операции, а адресная часть - адреса чисел, участвующих в операции.

Сама операция представляет собой переработку информации, совершаемой машиной под воздействием одной команды. Числа, участвующие в операции, называются операндами.

Программа является последовательностью команд. Если её сравнить с алгоритмом решения задачи, приведённом в описании второго этапа программирования, то станет очевидным, что программа представляет собой не что иное, как описание алгоритма решения задачи на языке данной машины (машинном языке).

Отладка программы является процессом проверки правильности составления программы и её пригодности для выполнения на машине. При отладке программа вводится в память машины; в ходе выполнения отдельных команд и частей программы обнаруживаются и устраняются допущенные ошибки.

Структура команды ЭВМ, использованная при рассмотрении принципа программного управления машиной, состоит из следующих элементов:

```
КОП А1 А2 А3
```

Где КОП - код операции, А1 - адрес первого операнда, А2 и А3 - соответственно адрес второго операнда и результата выполненной операции.

Эта команда является трёхадресной. ЭВМ, работающие по таким командам, называются трёхадресными.

Если результат операции записывается в ячейку ОЗУ, указанную во втором адресе -А2, или оставляется в сумматоре, то команда становится двух адресной:

```
КОП А1 А2
```

Если результат операции всегда остается в сумматоре и арифметические действия производятся над числами, одно из которых предварительно водится в

сумматор, а другое находится в ячейке ОЗУ, адрес которой указан в команде, то такая команда становится одноадресной: КОП А.

Вопрос 1.5.3. Системы счисления (с. с.). Арифметические операции в двоичной, восьмеричной с.с. Перевод чисел из одной системы счисления в другую

Совокупность приемов наименования и обозначения чисел называют **системой счисления**.

Исторически сложились два типа систем счисления: непозиционные и позиционные.

Непозиционная - это такая система счисления, у которой количественное содержание цифры определяется только ее графическим обозначением. Римская система счисления является непозиционной.

Позиционная - это такая система счисления, в которой один и тот же цифровой знак имеет различное количественное содержание в зависимости от его местоположения (позиции) в последовательности цифр.

Принцип записи в системах этого типа одинаков и состоит в следующем:

1. Число записывается как последовательность цифр;
2. Целая часть числа отделяется от дробной десятичной запятой (точкой);

Цифра, стоящая слева от запятой, показывает количество единиц, следующая - количество десятков, сотен и т. д.; вообще каждая цифра целой или дробной

части имеет значение в 10 раз большее, чем та же цифра на предыдущем (ближайшем справа) месте.

В десятичной системе счисления принято десять различных цифр: 0, 1,2, 3, 4, 5, 6, 7, 8 и 9, и запись 1920,47 означает: 1*103 +9*102 + 2*10 1 +0*100 +4*10 -1 + 7*10 -2.

Количество попарно различных цифр, применяемых в позиционной системе счисления, называют ее **основанием.**

В двоичной системе применяются две цифры 0, 1, число 2 записывается как 10, а следующие числа изображаются как 11, 100, 101, 110 и т. д.; в восьмеричной системе счисления восемь различных цифр: О, 1, 2, 3, 4, 5, 6, 7, а основание системы записывается в виде 8. В шестнадцатеричной системе шестнадцать цифр: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Арифметические операции в двоичной системе счисления

Сложение одноразрядных чисел производится в соответствии с равенствами:

```
0 + 0 = 0; 1 + 0 = 1;0 + 1; = 1; 1 + 1 = 10
```

Пример

Выполнить сложение

1. 1011011+110101=10010000
2. 101,11101 + 1,10011=111.10011

Вычитание в двоичной системе счисления выполняются по правилам: 0 - 0 = 0;

```
1 - 0 = 1; 1 - 1 = 0; 10 - 1 = 1
```

Пример

Найти разность чисел:

1. 1001011-110110=10101
2. 11,00101-1,11011= 1,01010

Умножение в двоичной системе счисления выполняются по правилам: 0 * 0 = 0;

```
1* 0 = 1; 1*1 = 1; 0*1 = 0
```

Пример.

Перемножить два числа: 100111,1 * 10,11=1101100,101.

Деление двоичных чисел выполняется с использованием таблиц умножения и вычитания.

Пример Найти частное с точностью до 0,001: 1101101: 1011= 1001,1110? 1001,111

Двоично-десятичная запись числа. При обработке больших массивов экономической информации переводы чисел из десятичной системы в двоичную и обратно требуют значительного машинного времени. Некоторые образцы ЭВМ имеют специальные блоки, которые обрабатывают десятичные целые числа в их двоично-десятичном представлении. Каждая десятичная цифра заменяется равным ей четырехразрядным двоичным число - двоичной тетрадой - в соответствии с таблицей:

```
010 = 00002/10
510 = 01012/10
110 = 00012/10
610 = 01102/10
210 = 00102/10
710 = 01112/10
310 = 00112/10
810 = 10002/10
410 = 01002/10
910 = 10012/10
```

Например:

```
398,1410 = 0011 1001 1000, 0001 01002/10
```

Перевод чисел из одной системы счисления в другую. Вся информация (команды, числа, константы) в ЗУ хранится в двоичной системе, операции над ними выполняются по правилам двоичной арифметики. Поэтому числа необходимо переводить из десятичной системы в двоичную, а результаты машинных вычислений из двоичной - в десятичную. В ЭВМ используется три позиционных системы счисления - двоичная, восьмеричная, десятичная (в машине ее двоично-десятичная запись). Существует два основных способа перевода.

Способ подстановки степени основания

Этот способ перевода p-ичной системы в q-ичную использует основные понятия позиционной записи чисел. Каждая p-ичная цифра и соответствующая ей степень основания переводятся в q-ичную систему, а

затем определяются q-ичные цифры путем выполнения арифметических операций умножения и сложения по q-ичным правилам:

Пример

Перевести из двоичной системы в десятичную

101101,1012=1*25+0*24+1*23+1*22+0*21+1*2-1+0*2-2+1*2-3=
32+0+8+4+0+1+1/2+0+1/8=45,62510

Перевести из десятичной системы в восьмеричную

95,410=9*101+5*100+4*10-1=118*1218+58*1208+4*12-
1=132+5+4/12=(137,4)8

Способ деления - умножения на основание системы

Для перевода целой и дробной части чисел используются разные приемы.

Перевод целых чисел.

Пусть необходимо перевести число N из p-ичной системы в q-ичную, где q - запись основания новой системы в p-ичной системе.

Правило перевода целых чисел следующее: для перевода целого числа из одной позиционной системы счисления (p-ичной) в другую (q-ичную) надо его, а затем целые частные последовательно делить нацело на основание системы q, в которую число переводится, по правилам исходной системы до получения частного, меньшего q. Число в новой системе получается как последовательность остатков от деления начиная с последнего. Последний остаток дает старшую цифру.

Пример

Перевести 34810 в 8-ную систему счисления: 348:8=43 остаток 4; 43:8=5 остаток 3; 34810 = 5348

Пример

Перевести 34810 в 16-ную систему счисления: 348:16=12 остаток 12; 21:16=1остаток 5;

34810 = 51C16

Пример

Перевести 1110012 в 10-ую систему счисления: 111001: 1010=101 остаток 111;

```
1110012=1011112
1110012=5710
```

Перевод дробных чисел

Пусть теперь M - правильная p-ичная дробь и необходимо перевести ее в q-ичную систему счисления.

Чтобы перевести правильную дробь из одной системы счисления (p-ичной) в другую (q-ичную), ее надо последовательно умножать на основание системы, в которую переводят (q). Дробь записывается как последовательность целых чисел получающихся произведений, начиная с первого.

Пример

Перевести число 0,72510 в 8-ную систему. 0.725*8=5.800;

```
0.800*8=6.400; 0.400*8=3.200; 0.200*8=1.600.
Результат 0,72510 = 0,5638
```

Пример

Перевести 0,72510 в 2-ную систему: 0.725*2=1.450; 0.450*2=0.900; 0.900*2=1.800; 0.800*2=1.600; 0.600*2=1.200; 0.200*2=0.400. Результат 0.72510 = 0.101112

Перевод смешанных чисел

Для перевода смешанного числа из одной системы счисления в другую переводят отдельно целую и дробную часть по соответствующим правилам и записывают в новой системе, разделяя десятичной запятой.

Пример

Перевести 79.6510 в 8-ную с.с.. 7910=1178; 0.6510=0.5158. Результат 79.6510 = 117.5158

Перевод чисел из восьмеричной системы в двоичную и обратно.

Чтобы перевести число из восьмеричной системы счисления в двоичную надо каждую ее цифру целой и дробной частей заменить равным ей трехзначным двоичным числом (8 = 23). И наоборот: чтобы перевести число из двоичной системы в восьмеричную, нужно разбить ее влево и вправо от запятой на группы по три цифры (двоичные триады) и каждую из них

заменить равной ей восьмеричной цифрой; если крайние левая и правая триады окажутся неполными, их дополняют соответственно слева и справа нулями - одними или двумя.

Пример

Перевести 00101012 в 8-ную с.с.. 00101012 = 001 111 010 011, 001 010 1002 = 1723,1248.

Пример

Перевести 968,51210 в 2-ную с.с.. 968,51210 = 1710,4068 = 001 111 001000,1000001102.

Вопрос 1.5.4. Представление чисел в ЭВМ. Арифметические операции в ЭВМ с фиксированной и с плавающей запятой

В ЭВМ числа хранятся в ячейках ЗУ. Ячейки памяти ЭВМ представляют собой набор разрядов, в каждом из которых может храниться двоичная цифра 0 или 1. В зависимости от способа представления чисел в виде числовых кодов различают машины с фиксированной и плавающей запятой.

В ЭВМ с фиксированной запятой используется естественная форма записи, при которой цифры числа записывают в строку, а целая часть числа от дробной отделяется запятой.

В ячейке памяти машины с фиксированной запятой один разряд, как правило, первый справа предназначается для записи знака числа, а остальные - цифровые. Некоторое постоянное число их отводится для записи целой части, оставшиеся - для хранения дробной части числа. Для изображения знака числа используются двоичные цифры: "+" = 0; "-" = 1.Если записывается число положительное, в знаковый ряд заносится нуль, и единица - если число отрицательное.

Во время исполнения арифметических операций могут получаться результаты, целая часть которых не умещается в отведенных для нее разрядах. Это явление называется переполнением разрядной сетки.

В ЭВМ с плавающей запятой используется так называемая полулогарифмическая, или нормальная, запись чисел. При этом каждое число характеризуется двумя величинами - мантиссой и порядком - и записывается в виде:

```
x = m*10p,
где |m|<1, а p - целое число.
```

Мантисса показывает последовательность цифр числа x, а порядок - положение запятой в числе.

Например, число 175,410 можно записать в виде: 175,4 = 0,1754*108,

здесь m = 0,1754, p = +3. Для числа 0,00029 =0,029*10-2 = 0,29*10-3. Для каждого числа может быть найдено множество нормальных форм. Чтобы повысить точность представления чисел, в машинах используется нормализованная форма, в которой на мантиссу накладывается еще одно требование - первая ее цифра после запятой должна быть значащей, так что мантисса удовлетворяет неравенствам:

$0,1<|m|<1$

В некоторых ЭВМ в начале ячейки размещается мантисса, за ней порядок, в других они меняются местами. Знаки мантиссы и порядка кодируются так же, как в машинах с фиксированной запятой ("+" = 0; "-" = 1).

Диапазон чисел, представляемых в машинах с плавающей запятой, намного шире, чем диапазон чисел, представляемых в ЭВМ с фиксированной запятой. Определим его для разрядной сетки с n разрядами для порядка и m разрядами для мантиссы в двоичной системе:

Диапазон чисел определяется количеством разрядов, выделенных для записи порядка: чем больше величина m, тем шире диапазон. Длина же мантиссы никак не влияет на диапазон.

Если в результате выполнения арифметической операции получается порядок, не умещающийся в отведенных для него разрядах ячейки, в машинах с плавающей запятой вырабатывается сигнал переполнения разрядной сетки. При переполнении мантиссы такого сигнала не вырабатывается, а производится автоматическая нормализация результата.

Современные ЭВМ имеют оба режима выполнения операций или только режим с плавающей запятой.

Во всех ЭВМ без исключения все операции выполняются над числами представленными специальными машинными кодами. Их использование позволяет обрабатывать знаковые разряды чисел, а также заменять операцию вычитания операцией сложения.

Различают прямой код, обратный код, дополнительный код . Эти коды определяются для двоичных дробных чисел вида x = ? 0 x1x2...xn.

Прямой, обратный и дополнительный коды числа x обозначаются соответственно [x]пр, [x]обр, [x]доп.

Для неотрицательных чисел все три кода имеют одинаковый вид и совпадают с записью самого числа: X=[x]пр=[x]обр=[x]доп для x? 0

Прямой код двоичного числа образуется из абсолютного значения этого числа и кода знака числа (нуль или единица) перед его старшим числовым разрядом.

Пример

A10= +10 A2= +1010 [A]пр= 0 : 1010. B10= -15 B2=-1111 [B]пр= 1 : 1111.

Обратный код отрицательного числа содержит единицу в знаковом разряде числа, а значащие разряды числа заменяются на ирверсные, т.е. нули заменяются единицами, а единицы нулями.

Пример

A10= +5, A2= +101 [A2]пр=[A2]обр=0 : 101; B10= -13 B2=-1101 [B2]обр=1 : 0010.

Дополнительный код отрицательного числа представляет собой результата суммирования обратного кода числа с единицей младшего разряда.

Пример

A10= +19 A2= +10011 [A2]пр=[A2]обр=[A2]доп=0 : 10010 B10= -13 B2= -1101 [B2]доп=1 : 0011.

Модифицированные обратные и дополнительные коды отличаются соответственно от обратных и дополнительных кодов удвоением значений знаковых разрядов. Знак "+" кодируется двумя нулевыми знаковыми разрядами, а знак "-"- двумя единичными разрядами.

Пример

```
A10= +9 A2= +1001 [A2]пр=[A2]обр=[A2]доп=0 : 1001;
[A2]мобр=[A2]мдоп=00 : 1001;
B10= -9 B2= -1001 [B2]доп=1 : 0111 [B2]обр=1 : 0110.
[B2]мдоп=11: 0111 [B2]мобр=11 : 0110.
```

Целью введения модифицированных кодов являются фиксация и обнаружение перполнения разрядной сетки. В этом случае перенос из значащего разряда может исказить значение младшего знакового разряда. Значение знаковых разрядов "01" свидетельствует о положительном переполнении разрядной сетки, "10" свидетельствует об отрицательном переполнении разрядной сетки.

Арифметические операции в ЭВМ с фиксированной запятой. Сложение чисел в АУ ЭВМ заменятся сложением модифицированных обратных или модифицированных дополнительных кодов в сумматорах соответствующего типа.

Контроль за переполнением осуществляется с помощью знаковых разрядов результатов: если они одинаковы, т.е. принимают значения 00 или 11, то результат - число нормальное, если знаковые разряды разные - произошло переполнение, причем при положительном результате в знаковых разрядах получается 01, при отрицательных - 10.

В связи с тем, что крайний слева разряд всегда равен нулю для положительных чисел и единице - для отрицательных (независимо от того, было переполнение или нет), он служит для определения знака результат и потому называется знаковым разрядом.

Следующий разряд совпадает со знаковым, если число по модулю меньше 1, и отличается от него, если число по модулю больше или равно 1. Этот разряд называется **разрядом переполнения**.

По команде "сложить два числа" машина выбирает оба слагаемых из ЗУ по указанным в команде адресам. В ячейках памяти числа хранятся в прямом коде, поэтому при передаче в регистры АУ, где будет производиться операция сложения, прямой код преобразовывается в модифицированный обратный (или дополнительный) код. Результат в сумматоре АУ получается в модифицированном обратном (или дополнительном) коде, и если его надо хранить в памяти машины, то по пути в ячейку ЗУ этот код преобразовывается в прямой.

Пример

Сложить числа x и y, используя сумматор с блокировкой переноса, если

Операция вычитания в ЭВМ сводится к сложению путем изменения знака вычитаемого на противоположный, так что вычисление разности x - y сводится к определению алгебраической суммы x + (-y).

В АУ ЭВМ умножение выполняется над прямыми кодами сомножителей; знак произведения получается путем сложения знаковых разрядов множимого и множителя в специальном одноразрядном сумматоре знака по следующим правилам: 0+0 =0;

`1+0 =1; 0+1=1; 1+1=0`

Таким образом, знак минус, как и в алгебре, получается, если у сомножителей разные знаки, и плюс - если одинаковые.

Операция деления основана на ручном способе деления двоичных чисел. Для получения очередной цифры частного из промежуточного делимого вычитается делитель и, если разность отрицательна - записывается нуль и восстанавливается промежуточное делимое, а затем происходит сдвиг вправо на один разряд (или остатка - влево) и процесс повторяется. Операция выполняется в прямом коде.

При делении в том случае, если модуль делимого больше модуля делителя (и, следовательно, частное получится больше 1), произойдет переполнение разрядной сетки. В АУ вырабатывается сигнал переполнения. Знак частного получается, как и при умножении, в одноразрядном сумматоре.

Арифметические операции в ЭВМ с плавающей запятой

При выполнении арифметических действий в машинах с плавающей запятой операции производятся над мантиссами и порядками операндов, поэтому АУ этих ЭВМ сложнее по конструкции, чем АУ машин с фиксированной запятой. Каждая операция выполняется как бы в несколько этапов, результат ее автоматически нормализуется.

Вопрос 1.5.5. Общие сведения о представлении других видов информации

Информация - это сведения об окружающем мире и протекающих в нем процессах, воспринимаемых человеком или специализированным устройством, например ЭВМ, для обеспечения целенаправленной деятельности. Информация может быть по своей физической природе: числовой, текстовой, графической, звуковой, видео и др. Она также может быть постоянной (неменяющейся), переменной, случайной, вероятностной. Представление информации может быть непрерывным и дискретным.

К важнейшим свойствам информации следует отнести: полноту, достоверность, ценность, актуальность, ясность. Информация передается с помощью языков. Основа любого языка - алфавит.

Алфавит - это конечный набор знаков (символов) любой природы, из которых конструируются сообщения на данном языке.

Алфавит может быть латинский, русский, десятичных чисел, двоичных и т.д. Кодирование - представление символов одного алфавита символами другого. Любая информация, обрабатываемая, в ЭВМ, кодируется в двоичном алфавите. Двоичный алфавит состоит из двух символов 0 и 1. Величина способная принимать лишь два значения (0 или 1), называется **бит.**

Бит - это минимальная единица количества информации

Количество информации измеряется в **битах** и **байтах** и кратных им величинах.

```
1 байт = 8 бит,
1 Килобайт(Кб)=1024 байта,
1 Мегабайт(Мб)=1024 Кб,
1 Гигабайт= 1024 Мб.
```

Различные виды информации могут быть разделены на две группы: статические и динамические. Так, числовая, логическая и символьная информация является статической - ее значение не связано со временем. Вся аудиоинформация имеет динамический характер.

Видеоинформация может быть как статической, так и динамической. Статическая видеоинформация включает текст, рисунки, графики, чертежи, таблицы и др.. Рисунки делятся также на плоские - двухмерные и объемные - трехмерные.

Динамическая видеоинформация - это видео-, мульт-, слайд-фильмы. В их основе лежит последовательное экспонирование на экране в реальной масштабе времени отдельных кадров в соответствии со сценарием.

Динамическая видеоинформация используется либо для передачи движущихся изображений (анимация), либо для последовательной демонстрации отдельных кадров (слайд-фильмы).

Для демонстрации анимационных и слайд фильмов используются различные принципы. Анимационный фильм демонстрируется так, чтобы зрительный аппарат человека не мог зафиксировать отдельных кадров. В современных высококачественных мониторах кадры сменяются до 70 раз в секунду, что позволяет высококачественно передавать движение объектов.

При демонстрации слайд-фильмов каждый кадр экспонируется на экране столько времени, сколько необходимо для восприятия его человеком. Слайд-фильмы можно отнести к статической видеоинформации.

По способу формирования видеоизображения бывают растровые, матричные и векторные.

Растровые видеоизображения применяются используются в телевидении, а в ЭВМ практически не применяются.

Матричные изображения получили в ЭВМ наболее широкое распространение. Изображение на экране рисуется электронным лучом точками.

Информация представляется в виде характеристик значений каждой точки -пикселя (picture element), рассматриваемого как наименьшей структурной единицей изображения. Количество одновременно высвечиваемых пикселей на экране определяется его разрешающей способностью. В качестве характеристик графической информации выступают: координаты точки (пикселя) на экране, цвет пикселя, цвет фона (градация яркости).Вся эта информация хранится в видеопамяти дисплея. При выводе графической информации на печать изображение также выводится по точкам.

Изображение может быть и в векторной форме. Тогда оно составляется из отрезков линий, для которых задаются: начальные координаты, угол наклонна и длинна отрезка.

Векторный способ имеет ряд преимуществ перед матричным: изображение легко масштабируется с сохранением формы, является "прозрачным" может быть наложено на любой фон.

Для кодирования символьной и текстовой информации применяются различные системы : при вводе информации с клавиатуры кодирование происходит при нажатии клавиши, на которой изображен требуемый символ, при этом в клавиатуре вырабатывается так называемый scan-код, представляющий собой двоичное число, равное порядковому номеру клавиши Номер нажатой клавиши никак не связан с формой символа, нанесенного на клавише. Опознание символа и присвоение ему внутреннего кода ЭВМ производится специальной программой по специальным кодовым таблицам. Дисплей по каждому коду символа должен вывести на экран изображение символа. Описание формы каждого символа хранится в специальной памяти дисплея - знакогенераторе.

Высвечивание символа на экране дисплея осуществляется с помощью точек, образующих символьную матрицу. Каждый пиксель в такой матрице является элементом изображения и может быть ярким или темным. Темная точка кодируется цифрой 0, светлая - 1.

Аудиоинформация является аналоговой. Кодирование аудиоинформации - процесс сложный. Для преобразования ее в цифровую форму используются аппаратные средства: аналого-цифровые преобразователи (АЦП), в результате работы которых аналоговый сигнал оцифровывается - представляется в виде цифровой последовательности. Для вывода оцифрованного звука на аудиоустройство необходимо проводить обратное преобразование, которое осуществляется с помощью цифро-аналоговых преобразователей (ЦАП).

Контрольные вопросы

1. Опишите схемы ручной, автоматической обработки информации.

2. Опишите структурную схему ЭВМ.
3. Опишите этапы решения задачи на ЭВМ. Какова структура команды в ЭВМ.
4. Каково понятие система счисления, позиционная с.с., непозиционная с.с., основание с.с.
5. Опишите правила арифметический операций в 2-ной с.с., в 8-ной с.с.
6. Перечислите формы и коды представления чисел в ЭВМ.
7. Опишите правила выполнения арифметических операций в ЭВМ фиксированной и с плавающей запятой.
8. Каково понятие информации? Опишите способы представления информации Перечислите и поясните свойства информации.
9. Каково поняти алфавит, кодирование? Опишите способы формирования видеоизображения в ЭВМ.

Тема 1.6. Элементная база ЭВМ

Цель занятия:

- изучить элементную базу ЭВМ.

Вопросы.

1. Логические основы ЭВМ.

Вопрос 1.6.1. Логические основы ЭВМ

Теоретической основой построения ЭВМ являются специальные математические дисциплины. Одной из них является алгебра логики или Булева алгебра (Дж.Буль - английский математик прошлого столетия, основоположник алгебры логики). Ее аппарат широко используется для описания схем ЭВМ, их оптимизации и проектирования. Вся информация в ЭВМ представляется в двоичной системе счисления. Поставим в соответствие входным сигналам отдельных устройств ЭВМ соответствующие значения $x_i(i=1,n)$, а выходным сигналам - значение функций y_j (j=1,m) (рис. 1.6.1)

Рисунок 1.6.1.

В этом случае зависимостями $y_j=f(x_1, x_2,...,x_n)$ можно описывать алгоритм работы любого устройства ЭВМ. Каждая такая зависимость y_i является "булевой функцией, у которой число возможных состояний каждой ее независимой переменной равно двум", т.е. функцией алгебры логики, а ее аргументы определены на множестве {0,1}. Алгебра логики устанавливает основные законы формирования и преобразования логических функций. Она позволяет представить любую сложную функцию в виде композиции простейших функций.

Количество всевозможных функций N от n аргументов выражается зависимостью

$N=2^{2n}$

При n=1 зависимость дает N=4. Представим зависимость значений этих функций от значений аргументов x в виде специальной таблицы истинности.

Рисунок 1.6.2. Таблица функций от одной переменной

X \ Yj	**Y0**	Y1	Y2	Y3
0	0	1	0	1
1	0	1	1	0

Таблицы истинности получили такое название, потому что они определяют значение функции в зависимости от комбинации входных сигналов.

Функция y_2 повторяет значение аргумента x:

$y_2=x$.

Схема, реализующая зависимость $y_2=x$, называется повторителем.

Функция y_3 принимает значение противоположное значению аргумента x:

$y_3=\neg x$.

Схема, реализующая зависимость $y_3=\neg x$, называется инвертором и читается y_3 есть не(¬) x.

При n=2 , N=16, т.е. от двух переменных, можно построить шестнадцать различных функций.

Рисунок 1.6.3. Таблица функций от двух переменных

X1 X2 \ Yj	**Y0**	Y1	Y2	…	Y4	Y5	Y6	Y7	Y8	Y9	…..	Y15
00	0	1	0		0	1	0	1	1	0		
01	0	1	0	…	1	0	0	1	0	1	…	
10	0	1	1		1	0	0	1	0	1		
11	0	1	1		1	0	1	0	1	0		

Функция y_4 представляет собой функцию логического сложения - дизьюнкцию. Она принимает значение единицы, если значение единицы принимает хотя бы одна переменная x_1 или x_2:

$y_4=x_1x_2 \vee x_1x_2 \vee x_1x_2=x_1 \vee x_2$.

Читается: y_4 есть x_1 или($\vee$) x_2.

Функция y_5 является инверсной функцией по отношению к y_4:

$y_5=\neg y_4=\neg(x_1 \vee x_2)=\neg x_1 * \neg x_2$.

Функция y_5 называется "отрицание дизьюнкции" или "стрелка Пирса ". Читается:

y_5 есть не($\neg$) x_1 и(*) не($\neg$) x_2.

Функция y_6 является функцией логического умножения. Функция y_6 принимает значение единицы в тех случаях, когда переменные x_1 и x_2 принимают значение единицы:

$y_6=x_1 * x_2$.

Читается: y_6 есть x_1 и(*) x_2.

Функция y_7 является инверсной функцией по отношению к y_6:

$y_7=\neg y_6=\neg(x_1 * x_2)=\neg x_1 \vee \neg x_2$.

Функция y_7 называется "отрицание коньюнкции" или " штрих Шеффера". Читается: y_7 есть не($\neg$) x_1 или($\vee$) не($\neg$) x_2.

Функция y_8 называется логической равнозначностью, она принимает значение единицы, если ее переменные имеют одинаковое значение (или 0 или 1):

$y_8=x_1*x_2 \vee x_1*x_2$.

Из перечисленных функций двух переменных можно строить сколь угодно сложные зависимости, отражающие алгоритмы преобразования информации, представленной в двоичной системе счисления. Алгебра логики устанавливает правила формирования логически полного базиса простейших функций, из которых могут строится любые более сложные. Наибольшее применение получили базисы: И, ИЛИ, НЕ; И-НЕ; ИЛИ-НЕ.

Законы алгебры логики.

Свойства:

x v 1=1

x v x=1

x v 0=x

x v x=x

x*0=0

x*1=x

x*x=0

x*x=x

x v x v x v ...v x=x

x*x* x*...*x=x

Законы:

Коммутативный $x_1*x_2=x_2*x_1$; $x_1vx_2=x_2vx_1$.

Ассоциативный $(x_1*x_2)*x_3=x_1*(x_2*x_3)$; $(x_1vx_2)vx_3= x_1v(x_2vx_3)$.

Дистрибутивный $x_1*(x_2vx_3)=x_1*x_2vx_1*x_3$; $x_1vx_2*x_3=(x_1vx_2)*(x_1vx_3)$.

Поглощения $x_1vx_1*x_2=x_1x_1*(x_1vx_2)=x_1$.

Склеивания $x_1*x_2vx_1*x_2=x_1(x_1vx_2)*(x_1vx_2)=x_1$.

Правило де Моргана $x_1vx_2=x_1*x_2x_1*x_2=x_1vx_2$.

Понятие о минимизации

Минимизация логических функций основана на применении законов склеивания и поглощения. Различают аналитический и табличный методы минимизации логической функции (ЛФ). Среди аналитических методов наиболее известным является метод Квайна-МакКласки, среди табличных методов - с применением диаграмм Вейча.

Пример 1.

Найти минимальную дизъюнктивную форму функции, заданной таблицей истинности с использованием диаграмм Вейча.

Таблица истинности функции $y=f(x_1,x_2,x_3)$

Рисунок 1.6.4.

X_1	X_2	X_3	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Рисунок 1.6.5. Диаграмма Вейча функции y

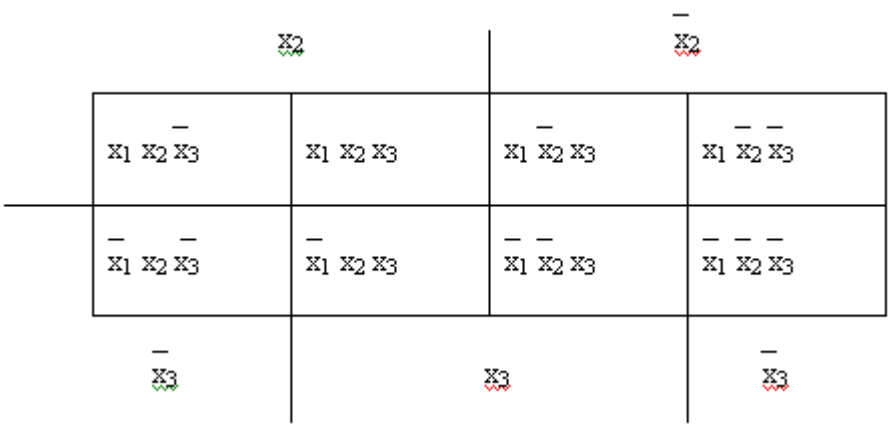
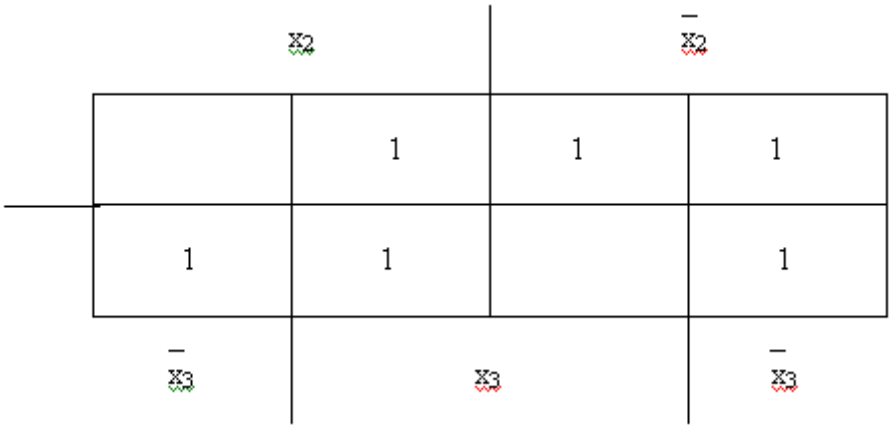


Диаграмма Вейча заполняется в следующем порядке:

1. Из таблицы истинности выбирают комбинации переменных, для которых функция принимает значение 1 - ($x_1x_2x_3$, $x_1x_2x_3$, $x_1x_2x_3$, $x_1x_2x_3$, $x_1x_2x_3$, $x_1x_2x_3$,);
2. В диаграмме Вейча в клетки с комбинациями переменных, для которых функция принимает значение 1, записывают 1.

3. Соседние клетки (две, четыре, восемь) с единицами объединяют и, принимая их за одну клетку, записывают комбинацию переменных, на пересечении которых стоит сформированная клетка. Например, при объединении двух нижних левых клеток формируется комбинация x_1 x_2, переменная x_3 склеивается. При объединении двух нижних крайних клеток - x_1 x_3 и т.д.

Рисунок 1.6.6. Диаграмма Вейча функции y



У данной функции существуют пять безызбыточных дизъюнктивных форм, из которых только две являются минимальными y_1 и y_4:

$y_1 = \neg x_1 x_2 \vee x_1 x_3 \vee \neg x_2 \neg x_3$

$y_4 = x_1 x_3 \vee x_2 x_3 \vee x_1 x_2$

Техническая интерпретация логических функций. По логическим выражениям проектируются схемы ЭВМ. При этом следует придерживаться следующей последовательности действий.

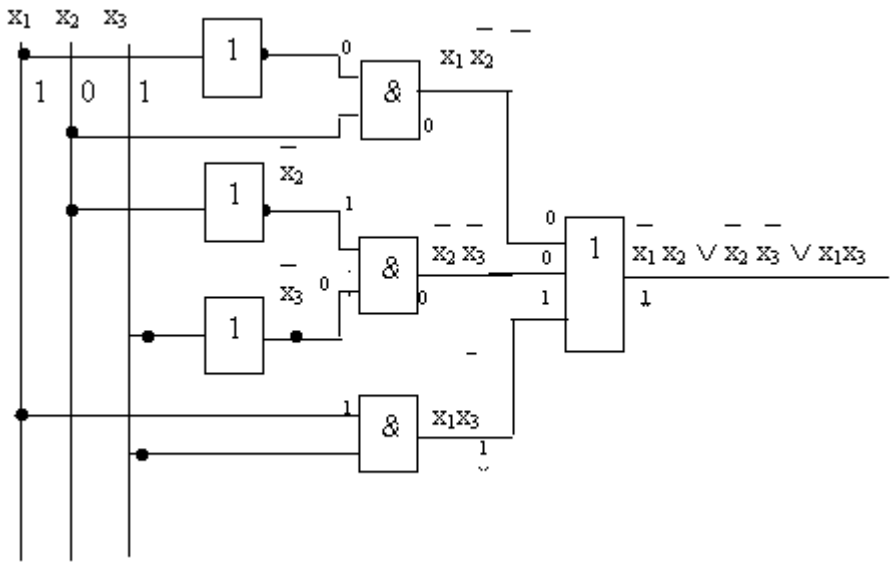
1. Словесное описание работы схемы.
2. Формализация словесного описания.
3. Запись функции в дизъюнктивной совершенной нормальной форме по таблицам истинности.
4. Минимизация логических зависимостей с целью их упрощения.
5. Представление полученных выражений в выбранном логически полном базисе элементарных функций.
6. Построение схемы устройств.

Выполнение пунктов 1,2,3,4,5 рассматривалось в примере 1. Выполнение пункта 6 рассматривается в примере 2.

Пример 2. Построить схему реализующую функцию y1.

Минимальная форма функции записана в базисе И,ИЛИ,НЕ. Для реализации функции потребуется один элемент ИЛИ на три входа, три элемента И на два входа, три элемента НЕ.

Рисунок 1.6.7.



Проверить работоспособность построенной схемы можно путем подачи на входы схемы значений переменных, заданных таблицей истинности функции и определения соответствия значений функции на выходе схемы и таблицы истинности. В данном примере выбрана комбинация $x_1=1$, $x_2=0$, $x_3=1$, в результате воздействия на выходе схемы формируется 1, что соответствует значению функции по таблице истинности.

Контрольные вопросы.

1. Перечислите и поясните основные операции алгебры логики.
2. Опишите порядок минимизации логической функции методом диаграмм Вейча.
3. Опишите последовательность проектирования схем ЭВМ по логическим выражениям.

Тема 1.7. Центральные устройства ЭВМ

Цель изучения темы:

- Изучить структурную схему оперативного запоминающего устройства;
- Изучить структуру и состав микропроцессора;
- объяснить работу микропроцессора при выполнении простейших программ.

Вопросы.

1. Основная память(ОП). Основные показатели. Классификация устройств ОП. Назначение ОЗУ, ПЗУ. Структурная схема ОЗУ. Основные характеристики ОЗУ.
2. Стековая память. Кэш-память первого, второго уровней. Размещение информации в ОП.
3. Эволюция микропроцессоров(МП). Основные технические характеристики МП. Классификация МП.
4. Структурная схема МП.
5. Полный цикл работы МП при выполнении команды.
6. Система команд МП.
7. Структурная схема базовой модели микропроцессора фирмы Intel.

Вопрос 1.7.1. Основная память(ОП). Структурная схема ОЗУ

Комплекс технических средств, реализующих функции памяти, называется запоминающим устройством (ЗУ). ЗУ делятся на основную память (ОП), сверхоперативную память (СОЗУ) и внешние запоминающие устройста (ВЗУ). ОП состоит из двух типов устройств :

- оперативное ЗУ (ОЗУ или RAM-Random Accses Memory);
- постоянное ЗУ (ПЗУ или ROM - Read Only Memory).

Основной составной частью микросхемы памяти является массив элементов памяти (ЭП). Каждый ЭП может хранить хранить 1бит информации. ЗУ, позволяющее обращаться по адресу к любому ЭП в произвольном порядке, называется ЗУ с произвольным доступом.

При матричной организации памяти адрес ЭП определяется двумя координатами X и Y. На пересечении этих координат находится ЭП. Структурная схема ОЗУ приведена на рис. 1.

Связь с ОЗУ осуществляется по системной магистрали (СМ). По шине управления передается сигнал, определяющий, какую операцию необходимо выполнить. ПО шине данных передается информация, записываемая в

память или считываемая из нее. По шине адреса передается адрес участвующих в обмене элементов памяти (поскольку данные передаются машинными словами, а один элемент памяти может воспринять только один бит информации, блок элементов памяти состоит из n матриц ЭП, где n-количество разрядов в машинном слове). Максимальная емкость памяти определяется количеством линий в шине адреса системной магистрали: если количество линий обозначить через m, то емкость памяти (т.е. количество элементов памяти, имеющих уникальных адреса) определяется как 2^m. При 32 разрядной адресной шине максимальный обьем ОП равен $2^{32} = 4Г6$.

В зависимости от способа хранения информации ОЗУ подразделяются на:

1. Стастические ОЗУ (SRAM - Static RAM).
2. Динамические (DRAM - Dynamic RAM).

Рисунок 1.7.1.

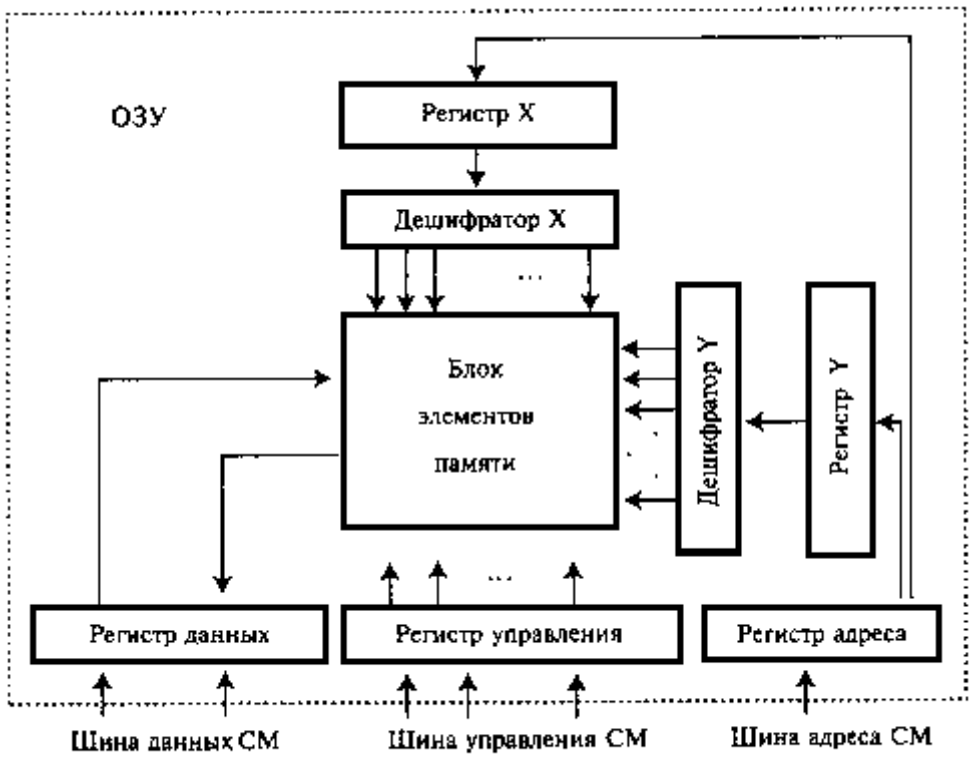


Рис. 5.1. Структурная схема ОЗУ

В статических ОЗУ в качестве ЭП используется статический триггер, который способен сохранять состояние 0 или 1 неограниченно долго (при включенном ПК). Динамические ОЗУ строятся на конденсаторах, реализованные внутри кристалла кремния. Динамические ЭП (конденсаторы) с течением времени саморазряжаются и записанная информация теряется. Динамические ЭП требуют периодического восстановления заряда - регенерации. Во время регенерации запись новой информации должна быть запрещена.

Основными характеристиками ОЗУ являются время доступа (быстродействие), емкость.

Время доступа- это промежуток времени, за который может быть записано или прочитано содержимое ячейки памяти после подачи ее адреса и соответствующего управляющего сигнала.

Емкость определяет количество ячеек или битов в устройстве памяти.

По сравнению со статическими, динамические ОЗУ имеют более высокую удельную емкость и меньшую стоимость, но большее энергопотребление и меньшее быстродействие. ОЗУ имеют модульную структуру. Увеличение емкости ОЗУ производится установкой дополнительных модулей. Время доступа к модулям DRAM составляет 60-70 нс.

На производительность ЭВМ влияет не только время доступа, но и такие параметры, как тактовая частота и разрядность шины данных системной магистрали. Тесли тактовая частота недостаточна высока, то ОЗУ простаивает в ожидании обращения. При тактовой частоте, превышающей возможности ОЗУ, в ожидании будет находиться системная магистраль, через которую поступил запрос в ОЗУ.

Интегральной характеристикой производительности ОЗУ с учетом частоты и разрядности шины данных является **пропускная способность**, которая измеряется в Мегабайтах в секунду. Для ОЗУ с временем доступа 60-70 нс и разрядностью шины данных 64 бита максимальная пропускная способность при тактовой частоте СМ 50 МГц составляет 400 Мбайт/с.

Микросхемы ПЗУ также построены по принципу матричной структуры. Функции ЭП в них выполняют перемычки в виде проводников, полупроводниковых диодов или транзисторов. В такой матрице наличие перемычки соответствует "1", а ее отсутствие -0. Запись информации в ПЗУ осуществляется посредством программатора.

Вопрос 1.7.2. Стековая память. Кэш-память первого, второго уровней

Сверхоперативные ЗУ используется для хранения небольших обьемов информации и имеют значительно меньшее время (в 2-10 раз) считывания/записи, чем ОЗУ. СОЗУ обычно строятся на регистрах и регистровых структурах.

Если к любому регистру можно обратиться для записи/чтения по его адресу, то такая регистровая структура образует СОЗУ с произвольным доступом.

Безадресные регистровые структуры могут образовывать два вида устройств памяти: магазинного типа и память с выборкой по содержанию (ассоциативные ЗУ).

Если запись в регистровую структуру производится через один регистр, а считывание - через другой, то такая память является аналогом задержки и работает по приципу "первым вошел - последним вышел" (FIFO-First Input, First Output).

Если же запись и чтение производится через один и тотже регистр, то такое усройство называется стековой памятью, работающее по принципу "первым вошел-последним вышел" (FILO-First Input, Last Output).

Стековая память получила широкое распространение. Для ее реализации в ОЗУ посредством программ операционной системы выделяется часть памяти под стек.Специальный регистр МП (указатель стека) постоянно хранит адрес ячейки ОЗУ, выполняющий функции вершины стека. Чтение числа всегда производится из вершины стека, после чего указатель стека изменяется и указывает на очередную ячейку стековой памяти.

В микропроцессорах ассоциативные ЗУ (память с выборкой по содержанию) используются в составе кэш-памяти для хранения адресной части команд и операндов исполняемой программы. При этом нет необходимости обращаться к ОЗУ за следующей командой или требуемым операндом: достаточно поместить в маску необходимый адрес, если искомая информация имеется в СОЗУ, то она сразу будет выдана. Обращение к ОП будет необходимо лишь при отсутствии требуемой информации в СОЗУ. За счет такого использования СОЗУ сокращается число обращений к ОЗУ, а это позволяет экономить время, так как обращение к СОЗУ требует в 2-10 раз меньше времени, чем обращение к ОЗУ.

Кэш-память может быть размещена в кристалле процессора (так называемая "кэш-память I уровня") или выполнена в виде отдельной микросхемы (внешняя кэш-память или кэш-память II уровня). Встроенная кэш-память (I уровня) в процессорах Pentium имеет объём около 16 Кбайт, время доступа - 5 - 10 нс, работает с 32-битными словами и при частотах 75 - 166 МГц обеспечивает пропускную способность от 300 до 667 Мбайт/с. внешняя кэш-память (II уровня) имеет объём 256 Кбайт - 1 Мбайт, время доступа 15 нс, работает с 64-битными словами при частоте 66 МГц обеспечивает максимальную пропускную способность 528 Мбайт/с

Вопрос 1.7.3. Эволюция микропроцессоров(МП). Основные технические характеристики МП. Классификация МП

Микропроцессором (МП) называют полупроводниковый кристалл или комплект кристаллов, на которых реализуется центральный процессор ЭВМ,

т.е. совокупность арифметико-логического устройства и центрального устройства управления. В 1969 году компании Intel удалось создать комплект из четырех интегральных микросхем (ИМС) с полным набором элементов характерным для процессора. Длина слова первого МП составляла всего 4 бита. В 1971 г. был выпущен 4-разрядный коммерческий МП Intel 4004, который стал применяться в микрокалькуляторах. Уже в 1972г. появился 8-разрядный МП Intel 8008, а в 1947г. - его улучшенный вариант Intel 8080. Он нашел применение в первых встроенных ЭВМ для управления производственными процессами.

Развитие МП шло по разным направлениям, важнейшим из которых -увеличение разрядности. Подавляющее большинство МП производимых в настоящее являются однокристальными 32-разрядными, за исключением ряда 64-разрядных комплектов. МП характеризуется :

- тактовой частотой;
- разрядностью;
- архитектурой.

Тактовая частота МП (более строго -тактовая частота, при которой способен работать МП) определяется максимальным временем выполнения элементарного действия в МП. Работа МП синхронизируется импульсами тактовой частоты от задающего генератора. Чем выше тактовая частота МП (при прочих равных условиях), тем выше его быстродействие.

Разрядностью МП называют максимальное количество разрядов двоичного кода, которые могут обрабатываться или передаваться одновременно. Понятие разрядность включает:

- разрядность внутренних регистров МП;
- разрядность шины данных;
- разрядность шины адреса.

От разрядности шины данных зависит скорость передачи информации между МП и другими устройствами. Разрядность щины адреса определяет адресное пространство МП.

Архитектура МП является емким понятием, имеющим при этом неоднозначное толкование. Архитектурой часто называют организацию МП с точки зрения пользователя.

Можно ограничить рассмотрение архитектуры МП следующими элементами:

- системой команд и способами адресации;
- возможностью совмещения выполнения команд во времени;
- наличие дополнительных устройств и узлов в составе МП;

- режимами работы МП.

Система команд представляет собой совокупность команд, которую способен выполнить МП. Она включает полный спмсок кодов операций, для каждой из которых указывается число операндов и допустимые способы их адресации. **Способы адресации** определяют технику вычисления адресов ячеек памяти и выполнения операций над адресными регистрами.

В соответствии с составом системы команд различают:

1. МП с CISC-архитектурой (CISC-complex instruction set computer-компьютер со сложной системой команд).
2. МП с RISC-архитектурой (RISC-reduced instruction set computer-компьютер с сокращенной системой команд).

МП с CISC-архитектурой являются традиционными, а их система команд включает большое количество команд для выполнения арифметических и логических операций, rкоманд управления, пересылки и ввода-вывода данных.

Система команд МП с RISC-архитектурой упрощена и сокращена до такой степени, что каждая команда выполняется за единственный такт. Такой подход упростить структуру МП и тем самым повысить его быстродействие.

Некоторые наиболее развитые МП обеспечивают **совмещение нескольких последовательно расположенных команд во времени,** организуя **конвейрную обработку**. Эта архитектурная особенность оказывает заметное влияние на скорость выполнения линейных участков программ.

МП поддерживает широкий спектр **режимов работы,** среди которых:

1. Однопрограммный режим.
2. Многопрограммный режим.
3. Система виртуальных машин.

В **однопрограммном режиме** в каждый момент времени может находиться в ОЗУ и выполняться только одна пользовательская программа.

В **многопрограммный режиме** обеспечивается хранение в ОЗУ несколько программ и попеременное их выполнение с той или иной дисциплиной обслуживания, что целесообразно главным образом при возможности совмещения во времени счета в МП и операций ввода-вывода. На основе многопрограммного режима работы МП могут быть организованы **однопользовательский многопрограммный,** а при наличии соответствующих аппаратных средств - и **многопользовательский многопрограммный режим** работы ПК. **Система виртуальных машин**

является дальнейшим развитием мультипрограммирования, основной признак которого - возможность одновременной работы нескольких ОС.

Вопрос 1.7.4. Структурная схема МП

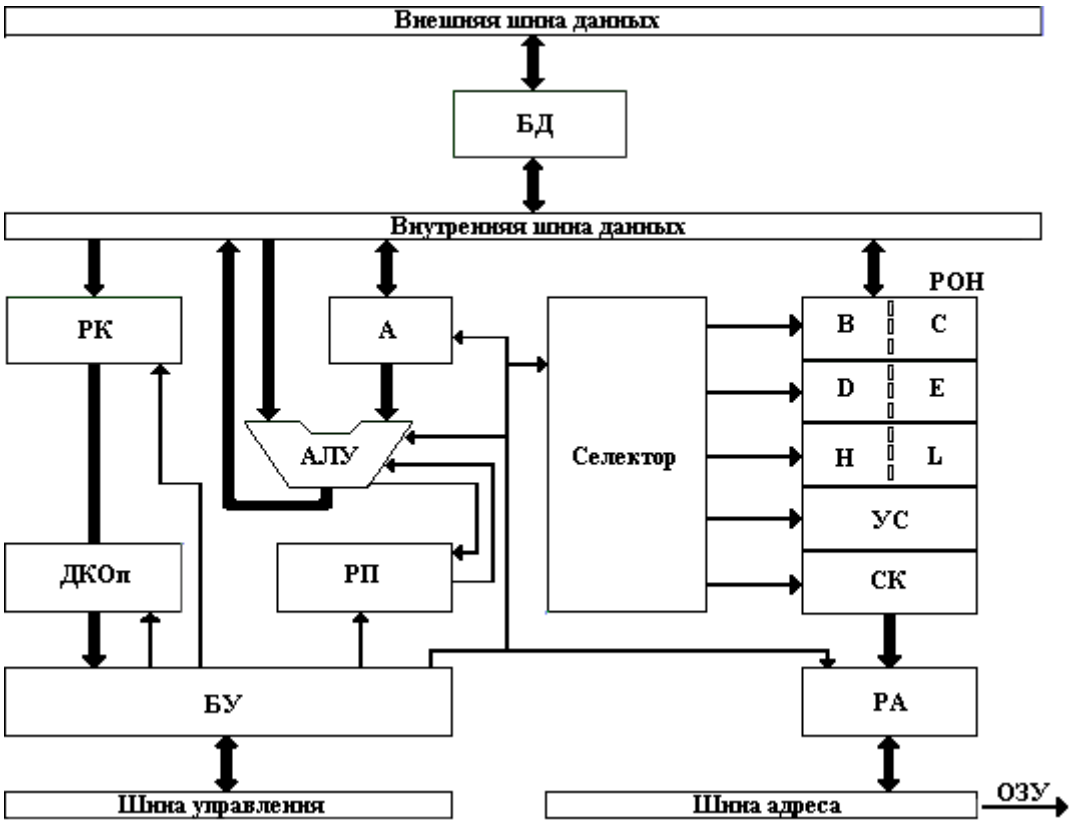
Общая структура гипотетического иллюстративного микропроцессора редставлена на рис.

МП выполнен на одном кристалле кремния в виде одной интегральной схемы, в которой реализоаны все основные функции процессора.

Входящие в его состав компоненты говорят о том, что он является процессором с программным управлением. Некоторые компоненты, программный счетчик (счетчик команд СК), стек (на схеме не указан), регистр команд (РК), служат для обработки команд. Такие компоненты арифметико-логическое устройство (АЛУ), триггер переноса (на схеме не указан), регистры общего назначения (РОН) и регистр адреса данных (РА), служат для обработки данных. Все остальные компоненты, а именно дешифратор команд (дешифратор кода операции ДКОп) и блок управления и синхронизации (БУ), управляют работой других компонентов. Взаимодействие компонентов осуществляется по внутренней шине данных. Связь МП с другими блоками (ЗУ и устройствами ввода/вывода) происходит по адресной щине, шине данных (внешней) и управляющей шине.

МП работает со словами, состоящими из 8 битов (1 байт). Адрес из 8 бит позволяет адресовать $2^8=256$ ячеек памяти. Для реальных задач этого, конечно, мало, поэтому для задания адреса памяти обычно используется 16 разрядов (два байта), и это позволяет прямо адресовать $2^{16}=65536$ ячеек. Адресная шина однонаправленная, 16-ти разрядная.

Рисунок 1.7.2. Структурная схема гипотетического иллюстративного микропроцессора.



Информация к МП и от него передается по шине данных. Шина данных двунаправленная, 8-ми разрядная. Управляющая шина состоит из 5 линий, ведущих к блоку управления и синхронизации, и 8 выходящих из него линий. По этим линиям передаются управляющие и тактирующие сигналы между компонентами МП и МП и другими блоками ПК.

Счетчик команд (СК) состоит из 16 битов и содержит адрес очередного байта команды, считываемого из памяти. Содержимое СК автоматически увеличивается на единицу после чтения каждого байта команды. Существует связь между СК и вершиной стека. Одна из функций стека-сохранение адреса возврата из подпрограммы. В стеке могут сохраняться данные из верхних трех регистров РОН и триггера переноса.

В то время как слово данных состоит из одного байта, команда может состоять из одного, двух и трех байтов. Первый байт любой команды поступает из ОЗУ по шине данных на регистр команд (РК). Этот первый байт подается на вход дешифратора команд (ДКОп-дешифратор кода операции), который определяет вид операции. В частности, дешифратор определяет является ли команда однобайтовой, или она состоит из большего числа байтов. В последнем случае дополнительные байты передаютяс по шинам данных из ОЗУ и принимаются или на регистр адреса данных, или на один из регистров РОН

Регистр адреса данных содержит адрес операнда для команд, обращающихся к памяти, адрес порта для команд ввода/вывода.или адрес

следующей команды для команд перехода. Регистры РОН (регистры общего назначения) содержат операнды для всех команд, работающие с данными. Для указания этих регистров используются 4-битовые коды от 0000 до 1110. Регистр 0000 называетяс аккумулятором (АК) и участвует во всех арифметических и логических операциях. В частности, он содержит один из операндов перед выполнением операции и получает результат после ее завершения. Обычно обращение к РОН осуществляется при помощи R-селектора.

Все арифметические и логические операции выполняются в АЛУ. Результаты из АЛУ передаютяс в аккумулятор по выходной 8-ми битовой шине.

Последний компонент МП - это **блок управления и синхронизации.** Он получает сигналы от дешифратора команд, который анализирует команду. В БУ из АЛУ и от триггера переноса поступают сигналы, по которым определяются условия для передачи управления. Все остальные компоненты МП получают от БУ управляющие и синхронизирующие сигналы, необходимые для выполнения команды. С помощью 13 внешних линий реализуется интерфейс устройства управления с другими модулями ПК.

Рисунок 1.7.3.

- **Указатель стека (УС)** представляет собой 16-разрядный регистр, служащий для формирования адреса стековой памяти.
- **Регистр признаков (РП)** используется для фиксации определенных ситуаций, возникающих в результате выполнения операции. Код ситуации называется признаком результата. Признак может принимать значение 0 или 1. Отметим некоторые из возможных признаков.

Признак переноса указывает на необходимость переноса единицы при сложении двух восьмиразрядных чисел в младший значащий разряд следующего байта. Признак нуля фиксирует появление нулевого результата операции, признак знака-знак результата операции. Эти и другие признаки используются для изменения хода обработки информации.

Буфер данных (БД) служит для организации обмена информацией между внутренней и внешней шинами данных.

Вопрос 1.7.5. Полный цикл работы МП при выполнении команды

Выборка каждого байта команды производится в два этапа.

1. На первом этапе адрес байта команды, содержащийся в СК, передается в РА, а затем по адресным шинам в ОЗУ, где происходит выбор адресуемого байта.
2. На втором этапе - передача содержимого адресуемого байта по шинам данных в МП и увеличение содержимого СК на 1. При этом, если выбирается не последний байт команды то новое новое содержимое СК будет представлять адрес следующего байта текущей команды, а если выбирается последний байт, то адрес первого байта следующей команды.
3. При выборке первого байта команды его содержимое передается и фиксируется в РК.
4. Дешифратор кода операции по содержимому РК, поступающему на его вход, определяет вид операции, число байтов в команде, а также способ адресации, используемый в команде.
5. В соответствии с видом операции БУ задает необходимую последовательность управляющих воздействий, приводящих к выборке следующих байтов команды или к использованию команды.

Пример 1. Описать процесс выборки и исполнения команды, по которой содержимое регистра РОН суммируется с содержимым аккумулятора А. Результат операции фиксируется в аккумуляторе. Команда однобайтовая с прямой регистровой адресацией. В коде команды $10000001_2=81_{16}$первые пять старших разрядов определяют код операции; три последних-код адреса регистра С. Команда записана в ячейку ОЗУ с адресом (номером) 0176_{16}.Число $2F_{16}$ записано в регистр С РОН. В аккумулятор А записано число $3A_{16}$.

Процесс выборки и исполнения команды можно условно разбить на несколько этапов:

1. Адрес команды 017616 записывается в РА.
2. Адрес команды из РА по шине адреса поступает ОЗУ.
3. ОЗУ находит ячейку с указанным адресом и считывает команду в буферный
4. регистр ОЗУ.
5. Команда из буферного регистра ОЗУ по внешней шине данных, буфер данных (БД), внутренней шине данных поступает в РК.
6. ДКОп распознает, что выборка команды закончна и над содержимым аккумулятора и регистра С РОН необходимо произвести суммирование.
7. Содержимое РА увеличивается на единицу и формируется адрес следующей ячейки ОЗУ 0177_{16}.

8. БУ вырабатывает последовательность сигналов, под воздействием которых числа $3A_{16}$из аккумулятора и $2F_{16}$из из регистра С РОН поступают в АЛУ. АЛУ производит операцию суммирования $3A_{16}+2F_{16}=69_{16}=01101001_2$.
9. Результат операции записывается в аккумулятор, замещая в нем прежнее число $3A_{16}$.
10. Выполнение команды закончено. Начинается выборка следующей команды с адресом 0177_{16}.

Вопрос 1.7.6. Система команд МП

Микропроцессор имеет базовую систему команд, в состав которой входят следующие группы:

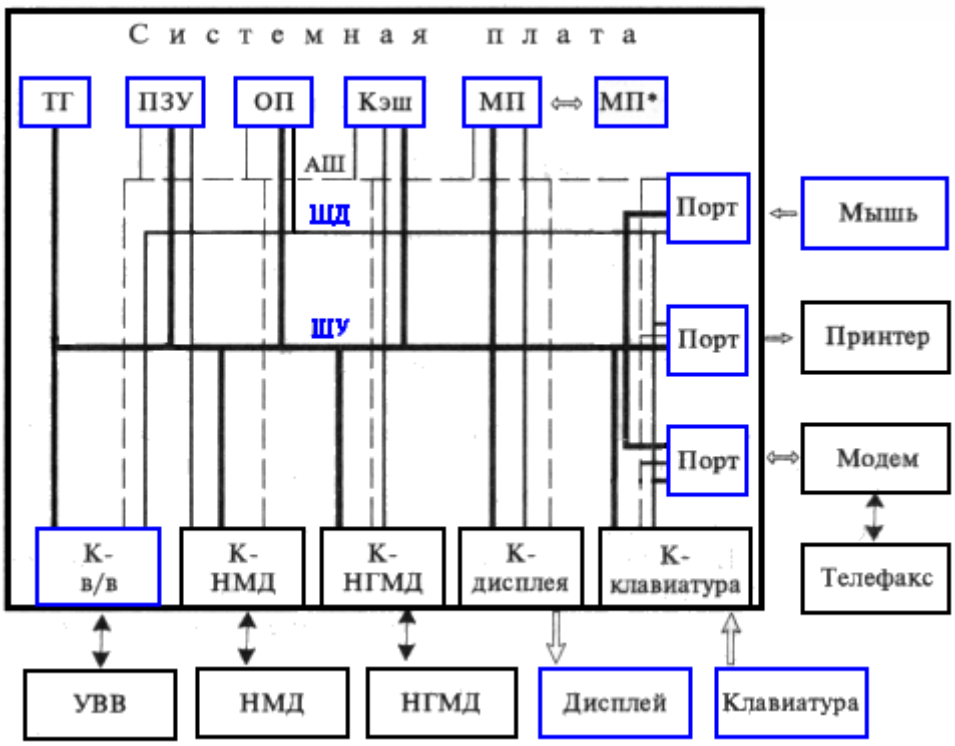
- команды пересылки данных:
- команды пересылки данных внутри МП (MOV, PUSH, POP, XCHNG и т.д.);
- команда ввода-вывода (IN, OUT);
- операции с флагами;
- операция с адресами (LEA, LDS и т.д.);
- арифметические команды:
- основные (сложение, вычитание, умножение, деление);
- дополнительные (INS, DEC и др.);
- логические команды (сдвиг, дизъюнкция, конъюнкция, отрицание равнозначности и др.);
- команды обработки строковых данных (пересылка, сравнение, сканирование, слияние/разделение и др.);
- команды передачи управления (безусловный переход, условный переход, прерывания, переход с возвратом);
- команды управления ("нет операций", "внешняя синхронизация" и т.д.).
- каждая команда имеет большое число модификаций, чаще всего определяемых режимом адресации данных (операндов).

Команды обращения к памяти

Команда: LOAD REGISTR (загрузка регистра)

Символическая форма: LDR r

Рисунок 1.7.4.



Описание: (M[< B2 > < B3 >])r

Содержимое ячейки памяти передаётся в общий регистр r.

Старшие 8 разрядов адреса ячейки берутся из второго байта команды, а младше 8 разрядов - из третьёго байта.

Программирование на машинном язык. Программа должна быть представлена в той форме, в которой её воспринимает машина. В частности, команды должны быть представлены в виде последовательностей из нулей и единиц, поскольку это единственная понятная ЭВМ форма. Однако выписывать длинные последовательности из нулей и единиц довольно утомительно. Поэтому при записи команд применяется более удобная шестнадцатеричная система.

Например, трёх битовая команда иллюстрированного процессора, загружается в аккумулятор содержимое ячейки памяти с адресом 0110 1111 1101 1011_2, имеет вид

01110000

01101111

11011011

в шестнадцатеричных обозначениях эта команда запишется так

70

6F

DB

Двоичное и шестнадцатеричное представление машинного языка неудобно и ненаглядно для человека. Поэтому, как правило, команды представляются в символической форме - используются легко запоминаемые мнемонические обозначения команд. Например предыдущую программу можно записать в виде

LDR 0

6F

DB

Второй и третий байты по-прежнему записаны шестнадцатеричными цифрами, и аккумулятор задан своим номером среди общих регистров, равным 0. Однако операция обозначена трёхбуквенным мнемоническим именем LDR.

Примерпрограммы, в которой к числу, находящемуся в ячейки $000A_{16}$ главной памяти, прибавляется шестнадцатеричная константа 3C. программа приведена в табл. 1. Она выполняется следующим образом.

Таблица 1. Пример программы: "прибавить константу к числу в памяти и записать результат снова в память"

Рисунок 1.7.5.

Номер ячейки памяти (шестнадцатеричный)	Команда на машинном языке	Команда в символической форме	Комментарий
0000	70	LDR 0	Передача содержимого ячейки памяти 000A в аккумулятор.
0001	00	00	
0002	0A	0A	Загрузка в регистр 1 величины 3C.
0003	61	LRI 1	
0004	3C	3C	Сложение регистров 0 и 1, сумма в аккумуляторе.
0005	81	ADD 1	
0006	74	STR 0	Передача содержимого аккумулятора в ячейку 000B
0007	00	00	
0008	0B	0B	Останов программы.
0009	FA	HLT	Исходное число.
000A			Сумма.
000B			

Прежде всего, подаётся прежний импульс на линию "сброс" БУ, по которому на программный счётчик устанавливается 0. Затем подаётся импульс на линию "пуск" БУ, и начинается выполнение программы. Выбираются 3 байта, составляющие первую команду, и программный счётчик увеличивается на 1 после выборки каждого байта. Первая команда пересылает содержимое ячейки памяти 000A в аккумулятор. Поскольку программный счётчик подвинулся на 3 единицы, он теперь содержит 0003. Это адрес следующей команды - загрузки регистра с непосредственным адресом. Команда содержит в себе операнд и удобна для задания константы. Её выполнение приводит к засылке значения 3C, содержащегося во втором байте команды, в общий регистр 1. Теперь программный счётчик содержит 0005, т.е. адрес третьей команды. Её выборка и выполнение приводит к прибавлению содержимого регистра 1 к содержимому аккумулятора и занесению суммы в аккумулятор. Следующая команда передаёт содержимое аккумулятора в ячейку памяти 000B. последняя команда останавливает микропроцессор.

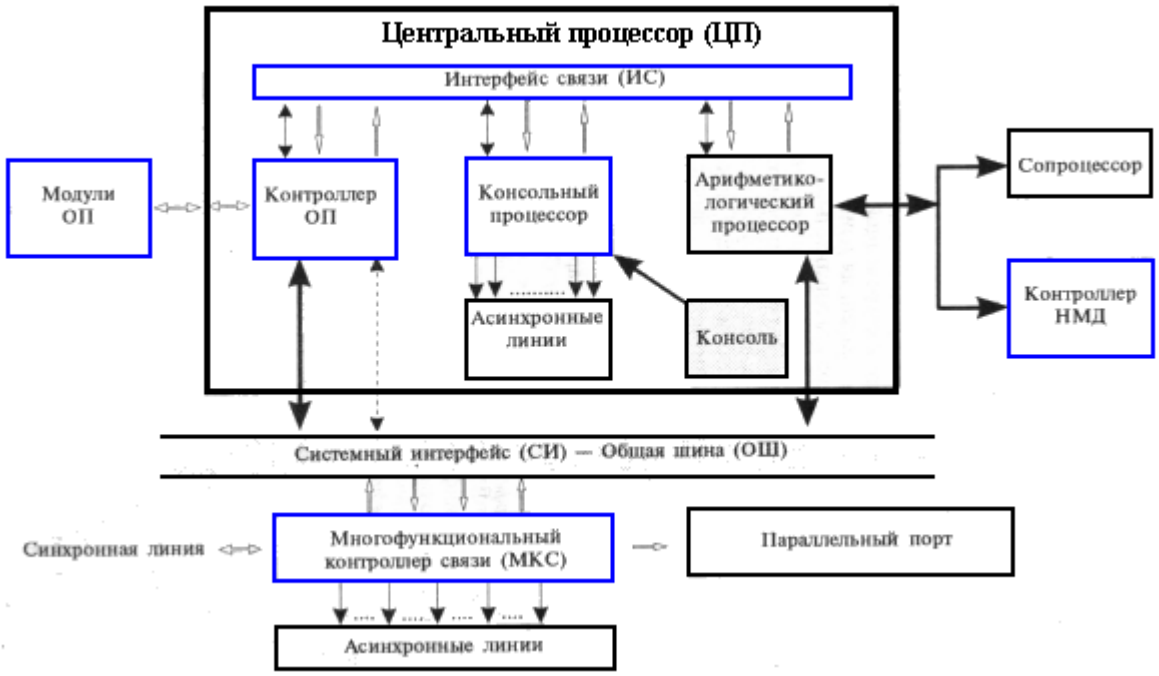
Вопрос 1.7.6. Структурная схема базовой модели микропроцессора фирмы Intel

Структурная схема базовой модели микропроцессора фирмы Intel приведена на рис. Условно МП можно разделить на две части : исполнительный блок (Execution Unit- EU) и устройство сопряжения с системной магистралью (Bus Interface Unit-BIU).

Арифметический блок включает арифметико-логическое устройство, вспомогательные регистры для хранения операндов и регистр флагов.

Восемь регистров исполнительного блока МП (AX,BX,CX,DX,SP,BP,SI,DI), имеющих длину, равную машинному слову, делятся на две группы. Первую группу составляют регистры общего назначения: AX,BX,CX и DX, каждый из которых представляет собой регистровую пару, составленную из двух регистров длинной в 1 байт: аккумулятор, или регистр AX состоит из регистров AH и Al. Регистр базы (Base Register) BX состоит из регистров BH и BL. Счетчик (Count RegiSter) CX включает

Рисунок 1.7.6.



регистры CH и CL. Регистры данных (Data Register) DX содержит регистры DH и DL. Каждый регистров может использоваться самостоятельно или в составе регистровой пары.

Вторую группу составляют адресные регистры SP,BP,SI и DI (в старших моделях количество адресных регистров увеличено).

Устройство сопряжения с ситемной магистралью содержит управляющие регистры, конвейер команд, АЛУ команд, устройство управление исполнительным блоком МП и интерфейс памяти (соединяющий внутреннюю магистраль МП с системной магистралью ПК).

Управляющие регистры BIU: CS (указатель командного сегмента), DS (указатель сегмента данных), SS (указатель сегмента стека), ES(указатель дополнительного сегмента) и др. служат для определения физических ОП-операндов и команд.

Регистр IP(Instruction Point) является указателем адреса команды, которая будет выбираться в конвейер команд в качестве очередной команды. Конвейер команд хранит несколько команд, что позволяет при выполнении линейных программ совместить подготовку очередной команды с выполнением текущей.

К управляющим регистрам МП относится и регистр флагов, разряд котрого имеет строго определенное назначение.

Контрольные вопросы.

1. Каково понятие ЗУ? Опишите структурную схему ОЗУ. Перечислите типы ОЗУ
2. по способу хранения информации.
3. Перечислите и поясните основные характеристики ОЗУ.
4. Опишите стековую память, кэш память.
5. Перечислите и поясните основные характеристики МП.
6. Перечислите и поясните основные элементы определяющие архитектуру МП.
7. Опишите структурную схему гипотетического МП.
8. Перечислите и опишите этапы работы МП при выполнении команды.
9. Перечислите основные группы команд МП. Опишите формат команды LOAD REGISTR.
10. Опишите пример программы в командах на машинном языке и символической форме.
11. Опишите структурную схему базовой модели МП фирмы Intel.

Тема 1.8. Периферийные устройства ЭВМ

Цель изучения темы:

изучить основные периферийные устройства.

Содержание.

1. Видеосистема.
2. Клавиатура.
3. Принтер.
4. Сканер.
5. Манипулятор типа мышь.
6. Накопитель на гибком магнитном диске.
7. Накопитель на жестком магнитном диске.
8. Накопитель на оптическом диске.

Видеосистема

Видеосистема предназначена для оперативного отображения информации. Обычно она состоит из монитора и адаптера. Монитор служит для визуализации изображения, адаптер - для связи монитора с микропроцессорным комплектом.

По принципу формирования изображения мониторы делятся на плазменные, электролюминесцентные, жидкокристаллические и электронно-лучевые.

Плазменные, электролюминесцентные и жидкокристаллические мониторы относятся к дисплеям с плоским экраном. Для них характерно: экран имеет малые физические размеры, не мерцает, полностью отсутствует рентгеновское излучение.

Электронно-лучевая трубка (ЭЛТ) представляет собой электровакуумный прибор в виде стеклянной колбы, дно которой является экраном. В колбе, из которой удалён воздух, расположены электроды: электронная пушка, анод, вертикально и горизонтально отклоняющие пластины и сетка.

По длительности хранения информации на экране мониторы делятся на регенерируемые и запоминающие.

В регенерируемых мониторах изображение после однократной прорисовки держится на экране не долго, доли секунды, постепенно угасая.

В запоминающих мониторах после однократной прорисовки изображение на экране держится в течении нескольких часов.

По способу управления яркости луча мониторы делятся на цифровые и аналоговые.

В цифровых мониторах для управления яркостью на сетку подаются дискретные сигналы, которые в зависимости от настройки могут полностью запирать трубку (0) или полностью отпирать её (1), снижать яркость до 1/2 (0) или обеспечивать полную яркость (1) и т.д.

В аналоговых мониторах на сетку подаётся непрерывный (аналоговый) сигнал, который может плавно изменять яркость от полного запирания до полного отпирания.

По цветности изображения мониторы делятся на монохромные и цветные.

По эргономическим характеристикам мониторы делятся на:

- обычные;
- с пониженным рентгеновским излучением;
- с антистатическим экраном;
- работающие в энергосберегающем режиме.

Два режима работы дисплея - это графический и символьный.

В символьном режиме на экран может выводиться ограниченный состав символов, имеющий чётко определённый графический образ: буквы, цифры, знаки пунктуации, математические знаки и знаки псевдографики. Состав этих символов определён системой кодирования, в этом ЭВМ.

В графическом режиме изображение на экране формируется из отдельных точек (пиксель), имеющих свои адреса.

Клавиатура

Клавиатура - это одно из основных устройств ввода информации в ЭВМ, позволяющее вводить различные виды информации. Вид вводимой информации определяется программой, интерпретирующей нажатые или отпущенные клавиши. С помощью клавиатуры можно вводить любые символы - от букв и цифр до иероглифов и музыкальной нотации. Клавиатура позволяет управлять курсором на экране дисплея.

Принтер

Принтер - это внешнее устройство ЭВМ, предназначенное для вывода информации на твёрдый носитель в символьном и графическом виде.

Классификация принтеров может быть проведена по следующим критериям: по способу вывода, по принципу формирования изображения, по способу регистрации и по принципу управления процессом печати.

По способу выбора изображение принтера делится на две группы: символьные и графические. Символьные принтеры могут выводить информацию в виде отдельных символов по мере их поступления в печатающее устройство (ПУ). Графические печатающие устройства выводят информацию не целыми символами, а отдельными точками или линиями.

По принципу формирования выводимого изображения ПУ делятся на три вида: литерные, матричные и координатные (векторные).

Литерные устройства выводят информацию в виде символов, каждый из которых является графическим примитивом данного устройства. Литеры сформированы при изготовлении принтера, нанесены на специальные рычаги или литерные колёса-шрифтоносители и при эксплуатации принтера без замены шрифтоносителя не меняются.

Матричные ПУ выводят информацию в виде символов, сформированных из отдельных точек, объединённых в символьную матрицу. Печатающая головка матричного принтера имеет вертикальный ряд иголок, каждая из которых может сделать оттиск самого аленького элемента изображения - пиксела (точки).

Координатные ПУ - плоттеры, графопостроители - выводят информацию как текстовую, так и графическую либо в виде отдельно адресуемых точек, либо сформированную из отдельных линий - так называемых "штриховое" изображение.

По способу регистрации изображения ПУ делятся на ударные и безударные.

ПУ ударного действия формируют изображение на бумаге, сжимая с помощью удара на коротки промежуток времени рельефное изображение символа или его части, красящей ленты и бумаги.

ПУ безударного действия характеризуется тем, что изображение на бумагу наносится через промежуточный носитель, чувствительный к электрическому воздействию, электрическому полю, магнитному полю, и др.

Сканер

Сканер - это внешнее устройство ПЭВМ, позволяющее вводить двухмерное (т.е. плоское) изображение.

Типы вводимого изображения: штриховое или полутоновое, монохромное или цветное.

Штриховое изображение (рисунок, текст) состоит из тёмных линий на светлом фоне. По яркости элементы рисунка могут быть либо тёмными, либо светлыми - промежуточных значений в штриховом рисунке нет.

Полутоновое изображение (рисунок, фотография) состоит из элементов, различающихся яркостью.

Принцип работы сканера заключается в том, что поверхность изображения освещается перемещающимся лучом света, а светочувствительный прибор (фотоэлемент) воспринимает отражённый свет, интенсивность которого зависит от яркости освещённого участка изображения, и преобразовывает его в электрический сигнал. Полученный электрический сигнал преобразуется из аналоговой в цифровую форму и в виде цифровой характеристики яркости точки поступает в ЭВМ.

Такой сканер считывает изображение в графическом виде; полученное изображение может быть сохранено в памяти ЭВМ, обработано графическим редактором или выведено на дисплей или принтер. Если был введён текст, то при отображении на дисплее или принтере его можно прочитать.

Конструктивно сканеры выпускаются в двух вариантах: портативные и настольные.

Портативные сканеры представляют собой устройство, внешне похожую на мышь, которое перемещается по вводимому в ЭВМ изображению.

Настольные сканеры выпускаются трёх типов:

Sheet-fed - строчный сканер;

Flat-bed - страничный сканер;

Over-head - сканер-планшет проекторного типа.

Анимационные устройства ввода-вывода

В состав анимационных устройства ввода-вывода входят видеокамера, видеомагнитофон и телевизор, а также преобразователи видео сигналов.

Манипулятор типа "мышь"

Манипуляторы (координатно-указательные устройства, устройство управления курсором) являются дополнительными периферийными

устройствами, позволяющими быстро перемещать курсор по экрану дисплея, выделять пункты меню, а также выделять фрагменты экрана.

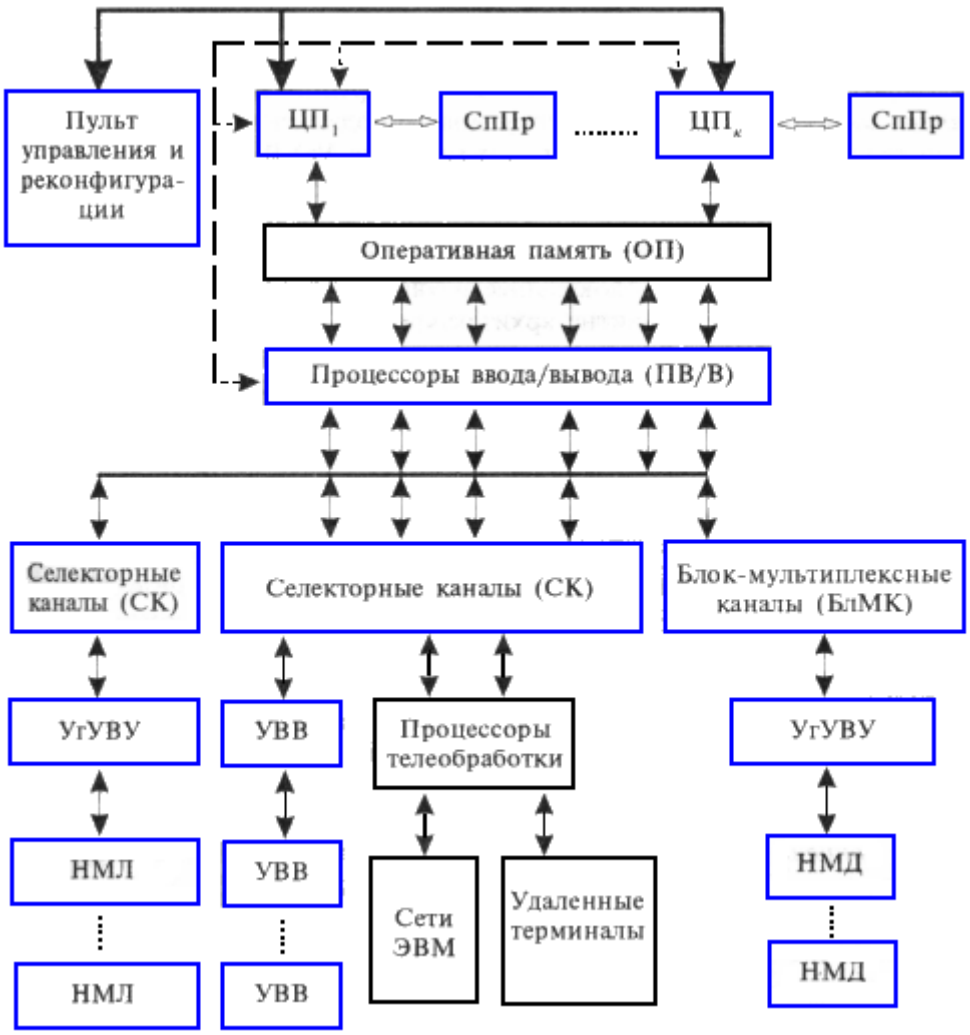
Манипулятор типа "мышь" (mouse) представляет собой приспособление для указания нужных точек на экране путем перемещения его вручную по плоской поверхности. Координаты местоположения "мыши" передаются В ПК и вызывают соответствующее перемещения курсора по экрану дисплея.

Основными компонентами устройства являются:

- корпус;
- шарик - датчик перемещения мыши;
- несколько кнопок для подачи команд;
- кабель для соединения мыши с компьютером;
- разъём для подключения к компьютеру.

Оптико-механические датчики манипулятора типа "мышь" представлен на рис.

Рисунок 1.8.1. Оптико-механические датчики.

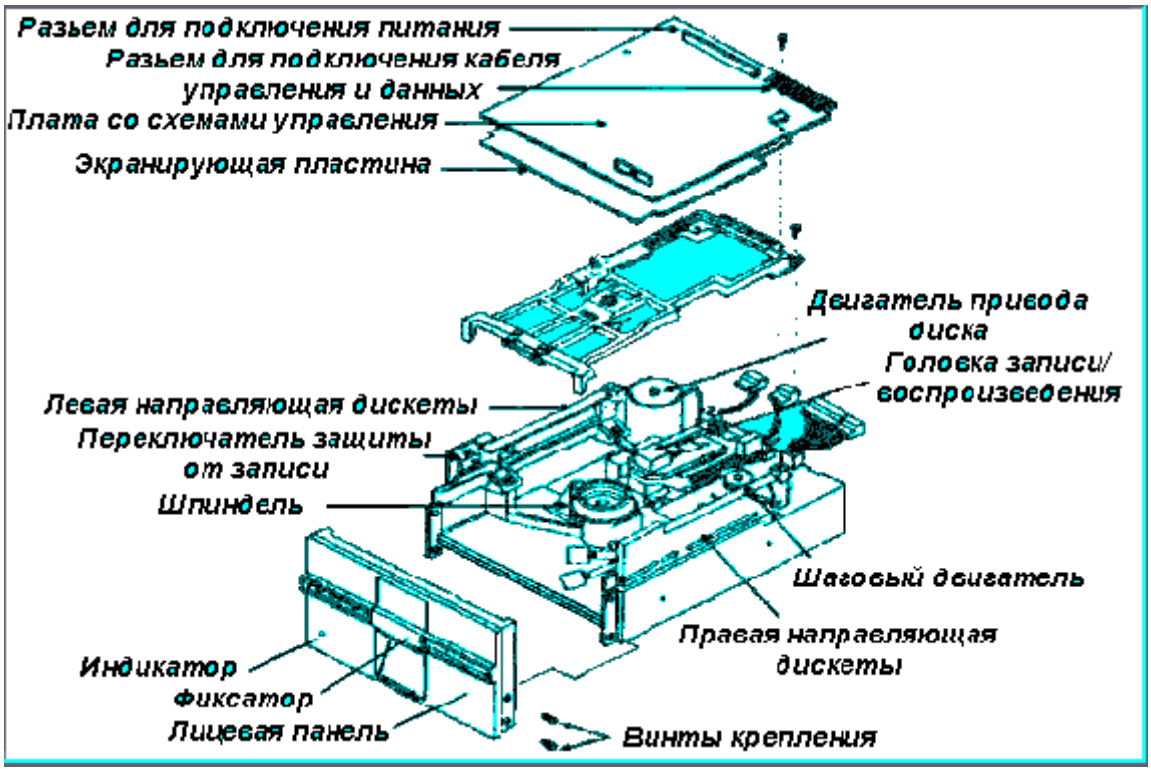


Накопитель на гибком магнитном диске

Накопитель на гибком магнитном диске НГМД является устройством со сменными носителями информации. В качестве носителя информации используется гибкий магнитный диск (ГД). В ПК ГД служат для долговременного хранения программ, которые при необходимости загружаются в ОЗУ для выполнения, и данных, используемых или формируемых программами. Сменные носители информации необходимы для решения таких задач, как:

- резервирование (дублирование) информации;
- обеспечение конфиденциальности данных;
- транспортирование информации;
- распространение ПО.

Рисунок 1.8.2. Накопитель на гибких дисках.

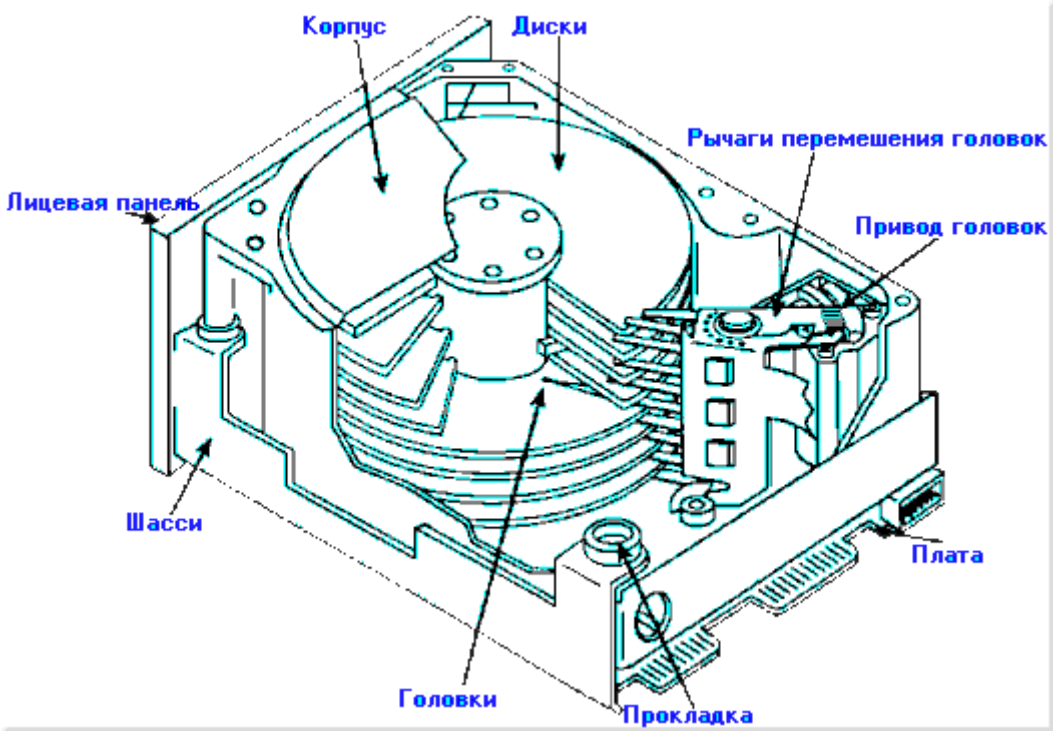


Накопитель на жестком магнитном диске

В ПК используются главным образом НЖМД с несъемными магнитными дисками. Поэтому накопители данного типа служат исключительно для хранения подлежащих выполнению программ и информационных массивов без возможности переноса на другие машины. Однако НЖМД существенно превосходят НГМД как по емкости, так и по быстродействию.

Основными элементами конструкции типичного накопителя на жёстких магнитных дисках являются (рис.):

Рисунок 1.8.3.



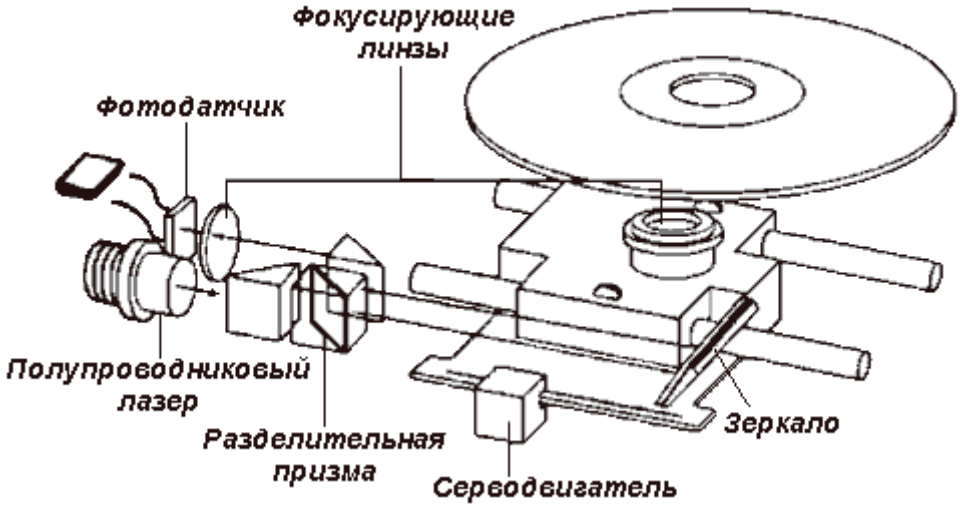
- диски,
- головки записи/воспроизведения,
- механизм привода головок,
- двигатель приводов дисков,
- печатная плата со схемами управления,
- кабели и разъемы,
- элементы конфигурирования (перемычки и переключатели),
- Лицевая панель (необязательный элемент).

Накопитель на оптическом диске

CD-ROM - это оптический носитель информации, на котором может храниться до 650 М данных, что соответствует примерно 333 000 страниц текста или 74 минутам высококачественного звучания, причём символьные и звуковые фрагменты могут быть объединены на одном диске.

Упрощённый вид устройства накопителя на CD-ROM показано на рис.

Рисунок 1.8.4. Устройство накопителей на CD-ROM.



Контрольные вопросы.

1. Каково назначение и состав видиосистемы? Перечислите и поясните основные классификационные признаки монитора.
2. Каково назначение принтера? Перечислите и поясните основные классификационные признаки принтера.
3. Каково назначение сканера? Опишите принцип работы сканера.
4. Каково назначение и устройство манипулятора "мышь"?
5. Каково назначение и устройство НГМД?
6. Каково назначение и устройство НЖМД?
7. Каково назначение и устройство CD-ROM?

Тема 1.9. Архитектура структуры ЭВМ

Цель изучения темы:

- Изучить общую архитектуру и состав сЭВМ, мини-ЭВМ, ЭВМ общего назначения,.
- Изучить структуру и состав системной платы персонального компьютера.

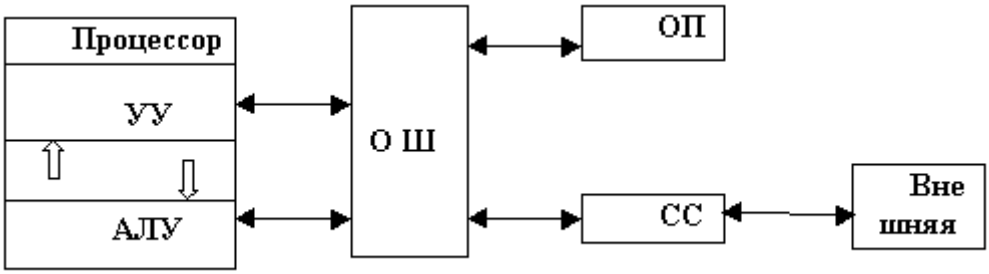
Вопросы.

1. Общие сведения о специальных ЭВМ (сЭВМ). Общая архитектура и состав сЭВМ.
2. Структурная схема системной платы ПК.
3. Общие сведения о мини-ЭВМ. Общая архитектура и состав мини-ЭВМ.
4. Общие сведения о ЭВМ общего назначения. Общая архитектура и состав ЭВМ общего назначения.

Вопрос 1.9.1. Общие сведения о специальных ЭВМ (сЭВМ). Общая архитектура и состав сЭВМ.

Специальные ЭВМ (сЭВМ) ориентированы на решение специальных вычислительных задач либо задач управления, решаемых в режиме реального времени Последние используются в различных системах управления и часто образуют специальные ГВМ, обрабатывающие аналого-цифровую информацию. Общая архитектура сЭВМ приведена на рис. 1.9.1.

Рисунок 1.9.1. Общая архитектура специальной ЭВМ (сЭВМ)



Основные компоненты сЭВМ:

- процессор, содержащий устройство управления (УУ) и арифметико-логической устройство (АЛУ), сложность и функциональные возможности которых определяются назначением данного типа машины;
- оперативной памяти (ОП);

- система сопряжения (СС), предназначенная для обеспечения интерфейса между памятью процессора и внешней средой, в качестве которой выступают различные источники и приемники информации (датчики, управляющие блоки, оконечные устройства и т.д.). Так как внешняя среда оперирует как с дискретной, так и с аналоговой информацией, то и СС в общем случае обеспечивает интерфейс между обеими формами информации различного типа (данные, логическая, управляющая и др.).
- Основные компоненты сЭВМ объединяются общей шиной (ОШ), или общей магистралью.

Процессор функционирует под управлением программы, находящейся в ОП или в постоянном запоминающем устройстве (ПЗУ), и его архитектура определяется, во многом, классом машины и спецификой решаемых задач. Если сЭВМ ориентирована на решение узкого класса задач по единой программе, но с различными данными, то программу можно поместить в ПЗУ, хранящее информацию даже при отключенном питании; ее в ряде случаев можно реализовать и аппаратно, т.е. "зашить" в электронные схемы машины. В последнем случае повышаются надежность и быстродействие сЭВМ. Такой подход широко используется, например, во многих видах бортовой ВТ, работающей в жестком режиме реального времени. В зависимости от класса сЭВМ обеспечиваются языками программирования различного уровня (от микропрограммного до высокого).

Повышенные требования к производительности сЭВМ (в ряде случаев не менее 100 млн оп/с) и спецификация решаемых задач стимулировали использование неклассической не - неймановской архитектуры организации вычислительного процесса. Отдельные классы задач допускают высокую степень распараллеливания (обработка сигналов, распознавание образов, сортировка, векторно-матричные вычисления, моделирование и др.). Данный подход привел к созданию сЭВМ как автономного, так и сателлитного принципа использования. В первом случае сЭВМ нетрадиционной архитектуры используются самостоятельно (автономно) для решения предназначенных для нее задач. Во втором - сЭВМ используются в качестве сопроцессора мини-ЭВМ, ПК или ЭВМ другого класса.

Основными направлениями развития класса сЭВМ являются работы по созданию оптических спецпроцессоров для обработки изобразительной информации и сигналов, Перспективным направлением разработки специальной ВТ нетрадиционной архитектуры можно считать мультипроцессорные, распределенные и иерархические системы и сети, образующие уже комплексы и системы сЭВМ.

Вопрос 1.9.2. Микропроцессорные ЭВМ и ПК.

Создание в начале 70-х годов первых универсальных МП можно считать началом эры микро-ЭВМ, а затем и персональных компьютеров (ПК). Первый из этой серии МП Intel-4004 выполнял все функции центрального процессора ЭВМ общего назначения и в совокупности с четырьмя микросхемами (памяти, УУ и интерфейса ввода/вывода) представлял собой компьютер, не уступающий по мощности большим ЭВМ середины 50-х г. 20 века.

Все ЭВМ можно условно разделить на четыре типа:

1. микро (micro)-;
2. мини(mini)-;
3. общего назначения (mainframe)-;
4. супер (super)-ЭВМ.

В настоящее время под микро-ЭВМ понимается микропроцессорная ВТ, используемая двояко:

1. в качестве универсального блока обработки данных и/или управления, выпускаемого в виде одной БИС/СБИС и предназначенного для встраивания в различные специализированные системы контроля и управления, и в другую технику для расширения ее функциональных и интеллектуальных возможностей, производительности, надежности, а также улучшения существующих и придания ей новых свойств;
2. в качестве надежной малогабаритной ЭВМ персонального пользования (ПК), предназначен для работы в интерактивном режиме, имеющей развитое ПО различного назначения и ориентированной на самый широкий круг пользователей различных возрастов и профессий, доступной по ценам массовому пользователю;

Практически все современные универсальные (относительно команд) микро-ЭВМ отражают классическую неймановскую архитектуру. Структурная сема ПК представлена на рис.

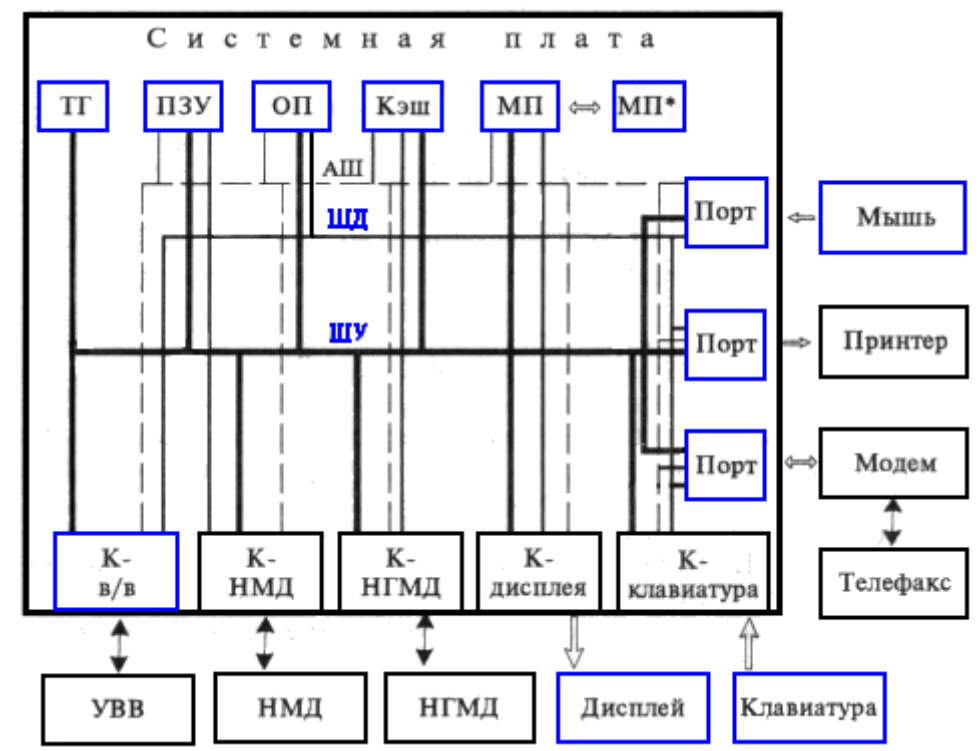
Данная схема поможет понять внутреннюю организацию некоторого типового ПК; вместе с тем, ее элементы характерны для любой микро- и мини-ЭВМ, а также в значительной степени и для ЭВМ общего назначения. Так, клавиатура и дисплей составляют консоль ЭВМ и являются наиболее типичными устройствами ввода и отображения информации, обеспечивая интерфейс пользователя с ЭВМ. Большинство современных ЭВМ в своем составе имеют в качестве внешней памяти (ВП) накопители на различного типа магнитных носителях (НМД, НГМД, НМЛ и др.); Для вывода и документирования данных и программ используются различного типа печатающие устройства (принтеры) и рисующие плоттеры, а для работы в системах телекоммуникации ЭВМ на основе модемов имеют возможность

обмена информацией с удаленными телефонными абонентами (например, с телефаксами или информационно- вычислительных сетях).

Системная плата ПК (рис.) содержит следующие основные компоненты:

- тактовый генератор (ТГ),
- постоянное запоминающее устройство (ПЗУ),
- оперативную память (ОЗУ),
- микропроцессор (МП) {и возможно, сопроцессор (МП)},
- контроллеры передачи данных,
- контроллеры ввода/вывода и порты ввода/вывода, а также шины управления, адресации и данных, образующие в совокупности общую шину

Рисунок 1.9.2. Структурная схема системной платы ПК.



Главной частью системной платы является МП, управляющий работой всей системы узлов ПК и программой, описывающей алгоритм решаемой задачи. МП имеет сложную структуру, реализованную в виде системы электронных логических схем; в качестве основных его компонент можно выделить:

- арифметико-логическое устройство (АЛУ);
- устройство управления (УУ);
- систему прерываний (СПр);
- устройство управления общей шиной (УОШ) - системным интерфейсом;
- специальные регистры.

Для расширения возможностей и повышения функциональных характеристик МП дополнительно может поставляться специальный сопроцессор (МП), как правило, служащий для расширения набора команд ведущего МП. В последнее время в качестве сопроцессоров при создании мультипроцессорных систем с однотипными МП и аппаратной поддержкой вычислительных процессоров все чаще используются транспьютеры, представляющие собой МП специального типа. Особенностью транспьютеров является наличие быстрых коммуникационных каналов связи, каждый из которых может одновременно передавать по одной магистрали данные в МП, а по другой - из него. В составе команд транспьютеров имеются команды управления процессами, поддержки предложений языков программирования высокого уровня (Fortran, Pascal, C). Высокая производительность транспьютеров обусловлена высокими скоростями передачи операндов команд в АЛУ и их обработки в нем. Типичными примерами транспьютеров являются известные модели T414 и T800 фирмы INMOS. В настоящее время наиболее распространенными и используемыми для создания ПК различных моделей являются 32-битные МП фирм Intel, Motorola, DEC, AND (США), NEC (Япония) и INMOS (Англия).

Кэш-память (Кэш) с малым временем доступа служит для временного хранения промежуточных результатов и содержимого наиболее часто используемых ячеек ОП и регистров МП; объем Кэш-памяти зависит от модели ПК и для большинства моделей составляет 256 Кбайт.

Контроллер ввода/вывода является необязательным и обычно применяется в многопользовательских системах; он берет на себя управление некоторыми операциями по вводу/выводу, при его отсутствии выполняемыми самим МП. Контроллеры внешних устройств служат для обеспечения прямой связи вторых с ОП, минуя МП (режим прямого доступа к ОП); как правило, они используются для устройств быстрого обмена данными с ОП (НГМД, НМД, дисплей и др.), а также для обеспечения работы в групповом и сетевом режимах.

Порты ввода/вывода служат для обеспечения обмена информацией ПК с внешними, не очень быстрыми устройствами (клавиатура, мышь, джойстик, телефонная сеть и др.). Порты бывают входными, выходными и универсальными (ввод/вывод), а также последовательными и параллельными. Последовательный порт ведет побитный, а параллельный - побайтный обмен информацией; поэтому принтер подключается к параллельному порту, а телефонная линия связи (через модем) - к последовательному. Большинство современных ПК имеет один параллельный и два последовательных порта ввода/вывода. Информация, поступающая через порт, направляется сначала в МП, а затем в ОП и наоборот. Так как клавиатура, мышь и дисплей имеют свои выделенные участки в ОП, то эти устройства связаны с системной платой контроллерами,

хотя клавиатура и мышь являются достаточно медленными устройствами ввода (однако они обеспечивают непосредственный интерфейс с ПК).

Наконец, все узлы системной платы (рис.) связаны системным интерфейсом (СИ) типа "общая шина", организация которого зависит от модели и типа ПК и микро-ЭВМ. Он представляет собой систему линий передачи адресов, данных и управляющих сигналов различного типа и назначения.

Вопрос 1.9.3. Общие сведения о мини-ЭВМ. Общая архитектура и состав мини-ЭВМ.

Составляют достаточно малый класс, занимающий промежуточное положение между классами микро-ЭВМ (ПК) и ЭВМ общего назначения; мини-ЭВМ имеют ОП объемом порядка от 100 Мбайт до нескольких Гигабайт, приближаясь по вычислительным возможностям к ЭВМ общего назначения. В этом отношении среди мини-ЭВМ даже выделяют подкласс супер- мини-ЭВМ. В связи с развитием элементной базы, во многом общей для ЦВТ, традиционных и перспективных архитектур грань между классами ЭВМ становится весьма размытой и во многом начинает носить условный характер.

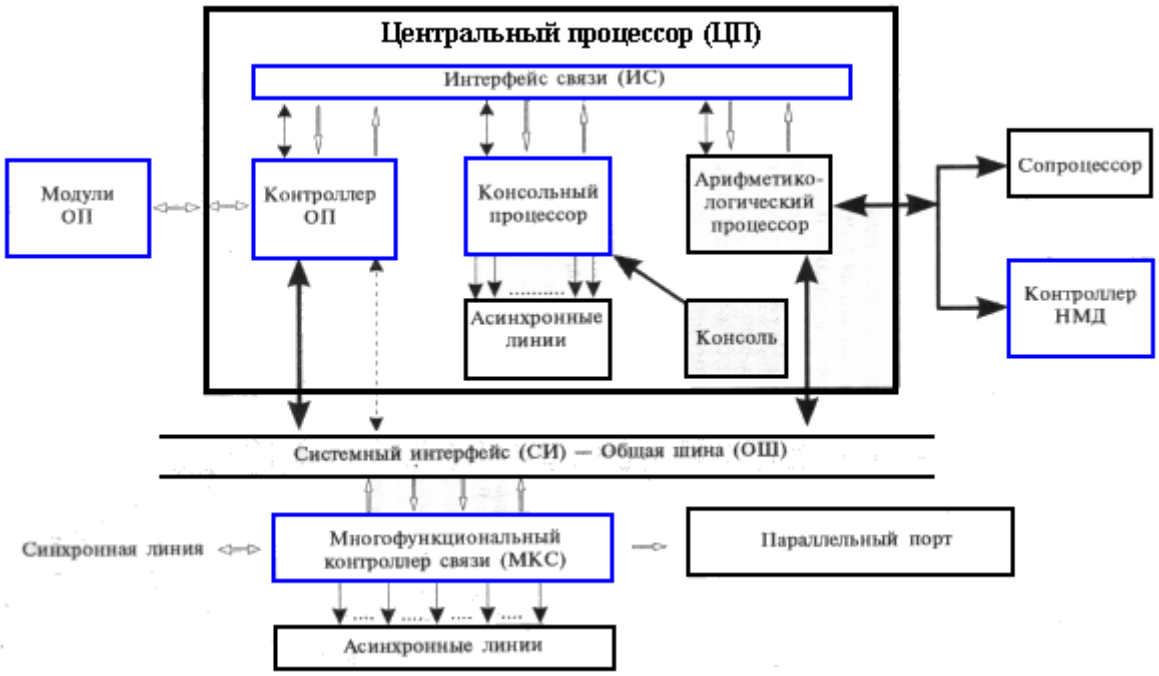
Характерной чертой современных мини-ЭВМ можно считать развитый многопользовательский режим доступа к вычислительным ресурсам, соответствующим по возможностям ЭВМ общего назначения среднего класса.

Современное архитектурное развитие мини-ЭВМ строится на использовании идей мультипроцессорности, параллельной обработки и RISC- подхода (RISC - Reduced Instruction Set Computer - ЭВМ с сокращенным набором команд). При этом большое внимание уделяется развитию системных интерфейсов (СИ), составляющих важную компоненту архитектуру мини-ЭВМ. СИ обеспечивает объединение основных блоков ЭВМ в единую информационную систему с центральным процессором (ЦП), ОП, контроллерами и портами внешних устройств. По типу реализуемой системы ввода/вывода СИ подразделяются на четыре основных класса:

1. сосредоточенные ;
2. локально - сосредоточенные;
3. локальные;
4. локально - распределенные.

Наряду с СИ, для построения мультипроцессорных систем, локальных и распределенных сетей используется межмашинный интерфейс.

Рисунок 1.9.3. Принципиальная схема мини-ЭВМ серии СМ-1700



Базовая мини-ЭВМ СМ-1700 включает центральный процессор (ЦП), ОП, математический сопроцессор, контроллер НМД, многофункциональный контроллер связи и СИ. В свою очередь ЦП состоит из консольного процессора (КП) и контроллера ОП. АЛП является микро-ЭВМ со своей системой микрокоманд АЛУ. Система микрокоманд использует семь основных форматов их дешифрации. АЛП имеет буфер предвыборки команд, позволяющий совмещать операции выборки и выполнения команд с быстродействием порядка 7 млн оп/с. Система команд СМ-1700 ориентирована на обработку большого количества типов данных: целые короткие и длинные, числа с плавающей запятой, битовые поля переменной длины, символьные и десятичные строки и т.д.

КП обеспечивает связь с консолью (оператором) мини-ЭВМ одновременно осуществляя диагностику системы и контроль параметров сети. К КП через С2-интерфейс подключают асинхронные линии связи для работы с удаленным терминалом, внешними устройствами памяти (НМЛ, НГМД). КП используются также для начальной загрузки диагностических программ и операционной системы и обеспечивает программируемые задержки и прерывания. Контроллер ОП обеспечивает управление модулями ОП, реализуя страничную организацию памяти и работу процессора без его блокировки при обращении к ОП по каналу прямого доступа. Он выполняет трансляцию поступающих виртуальных адресов (32-битных; процессор и 24-битных; ОШ) в физические 24-битные адреса памяти (обеспечивающие адресацию в 16 Мбайт), осуществляет работу канала прямого доступа в ОП согласно требованиям ОШ и защиту ОП от несанкционированного доступа; при чтении данных из ОП обеспечивается коррекция одиночных и обнаружение двоичных ошибок. Управление контроллера микропрограммное на основе 72-битных микрокоманд с объемом ПЗУ на 512

слов. Интерфейс связи обеспечивает логическую связь и связь по данным всех компонент ЦП: АЛП, КП и контроллера ОП.

ОП имеет модульную структуру, для которой обеспечивается максимальная адресация в 64 Мбайт. Математический сопроцессор предназначен для ускоренного выполнения команд (получаемых от АЛП) обработки чисел с плавающей точкой, преобразования целых чисел в числа с плавающей точкой и наоборот, а также для работы с полиномами. Контроллер НМД работает под управлением АЛП и обеспечивает обмен данными между АЛП и НМД типа "Винчестер". Многофункциональный контроллер связи (МКС) подключается непосредственно к ОШ и поддерживает режим прямого доступа к ОП. Использование режима совместимости и ОШ позволяет подключать к моделям СМ-1700 ранее разработанные 16-разрядные модели СМ ЭВМ; операционная система МОС ВМ поддерживает режим совместимости м ОС РВ, а созданные для СМ-4 устройства подключаются к СМ-1700 через ОШ. Характерной особенностью является наличие собственной системы микропрограммного управления для рассмотренных функциональных узлов мини-ЭВМ, что потребовало создания специальной системы автоматизации микропрограммирования.

Современные мультипроцессорные мини-ЭВМ с не - неймановской архитектурой развиваются по двум основным направлениям:

1. распараллеливание по программам;
2. распараллеливание по данным.

В первом случае на процессорах системы одновременно выполняются разные программы, во втором - процессоры ведут идентичную обработку разных частей данных. Наряду с развитием традиционной архитектуры на основе совершенствования элементной базы и нетрадиционных подходов к ней, используется и подход на основе сопроцессоров (векторных, матричных, функциональных, математических и др.). Основные особенности таких сопроцессоров состоят в аппаратной реализации алгоритмов и развитой системе векторных команд над кортежами данных в качестве операндов. Из направлений развития не - неймановской архитектуры мини-ЭВМ можно отметить следующие наиболее перспективные в настоящее время:

- **параллельные** ВС, ориентированные н6а решение задач моделирования больших систем, проведение сложных научно - технических расчетов, прогнозирование и др.;
- **единые** многофункциональные семейства моделей на основе RISC - подхода, ориентированные на решение сложных технических, технологических, производственных, управленческих и конструкторских задач;

- создание **спецпроцессоров** быстрого ассоциативного доступа к БД/БЗ, позволяющих использовать мини-ЭВМ в качестве эффективных машин баз данных и заданий;
- использование в архитектуре мини-ЭВМ принципов искусственного интеллекта (ИИ) и сопроцессоров векторного типа; при этом расширение внедрения элементов ИИ определяется эффективностью их сопряжения с традиционными архитектурами мини-ЭВМ.

Программное обеспечение (ПО) мини-ЭВМ носит многофункциональный характер, ориентировочный состав которого проиллюстрируем на примере СМ-1700:

- операционная система МОС ВП с СУБД СУД-32 и языками программирования: Макроассемблер, Fortran, Cobol, Pascal, PL/1, C, Basic, МОДУЛА, БЛИСС-32, КОРАЛ, ДИАМС;
- операционная система ДЕМОС-32 с СУБД реляционного типа, графическими средствами, средствами поддержки сетей, дополнительнbыми системами программирования Lisp и ПРОЛОГ;
- ПО организации и управления БД включает многофункциональную информационную систему (МИС СМ) и комплексную автоматизированную реляционную систему;
- ПО телеобработки данных включает ПО для обеспечения распределенных сетей (ТРАЛ), локальных сетей (МАГИСТР), распределенных многомашинных комплексов на базе СМ-1700 и ЕС ЭВМ (ЭМУЛЯТОР), сетей СМ ЭВМ с малыми ресурсами (МИНИ), локальных сетей кольцевого типа (КОЛОС) и однородную операционную среду для сети ЭВМ различных типов (ДЕМОС);
- ПС машинной графики и САПР включает базовое ПО автоматизации рабочих мест, средства машинной графики и проблемно-ориентированные ППП.

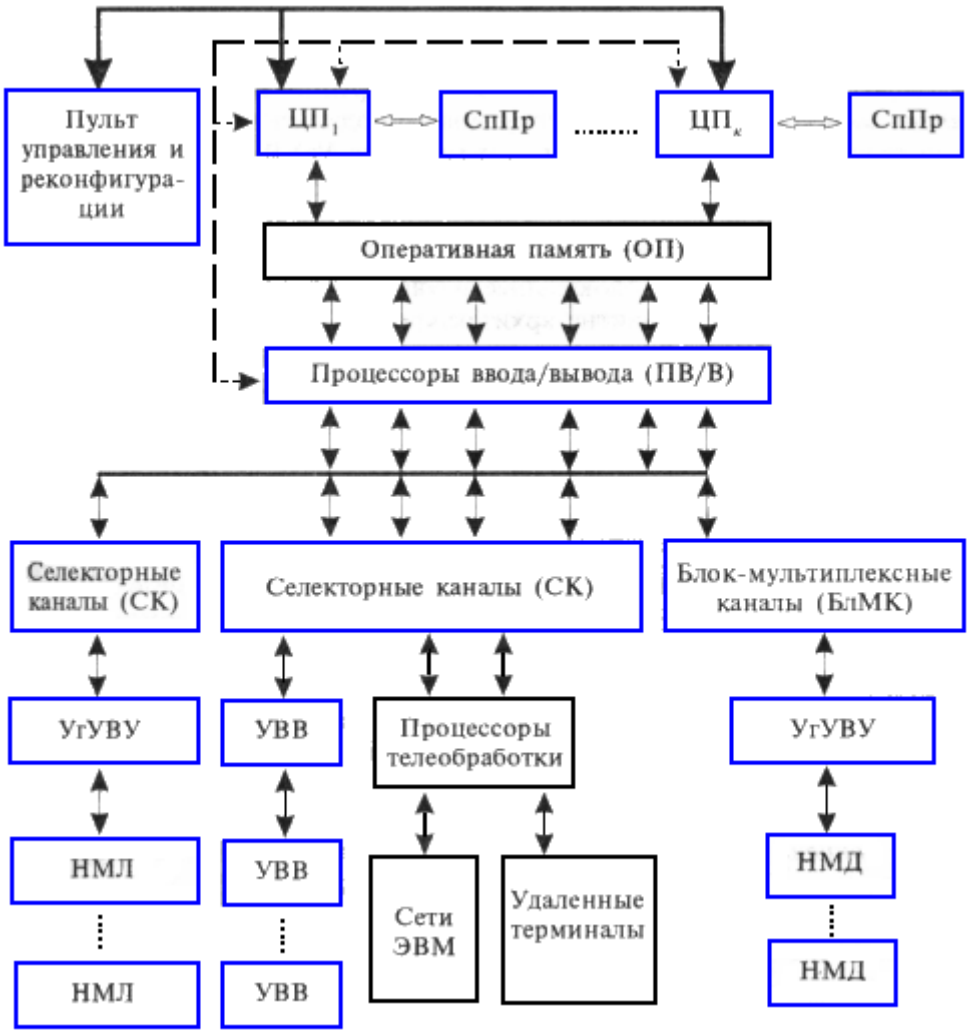
Вопрос 1.9.3.ЭВМ общего назначения

Обладают значительно большими быстродействием и вычислительными возможностями, чем мини-ЭВМ. Доступ к ним строго санкционирован по причине важности хранящейся в них информации, их стоимости и необходимости поддержания специального микроклимата. Основная память ЭВМ колеблется от сотен мегабайт до гигабайт, имея возможности для наращивания; их производительность измеряется десятками и сотнями миллионов операций в секунду и они могут поддерживать работу с тысячами удаленных терминалов и/или рабочих станций. ЭВМ общего назначения производятся в виде серий совместимых снизу вверх моделей с возрастающими возможностями (от младшей моделей к старшей). По производительности моделей серий ЭВМ принято делить на:

- младшие;
- средние;
- старшие.

Универсальность применения ЭВМ определяет следующие, определяющие этот класс машин, характерные черты: универсальность, совместимость, развитое ПО, агрегатность технических средств при широкой номенклатуре периферийных устройств, высокая технологичность и соответствие широко распространенным мировым стандартам. При этом под универсальностью понимается возможность эффективно решать задачи различных классов и типов из всех областей человеческой деятельности. Совместимость реализуется на аппаратно-программном уровне и предполагает наличие единого системного и прикладного ПО, совместимого снизу вверх для всех моделей одной и той же серии ЭВМ. При этом СПО ЭВМ характеризуется наличием развитых операционных систем, являющихся программным расширением аппаратных средств ЭВМ; обширное прикладное ПО ориентировано на самый широкий круг приложений. Агрегатный принцип организации технических средств, стандартный интерфейс ввода/вывода, позволяющий подключать различные периферийные устройства широкой номенклатуры, совместно с развитым ПО позволяют создавать разнообразные вычислительные комплексы, наиболее отвечающие конкретными приложениям. В качестве типичного примера архитектуры ЭВМ общего назначения можно привести организацию моделей ЕС ЭВМ, совместимых с IBM- сериями 360/370 (рис.).

Рисунок 1.9.4. Архитектура ЭВМ общего назначения.



Пульт управления и реконфигурации служит для обеспечения интерфейса оператора с ЭВМ, инженерного обслуживания и при наличии мультипроцессорного/многомашинного комплекса для реконфигурации системы. Центральные процессоры обеспечивают непосредственную обработку информации и управление основными устройствами системы. Центральные процессоры работают с ОП, имеющей модульную организацию и хранящей программы, управляющую информацию и информацию для оперативной обработки. Подсистема ввода/вывода включает процессоры ввода/вывода (ПВ/В) и три типа каналов:

1. селекторные (СК);
2. байтмультиплексные (БМК);
3. блок-мультиплексные (БлМК).

Она обеспечивает ввод/вывод информации, осуществляя связь с каналами и ОП под управлением процессоров.

Селекторные каналы работают в монопольном режиме с быстрыми внешними запоминающими (ВЗУ) устройствами. Мультиплексные каналы

представляют собой совокупность отдельных логических подканалов двух типов:

- неразделенных;
- разделенных.

Неразделенный подканал обслуживает только одно выделенное ему внешнее устройство (ВУ); при этом несколько неразделенных подканалов могут использовать общие для них ВУ. Разделенный подканал может в режиме разделения времени обслуживать разные ВУ.

Подсистема периферийных устройств выполняет функции хранения и ввода/вывода информации; она включает:

- устройства группового управления (УгУВУ);
- устройства ввода/вывода (УВВ);
- ВЗУ на основе НМЛ и НМД;
- процессорные телеобработки.

На основе стандартной схемы сопряжения: канал? УУ? ВЗУ к процессору можно подключать несколько каналов указанных типов, к каналу - несколько УгУВУ, а каждое УгУВУ может обслуживать группу ВЗУ. ЭВМ располагает весьма обширной номенклатурой различного типа ВУ. Подсистема телеобработки обеспечивает функции связи с удаленными абонентами и содержит процессоры телеобработки, локальные и удаленные абонентские пункты, а также сети ЭВМ. Развитие архитектуры ЭВМ приводит к созданию виртуальных ЭВМ, когда для пользователя снимаются ограничения на используемые вычислительные ресурсы (процессор, память, ввод/вывод) в разумных пределах и с различными оговорками.

Основным направлением нынешнего состояния и последующего развития архитектуры является интеллектуализация ЭВМ, определяющая эволюцию ЭВМ к системам искусственного интеллекта (ИИ). Новые качества ЭВМ 4-го поколения обеспечивают для пользователя следующие основные возможности:

- общение с ЭВМ без необходимости знания ее устройств и принципов функционирования;
- автоматизация разработки ПО;
- использование БД/БЗ с выходом в сети ЭВМ;
- работа с развитыми экспертными системами.

Предполагается, что ЭВМ 5-го поколения станут ядром распределенных локальных, региональных и глобальным информационно-вычислительных сетей, персональных и коллективных интеллектуальных рабочих станций с доступом ко всему объему накопленных человечеством знаний.

Концептуальная архитектура интеллектуальной ЭВМ включает следующие основные подсистемы:

- интеллектуального интерфейса с пользователем (доступ к ЭВМ и взаимодействие в процессе постановки и решения задач, используя естественные формы представления информации и понятия конкретной предметной области);
- анализа и логического вывода (выполнение метафункций выбора метода решения и синтеза программ с учетом контекста и содержания задачи, которые дополняются необходимым знаниями из БЗ; результатом является формирование алгоритма решения задачи);
- решения задач (поддерживается при необходимости проблемно-ориентированными спецпроцессорами);
- управления и обслуживания (обеспечение: надежности системы, перемещаемости программ и данных, автоматического контроля, диагностики и восстановления с целью поддержания живучести системы и других сервисных функций);
- база знаний (накопление, обработка и хранение разнообразных знаний, способствующих и обеспечивающих функциональные возможности ЭВМ интеллектуального характера, а также знание - ориентированных систем).
- Современные ЭВМ в той или иной степени включают в качестве составляющих своих архитектур характерные элементы отмеченных подсистем.

Важной составляющей развития архитектуры ЭВМ, является совершенствование ее системы команд, проводимое по трем основным направлениям. Традиционное направление связано с определением состава и содержания команд, способов адресации и спецификации операндов команд. Здесь четко прослеживается тенденция к увеличению числа команд, допускающих комплексную обработку данных; повышение смыслового уровня операций, обеспечивающих более компактное представление программ и эффективную микропрограммную реализацию команд. Второе направление связано с использованием ограниченных наборов команд (RISC-архитектура) и структур управляемым потоком данных. Третье направление состоит в повышении уровня машинного языка; прежде всего это выражается в развитии описания типов данных. Основной тенденцией в развитии структур ЭВМ общего назначения является разделение функций системы и максимальная специализация подсистем (обрабатывающей, памяти, ввода/вывода, управляющей и обслуживающей) для выполнения данных функций. В русле развития мини- и супер-ЭВМ ЭВМ общего назначения в качестве архитектурных используют многомашинные и мультипроцессорные решения, при необходимости включающие спецпроцессоры, преследующие цели повышений производительности, надежности и живучести ВС.

Контрольные вопросы.

1. Опишите общую архитектуру специальных ЭВМ.
2. Опишите структурную схему системной платы ПК.
3. Опишите общую архитектуру мини-ЭВМ.
4. Опишите общую архитектуру ЭВМ общего назначения.

Тема 1.10. Программное обеспечение ЭВМ

Вопросы.

1. Структура программного обеспечения ЭВМ.
2. Операционные системы. Структура ДОС ПЭВМ.
3. Системы автоматизации программирования.
4. Режимы работы ЭВМ.

Цели занятия:

- изучить структуру программного обеспечения ЭВМ
- изучить основные этапы загрузки ДОС ПК
- изучить режимы работы ЭВМ

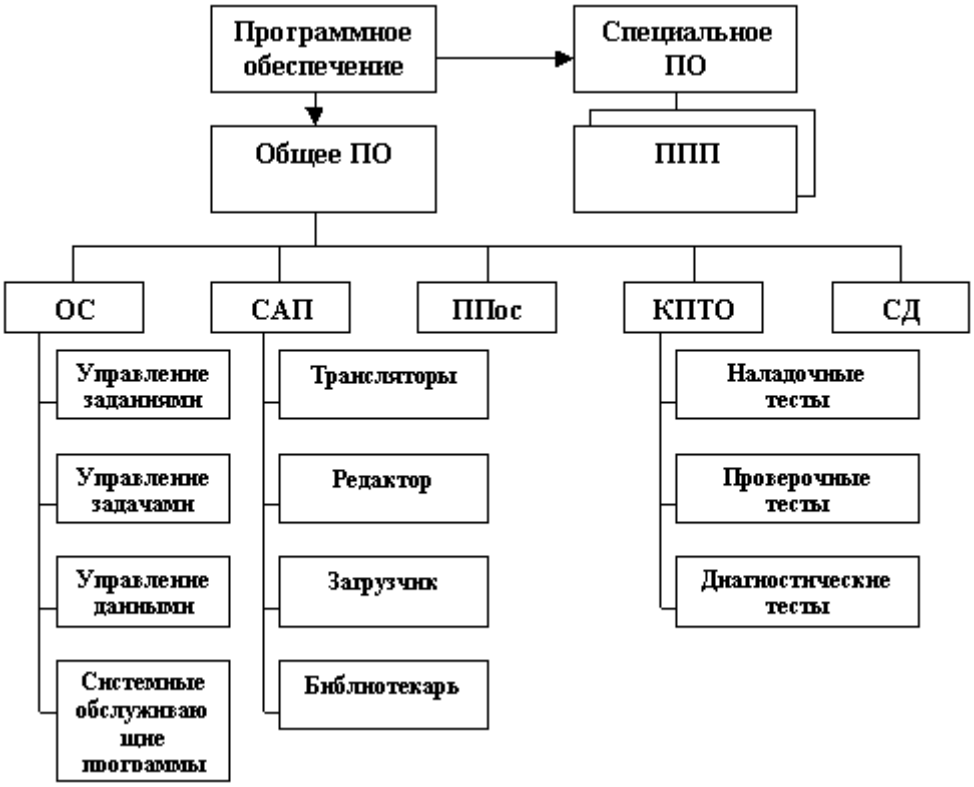
Вопрос 1.10.1. Структура программного обеспечения ЭВМ

Программное обеспечение ЭВМ разделяют на общее, или системное, и специальное, или прикладное (рис. 1.10.1).

Общее ПО объединяет программные компоненты, обеспечивающие многоцелевое применение ЭВМ и мало зависящие от специфики вычислительных работ пользователей. Сюда входят программы, организующие вычислительный процесс в различных режимах работы машин, программы контроля работоспособности ЭВМ, диагностики и локализации неисправностей, программы контроля заданий пользователей, их проверки, отладки и т.д.

Специальное ПО (СПО) содержит пакеты прикладных программ пользователей (ППП), обеспечивающее специфическое применение ЭВМ и вычислительной системы.

Рисунок 1.10.1. Структура программного обеспечения.



Прикладной программой называется программный продукт, предназначенный для решения конкретной задачи пользователя.

Специализация пакета определяется характером определяемых задач или необходимостью управления специальной техникой.

Общее ПО включает в свой составоперационную систему (ОС), средства автоматизации программирования (САП), комплекс программ технического обслуживания (КПТО), пакеты программ, дополняющие возможности ОС (ППос), и систему документации (СД).

Вопрос 1.10.1. Операционные системы. Структура ДОС ПЭВМ

Операционные системы. Центральное место в структуре ПО занимает операционная система. Она представляет собой "систему программ".

Программные компоненты ОС обеспечивают управление вычислениями и реализует такие функции, как планирование и распределение ресурсов, управлением ввода-вывода информации, управление данными.

Применение ОС имеет следующие цели:

- увеличение пропускной способности ЭВМ
- уменьшение времени реакции системы
- контроль работоспособности технических и программных средств

- помощь абонентам и операторам при использовании ими технических и программных средств, облегчения их работы
- управления программами и данными в ходе вычислений
- обеспечение адаптации ЭВМ, её структурной гибкости, заключается в способности изменяться, пополнятся новыми техническими и программными средствами.

Любая ОС имеет средства приспособления к классам решаемых пользователями задач и конфигурации средств, включаемых в ВС.

В больших ЭВМ формирование конкретной конфигурации ОС осуществляется на нескольких уровнях. Предварительно этот состав определяется при генерации ОС. "Генерация системы - это процесс выделения отдельных частей операционной системы и построения частотных операционных систем, отвечающих требованиям системы обработки данных". Коррекция же состава используемых услуг может быть выполнена непосредственно перед решением задач операторами вычислительного центра или самими пользователями.

Вычислительный процесс в системе представляется в виде последовательности, как правило, ветвящийся, простых процессов - однообразных работ, выполняемых ресурсами ВС.

Ресурсы ВС - это средства, необходимые для вычислений. К ресурсам ВС относятся: машинное время ЭВМ, объёмы внешней и оперативной памяти, любые внешние устройства, подключаемых к ВС.

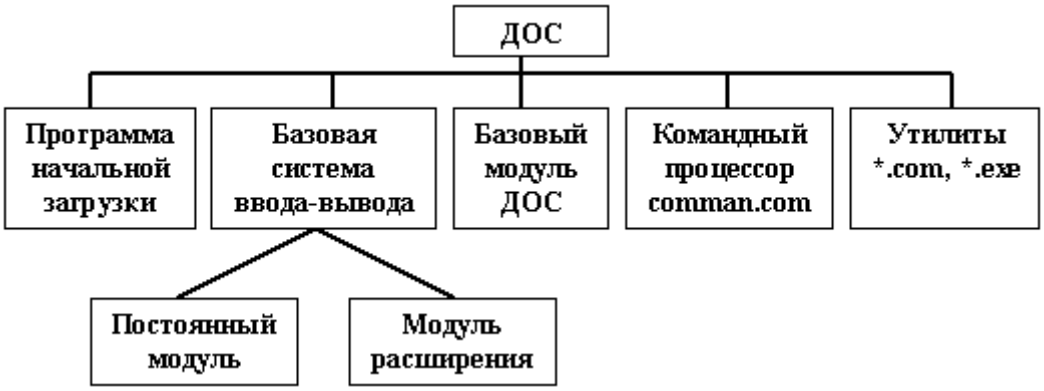
Основу любой ОС составляет управляющая программа, основными функциями которой являются: управление заданиями, управление задачами - управление ходом выполнения отдельных программ, и управление данными.

Задание - это требование пользователя на выполнение некоторого объёма вычислительных работ.

Каждое задание реализуется как определённая последовательность отдельных программ - задач. Задачи образуют отдельные программы вместе с обрабатываемыми ими данными.

Структурно ОС состоит из следующих элементов, представленных на рис 1.10.2. Кроме программных компонентов, указанных на рисунке, к ДОС относят ещё вспомогательные файлы autoexec.bat и config.sys. Они предназначаются для настройки на конкретные режимы работы.

Рисунок 1.10.2. Структура ДОС ПЭВМ



Программа начальной загрузки (Boot Record) находится в первом секторе на нулевой дорожке системного диска. Она занимает объём 512 байт. После включения компьютера и его проверки постоянный модуль BIOS формирует вызов данной программы и её запуск. Назначением программы начальной загрузки является вызов модуля расширения IO.sys и базового модуля ДОС MS DOS.sys.

Базовая система ввода-вывода (BIOS) является надстройкой аппаратурной части компьютера. Постоянный модуль BIOS отвечает за тестирование компьютера после его включения, вызов программы начальной загрузки. Модуль BIOS обрабатывает прерывание вычислительного процесса нижнего уровня и обслуживает стандартную периферию: дисплей, клавиатуру, принтер, дисководы.

Модуль расширения BIOS обеспечивает подключение к компьютеру дополнительных периферийных устройств, изменения некоторых параметров ДОС, замещение некоторых стандартных функций, загрузку командного процессора и его запуск.

Командный процессор (command.com) предназначен для выполнения команд, загружаемых в командную строку ДОС. Все команды ДОС делятся на внутренние и внешние. Внутренние команды содержаться внутри самого файла comand.com. Внешние команды - это требования запуска каких-либо программ находящихся на дисках.

Командный процессор ещё выполняет программы, записанные в файле autoexec.bat, если он имеется на диске.

Файл autoexec.bat содержит список команд, выполнение которых позволяет развернуть в оперативной памяти некоторый набор вспомогательных программ.

Файл config.sys отражает специфические особенности формирования конфигураций компьютера, т.е. состава его технических и программных средств.

Вопрос 1.10.2. Система автоматизации программирования

К системам (или средствам) автоматизации программирования (САП) относят языки программирования, языковые трансляторы, редакторы, средства отладки и другие вспомогательные программы.

Языки программирования служат средством передачи информации, средством записи текстов исходных программ.

Различаю три уровня пользователей, работающих с языковыми средствами: пользователи-прикладники, системные программисты и инженерно-технический персонал, обеспечивающий техническое обслуживание.

Качество программ определяется длиной программы, а так же временем выполнения этих программ. Чем выше уровень языка рис. 1.10.3, тем меньше трудоёмкость программирования, но тем сложнее средства САП, привлекаемые для получения машинных программ, тем ниже качество генерируемых программных продуктов.

Рисунок 1.10.3. Классификация языков программирования.

Машинные языки современых ЭВМ практически не используются для программирования.

К машинно-ориентированным языкам программирования (ЯП) относится язык Ассемблер.

Из процедурно-ориентированных языков широко известны языки Фортран, Алгол, Кобол, Basic, Pascal, Ада, Си. Перечисленные ЯП используют так называемые пошаговые описания алгоритмов, что вызывает большие трудности подготовки задач к решению. Непроцедурные описательные языки состоят только из описаний и не имеют как таковых команд-инструкций. Примером такого языка служит язык Пролог (ПРОграммирование ЛОГики), который широко используется специалистами в области интеллекта. В состав САП включаются языковые трансляторы. Различают трансляторы двух типов: трансляторы-интерпретаторы, трансляторы-компиляторы.

Трансляторы-интерпретаторы предназначаются для последовательного пооператорного преобразования каждого предложения исходного модуля программы в блок машинных команд с одновременным их выполнением. Машинная программа в полном обьеме при этом не создается.

Трансляторы-компиляторы предназначены для формирования полного загрузочного модуля по исходным программам пользователя.

При подготовке вычислений используются системные обслуживающие программы: редактор, загрузчик, отладчик, библиотекарь, средства отладки.

Вопрос 1.10.4. Режим работы ЭВМ

Под режимом работы понимают принцип структурной и функциональной организации аппаратных и программных средств. В общем случае режимы использования ЭВМ подразделяют на однопрограммные и многопрограммные.

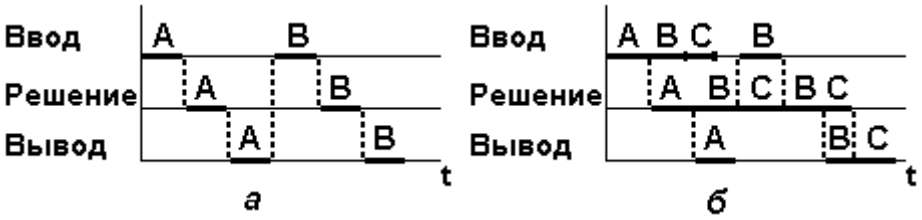
Однопрограммный режим имеет модификации: однопрограммный режим непосредственного доступа и однопрограммный режим косвенного доступа.

В режиме непосредственного доступа пользователь получает ЭВМ в полное распоряжение: он сам готовит ЭВМ к работе, загружает задания, инициализирует их, наблюдает за ходом решения и выводом результатов. По окончании работ одного пользователя все ресурсы ЭВМ передаются в распоряжения другого (рис.1.10.4 а). Этот тип режима характеризуется весьма низкой полезной загрузкой технических средств. К снижению производительности ЭВМ из-за простоев процессора приводят затраты времени на подготовку ЭВМ к работе и большое время реакции пользователя. По этим причинам режим практически не используется в универсальных ЭВМ.

В режиме косвенного доступа пользователь не имеет прямого контакта с ЭВМ. Этот режим имел целью обеспечить более полную загрузку процессора за счёт сокращения непроизводительных его простоев. В настоящее время он практически не используется.

Суть режима состоит в следующем.

Рисунок 1.10.4.



Из подготовленных заданий пользователь составляет пакет заданий. Процессор обслуживает программы пользователя строго в порядке их следования в пакете. Процесс выполнения очередной программы не прерывается для полного его завершения. Только после этого процессор как ресурс отдается в монопольное владение следующей очередной программе. При этом режиме обеспечивается параллельная работа устройств ввода-вывода и процессора (рис. 1.10.4 б).

Многопрограммный режим работы ЭВМ позволяет одновременно обслуживать несколько программ пользователей. Реализация режима требует соблюдения следующих непременных условий:

- независимости подготовки заданий пользователями
- разделения ресурсов ЭВМ в пространстве и времени
- автоматического управления вычислениями

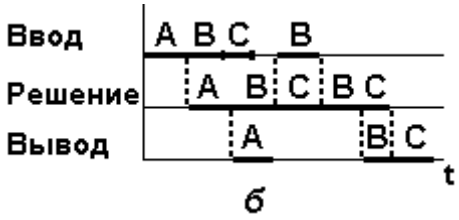
Разделение ресурсов ЭВМ между программами пользователь обеспечивается аппаратно-программными средствами системами. Программы управления заданиями ОС определяют видами требуемых ресурсов в заданиях пользователей и регламентируют их использование. Отдельные виды ресурсов, например области оперативной и внешней памяти, допускают одновременное их использование программами пользователей. В этом случае пространство адресов памяти разбивается на непересекающиеся зоны или разделы. "Охрану границ" этих зон обеспечивают схемы защиты памяти - аппаратурные или программные средства ЭВМ. Некоторые виды ресурсов допускают только последовательное их использование программами пользователей, например, в однопроцессорной ЭВМ время работы единственного процессора является неразделимым ресурсом. Его использование предполагает упорядочение потока заявок и поочередное его использование программами. В современных ЭВМ упорядочение потока заявок обеспечивается на основе их приоритетов, где приоритет - некоторая априорная характеристика заявки, определяющая её место в очереди на обслуживание. Формирование очередей обеспечивают программные компоненты ОС. Обслуживание очередей заявок выполняется с использованием системы прерываний и приоритетов. Последняя выделяет из группы одновременно поступающих заявок одну, наиболее приоритетную.

Автоматическое управление вычислительным процессом в многопрограммном режиме выполняется центральной программой управления задачами. Сущность управления сводится к управлению ресурсами. При этом ОС составляет таблицы управления, выделяет ресурсы, запускает их в работу и корректирует таблицы.

Различные формы многопрограммных режимов работы различаются в основном значимостью различного рода ресурсов и правилами перехода от обслуживания одной программы пользователя к другой.

Различают следующие виды многопрограммной работы: классическое мультипрограммирование, режим разделения времени, режим реального времени.

Рисунок 1.10.5.

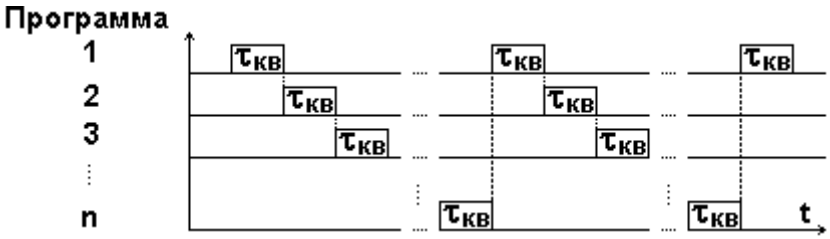


б

Режим классического мультипрограммирования, или пакетной обработки, применительно к однопроцессорным ЭВМ является основой для построения всех других видов для построения многопрограммной работы. Режим имеет цель обеспечить минимальное время обработки заданий и максимальную загрузку процессора. Пакет заданий упорядочивается в соответствие с приоритетами заданий, и обслуживание программ ведётся в порядке очерёдности. Обычно процессор обслуживает наиболее приоритетную программу. Как только её решение завершается, процессор переключается на следующую по приоритетности программу. Если при обслуживании наиболее приоритетной программы создаётся ситуация, что вычисления не могут быть продолжены, то прерывание обслуживания сопровождается передачей управления следующей по приоритетности программе. Но как только условия, препятствующие продолжению наиболее приоритетной задачи, отпадут, процессор вновь возвращается к решению ранее прерванной программы. Этот случай иллюстрируется на рис 1.10.5. При выполнении задания В.

Режим разделения времени является более развитой формы многопрограммной работы ЭВМ. В этом режиме, обычно совмещённом с фоновым режимом классического мультипрограммирования, отдельные наиболее приоритетные программы пользователей выделяются в одну или несколько групп. Для каждой такой группы устанавливается круговое циклическое обслуживание, при котором каждая программа группы периодически получает для обслуживания достаточно короткий интервал времени - время кванта - $\tau_{кв}$ (рис. 1.10.6).

Рисунок 1.10.6. Режим разделения времени.



После завершения очередного цикла процесс выделения квантов повторяется. Это создаёт у пользователей ощущения кажущейся одновременности выполнения их программ. Если к пользователю к тому же предоставляются средства прямого доступа для вывода результатов решения, то это впечатление ещё более усиливается, так как результаты выдаются в ходе вычислений по программе, не ожидая завершения обслуживания всех программ группы или пакета программ.

Условием прерывания текущей программы является либо истечение выделенного кванта времени, либо естественное завершение решения, либо прерывание по вводу-выводу. Для реализации режима разделения времени необходимо, чтобы ЭВМ имела в своём составе развитую систему измерения времени: интервальный таймер, таймер процессора, электронные часы и т.д. Это позволяет формировать группы программ с постоянным или переменным кванта времени - $\tau_{кв}$. Разделение времени находит широкое применение при обслуживании ЭВМ сети абонентских пунктах.

Более сложной формой разделения времени является режим реального времени. Этот режим имеет специфические способности:

- поток заявок от абонентов носит, как правило, случайный, непредсказуемый характер
- потери поступающих на вход ЭВМ заявок и данных к ним не допускаются, поскольку их не всегда можно восстановить
- время реакции ЭВМ на внешние воздействия, а также время выдачи результатов i-й задачи должны удовлетворять жестким ограничениям вида $t_р \leq t_р^{доп}$, где $t_р$ - время решения задачи; $t_р^{доп}$ - допустимое время решения.

Специфические особенности режима реального времени требуют наиболее сложных ОС. Именно на базе этого режима строятся так называемый диалоговые системы, обеспечивающие многопользовательский режим: одновременную работу нескольких пользователей с ЭВМ. Диалоговые системы могут иметь различное содержание: системы, обслуживающие наборы данных; системы разработки документов, программ, схем, чертежей; системы выполнения программ в комплексе "человек - машина" и др. Диалоговый режим обслуживания предполагает использование дисплеев - устройств оперативного взаимодействия с ЭВМ. Они получили широкое

распространение в различных информационных и автоматизированных системах управления.

Контрольные вопросы

1. Опишите структуру программного обеспечения ЭВМ.
2. Опишите структуру ОС ДОС ПЭВМ.
3. Опишите систему автоматизации программирования.
4. Перечислите и поясните режимы работы ЭВМ.

Раздел II. Архитектура вычислительных систем [16]

Тема 2.1. Классификация вычислительных систем

Цель изучения темы:

- усвоить основные понятия и признаки квалификации ВС.
- изучить основные типы ВС.

Вопросы

1. Понятие ВС. Принципы построения ВС. Классификация ВС.
2. Типы ВС.

Вопрос 2.1.1. Понятие ВС. Принципы построения ВС. Классификация ВС.

ЭВМ построенные на основе неймановской архитектуры и рассчитанные на на последовательное выполнение команд имеют небольшой предел роста производительности. Во многих теоретических и прикладных областях требуется быстродействие порядка 200 млрд оп/с., что принципиально недостижимо на ЭВМ с традиционной последовательной архитектурой. Решение данной проблемы возможно при переходе на параллельную не-неймановскую архитектуру ЭВМ. Параллельная архитектура предусматривает построение многопроцессорных систем и сетей, обьединяющих множество отдельных процессоров или ЭВМ.

Под **вычислительной системой** (ВС) понимают совокупность взаимосвязанных и взаимодействующих процессоров или ЭВМ, периферийного оборудования и программного обеспечения, предназначенную для подготовки и решения задач пользователей.

Основные принципы построения ВС:

- возможность работы в разных режимах;
- модульность структуры технических и программных средств, что позволяет совершенствовать и модернизировать вычислительные системы без коренных их переделок;
- унификация и стандартизация технических и программных решений;
- иерархия в организации управления процессами;
- способность систем к адаптации, самонастройке и самоорганизации;
- обеспечение необходимым сервисом пользователей при выполнении вычислений.

Структура ВС- это совокупность комплексируемых элементов и их связей. В качестве элементов ВС выступают отдельные ЭВМ и процессоры.

ВС классифицируются по следующим признакам:

- по целевому назначению и выполняемым функциям;
- по типам и числу ЭВМ или процессоров;
- по архитектуре системы;
- режимам работы;
- методам управления элементами системы;
- степени разобщенности элементов вычислительной системы .

По назначению вычислительные системы делят на:

- универсальные;
- специализированные.

Универсальные ВС предназначаются для решения самых различных задач.

Специализированные системы ориентированы на решение узкого класса задач. Специализация ВС может устанавливаться различными средствами:

- во-первых, сама структура системы (количество параллельно работающих элементов, связи между ними и т.д.) может быть ориентирована на определенные виды обработки информации: матричные вычисления, решение алгебраических, дифференциальных и интегральных уравнений и т.д. Практика разработки ВС типа суперЭВМ показала, чем выше их производительность, тем уже класс эффективно решаемых ими задач;
- во-вторых, специализация ВС может закладываться включением в их состав специального оборудования и специальных пакетов обслуживания техники.

По типу вычислительные системы различаются на:

- многомашинные;
- многопроцессорные.

По **типу ЭВМ или процессоров**, используемых для построения ВС, различают однородные и неоднородные системы.

По **степени территориальной разобщенности** вычислительных модулей ВС делятся на системы **совмещенного (сосредоточенного) и распределенного (разобщенного)** типов. Обычно такое деление касается только ММС. Многопроцессорные системы относятся к системам совмещенного типа. При появлении новых СБИС (сверхбольших интегральных схем) появляется

возможность иметь в одном кристалле несколько параллельно работающих процессоров.

Совмещенные и распределенные ММС сильно различаются оперативностью взаимодействия от удаленности ЭВМ. Время передачи информации между соседними ЭВМ, соединенными простым кабелем, может быть много меньше времени передачи данных по каналам связи. Как правило, ЭВМ имеют средства прямого взаимодействия и средства подключения к сетям ЭВМ. Для ПЭВМ такими средствами являются нуль - модемы, модемы и сетевые карты как элементы техники связи.

Под **методом управления элементами** ВС различают централизованные, децентрализованные и со смешанным управлением. Помимо параллельных вычислений, производимых элементами системы, необходимо выделять ресурсы на обеспечение управления этими вычислениями. В **централизованных** ВС за это отвечает главная, или диспетчерская, ЭВМ (процессор). Ее задачей является распределение нагрузки между элементами, выделение ресурсов, контроль состояния ресурсов, координация взаимодействия. Централизованный орган управления в системе может быть жестко фиксирован или эти функции могут передаваться другой ЭВМ (процессору), что способствует повышению надежности системы. Централизованные системы имеют более простые ОС. В **децентрализованных** системах функции управления распределены между ее элементами. Каждая ЭВМ (процессор) системы сохраняет известную автономию, а необходимое взаимодействие между элементами устанавливается по специальным наборам сигналов. С развитием ВС и, в частности, сетей ЭВМ интерес к децентрализованным системам постоянно растет. В системах **со смешанным управлением** совмещаются процедуры централизованного и децентрализованного управления. Перераспределение функций осуществляется в ходе вычислительного процесса исходя из сложившейся ситуации.

По **принципу закрепления вычислительных функций** за отдельными ЭВМ (процессорами) различают системы с **жестким и плавающим закреплением функций.** в зависимости от типа ВС следует решать задачи статического или динамического размещения программных модулей и массивов данных, обеспечивая необходимую гибкость системы и надежность ее функционирования.

По **режиму работы** ВС различают системы, работающие в **оперативном и неоперативном временных режимах.** Первые, как правило, используют режим реального масштаба времени. Этот режим характеризуется жесткими ограничениями на время решения задач в системе и предполагает высокую степень автоматизации процедур ввода-вывода и обработки данных.

Эффективность применения ВС определяется структурными признаки ВС. Структурные признаки ВС определяются: топологией управляющих и информационных связей между элементами системы, способностью системы к перестройке и перераспределению функций, иерархией.

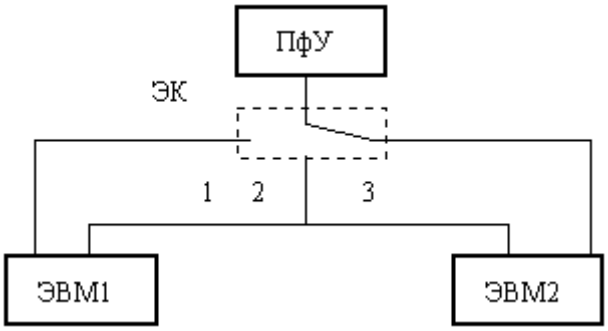
Вопрос 2.1.2. Типы ВС.

Различают два типа вычислительных систем:

- многомашинные;
- многопроцессорные.

Многомашинные вычислительны системы (ММС) обеспечивают: повышения производительности, надежности и достоверности вычислений. Для этих целей использовали комплекс машин, схематически показанный на рис. 2.2.1.

Рисунок 2.1.1. Многомашинный комплекс.



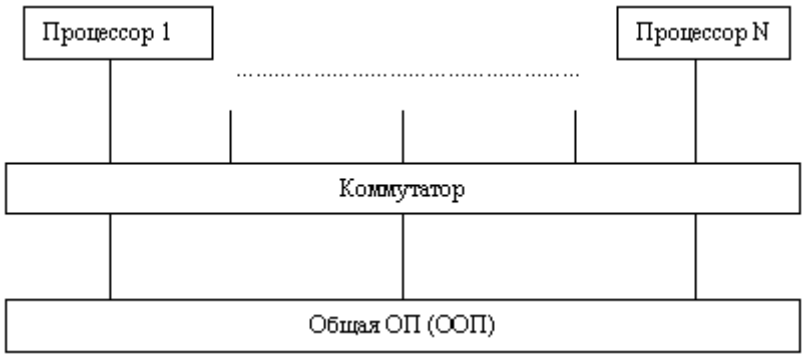
Положения 1 и 3 электронного ключа (ЭК) обеспечивало режим повышенной надежности. При этом одна из машин выполняла вычисления, а другая находилась в "горячем" или "холодном" резерве, т.е. в готовности заменить основную ЭВМ. Положение 2 электронного ключа соответствовало случаю, когда обе машины обеспечивали параллельный режим вычислений. Здесь возможны две ситуации:

1. обе машины решают одну и ту же задачу и периодически сверяют результаты решения. Тем самым обеспечивался режим повышенной достоверности, уменьшалась вероятность появления ошибок в результатах вычислений;
2. обе машины работают параллельно, но обрабатывают собственные потоки заданий. Возможность обмена информацией между машинами сохраняется. Этот вид работы относится к режиму повышенной производительности. Такая схема используется для организации работ на крупных вычислительных центрах, оснащенных несколькими ЭВМ высокой производительности.

Основные отличия ММС разных модификаций заключаются, как правило, в организации связи и обмена информацией между ЭВМ комплекса. Каждая из них сохраняет возможность автономной работы и управляется собственной ОС. Любая другая подключаемая ЭВМ комплекса рассматривается как специальное периферийное оборудование. В зависимости от территориальной разобщенности ЭВМ и используемых средств сопряжения обеспечивается различная оперативность их информационного воздействия.

Многопроцессорные вычислительные системы (МПС) строятся при комплексировании нескольких процессоров (рис. 2.2.2). Общая оперативная память (ООП) в МПС используется в качестве общего ресурса. Параллельная работа процессоров и использование ООП обеспечивается под управлением единой общей операционной системы. По сравнению с ММС здесь достигается наивысшая оперативность взаимодействия вычислителей - процессоров.

Рисунок 2.1.2. Многопроцессорные системы.



Недостатки МПС:

1. При большом количестве комплексируемых процессоров возможно возникновение конфликтных ситуаций, когда несколько процессоров обращаются с операциями типа "чтение" и "запись" к одним и тем же областям памяти.
2. Проблема коммутации абонентов и доступа их к ООП, т,к, помимо процессоров к ООП подключаются все каналы (процессоры ввода-вывода), средства измерения времени и т.д.

От того, насколько удачно решаются эти проблемы, и зависит эффективность применения МПС. Это решение обеспечивается аппаратно-программными средствами. Процедуры взаимодействия очень сильно усложняют структуру ОС МПС. МПС эффективны при небольшом числе комплексируемых процессоров (2,4 до 10). В отечественных системах Эльбрус обеспечивалась возможность работы до десяти процессоров, до 32 модулей памяти, до 4 процессоров ввода-вывода и до 16 процессоров связи. Все связи в системе обеспечивались коммутаторами.

Создание подобных коммутаторов представляет сложную техническую задачу, тем более что они должны быть дополнены буферами для организации очередей запросов. Для разрешения конфликтных ситуаций необходимы схемы приоритетного обслуживания.

По **типу ЭВМ или процессоров**, используемых для построения ВС, различают однородные и неоднородные системы.

Однородные системы предполагают комплексирование однотипных ЭВМ (процессоров), неоднородные - разнотипных. В однородных системах значительно упрощаются разработка и обслуживание технических и программных (в основном ОС) средств. В них обеспечивается возможность стандартизации и унификации соединений и процедур взаимодействия элементов системы. Упрощается обслуживание систем, облегчаются модернизация и их развитие. Вместе с тем существуют и неоднородные ВС, в которых комплексируемые элементы очень сильно отличаются по своим техническим и функциональным характеристикам. Обычно это связано с необходимостью параллельного выполнения многофункциональной обработки. Так, при построении ММС, обслуживающих каналы связи, целесообразно объединять в комплекс связанные, коммуникационные машины и машины обработки данных. В таких системах коммуникационные ЭВМ выполняют функции связи, контроля получаемой и передаваемой информации, формирования пакетов задач и т.д. ЭВМ обработки данных не занимаются несвойственными им работами по обеспечению взаимодействия в сети, а все их ресурсы переключаются на обработку данных. Неоднородные системы находят применение и в МПС. Многие ЭВМ, в том числе и ПЭВМ, могут использовать сопроцессоры: десятичной арифметики, матричные и т.п.

Контрольные вопросы

1. Каково понятие ВС? Перечислите основные принципы построения ВС. Каково понятие структура ВС?
2. Перечислите и поясните основные кфалификационные признаки ВС.
3. Опишите многомашинные вычислительны системы (ММС).
4. Опишите многопроцессорные вычислительны системы (ММС).

Тема 2.2. Архитектура вычислительных систем

Цель изучения темы:

- Усвоить четыре основные архитектуры ВС.
- Изучить отличительные особенности архитектур ВС.

Вопросы.

1. Четыре основные архитектуры ВС.
2. Отличительные особенности архитектур ВС.

Вопрос 2.2.1. Четыре основные архитектуры ВС.

Большое разнообразие структур ВС затрудняет их изучение. Поэтому ВС классифицируют с учетом их обобщенных характеристик. С этой целью вводится понятие **архитектура системы.**

Архитектура ВС - совокупность характеристик и параметров, определяющих функционально - логическую и структурную организацию системы. Понятие архитектуры охватывает общие принципы построения и функционирования, наиболее существенные для пользователей, которых больше интересуют возможности систем, а не детали их технического исполнения. Поскольку ВС появились как параллельные системы, то и рассмотрим классификацию архитектур под этой точкой зрения.

Эта классификация была предложена Флинном (M. Flynn) в начале 60-х гг. В ее основу заложено два возможных вида параллелизма:

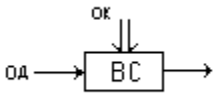
1. независимость потоков заданий (команд), существующих в системе;
2. независимость (несвязанность) данных, обрабатываемых в каждом потоке.

С появлением систем, ориентированных на потоки данных и использующих ассоциативную обработку, классификация может быть некорректной.

Согласно данной классификации существуют четыре основные архитектуры ВС:

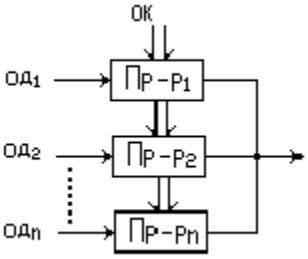
1. Одиночный поток команд - одиночный поток данных (ОКОД), в англоязычной аббревиатуре Single Instruction Single Data (SISD) - одиночный поток инструкций - одиночный поток данных (рис 2.2.1).

   Рисунок 2.2.1. ОКОД (SISD) - архитектура.



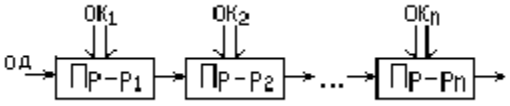
2. Одиночный поток команд - множественный поток данных (ОКМД), или Single Instruction Multiple Data (SIMD), - одиночный поток инструкций - одиночный поток данных (рис.2.2.1).

Рисунок 2.2.2. ОКМД (SIMD) - архитектура.



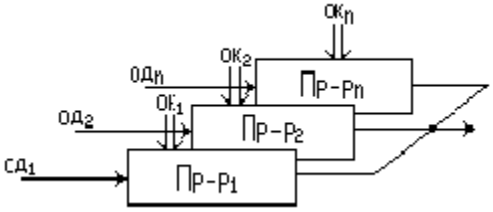
3. Множественный поток команд - одиночный поток данных (МКОД), или Multiple Instruction Single Data (MISD), - множественный поток инструкций - одиночный поток данных (рис. 2.2.3).

Рисунок 2.2.3. МКОД (MISD) - архитектура.



4. Множественный поток команд - множественный поток данных (МКМД), или Multiple Instruction Multiple Data (MIMD), - множественный поток инструкций - множественный поток данных (рис 2.2.4).

Рисунок 2.2.4. МКМД (MIMD) - архитектура



Вопрос 2.2.2. Отличительные особенности архитектур ВС.

Архитектура ОКОД (SISD) охватывает все однопроцессорные и одномашинные варианты систем, т.е. с одним вычислителем. Все ЭВМ

классической структуры попадают в этот класс. Здесь параллелизм вычислений обеспечивается путем совмещения выполнения операций отдельными блоками АЛУ, а также параллельной работой устройств ввода-вывода информации и процессора.

Архитектура ОКМД (SIMD) предполагает создание структур векторной или матричной обработки. Системы этого типа обычно строятся как однородные, т.е. процессорные элементы, входящие в систему, идентичны, и все они управляются одной и той же последовательностью команд. Однако каждый процессор обрабатывает свой поток данных. Под эту схему хорошо подходят задачи обработки матриц или векторов (массивов), задачи решения систем линейных и нелинейных, алгебраических и дифференциальных уравнений, задачи теории поля и др. В структурах данной архитектуры желательно обеспечивать соединения между процессорами, соответствующие реализуемым математическим зависимостям. Как правило, эти связи напоминают матрицу, в которой каждый процессорный элемент связан с соседними.

Архитектура МКОД (MISD) предполагает построение своеобразного процессорного конвейера, в котором результаты обработки передаются от одного процессора к другому по цепочке. Выгоды такого вида обработки понятны. Прототипом таких вычислений может служить схема любого производственного конвейера. В современных ЭВМ по этому принципу реализована схема совмещения операций, в которой параллельно работают различные функциональные блоки, и каждый из них делает свою часть в общем цикле обработки команды.

В ВС этого типа конвейер должны образовывать группы процессоров. Однако при переходе на системный уровень очень трудно выявить подобный регулярный характер в универсальных вычислениях. Кроме того, на практике нельзя обеспечить и "большую длину" такого конвейера, при которой достигается наивысший эффект. Вместе с тем конвейерная схема нашла применение в так называемых скалярных процессорах суперЭВМ, в которых они применяются как специальные процессоры для поддержки векторной обработки.

Архитектура МКМД(MIMD) предполагает, что все процессоры системы работают по своим программам с собственным потоком команд. В простейшем случае они могут быть автономны и независимы. Такая схема использования ВС часто применяется на многих крупных вычислительных центрах для увеличения пропускной способности центра. Большой интерес представляет возможность согласованной работы ЭВМ (процессоров), когда каждый элемент делает часть общей задачи. Общая теоретическая база такого вида работ практически отсутствует. Но можно привести примеры большой эффективности этой модели вычислений. Подобные системы могут

быть многомашинными и многопроцессорными. Например, отечественный проект машины динамической архитектуры (МДА) - ЕС-2704, ЕС-2727 позволял одновременно использовать сотни процессоров.

Контрольные вопросы.

1. Каково понятие архитектуры ВС? Перечислите основные архитектуры ВС. Представте их структурные схемы.
2. Опишите основные архитектуры ВС.

Тема 2.3. Типовые структуры вычислительных систем

Цель изучения темы: перечислить и охарактеризовать типовые структуры ВС; изучить структуру и состав суперЭВМ; изучить структуру и состав параллельной суперЭВМ.

Содержание:

- типовые структуры ВС;
- три аспекта совместимости: аппаратный, программный, информационный;
- уровни комплексирования. Классификация уровней программного параллелизма: независимые задания, отдельные части задания, программы и подпрограммы, циклы и итерации, операторы и команды, фазы отдельных команд. Структуры ВС: ОКОД, МКМД;
- организация функционирования ВС. Операционные системы многомашинных ВС. Программное обеспечение многопроцессорных ВС.

Для построения ВС необходимо, чтобы элементы или модули, комплексируемые в систему, были совместимы. Понятие совместимости имеет три аспекта:

1. Аппаратный, или технический;
2. Программный;
3. Информационный.

Техническая (HardWare) совместимость предполагает, что еще в процессе разработки аппаратуры обеспечиваются следующие условия:

- подключаемая друг к другу аппаратура должна иметь единые стандартные, унифицированные средства соединения: кабели, число проводов в них, единое назначение пределов, разъемы, заглушки, адаптеры, платы и т.д.;
- параметры электрических сигналов, которыми обмениваются технические устройства, тоже должны соответствовать друг другу: амплитуды импульсов, полярность, длительность и т.д.;
- алгоритмы взаимодействия (последовательности сигналов по отдельным проводам) не должны вступать в противоречие друг с другом.

Последний пункт тесно связан с программной совместимостью. Программная совместимость (SoftWare) требует, чтобы программы, передаваемые из одного технического средства в другое (между ЭВМ, процессорами, между процессорами и внешними устройствами), были правильно поняты и

выполнены другим устройством. Если обменивающиеся устройства идентичны друг другу, то проблем обычно не возникает. Если взаимодействующие устройства относятся к одному и тому же семейству ЭВМ, но стыкуются разные модели (например, ПК на базе i286 и Pentium), то в таких моделях совместимость обеспечивается снизу-вверх, т.е. ранее созданные программы могут выполняться на более поздних моделях, но не наоборот. Если стыкуемая аппаратура имеет совершенно разную систему команд, то следует обмениваться исходными модулями программ с последующей их трансляцией.

Информационная совместимость комплексируемых средств предполагает, что передаваемые информационные массивы будут одинаково интерпретироваться стыкуемыми модулями ВС. Должны быть стандартизованы алфавиты, разрядность, форматы, структура и разметка файлов, томов.

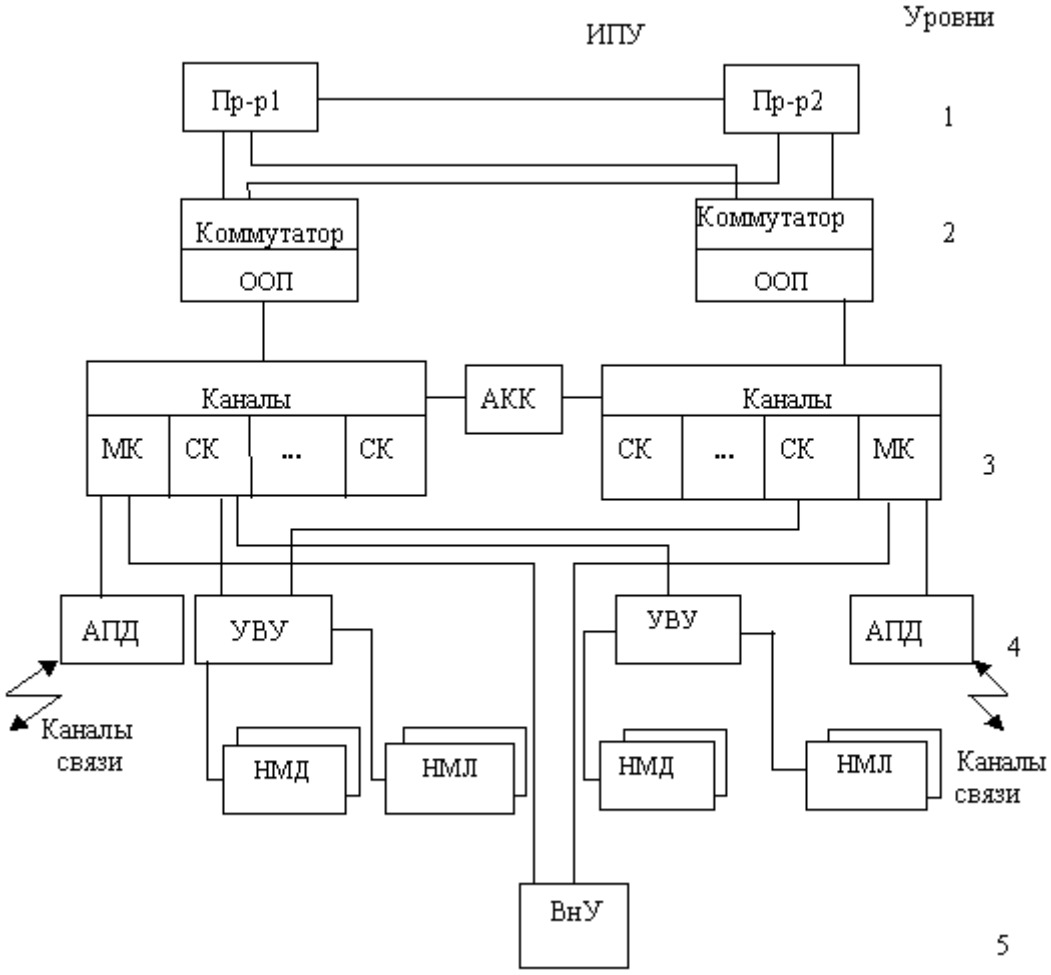
В создаваемых ВС стараются обеспечить несколько путей передачи данных, что позволяет достичь необходимой надежности функционирования, гибкости и адаптируемости к конкретным условиям работы. Эффективность обмена информацией определятся скоростью передачи и возможными объемами данных, передаваемыми по каналу взаимодействия. Эти характеристики зависят от средств, обеспечивающих взаимодействие модулей и уровня управления процессами, на котором это взаимодействие осуществляется. Сочетание различных уровней и методов обмена данными между модулями ВС наиболее полно представлено в универсальных суперЭВМ и больших ЭВМ, в которых сбалансировано использовались все методы достижения высокой производительности. В этих машинах предусматривались следующие уровни комплексирования:

1. Прямого управления (процессор - процессор);
2. Общей оперативной памяти;
3. Комплексируемых каналов ввода-вывода;
4. Устройств управления внешними устройствами (УВУ);
5. Общих внешних устройств.

На каждом из этих уровней используются специальные технические и программные средства, обеспечивающие обмен информацией.

Уровень прямого управления служит для передачи коротких однобайтовых приказов-сообщений. Последовательность взаимодействия процессоров сводится к следующему. Процессор-инициатор обмена по интерфейсу прямого управления (ИПУ) передает в блок прямого управления байт - сообщение и подает команду "прямая запись". У другого процессора эта команда вызывает прерывание, относящееся к классу внешних.

Рисунок 2.3.1. Уровни и средства комплексирования



В ответ он вырабатывает команду "прямое чтение" и записывает передаваемый байт в свою память. Затем принятая информация расшифровывается и по не принимается решение. После завершения передачи прерывания снимаются, и оба процессора продолжают вычисления по собственным программам. Видно, что уровень прямого управления не может использоваться для передачи больших массивов данных, однако оперативное взаимодействие отдельными сигналами широко используются в управлении вычислениями. У ПЭВМ типа IBM PC этому уровню соответствует комплексирование процессоров, подключаемых к системной шине.

Уровень общей оперативной памяти (ООП) является наиболее предпочтительным для оперативного взаимодействия процессоров. В этом случае ООП эффективно работает при небольшом числе обслуживаемых абонентов.

Уровень комплексируемых каналов ввода-вывода предназначается для передачи больших объемов информации между блоками оперативной памяти, сопрягаемых в ВС. Обмен данными между ЭВМ осуществляется с помощью адаптера "канал-канал" (АКК) и команд "чтение" и "запись".

Адаптер - это устройство, согласующее скорости работы сопрягаемых каналов. Обычно сопрягаются селекторные каналы (СК) машин как наиболее быстродействующие. Скорость обмена данными определяется скоростью самого медленного канала. Скорость передачи данных по этому уровню составляет несколько Мбайт в секунду. В ПЭВМ данному уровню взаимодействия соответствует подключение периферийной аппаратуры через контроллеры и адаптеры.

Уровень устройства управления внешними устройствами (УВУ) предполагает использование встроенного в УВУ двухканального переключателя и команд "зарезервировать" и "освободить". Двухканальный переключатель позволяет подключать УВУ одной машины к селекторным каналам различных ЭВМ. По команде "зарезервировать" канал - инициатор обмена имеет доступ через УВУ к любым накопителям на дисках НМД или на магнитных лентах НМЛ. На рис. 2.3.1 схематически показано, что они управляются одним УВУ. На самом деле УВУ магнитных дисков и лент - совершенно различные устройства. Обмен канала с накопителями продолжается до полного завершения работ и получения команды "освободить". Только после этого УВУ может подключиться к конкурирующему каналу. Только такая дисциплина обслуживания требований позволяет избежать конфликтных ситуаций.

На четвертом уровне с помощью аппаратуры передачи данных (АПД) (мультиплексоры, сетевые адаптеры, модемы и др.) имеется возможность сопряжения с каналами связи. Эта аппаратура позволяет создавать сети ЭВМ.

Пятый уровень предполагает использование **общих внешних устройств.** Для подключения отдельных устройств используется автономный двухканальный переключатель.

Пять уровней комплексирования получили название **логических** потому, что они объединяют на каждом уровне разнотипную аппаратуру, имеющую сходные методы управления. Каждое из устройств может иметь логическое имя, используемое в прикладных программах. Этим достигается независимость программ пользователей от конкретной физической конфигурации системы. Связь логической структуры программы и конкретной физической структуры ВС обеспечивается операционной системой по указаниям - директивам пользователя, при генерации ОС и по указаниям диспетчера - оператора вычислительного центра. Различные уровни комплексирования позволяют создавать самые различные структуры ВС.

Второй логический уровень позволяет создавать многопроцессорные ВС. Обычно он дополняется и первым уровнем, что позволяет повышать оперативность взаимодействия процессоров. Вычислительные системы

сверхвысокой производительности должны строиться как многопроцессорные. Центральным блоком такой системы является быстродействующий коммутатор, обеспечивающий необходимые подключения абонентов (процессоров и каналов) к общей оперативной памяти.

Уровни 1, 3, 4, 5 обеспечивают построение разнообразных машинных комплексов. Особенно часто используется третий в комбинации с четвертым. Целесообразно их дополнять и первым уровнем.

Пятый уровень комплексирования используется в редких специальных случаях, когда в качестве внешнего объекта используется какое-то дорогое уникальное устройство. В противном случае этот уровень малоэффективен. Любое внешнее устройство - это недостаточно надежное устройство точной механики, а значит, выгоднее использовать четвертый уровень комплексирования, когда можно сразу управлять не одним, а несколькими внешними устройствами, включая и резервные.

Сочетание уровней и методов взаимодействия позволяет создавать самые различные многомашинные и многопроцессорные системы.

Типовые структуры вычислительных систем

С момента появления первых систем было опробовано большое количество разнообразных структур систем, отличающихся друг от друга различными техническими решениями. Практика показала, что каждая структура вычислительной системы эффективно обрабатывает лишь задачи определенного класса. При этом необходимо, чтобы структура вычислительной системы максимально соответствовала структуре решаемых задач. Только в этом случае система обеспечивает максимальную производительность.

Универсальной структуры вычислительной системы, одинаково хорошо обрабатывающей задачи любого типа, не существует.

Классификация уровней программного параллелизма включает в себя шесть позиций:

1. Независимые задания;
2. Отдельные части заданий;
3. Программы и подпрограммы;
4. Циклы и итерации;
5. Операторы и команды;
6. Фазы отдельных команд.

Для каждого из них имеются специфические свойства параллельной обработки, опробированные в различных структурах вычислительных систем. Данный перечень совершенно не затрагивает этапы выбора алгоритмов решения, на которых могут анализироваться альтернативные алгоритмы (а значит, и программы), дающие различные результаты.

Для каждого вида параллельных работ имеются структуры вычислительных средств, используемые в различных вычислительных системах. Верхние три уровня, включающие независимые задания, шаги или части заданий и отдельные программы, имеют единое средство параллельной обработки - мультипроцессирование, т.е. многопроцессорные вычислительные системы, относящиеся к архитектуре МКМД. Программные циклы и итерации требуют использования векторной обработки (архитектура ОКМД). Операторы и команды, выполняемые ЭВМ, ориентированы на многофункциональную обработку. Параллельная обработка фаз последовательно выполняемых команд приводит к организации конвейера команд.

Рассмотрим возможные структуры вычислительных систем, которые обеспечивают перечисленные виды программного параллелизма.

ОКОД - структуры

Данный тип архитектуры объединяет любые системы в однопроцессорном (одномашинном) варианте.

За пятьдесят лет развития электронной вычислительной техники классическая структура ЭВМ претерпела значительные усовершенствования. Однако основной принцип программного управления не был нарушен. Данная структура оказалась сосредоточенной вокруг оперативной памяти, так как именно цепь "процессор - оперативная память" во многом определяет эффективную работу компьютера. При выполнении каждой команды необходимо неоднократное обращение к оперативной памяти: выбор команды, операндов, отсылка результатов и т.д.

Можно перечислить большое число приведенных улучшений классической структуры ЭВМ, ставших в настоящее время определенными стандартами при построении новых ЭВМ: иерархическое построение памяти ЭВМ, появление адресации памяти, разделение процессов ввода-вывода и обработки задач, появление систем прерывания и приоритетов и т.д.

В этом ряду следует рассматривать и организацию конвейера последовательно выполняемых команд: формирование адреса команды, выбор команды, формирование адресов и выбор операндов, выполнение команды, запись результата. Однако примитивная организация памяти (память линейна и одномерна) не позволяет организовать длинный и

эффективный конвейер. Линейные участки современных программ редко превышают десяток, полтора последовательно выполняемых команд. Поэтому конвейер часто перезапускается, что снижает производительность ЭВМ в целом. Многофункциональная обработка также нашла свое место при построении ЭВМ. Например, даже в персональных ЭВМ, построенных на микропроцессорах i486 и Pentium, в состав ЭВМ могут включаться и другие специализированные средства обработки: умножители, делители, сопроцессоры или блоки десятичной арифметики, сопроцессоры обработки графической информации и другие. Все они совместно с центральным процессором ЭВМ позволяют создавать своеобразные микроконвейеры, целью которых является повышение скорости вычислений.

В последние годы широко используется еще несколько модификаций классической структуры. В связи с успехами микроэлектроники появилась возможность построения RISC - компьютеров (Reduced Instruction Set Computing), т.е. ЭВМ с сокращенным набором команд.

Большие ЭВМ предыдущих поколений не имели большой сверхоперативной памяти, поэтому они имели достаточно сложную систему команд CISC (Complete Instruction Set Computing - вычисления с полной системой команд). В этих машинах большую долю команд составили команды типа "память-память", в которых операнды и результаты операций находились в оперативной памяти. Время обращения к памяти и время вычислений соотносились примерно 5:1. В RISC - машинах с большой сверхоперативной памятью большой удельный вес составляют операции "регистр-регистр" и отношение времени обращения к памяти по времени вычислений составляет 2:1.

Поэтому в RISC - ЭВМ основу системы команд составляют наиболее употребительные, "короткие" операции типа алгебраического сложения. Сложные операции выполняются как подпрограммы, состоящие из простых операций. Это позволяет значительно упростить внутреннюю структуру процессора, уменьшить фазы дробления конвейерной обработки и увеличить частоту работы конвейера. Но здесь необходимо отметить, что за эффект приходится расплачиваться усложнением процедур обмена данными между регистрами сверхоперативной памяти и кэш-памяти с оперативной памятью.

Другой модификацией классической структуры ЭВМ является VLIW (Very Large Instruction Word) - ЭВМ с "очень длинным командным словом". ЭВМ этого типа выбирает из памяти суперкоманду, включающую несколько команд. Здесь возможны варианты. В самом простом случае это приводит к появлению буфера команд (кэш-команд) с целью ускорения конвейера операций. В более сложных случаях в состав суперкоманд стараются включать параллельные команды, не связанные общими данными. Если процессор ЭВМ при этом построен из функционально независимых

устройств (устройства алгебраического сложения, умножения, сопроцессоры), то в этом случае обеспечивается максимальный эффект работы ЭВМ. Но это направление связано с кардинальной перестройкой процессоров трансляции и исполнения программ. Здесь значительно усложняются средства автоматизации программирования.

VLIW - компьютеры могут выполнять суперскалярную обработку, т.е. одновременно выполнять две или более команд. В целом ряде структур суперЭВМ использовалась эта идея. В ПЭВМ последних выпусков имеется возможность выполнения двух команд одновременно. Эта реализация имеет две цели:

1. Уменьшение отрицательного влияния команд ветвления вычислительного процесса путем выполнения независимых команд двух различных ветвей программы. При этом в какой-то степени исключаются срывы конвейера в обработке команд программы;
2. Одновременное выполнение двух команд (независимых по данным и регистрам их хранения), например команды пересылки и арифметические операции.

Для реализации программного параллелизма, включающего циклы и итерации, используются матричные или векторные структуры. В них эффективно решаются задачи матричного исчисления, задачи решения систем алгебраических и дифференциальных уравнений, задачи теории поля, геодезические задачи, задачи аэродинамики. Теоретические проработки подобных структур относятся к концу пятидесятых, шестидесятым годам. Данные структуры очень хорошо зарекомендовали себя при решении перечисленных задач, но они получились очень дорогими по стоимости и эксплуатации. Кроме того, в тех случаях, когда структура параллелизма отличалась от матричной, возникает необходимость передачи данных между процессорами через коммутаторы. При этом эффективность вычислений резко снижается. Подобные структуры могут использоваться как сопроцессоры в системах будущих поколений.

МКОД - структуры

МКОД - структуры большой практической реализации не получили. Задачи, в которых несколько процессоров могли эффективно обрабатывать один поток данных, в науке и технике неизвестны. С некоторой натяжкой к этому классу можно отнести фрагменты многофункциональной обработки, например обработку на разных процессорах команд с фиксированной или плавающей точкой.

Так же как фрагмент такой структуры, можно рассматривать локальную сеть персональных компьютеров, работающих с единой базой данных, но скорее всего это - частный случай использования МКМД - структуры.

МКМД - структуры

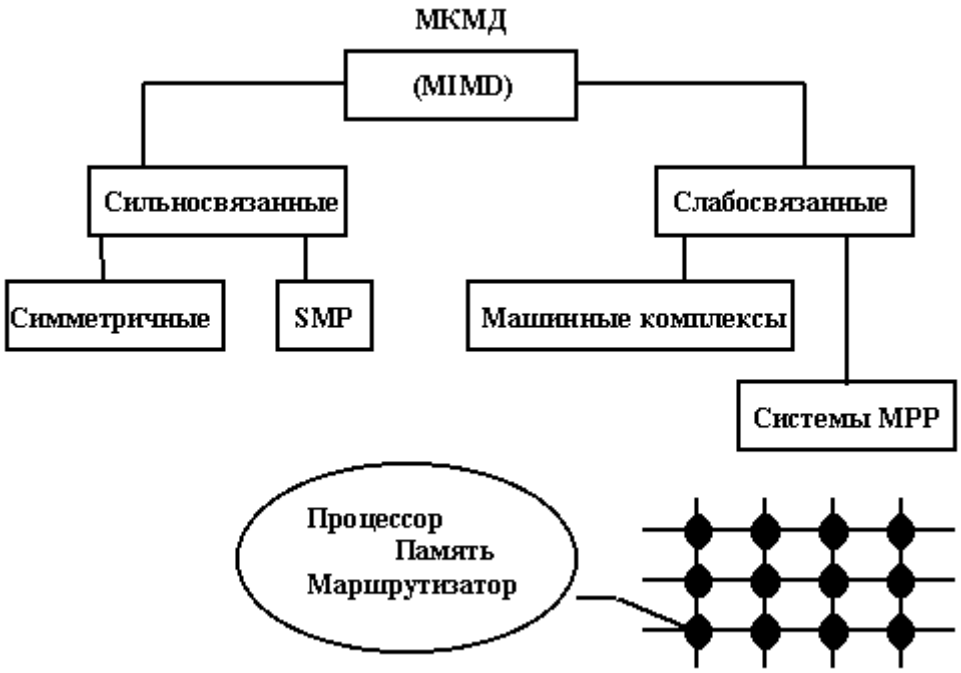
МКМД - структуры являются наиболее интересным классом структур вычислительных систем. После разочарований в структурах суперЭВМ, основанных на различном сочетании векторной и конвейерной обработки, усилия теоретиков и практиков обращены в этом направлении.

Уже из названия МКМД - структур видно, что в данных системах можно найти все перечисленные виды параллелизма. Этот класс дает большое разнообразие структур, сильно отличающихся друг от друга своими характеристиками (рис. 2.3.2).

Важную роль здесь играют способы взаимодействия ЭВМ или процессоров в системе. В сильносвязанных системах достигается высокая оперативность взаимодействия процессоров посредством общей оперативной памяти. При этом пользователь имеет дело с многопроцессорными вычислительными системами. Наиболее простыми по строению и организации функционирования являются однородные симметричные структуры. Они обеспечивают простоту подключения процессоров и не требуют очень сложных централизованных операционных систем, размещаемых на одном из процессоров.

Однако при построении таких систем возникает много проблем с использованием общей оперативной памяти. Число комплексируемых процессоров не может быть велико, оно не превышает 16. Для уменьшения числа обращений к памяти и конфликтных ситуаций могут использоваться многоблочное построение ОП, функциональное закрепление отдельных блоков за процессорами, снабжение комплексируемых процессоров собственной памятью типа "кэш". Но все эти методы не решают проблемы повышения производительности ВС в целом. Аппаратные затраты при этом существенно возрастают, а производительность систем увеличивается незначительно.

Рисунок 2.3.2. Типовые структуры ВС в МКМД (MIMD) - классе



Появление мощных микропроцессоров типа "Pentium" привело к экспериментам по созданию многопроцессорных систем на их основе. Так, для включения мощных серверов в локальные сети персональных компьютеров была предложена несколько измененная структура использования ООП - SMP (Share Memory multiProcessing - мультипроцессирование с разделением памяти). На общей шине оперативной памяти можно комплексировать до четырех микропроцессоров.

Слабосвязанные МКМД - системы могут строиться как многомашинные комплексы или использовать в качестве средств передачи информации общее поле внешней памяти на дисковых накопителях большой емкости.

Невысокая оперативность взаимодействия заранее предопределяет ситуации, в которых число межпроцессорных конфликтов при обращении к общим данным и к друг другу было бы минимальным. Для этого необходимо, чтобы ЭВМ комплекса обменивались друг с другом с небольшой частотой, обеспечивая автономность процессов (программы и данные к ним) и параллелизм из выполнения. Только в этом случае обеспечивается надлежащий эффект. Эти проблемы решаются в сетях ЭВМ.

Успехи микроинтегральной техники и появление БИС и СБИС позволяют расширить границы и этого направления. Возможное построение систем с десятками, сотнями и даже тысячами процессорных элементов, с размещением их в непосредственной близости друг от друга. Если каждый процессор системы имеет собственную память, то он также будет сохранять известную автономию в вычислениях. Подобные ВС получили название **систем с массовым параллелизмом** (MPP - Mass-Parallel Processing). Все

процессорные элементы в таких системах должны быть связаны единой коммутационной средой. Здесь возникают проблемы, аналогичные ОКМД - системам, но на новой технологической основе.

Передача данных в MPP - системах предполагает обмен не отдельными данными под централизованным управлением, а подготовленными процессами (программами вместе с данными). Этот принцип построения вычислений уже не соответствует принципам программного управления классической ЭВМ. Передача данных процесса по его готовности больше соответствует принципам построения "потоковых машин" (машин, управляемых потоками данных). Подобный подход позволяет строить системы с громадной производительностью и реализовывать проекты с любыми видами параллелизма, например перейти к "систолическим вычислениям" с произвольным параллелизмом. Однако для этого необходимо решить целый ряд проблем, связанных с описанием, программированием коммутаций процессов и управления ими. Математическая база этой науки в настоящее время практически отсутствует.

Организация функционирования вычислительных систем

Управления вычислительными процессами в ВС осуществляют операционные системы, которые являются частью общего программного обеспечения. В состав ОС включают как программы централизованного управления ресурсами системы, так и программы автономного использования вычислительных модулей. Последнее условие необходимо, так как в ВС обычно предусматривается более высокая надежность функционирования, например требование сохранения работоспособности при наличии в ней хотя бы одного исправного модуля. Требование увеличения производительности также предполагает возможность параллельной и даже автономной работы модулей при обработке отдельных заданий или пакетов заданий.

В зависимости от структурной организации ВС можно выявить некоторые особенности построения их операционных систем.

Операционные системы многомашинных ВС являются более простыми. Обычно они создаются как надстройка автономных ОС отдельных ЭВМ, так как здесь каждая ЭВМ имеет большую автономию в использовании ресурсов (своя оперативная и внешняя память, свой обособленный состав внешних устройств и т.д.). В них широко используются программные методы локального (в пределах вычислительного центра) и дистанционного (сетевая обработка) комплексирования.

Общим для построения ОС многомашинных комплексов служит тот факт, что для каждой машины ВС другие играют роль некоторых внешних устройств, и их взаимодействие осуществляется по интерфейсам, имеющим

унифицированное программное обеспечение. Все обмены данными между ЭВМ должны предусматриваться пользователями путем включения в программы специальных операторов распараллеливания вычислений. По этим обращениям ОС ВС включает особые программы управления обменом. При этом ОС должна обеспечивать распределение и последующую пересылку заданий или их частей, оформляя их в виде самостоятельных заданий. Такие ОС, организуя обмен, должны формировать и устанавливать связи, контролировать процессы обмена, строить очереди запросов, решать конфликтные ситуации.

В многомашинных ВС диспетчерские функции могут решаться на централизованной или децентрализованной основе. Связь машин обычно устанавливается в порядке подчиненности: "главная ЭВМ - вспомогательная ЭВМ". Например, в пакете Norton Commander имеется возможность установить подобную связь: "Master" - "Slave".

Программное обеспечение многопроцессорных ВС отличается большей сложностью. Это объясняется глубиной и сложностью всестороннего анализа процессоров, формируемых в ВС, а также сложностью принятия решения в каждой конкретной ситуации. Здесь все операции планирования и диспетчеризации связаны с динамическим распределением ресурсов (оперативной и внешней памяти, процессоров, данных системных таблиц, программ, периферийного оборудования и т.п.). Центральное место в этом играют степень использования и методы управления общей оперативной памятью. Здесь очень часто могут формироваться множественные конфликты, требующие сложных процедур решения, что приводит к задержкам в вычислениях. Как таковые автономные ОС отдельных процессоров отсутствуют.

Для обеспечения эффективной работы многопроцессорных систем их операционные системы специализируют по следующим типовым методам взаимодействия процессоров:

- "ведущий - ведомый";
- симметричная или однородная обработка во всех процессорах;
- раздельная независимая работы процессоров по обработке заданий.

Выбор метода "ведущий - ведомый" в наибольшей степени соответствует ВС с централизованным управлением. Тут имеется определенная аналогия с многомашинными системами, организованными по принципу "главная ЭВМ - вспомогательная ЭВМ".

Диспетчерские функции выполняются только одним процессором системы. Закрепление этих функций может быть фиксированным и плавающим. Для этого может выделяться специализированный процессор или обычный

процессор универсального типа, переключающийся на выполнение вычислений.

Системы типа "ведущий - ведомый" отличаются довольно простым аппаратным и программным обеспечением. Они должны получить распространение в МРР - структурах, но следует иметь в виду, что длительное время планирования может быть причиной простоев ведомых вычислителей.

Симметричная или однородная обработка в матрице процессоров возможна при использовании однотипных процессорных элементов, каждый из которых имеет непосредственные связи по передаче данных с другими. В отличие от ОКМД - структур ранних выпусков, в которых синхронизировалось выполнение отдельных команд, в МРР - структурах симметричная обработка должна обеспечивать синхронизацию выполнения целых процессов. Ни один из существующих языков программирования не содержит эффективных средств управления параллельными вычислениями. Такая система имеет большие достоинства. Она обладает существенно более высокой живучестью и сохраняет работоспособность при выходе из строя даже нескольких процессоров матрицы, так как здесь имеется более высокий уровень резервирования. В ней обеспечивается более полная загрузка процессоров с лучшим использованием их процессорного времени. Расход других общесистемных ресурсов также эффективнее.

В связи с успехами микроэлектроники появилась возможность реализовывать эти структуры в виде сверхбольших интегральных схем (СБИС), что позволяет получить дополнительные преимущества:

- короткие соединительные линии между процессорными элементами. Это приводит к расширению полосы пропускания и уменьшению задержек;
- регулярность структуры, позволяющая увеличить плотность упаковки СБИС и упрощать ее разработку;
- высокую степень распараллеливания вычислений, что позволяет обеспечить высокую производительность.

Для управления процессом вычислений из однородной среды процессорных элементов выделяется один, играющий роль ведущего. Эти функции при необходимости могут передаваться от одного процессора к другому.

Раздельная или независимая работы вычислителей в многопроцессорных ВС осуществляется при параллельной обработке независимых заданий. Это позволяет получить максимальную производительность системы. Процедуры управления ею достаточно просты и уже опробированы в практических вариантах.

Структурная схема отечественной суперЭВМ

Общие сведения

Супер-ЭВМ занимают самую правую позицию в спектре современной ЦВТ и обладают производительностью, достигающей 1011 оп/с и выше. Такие ВС могут не только удовлетворительно решать сложнейшие научно-технические задачи, но и обеспечивать работу более чем с 10000 отдельных рабочих станций, для чего им требуется в качестве **координатора системы** ввода/вывода специальные мини- или ЭВМ общего назначения. Типичными областями применения супер-ЭВМ являются: научные исследования, прогнозирование погоды, проектирование авиационной и космической техники и другие области, требующие быстрой обработки очень большого количества данных.

Оценки возможной пиковой производительности одного процессора на полупроводниковой элементной базе только из-за задержек в связях между логическими элементами не превышает 108 оп/с для скалярных вычислений. Следовательно, получение большей производительности может быть получено только в рамках не - неймановской параллельной архитектуры ЦВТ. Использование последовательно - параллельной обработки одним процессором принципиально не меняет сути дела, поэтому вполне естественным является мультипроцессорная или многомашинная организация ВС. Аппаратно современный параллелизм ЭВМ поддерживается на четырех основных уровнях:

- многомашинном;
- мультипроцессорном;
- однопроцессорном с несколькими исполнительными устройствами;
- конвейеризацией обработки данных.

Используются и смешанные архитектурные решения.

Для увеличения производительности скалярных вычислений в состав ЦП включаются векторные и скалярные исполнительные устройства с быстрыми регистрами и индексной арифметикой, т.е. объединяются два спецпроцессора для обработки одной последовательности команд.

Все современные параллельные ВС (ПВС) являются мультипроцессорными с различной архитектурой, наиболее распространенные из которых можно классифицировать по способу использования ОП и режиму выполнения команд процессорами системы. Относительно использования ОП ПВС можно классифицировать по двум основным группам:

- ОП распределяется по процессорам;
- процессоры разделяют общую ОП.

В рамках другого подхода ПВС можно классифицировать также по двум группам:

- SIMD (Single Instruction - Multiple Data);
- MIMD (Multiple Instruction - Multiple Data).

В первом случае все процессоры ПВС обрабатывают единый поток команд, во втором - многие потоки команд (или программ). Системы SIMD не требуют синхронизации вычислений, тогда как MIMD- системы моделируют SIMD - системы, но не наоборот.

Одной из причин, сдерживающих степень распараллеливания вычислений в мультипроцессорных ВС, является существенная сложность аппаратной реализации системы коммутации процессоров с общей ОП; система коммутации не должна существенно понижать производительность каждого процессора. Примером организации мультипроцессорной супер-ЭВМ может служить отечественная Эльбрус-3, в архитектуре которой использован целый ряд интерфейсных решений.

Рисунок 2.3.3.

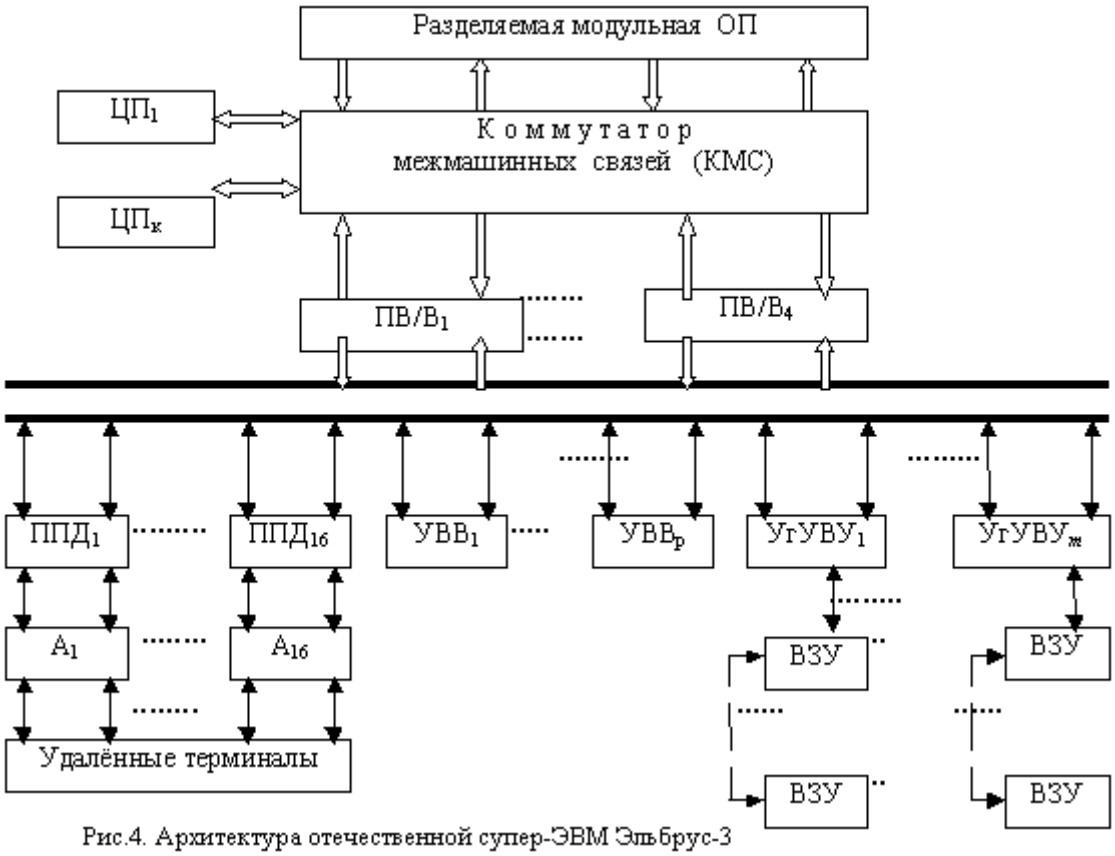


Рис.4. Архитектура отечественной супер-ЭВМ Эльбрус-3

Центральным узлом управления при такой организации является быстродействующий коммутатор межмодульных связей (КМС), где в

качестве модулей выступают процессоры (ЦП), модули памяти (ОП) и процессоры ввода/вывода (ПВ/В). КМС обеспечивает возможность установления связи любого ЦП с любым модулем ОП; при этом, возможны одновременная связь и передача данных через КМС между различными парами модулей ВС. Следовательно, принцип КМС реализует пространственное разделение при коммутации соединений в отличие от временного разделения в случае использования общей шины, а также существенно уменьшает число конфликтных ситуаций в ВС. Однако с целью получения высокой общей производительности ВС требуется быстрая реактивность переключений КМС, что является весьма трудной технической задачей ввиду сложной логики КМС. Используется несколько типов КМС:

- матричный;
- мультиплексная шина;
- многоуровневые коммутационные сети.

Супер-ЭВМ Эльбрус-3 содержит до 16 ЦП, исполнительные устройства (сложения, умножения, деления, логической обработки, десятичных преобразований, обработки строк, вызова и записи операндов, программ и индексации), которые работают в максимально доступном режиме распараллеливания. Каждый из процессоров супер-ЭВМ обеспечивает: эффективную трансляцию и выполнение программ за счет аппаратной реализации наиболее массовых алгоритмов языков высокого уровня и операционной системы; мультипрограммный режим с аппаратной защитой данных пользователя и операционной системы; а высокую производительность порядка 12 млн оп/с обеспечивают современная интегральная технология, высокий уровень параллельной организации и совмещения выполнения команд. ЦП аппаратно реализует стековую сверхоперативную память, поточную обработку команд и безадресную систему команд, базирующуюся на польской инверсной записи; управляющая информация имеет формат слова (64 бита), числовые данные имеют длину в полслова (32 бита), слово (64 бита) и двойное слово (128 бит); каждое слово снабжено 6-битным тегом, определяющим его тип; допускается работа с нечисловой информацией форматов битового, цифрового или байтового наборов.

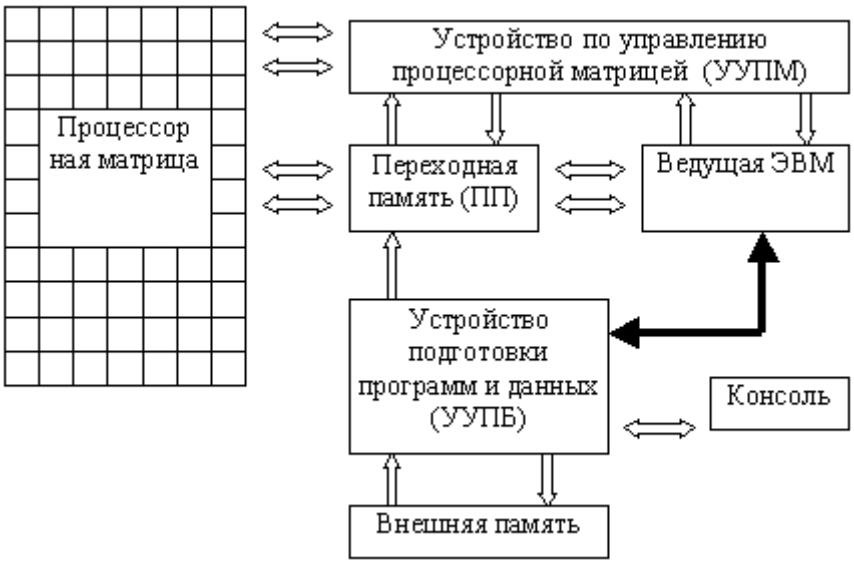
Структурная схема параллельной суперЭВМ МРР

Общие сведения

Весьма интересным подходом к созданию супер-ЭВМ высокой производительности является использование параллельных архитектур, базирующихся на большом числе относительно медленных, но простых микропроцессоров. Такие архитектуры используют не - неймановскую параллельную ОС - модель вычислителя, требуют нового подхода к

организации вычислительного процесса алгоритмизации задач и нового высокопараллельного ПО как системного, так и инструментального. Типичным примешом такой архитектуры может служить супер-ЭВМ MPP фирмы Goodyear Aerospace.

Рисунок 2.3.4. Архитектура параллельной супер-ЭВМ MPP



Параллельную архитектуру MPP - компьютера определяют следующие компоненты:

- процессорная матрица (ПМ) для обработки изображений;
- устройство управления процессорной матрицей (УУПМ);
- устройство подготовки программ и данных (УППД);
- переходная память (ПП) для буферизации и перекомпоновки данных;
- ведущая и интерфейсная ЭВМ, внешняя память (ВП) и система интерфейсов.

ПМ представляет собой матрицу (128*128)-размерности процессорных элементов (ПЭ), оперирующую с двумерными массивами данных. Каждый ПЭ имеет собственную локальную память произвольного доступа объемом 128 байт (расширяется до 8 Кбайт) и связан со своими соседями согласно индексу соседства Неймана; возможно использование и других типов коммутации. Каждый ПЭ имеет непосредственный доступ только данным из своей локальной памяти; доступ к другим данным возможен только через межпроцессорную связь или через ПП.

ПЭ представляет собой специальный МП с тактовой частотой 10 МГц; в каждом такте все ПЭ матрицы выполняют одну и ту же операцию - каждый над своим потоком данных. Операции, соответствующие системе команд отдельного ПЭ, представляют собой элементарные операции, образующие набор команд ПМ. Управляющие сигналы поступают из устройства

управления ПМ (УУПМ), а микрокоманды считываются из специальной памяти, входящей в состав УУПМ. ПЭ выполняют операции: логико-арифметические, маршрутизации, выборки данных из интерфейсной шины, ввода/вывода и некоторые другие. УУПМ управляет работой ПМ и содержит три компоненты:

- устройство управления ПЭ;
- устройство управления вводом/выводом;
- центральное устройство, обеспечивающее выбор внешней среды, с которой должен работать МРР.

Программное обеспечение находится в ведущей ЭВМ и частично в УППД. В некоторых специализированных ЭВМ можно отказаться от сложной системы коммутации ПЭ; массивы ПЭ с непосредственной информационной связью между своими ближайшими соседями называются **систолическими**.

Основные вычислительные возможности МРР обеспечиваются ПМ, работающей под управлением УУПМ; обмен данными с которой производится через ПП. Пользователь работает с МРР через консоль, подключенную к УППД. К УППД подключается внешняя память и принтер. МРР допускает расширение конфигурации за счет дополнительных ПМ, УУПМ и ПП СПО супер-ЭВМ МРР включает: ассемблеры, системные подпрограммы, макросы подпрограмм и ввода/вывода, программу управления ПП, драйверы внешних устройств, CAD- отладчик и параллельный Pascal, позволяющий непосредственно использовать матричную архитектуру МРР посредством удобных языковых средств. Язык дает возможность работать с объектами, в качестве которых выступают параллельные массивы. Отладчик CAD обеспечивает основной интерфейс пользователя с МРР, осуществляя отладку и исполнение прикладного ПО; он обеспечивает загрузку и запуск на выполнение программ и содержит широкий набор команд для их отладки.

Контрольные вопросы

1. Каковы основные предпосылки появления и развития ВС?
2. По каким признакам классифицируются вычислительные системы?
3. Каковы принципиальные различия между многомашинными и многопроцессорными ВС?
4. Какие принципы положены в основу классификации архитектур ВС?
5. Раскройте содержание понятия совместимости в ВС.
6. С какой целью используется несколько уровней комплексирования в ВС?
7. Какие преимущества обеспечивают системы массового параллелизма МРР перед другими типами ВС?
8. Какие типы ВС могут создаваться на базе ПЭВМ?

9. Каковы принципы организации вычислительного процесса в ВС?

Раздел III. Телекоммуникационные вычислительные сети [16]

Тема 3.1. Принципы построения ТВС

Цель изучения темы: усвоить основные понятия: ТВС, абонентская система (АС), коммуникационная подсеть, или телекоммуникационная система (ТКС), прикладной процесс; изучить основные типы, режимы работы ТВС.

Вопросы

1. Общие сведения.
2. Классификация сетей.
3. Управление взаимодействием прикладных процессов

3.1.1. Общие сведения

Телекоммуникационная вычислительная сеть (ТВС) - это сеть обмена и распределенной обработки информации, образуемая множеством взаимосвязанных абонентских систем и средствами связи.

Средства передачи и обработки информации в ТВС ориентированы на коллективное использование общесетевых ресурсов - аппаратных, информационных, программных.

Абонентская система (АС) - это совокупность ЭВМ, программного обеспечения, периферийного оборудования, средств связи с коммуникационной подсетью вычислительной сети, выполняющих прикладные процессы.

Коммуникационная подсеть, или телекоммуникационная система (ТКС), - это совокупность физической среды передачи информации, аппаратных и программных средств, обеспечивающих взаимодействие АС.

Прикладной процесс - это различные процедуры ввода, хранения, обработки и выдачи информации, выполняемые в интересах пользователей и описываемые прикладными программами.

Основные задачи ТВС:

1. Обеспечение неограниченного доступа к ЭВМ пользователей независимо от их территориального расположения.
2. Возможность оперативного перемещения больших массивов информации на любые расстояния.

Для ТВС принципиальное значение имеют следующие обстоятельства:

- ЭВМ, находящиеся в составе разных абонентских систем одной и той же сети или различных взаимодействующих сетей, связываются между собой автоматически;
- каждая ЭВМ сети должна быть приспособлена как для работы в автономном режиме под управлением своей операционной системы (ОС), так и для работы в качестве состава звена сети.

Режимы работы ТВС:

- обмен данными между АС;
- запрос и выдачи информации;
- сбор информации;
- пакетная обработка данных по запросам пользователей с удаленных терминалов;
- работа в диалоговых режимах.

По сравнению с адекватной по вычислительной мощности совокупностью автономно работающих ЭВМ сеть имеет ряд преимуществ:

- обеспечение распределенной обработки данных и параллельной обработки многими ЭВМ;
- возможность создания распределенной базы данных (РБД), размещаемой в памяти различных ЭВМ;
- возможность обмена большими массивами информации между ЭВМ, удаленными друг от друга на значительном расстоянии;
- коллективное использование дорогостоящих ресурсов;
- предоставление таких услуг, как электронная почта (ЭП), телеконференции, электронные доски объявлений (ЭДО), дистанционное обучение;
- повышение эффективности использования средств вычислительной техники и информатики (СВТИ) за счет более интенсивной и равномерной их загрузки;
- возможность оперативного перераспределения вычислительных мощностей между пользователями сети в зависимости от изменения их потребностей, а также резервирования этих мощностей и средств передачи данных на случай выхода из строя отдельных элементов сети;
- сокращение расходов на приобретение и эксплуатацию СВТИ (за счет коллективного их использования);
- облегчение работ по совершенствованию технических, программных и информационных средств.

Характеризуя возможности той или иной ТВС, следует оценивать ее аппаратное, информационное и программное обеспечение.

- общесетевое ПО, образуемое распределенной операционной системой (РОС) сети и программными средствами, входящими в состав комплекта программ технического обслуживания (КПТО) сети
- специальное ПО, представленное прикладными программными средствами;
- базовое программное обеспечение ЭВМ абонентских систем, включающее операционные системы ЭВМ, системы автоматизации программирования, контролирующие и диагностические тест - программы.

Распределенная операционная система сети управляет работой сети во всех ее режимах, обеспечивает реализацию запросов пользователей, координирует функционирование звеньев сети.

Набор управляющих и обслуживающих программ РОС обеспечивает:

- удовлетворение запросов пользователей по использованию общесетевых ресурсов;
- организацию связи между отдельными прикладными программами комплекса пользовательских программ, реализуемыми в различных АС сети;
- синхронизацию работы пользовательских программ при их одновременном обращении к одному и тому же общесетевому ресурсу;
- удаленный ввод заданий в любой АС сети и их выполнение в любой другой АС сети в пакетном или оперативном режиме;
- обмен файлами между АС сети, доступ к файлам, хранимым в удаленных ЭВМ, и их обработку;
- передачу текстовых сообщений пользователям в порядке реализации функций службы электронной почты, телеконференций, электронных досок объявлений, дистанционного обучения;
- защиту информации и ресурсов сети от несанкционированного доступа;
- выдачу справок, характеризующих состояние и использование аппаратных, информационных и программных ресурсов сети.

3.1.2. Классификация сетей

В основу классификации сетей положены наиболее характерные функциональные, информационные и структурные признаки.

По степени территориальной рассредоточенности элементов сети (абонентских систем, узлов связи) различают глобальные (государственные), региональные и локальные вычислительные сети (ГВС, РВС и ЛВС).

По характеру реализуемых функций сети делятся на вычислительные (основные функции таких сетей - обработка информации), информационные

(для получения справочных данных по запросам пользователей), информационно-вычислительные,

По способу управления ТВС делятся на сети с **централизованным** (в сети имеется один или несколько управляющих органов), **децентрализованным** (каждая АС имеет средства для управления сетью) и **смешанным управлением**, в которых в определенном сочетании реализованы принципы централизованного и децентрализованного управления.

По организации передачи информации сети делятся на сети с селекцией информации и маршрутизацией информации. В сетях с **селекцией информации**, строящихся на основе моноканала, взаимодействие АС производится выбором (селекцией) адресованных им блоков данных (кадров): всем АС сети доступны все передаваемые в сети кадры, но копию кадра снимают только АС, которым они предназначены. В **сетях с маршрутизацией информации для** передачи кадров от отправителя к получателю может использоваться несколько маршрутов. С помощью коммуникационных систем сети решается задача выбора оптимального маршрута.

По типу организации передачи данных сети с маршрутизацией информации делятся на сети с коммутацией цепей (каналов), коммутацией сообщений и коммутацией пакетов. В эксплуатации находятся сети, в которых используются смешанные системы передачи данных.

По топологии, т.е. конфигурации элементов в ТВС, сети делятся на два класса: широковещательные и последовательные. Широковещательные конфигурации и значительная часть последовательных конфигураций (кольцо, звезда с интеллектуальным центром, иерархическая) характерны для ЛВС. Для глобальных и региональных сетей наиболее распространенной является произвольная (ячеистая топология). Нашли применение также иерархическая конфигурация и "звезда".

В **широковещательных конфигурациях** в любой момент времени на передачу кадра может работать только одна рабочая станция (абонентная система) остальные РС сети могут принимать этот кадр, т.е. такие конфигурации характерны для ЛВС с селекцией информации. Основные типы широковещательной конфигурации - общая шина, дерево, звезда с пассивным центром.

В **последовательных конфигурациях**, характерных для сетей с маршрутизацией информации, передача данных осуществляется последовательно от одной РС к соседней.

К последовательным конфигурациям относятся: произвольная (ячеистая), иерархическая, кольцо, цепочка, звезда с интеллектуальным центром,

снежинка. В ЛВС наибольшее распространение получили кольцо и звезда, а также смешанные конфигурации - звездно-кольцевая, звездно-шинная.

Глобальные и региональные, как и локальные, в принципе могут быть однородными (гомогенными), в которых применяются программно-совместимые ЭВМ, и неоднородными (гетерогенными), включающими программно-несовместимые ЭВМ.

3.1.3. Управление взаимодействием прикладных процессов

Практика создания и развития ТВС привела к необходимости разработки стандартов по всему комплексу вопросов организации сетевых систем. В 1978 г. Международная организация по стандартизации (МОС) предложила **семиуровневую эталонную модель взаимодействия открытых систем** (ВОС).

В соответствии с эталонной моделью ВОС абонентская система представляется прикладными процессами и процессами взаимодействия АС. Последние разбиваются на семь функциональных уровней. Функции и процедуры, выполняемые в рамках одного функционального уровня, составляют соответствующий уровневый протокол. Нумерация уровневых протоколов идет снизу вверх. Функциональные уровни взаимодействуют на строго иерархической основе: каждый уровень пользуется услугами нижнего уровня и, в свою очередь, обслуживает уровень, расположенный выше. Создание ТВС в соответствии с эталонной моделью ВОС открывает возможность использования сети ЭВМ различных классов и типов. Поэтому сеть, удовлетворяющая требованиям эталонной модели, называется открытой.

Функциональные уровни рассматриваются как составные независимые части процессов взаимодействия АС. Основные функции, реализуемые в рамках уровневых протоколов, состоят в следующем.

Физический уровень- непосредственно связан с каналом передачи данных, обеспечивает физический путь для электрических сигналов, несущих информацию. На этом уровне осуществляются установление, поддержка и расторжение соединения с физическим каналом, определение электрических и функциональных параметров взаимодействия ЭВМ с коммуникационной подсетью.

Канальный уровень - определяет правила совместного использования физического уровня узлами связи. Главные его функции: управление передачей данных по информационному каналу (генерация стартового сигнала и организация начала передачи информации, передача информации по каналу, проверка получаемой информации и исправление ошибок, отключение канала при его неисправности и восстановление передачи после

ремонта, генерация сигнала окончания передачи и перевода канала в пассивное состояние) и управление доступом к передающей среде.

Сетевой уровень - реализует функции буферизации и маршрутизации, т.е. прокладывает путь между отправителем информации и адресатом через всю сеть. Основная задача сетевого протокола - прокладка в каждом физическом канале совокупности логических каналов. Два пользователя, соединенные логическим каналом, работают так, как будто только в их распоряжении имеется физический канал.

Транспортный уровень обеспечивает связь между коммуникационной подсетью и верхними тремя уровнями, отделяет пользователя от физических и функциональных аспектов сети. Главная его задача - управление трафиком (данными пользователя) в сети. При этом выполняются такие функции, как деление длинных сообщений, поступающих от верхних уровней, на пакеты данных (при передачи информации) и формирование первоначальных сообщений из набора пакетов, полученных через канальный и сетевой уровни, исключая их потери или смещение (при приеме информации). Транспортный уровень есть граница, ниже которой пакет данных является единицей информации, управляемой сетью. Выше этой границы в качестве единицы информации рассматриваются только сообщения. Транспортный уровень обеспечивает также сквозную отчетность в сети.

Сеансовый уровень - предназначен для организации и управления сеансами взаимодействия прикладных процессов пользователей (сеанс создается по запросу процесса пользователя, переданному через прикладной и представительный уровни). Основные функции: управление очередностью передачи данных и их приоритетом, синхронизация отдельных событий, выбор формы диалога пользователей (полудуплексная, дуплексная передача).

Представительный уровень (уровень представления данных) - преобразует информацию к виду, который требуют прикладные процессы пользователей, представительный уровень занимается синтаксисом данных.

Прикладной уровень является границей между процессами сети и прикладными (пользовательскими) процессами. На этом уровне выполняются вычислительные, информационно - поисковые и справочные работы, осуществляется логическое преобразование данных пользователя.

3.1.4. Протоколы передачи данных нижнего уровня. Управление доступом к передающей среде

Метод доступа- это способ "захвата" передающей среды, способ определения того, какая из рабочих станций сети может следующей использовать ресурсы сети.

Протокол в общем виде - это набор правил для связи между рабочими станциями (компьютерами) сети, которые управляют форматом сообщений, временными интервалами, последовательностью работы и контролем ошибок.

Протоколы передачи данных нижнего уровня (протокол управления каналом) - это совокупность процедур, выполняемых на нижних уровнях семиуровневой эталонной моделью ВОС по управлению потоками данных между рабочими станциями сети на одном физическом канале связи.

Методы доступа к передающей среде, определяющие правила ее "захвата", могут быть разделены на следующие классы.

1. Селективные методы, при реализации которых с помощью соответствующего ППД рабочая станция осуществляет передачу только после получения разрешения, которое либо направляется каждой РС по очереди центральным управляющим органом сети (такой алгоритм называется циклическим опросом), либо передается от станции к станции (алгоритм передачи маркера).
2. Методы, основанные на соперничестве (методы случайного доступа, методы "состязаний" абонентов), когда каждая РС пытается "захватить" передающую среду.
3. Методы, основанные на резервировании времени. Любая РС осуществляет передачу только в течение временных интервалов (слотов), заранее для нее зарезервированных.
4. Кольцевые методы предназначены специально для ЛВС с кольцевой топологией. К ним относятся два метода - вставка регистров и сегментированная передача (метод временных сегментов).

При реализации метода вставки регистра рабочая станция содержит регистр (буфер), подключаемый параллельно к кольцу. В регистр записывается кадр для передачи, и станция ожидает межкадрового промежутка в моноканале. С его появлением регистр включается в кольцо (до этого он был отключен от кольца) и содержимое регистра передается в линию. Если во время передачи станция получает кадр, он записывается в буфер и передается вслед за кадром, передаваемым этой станцией. Этот метод допускает "подсадку" в кольцо нескольких кадров.

При использовании в ЛВС с кольцевой топологией сегментированной передачи временные сегменты формируются управляющей станцией сети. Они имеют одинаковую протяженность и циркулируют по кольцу. Каждая станция, периодически обращаясь в сеть, может дождаться временного сегмента, помеченного меткой "свободный". В этот сегмент станция помещает свой кадр фиксированной длины, при этом в сегменте метка "свободный" заменяется меткой "занятый". После доставки кадра адресату

сегмент вновь освобождается. Важным преимуществом такого метода является возможность одновременной передачи кадров несколькими РС. Однако передача допускается только кадрами фиксированной длины.

Используется и другая классификационная структура. Все ППД делятся на два класса:

1. ППД типа первичный/вторичный;
2. равноранговые ППД.

При реализации ППД первого класса в сети выделяется первичный (главный) узел, который управляет всеми остальными (вторичными) узлами, подключенными к каналу, и определяет, когда и какие узлы могут производить обмен данными. В сетях, где реализуются равноранговые (одноуровневые, одноранговые) протоколы, все узлы имеют одинаковый статус. Однако, если предварительно узлам присвоить разные приоритеты, то для них устанавливается неравноправный доступ в сеть.

Контрольные вопросы.

1. Каковы понятия: телекоммуникационная вычислительная сеть, абонентская система, коммуникационная подсеть, или телекоммуникационная система, прикладной процесс.
2. Перечислите основные задачи, режимы работы ТВС.
3. Каковы преимущества ТВС по сравнению с автономно работающими ЭВМ
4. Каковы понятия: аппаратное, информационное, программное обеспечение ТВС.
5. Опишите основные классификационные признаки сетей.
6. Опишите семиуровневую эталонную модель взаимодействия открытых систем (ВОС).
7. Каковы понятия: метод доступа, протокол, протоколы передачи данных нижнего уровня (протокол управления каналом).
8. Опишите методы доступа к передающей среде.

Тема 3.2. Телекоммуникационные системы (ТКС)

Цель изучения темы: усвоить основные функции ТКС; изучить типы сетей, линий и каналов связи; изучит аналоговое и цифровое кодирование цифровых данных; изучить метод импульсно-кодовой модуляция (ИКМ); изучить метод коммутации в цепях; изучить цели и способы маршрутизации пакетов; изучить методы защиты от ошибок.

Вопросы

1. Функции ТКС.
2. Типы сетей, линий и каналов связи.
3. Коммутируемые и выделенные каналы.
4. Аналоговое и цифровое кодирование цифровых данных.
5. Цифровые сети связи.
6. Спутниковые сети связи.
7. Коммутация в сетях.
8. Маршрутизация пакетов в сетях.
9. Методы защиты от ошибок.

В 3.2.1. Функции ТКС.

Основная функция телекоммуникационных систем (ТКС), или систем передачи данных (СПД), в условиях функционирования ТВС заключается в организации оперативного и надёжного обмена информацией между абонентами. Время доставки информации зависит от ряда факторов:

1. структуры сети связи,
2. пропускной способности линии связи,
3. способов соединения каналов связи между взаимодействующими абонентами,
4. протокол информационного обмена,
5. методов доступа абонентов к передающей среде,
6. методов маршрутизации пакетов.

3.2.2. Типы сетей, линий и каналов связи

В ТВС используются сети связи - телефонные, телеграфные, телевизионные, спутниковые. В качестве линий связи применяются: кабельные (обычные телефонные линии связи, витая пара, коаксиальный кабель, волоконнооптические линии связи (ВОЛС), или световоды), радиорелейные, радиолинии.

В ТВС нашли применения следующие типы каналов связи:

- симплексные, когда передатчик и приёмник связываются одной линией связи, по которой информация передается только в одном направлении (это характерно для телевизионных сетей связи);
- полудуплексные, когда два узла связи соединены также одной линией, по которой информация передаётся попеременно то в одном направлении, то в противоположном (это характерно для информационно-справочных, запрос-ответных систем);
- дуплексные, когда два узла связи соединены двумя линиями (прямой линией связи и обратной), по которым информация одновременно передаётся в противоположных направлениях. Дуплексные каналы применяются в системах с решающей информационной обратной связью.

3.2.3. Коммутируемые и выделенные каналы связи

В ТКС различают выделенные (некоммутируемые) каналы связи и с коммутацией на время передачи информации по этим каналам.

При использовании выделенных каналов связи постоянно соединена между собой. Этим обеспечиваются высокая степень готовности системы к передачи информации, более высокое качество связи, поддержка большого объёма трафика.

Для коммутируемых каналов связи, создаваемых только на время передачи фиксированного объёма информации, характерны высокая гибкость и сравнительно небольшая стоимость (при малом объёме трафика). Недостатки таких каналов: потери времени на коммутацию (установление связи между абонентами), возможность блокировки из-за занятности отдельных участков линии связи, более низкое качество связи, большая стоимость при значительном объёме трафика.

3.2.4. Аналоговое и цифровое кодирование цифровых данных

Пересылка данных от одного узла ТКС к другому осуществляется последовательной передачей всех битов сообщения от источника к пункту назначения. Физически информационные биты передаются в виде аналоговых или цифровых электрических сигналов. Аналоговыми называются сигналы, которые могут представлять бесчисленное количество значений некоторой величены в пределах ограниченного диапазона.

Цифровые (дискретные) сигналы могут иметь одно или конечный набор значений. При работе с аналоговыми сигналами для передачи закодированных данных используется аналоговый несущий сигнал синусоидальной формы, а при работе с цифровыми сигналами - двухуровневый дискретный сигнал. Аналоговые сигналы менее чувствительны к искажению, обусловлено затуханием в передающей среде,

зато кодирование и декодирование данных проще осуществляются для цифровых сигналов.

Аналоговое кодирование применяется при передаче цифровых данных по телефонным (аналоговым) линиям связи. Перед передачей цифровые данные, поступающие обычно из ЭВМ, преобразуются в аналоговую форму с помощью модулятора-демодулятора (модема), обеспечивающего цифро-аналоговый интерфейс.

Цифровое кодирование цифровых данных выполняется напрямую, путём изменения уровней сигналов, несущих информацию.

Например, если в ЭВМ цифровые данные представляются сигналами уровней 5В для кода 1 и 0,2В для кода 0, то при передачи этих данных в линию связи уровни сигналов преобразуются в соответственно в +12В и -12В. Такое кодирование осуществляется, в частности, с помощью асинхронных последовательных адаптеров RS-232-C при передачи цифровых данных от одного компьютера к другому на небольшие (десятки и сотни метров) расстояния.

Синхронизация - это часть протокола связи.

В процессе синхронизации связи обеспечивается синхронная работа аппаратуры приёмника и передатчика, при которой приёмник осуществляет выборку поступающих информационных битов (т.е. замер уровня сигнала в линии связи) строго в момент их прихода. Синхросигналы настраивают приёмник на передаваемое сообщение ещё до его прихода и поддерживают синхронизацию приёмника с приходящими битами данных.

В зависимости от способов решения проблемы синхронизации различают синхронную передачу, асинхронную передачу и передачу с автоподстройкой.

Синхронная передача отличается наличием дополнительной линии связи для передачи синхронизирующих импульсов (СИ) стабильной частоты.

Асинхронная передача не требует дополнительной лини связи. Передача осуществляется небольшими блоками фиксированной длины (байтами).

Передача с автоподстройкой, также не требующая дополнительной линии связи, применяется в современных высоко скоростных системах передачи данных. Синхронизация достигается за счёт использования самосинхронизирующих кодов (СК).

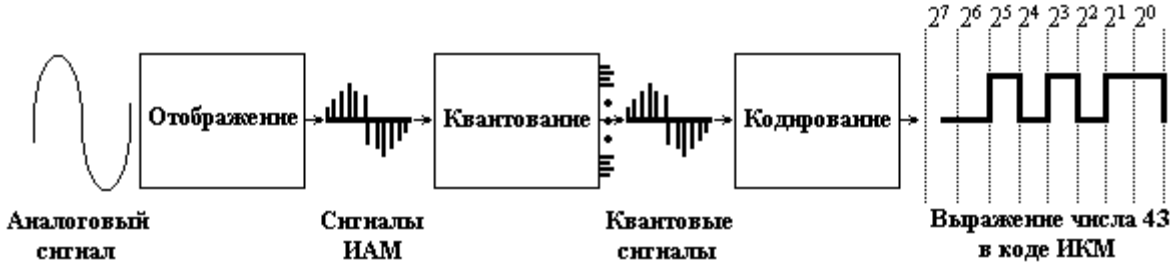
3.2.5 Цифровые сети связи (ЦСС)

В ЦСС при передаче информации осуществляется преобразование аналогового сигала в последовательность цифровых значений, а при приёме - обратное преобразование.

Преобразование аналоговых сигналов в цифровые осуществляется различными методами.

Один из них - импульсно-кодовая модуляция (ИКМ). При использовании ИКМ процесс преобразования включает три этапа: отображение, квантование и кодирование (рис. 3.2.1).

Рисунок 3.2.1. Преобразование аналогового сигнала в 8-элементный цифровой код



Первый этап (отображение) основан на теории отображения Найквиста. Основное положение этой теории гласит: "Если аналоговый сигнал отображается на регулярном интервале с частотой не менее чем в два раза выше максимальной частоты исходного сигнала в канале, то отображение будет содержать информацию, достаточную для восстановления исходного сигнала". При передаче акустических сигналов (речи) предоставляющие их электрические сигналы в телефонном канале занимают полосу частот от 300 до 3300 Гц. Поэтому в ЦСС принята частота отображений, равная 8000 раз в секунду. Отображения, каждое из которых называется сигналом импульсно-амплитудной модуляции (ИАМ), запоминаются, а затем трансформируются в двоичные образы.

На этапе квантования каждому сигналу ИАМ придаётся квантованное значение, соответствующее ближайшему уровни квантования. В ЦСС весь диапазон изменения амплитуды сигналов ИАМ разбивается но 128 или 256 уровней квантования. Чем больше уровней квантования, тем точнее амплитуда ИАМ-сигнала предоставляется квантованным уровнем.

На этапе кодирования каждому квантованному отображения ставится в соответствие 7-разрадяный (если число уровней квантования равно 128) или 8-разрадяный (при 256-шаговом квантовании) двоичный код. На рис. 1 показаны сигналы 8-элементного двоичного кода 00101011, соответствующего квантовому сигналу с уровнем 43. При кодировании 7-элементными кодами скорость передачи данных по каналу должна

составлять 56 Кбит/с (это произведение частоты отображения на разрядность двоичного кода), а при кодировании 8-элементнгыми кодами - 64 Кбит/с.

3.2.6 Спутниковые сети связи

В спутниковых системах связи используются антенны СВЧ-диапазона частот для приёма радиосигналов от передающих наземных станций и для ретрансляции этих сигналов обратно на наземные станции. Способность спутника принимать и передавать сигналы обеспечивается специальным устройством - транспондером. Взаимодействие между абонентами осуществляется по цепи: абонентская станция (отправитель информации) - передающая наземная радиотелеметрическая станция (РТС) - спутник - приёмная наземная радиотелеметрическая станция - абонентская станция (получатель информации). Одна наземная РТС обслуживает группу ближайших АС.

К основным преимуществам спутниковых сетей связи относятся следующие:

- большая пропускная способность, обусловленная работой спутников в широком диапазоне гигагеровых частот. Спутник может поддерживать несколько тысяч речевых каналов связи;
- обеспечение связи между станциями, расположенных на очень больших расстояниях, и возможность обслуживания абонентов в самых труднодоступных точках.

3.2.7 Коммутация в сетях

Узлы сети подключаются к некоторому коммутирующему оборудованию, избегая таким образом необходимости создания специальных линий связи.

Коммутируемой транспортной сетью называется сеть, в которой между двумя (или более) конечными пунктами устанавливается связь по запросу. Примером такой сети является коммутированная телефонная сеть.

Существуют следующие методы коммутации:

- коммутация сетей (каналов);
- коммутация с промежуточным хранением, в свою очередь, разделяемая на коммутацию сообщений и коммутацию пакетов.

При коммутации цепей (каналов) между связываемыми конечными пунктами на протяжении всего временного интервала соединения обеспечивается обмен в реальном масштабе времени, причём биты передаются с неизменной скоростью пол каналу с постоянной полосой пропускания. Между абонентами устанавливается сквозной канал связи до начала передачи

информации. Этот канал формируется из отдельных с одинаковой пропускной способностью

Вызывающий объект посредством набора номера или через выделенную линию связывается только с ближайшим узлом сети и передаёт ему информационные биты. В каждом узле имеется коммутатор, построенный на базе коммуникационной ЭВМ с запоминающим устройством (ЗУ). Передаваемая информация должна храниться в каждом узле по пути к пункту назначения, причём задержка в хранении, как правило, будет различной для узлов.

При коммутации пакетов пользовательские данные (сообщения) перед началом передачи разбиваются на короткие пакеты фиксированной длины. Каждый пакет снабжается протокольной информацией: коды начала и окончания пакета, адреса отправителя и получателя, номер пакета в сообщении, информация для контроля достоверности передаваемых данных в промежуточных узлах связи и в пункте назначения. Будучи независимыми единицами информации, пакеты, принадлежащие одному и тому же сообщению, могут передаваться одновременно по различным маршрутам в составе дейтограмм. Управление передачей и обработкой пакетов в узлах связи осуществляется с центрами коммутации пакетов (ЦКП) с помощью компьютеров. Длительное хранение пакетов ЦКП не предполагается, поэтому пакеты доставляются в пункт назначения с минимальной задержкой, где из них формируется первоначальное сообщение.

Иначе - субпакетная коммутация, или метод общего пакета представляет собой разновидность пакетной коммутации. Она применяется в случае, когда пакет содержит информационные биты, принадлежащие различным пользователям.

При использовании символьной коммутации оптимальный размер пакета для конкретной передающей среды сохраняется с одновременным уменьшением времени задержки пакета в сети. Это достигается за счёт приёма от нескольких пользователей небольшого количества символов (информационных битов) и загрузки их в один пакет общего доступа.

3.2.8 Маршрутизация пакетов

Задача маршрутизации в выборе маршрута для передачи от отправителя к получателю. Она имеет смысл в сетях, где не только необходим, но и возможен выбор оптимального или приемлемого маршрута. В современных сетях со смешанной топологией решается задача выбора маршрута для передачи кадров, для чего используется соответствующие средства, например маршрутизаторы.

Выбор маршрутов в узлах связи ТКС производится в соответствии с реальным алгоритмом (методом) маршрутизации.

Алгоритм маршрутизации - это правило назначения выходной линии связи данного узла связи ТКС для передачи пакета, базирующегося на информации, содержащейся в заголовке пакета (адреса отправителя и получателя), и информации о загрузке этого узла (длина очередей пакетов) и, возможно, ТКС в целом.

Цели маршрутизации:

- минимальной задержки пакета при его передачи от отправителя к получателю;
- максимальной пропускной способностью сети;
- максимальной защиты пакета от угроз безопасности содержащейся в нём информации;
- надёжности доставки пакета адресату;
- минимальной стоимости передачи пакета адресату.

Различают следующие способы маршрутизации:

1. централизованная маршрутизация реализуется обычно в сетях с централизованным управлением. Выбор маршрута для каждого пакета осуществляется в центре управления сетью, а узлы сети связи. Только воспринимают и реализуют результаты решения задачи маршрутизации.
2. Распределённая (децентрализованная) маршрутизация выполняется главным образом в сетях с децентрализованным управлением. Функции управления маршрутизации распределены между узлами сети, которые располагаются для этого соответствующими средствами.
3. Смешанная маршрутизация характеризуется тем, что в ней определённом соотношении реализованы принципы централизованной и распределённой маршрутизации.

Для выбора оптимального маршрута каждый узел связи должен располагать информацией о состоянии ТКС в целом - всех остальных узлов и линий связи.

3.2.9 Методы защиты ошибок

Причины возникновения ошибок при передачи информации в сетях:

- сбой в какой-то части оборудования сети или возникновение неблагоприятных объективных событий в сети (например, коллизий при использовании метода случайного доступа в сеть);

- помехи, вызванные внешними источниками и атмосферными явлениями. Помехи - это электрические возмущения, возникающие в самой аппаратуре или попадающие в неё извне.

Выделяются три группы методов защиты от ошибок:

1. групповые методы;
2. помехоустойчивое кодирование;
3. методы защиты от ошибок в системах передачи с обратной связью.

Из **групповых методов** получили широкое применение мажоритарный метод и метод передачи информационными блоками с количественной характеристикой блока.

Суть мажоритарного метода, давно и широко используемого в телеграфии, состоит в следующем. Каждое сообщение ограниченной длины передаётся несколько раз, чаще всего три раза. Принимаемые сообщения запоминаются, а потом производится их поразрядное сравнение. Суждение о правильности передачи выносится по совпадению большинства из принятой информации методом "два из трёх". Например, кодовая комбинация 01101 при трёх разовой передаче была частично искажена помехами, поэтому приёмник принял такие комбинации: 10101, 01110, 01001. В результате проверки каждой позиции отдельно правильной считается комбинация 01101.

Другой групповой метод, также не требующий перекодирования информации, предполагает передачу данных блоками с количественной характеристикой блока. Такими характеристиками могут быть: число единиц или нулей в блоке, контрольная сумма передаваемых символов в блоке, остаток от деления контрольной суммы на постоянную величину и др. На приёмном пункте эта характеристика вновь подсчитывается и сравнивается с переданной по каналу связи. Если характеристики совпадают, считается, что блок не содержит ошибок. В противном случае на передающею сторону передаётся сигнал с требованиями повторной передачи блока. В современных ТВС такой метод получил самое широкое распространение.

Помехоустойчивое (избыточное) кодирование, предлагающая разработку и использование корректирующих (помехоустойчивых) кодов, широко применяется не только в ТКС, но и в ЭВМ для защиты от ошибок при передачи информации между устройствами машины.

делятся на системы с решающей Системы передачи с обратной связью обратной связью и системы с информационной обратной связью.

Особенностью систем с решающей обратной связью является то, что решение о необходимости повторной передачи информации принимает приёмник.

В системах с информационной обратной связью передача информации осуществляется без помехоустойчивого кодирования. Приёмник, приняв информацию по прямому каналу обратной связи передатчику, где переданная и возвращённая информация сравнивается. При совпадении передатчик посылает приёмнику сигнал подтверждения, в противном случае происходит повторная передача всей информации. Таким образом, здесь решение о необходимости повторной передачи принимает передатчик.

Контрольные вопросы.

1. Какова основная функция телекоммуникационных систем (ТКС), или систем передачи данных (СПД), в условиях функционирования ТВС ?
2. Перечислите и опишите типы сетей, линий и каналов связи.
3. Каково понятие коммутируемые и выделенные каналы?
4. Каково понятие аналогового и цифрового сигналов?
5. Опишите аналоговое и цифровое кодирование цифровых данных.
6. Опишите преобразование аналоговых сигналов в цифровые методом импульсно-кодовой модуляции (ИКМ).
7. Перечислите и опишите методы коммутации в сетях.
8. Перечислите цели и способы маршрутизации пакетов в сетях.
9. Перечислите методы защиты от ошибок.

Тема 3.3. Глобальные вычислительные сети и сетевые технологии

Вопросы.

1. Протоколы обмена данными.
2. Протокол TCP/IP.
3. Системы сетевых коммутаций.
4. Электронная почта стандарта X400.
5. Электронная почта стандарта Internet.
6. Системы телеконференций.
7. Сеть Internet.
8. Отечественные глобальные сети.

Цели занятий.

- Изучить способы обмена даннымив сетях.
- Изучить протоколы обмена данными X25, TCP/IP.
- Усвоить основные отличия сети Internet от других сетей.

3.3.1 Протоколы обмена данными в сетях

Организация обмена данными в сетях может осуществляться двумя различными способами: без установления логического соединения между передающим и принимающим узлами сети и с установлением логического соединения (с установлением сеанса связи).

Способ связи без установления логического соединения характеризуется в следующем:

- он используется в сетях с коммутацией пакетов, причём каждый пакет рассматривается как индивидуальный объект, независимая единица передачи информации;
- пакеты от отправителя можно передавать в произвольные моменты, одновременно множеству адресатов по различным маршрутам;
- перед передачей данных сквозная связь между отправителем и получателем заранее не устанавливается, не требуется так же синхронизации аппаратуры связи на передающем и приёмных пунктах;
- из-за занятность отдельных участков маршрута может осуществляться буферизация пакетов в промежуточных узлах связи;
- передача сигнала от адресата к отправителю, подтверждающее получение информации, не производится.

Способ связи (или режим связи), ориентированный на логическое соединение, относится к более поздней технологии. Он обеспечивает более высокий уровень сервиса по сравнению с дейтаграммой связью.

Особенности организации обмена данными с установлением логического соединения:

- перед передачей информации между взаимодействующими абонентами (отправителем и получателем) устанавливается логический (виртуальный) канал, причём технология создания (установления) канала такова: отправитель отправляет запрос на соединение удалённому адресату через ряд промежуточных узлов связи; адресат, получив этот запрос, в случае "согласия" на установления логического канала посылает отправителю сигнал подтверждения; после получения сигнала подтверждения отправителем начинается обмен данными с управлением потоком, сегментацией и исправлением ошибок;
- после завершения обмена данными адресат посылает пакет подтверждения этого события отправителю (клиенту - инициатору установления логического канала), который воспринимается как сигнал для разъединения канала. Следовательно, при использовании этого способа связи выделяются три этапа: установление канала, обмен данными, разъединения канала.

Протоколы обмена данными, или протокол верхнего уровня служат, для управления обменом данных. Независимо от внутренней конструкции каждого конкретного протокола верхнего уровня для них характерно наличие общих функций: инициализация связи, передача и приём данных, завершение обмена.

К наиболее распространеннм протоколам обмена данными относятся: X25, TCP/IP.

3.3.2. Протокол X 25

Большинству функционирующих в настоящие время общедоступных ТВС базируется на протоколе X25, одобрено МККТТ в качестве стандарта. Сети, использующие этот протокол, называются так же сетями X 25.

Сети X25, как правило, работают по аналоговым каналам. Следовательно, для подключения компьютера к сети необходим преобразователь цифровых данных в аналоговые сигналы - модем. В сетях X25 они называются адаптерами или модемами. Сети на основе X25 - это сети с коммутацией пакетов. Каждая рабочая станция такой сети соединена с центром коммутации пакетов (ЦКП), а посылаемая в сеть информация комплектуется в последовательность пакетов. Длина пакета ограничена, максимальный

размер - 1024 байта. В пакете имеется адресная часть, которая используется центром коммутации для рассылки пакетов по адресам.

Протокол X25 описывает интерфейс между рабочей станцией и аппаратурой синхронной передачи данных в сеть на трёх нижних уровнях семиуровневой модели ВОС: физическом уровне (стандартизация на уровне сигналов), канальном (процедуры установления связи) и сетевом (протоколы этого уровня описывают форматы пакетов). На основе сети X25 можно построить сеть с протоколом TCP/IP.

Скорость передачи данных в сети X25 составляет 64 Кбайт/с, что превосходит максимальную скорость последовательного порта компьютера которая равна 56 Кбайт/с Сети стандарта X25 легко интегрируются, поэтому они объединены в глобальную мировую сеть X25, в которой система сетевой адресации определена рекомендациями МККТТ X 121.

Цифровые сети интегрального обслуживания (ISDN - Integrated Services Digital Network) рассматриваются как ближайшее будущие сетей общего пользования, в том числе и сети X25. Поскольку в цифровой сети не требуется модуляции аналогового сигнала, связь компьютера с сетью осуществляется не через модем, а через адаптер ISDN. Линия ISDN имеет три цифровых канала, по каждому из которых данные передаются со скоростью 64 Кбайт/с.

При построении сети ISDN используется также скоростные магистральные цифровые каналы T1, основанные на технике импульсно-кодовой модуляции - способе преобразования аналоговых сигналов в цифровые. По каналу T1 цифровые сигналы передаются по проводной паре со скоростью 1544000 бит/с. это число получено следующим образом. Канал T1 обеспечивает передачу данных 24 каналами ИКМ, что является общепринятой системой в США. Сигналы мультиплексируются в линию связи в такой последовательности: сначала 8 бит с первого канала ИКМ, затем 8 бит со второго канала и так далее до канала 24, потом добавляется один бит - разделитель кадра. Длина кадра получается равной 24*8+1=193 битам. Таких кадров передаётся 8000 в секунду. Общая скорость получается 193*8000=1544000 бит/с.

В Европе и России принята другая схема. Используется канал E1, включающий 32 канала ИКМ по 64 бита/с (общая скорость 2048 Кбит/с), из них 30 каналов - для передачи пользовательских данных и 2 канала - для передачи служебной информации. Следовательно, в магистрали одновременно функционируют 30 телефонных каналов с пропускной способностью 64 Кбит/с каждый.

Существуют и другие системы TX и EX, отличающиеся количеством каналов на провод, T4M - 4032 канала на коаксиальный кабель. В системах, где

используются оптоволоконные линии связи, количество каналов ИКМ значительно больше: FT3 - 672 канала, FT3C - 1374, FT-4E-144 - 2016, FT-4E-432 - 6048 каналов по 64 Кбайт/с каждый.

3.3.3 Протокол TPC/IP

Это целое семейство программно реализованных протоколов (более 100) старшего уровня, неработающих с аппаратными прерываниями.

Технически протокол TCP/IP состоит двух частей - TCP и IP. Протокол IP (Internet Protocol - межсетевой протокол) описывает формат пакета данных, передаваемого в сети. Он является первым и самым низким уровнем (а именно третьим, т.е. сетевым уровнем модели BOC) в иерархии TCP/IP.

Сети, в которых используется протокол IP, называется IP-сетями. Они также работают в основном по аналоговым каналам (т.е. для подключения компьютера к сети требуется IP-модем), и являются с сетями с коммутацией пакетов. В IP-сетях максимальный размер пакета равен 576 байтам. Пакет здесь называется дейтограммой, он является бит-ориентированным, и его размер определяется в битах. Протокол IP был разработан в США для организации связи локальных сетей департаментом обороны.

Высокоуровневый протокол связи TPC (Transfer Control Protocol - протокол контроля передачи) предназначен для контроля передачи, контроля целостности передаваемой информации. Он обеспечивает связь между двумя узлами с гарантированной доставкой информации. Следовательно, компьютеры сети обмениваются пакетами протокола IP, а контроль передачи осуществляется по протоколу TCP (на уровне IP коррекция ошибок не производится). Для компьютеров протокол TCP/IP - это как правило разговор для людей. Он принят в качестве официального стандарта в сети Internet, т.е. сетевая технология TCP/IP де-факто стала технологией всемирной сети.

Протокол TCP/IP основывается на концепции одноранговых сетей. Все рабочие станции, соединены при помощи этого протокола, имеют одинаковый статус. Однако любая из них, располагая соответствующими средствами, может временно выполнять дополнительные функции, связанные, например, с управлением ресурсами сети. Ключевую часть протокола составляет схема маршрутизации пакетов, основанная на уникальных адресах сети Internet. Каждая рабочая станция, входящая в состав локальной или глобальной сети, имеет уникальный адрес, который включает две части, определяющие адрес сети и адрес станции внутри сети. Такая схема позволяет передавать сообщения как внутри данной сети, так и во внешней сети. Часть протокола TCP/IP, отвечающая за распознавание адреса, называется IRP (протокол распознавания адреса).

3.3.4 Электронная почта стандарта X400

Система электронной почты X400 рекомендована международными стандартизирующими организациями. Имеет место тенденция государственных органов во всём мире при построении подведомственных им сетевых образований ориентироваться преимущественно на применении X400. X400 не сеть, а стандарт для организации службы ЭП. Следовательно, абоненты, имея доступ и адреса в системе X400, должны обмениваться письмами через сети, услугами которых они пользуются. В системе X400, как и в большинстве других системах ЭП, предоставляются услуги по доставки твёрдой копии электронного письма тем пользователям, которые не имеют доступа к компьютеру. В этом случае письмо посылается адресу компьютера, ближайшего адресата, для которого письмо необходимо отпечатать на бумаге и доставить посыльным.

3.3.5 Электронная почта стандарта Internet

В значительной части мировых ТВС используется система электронной почты стандарта Internet. В России действует система электронной почты РЕЛКОМ, которая на правах национальной сети имеет доступ в европейскую сеть EVNET, представляющую собой составную часть сетевого конгломерата, называемого Internet.

Электронное письмо (тестовый файл) составляется пользователем по определённым правилам. Оно состоит из заголовка и текста письма. Заголовок включает реквизиты, называемые полями. Каждое поле состоит из имени и значения поля. Заголовок обычно содержи адреса отправителя и получателя, дату создания письма и его тему.

Для внешних средств коммутации нашли место два стиля, или две системы адресации:

- явная адресация, называемая стилем UUCP;
- доменная адресация, называемая стилем Internet.

При явной адресации маршрут к адресату задаётся перечислением имён компьютеров, через которые последовательно передаётся электронные письмо или любое другое сообщение. Последним именем в этой последовательности является имя адресата на последнем указанном компьютере. При модемной связи в качестве имени компьютеров указывается телефонный номер, т.е. адрес абонента выглядит так: имя узлового компьютера - имя компьютера абонента - сетевое имя абонента.

К числу недостатков явной адресации относятся: возможность транспортировки писем по весьма протяжённым маршрутам, вероятность отказа одного (или нескольких) из компьютеров в цепочке из машин указанного маршрута.

В доменной системе адресации Internet каждый корреспондент получает сетевой адрес, включающий две составляющие: идентификатор пользователя (userid) и изменение узла (noteid). Идентификатор userid является уникальным для узла сети. Идентификатор noteid представляет собой текстовую строку, состоящею из доменов, разделяемых точкой. Адрес читается с права на лево и состоит из зарегистрированных доменов в сети.

В системе DNS ключевым является понятие "полностью определённое имя домена" - это имя домена, которые включает все домены более высокого уровня и образует таким образом полное, целое имя. Структуру DNS можно представить в виде дерева, каждый уровень которого имеет своё название. Для каждого конкретного "полностью определённое имя домена" будет состоять из его имени и имени всех узлов, связывающих его с корнем дерева, причём корневой домен всегда нулевой.

3.3.6 Система телеконференций

В системе телеконференции (ТК) принцип электронной почты получил дальнейшее развитие. Если в системе ЭП сообщения адресуются "один к одному" и каждому пользователю предоставляется индивидуальный "почтовый ящик", то в системе ТК адресация осуществляется по принципу "один ко всем" и на всех участников ТК выделяется один ящик.

В развитии мировых сетей важнейшую роль играет метасеть телеконференций USENET неразрывно связанная с сетью Internet. Датой образования USENET считается 1979 г.

Список конференций USENET включает тысячи тем, поэтому важно знать правила, в соответствии с которыми устанавливаются иерархические имена конференций. Эти имена уточняют принадлежность конференции к определённому тематическому разделу - иерархии. Часто темы пересекаются, и многие статьи отправляются сразу в несколько конференций.

В число основных иерархий конференций, присутствующих на всех крупных системах сети ТК USENET, входят:

- comp - конференции по вопросам, связанные с компьютерами и программированием;
- misc - темы, не входящие не в один из основных классов или относящихся сразу к нескольким;
- news - вопросы по программам обмена новостями и развитию системы конференции;
- rec - вопросы отдыха, хобби, увлечений;
- sci - конференции для дискуссий и обмена опытом по различным научным дисциплинам;
- soc - вопросы общественной жизни;

- talk - конференции, ориентированы на обсуждение спорных вопросов по любой тематике.

Для получения от сервера списка конференций по интересующей тематике необходимо послать в адрес сервера E-mail соответствующую команду-запрос.

3.3.7 Сеть Internet

Общее число только достаточно крупных компьютерных сетей в мире достигает несколько сотен. Наиболее известными являются:

- Telenet (США);
- TYMNET (США);
- Datapac (Канада);
- PSS (Англия);

Для координации работ в области мировой сетевой интеграции была создана широко известная сегодня организация - консорциум Internet. Это всемирная компьютерная сеть, сеть сетей, объединяющая посредством межсетевых интерфейсов многие сети, поддерживающие протокол TCP/IP. Все сети Internet используют протоколы TCP/P, для удалённого доступа большинство поддерживает протоколы TELENET, для передачи файлов - FTR, для электронной почты - SMTR.

С технической точки зрения Internet - объединение транспортных компьютерных сетей, работающих по самым разнообразным протоколам, связывающие всевозможные типы компьютеров, физически передающих данные по телефонным проводам и оптоволокну, через спутники и радиомодемы.

Основные отличия сети Internet от других сетей заключается в её протоколах TCP/IP, охватывающих целое семейство протоколов взаимодействия между компьютерами в сети, прикладные программы и саму сеть. TCP/IP - это технология межсетевого взаимодействия, технология Internet. Сеть, реализующая эту технологию называется "internet". Если же речь идёт о глобальной сети, объединяющей множество сетей с технологией internet, то есть её называют Internet.

В Internet, как и во всякой другой сети имеются семь уровней прикладных процессов, причём каждому уровню соответствует набор протоколов.

Протоколы физического уровня определяют вид и характеристики линий связи между компьютерами. Для каждого типа линий связи разработан протокол канального (логического) уровня, занимающийся управлением передачи информации по каналу. Протоколы сетевого уровня обеспечивают

маршрутизацию пакетов в сети, т.е. отвечают за передачу данных между компьютерами в разных сетях. К ним относятся протоколы IP и ARP (Address Resolution Protocol).

На транспортном уровне управление передачей данных осуществляется протоколами TCP и UDP (User Datagram Protocol). Протоколы сеансового уровня отвечают за установку, поддержание и уничтожение соответствующих каналов. В Internet этим занимаются протоколы TCP, UDP, UUCP.

На представительном уровне протоколы занимаются обслуживанием прикладных программ. К программам предварительного уровня относятся, программы, telnet-сервер, FTP- сервер, Gopher- сервер, NNTP (Net News Transfer Protocol), SMTP (Simple Mail Transfer Protocol) и др. К протоколам прикладного уровня относятся программы предоставления сетевых услуг.

Все сервисы сети Internet можно разделить на три группы:

1. Интерактивные;
2. Прямого обращения;
3. Отлажённого чтения.

К группе интерактивных сервисов относятся такие, где требуется немедленная реакция от получателя информации, т.е. получаемая информация, в сущности, является запросом.

Сервисы прямого обращения характеризуются тем, что информация по запросу возвращается немедленно.

Наиболее распространенные являются сервисы отложенного чтения, например электронная почта. Для них основным признаком служат та особенность, что запрос и получение информации могут быть достаточно сильно разделены во времени. Сервисы отложенного чтения наиболее универсальны и наименее требовательны к ресурсам ЭВМ и линиям связи. Электронная почта - типичный сервис отложенного чтения ("off-line").

Система электронной почты E-mail стандарта Internet универсальна: сети, построены на совершенно разных принципах и протоколах, могут обмениваться электронными письмами с Internet, получая тем самым доступ к прочим его ресурсам.

В Internet есть возможность отправки как текстовых, так и двоичных файлов. На размер почтового сообщения в сети накладывается ограничение: он не должен превышать 64 Кбайта.

FTP-File Transfer Protocol - протокол передачи файлов. Это не просто протокол, а именно сервис-доступ к файлам в файловых архивах. FTP - это стандартная программа, работающая по протоколу TCP. Она обеспечивает передачу фалов между компьютерами, взаимодействующими в сетях TCP/IP: на одном из них работает программа-сервер, а на другом пользователь запускает программу-клиент, которая соединяется с сервером и передаёт или получает по протоколу FTP файлы.

Система WWW. World Wide Web - всемирная паутина. WWW - самое популярное и удобное средство работы с информацией. В Internet реализуются две стороны поиска информации, разные по методам, но едины в целях: каталоги и поисковые серверы. Условно можно сказать, что каталоги - средства сфокусированного поиска информации, а поисковые сервера - рассеянного.

Поисковые серверы - это выделенные компьютеры, которые, автоматически просматривая все ресурсы сети Internet, могут найти запрашиваемые ресурсы и проиндексировать их содержание.

Каталоги Internet - средства хранения тематически систематизированных коллекций ссылок на различные сетевые ресурсы, в первую очередь на документы WWW.

Кроме описанных услуг сетью Internet предоставляются и другие услуги. Краткие сведения о некоторых из них даются ниже.

Telenet - удалённый доступ. Пользователь может работать на любом компьютере сети, как на своём собственном.

В 3.3.8 Отечественные глобальные сети.

На российском рынке ГВС наиболее активно и эффективно функционируют следующие сетевые ресурсы:

- информационная сеть Internet/Россия, существующая с 1990 г. и объединяющая более 300000 абонентов, среди которых - научные, учебные и правительственные организации, банки, биржи, информационные агентства, частные лица. Эта сеть является частью международной ассоциации компьютерных сетей Internet;
- сеть IASNET, созданная в 1985 г. Институтом автоматизированных систем (ИАС);
- сеть Исток-К, реализованная на отечественных технических средствах и представляющая собой совокупность центров коммутации пакетов и сообщений, соединённых стандартными каналами государственной сети связи;

- сеть РОПАК, имеющая в настоящие время в своём составе узлы коммутации пакетов в 50 городах России.
- сеть документального обмена общего использования РЕЛКОМ, созданная в 1990 г. Эта сеть в качестве национальной стала частью европейской сети EVNET и, следовательно, имеет выход в сеть Internet;
- "Спринт-сеть", созданная в 1990 г., является сетью передачи данных и документального обмена, обеспечивает выход на международные сети;
- ведомственная сеть "АКАДЕМСЕТЬ", созданная Всероссийским научно-исследовательским институтам прикладных автоматизированных систем (ИАС) и предназначенная для обеспечения доступа научных и исследовательских организаций к базам данных информационных центров.
- сеть "ИНФОТЕЛ", созданная в 1992 г., предоставляющая собой сеть передачи данных и документального обмена.

Контрольные вопросы

1. Перечислите и опишите способы организация обмена данными в сетях.
2. Опишите протокол обмена данными X25.
3. Опишите особенности цифровые сети интегрального обслуживания (ISDN - Integrated Services Digital Network).
4. Опишите скоростной магистральный цифровой каналы T1.
5. Опишите скоростные магистральные цифровые каналы E1 TX и EX, , T4M - 4032 FT3, FT3C, FT-4E, FT-4E.
6. Опишите протокол TPC/IP. Опишите сеть Internet.

ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа № 1

Задачи по моделям памяти и структурам программ [30]

1) Подготовка простейшей программы типа .EXE и изучение ее структуры. Создайте файл с приведенным ниже текстом программы. Оттранслируйте и скомпонуйте его, получив файл с расширением .EXE. Убедитесь в том, что программа нормально запускается и выполняет предусмотренные в ней действия. Научиться исключать файлы перекрестных ссылок и листинга по переменке или оба сразу в строке вызова транслятора.

```
;Подготовка программы
;TASM имя_ис_прог,,  - ассемблирование исходной программы.
;TLIK имя_об_прог    - компоновка объектной программы.
text    segment 'code'
        assume  cs:text,ds:data,es:data,ss:stack
;Определения
stdout  equ     1           ;Дескриптор стандартного вывода
cr      equ     10          ;Возврат каретки
lf      equ     13          ;Перевод каретки
myproc  proc
        mov     AX,data     ;Инициализируем
        mov     DS,AX       ;сегментный регистр DS
;Выведем на экран строку текста
        mov     AH,40h      ;Функция вывода
        mov     BX,stdout   ;Дескриптор
        mov     CX,meslen   ;Длина сообщения
        mov     DX,offset mes  ;Адрес сообщения
        int     21h         ;Прерывание MS-DOS
;Завершим программу
outprog: mov    AH,4Ch      ;Функция завершения
        mov     AL,00h      ;Код завершения (0)
        int     21h         ;Прерывание MS-DOS
myproc  endp
text    ends
data    segment
;Поля данных
mes     db      'Программа .EXE стартовала',cr,lf
meslen  equ     $-mes
data    ends
stack   segment para stack 'stack'
```

```
        db      128 dup (?)    ;Область стека
stack   ends
        end     myproc
```

2) Подготовка простейшей программы типа .COM и изучение ее структуры. Задание аналогично 1).

```
;Подготовка программы
;TASM имя_ис_прог,,
;TLIK /t имя_об_прог
text    segment 'code'
        org     100h
        assume  cs:text,ds:text,es:text,ss:text
;Определения
stdout  equ     1               ;Дескриптор стандартного вывода
cr      equ     10              ;Возврат каретки
lf      equ     13              ;Перевод каретки
myproc  proc
;Выведем на экран строку текста
        mov     AH,40h          ;Функция вывода
        mov     BX,stdout       ;Дескриптор
        mov     CX,meslen       ;Длина сообщения
        mov     DX,offset mes   ;Адрес сообщения
        int     21h             ;Прерывание MS-DOS
;Завершим программу
outprog: mov    AX,4C00h        ;Функция завершения, код
                                ;завершения = 0
        int     21h             ;Прерывание MS-DOS
myproc  endp
;Поля данных
mes     db      'Программа .COM стартовала',cr,lf
meslen  equ     $-mes
text    ends
        end     myproc
```

Лабораторная работа № 2

Задачи по программированию на языке ассемблера [30]

Задача 1. Циклы. Создать текстовый символьный массив (64 символа, от символа пробела до символа _). Вывести созданную строку на экран.

```
;Основные фрагменты программы
;Заполним строку
     mov   CX,64        ;Число символов
     mov   AL,' '       ;Первый символ - пробел
     mov   SI,0         ;Начальное смещение в строке
pov:  mov   mes[SI],AL   ;Перешлем байт в строку
     inc   SI           ;Увеличиваем на 1 - обращение
                        ;к следующему байту
     inc   AL           ;Следующий символ
     loop  pov          ;Цикл CX раз
;Выведем строку mes на экран
     mov   AH,40h       ;Файловая функция вывода
     mov   BX,1         ;Дескриптор стандартного вывода
     mov   CX,64        ;Столько байт вывести
     mov   DX,offset mes ;Адрес строки
     int   21h          ;Выход в MS-DOS
;Завершим программу
     . . .
;Поля данных
mes   db    64 dup('~')  ;Строка с контрольным содержимым
```

Задача 2. Циклы. Просмотреть изображение символов второй половины кодовой таблицы (коды 128 - 255, всего 128 символов). Для этого создать текстовый символьный массив, состоящий из кодов этих символов, и вывести его на экран.

Задача3. Вложенные циклы. Создать программную задержку. Определить значение параметров программы, позволяющие получить задержки 10, 3 и 1 с.

```
;Основные фрагменты программы
;Организуется демонстрационный цикл из 10 шагов
       mov     CX,10
loop1:  push    CX
;Вывод строки nes на экран
       . . .
;Организуем программную задержку
```

```
        mov     CX,time       ;Число шагов внешнего цикла
outer:  push    CX            ;Сохраним его в стеке
        mov     CX,0          ;64 Кбайт во внутреннем цикле
inner:  loop    inner         ;Тело цикла - из одной строки
        pop     CX            ;Восстановим CX перед ком.
        loop    outer         ;loop внешнего цикла outer
        pop     CX            ;Восстановим CX демонстраци-
        loop    loop1         ;онного цикла loop1
;Поля данных
mes     db      '<>'          ;Демонстрационная строка
time    dw      10            ;Настройка времени задержки
```

Задача 4. Пересылка строк. Переслать строку(массив) произвольной длины, включив ее в состав другой, более длинной строки. Вывести полученную строку на экран.

```
;Основные фрагменты программы
;Перешлем строку
        mov     CX,strlen     ;CX=длина строки
        mov     AX,DX         ;Настроим сегментный ре-
        mov     ES,AX         ;гистр EC на наши данные
        mov     SI,offset str ;DS:SI=адрес исходной строки
        lea     DI,mes+3      ;ES:DI=адрес пересылки
rep:    movsb                 ;Пересылка CX байт
;Выведем строку на экран
        mov     AH,40h        ;Файловая функция вывода
        mov     BX,1          ;Дескриптор стандартного вывода
        mov     CX,80         ;Столько байт вывести
        mov     DX,offset mes ;Адрес строки
        int     21h           ;Переход в MS-DOS
;Поля данных
str     db      ' Пересылаемая строка '
strlen  equ     $-str         ;Длина строки
mes     db      80 dup ('*')  ;Строка приемник
```

Задача 5. Сравнение строк. Сравнить две одинаковые строки, вывести на экран результат сравнения. Модифицировать поля данных программы, сделав строки не равными и выполнить повторно.

```
;Основные фрагменты программы
;Сравним строки
    mov   CX,32             ;Длина строк
    mov   AX,seg str2       ;Сегментный адрес str2
    mov   ES,AX             ;Настроим на него ES
    mov   SI,offset str1    ;DS:SI=адрес str1
    mov   DI,offset str2    ;ES:DI=адрес str2
```

```
repe  cmpsb                ;Сравнение CX байт
     jne   notequ          ;Если не равны, на notequ
;Выведем на экран сообщение mes1 о равенстве строк
     . . .
     jmp   outprog         ;На завершение программы
notequ:
;Выведем на экран сообщение mes2 о неравенстве строк
     . . .
;Поля данных
str1  db    32 dup ('&')      ;Первая строка
str2  db    32 dup ('&')      ;Вторая строка
mes1  db    'Строки одинаковы',10,13
mes2  db    'Строки различаются!',10,13
```

Задача 6. Составить макроопределения типовых действий (например, программной задержки и вывода строки на экран). Написать программу, содержащую макровызовы с разными значениями параметров и проверить правильность ее выполнения.

```
;Основные фрагменты программы
;Макроопределение для программной задержки
;Параметр=задержка в секундах
delay macro time              ;Имя delay, параметр time
      local   outer,inner     ;Локальные метки
      mov     CX,time*3       ;CX=время в сек (коэффициент
                              ;подбирается экспериментально)
outer:  push    CX            ;Сохраним его в стеке
      sub     CX,CX           ;64К шагов во внутреннем цикле
inner:  loop    inner         ;Тело цикла из одной строки
      pop     CX              ;Восстановление CX перед ком.
      loop    outer           ;loop внешнего цикла outer
      endm                    ;Конец макроопределения

;Макроопределение вывода строки на экран
;Первый параметр = адрес строки, второй параметр = ее длина
outstring macro  mes,len
      mov     AH,40h
      mov     BX,1
      mov     CX,len
      mov     DX,offset mes
      int     21h
      endm

;Начало главной процедуры
myproc  proc
```

```
        mov     AX,data     ;Инициализация сегментного
        mov     DS,AX       ;регистра DS
        outstring m1,3      ;Вывести строку m1, 3 байта
        delay   10          ;Задержка на 10 с
        outstring m2,4
        delay   5
        outstring m3,1
;Поля данных
m1      db      '<> '
m2      db      '*** '
m3      db      '!'
```

Задача 7. Оформить макроопределения из предыдущей задачи в виде отдельного файла с именем, например, MACRO.LIB, создать тем самым макробиблиотеку пользователя. Написать программу, содержащую обращение к макроопределениям из макробиблиотеки.

```
;Основные фрагменты программы
        include macro.lib
        . . .
        outstring m1,3
        delay   10
        outstring m2,4
        delay   5
        outstring m3,1
;Поля данных
m1      db      '<> '
m2      db      '*** '
m3      db      '!'
```

Задача 8. Подпрограммы. Оформить фрагменты, организующие программную задержку и вывод строки на экран, в виде подпрограмм, включенных в текст основной программы (в виде отдельных процедур). Вызвать процедуры из главной программы, передавая параметры через выделенные для этого регистры.

```
;Основные фрагменнты программы
;Подпрограмма Delay задержки
;На входе CX=значение задержки в секундах
delay   proc    near
        push    DX      ;Сохраним регистры, неявно исполь-
        push    AX      ;зуемые подпрограммой
        xchg    AX,CX   ;AX=значение задержки
        mov     CX,3    ;Коэффициент перевода в секунды
                        ;(следует подобрать экспериментально)
```

```
        mul     CX      ;Умножим CX на AX
        mov     CX,AX   ;Перешлем результат в CX
        pop     AX      ;Восстановим значение AX и
        pop     DX      ;DX из вызывающей программы
outer:  push    CX      ;Далее как и раньше
        mov     CX,0
inner:  loop    inner
        pop     CX
        loop    outer
        ret             ;Возврат в вызывающую процедуру
delay   endp            ;Конец процедуры
;Подпрограмма outstring вывода строки на экран
;На входе CX=число выводимых байт, DX=адрес строки
outstring  proc near
        mov     AH,40
        mov     BX,1
        int     21h
        ret
outstring  endp
;Начало главной процедуры
myproc  proc
        . . .
        mov     CX,meslen     ;Настроим CX
        mov     DX,offset mes ;Настроим DX
        call    outstring     ;Вызов процедуры вывода строки
        mov     CX,10         ;Настроим CX
        call    delay         ;Вызов процедуры задержки
        mov     CX,meslen-6   ;Еще раз выведем строку
        mov     DX,offset mes+3 ;(измененную)
        call    outstring
;Поля данных
mes     db      '*** Сообщение ***',10,13
meslen  equ     $-mes
```

Задача 9. Оформить подпрограммы из предыдущего примера в виде отдельных исходных файлов. Оттранслировать их и получить объектные модули. С помощью программы-библиотекаря TLIB.EXE создать объектную библиотеку пользователя и включить в нее объектные модули процедур-подпрограмм. Написать главную программу, содержащую вызовы процедур из объектной библиотеки. Скомпоновать главную программу с модулями из объектной библиотеки и проверить ее работоспособность.

Лабораторная работа № 3

Задачи по программированию операций над файлами, каталогами и дисками [30]

Задача 1. Создание файла. В директории ASM диска создать файл с MYFILE.001 и записать в него символьную строку. После выполнения программы убедится в его наличии и содержимом.

```
;Определения
cr      equ     0Dh
lf      equ     0Ah
text    segment 'code'
        assume  CS:text,DS:data
myproc  proc
        mov     AX,data
        mov     DS,AX
;Создание файла
        mov     AH,3Ch              ;Функция создания файла
        mov     CX,0                ;Без атрибутов
        mov     DX,offset filename  ;Адрес имени файла
        int     21h
        mov     handle,AX           ;Сохраним дескриптор файла
;Запись строки в файл
        mov     AH,40h              ;Функция закрытия
        mov     BX,handle           ;Дескриптор
        mov     CX,stringln         ;Длина строки
        mov     DX,offset string    ;Адрес строки
        int     21h
;Закроем файл (больше его не используем)
        mov     AH,3Eh              ;Функция закрытия
        mov     BX,handle           ;Дескриптор
        int     21h
;Завершим программу
outprog: mov    AX,4C00h        ;Функция завершения, код за-
                                ;вершения = 0
        int     21h
myproc  endp
text    ends
data    segment
string  db      'Текстовая строка',cr,lf ;Строка для записи в
                                         ;файл
stringln equ    $-string            ;Ее длина
handle  dw      ?                   ;Ячейка для дескриптора
filname db      'MYFILE.001',0      ;Имя файла в формате ASCIIZ
data    ends
```

```
stack   segment para stack 'STACK'
        db      128 dup (?)
stack   ends
        end     myproc
```

Задача 2. Чтение файла. Прочитать содержимое файла MYFILE.001 в память и вывести его на экран. Предполагается, что размер файла не более 80 байт.

```
;Определение
stdout  equ     1               ;Дескриптор стандарт. вывода
;Основные фрагменты программы
;Открываем файл
        mov     AH,3Dh          ;Функция открытия файла
        mov     AL,2            ;Доступ для чтения/записи
        mov     DX,offset string ;Адрес имени файла
        int     21h
        mov     handle,AX       ;Получение дескриптора
;Попытка чтения 80 байт
        mov     AH,3Fh          ;Функция чтения
        mov     BX,handle       ;Дескриптор
        mov     CX,80           ;Столько читать
        mov     DX,offset bufin  ;сюда
        int     21h
        mov     CX,AX           ;Столько реально прочитали
;Вывод прочитанного на экран
        mov     AH,40h          ;Функция записи
        mov     BX,stdout       ;Дескриптор
        mov     DX,offset bufin  ;Отсюда выводить CX байт
        int     21h
;Завершение программы
        ...
;Поля данных
bufin   db      80 dup (' ')    ;Буфер ввода
handle  dw      ?               ;Ячейка для дескриптора
filename db     'MYFILE.001',0   ;Имя файла в формате ASCIIZ
```

Задача 3. Прямой доступ к файлу. Прочитать 8 байт из созданного ранее файла MYFILE.001, НАЧИНАЯ с байта 5, вывести их на экран.

```
;Определение
stdout  equ     1
;Основные фрагменты программы
;Открыть файл
        ...
```

```
;Установить указатель
        mov     AH,42h
        mov     AL,0
        mov     BX,handle
        mov     CX,0
        mov     DX,5
        int     21h
;Прчитать 8 байт данных с помощью функции 3Fh
        ...
;Выведем прочитанное на экран с помощью функции 40h
        ...
;Завершить программу
        ...
;Поля данных
bufin   db      80 dup ('*')     ;Буфер ввода
handle  dw      ?                ;Ячейка для дескриптора
filename db     'MYFILE.001',0   ;Имя файла в формате ASCIIZ
```

Задача 4. Добавление данных к файлу. Добавить символную строку к концу символьного файла. Проконтролировать содержимое и длину файла до и после добавления (средствами NC). Повторить выполнение программы и еще раз проконтролировать результат.

```
;Основные фрагменты программы
;Откроем файл с указанным именем функцией 3Dh и сохраним полу-
;ченый дескриптор в ячейке handle
        ...
;Установим указатель на конец файла
        mov     AH,42h          ;Функция установки указателя
        mov     AL,02           ;От конца файла
        mov     BX,handle       ;Дескриптор
        mov     CX,0            ;Старшая половина указателя
        mov     DX,0            ;Младшая половина указателя
        int     21h
;Допишем новую строку
        mov     AH,40h          ;Функция записи
        mov     BX,handle       ;Дескриптор
        mov     CX,stringln     ;Длина строки
        mov     DX,offset string ;Ее адрес
        int     21h
;Завершим программу
        ...
;Поля данных
string  db      'Новая строка'   ;Строка для вывода в файл
stringln equ    $-string         ;Длина строки
```

```
handle  dw      ?               ;Ячейка для дескриптора
filename db     'MYFILE.001',0   ;Имя файла
```

Задача 5. Изменение атрибутов файла. Установить у файла MYFILE.001 атрибут "только для чтения". После завершения программы проконтролировать результат с помощью средств NC (DOS).

```
;Основные фрагменты программы
;Установим атрибут "только для чтения"
        mov     AH, 43h         ;Функция работы с атрибутами
        mov     AL,1            ;Установка атрибутов
        mov     CX, 1           ;"Только для чтения"
        mov     DX,offset filename ;Адрес имени файла
        int     2lh
```

Задача 6. Изменение характеристик файла. Изменить дату и время создания файла MYFILE.001.

```
;Основные фрагменты программы
;Откроем файл и сохраним его дескриптор
            ...
;Изменим дату и время создания файла
        mov     AH, 57h    ;Функция даты/времени
        mov     AL,1       ;Установить дату/время
        mov     BX, handle ;Дескриптор файла
        mov     CX,0       ;Очистим CX
        or      CX,sec     ;Добавим секунды
        or      CX,min     ;Добавим минуты
        or      CX,hour    ;Добавим часы
        xor     DX,DX      ;Очистим DX
        or      DX,day     ;Добавим день
        or      DX,mon     ;Добавим месяц
        or      DX, year   ;Добавим год
        int     21h
;Завершим программу
;Поля данных
fname   db      'MYFILE.001',0 ;Имя файла
handle  dw      ?       ;Ячейка для дескриптора
sec     dw      6/2     ;6 секунд
min     dw      15*32   ;15 минут
hour    dw      16*2048 ;16 часов
day     dw      25      ;25 число
mon     dw      3*32    ;Март
year    dw      13*512  ;13 лет от 1980, т.е. 1993 г.
```

Задача 7. Переименование файла. Переименовать файл MYFILE. 001, находящийся в текущем каталоге, дав ему имя NEWNAME. DAT.

```
;Основные фрагменты программы
;Настроим сегментный регистр ES на наш сегмент данных
        push    DS
        pop     ES
;Переименуем файл
        mov     AH,56h          ;Функция переименования
        mov     DX,offset oldname ;Адрес старого имени
        mov     DI,offset newname ;Адрес нового имени
        int     21h
;Завершить программу
        ...
;Поля данных
oldname db      'MYFALE.001',0    ;Старая спецификация
newname db      'NEWNAME.DAT',0   ;Новая спецификация
```

Задача 8. Пересылка файла в другой каталог на том же диске
Переслать файл NEWNAME.DAT из текущего каталога в нижележащий каталог NEWDIR, изменив при этом имя файла на NEWNAME.LEX. Отладив программу, проверить ее возможности, помещая файл в разные каталоги и пересылая его в другие.

```
;Основные фрагменты программы
;Настроим сегментный регистр ES на наш сегмент данных
        ...
;Переименуем файл
        mov     AH,56h          ;Функция переименования
        mov     DX,offset oldname ;Адрес старого имени
        mov     DI,offset newname ;Адрес нового имени
        int     21h
;Завершить программу
        ...
;Поля данных
oldname db      'NEWNAME.DAT',0   ;Старая спецификация
newname db      'NEWDIR\'NEWNAME.LEX',0   ;Новая спецификация
```

Задача 9. Создание и удаление каталогов. Создать в текущем каталоге подкаталог NEWDIR. Удалить из текущего каталога подкаталог OLDDIR.

```
;Основные фрагменты программы
;Создадим новый каталог
        mov     AH,39h          ;Функция создания каталога
        mov     DX,offset newname ;Адрес нового имени
        int     21h
;Удалим ненужный каталог
        mov     AH,3Ah          ;Функция удаления каталога
        mov     DX,offset oldname ;Адрес имени каталога
```

```
        int     21h
;Завершить программу
        ...
;Поля данных
oldname db      'OLDDIR'    ;Старая спецификация
newname db      'NEWDIR'    ;Новая спецификация
```

Задача 10. Смена диска. Сделать текущим диск A: .

```
;Сделаем текущим диск A:
        mov     AH,0Eh      ;Функция смены диска
        mov     DL,0        ;Код диска A (A=0,B=1,C=2 и т.д.)
        int     21h
;Завершить программу
        ...
```

Задача 11. Создание метки тома. Создать в корневом каталоге дискеты запись с меткой тома PROGRAMDISK. После выполнения программы проверить наличие метки командами DOS: DIR или VOL.

```
;Основные фрагменты программы
;Создадим "файл" - метку тома
        mov     AH,3Ch      ;Функция создания файла
        mov     CX,8        ;Атрибут метки тома
        lea     DX,labela   ;Адрес имени метки
        int     21h
;Завершим программу
;Поля данных
labela  db      'A:\PROGRAMD.ISK',0
```

Задача 12. Проверка метки тома. Найти в корневом каталоге дискеты запись с меткой тома и убедиться, что на дисковод установлена требуемая дискета.

```
;Основные фрагменты программы
;Установим нашу область передачи диска (DTA), чтобы не
;искать ее в префиксе программного сегмента
        mov     AH,1Ah      ;Функция установки DTA
        mov     DX,offset dta ;Адрес нашей DTA
        int     21h
;НАЙДЕМ МЕТКУ
        mov     AH,4Eh      ;Функция поиска первого файла
        mov     CX,8        ;Атрибут "метки тома"
        mov     DX,offset dname ;Адрес спецификации файла
        int     21h
        jc      nolabel         ;Метка отсутствует
;Сравним метки
        mov     SI,offset lbl   ;Адрес имени требуемой метки
```

```
        mov     AX,seg_dta      ;Настройка сегментного регист-
        mov     ES,AX           ;ра ES на наш сегмент
        mov     DI,offset dta+1Eh ;Место для имени файла в DTA
        mov     CX,lbllen       ;Длина имени метки
        cld                     ;Будем сравнивать в перед
repe    cmpsb                   ;Сравнение
        jne     error           ;Метки не совпадают
;Вывести сообщение mes1 о том, что на дисководе стоит требуе-
;мая дискета
        ...
        jmp     outprog
nolabel:
;Вывести сообщение mes2 об отсутствии метки на дискете
        ...
        jmp     outprog
error:
;Вывести сообщение о том, что на дисководе не та дискета
        ...
;Завершить программу
        ...
;Поля данных
dname   db      'A:\*.*',0 ;
dta     db      43 dup(0)
lbl     db      'PROGRAMD.ISK'
lbllen  equ     $-lbl
mes1    db      'На дисководе стоит правильная дискета',cr,lf
mes1len equ     $-mes1
mes2    db      'На дискете НЕТ метки!',cr,lf
mes2len equ     $-mes2
mes3    db      'На дисководе НЕ ТА дискета!',cr,lf
mes3len equ     $-mes3
```

Лабораторная работа № 4

Задачи по защите программ от копирования и несанкционированного использования [30]

Задача 1:

Защита программ от копирования путем привязки к ее местоположению на жестком диске.

Написать рабочую программу WORK1.COM, выводящую на экран некоторый текст. В начале выполнения программа орпеделяет номер своего первого кластера и сравнивает его с числом, записанным в ее полях данных в процессе установки на диске данной копии программы. Если эти числа несовпадают, программа аварийно завершается с выдачей соответствующего сообщения.

Написать установочную программу INSTALL1.EXE, которая определяет начальный кластер файла рабочей программы WORK1.COM и записывает его в поле данных рабочей программы.

Вызвать рабочую программу WORK1.COM, убедиться в том, что в неустановленном состоянии она аварийно завершается.

Установить конкретную копию программы WORK1.COM с помощью программы INSTALL1.EXE. Убедиться, что она установленная рабочая программа функционирует нормально.

Скопировать рабочую программу в другой каталог. С помощью команд DOS COMP или FC убедиться в полной тождественности двух копий программ WORK1.COM. Проверить работу "незаконно скопированной" (неустановленной) и исходной, установленной копий программы WORK1.COM. Установить вторую копию рабочей программы и проверить ее работу.

Программа WORK1.COM

```
;Рабочая программа. Перед запуском требует установки
;на жестком диске
;Основные фрагменты программы
text   segment 'code'
       assume cs:text,ds:text
       org 100h
myproc     proc
           jmp entry         ;Обойдем слово для ключа
cluster    dw   0            ;Слово для ключа
entry:
;откроем файл WORK1.COM с рабочей программой, чтобы он
;попал в таблицу файлов
       mov  AH,3Dh           ;Функция открытия файла
       mov  AL,2             ;Доступ для чтения/записи
       lea  DX,fname         ;Поле с именем файла
       int  21h
       jnc  go1              ;Файл открылся нормально
       jmp  notopen          ;Файл не открылся
;получим доступ к таблице файлов
```

```
go1:   mov  AX,5200h        ;Не документированная функция
       int  21h
       mov  DI,ES:[BX+4]    ;Адрес первой таблицы файлов
       mov  AX,ES:[BX+6]    ;Сегмент первой таблицы файлов
       mov  ES,AX           ;ES:DI -> первую таблицу файлов
;обрабатываем таблицы файлов по очереди
;сначала сохраним адрес следующей таблицы
next:  mov  AX,ES:[DI]      ;Смещение к следующей таблице
                            ;или FFFFh для последней
       mov  word ptr next_blk,AX  ;Сохраним его
       mov  AX,ES:[DI+2]    ;Сегмент следующей таблицы
       mov  word ptr next_blk+2,AX ;Сохраним его
;и организуем поиск имени нашего файла в текущей таблице
       mov  CX,ES:[DI+4]    ;Количество блоков управления
                            ;файлами в этой таблице
       add  DI,6            ;ES:DI -> первый блок управления файлами
cmpfile:
       lea  SI,workcom      ;Адрес имени файла
       call compname        ;Сравним имена
       je   found           ;Нашли
       add  DI,59           ;Перейдем к следующему блоку
       loop cmpfile         ;Цикл по блокам этой таблицы
       cmp  word ptr next_blk,0FFFFh ;Конец таблицы
       jne  go2             ;Нет, перейдем к следующей таблице
       jmp  notfnd          ;Таблицы кончились, а нашего файла нет
;настроим ES:DI на следующую таблицу файлов
go2:   mov  DI,word ptr next_blk
       mov  AX,word ptr next_blk+2
       mov  ES,AX
       jmp  next            ;Будем повторять поиск имени нашего файла
;наконец наш файл найден
found: add  DI,11           ;DI -> номер первого кластера
       mov  AX,cluster      ;Получим ключ
       cmp  AX,ES:[DI]      ;Сравним с истинным номером
       jne  notins          ;Ключи не совпадают
;ключи совпадают, программа установлена и с ней можно работать
;выведем сообщение mes (функция 40h, int 21h)
       ;Далее должна идти содержательная часть рабочей программы.
       ;В данной задаче ее может и не быть
       . . .

;заверши мпрограмму обычным образом
       . . .
notins:
;вывод сообщения mes1 о том, что файл не открылся
       . . .
       jmp  outprog
notopen:
```

```
;вывод сообщения mes2 о том, что файл не открылся
      . . .
      jmp  outprog
notfnd:
;вывод сообщения mes3 о том, что файл рабочей программы
;не найден в таблице файлов
      . . .
      jmp  outprog
;Подпрограмма сравнения имен файлов.
;На входе :
;DS:DI -> имя в программе
;ES:DI -> начало блока управления  управления файлами
;на выходе:
;при равенстве имен ZF=1
;при неравенстве ZF=0
compname:
      push  CX       ;Сохраним регистры, которые
      push  DI       ;понадобятся в подпрограмме
      add   DI,32    ;DI -> имя файла
      mov   CX,11    ;Длина имени файла
      cld            ;Сравнение вперед
repe   cmpsb
      pop   DI       ;Восстановим
      pop   CX       ;регистры
      ret            ;Возврат в основную программу
myproc endp
;Поля данных
next_blk  dd 0
fname     db 'work1.com',0 ;Имя файла в формате файловых функций
workcom   db 'WORK1   COM' ;Имя файла, как оно записано таблице файлов
mes       db 'Программа установлена и будет работать нормально',10,13
meslen    equ $-mes
mes1      db 'Программа не установлена и не может быть запущена',10,13
mes1len   equ $-mes1
mes2      db 'Файл WORK1.COM не открылся',10,13
mes2len   equ $-mes2
mes3      db 'Файл WORK1.COM не найден в таблице файлов',10,13
mes3len   equ $-mes3
text      ends
         ends myproc
```

Программа INSTALL1.EXE

```
;Установочная программа, предназначенная для установки
;на жестком диске рабочей программы WORK1.COM
;Основные фрагменты программы
;Откроем наш файл, чтобы он попал в таблицу файлов
       ...
;Получим доступ к таблице файлов
```

```
        ...
;Обрабатываем таблицы файлов по очереди
;Сначала сохраним адрес следующей таблицы
        ...
;И организуем поиск имени нашего файла в текущей таблице
        ...
;Найдем имя нашего файла в таблице файлов
        ...
;Настроим ES:DI на следующую таблицу файлов
        ...
;Файл найден
found:  add     DI,11         ;DI -> номер первого кластера
        mov     AX,ES:[DI]    ;Получим его
        mov     startclst,AX  ;и отправим в startclst
;У нас есть номер первого кластера файла WORK1.COM, но нет
;соответсвующего номера сектора - ни логического, ни физического.
;Прочитаем загрузочную запись нашего диска в буфер sysarea,
;чтобы определить характеристики диска:
;число FAT, размер корневого каталога, число секторов в кластере
        mov     AL,4          ;Дисковод E:
        mov     CX,1          ;Число читаемых секторов
        mov     DX,0          ;Логический номер сектора
        lea     BX,sysarea    ;Буфер для чтения сектора
        int     25h           ;Абсолютное чтение с диска
        pop     AX            ;Подправим стек после int 25h
;Найдем относительный номер начального сектора нашего файла
        mov     accum,1       ;Пропустим загрузочный сектор
        mov     AL,sysarea+10h ;Число FAT
        xor     AH,AH         ;Будет умножение слов
        mul     word ptr sysarea+16h ;умножим на число секторов в FAT
        add     accum,AX      ;Добавим в аккумулятор
        mov     ax,word ptr sysarea+11h  ;Число записей в корневом
                              ;каталоге
        xor     DX,DX         ;Будет деление слов
        mov     BX,16         ;Число записей на сектор
        div     BX            ;AX=число секторов в корневом каталоге
        add     accum,AX      ;Получили полный размер системной
                              ;области в секторах
        mov     AX,startclst  ;Первый кластер нашего файла
        sub     AX,2          ;Нумерация кластеров с 2
        mov     BL,sysarea+0dh ;Число секторов в кластере
        xor     BH,BH         ;Будет умножение слов
        mul     BX            ;Число секторов от системной
                              ;области до нашего файла
        add     accum,AX      ;Получили относительный номер
                              ;первого сектора файла
;Запишем номер начального кластера из startclst в ячейку
;cluster файла с программой WORK1.COM. Сначала прочитаем
```

```
;первый сектор WORK1.COM по его относительному номеру
      mov   AL,4          ;Дисковод E:
      mov   CX,1          ;Число читаемых снкторов
      mov   DX,accum      ;Логический номер сектора
      lea   BX,sysarea    ;Воспользуемся тем же буфером
      int   25h           ;Аьсолютное чтение с диска
      pop   AX            ;Подправим стек после Int 25h
;Запишем в ячейку cluster номер кластера
      mov   AX,startclst
      mov   word ptr sysarea+3,AX;3 байта занимает код команды
                          ;jmp entry в программе WORK1.COM
;Перешлем исправлунный сектор назад на диск
      mov   AL,4          ;Дисковод E:
      mov   CX,1          ;Число записываемых секторов
      mov   DX,accum      ;Логический номер сектора
      lea   BX,sysarea    ;ИЗ того же буфера
      int   26h           ;Аьсолютное запись на диск
      pop   AX            ;Подправим стек после Int 26h
;Выведем сообщение mes о нормальном завершении программы
          . . .
;Завершим программу обычным образом
          . . .
notopen:
;Вывод сообщения mes2 о том , что файл WORK1.COM не открылся
          . . .
      jmp outprog
notfnd:
;Вывод сообщения mes3 о том , что файл WORK1.COM не найден
;в таблице файлов
          . . .
      jmp outprog
;Подпрограмма сравнения имён файлов
          . . .
;Поля данных
next_blk dd 0
fname    db     'work1.com',0 ;Имя файла в формате файловых функций DOS
workcom  db     'WORK1.COM'   ;Имя файла КАК ОНО ЗАПИСАНО
                      ;в таблице файлов
sysarea  db     512 dup (0)  ;буфер для чтения сектора
accum    dw     0            ;ячейка - аккумулятор
startclst dw    0           ;начальный кластер файла
                      ;WORK1.COM
mes      db     'Программа WORK1.COM установлена на'
         db     'жёстком диске',10,13
meslen   equ    $-mes
mes2     db     'Файл WORK1.COM не открылся',10,13
mes2len  equ    $-mes2
mes3     db     'Файл WORK1.COM не найден в таблице файлов',10,13
```

```
mes3len   equ     $-mes3
```

Задача 2.

Защита программ от копирования путём записи ключа в свободное пространство кластера за пределами файла с программой.

Написать рабочую программу WORK2.EXE(.COM),выводящую на экран некоторый текст.В начале выполнения программа считывает последний кластер файла WORK2.EXE(.COM),находит первое слово за логическим концом файла , считывает из него ключ и сравнивает с ключом ,записанным впрограмме.Если ключи не совпадают,программа аварийно завершается с выдачей соответствующего сообщения.

Написать установочную программу INSTALL2.EXE,которая определяет последний кластер файла рабочей программы WORK2.EXE,считывает с диска этот кластер, записывает его за логическим концом файла заданный ключ и переносит модифицированный кластер на диск.

Программа WORK2.EXE

```
;Рабочая прог.Перед запуском требует установки
;на жёстком диске
;Основные фрагменты программы
;Откроем файл WORK2.EXE с рабочей прог.,чтобы
;он попал в таблицу файлов
        . . .
;Получем доступ к таблице файлов
        . . .
;Обрабатываем таблици файлов по очереди
;Сначало сохраним адрес следующей таблици
        . . .
;И организуем поиск имени нашего файла в текущей таблице
        . . .
;Настроим ES:DS на следующий блок
        . . .
;Прочитаем с помощью прерывания Int 25h загрузочную запись
;нашего диска в буфер 'sysarea',чтобы определить размер
;кластера на данном жестком диске
 found:
        . . .
 mov  AL,sysarea+0Dh  ;  Сохраним размер кластера
 mov  clustsize,Al    ;   в ячейке clustsize

;Определим число  подобных  кластеров в файле и число байт
;файла в последнем неполном кластере
;Сначала найдём длинну кластера в байтах
```

```
 mov  AL,clustsize     ;Число секторов в кластере
 xor  AL,AH            ;Будет умножение слов
 mov  BX,512           ;В секторе всегда 512 байт
 mul  BX               ;Результат умнажения на AX
 mov  clustbyte,AX     ;Сохраним его

;Теперь получим двухсловную длинну программы и поделим
;на размер кластера

 mov  AX,word ptr ES:[DI+17]    ;Младшая часть длинны
 mov  DX,word ptr ES:[DI+17]    ;Старшая часть длинны
 div  clustbyte                 ;Разделим на размер кластера
 mov  progsize,AX               ;Целое число кластеров в файле
 mov  endic,DX                  ;Остаток байт сверх целого
                                ;числа кластеров
 inc  progsize                  ;Число кластеров в файле
                                ;включая последний (неполный)
;Прочитаем последовательно все кластеры файла до последнего
  mov  CX,progsize              ;Число кластеров в файле
read:  push CX                  ;Временно сохраним его
  mov  AH,3Fh                   ;Обычная файловая функция чтения
  mov  BX,handle                ;Дескриптор открытого файла
                                ;WORK2.EXE
  mov  CX,clustbyte             ;Число байтов в кластере
  lea  DX,buffer                ;Буфер размером 8 секторов
                                ;(Лучше бы выделить динамически
  int  21h                      ;столько,сколько надо
  pop  CX                       ;Восстановим счётчик кластеров
  lop  read                     ;Повторим столько раз,сколько
                                ;кластеров в файле

;Теперь в buffer последний кластер, а в таблице файлов-его номер
;По номеру кластера найдём логический номер
;его последнего сектора
;Сначала определим размер системной области

  mov  accum,1                  ;Пропустим загрузочный сектор
  mov  AL,sysarea+10h           ;Число FAT
  xor  AH,AH                    ;Будет умножение слов
  mul  word ptr sysarea+16h     ;Умножим на число секторов в FAT
  add  accum,AX                 ;Добавим в аккомулятор
  mov  AX,word ptr sysarea+11h  ;Число записей в корневом
                                ;каталоге(\)
  xor  DX,DX                    ;Будет деление слов
  mov  BX,16                    ;Число записей на сектор
  div  BX                       ;AX=число секторов в каталоге \
  add  accum,AX                 ;Получился размер системной
                                ;облости в секторах
```

```
;Теперь прибавим число секторов в файле

  mov  AX,ES[DI]+50h          ;Текущий номер кластера
  sub  AX,2                  ;Нумерация кластеров с 2
  mov  BL,sysarea+0Dh         ;Число секторов кластере
  xor  BH,BH                  ;Будет умножение до нашего
  mul  BH                    ;кластера в области данных

  add  accum,AX               ;Логический сектор начала
                             ;последнего кластера файла
;Последний кластер уже прочитан только до конца файла,
;а ключ находится за его концом.Прочитаем кластер целиком

  mov  AL,3                  ;Дисковод Д:
  mov  CL,clustsize           ;число читаемых секторов
  xor  CH,CH
  mov  DX,accum               ;Логический номер сектора
  lea  BX.buffer              ;Буфер для чтения сектора
  int  25h                   ;Абсолютное чтение с диска
  pop  AX                    ;Подправим стек после INT 25h
;Прочитаем ключ из файла и сравним с ключем в програме
  mov  BX,endic               ;Число байтов до конца файла
  mov  AX,word ptr buffer[BX]    ;Прочитаем слово за
                             ;пределами файла
  cmp  AX,key                 ;Сравним с ключем в программе
  je   go3                   ;Равны!
  jmp  notins                 ;Не равны ,программа не установлена
;Программа устанолена можно с ней работать
;Выведем сообщение 'mes' об этом
 go3:
      ;Далее должна идти содержательная часть рабочей программы
      ;В данной задаче её может и не быть
                  . . .
  outprog:
;Завершим программу обычным образом
                  . . .
  notins:
;Вывод сообщения 'mes1' о том,что програма не установлена
                  . . .
  jmp  outprog
  notopen:
;Вывод сообщения 'mes2' о том,что файл не открылся
   jmp  outprog
  notfnd:
;Вывод сообщения 'mes3' о том,что файл не найден в таблице
;файлов
   jmp  outprog
;Подпрограмма сравнения имён файлов
```

```
                    . . .
;Поля данных
  key dw  1234h
next_blk  dd 0
fname     db   'work2.exe',0
workcon   db   'WORK2.EXE'
sysarea   db   512  dup (0)   ;Буфер для чтения сектора
accum     dw    0             ;Ячейка- аккомулятора
handle    dw    0   ;Ячейка для дескриптора файла WORK2.EXE
progsize  dw    0             ;Размер программы
endic     dw    0   ;Число байт файла в последнем кластере
clustsize db    0             ;Число секторов в кластере
clustbyte dw    0             ;Число байт в кластере
buffer    db    8*512 dup(0)
mes       db    'Программа установлена и будет работать нармально',10,13
meslen    equ    $-mes
mes1      db   'Программа не установлена',10,13
mes1len   equ    $-mes1
mes2      db   'Файл WORK2.EXE не открылся',10,13
mes2len   equ    $-mes2

mes3       db  'Файл WORK2.EXE не найден',10,13
mes3len    equ    $-mes3

;ПРОГРАММА INSTALL2.EXE

;Программа предназначенная для установки на жёстком диске
;рабочей программы WORK2.EXE
;Основные фрагменты программы
;Откроем файл WORK2.EXE с рабочей прог.,чтобы он
;попал в таблицу файлов
        . . .
;Получим доступ к таблице файлов
        . . .
;Обработаем таблицы файлов по очереди
;Сначала сохраним адрес следующей таблицы
        . . .
;И организуем поиск имени нашего файла в текушей таблице
        . . .
;Настроим ES:DI на следующий блок
        . . .
;Наконец наш файл найден
;Прочитаем с помощью прерывания Int 25h загрузочную запись
;нашего диска в буфер 'sysarea',чтобы определить размер
;кластера на данном жестком диске
 found:
            . . .
;Теперь получим двухсловную длинну программы и поделим
```

```
;на размер кластера
          . . .
;Для простоты не будем анализировать возможность того,что
;файл WORK2.EXE занемает целое число  кластеров без остатка
;Прочитаем последовательно все кластеры файла до последнего
          . . .
;Теперь в buffer последний кластер, а в таблице файлов-его номер
;По номеру кластера найдём логический номер
;его последнего сектора
;Сначала определим размер системной области
          . . .
;Теперь прибавим число секторов в файле
          . . .
;Последний кластер уже прочитан и находится в буфере buffer
;Запишем ключ сначала в буфер

  mov  BX,endic                ;Число байтов  файла в последнем
                          ;кластере
  cmp  AX,key                  ;Ключ
  mov  AX,word ptr buffer[BX]    ;Запишем ключ в буфер

;А затем и в файл
  mov  AL,3                   ;Дисковод Д:
  mov  CL,clustsize             ;число читаемых секторов
  xor  CH,CH
  mov  DX,accum                 ;Логический номер сектора
  lea  BX.buffer               ;Буфер для чтения сектора
  int  25h                    ;Абсолютное чтение с диска
  pop  AX                      ;Подправим стек после INT 25h

;Выведем сообщение 'mes' онармальной работе
                    . . .
   outprog:
;Завершим программу обычным образом
                    . . .
   notopen:
;Вывод сообщения 'mes2' о том,что файл не открылся
                    . . .
   jmp  outprog
   notfnd:
;Вывод сообщения 'mes3' о том,что файл не найден в таблице файлов
         . . .
    jmp  outprog
;Подпрограмма сравнения имён файлов
                     . . .
;Поля данных
next_blk  dd    0
fname     db   'work2.exe',0
```

```
workcon   db   'WORK2.EXE'
sysarea   db   512  dup (0)   ;Буфер для чтения сектора
accum     dw    0            ;Ячейка- аккомулятора
handle    dw    0   ;Ячейка для дескриптора файла WORK2.EXE
key       dw  1234h
progsize  dw    0            ;Размер программы
endic     dw    0   ;Число байт файла в последнем кластере
clustsize db    0            ;Число секторов в кластере
clustbyte dw    0            ;Число байт в кластере
buffer    db    8*512 dup(0)
mes       db    'Программа установлена на жёском диске',10,13
meslen    equ    $-mes
mes2      db   'Файл WORK2.EXE не открылся',10,13
mes2len   equ    $-mes2
mes3      db   'Файл WORK2.EXE не найден в таблице'файлов,10,13
mes3len   equ    $-mes3
```

Задача 3. Защита программ от копирования с помощью ключевой дискеты с нестандартным форматированием.

Написать программу, форматирующую дополнительную 41-ю дорожку дискеты (с номером 40) так, чтобы на ней были сектора нестандартного размера (например, не 512, а 256 байт). Записать в первый сектор этой нестандартной дорожки некоторый ключ.

Написать программу, которая перед выполнением считывает ключ с нестандартной дорожки и анализирует его правильность.

Программа INSTALL3.EXE

```
;Программа нестандартного форматирования дискеты и записи на нее
;ключа
;Основные фрагменты программы
;Установим тип дискеты (только для машин типа AT)
     mov   AH,17h
     mov   AL,02
     mov   DL,0
     int   13h
;Найдем и сохраним адрес таблицы параметров дискеты (вектор преры-
;вания 1Eh)
     mov   AX,351Eh
     int   21h
     mov   word ptr dpt,BX
     mov   word ptr dpt+2,ES
;Установим новый размер сектора 256 байт вместо 512
     mov   byte ptr ES:[DI+3],1
;Отформатируем дополнительнуб 41-ю дорожку
     mov   AH,05
     mov   CH,40
     mov   DH,0
     mov   DL,0
```

```
    push  DS
    pop   ES
    lea   BX,afd
    int   13h
;Записываем на дорожку ключ (размер min в целый сектор)
    mov   AH,03
    mov   AL,1
    mov   CH,40
    mov   CL,1
    mov   DH,0
    mov   DL,0
    lea   BX,key
    int   13h
;Восстановим таблицу параметров дискеты
go:   mov   ES,word ptr dpt+2,BX
    mov   byte ptr ES:[DI+3],2
;Поля данных
dpt   dd    0
afd   db    40,0,1,1
    db    40,0,2,1
    db    40,0,3,1
    db    40,0,4,1
    db    40,0,5,1
    db    40,0,6,1
    db    40,0,7,1
    db    40,0,8,1
    db    40,0,9,1
key   dw    9999h
    db    254 dup (0)
```

Программа WORK3.EXE

```
;Программа чтения нестандартно отформатированной дискеты и записи
;Основные фрагменты программы
;Установим тип дискеты - только для AT
    . . .
;Найдем и сохраним адрес таблицы параметров дискеты (вектор типа 1Eh)
    . . .
;Установим размер сектора нестандартной дорожки
;Прочитаем первый сектор 41-й дорожки
    mov     AH,02          Функция чтения сектора;
    mov     AL,1           ;Один сектор
    mov     CH,40          ;Цилиндр
    mov     CL,1           ;Сектор
    mov     DN,0           ;Головка
    mov     DL,0           ;Дисковод A:
    push    DS
    pop     ES
    lea     BX,buf         ;Адрес буфера в ES:BX
```

```
    int       13h
    jc        noform            ;Не читается
    cmp       word ptr buf,9999h  ;Сравним ключи
    jne       nokey             ;Неправильный ключ
;Выведем сообщение mes о нормальной работе
     . . .
;Завершим программу обычным образом, восстановив сначала
;таблицу параметров дискеты
outprog: mov     ES, word pptr dpt+2
        mov     byte ptr ES :[DI+3],2
noform:
;Введем сообщение mes2 о несоответствии форматов
     jmp outprog
nokey:
; Выведем сообщение mes3 о несоответствии ключа ожидаемому
     . . .
     jmp outprog
;Поля данных
dpt         dd       0
buf         db       256   dup (0)
mes         db       'Ключевая дискета установлена.',10,13
           db       'Программа может работать',10,13
mes  en     equ      $ -mes
mes2        db       'Установленна НЕ ключевая дискета',10,13
mes2 en     equ      $-mes2
mes3        db       'На дискете неправильный ключ',10,13
mes3 en     equ      $-mes3
```

Лабораторная работа № 5

Задачи по программированию ввода с клавиатуры [30]

Задача 5.1.

Ввод с клавиатуры и вывод на экран символьной информации.

Вывести на экран через дескриптор стандартной ошибки служебное сообщение. Ввести с клавиатуры через дескриптор стандартного ввода символьную строку, вывести её на экран через дескриптор стандартного вывода. Изучить действие операторов перенаправления ввода и вывода, выполнив следующие действия:

- перенаправить вывод программы во вновь создаваемый файл;
- перенаправить вывод программы в имеющийся файл с добавлением новой строки к содержимому файла;
- перенаправить ввод программы, получая данные для неё не с клавиатуры, а из файла;
- перенаправить и ввод и вывод программы, получая данные из одного файла и занося их в другой (в сущности, реализация копирования символьных файлов).

```
;определения
stdin  equ   0              ;дескриптор стандартного ввода
stdout equ   1              ;дескриптор стандартного вывода
stderr equ   2              ;дескриптор стандартной ошибки
;Выведем служебное сообщение msg
       ...
;Поставим запрос на ввод строки в буфер buf
       mov   AH,3Fh                ;Функция ввода
       mov   BX,stdin      ;Дескриптор стандартного ввода
       mov   CX,80         ;Ввод максимум 80 байт
       mov   DX,offset buf         ;Адрес буфера ввода
       int   21h
       mov   actlen,AX     ;Фактически введено
;Выведем на экран
       mov   AH,40h                ;Функция вывода
       mov   BX,stdout     ;Дескриптор стандартного вывода
       mov   CX,actlen     ;Длинна сообщения
       mov   DX,offset buf         ;Адрес сообщения
       int   21h
;Завершим программу
       ...
;Поля данных
msg    db    'Вводите! '
msglen         equ   $-msg          ;Длинна сообщения
buf    db    80 dup (0)    ;Буфер ввода
actlen dw    0
```

Задача 5.2.

Ввод с клавиатуры, обработка и вывод на экран символьной информации.

Усложнить программу из примера 5.1, предусмотрев фильтрацию входного символьного потока, например, преобразование строчных латинских (или русских, или и тех, и других) букв в прописные. Отладив программу, и используя её вместе с предидущей,изучить процедуру конвейеризации программ на примере следующих
операций (программа из примера 5.1 названа P51, из примера 5.2 - P52):

1) P51|P52
2) P51|P52>FILE1.TXT
3) P51<FILE2.TXT |P52
4) P51<FILE2.TXT |P52>FILE3.TXT

Файл FILE2.TXT, содержащий, в частности, строчные буквы, следует создать либо с помощью той же программы P51, либо с помощью любого техтового редактора.

```
;Определения stdin, stdout, stderr
        ...
;Основные фрагменты программы
;Выведем служебное сообщение msg
        ...
;Поставим запрос на ввод строки в буфер buf, отправив длинну
;фактичес введённой строки в actlen
        ...
;Превратим строчные латинские буквы в прописные
        mov   CX,actlen         ;Длинна введённой строки
        mov   SI,0              ;Указатель в буфере
filter: mov   AL,buf[SI]        ;Возмём символ
        cmp   AL,'a'            ;Меньше 'a'?
        jb    nolet             ;Да, не преобразовывать
        cmp   AL,'z'            ;Больше 'z'?
        ja    nolet             ;Да, не преобразовывать
        sub   AL,20h                    ;Преобразуем в прописную
        mov   buf[SI],AL        ;И отправим назад в buf
nolet:  inc   SI                ;Сместим указатель
        loop  filter            ;Цикл
;Выведем введённое на экран
        ...
;Завершим программу
        ...
;Поля данных
msg     db    'Войдите! '
msglen  equ   $-msg                     ;Длинна сообщения
buf     db    80 dup (0)        ;Буфер ввода
actlen  dw    0
```

Задача 5.3.

Посимвольный ввод с эхом.

Ввести в программу один символ с клавиатуры. Проанализировать его.Если введено <E>, завершить программу; в случае ввода любого другого символа повторить ввод. После отладки программы проверить её реакцию на ввод <Ctrl>/c. Исследовать работу программы в случае перенаправления её ввода или вывода в файл. Исследовать реакцию программы на ввод с клавиатуры <Ctrl>/c в случае перенаправления её ввода и вывода. Исследовать реакцию программы на ввод с упреждением, для чего сразу после ввода команды активизации программы, но ещё до её фактического запуска нажать на какие-то клавиши.

```
;Основные фрагменты программы
;Введём и проанализируем его на код <E>
;Если введено <E>, завершим программу
again: mov   AH,01h                ;Функция ввода с эхо
       int   21h
       cmp   AL,'E'            ;Введено <E>?
       je    outprog                ;Да
       jmp   again
outprog:
;Завершим программу
       ...
```

Задача 5.4.

Посимвольный ввод без эха.

Заменить в примере 5.3 функцию ввода с эхом 01h на функцию фильтрованного ввода
без эха 08h. После отладки проверить реакцию программы на ввод <Ctrl>/c. Для удобства работы предусмотреть в программе вывод на экран запроса на ввод символа.

```
;Основные фрагменты программы
;Выведем запрос req на ввод символа
       ...
;Выведем символ и проанализируем его на код <E>
;Если введено <E>, завершим программу
again: mov   AH,08h
       int   21h
       cmp   AL,'E'
       je    outprog
       jmp   again
outprog:
;Завершим программу
       ...
;Поля данных
req    db    'Введите команду: '
reqlen equ   $-req
```

Задача 5.5.

Управление программой от функциональных клавиш.

С помощью функции 08h организовать ввод в программу управляющих кодов от функциональных клавиш <F1>...<F10> или других клавиш, дающих расширенные коды
ASCII (сочетания <Alt>/<буква>, <Alt>/<цифра> и др.) Вводимые коды использовать
для управления ходом программы.

```
;Основные фрагменты программы
;Ожидаем нажатия клавиши
again: mov   AH,08h              ;Функиция ввода без эха
       int   21h
       cmp   AL,0          ;Младший байт кода = 0?
       jne   again         ;Нет, повторить
       mov   AH,08h              ;Да, введём старший байт кода
       int   21h
       cmp   AL,59         ;Нажата <F1>?
       je    f1            ;Да
       cmp   AL,84         ;Нажаты <Shift>/<F1>?
       je    shiftf1       ;Да
       cmp   AL,30         ;Нажаты <Alt>/<A>?
       je    alta          ;Да
       cmp   AL,120              ;Нажаты <Alt>/<1>?
       je    alt1          ;Да
       jmp   again         ;Нажато незапланированное
;Вывод соответствующих сообщений
f1:
;Вывод сообщения mf1
       ...
       jmp   outpr
shiftf1:
;Вывод сообщения mshf1
       ...
       jmp   outpr
alta:
;Вывод сообщения malta
       ...
       jmp   outpr
alt1:
;Вывод сообщения malt1
       ...
       jmp   outpr
outpr:
;Завершим программу
       ...
```

```
;Поля данных
mf1         db    'Введено <F1>'
mf1len      equ   $-mf1
mshf1       db    'Введено <Shift>/<F1>'
mshf1len    equ   $-mshf1
malta       db    'Введено <Alt>/A'
maltalen    equ   $-malta
malt1       db    'Введено <Alt>/1'
malt1len    equ   $-malt1
```

Задача 5.6

Ввод клавиатуры с предварительной очисткой буфера.

Организовать цикл ввода в программу данных по её запросу. Анализировать введённый символ. Если введено <Q>, завершить программу. После отладки проверить реакцию программы на ввод с упреждением. Проверить возможность управления программой кодами, вводимыми через цифровую клавиатуру (при нажатой клавише <Alt>).

```
;Основные фрагменты программы
;Выведем запрос req на ввод символа
        ...
;Очистим буфер ввода и поставим запрос на ввод с клавиатуры
again: mov   AH,0Ch              ;Функция ввода с отчисткой буфера
       mov   AL,01h              ;Выберем функцию ввода 01h
       int   21h           ;(можно 01h, 06h, 07h, 08h)
;Проанализируем введённое
       cmp   AL,'Q'        ;Введено <Q>?
       je    outprog             ;Да
       jmp   again
outprog:
;Завершим программу
       ...
;Поля данных
req    db    10,13,'Вводите команду:'
reqlen equ   $-req
```

Задача 5.7.

Чтение двухбайтового кода из кольцевого буфера ввода.

Программа, несмотря на примитивность, позволяет наглядно ознакомиться с принципами трансляции scan-кодов программой INT 09h. Для этого прграмму следует запустить в отладчике CodeView, установить режим индикации регистров просессора и, многократно выполняя строки программы, наблюдать, какие двухбайтовые коды образуются при нажатии различных клавиш и их комбинаций. Любопытно посмотреть коды таких клавиш, как <ESC>, <TAB>, <PgDn>, <Home>, <End>, клавиш со стрелками, а также сочетание управляющих (<Ctrl>, <Alt>, <Shift>) и <обычных> клавиш.

```
;Основные фрагменты программы
;Будем ждать ввода символа
again: mov   AH,00h                    ;Функция чтения двухбайтового кода
       int   16h
       jmp   again
```

Задача 5.8.

Управление циклической программой от клавиатуры с помощью функции чтения состояния клавиатуры.

Организовать выход из циклического участка программы при нажатии на любую клавишу. При желании можно усложнить задачу, введя в программу анализ нажатой клавиши и переход на то или иное продолжение программы в зависимости от
результатов анализа. В приведённом примере такой анализ отсутствует. Любопытно также удалить из программы строки извлечения символа из кольцевого буфера
(функция 00h) и проанализировать работу такого варианта программы.

```
;Определение
stdout equ   1              ;Дескриптор стандартного вывода
;Основные фрагменты программы
;Организуем цикл каких-нибудь действий с периодическим опросом
;клавиатуры
;Конкретные действия - вывод символа на экран с небольшой задержкой
again: mov   AH,40h             ;Функция вывода
       mov   BX,stdout    ;Дескриптор стандартного вывода
       mov   CX,1         ;Один символ
       mov   DX,offset sym      ;Адрес символа
       int   21h
;Организуем небольшую задержку
       ...
;Получим состояние клавиатуры
       mov   AH,01h             ;Функция чтения состояния
       int   16h          ;клавиатуры
       jz    again        ;Если Z=1, символа нет
;Сообщим о выходе из цикла с помощью функции 40h
       ...
;Заберём введённый символ из кольцевого буфера ввода
       mov   AH,00h             ;Получим символ в AX
       int   16h          ;и пусть он пропадает
;Завершим программу
       ...
;Поля данных
sym    db    '*'
mes    db    'Программа завершена оператором'
```

КАРТА ОБЕСПЕЧЕННОСТИ ДИСЦИПЛИНЫ КАДРАМИ ПРОФЕССОРСКО-ПРЕПОДАВАТЕЛЬСКОГО СОСТАВА

Виды учебных занятий	ФИО преподавателя, ученая степень, должность
Лекции	Шевко Денис Геннадьевич, к.т.н., доцент кафедры ИиУС
Лабораторные работы	Шевко Денис Геннадьевич, к.т.н., доцент кафедры ИиУС

СОДЕРЖАНИЕ

Денис Геннадьевич ШЕВКО

доцент кафедры Информационных и управляющих систем АмГУ, кандидат технических наук

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ

«Автоматизированные комплексы информационных систем»

для направления подготовки 230100.68 «Информатика и вычислительная техника»

Издательство АмГУ