МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тульский государственный университет»

Институт высокоточных систем им. В.П. Грязева

Кафедра систем автоматического управления

Утверждено на заседании кафедры
«Системы автоматического управления»
«13» января 2021 г., протокол № 8

Заведующий кафедрой

_____О.В.Горячев

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по выполнению лабораторных работ
по дисциплине (модулю)

«Вычислительные машины, системы и сети»

основной профессиональной образовательной программы
высшего образования – программы специалитета

по специальности
24.05.06 Системы управления летательными аппаратами

со специализацией
Системы управления беспилотными летательными аппаратами

Форма обучения: очная

Идентификационный номер образовательной программы: 240506-01-21

Тула 2021 год

Разработчик(и) методических указаний

Морозов О.О., доцент, к.т.н. _____

(ФИО, должность, ученая степень, ученое звание) (подпись)

Лабораторная работа № 1. Изучение конструкции и составных элементов современных микропроцессорных систем (МПС).

I. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Изучение аппаратных средств современных микропроцессорных систем (МПС) на примере IBM- совместимого персонального компьютера. Рассмотрение основных электронных модулей, входящих в состав системного блока. Знакомство с основными слотами расширения персонального компьютера, изучение характеристик, соответствующих типов шин расширения. Знакомство с материнской платой персонального компьютера, рассмотрение основных карт расширения. Подключение к системной плате питания и кабелей ввода/вывода.

II. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1 Основные конструктивные компоненты персонального компьютера.

В классическом варианте исполнения персональный компьютер (ПК) конструктивно включает в себя три основных компонента: системный блок, монитор и клавиатура.

В **системном блоке** находится вся электронная начинка ПК: системная (материнская) плата (СП) (System Board или Mother Board) с установленными на ней центральными компонентами компьютера – процессором, оперативной памятью, вспомогательными схемами и щелевыми разъемами слотами, в которые можно устанавливать платы расширения. В корпусе системного блока имеются отсеки (Bay) для установки дисковых накопителей (дисководы жестких и гибких дисков, CD-ROM- накопитель на компакт диске), а также блок питания. Компоновка традиционного системного блока показана на рис. 1.

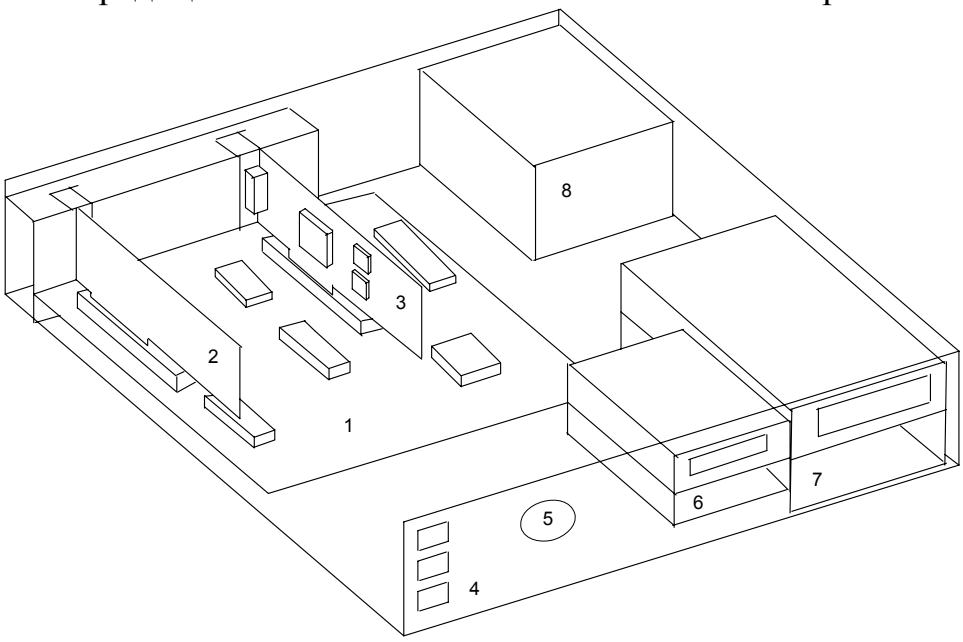


Рис.1.

На рис.1 обозначены: 1- системная плата; 2- карта расширения ISA; 3- карта расширения PCI; 4- элементы лицевой панели; 5 – динамик; 6,7- отсеки 3” и 5”; 8- блок питания.

Клавиатура и **монитор** относятся к периферийным устройствам, которые связывают ПК с внешним миром и позволяют соответственно вводить и отображать обрабатываемую информацию.

Системная, или материнская, плата персонального компьютера (System board или Mother board) является основой системного блока, определяющей архитектуру и производительность компьютера. На ней устанавливаются следующие обязательные компоненты:

1. Процессор(ы) и сопроцессор.
2. Память: постоянная (ROM или Flash BIOS), оперативная (DRAM), кэш (SRAM).
3. Обязательные системные средства ввода/вывода.
4. Интерфейсные схемы и разъемы шин расширения.
5. Кварцевый генератор синхронизации со схемой формирования сброса системы по сигналу PowerGood от блока питания или кнопки RESET.
6. Дополнительные стабилизаторы напряжения питания для **низковольтных** процессоров VRM (Voltage Regulation Module).

Кроме этих сугубо обязательных средств, на большинстве системных плат устанавливают и контроллеры интерфейсов для подключения гибких и жестких дисков (IDE, SCSI), графический адаптер, аудиоканал, а также адаптеры COM-и LPT-портов, «мыши» и другие. Контроллеры, требующие интенсивного обмена данными (IDE, SCSI, графический адаптер), используют преимущества локального подключения к шине процессора. Цель размещения других контроллеров на системной плате — сокращение общего числа плат компьютера.

2.2. Шинная архитектура ПК

Все электронные компоненты и модули расположенные на системной плате связываются между собой при помощи магистрали, включающей в себя набор системных и локальных шин адреса, данных, управления. На рис.2 схематически изображена коммутация основных компонентов системной платы.

Системные платы первых PC, выполненных на процессорах 8088/86, кроме процессора содержали несколько периферийных БИС (контроллеры прерываний, прямого доступа к памяти, контроллер шины) и связующую логику на микросхемах малой и средней степени интеграции. Современные платы исполняются на основе *чипсетов* (Chipset) — наборов из нескольких БИС, реализующих все необходимые функции связи основных компонентов — процессора, памяти и шин расширения. Чипсет определяет возможности применения различных типов процессоров,

основной и кэш-памяти и ряд других характеристик системы, определяющих возможности ее модернизации.

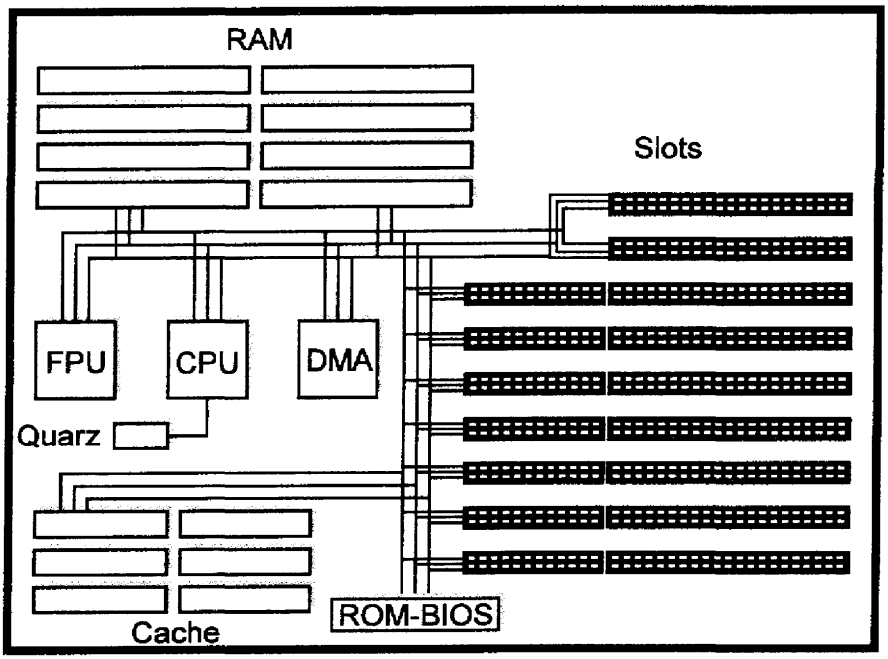


Рис.2.

Тип чипсета существенно влияет и на производительность — при одинаковых установленных компонентах (процессор, память, графический адаптер и жесткий диск) производительность компьютеров, собранных на разных системных платах (чипсетах), — может отличаться на 30%.

2.3. Типы системных плат.

Унификация и стандартизация компонентов PC распространяется на системные платы, предназначенные для установки в корпуса обычного исполнения. Некоторые "фирменные" платы имеют специфические габаритные и присоединительные размеры, и их можно устанавливать только в «родные» корпуса. Таким специфическим конструктивом отличаются, например, платы и корпуса компьютеров IBM PS/2, Acer, Compaq, Digital, Packard Bell и ряд других. К ним, естественно, некоторые последующие тезисы неприменимы. Здесь будут рассмотрены конструктивы системных плат, предназначенных для установки в корпуса машин класса AT. Системные платы класса XT интереса уже почти не представляют, а использование корпуса XT для компьютера AT в принципе возможно, но вызывает массу неудобств и ограничений.

Традиционные платы *Full AT* (305x350 мм) и Baby AT (220x330 мм) для корпусов *Desktop, Baby AT, Mini-*, *Midi-* и *Big Tower* имеют фиксированное расположение слотов и разъема клавиатуры относительно задней кромки платы и унифицированную систему крепежных отверстий платы, что облегчает ремонтопригодность и модернизируемость системных блоков. Их примерный вид изображен на рис.3.

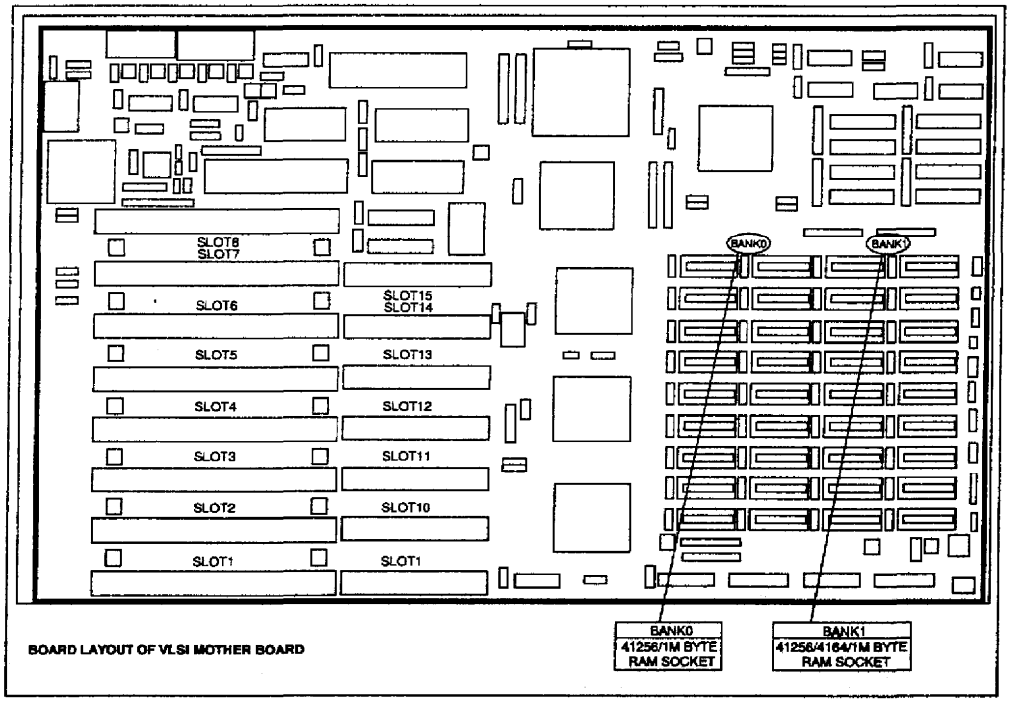


Рис.3.а.

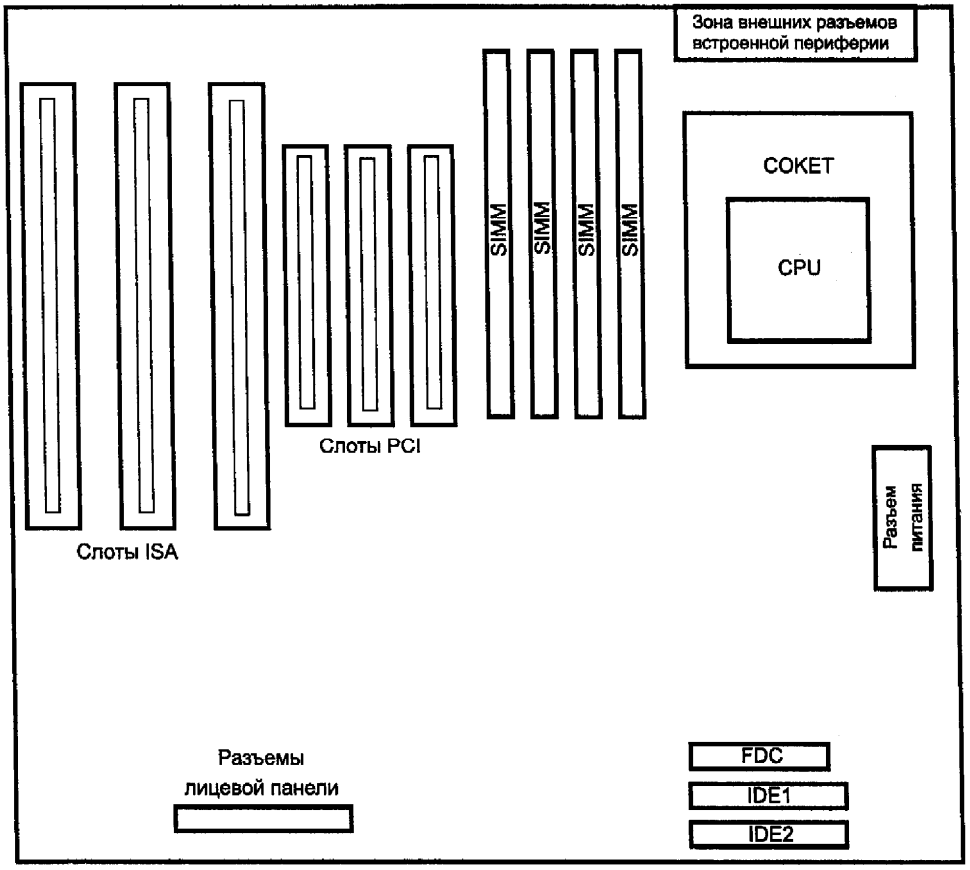


Рис.3.б

На рис.3, изображены системные платы AT, построенных на базе МП 80386 (а) и системные платы ATX, построенных на базе МП Pentium.

2.4. Шины расширения ввода/вывода.

Стандартизованные шины расширения ввода/вывода обеспечивают основу функциональной расширяемости PC-совместимого персонального компьютера, который с самого рождения не замыкался на выполнении сугубо вычислительных задач. Хотя многие компоненты, ранее размещаемые на платах расширения, постепенно переселяются на системную плату, для настольных компьютеров набор шин расширения ввода/вывода имеет принципиальное значение. Практически они реализуются в виде слотов на системной плате.

К шинам расширения ввода/вывода, относятся следующие:

1. ISA-8 и ISA-16 — традиционные универсальные слоты подключения периферийных адаптеров, не требующих высоких скоростей обмена (раньше была единственной шиной расширения).
2. EISA — дорогая (по стоимости и системной платы, и плат расширения) 32-битная шина средней производительности, применяемая в основном для подключения контроллеров дисков и адаптеров локальных сетей в серверах. В настоящее время вытесняется шиной PCI, хотя и применяется в серверных платформах, где необходимо установить множество дополнительных плат расширения (системную плату, у которой слотов PCI больше, чем 4, найти довольно трудно, а для шины EISA 6-8 слотов — явление обычное).
3. MCA — шина компьютеров PS/2, до сих пор применяемая и в некоторых серверных платформах. Производительность средняя. Адаптеры для шины MCA распространены не широко.
4. VLB — быстродействующее 32- (64-) битное расширение (локальная шина процессора), используемое в паре со слотом ISA/EISA, применявшееся в среднем поколении системных плат компьютеров на процессоре 486. Используется для подключения контроллеров дисков, графических адаптеров и контроллеров локальных сетей. С процессорами пятого поколения и старше не применяется.
5. PCI — самая распространенная высокопроизводительная 32/64-битная шина, применяемая в компьютерах на процессорах 486 и старше. Используется для подключения адаптеров дисков, контроллеров SCSI, графических, видео-, коммуникационных и других адаптеров. На системной плате чаще всего устанавливают 3 или 4 слота PCI. Слот PCI иногда имеет дополнительный маленький слот расширения Media BUS, на который выведены сигналы шины ISA (это позволяет на платы PCI устанавливать и дешевые ISA-устройства, например звуковой канал).
6. PC Card, он же PCMCIA - слот расширения блокнотных компьютеров, который, в принципе, может присутствовать и в компьютерах настольного исполнения (встречать на практике не доводилось). Предназначен для обеспечения еще одного уровня совместимости блокнотных и настольных PC.

Кроме шин, реализованных щелевыми разъемами-слотами, имеется ряд шин, в которых устройства соединяются кабелями. К ним относятся следующие:

1. SCSI — интерфейсная шина системного уровня, предназначенная для подключения широкого спектра внутренних и внешних периферийных устройств, требующих высокой производительности обмена данными. Системная плата со встроенным SCSI-адаптером имеет разъем одного из типов, принятых для этого интерфейса, который со внутренними и внешними устройствами соединяется обычно ленточным кабелем-шлейфом.
2. USB — последовательная шина среднего быстродействия для подключения разнообразных внешних устройств, включая принтеры, сканеры, диски (разные), цифровые акустические системы и многие другие. Системная плата может иметь два или четыре порта USB, выведенных на 4-штырьковые разъемы. Внешние разъемы устанавливаются на задней или лицевой панели корпуса компьютера.
3. FireWire (он же IEEE 1394) — высокопроизводительная последовательная шина подключения внешних устройств, предназначенная в основном для подключения видеоаппаратуры. С помощью этой же шины возможно и объединение нескольких компьютеров в локальную сеть. Системные платы с этой шиной пока еще редкость.

Кроме шин расширения, современные системные платы могут иметь и вспомогательные шины, используемые для тестирования и передачи конфигурационной информации. К ним относятся следующие:

1. JTAG — последовательный интерфейс тестирования, реализованный в большинстве процессоров старших поколений, а также входящий в спецификацию разъема шины PCI.
2. I2C — последовательная шина, используемая для передачи конфигурационной информации новых модулей памяти DIMM, а также в цифровом канале связи с монитором (DDC).

Карты расширения (интерфейсные карты) устанавливаются в соответствующие слоты системной платы. Их количество и состав на различных платах варьируется. Типы слотов легко определить визуально, в этом поможет рис. 4. На этом рисунке, конечно, присутствие всех типов шин показано условно — реально на системных платах присутствует не более двух-трех типов слотов. Распространенные сочетания: ISA+PCI, ISA+VLB, EISA+PCI, EISA+VLB. Шина MCA обычно держится особняком. Слот «Media BUS», дополняющий слот PCI сигналами шины ISA, применяется, пожалуй, только фирмой ASUSTek.

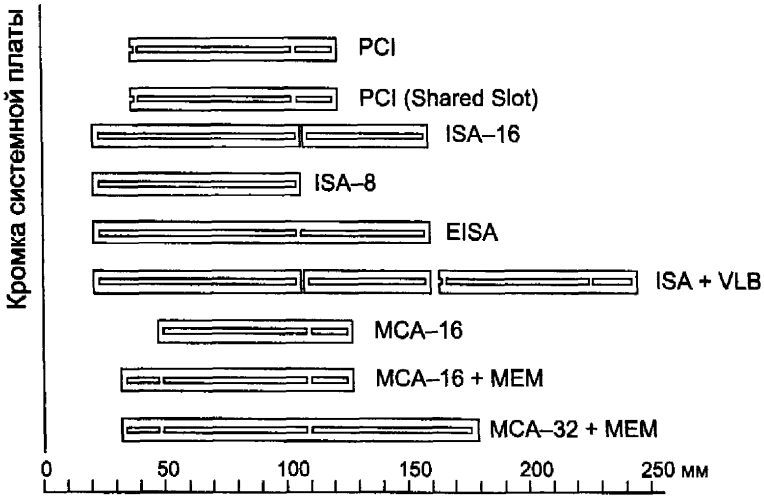


Рис.4.

У адаптеров для шины PGI, в отличие от ISA/EISA и VLB, компоненты расположены на левой стороне печатной платы. Для экономии площади печатной платы часто используют так называемый *разделяемый слот* (Shared Slot). На самом деле это разделяемое окно на задней стенке корпуса, которое может использоваться либо картой ISA, либо картой PCI. Таким образом, максимальное суммарное количество установленных адаптеров ISA и PCI оказывается на единицу меньшим, чем видимое количество слотов на системной плате.

Общий вид разъемов реализующих некоторые из перечисленных шин расширения приведены на рис. 5 –рис.7.

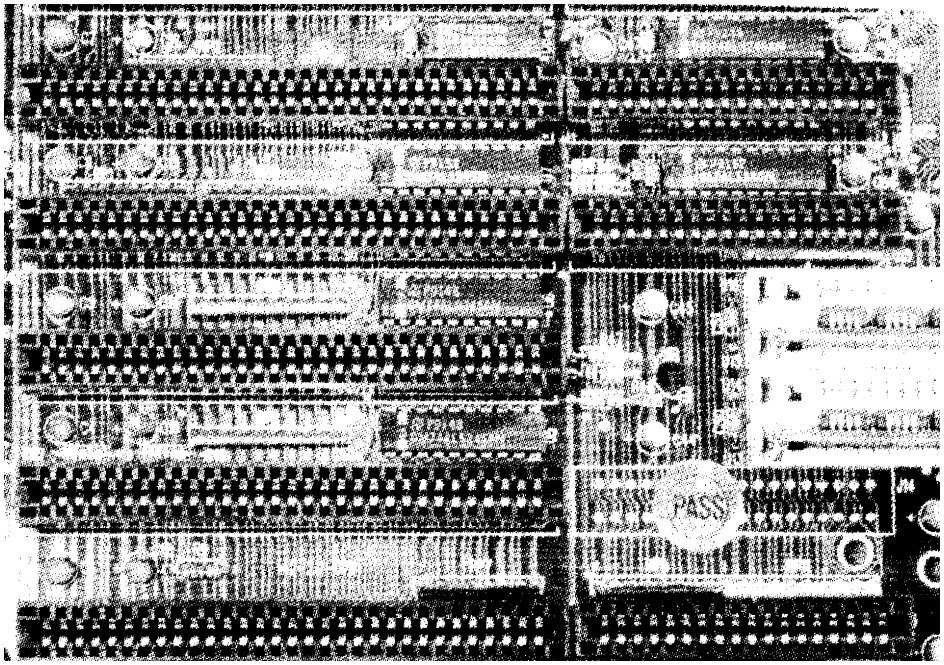


Рис.5.

На рис.5 представлен общий вид слотов 16-разрядной и 8-ми разрядной шины ISA.

Шина EISA является 32-разрядной шиной. Она представляет собой стандартизованное расширение шины ISA (рис.6).

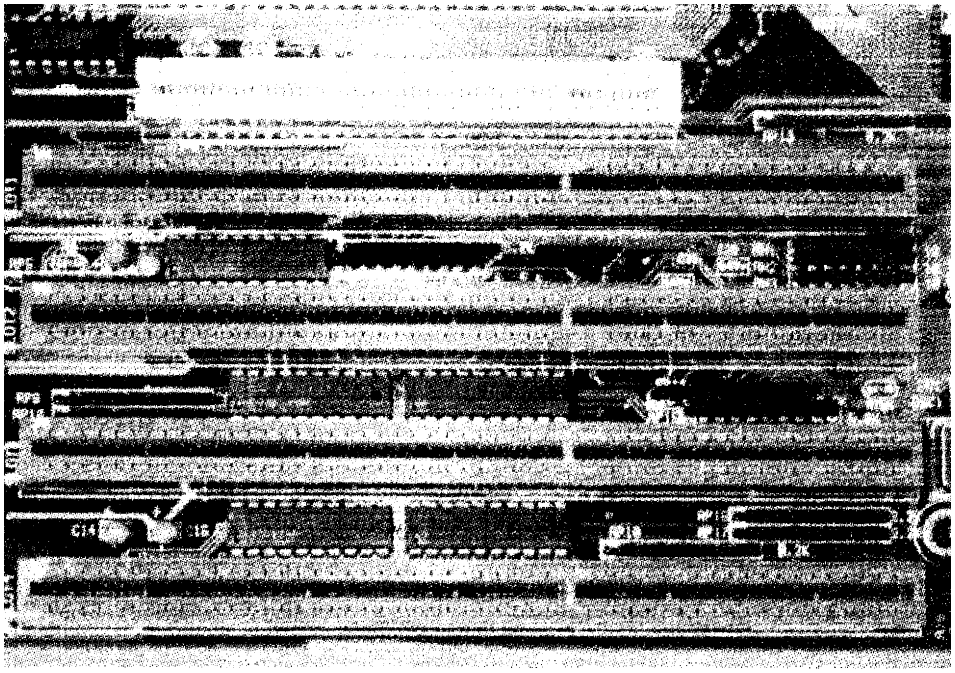


Рис.6.

Слот шины EISA имеет «двухэтажную» конструкцию (рис.7), благодаря чему в слоты EISA можно устанавливать обычные ISA адаптеры.

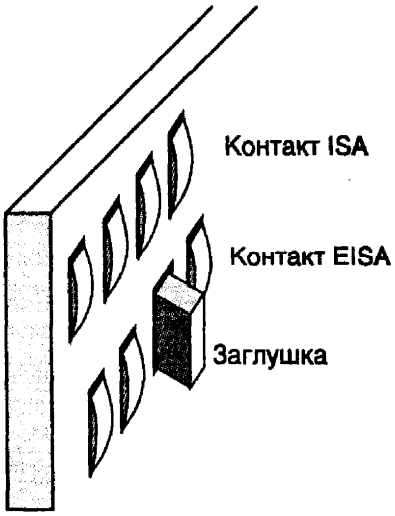


Рис.7.

Локальная шина VESA (VLB) представляет собой не новое устройство на материнской плате, а скорее расширение шины ISA для обмена видеоданными (рис.8). Обмен данными осуществляется под управлением контроллеров, расположенных на специальной карте, устанавливаемой в слот VLB.

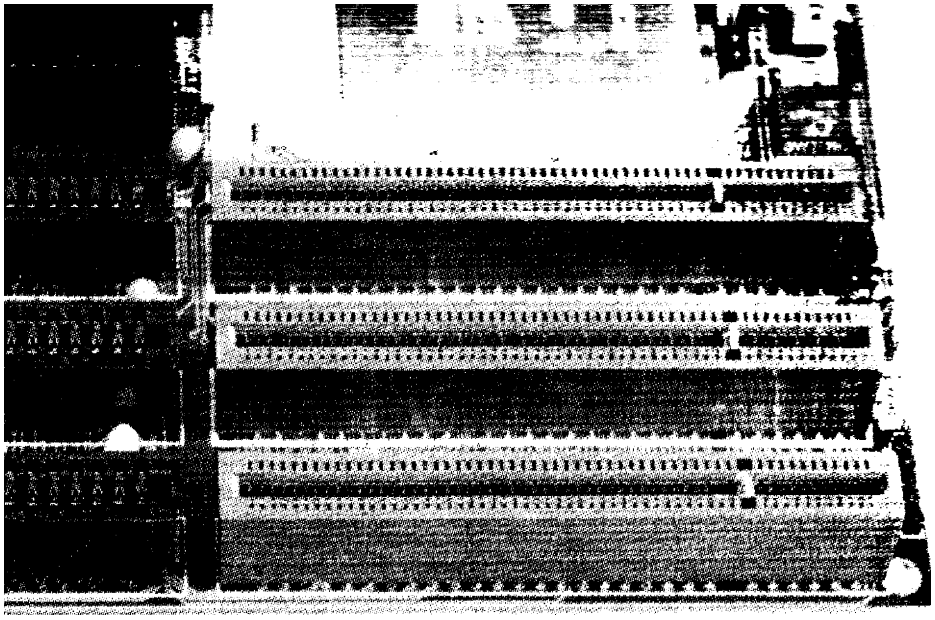


Рис.8.

Шина PCI работает с фиксированной частотой 33МГц, осуществляя передачу 32-разрядных данных, она поддерживает работу до 10 периферийных устройств. Теоретическая максимальная пропускная способность шины-132Мб/с, что соответствует современным МП. Разъёмы PCI имеют 50 штырьков для сопряжения с другими компонентами(типа PAD-вилка) (рис.9.).

Строго говоря, PCI относится к классу т.н. mezzanine-шин, т.е. шин - “пристроек”, поскольку между локальной шиной МП и самой PCI находится специальная микросхема согласующего “моста”(bridge).

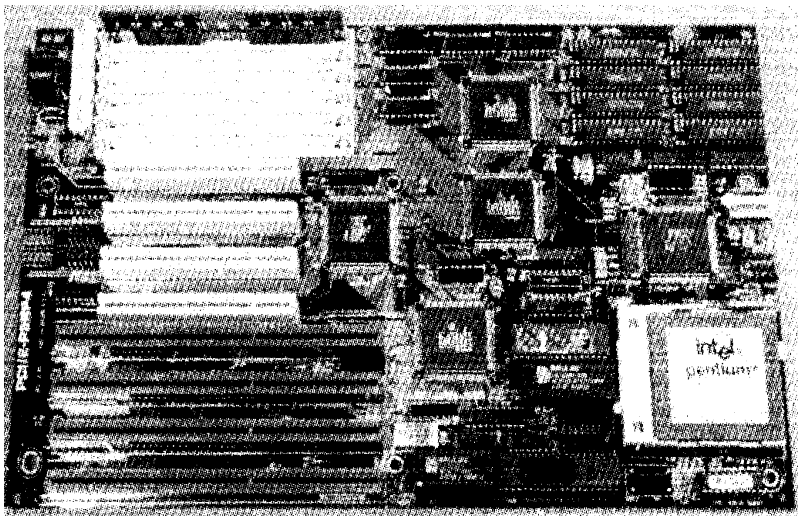


Рис.9.

2.5. Чипсет

Чипсет является связующим звеном между всеми компонентами системной платы. Его положение в архитектуре системных плат с шинами

PCI, ISA и интегрированными контроллерами периферии иллюстрирует функциональная схема, приведенная на рис. 10.

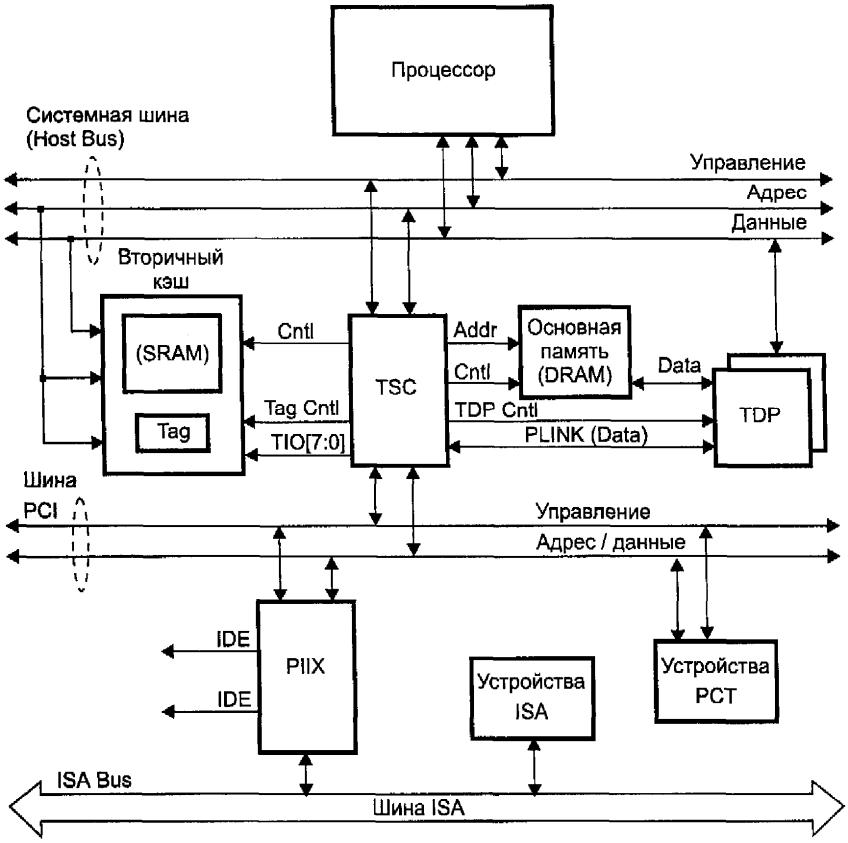


Рис.10.

Центральную роль в архитектуре играет, конечно же, процессор. К его локальной шине (Host Bus) адреса и данных подключаются модули (микросхемы) вторичного кэша. Основная динамическая память имеет собственную мультиплексированную шину адреса и шину данных, обычно изолированную от локальной шины процессора. На этом «этаже» архитектуры чипсет решает следующие задачи:

1. Обслуживание управляющих и конфигурационных сигналов процессора.
2. Мультиплексирование адреса и формирование управляющих сигналов динамической памяти, связь шины данных памяти с локальной шиной.
3. Формирование управляющих сигналов вторичного кэша, сравнение его тегов текущим адресом обращения на локальной шине.
4. Обеспечение когерентности (согласованности) данных в обоих уровнях кэш-памяти и основной памяти при обращении как со стороны процессора (процессоров), так и от контроллеров шины PCI.
5. Связь мультиплексированной шины адреса и данных шины PCI с локальной шиной процессора и шиной динамической памяти.
6. Формирование управляющих сигналов шины PCI, арбитраж контроллеров шины.

Взятая в качестве данного примера структура реализуется на трех микросхемах чипсета Intel82430FX — системном контроллере (TSC

82437FX) и двух корпусах коммутаторов данных (TDP 82371FB). Эта часть чипсета определяет:

1. Типы и частоты поддерживаемых процессоров.
2. Политику записи, возможности применения различных микросхем и скоростные характеристики вторичного кэша, возможный размер кэша и кэшируемой области основной памяти.
3. Типы, объемы и максимальное количество модулей динамической памяти, возможность чередования банков, скоростные характеристики обмена.
4. Поддерживаемые частоты шины PCI, возможное количество контроллеров шины, способы буферизации.

Следующий (вниз) этаж архитектуры — устройства, подключенные к шине PCI. Эта шина является центральной в современных системных платах, и все интерфейсные адаптеры, а также системные сродства ввода/вывода в конечном счете общаются с ядром системы (процессором и памятью) через шину PCI. Кроме плат расширения, устанавливаемых в слоты тины PCI, ее абонентом является и мост PIIX — практически неотъемлемая часть современных плат.

PIIX (PCI IDE ISA Xcelerator) представляет собой многофункциональное устройство, на которое возлагаются следующие функции:

1. Организация моста между шинами PCI и (E)ISA с согласованием частот синхронизации этих шин.
2. Реализация высокопроизводительного, обычно двухканального интерфейса ATA (IDE), подключенного к шине PCI.
3. Реализация стандартных системных средств ввода/вывода — двух контроллеров прерываний, двух контроллеров прямого доступа к памяти, трехканального системного счетчика-таймера, канала управления динамиком, логики немаскируемого прерывания.
4. Коммутация линий запросов прерывания шин PCI и ISA, а также встроенной периферии на линии запросов контроллеров прерываний, управление их чувствительностью (по перепаду или уровню), обслуживание прерывания от сопроцессора.
5. Коммутация каналов прямого доступа к памяти.
6. Поддержка режимов энергосбережения — обработка SMM, программирование событий управления потреблением, управление частотой процессора (используя сигнал STPCLK#), переключение режимов.
7. Реализация моста с внутренней шиной X-Bus, используемой для подключения микросхем контроллера клавиатуры, BIOS, CMOS RTC, контроллеров гибких дисков и интерфейсных портов.
8. Реализация контроллера USB.

В чипсете Intel 82430FX эти функции (кроме контроллера USB) выполняет микросхема 82371FB. Реализация перечисленных функций влияет на такие параметры системной платы, как:

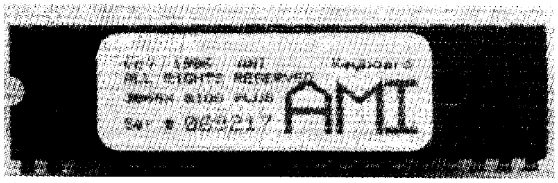
1. Производительность шины ISA (мост PCI-ISA использует буферизацию запросов операций, и существенна логика работы и емкость буферов).
2. Производительность, число каналов ATA, возможные режимы обращения (если режимы РЮ Mode 3 или 4 поддерживают почти все чипсеты, то ATA-33 еще не столь популярен).
3. Гибкость системы управления энергопотреблением. - Гибкость конфигурирования системных ресурсов и поддержка РпР.
4. Гибкость управления адресацией ROM BIOS, позволяющая использовать большой объем хранимого кода, не занимая излишнего пространства в верхней памяти.

Контроллеры гибких дисков, интерфейсных портов, клавиатуры, CMOS RTC могут входить собственно в чипсет, а могут быть реализованы и на отдельных «инородных» микросхемах. От них зависят следующие параметры системной платы:

1. Типы поддерживаемых дисководов (2,88 Мбайт поддерживают теперь почти все контроллеры, так что очередь за распространением соответствующих дисководов и дискет).
2. Режимы параллельного порта (стандартный, двунаправленный, ECP, EPP).
3. Режимы последовательных портов (хорошим тоном считается совместимость с 16550A и поддержка FIFO и DMA).

2.6. BIOS

Базовая система ввода/вывода *BIOS* является ключевым элементом системной платы, без которого все ее замечательные компоненты представляют собой лишь набор дорогих «железок». BIOS, пользуясь средствами, предоставляемыми чип-сетом, управляет всеми компонентами и ресурсами системной платы. Из этого следует, что используемая версия BIOS очень сильно привязана к чипсету, и, кроме того, она должна знать особенности применяемых компонентов (процессор, память, интегрированные контроллеры). Код BIOS хранится в микросхеме энергонезависимой постоянной *(ROM BIOS),* общий вид которых представлен на рис.11, или флэш-памяти *(Flash BIOS).*



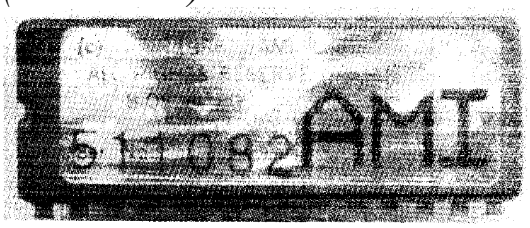


Рис.11.

2.6. Оперативная память.

Оперативная память в настоящее время физически реализуется конструктивными модулями. Наиболее распространенными типами корпусов ОЗУ является модули с однорядным расположением выводов типа SIMM (Single In Line Memory Module), представленный на рис. 12, и с двухрядным расположением выводов DIMM. Указанные конструктивные модули обычно содержат целый набор микросхем ОЗУ, спаянных на одной плате, и имеют особый торцевой соединитель, который предназначен для подключения к разъему (слоту) шины компьютера.

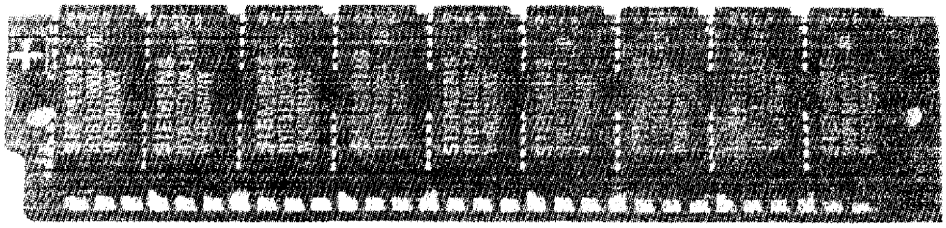


Рис.12.

2.6. Карты расширения.

Карты расширения, используемые для подключения периферийных устройств, представлены на рис 13-15.

На рис.13 представлен внешний вид графической видеокарты VGA (для шины ISA).

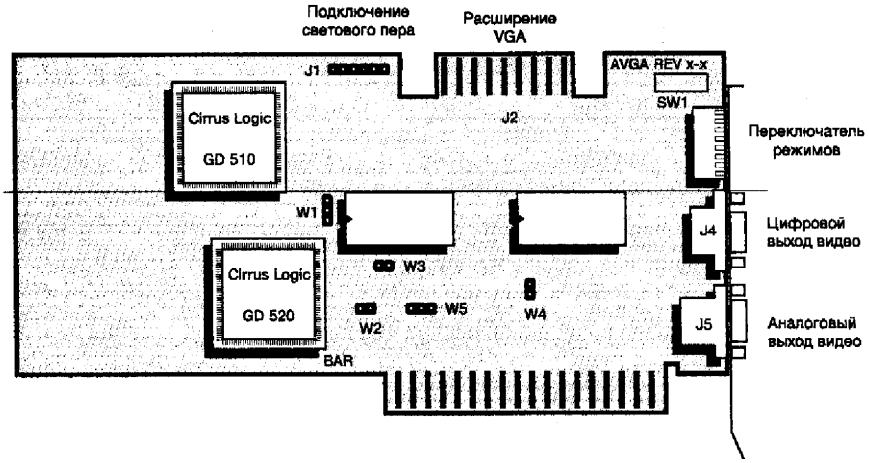


Рис.13.

На рис.14 представлен комбинированный контроллер для FDD, HDD, двух последовательных и одного параллельного интерфейсов.

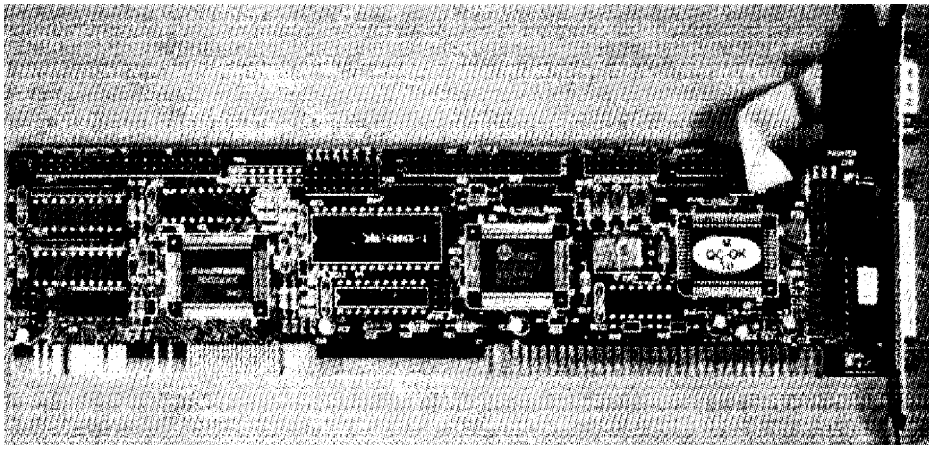


Рис.14.

Внешний вид мультикарты с IDE-контроллером представлен на рис.15. Здесь же изображены разъемы для подключения FDD, разъемы игрового порта, порта принтера, и двух последовательных портов.

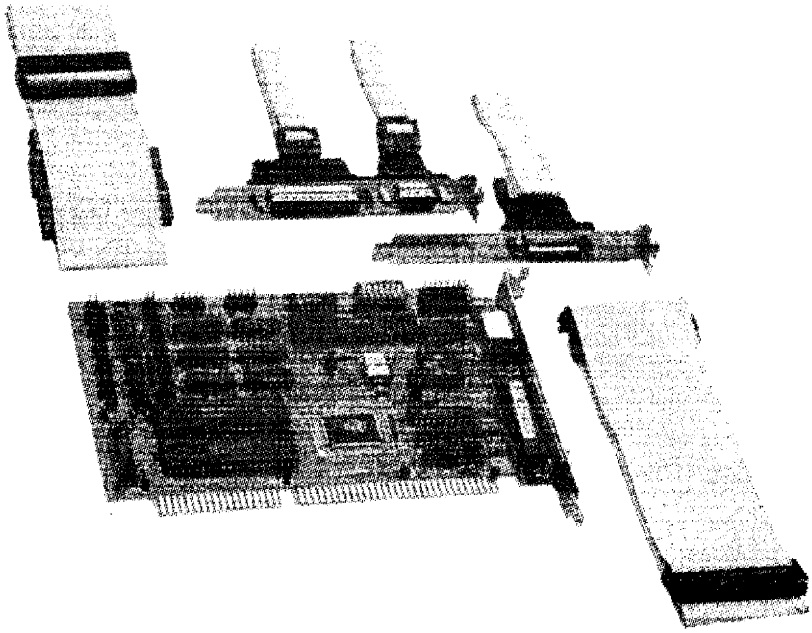


Рис.15.

Разъемы для подключения передней панели компьютера к системной плате показаны на рис.16.

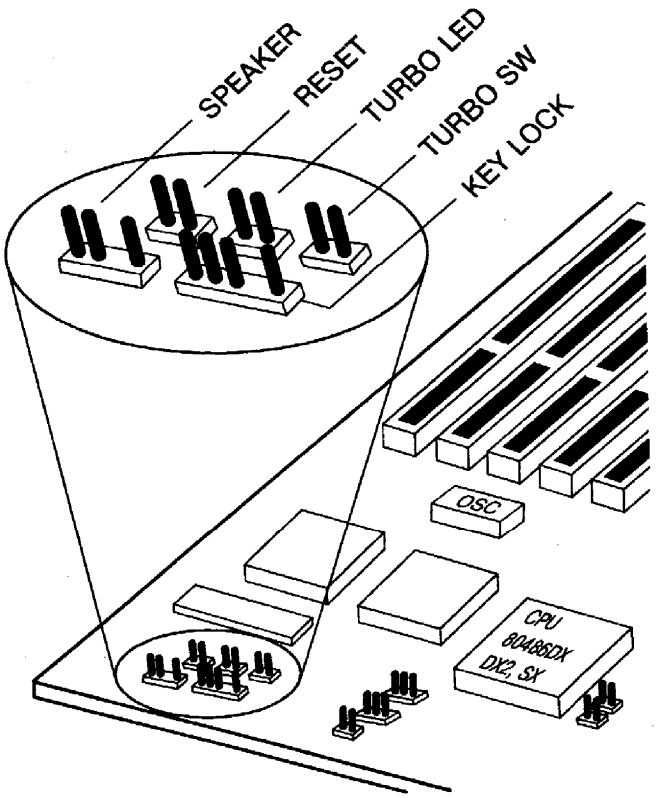


Рис.16.

Разъемы для подключения к системной плате питания представлены на рис. 17.

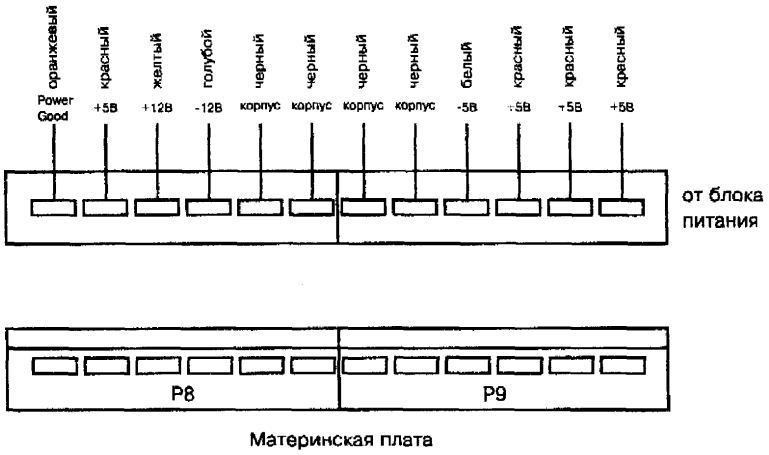


Рис.17.

Подключение накопителей

Накопители (FDD, HDD), подключаются к контроллерам соответствующими кабелями (шлейфами), представленными на рис.18 и 19.

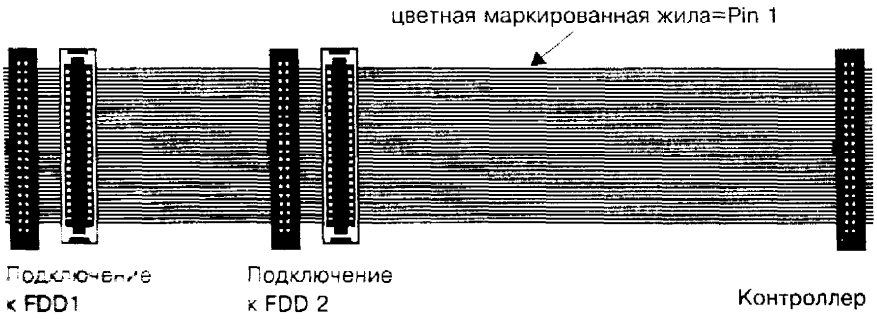


Рис.18.

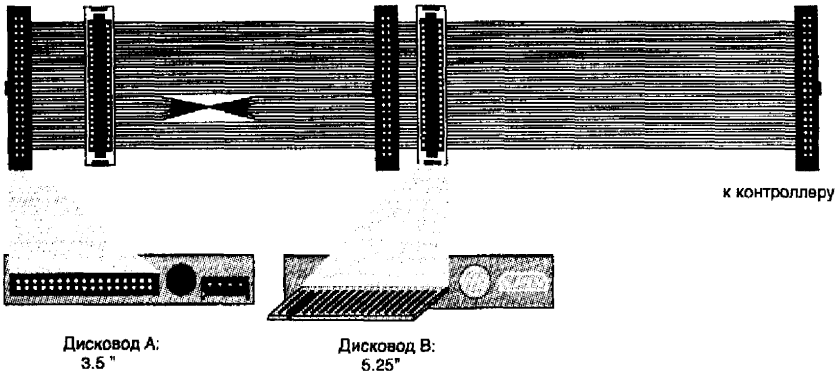


Рис.19.

III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

Персональный компьютер, набор системных плат и основных комплектующих ПК.

IV. ЗАДАНИЕ НА РАБОТУ

Изучить состав системного блока ПК. Рассмотреть различные типы системных плат ПК, представленных преподавателем. Изучить различные стандарты шин ПК и их конструктивную реализацию.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Ознакомиться с кратким описанием состава ПК и шин расширения по предложенному методическому описанию.

Под руководством преподавателя снять с системного блока, выключенного ПК, защитный кожух.

Ознакомиться с основными электронными модулями, входящими в системный блок.

Изучить системную плату, выданную преподавателем. Разработать функциональную схему указанной системной платы.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

В отчете должны быть представлены конструктивная схема изучаемого системного блока ПК, описание шин расширения ПК,

разработанная функциональная схема и краткое описание системной платы, выданной преподавателем.

VII. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите основные электронные модули ПК.
2. Что такое шинная архитектура ПК?
3. Каким образом осуществляется подключение периферийных устройств к ПК?
4. Что такое чипсет? Назовите стандартные функции выполняемые чипсет.

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.- 151с.
2. Гук М. Аппаратные средства PC. Энциклопедия – СПб: Питер Ком, 1998. –816 с.

Лабораторная работа № 2. Изучение архитектуры МПС на примере ПК.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Знакомство с архитектурой МПС на примере изучения структуры, функций и порядка работы базовой системы ввода-вывода и программы конфигурирования ПК – BIOS.

a. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1. Общие сведения о BIOS.

BIOS (Basic Input/Output System) - базовая система ввода-вывода - это встроенное в компьютер программное обеспечение, которое ему доступно без обращения к диску. На PC BIOS содержит код, необходимый для управления клавиатурой, видеокартой, дисками, портами и другими устройствами, входящими в базовую архитектуру ПК.

Обычно BIOS размещается в микросхеме ПЗУ (ROM), размещенной на материнской плате компьютера (поэтому этот чип часто называют ROM BIOS). Эта технология позволяет BIOS всегда быть доступным, несмотря на повреждения, например, дисковой системы. Это также позволяет компьютеру самостоятельно загружаться. Поскольку доступ к RAM (оперативной памяти) осуществляется значительно быстрее, чем к ROM, многие производители компьютеров создают системы таким образом, чтобы при включении компьютера выполнялось копирование BIOS из ROM в оперативную память. Задействованная при этом область памяти называется Shadow Memory (теневая память).

В настоящее время, почти все материнские платы комплектуются Flash BIOS, BIOSом, который в любой момент может быть перезаписан в микросхеме ROM при помощи специальной программы.

BIOS PC стандартизирован, поэтому, в принципе менять его, также как, например, операционные системы нет необходимости. Дополнительные возможности компьютера можно использовать только использованием нового программного обеспечения.

BIOS, который поддерживает технологию Plug-and-Play, называется PnP BIOS. При использовании этой технологии BIOS должен быть обязательно прошит во Flash ROM.

Определить тип микросхемы ПЗУ, установленной на материнской плате, несложно. Для этого необходимо посмотреть на маркировку чипа ROM (28 или 32-контактная микросхема с наклейкой производителя BIOS), отодрав наклейку. Маркировка означает следующее (xxx означает три произвольных цифры):

1. 28Fxxx - 12V Flash память
2. 29Cxxx - 5V Flash память
3. 29LVxxx - 3V Flash memory (раритет)

4. 28Cxxx - EEPROM, почти тоже, что и Flash память
5. 27Cxxx - с окошком. EPROM: только для чтения, требует программатор для записи и ультрафиолетовую лампу для стирания
6. PH29EE010: SST ROM Чип - перепрошиваемый
7. 29EE011: Winbond чип - 5V Flash память
8. 29C010: Atmel Chip - 5V Flash память

Любые другие микросхемы, не имеющие окошка с маркировкой, не начинающейся с цифр 28 или 29, являются, скорее всего не Flash-памятью. Если же на микросхеме есть окошко - это верный признак того, что это не Flash.

Существует несколько причин, по которым это приходится делать перепрошивку новых версий BIOS. Основная из них - Windows 95 не всегда хорошо конфигурируется, если используются старые версии BIOS.

В настоящее время используются жесткие диски объемом более 528Мбайт. Для работы такого диска в системе необходимо поддержка LBA со стороны BIOS. Если BIOS не поддерживает LBA, то для работы с большими жесткими дисками приходится применять специальные утилиты. Применение таких утилит вызовет работу Windows в compatible mode, что отрицательно сказывается на быстродействии.

Полная поддержка Plug-and-Play со стороны Windows возможна только в случае применения PnP BIOS. Это очень веская причина для перепрошивки BIOS.

Кроме вышеуказанного, в новых версиях BIOS исправляются мелкие ошибки и недоработки. Новые версии могут содержать новые возможности, как то загрузка с CD ROM, SCSI перед IDE и т.п.

Источником новых версий BIOS являются: во-первых новые версии BIOS доступны на сайтах их производителей, во-вторых обычно производители материнских плат предлагают BIOSы для своих изделий. Так что, если знать производителя и название материнской платы, проблем не возникнет.

В принципе, название материнской платы можно и не знать. Обычно допускается прошивка BIOS от других плат, если на них установлен такой же чипсет и контроллер ввода-вывода. Однако такой возможностью следует пользоваться только в случае крайней необходимости, так как возможны другие несоответствия, например в количестве слотов и т.п., а прошивка неправильного BIOS может привести к тому, что материнскую плату придется выкидывать.

Чаще всего, производителя и название материнской платы можно определить по идентификационной строке, которую выдает BIOS сразу при включении компьютера.

Для **AMI BIOS** эта строка имеет вид, похожий на `51-0102-`**`1101`**`-00111111-101094-AMIS123-P` или `40-01S5-`

ZZ**1124**-10101111-060691-OPWBSX-F,

где производитель определяется третьей группой цифр. Далее необходимо найти свой идентификационный номер в таблице соответствия номеров и названий производителей. Определить название материнской платы можно по всей идентификациолнной строке, пользуясь тем же документом.

Идентификационная строка **Award BIOS** имеет вид 2A59CQ1CC

и позволяет определить чипсет (первые пять цифр и букв - 2A59C), производителя материнской платы (следующие два символа - Q1) и модель материнской платы (оставшаяся часть строки - CC). Далее необходимо посмотреть обозначения чипсетов, производителей и моделей в идентификационных строках.

Для выполнения операции перепрошивки Flash BIOS необходимо иметь програму-прошивальщик и файл с BIOS. Программы для перепрошивки обычно поставляются с материнскими платами, в крайнем случае их можно скачать у производителей материнских плат и BIOS.

2.2. Описание настроек SETUP BIOS

В работе приводятся практически все (по мере создания) параметры, устанавливаемые в программе SETUP для BIOS фирмы AWARD Software International Inc. В конкретной материнской плате каких-то из описываемых параметров может и не быть. Одни и те же параметры могут называться по разному в зависимости от производителя материнской платы, поэтому здесь в некоторых случаях приведено несколько вариантов.

Раздел BIOS FEATURES SETUP

1. **Virus Warning** (Предупреждение о вирусе) - разрешение этого параметра запрещает любую запись в загрузочный сектор жесткого диска без разрешения пользователя. Он введен для защиты от так называемых boot-вирусов, поражающих загрузочный сектор. Рекомендуется всегда разрешать этот параметр, но следует учесть, что, например, Windows 95 при установке "зависает", если Virus Warning установлен в Enable (при этом на экране появляется черный квадрат).Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
2. **Boot Virus Detection** (Определение вируса в загрузочном секторе) - смысл этого параметра сильно отличается от Virus Warning. Идея заключается в следующем - если этот параметр запрещен, то до загрузки операционной системы BIOS переписывает загрузочный сектор во флэш-память и сохраняет его там. После установки параметра в значение Enabled BIOS не будет загружать систему с жесткого диска, если содержимое boot-сектора отличается от сохраненного в памяти. Далее, по усмотрению пользователя,

возможно либо загрузить систему с жесткого диска, либо с дискеты. Может принимать значения:

a. **Enabled** - разрешено
b. **Disabled** - запрещено

3. **CPU Internal Cache/External Cache** (Внутренний/Внешний кэш процессора) - разрешается/запрещается внутренний или внешний кэш процессора. Запрещать какой-либо вид кэш-памяти следует только в случае необходимости искусственно замедлить работу компьютера, например, при установке какой либо старой платы расширения. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
4. **CPU Level 1 Cache/CPU Level 2 Cache** (Кэш процессора первого уровня/Кэш процессора второго уровня) - разрешается/запрещается кэш первого уровня или кэш процессора второго уровня для процессоров архитектуры Pentium Pro (Pentium II, Deshutes и т.п.). Запрещать какой-либо вид кэш-памяти следует только в случае необходимости искусственно замедлить работу компьютера, например, при установке какой либо старой платы расширения. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
5. **CPU Level 2 Cache ECC Check** (Включить ECC для кэш-памяти процессора 2 уровня) - параметр может присутствовать только для плат с процессорами архитектуры Pentium II. Разрешать его есть смысл только в том случае, если установленный процессор класса Pentium II имеет кэш-память второго уровня с возможностью ECC контроля. В некоторых процессорах допущена ошибка и включение этого режима может привести к нестабильной работе компьютера. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
6. **BIOS Update** (Обновление BIOS) - процессоры семейства P6 (Pentium Pro, Pentium II, Celeron, Xeon и т.д.) имеют особый механизм, называемый "программируемым микрокодом", который позволяет исправить некоторые виды ошибок, допущенных при разработке и/или изготовлении процессоров за счет изменения микрокода. Обновления микрокода остаются в BIOS и загружаются в процессор после включения компьютера и запуска программы BIOS. Именно поэтому BIOS для материнских плат с Pentium II и выше необходимо регулярно обновлять. Может принимать значения:
   a. **Enabled** - разрешено

   b. **Disabled** - запрещено

7. **CPU Fast String** (Быстрые операции со строками) - Разрешение этого параметра позволяет использовать некоторые специфические особенности архитектуры семейства Pentium Pro (Pentium II, Deshutes и т.п.), в частности, возможность кэширования операций со строками. Надо только понимать, что и в самой программе должны быть выполнены условия для включения этого механизма. Эти условия указаны в документации на любой процессор данного семейства. Параметр рекомендуется оставлять в состоянии "Разрешено". Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
8. **HDD S.M.A.R.T Capability** (Возможность S.M.A.R.T диагностики) - позволяет разрешать/запрещать возможность диагностики состояния жесткого диска в соответствии с требованиями стандарта S.M.A.R.T. Авторы BIOS, к сожалению, не раскрывают механизма функционирования S.M.A.R.T диагностики в BIOS, поэтому не совсем понятно, каким образом обрабатывается информация от жесткого диска, так как граничные значения параметров жесткого диска зависят от конкретного производителя. При разрешении параметра и нарушении нормального функционирования жесткого диска BIOS выдает на экран соответствующее сообщение до появления таблицы с характеристиками компьютера. Следует учесть, что разрешение этого параметра снизит производительность компьютера на несколько процентов. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
9. **Deturbo Mode** (Режим deturbo) - при разрешении этого параметра сигнал FLUSH# становится активным и никакие данные после этого не кэшируются процессором в свой внутренний кэш (кэш первого уровня) процессорами архитектуры Pentium Pro (Pentium II, Deshutes и т.п.). Разрешение этого параметра следует использовать только при необходимости сознательно замедлить работу компьютера. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
10. **Quick Power On Self Test** (быстрый тест компьютера после включения питания) - разрешение этого параметра приводит к значительному сокращению времени на начальное тестирование компьютера BIOS'ом, особенно при значительных объемах оперативной памяти. Следует только учесть, что память, например, в этом случае не тестируется, а только проверяется ее размер. Может принимать значения:

a. **Enabled** - разрешено
b. **Disabled** - запрещено

11. **Boot Sequence** (последовательность загрузки) - параметр устанавливает последовательность опроса устройств, с которых может быть загружена операционная система. Эти устройства обозначаются либо буквами для физических жестких дисков и обычных дисководов, либо названием устройства - CD-ROM для накопителей CD-ROM, LS для накопителей a:drive на 120 Mb или ZIP для накопителей ZIP IDE на 100 Mb. Для современных версий возможные значения могут выглядеть так: **A**,**C**; **C only**; **CD-ROM**, **C**; **C**, **A**; **D**, **A**; **LS/ZIP**, **C**.
12. **Boot Up System Speed** (скорость системы после загрузки) - скорость системной шины и соответственно, процессора после включения компьютера. Параметр используется для искуственного снижения скорости компьютера из-за старых программ и/или плат расширения. Может принимать значения:
    a. **High** - номинальная скорость процессора и номинальная частота системной шины
    b. **Low** - пониженная скорость процессора и частота системной шины
13. **Gate A20 Option** (вариант включения шины A20) - параметр позволяет управлять способом включения адресной шины A20 и, тем самым, обращаться к памяти за 1 мегабайтом. Этот параметр в современных платах не управляется пользователем и всегда в значении Fast. Может принимать значения:
    a. **Fast** - управление осуществляется chipset, что повышает скорость работы
    b. **Normal** - управление осуществляется через контроллер клавиатуры
14. **Swap Floppy Drive** (перестановка дисководов) - если разрешен, то дисководы A и B как бы меняются местами. Имеет смысл только при наличии 2-х дисководов в компьютере. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
15. **Boot Up Floppy Seek** (поиск загрузочного дисковода после включения компьютера) - если этот параметр разрешен, то BIOS обращается к каждому дисководу на предмет распознавания его формата (40 или 80 дорожек он поддерживает). Поскольку уже с 1993 года дисководы на 40 дорожек не выпускаются, не следует разрешать этот параметр, так как BIOS каждый раз будет тратить несколько секунд на выяснение формата дисковода. Может принимать значения:

   a. **Enabled** - разрешено
   b. **Disabled** - запрещено

16. **Boot Up NumLock Status** (включение цифровой клавиатуры по включению компьютера) - разрешение этого параметра включает индикатор NumLock и цифровая клавиатура генерирует коды цифр и знаков, в противном случае генерируются коды стрелок, Ins, Del и т.д. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
17. **Typematic Rate Setting** (установка скорости ввода символов) - разрешает или запрещает установку скорости повторения ввода символов клавиатурой при нажатой клавише. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
18. **Typematic Rate (Chars/Sec)** (частота повторения симв/сек) - параметр оказывает действие только в том случае, если разрешен Typematic Rate Setting. Частота повторения имеет ряд фиксированных значений, которые и может принимать этот параметр:
   a. **6**, **8**, **10**,**12**, **15**, **20**, **24** или **30**
19. **Typematic Delay (Msec)** (задержка повторения в msec) - устанавливает значение задержки от момента нажатия клавиши до начала повторения символа клавиатурой. Оказывает действие только в том случае, если разрешен Typematic Rate Setting. Значение может быть выбрано из ряда:
   a. **250**, **500**, **750** или **1000**
20. **PS/2 Mouse Function Control** (управление функциями порта мыши PS/2) - разрешение этого параметра отдает IRQ12 только для порта мыши PS/2. В противном случае, при отсутствии подключенной к компьютеру мыши с интерфейсом PS/2, IRQ12 свободно для других устройств. Рекомендуется устанавливать значение Auto. Может принимать значения:
   a. **Enabled** - разрешено и IRQ12 занято.
   b. **Auto** - BIOS определяет присутствие или отсутствие PS/2 мыши.
21. **OS/2 Onboard Memory > 64MB** (выбор значения для OS/2, если памяти больше, чем 64 Mb) - требует разрешения при выполнении двух условий - в компьютере установлено больше, чем 64 Mb памяти и используется OS/2 в качестве операционной системы. Может принимать значения:

a. **Enabled** - разрешено
b. **Disabled** - запрещено

22. **PCI/VGA Palette Snoop** (корректировка палитры VGA видеокарты на PCI) - параметр следует разрешать только в том случае, если на экране некорректно отображаются цвета. Как правило, этот эффект может возникать при использовании таких нестандартных устройств, как MPEG карты, 3D ускорители и т.п. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
23. **Video ROM BIOS Shadow** (видео BIOS в память) - разрешение этого параметра приводит к переносу видео BIOS из ПЗУ (постоянного запоминающего устройства) на видеокарте в основную память компьютера, что значительно ускоряет работу с видео BIOS (это необходимо и видно в DOS). Ускорение объясняется как и тем, что обращение к ПЗУ намного медленнее, чем к ОЗУ, так и тем, что обращение к ПЗУ, как правило, выполняется в 8-ми разрядной сетке, а к ОЗУ - 32-х или 64-х сетке. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
24. **Floppy Disk Access Control** (R/W) (управление чтением/записью на гибкий диск) - разрешение этого параметра позволяет записывать информацию на дискету, в противном случае дискету можно читать. Параметр следует использовать для защиты от несанкционированного копирования с компьютера. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
25. **Report No FDD For WIN 95** (сообщение об отсутствии накопителя на гибких дисках для Windows 95) - параметр используется, как правило, в сетевых компьютерах без флоппи-дисковода или в случае необходимости размещения в компьютере устройства, для которого не хватает прерывания. При выборе Yes и одновременном выборе Disable в параметре Onboard FDC Controller, Windows 95 освободит IRQ 6, занимаемое контроллером флоппи-дисковода, для использования другими устройствами. Совместное разрешение этих параметров обязательно в компьютерах без дисководов (используемых как сетевые в организациях) для уменьшения времени запуска Windows 95. Может принимать значения:
    a. **Yes** - освободить IRQ 6
    b. **No** - не освобождать (независимо от того, есть ли флоппи-дисковод или нет)

26. **Delay IDE Initial** (задержка инициализации IDE устройства) - в данном параметре устанавливается значение времени (в секундах) в течении которого IDE устройство не будет опрашиваться BIOS'ом после включения питания или сброса. Ненулевое значение параметра рекомендуется устанавливать только в случае применения каких-либо старых жестких дисков или приводов CD-ROM. Параметр принимает значения в диапазоне от 0 до 30 сек., в зависимости от производителя материнской платы.
27. **MPS 1.4 Support**(поддержка режима MPS 1.4) - параметр появляется только в BIOS материнских плат, допускающих установку нескольких процессоров. Выбор режима влияет на распределение ресурсов компьютера. При запрещении устанавливается режим MPS 1.1. Строго говоря, для различных Windows можно устанавливать любое значение (даже лучше Disable), а для Novell Netware рекомендуется Enable. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** – запрещено

Раздел CHIPSET FEATURES SETUP

Установка параметров для FPM DRAM, EDO DRAM и Synchronous DRAM

1. **AUTO Configuration** (автоматическая конфигурация) - имеет 3 значения
    a. **60 ns** - устанавливает парамеры доступа для DRAM с быстродействием 60 ns
    b. **70 ns** - то же для памяти с быстродействием 70 ns
    c. **Disabled** (запрещена) - позволяет установить любые возможные параметры доступа к DRAM памяти
2. **DRAM RAS# Precharge Time** (время предварительного заряда по RAS) - Эта функция позволяет определить количество тактов системной шины для формирования сигнала RAS. Уменьшение этого значение увеличивает быстродействие, но чрезмерное для конкретной памяти уменьшение может привести к потере данных. Принимает значения:
    a. **3**
    b. **4**
3. **DRAM R/W Leadoff Timing** (число тактов при подготовке выполнения операции чтения/записи) - определяет число тактов на шине до выполнения любых операций с DRAM. Параметр может принимать значения:

   a. **8/7** - восемь тактов для чтения и семь тактов для записи
   b. **7/5** - семь тактов для чтения и пять тактов для записи

4. **DRAM RAS to CAS Delay** (задержка между RAS и CAS) - Во время доступа к памяти обращения к столбцам и строкам выполняются отдельно друг от друга. Этот параметр и определяет отстояние одного сигнала от другого. Параметр может принимать значения:
   a. **3** - три такта задержки
   b. **2**-два такта задержки

   Уменьшение значения увеличивает быстродействие.

5. **DRAM Read Burst Timing** (время пакетного чтения памяти) - Запрос на чтение и запись генерируется процессором в четыре раздельные фазы. В первой фазе инициируется обращение к конкретной области памяти, а в оставшихся происходит собственно чтение данных. Параметр может принимать значения:
   a. **x2222** - два такта задержки
   b. **x3333** - три такта задержки
   c. **x4444** - четыре такта задержки
      Уменьшение суммарного количества тактов увеличивает быстродействие.

6. **Speculative Leadoff** (опережающая выдача сигнала чтения) - разрешение этого параметра позволяет выдавать сигнал чтения немного ранее, чем адрес будет декодирован. Этот прием снижает общие затраты времени на операцию чтения. Другими словами, процессор будет инициировать сигнал чтения одновременно с генерацией того адреса, где находятся необходимые данные. Сигнал чтения воспринимается контроллером DRAM и, если параметр Speculative Leadoff разрешен, то контроллер выдаст сигнал чтения до завершения декодирования адреса. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено

7. **Turn-Around Insertion** (задержка между циклами) - Если этот параметр разрешен (Enabled), то между двумя последовательными циклами обращения к памяти включается один дополнительный такт. Разрешение уменьшает быстродействие, но увеличивает достоверность при операциях чтения/записи. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено

8. **Data Integrity (PAR/ECC)** (целостность данных) - разрешает/запрещает контроль памяти на ошибки. Вид контроля

устанавливается параметром DRAM ECC/PARITY Select. Может принимать значения:

a. **Enabled** - разрешено
b. **Disabled** - запрещено

9. **DRAM ECC/PARITY Select** (режим коррекции ошибок/проверка по четности) - Параметр появляется только в тех материнских платах, в которых chipset поддерживает ECC и только в том случае, если установлены модули памяти с истинной четностью. В некоторых вариантах BIOS этим параметром может устанавливаться только вид проверки, а разрешение на проверку устанавливается параметром Data Integrity (PAR/ECC). Такие планки часто называют также 36-разрядными. Может принимать значения:
   a. **Parity** - в случае возникновения ошибки на монитор выдается сообщение о сбое по четности в памяти и работа компьютера останавливается
   b. **ECC** -**E**rror **C**ontrol **C**orrection- в случае возникновения одиночной ошибки она исправляется и работа продолжается. Если имеет место не одиночная ошибка, то работа компьютера также прекращается. Следует только учесть, что по данным Intel, скорость обмена с памятью при включении этого режима уменьшается приблизительно на 3%
10. **Fast RAS# to CAS# Delay** (интервал между RAS и CAS) - При регенерации памяти строки и столбцы адресуются отдельно, поэтому этот параметр устанавливает интервал между сигналами RAS и CAS.
11. **SDRAM Configuration** (Конфигурация SDRAM) - параметром определяется, должна ли программа BIOS сама определять временные характеристики доступа к памяти на основании информации из блока SPD или разрешить это сделать пользователю. Может принимать значения:
   a. **By SPD** - параметры доступа устанавливаются по SPD
   b. **7 ns (143 Mhz)** - параметры доступа устанавливаются BIOS как для памяти с временем доступа 7 ns и частотой шины 143 MHz
   c. **8 ns (125 Mhz)** - параметры доступа устанавливаются BIOS как для памяти с временем доступа 8 ns и частотой шины 125 MHz
   d. **Disabled** - устанавливаются пользователем
12. **SDRAM RAS Precharge Time** (Синхронная память - время предварительного заряда) - параметр позволяет определять быстрое или медленное накопление заряда по RAS до начала цикла регенерации памяти. Установка значения Fast увеличивает быстродействие, но Slow повышает стабильность работы компьютера, поэтому значение Fast следует устанавливать в случае уверенности в качестве памяти. Может принимать значения:

   a. **Fast** -быстро
   b. **Slow** - медленно

13. **SDRAM (CAS Lat/RAS-to-CAS)** (Синхронная память - задержка CAS/От RAS к CAS) - этот параметр позволяет комбинировать между длительностью сигнала CAS и задержкой между сигналами RAS и CAS. Значение этого параметра зависит от характеристик SDRAM, примененной в материнской плате и от быстродействия процессора. Поэтому изменять этот параметр стоит крайне осторожно. Может принимать значения:
   a. **2/2**
   b. **3/3**
14. **SDRAM CAS to RAS Delay** (задержка между CAS и RAS) - параметр определяет значение задержки после выдачи сигнала RAS до появления сигнала CAS для синхронной памяти. Чем меньше это значение, тем быстрее доступ к памяти. Тем не менее изменять его следует осторожно. Параметр может принимать значения:
   a. **3** - три такта задержки
   b. **2** - два такта задержки
15. **SDRAM CAS# Latency** (задержка CAS для SDRAM) - Устанавливает значение задержки выдачи сигнала CAS для SDRAM. Меньшее значение увеличивает производительность системы. Рекомендуется устанавливать меньшее значение для SDRAM с быстродействием 10 nc или лучше.Может принимать значения:
   a. **2T**
   b. **3T**
16. **SDRAM Banks Close Policy** (правила закрытия банков SDRAM памяти) - параметр был введен для плат с набором 440LX из-за того, что память с 2-х банковой организацией некорректно работает в этих платах, если параметры доступа к банкам памяти установлены по умолчанию. В наборе 430TX этого не требовалось, так правила доступа для различной памяти были одинаковы. Изменять установки BIOS по умолчанию для этого параметра следует только в случае нестабильной работы памяти. Может принимать значения:
   a. **Page Miss** - используется для двухбанковой памяти
   b. **Arbitration** - для памяти из 4-х банков
17. **DRAM Idle Timer** (Таймер пассивного состояния памяти) - этим параметром устанавливается время (в тактах) до закрытия всех открытых страниц памяти. Влияет как на EDO так и на SDRAM память. Может принимать значения 0, 2, 4, 8, 10, 12, 16, 32.
18. **Snoop Ahead** (Предвидение) - разрешение этогопараметра позволяет потоковый обмен данными между PCI и памятью. Может принимать значения:

   a. **Enabled** - разрешено
   b. **Disabled** - запрещено

19. **Host Bus Fast Data Ready** (Быстрая готовность данных на шине) - разрешение этого параметра позволит снимать данные с шины одновременно с их выборкой. В противном случае данные будут удерживаться на шине один дополнительный такт. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
20. **Refresh RAS# Assertion** (задание RAS для регенерации) - Этим параметром устанавливается количество тактов (т.е. длительность RAS) для цикла регенерации. Принимаемые значения определяются качеством памяти и набором микросхем (chipset). Меньшее значение увеличивает производительность.
21. **MA Wait State** (такты ожидания до чтения памяти) - параметр позволяет установить или снять дополнительный такт ожидания до начала чтения памяти. Для памяти типа EDO один такт всегда есть по умолчанию и установка значения Slow добавляет еще один такт ожидания. Для SDRAM нет такта ожидания по умолчанию и установка Slow один такт вводит. Может принимать значения:
   a. **Slow** - добавляется один такт
   b. **Fast** - нет дополнительного такта ожидания
22. **SDRAM Speculative Read** (SDRAM опережающее чтение) - разрешение этого параметра позволяет выдавать сигнал чтения немного ранее, чем адрес будет декодирован. Этот прием снижает общие затраты времени на операцию чтения. Другими словами, процессор будет инициировать сигнал чтения одновременно с генерацией того адреса, где находятся необходимые данные. Сигнал чтения воспринимается контроллером DRAM и, если параметр SDRAM Speculative Read разрешен, то контроллер выдаст сигнал чтения до завершения декодирования адреса. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
23. **Spread Spectrum Modulated** (Спред модулированного спектра) - разрешение этого параметра позволяет уменьшить электромагнитное излучение от компьютера за счет уменьшения значения выбросов сигнала тактового генератора. Уменьшение может достигать 6%. Следует заметить, что это может отрицательно отразиться на работе чувствительных к форме сигнала устройств, например, жестких дисках с интерфейсом Fast Wide SCSI, поэтому параметр рекомендуется разрешать только при испытаниях компьютеров на электромагнитную совместимость. Может принимать значения:

a. **Enabled** - разрешено
b. **Disabled** - запрещено

i. Установка параметров для кэширования

2. **System BIOS Casheable** (кэширование области BIOS системы) - Разрешение этого параметра приводит к появлению возможности кэширования области памяти по адресам системного BIOS с F0000H по FFFFFH в кэш-память. Параметр будет использован только в том случае, если использование кэш-памяти разрешено в разделе BIOS Features Setup. Если какая-либо программа попытается выполнить операцию записи в эти адреса, то система выдаст сообщение об ошибке. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено
3. **Video BIOS Cacheable** (кэширование области BIOS видекарты) - Разрешение этого параметра приводит к появлению возможности кэширования области памяти по адресам BIOS видеокарты с C0000H по C7FFFH в кэш-память. Параметр будет использован только в том случае, если использование кэш-памяти разрешено в разделе BIOS Features Setup. Если какая-либо программа попытается выполнить операцию записи в эти адреса, то система выдаст сообщение об ошибке. Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено

Конфигурирование шин PCI, AGP, портов ввода/вывода и установка параметров IDE контроллера

4. **Video Memory Cache Mode** (Режим кэширования для видеопамяти) - параметр действителен только для процессоров архитектуры Pentium Pro (Pentium II, Deshutes и т.п.). В процессоре Pentium Pro была предусмотрена возможность изменять режим кэширования в зависимости от конкретной области памяти через специальные внутренние регистры, называемые Memory Type Range Registers - MTRR. С помощью этих регистров для конкретной области памяти могут быть установлены режимы UC (uncached - не кэшируется), WC (write combining - объединенная запись), WP (write protect - защита от записи), WT (write through - сквозная запись) и WB (write back - обратная запись). Установка режима USWC (uncached, speculative write combining - не кэшировать, режим объединенной записи) позволяет значительно ускорить вывод данных через шину PCI на

видеокарту (до 90 MB/c вместо 8 MB/c). Следует учесть, что видеокарта должна поддерживать доступ к своей памяти в диапазоне от A0000 - BFFFF (128 kB) и иметь линейный буфер кадра. Поэтому лучше установить режим USWC, но в случае возникновения каких-либо проблем (система может не загрузиться) установить значение по умолчанию UC. Может принимать значения:

a. **UC** - uncached - не кэшируется
b. **USWC** - uncached, speculative write combining - не кэшировать, режим объединенной записи.

5. **Graphics Aperture Size** (размер графической апертуры для AGP) - в этом параметре указывается максимальный размер области памяти для использования видеокартой с интерфейсом AGP. Значение по умолчанию, устанавливаемое по включению питания или сбросу, 4 MB. После инициализации BIOS'ом принимает значение, выбранное производителем материнской платы (как правило, 64 MB). Разрешенный ряд значений графической апертуры 4 MB, 8 MB, 16 MB, 32 MB, 64 MB, 128 MB и 256 MB.
6. **PCI 2.1 Support** (поддержка спецификации шины PCI 2.1) - при разрешении этого параметра поддерживаются возможности спецификации 2.1 шины PCI. Спецификация 2.1 имеет два основных отличия от 2.0 - максимальная тактовая частота шины увеличена до 66 MHz и вводится механизм моста PCI - PCI, позволяющий снять ограничение спецификации 2.0, согласно которой допускается установка не более 4-х устройств на шине. Запрещать этот параметр имеет смысл только при возникновении проблем после установки PCI платы (как правило, они возникают только с достаточно старыми платами). Может принимать значения:
   a. **Enabled** - разрешено
   b. **Disabled** - запрещено

7. **8 Bit I/O Recovery Time** (время восстановления для 8-ми разрядных устройств) - Параметр измеряется в тактах процессора, и определяет, какую задержку система будет устанавливать после выдачи запроса на чтение/запись устройства (или, как принято у Intel - порта) ввода/вывода. Эта задержка необходима, так как цикл чтения/записи для устройств ввода/вывода существенно дольше, чем для памяти. Кроме этого 8-ми разрядные устройства ввода/вывода сами по себе, как правило, медленнее 16-ти разрядных устройств ввода/вывода. Значение этого параметра по умолчанию 1 и его следует увеличивать только в случае установки в компьютер какого-либо медленного 8-ми разрядного устройства. Может принимать значения от 1 до 8 тактов.
8. **16 Bit I/O Recovery Time** (время восстановления для 16-ми разрядных устройств) - Параметр измеряется в тактах процессора, и

определяет, какую задержку система будет устанавливать после выдачи запроса на чтение/запись устройства (или, как принято у Intel - порта) ввода/вывода. Эта задержка необходима, так как цикл чтения/записи для устройств ввода/вывода существенно дольше, чем для памяти. Значение этого параметра по умолчанию1 и его следует увеличивать только в случае установки в компьютер какого-либо медленного 16-ти разрядного устройства. Может принимать значения от 1 до 4 тактов.

9. **Memory Hole At 15M-16M** ("дырка" в памяти внутри 15-го мегабайта памяти) - Разрешение этого параметра позволяет обращаться к устройствам ввода/вывода как к памяти и за счет этого увеличить скорость доступа к таким устройствам. Для функционирования этого механизма необходимо исключить для всех обычных программ возможность использования определенной области памяти (15-ый мегабайт), что и делает BIOS при разрешении этого параметра. Разрешать этот параметр следует в том случае, если это требуется в документации на установленную в данном компьютере плату. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
10. **Peer Concurrancy** (паралельная работа) - Этот параметр разрешает или запрещает одновременную работу нескольких устройств на шине PCI. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
11. **Chipset Special Features** (специальные возможности chipset) - Данный параметр разрешает/запрещает все новые функции, появившиеся в наборах HX, VX или TX по сравнению с FX. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
12. **Passive Release** (пассивное разделение) - Этот параметр включает/выключает механизм паралельной работы шин ISA и PCI. Если этот параметр разрешен, то доступ процессора к шине PCI позволен во время пассивного разделения. Необходимость запрещения данного параметра может возникнуть при использовании плат ISA, активно использующих каналы DMA. Может принимать значения:
    a. **Enabled** - разрешено
    b. **Disabled** - запрещено
13. **PCI Delayed Transaction** (задержанная транзакция на PCI) - Присутствие этого параметра означает, что на материнской плате

есть встроенный 32-битный буфер для поддержки удлинненного цикла обмена на PCI. Если этот параметр разрешен, то доступ к шине PCI разрешен во время доступа к 8-ми разрядным устройствам на шине ISA. Это существенно увеличивает производительность, так как цикл такого обращения на ISA занимает 50-60 тактов шины PCI. При установке в компьютер платы, не поддерживающей спецификации PCI 2.1, этот параметр следует запретить. Может принимать значения:

a. **Enabled** - разрешено
b. **Disabled** - запрещено

14. **Parallel Port Mode (ECP+EPP)** (режим работы параллельного порта) - параметр позволяет задать режимы работы параллельного порта в соответствии со стандартом IEEE 1284. Следует учитывать, что скорость обмена для некоторых устройств может быть существенно увеличена при правильной установке режима работы порта принтера, например, для внешних устройств хранения информации типа Iomega ZIP Drive LPT. Может принимать значения:

    a. **Normal** - обычный интерфейс принтера, также называется SPP
    b. **ECP** - порт с расширенными возможностями
    c. **EPP** - расширенный принтерный порт
    d. **ECP + EPP**- можно использовать оба режима

15. **Parallel Port Mode** (режим работы параллельного порта) - параметр аналогичен Parallel Port Mode (ECP+EPP), но с некоторыми расширениями. Дело в том, что существуют устройства, выполненные с отклонениями от стандарта IEEE 1284, например, некоторые платы от фирмы Xircom. Для совместимости с такими платами в некоторых BIOS существуют параметры установки версии варианта ECP+EPP порта. Какую версию выбрать - необходимо "выловить" из документации на подключаемое устройство или проверить экспериментально. Может принимать значения:

    a. **SPP** - обычный интерфейс принтера, также называется SPP
    b. **ECP** - порт с расширенными возможностями
    c. **EPP** - расширенный принтерный порт
    d. **EPP 1.9** - версия 1.9 исполнения интерфейса
    e. **EPP 1.7** - версия 1.7 исполнения интерфейса

16. **ECP DMA Select** (выбор канала DMA для режима ECP) - параметр появляется только при разрешении режима ECP или ECP+EPP в Parallel Port Mode (ECP+EPP). Для нормальной поддержки режима ECP требуется задействовать канал DMA, который выбирается из каналов 1 или 3. Может принимать значения:

    a. **1** - канал 1

b. **3** - канал 3

c. **Disabled** - запрещено использовать DMA

17. **Onboard PCI IDE Enable** (разрешение работы интегрированного контроллера IDE) - Этот параметр управляет разрешением/запрещением работы каждого из двух каналов контроллера IDE, установленного на материнской плате. Может принимать значения:

    a. **Primary** - разрешена работа только первого канала

    b. **Secondary** - разрешена работа только второго канала

    c. **Both** - разрешена работа обеих каналов

    d. **Disable** - запрещена работа обеих каналов

18. **Onboard FDC Controller** (разрешение работы контроллера накопителя на гибких дисках) - Этот параметр управляет разрешением/запрещением работы контроллера накопителя на гибких дисках, установленного на материнской плате. Может принимать значения:

    a. **Enable** - контроллер разрешен

    b. **Disable** - контроллер запрещен

i. Выбор режима работы каждого диска

Эти четыре параметра позволяют устанавливать режимы работы каждого диска индивидуально или разрешить BIOS автоматическую установку самого высокоскоростного режима для диска. Для каждого диска допустимые параметры одинаковы. Например, для IDE 0 Master Mode допустимые значения: 0, 1, 2, 3, 4 и AUTO. Параметр UDMA может иметь значение Auto или Disable.

Звуковые сигналы выдаваемые AMI BIOS при загрузке

Число сигналов	Описание проблемы	Решение
1	DRAM refresh failure	Вставьте память еще раз. Если не помогает, то это проблема с памятью.
2	Parity Circuit Failure	
3	Base 64K RAM failure	
4	System Timer Failure	Материнская плата неработоспособна
5	Processor	

	Failure	
6	Keyboard Controller / Gate A20 Failure	Вытащите и вставьте чип контроллера клавиатуры. Если не помогает, замените контроллер клавиатуры. Если и это не помогает, проверьте клавиатуру и ее кабель
7	Virtual Mode Exception Error	Материнская плата неработоспособна
8	Display Memory Read/Write Failure	Ошибка памяти видеоадаптера. Переисталлируйте видеокарту и память на ней. Если не помогает, замените видеокарту.
9	ROM BIOS Checksum Failure	Ошибка в микросхеме BIOS. Попробуйте вытащить и вставить заново этот чип. Если не помогает, необходимо перепрошить его содержимое или заменить микросхему.
10	CMOS Shutdown Register Read/Write Error	Материнская плата неработоспособна
1 длинный 2 коротких	Video failure	Переинсталлируйте видеокарту. Если не помогает, придется ее заменить
1 длинный 3 коротких	Video failure	
1 длинный	POST passed.	Все идет по плану

2.4.Сообщения об ошибках

Сообщение	Проблема	Решение
CH-2 Timer Error	Не фатально. Может быть вызвано переферией	
INTR #1 Error	Первый канал прерываний не прошел POST	Проверьте устройства, занимающие IRQ 0-7.
INTR #2 Error	Второй канал прерываний не прошел POST	Проверьте устройства, занимающие IRQ 8-15.
CMOS Battery State Low		Замените батарейку
CMOS Checksum Failure	Контрольная сумма данных в CMOS-памяти не сходится с вычисленной ранее	Запустите Setup
CMOS Memory Size	Размер основной памяти не соответствует числу,	Запустите Setup

Mismatch	записанному в CMOS-память	
CMOS System Optons Not Set	Данные в CMOS повреждены или отсутствуют	Запустите Setup
Display Switch Not Proper	Неправильно выставлен тип монитора (цветной или моно) на материнской плате	Переставьте джампер в правильное положение
Keyboard is locked ... Unlock it		Разблокируйте клавиатуру
Keyboard Error	Пороблема с клавиатурой	Проверьте соответствие типа клавиатуры (AT/XT) контроллеру. Можно попытаться отключить тестирование клавиатуры при загрузке в Setup
K/B Interface Error	Проблема с подсоединением клавиатуры к материнской плате	
FDD Controller Failure	BIOS не может связаться с контроллером гибких дисков	Проверьте подсоединение дисковода и его разрешенность на мультикарте
HDD Controller Failure	То же, но с винчестером	
C: Drive Error	Диск C не отвечает	Либо в Setup выставлен неправильный тип диска, либо диск не отформатирован, либо плохо подключен
D: Drive Error	То же самое, но с диском D	То же
C: Drive Failure	Диск C находится, но не работает. Очень серьезная проблема	
D: Drive Failure	То же про диск D	
CMOS Time and Date Not Set		Запустите Setup
Cache Memory Bad, do Not Enable Cache!		Действительно плохой кеш, придется заменить. Хотя сначала попробуйте просто

		перезагрузиться
8042 Gate-A20 Error!	Линия A20 контроллера клавиатуры не работает	Замените контроллер клавиатуры (8042)
Address Line Short	Проблема со схемой адресации памяти	Попробуйте перезагрузиться, (выключив компьютер и подождав секунд 30). Проблема может разрешиться сама
DMA #1 Error	Ошибка первого канала DMA	Может быть вызвана соответствующим переферийным устройством
DMA Error	Ошибка контроллера DMA	
No ROM Basic.	Система не может загрузиться. (например, невозможно найти операционную систему)	Установите загрузочный диск или измените его в Setup
Diskette Boot Failure	Не удается загрузиться с дискеты	
Invalid Boot Diskette	Аналогично, но дискета читается	
On Board Parity Error	Ошибка контроля четности	Может быть вызвана соответствующей переферией, занимающей адрес, указанный в сообщении об ошибке
Off Board Parity Error	Аналогично	То же
Parity Error	Аналогично	То же
Memory Parity Error at XXXX	Ошибка памяти	То же
I/O Card Parity Error at XXXX	То же	То же
DMA Bus Time-out	Устройство не отвечает в течении 7.8мкс	Проблема в платах расширения (попытайтесь найти ту плату, которая вызывает эту ошибку и замените ее)
Memory mismatch, run Setup		Установите в Setup Memory Relocation в Disable
EISA CMOS Checksum Failure	Не сходится контрольная сумма EISA CMOS, или села батарейка	
EISA CMOS Inoperational	Ошибка чтения/записи в CMOS RAM	Батарейка может быть плохой

Expansion Board not ready at Slot X	AMI BIOS не может найти плату в слоте номер X	Проверьте установку платы в этом слоте
Fail-Safe Timer NMI Inoperational	Ошибка таймера NMI	
ID information mismatch for Slot X	ID EISA-карты в слоте X не соответствует ID, записанному в CMOS RAM.	
Invalid Configuration Information for Slot X	Конфигурационная информация о карте в слоте X EISA некорректна	Запустите ECU
Software Port NMI Inoperational	Программный порт NMI не работает	
BUS Timeout NMI at Slot X	Карта в слоте X NMI не отвечает	
(E)nable (D)isable Expansion Board?		Выберите E для разрешения использования карты в слоте X NMI или D в противном случае
Expansion Board disabled at Slot X	Плата расширения в слоте X NMI недоступна	
Fail-Safe Timer NMI	Таймер NMI сгенерировал ошибку	
Software Port NMI	Генерируется программным портом NMI	

Звуковые сигналы выдаваемые Award BIOS при загрузке

Число сигналов	**Проблема**	**Решение**
1 длинный 2 коротких	Video error	Переинсталлируйте видеокарту. Проверьте видеопамять. Замените карту

Сообщения об ошибках

Сообщение	**Проблема**	**Решение**
CMOS BATTERY HAS FAILED	Села батарейка	Замените батарейку
CMOS CHECKSUM ERROR	Неправильная контрольная сумма CMOS. Данные в CMOS повреждены. Возможно,	Проверьте батерейку и замените ее в случае необходимости

	батарейка села	
DISK BOOT FAILURE, INSERT SYSTEM DISK AND PRESS ENTER	Не найден загрузочный диск	Загрузитесь с дискеты и проверьте системные файлы на жестком диске
DISKETTE DRIVES OR TYPES MISMATCH ERROR - RUN SETUP	Типы дисководов, фактически установленных в системе и их описания в CMOS не сходятся	Запустите Setup и введите правильный тип дисковода
DISPLAY SWITCH IS SET INCORRECTLY	Неправильно выставлен тип монитора (цветной или моно) на материнской плате	Переставьте джампер в правильное положение
DISPLAY TYPE HAS CHANGED SINCE LAST BOOT	Изменился тип дисплея со времени последней загрузки	Запустите Setup и введите правильный тип дисплея
EISA Configuration Checksum Error	Не сходится контрольная сумма энергонезависимой памяти EISA	Запустите программу настройки EISA
EISA Configuration is Not Complete	Информация в EISA – неполная	Запустите программу настройки EISA
ERROR ENCOUNTERED INITIALIZING HARD DRIVE	Не инициализируется жесткий диск	Проверьте установку контроллера жестких дисков и соединительные кабеля
ERROR INITIALIZING HARD DRIVE CONTROLLER	Контроллер жестких дисков не инициализируется	Проверьте установку контроллера, и параметры жесткого диска, указанные в Setup. Также проверьте джампера на жестком диске
FLOPPY DISK CNTRLR ERROR OR NO CNTRLR PRESENT	Невозможно инициализировать контроллер гибких дисков	Проверьте установку контроллера, и параметры дисковода, указанные в Setup
Invalid EISA Configuration	Данные о конфигурации EISA не верны	Запустите EISA Configuration Utility
KEYBOARD ERROR OR NO KEYBOARD PRESENT	Невозможно инициализировать клавиатуру	Проверьте подключение клавиатуры и ее тип. В крайнем случае отключите контроль клавиатуры при загрузке
Memory Address	Ошибка памяти	Замените память

Error at XXXX		
Memory parity Error at XXXX	Ошибка контроля четности	Замените память
MEMORY SIZE HAS CHANGED SINCE LAST BOOT	Размер памяти изменился со времени последней загрузки	Если есть EISA, запустите EISA Configuration Utility. В противном случае войдите в Setup
Memory Verify Error at XXXX	Ошибка при тестировании памяти	Замените память
OFFENDING ADDRESS NOT FOUND	Беда с контроллером памяти	
OFFENDING SEGMENT:	То же самое	
PRESS A KEY TO REBOOT	Сообщение возникает при обнаружении ошибок и необходимости перезагрузки	Нажмите любую кнопку
PRESS F1 TO DISABLE NMI, F2 TO REBOOT	Проблемы с немаскируемыми прерываниями	
RAM PARITY ERROR - CHECKING FOR SEGMENT	Ошибка четности в RAM	
Should Be Empty But EISA Board Found	ID одной из карт расширения не соответствует конфигурации	Запустите программу настройки EISA
Should Have EISA Board But Not Found	Карта не отвечает на запрос по ID	Запустите программу настройки EISA
Slot Not Empty	Слот, записанный в конфигурации как пустой, занят	Запустите программу настройки EISA
SYSTEM HALTED, (CTRL-ALT-DEL) TO REBOOT ...	Обозначает остановку процесса загрузки	Нажмите Ctrl-Alt-Del
Wrong Board in Slot	Установлена карта не с тем ID	Запустите программу настройки EISA

III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

В работе используется IBM – совместимый персональный компьютер.

Объектом исследования является BIOS ПК.

IV. ЗАДАНИЕ НА РАБОТУ

i. Ознакомиться с описанием системы BIOS, приведенном в данных методических указаниях.
ii. Изучить установки Вашей системы BIOS. Определить фирму-разработчика BIOS.
iii. Определить параметры используемого винчестера при помощи установок BIOS.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. изучить основные установки и функции системы BIOS;
2. включить ПК и войти в BIOS, установленный на Вашем компьютере;
3. определение параметров винчестера;
4. оформление отчета по лабораторной работе.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать :

1. описание основных установок системы BIOS, представленных в данных методических указаниях;
2. описание основных установок системы BIOS, установленной на Вашем ПК;
3. параметры винчестера.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое BIOS?
2. Как определить, что установленный на материнской плате BIOS, прошит во Flash ROM?
3. Зачем необходима перепрошивка новых версий BIOS?
4. Где можно найти новые версии BIOS?
5. Что делать, если производителя и название материнской платы определить не удается?
6. Что будет, если запортить BIOS или прошить неправильную версию?
7. Как сбросить установки BIOS (включая пароль) в значения по умолчанию из DOS?
8. Что означают аварийные сигналы, выдаваемые AMI BIOS при загрузке?
9. Что означают аварийные звуковые сигналы, выдаваемые Award BIOS при загрузке?

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.

2. Гук М. Аппаратные средства PC. Энциклопедия – СПб: Питер Ком,1998. –816 с.

Лабораторная работа № 3. Практикум прикладного программирования на языке ассемблер процессора Intel x86.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Изучение правил написания Паскаль программ, содержащих ассемблерные вставки и практическое знакомство с основными операторами языка Ассемблер-86, с целью освоения практических навыков составления и отладки простейших драйверов для управления нестандартным периферийным оборудованием.

II. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Основные операторы языка Ассемблер-86 для МП i8086 изучаются на аудиторных занятиях. Дополнительную информацию можно получить в работах /1,2/.

Листинг простой программы на языке Турбо-паскаль 7.0 с ассемблерной вставкой приведен на рис. 1.

```
program P_asm;
var
   x,y: INTEGER;

function ALU (XX,YY: INTEGER): Integer; assembler;
asm
   MOV     AX,XX
   ADD     AX,YY   ;{сложение кодов}
end;

begin
   readln(x,y);
   writeln(' результат: ', ALU ( x,y));
   readln;
end.
```

Рис. 1.

Приведенная программа выполняет суммирование переменных целого типа (длиной 2 байта) при помощи аккумулятора (регистра AX).

Управления клавиатурой при помощи программного прерывания 21H.

Клавиатура ПК содержит микропроцессор, который воспринимает каждое нажатие на клавишу и формирует соответствующий ей скан-код.

Скан-код - это однобайтовое число, младшие 7 бит которого представляют собой ASCII-код клавиши.

Код ASCII - это байтовые числа, соответствующие набору обычных символов пишушей машинки, а также ряд специальных букв и символов псевдографики.

Все необходимые клавиатурные операции можно выполнить с помощью команды вызова программного прерывания INT 21H. Конкретизация действия вызываемой функции осуществляется путем загрузки в регистры центрального микропроцессора соответствующих кодов.

Рассмотрим примеры использования указанного прерывания для работы с клавиатурой.

Функция 0C прерывания 21H выполняет функцию ввода с клавиатуры, причем перед этим чистит буфер клавиатуры. Для вызова функции необходимо поместить в AH код 0CH, а в AL - 1. Фрагмент программы приведен на рис.2.

```
;   ОЧИСТКА  БУФЕРА ПЕРЕД ОЖИДАНИЕМ НАЖАТИЯ КЛАВИШИ
            MOV  AH,0CH
            MOV  AL,1
            INT  21H    ;   ЧИСТИМ БУФЕР ЖДЕМ ВВОДА
```

Рис.2.

Листинг программы, использующей функцию ввода с ожиданием приведен на рис.3. Указанная программа выводит на экран терминала ASCII-код нажатой клавиши.

```
            program in_kl;
             uses crt;
              var
              data: byte;
            function inpb:byte ;
             begin
              asm
        ;   ОЧИСТКА БУФЕРА ПЕРЕД ОЖИДАНИЕМ НАЖАТИЯ КЛАВИШИ
            MOV  AH,0CH
            MOV  AL,1
            INT  21H    ;ЧИСТИМ БУФЕР ЖДЕМ ВВОДА
            MOV  @Result,AL
              end;
             end;

             begin
              clrscr;
                writeln('проверка ASCII кодов клавиш');
                writeln('выход осуществляется нажатием на клавишу <ESC> ');
              repeat
```

```
    data := inpb ;
    writeln('  значение ASCII кода нажатой клавиши =',data);
  until data = 27;
end.
```

Рис.3.

Программы, работающие в реальном времени (управляющие, игровые и т.д.), часто используют клавиши для управления процессом вычислений. При этом они должны принимать символ из буфера клавиатуры без ожидания нажатия клавиши. Например, алгоритм работы управляющей программы может быть представлен в виде (Рис.4.а):

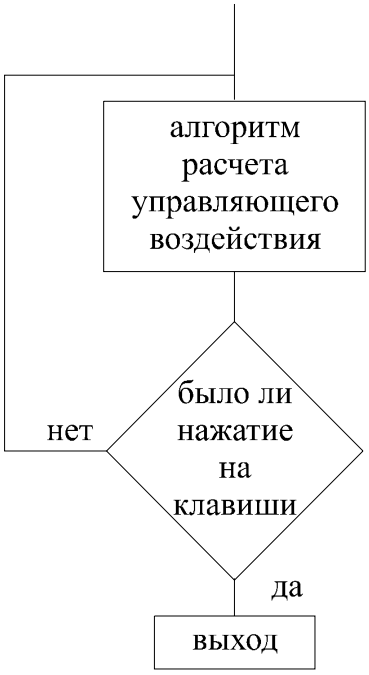


Рис.4.а

```
MOV  AH,6
MOV  DL,0FFH
INT  21H  ;
```

Рис.4.б

Введенный символ с клавиатуры можно получить при помощи функции 6 прерывания 21H. Эта функция не дает эха на экран. Перед вызовом прерывания в DL должно быть помещено 0FFH. Если символ принят, то он помещается в AL. Фрагмент программы приведен нарис. 4.б

III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

В работе используется персональный компьютер, на котором установлена интегрированная среда Турбо Паскаль 7.0.

IV. ЗАДАНИЕ НА РАБОТУ

1.Написать и отладить программу на языке Паскаль 7 с ассемблерной вставкой рассмотренную в примере 1.

Написать процедуру на языке Паскаль с ассемблерной вставкой выполняющей над двухбайтовыми переменными X1, X2, X3, X4 следующие действия:

1. Умножение двухбайтовых слов X1*X2;
   Сохранение результата в регистровой паре BX:CX;
   Пересылка третьего двухбайтового слова (X3) в аккумулятор;
   Расширение знака аккумулятора на регистр DX;
   Сложение полученного двойного слова с содержимым регистровой пары
   BX:CX;
   Вычесть из результата, хранимого в регистровой паре DX:AX;
   четвертую переменную X4;
   Поделить полученный результат на 10;
   Передать результат в вызывающую программу.

В среде Турбо-Паскаля ввести и отладить программу. Получить результат для следующих значений переменных:

X1= 300 , X2= 200 , X3=-32000 , X4=8000.

3. Написать процедуру на языке Паскаль с ассемблерной вставкой выполняющей над байтовой переменной X1 следующие логические действия:

Установка 0 и 2 бита в "1";
Инвертирование 1 и 6 бита;
Сбросить 2 и 3 биты в "0";
Инвертировать результат;
Передать результат в вызывающую программу.

В среде Турбо-Паскаля ввести и отладить программу. Получить результат для конкретного значения переменной.

4. Написать процедуру на языке Паскаль с ассемблерной вставкой выполняющей поиск наибольшего элемента массива целых переменных. Размерность массива - 10.

В среде Турбо-Паскаля ввести и отладить программу. Получить результат для конкретнх значений элементов массива.

5. Написать процедуру с ассемблерной вставкой, реализующей бесконечный цикл. Выход из цикла осуществляется при нажатии определенной клавиши, ASCII -код которой передается в процедуру из вызывающей программы.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. разработка алгоритма и составление программы, основной модуль которой написан на языке Паскаль, а процедуры/функции содержат ассемблерные вставки;
2. ввод и отладка программы;
3. проверка работы программы;
4. оформление отчета по лабораторной работе.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать :
- описание используемых операторов языка Ассемблер;
- описание способов работы с клавиатурой по прерыванию 21Н;
- алгоритмы и листинги Паскаль -программ с ассемблерными вставками.

VII. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите основные группы команд языка Ассемблер-86.
2. Работа с битами средствами языка Ассемблер –86.
3. Способы передачи параметров в ассемблерные вставки из основной программы, написанной на языке Паскаль в среде Турбо Паскаль 7.0.
4. Что такое программные прерывания?
5. Назовите программные прерывания, используемые для работы с клавиатурой.

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.
2. Абель П. Язык ассемблера для IBM PC и программирования: Пер с англ. Ю.В.Сальникова. М.:Высшая школа, 1992.-447с.
3. Интегрированная среда Турбо Паскаль 7.0.

Лабораторная работа № 4. Организация обмена информацией в МПС. Ввод-вывод данных во внешнее устройство.

II. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Изучение возможностей ввода/вывода цифровой информации в ПК с целью ее последующей обработки, знакомство с адресами портов ввода/вывода IBM PC/AT.

Изучение схемы и принципа работы интерфейсной платы на базе программируемого параллельного интерфейса КР580ВВ55.

II. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Одно из наиболее распространенных направлений применения ПК - сбор и обработка информации о состоянии датчиков, управление различными механизмами и технологическими системами. Типичная проблема, возникающая при этом: как ввести в ПК и вывести из него необходимые сигналы, число которых нередко достигает нескольких сотен. Часто приходится разрабатывать специальный блок, принимающий сигналы датчиков и преобразующий их в сигналы одного из стандартных интерфейсов, которыми оборудован ПК, например последовательного интерфейса RS-232C ("Стык-С2"). Обычно этот же блок решает и обратную задачу - преобразует сигналы стандартного интерфейса в вид, необходимый для управления исполнительными устройствами.

К сожалению такое решение имеет два существенных недостатка. Во-первых, стандартный интерфейс нередко бывает занят, например, связью с другими ПК, принтером и другими подобными устройствами. Во-вторых, необходимость постоянного приема и передачи большого числа сигналов через сравнительно медленный последовательный интерфейс может сильно повлиять на скорость работы системы в целом.

Во многих ПК, в том числе и в IBM PC/AT, предусмотрена возможность подключения дополнительных устройств к системной шине. Для этого на основной плате ПК установлены специальные розетки ("слоты"), в которые могут вставляться дополнительные платы, выполняющие функции, не предусмотренные исходной конфигурацией ПК. Скорость обмена данными по системной шине - максимально возможная для данного ПК и ограничивается в основном быстродействием процессора.

Для подключения дополнительных устройств к системной шине ПК необходимо знать распределение адресов портов ввода/вывода ПК. Распределение адресного пространства IOSEG для ПК IBM PC/AT показано в таблице 1.

Адрес	Описание
000-01F	DMA контроллер #1, 8237
020-03F	Контроллер прерываний 8259
040-05F	Таймер 8253 (AT: 8254)
060-06F	Контроллер клавиатуры 8042
070-07F	Реальные часы, КМОП-память, маски NMI, диагностическая контрольная точка
080-09F	DMA регистр страниц
0A0-0BF	Контроллер прерываний #2, 8259
0C0-0DF	DMA контроллер #2, 8237
0F0-0FF	Математический сопроцессор
170-177	Жесткий диск (вторичный)
1F0-1F8	Жесткий диск
200-207	Игровой адаптер
278-27F	Параллельный принтер #2 (#3 с DMA)
2CO-2DF	EGA #2
2F8-2FF	Порт #2 асинхронного адаптера
300-31F	**Плата прототипа**
360-36F	**Резерв**
378-37F	Параллельный принтер #1 (#2 с DMA)
380-38F	Адаптер синхронной связи #2 (SDLC)
3A0-3AF	Адаптер синхронной связи #1 (SDLC)
3B0-3BF	Монохромный адаптер и принтер #1
3C0-3CF	**Резерв**
3D0-3DF	CGA и EGA
3F0-3F7	Контроллер дискет
3F8-3FF	Порт #1 асинхронного адаптера

В приведенной таблице выделены свободные адреса, к которым может быть подключена дополнительная интерфейсная плата.

Принципиальная схема интерфейсной платы показана на рис.1.

Интерфейсная плата позволяет вводить/выводить из ПК до 24 логических сигналов. На микросхемах DD2-DD5 выполнен дешифратор, на который поданы сигналы A2-A15 шины адреса ПК. При выполнении ПК команд чтения из портов с адресами от 300H до 30FH или записи в те же порты на выводе _9 DD5 формируется импульс низкого логического уровня, разрешающий работу микросхемы DD4.

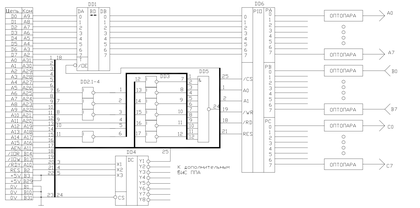

Шина ISA
D0 A9
D1 A8
D2 A7
D3 A6
D4 A5
D5 A4
D6 A3
D7 A2
A0 B31
A1 B30
A2 B29
A3 B28
A4 B27
A5 B26
A6 B25
A7 B24
A8 B23
A9 B22
A10 B21
A11 B20
A12 B19
A13 B18
A14 B17
A15 B16
/MEM_R A11
/MEM_W A10
/IOR_R B15
/IOW_R B14
RESET B2
-12V B3
+12V B4
0V B5
0V B20
0V B22
DD1
DD1:1-4
DD2
DD3
DD4
DD5
DD6
/CS
A0
A2
/WR
/RD
RESET
C0
C1
C2
C3
C4
C5
Y1
Y2
Y3
Y4
Y5
Y6
Y7
Y8
К дополнительным БиС ПЛА
ОПТОПАРА
A0
A7
B0
B7
C0
C7

Рис.1.

Разряды адреса A2, A3, A4 являются входными сигналами для дешифратора DD4, выходы которого непосредственно управляют выбором кристалла ППА (вход CS) и микросхемы DD1. Сигналы A0 и A1 подаются непосредственно на адресные входы ППА (DD6). Таким образом к порту A этой БИС можно обращаться по адресу - 300H; к порту B- по адресу 301H; к порту C - 302H, а к регистру управляющего слова - 303H.

Операции чтения/записи производятся по формируемым процессором ПК сигналам /IOR или /IOW. Однако в ПК эти сигналы могут быть сформированы не только МП, но и контроллером прямого доступа к памяти (ПДП). Для исключения сбоев на дешифратор подан сигнал AEN, блокирующий его работу при работе ПК в режиме ПДП.

Несколько слов о назначении шинного формирователя DD1. Если плату предполагается использовать только для вывода данных, то вполне можно обойтись и без этой микросхемы: буфер шины данных ПК имеет достаточный запас нагрузочной способности для управления непосредственно подключенной к нему шиной данных микросхемы DD5. Однако для передачи в обратном направлении нагрузочной способности этой микросхемы недостаточно, поэтому требуется мощный шинный формирователь.

Если требуется увеличить число выводов для подключения внешних устройств, на интерфейсной плате можно установить дополнительные микросхемы ППА, управление которыми можно осуществлять при помощи дешифратора DD4.

Для исключения влияния помех на работу ПК, в рассматриваемой интерфейсной плате осуществлена гальваническая развязка электрических цепей платы от электрических цепей подключаемого к ней периферийного оборудования при помощи оптопар DD7-DD30.

Схема включения оптопары показана на рис. 2.

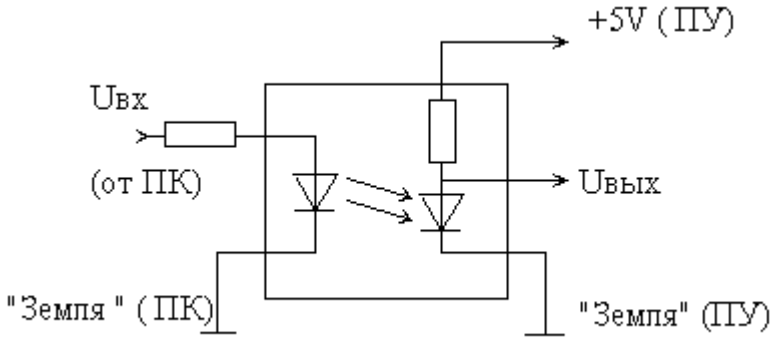


Рис.2.

ПРОГРАММИРУЕМЫЙ ПЕРИФЕРИЙНЫЙ АДАПТЕР.

Программируемый периферийный адаптер (ППА) КР580ВВ55 представляет собой однокристальное программируемое устройство параллельного ввода/вывода (ВВ) информации произвольного формата. В состав осуществляемых им функций входят параллельный обмен данными с квитированием или без него , как в режиме программного управления, так и по прерываниям. При этом обеспечивается возможность не только однонаправленного, но и двунаправленного ВВ. Определение и переопределение функций ППА выполняется программными методами с помощью процедур инициализации.

Структурная схема ППА показана на рис.3. В состав ППА входят три двунаправленных 8-ми разрядных порта, разбитых на две группы, два устройства управления группами портов и интерфейсная логика для согласования с системной магистралью.

ППА поддерживает три режима работы портов:
- режим 0 - однонаправленный ВВ без квитирования (применим к любому из трех портов);
- режим 1 - однонаправленный ВВ с квитированием (применим к портам А и В, порт С используется для передачи/приема сигналов управления);
- режим 2 - двунаправленный ВВ (допускается только для порта А, при этом совместно с портом А используется младшая тетрада порта С).

При работе портов А и В в режимах 1 и 2, часть линий порта С из соответствующей группы используется для управления обменом с внешними периферийными устройствами.

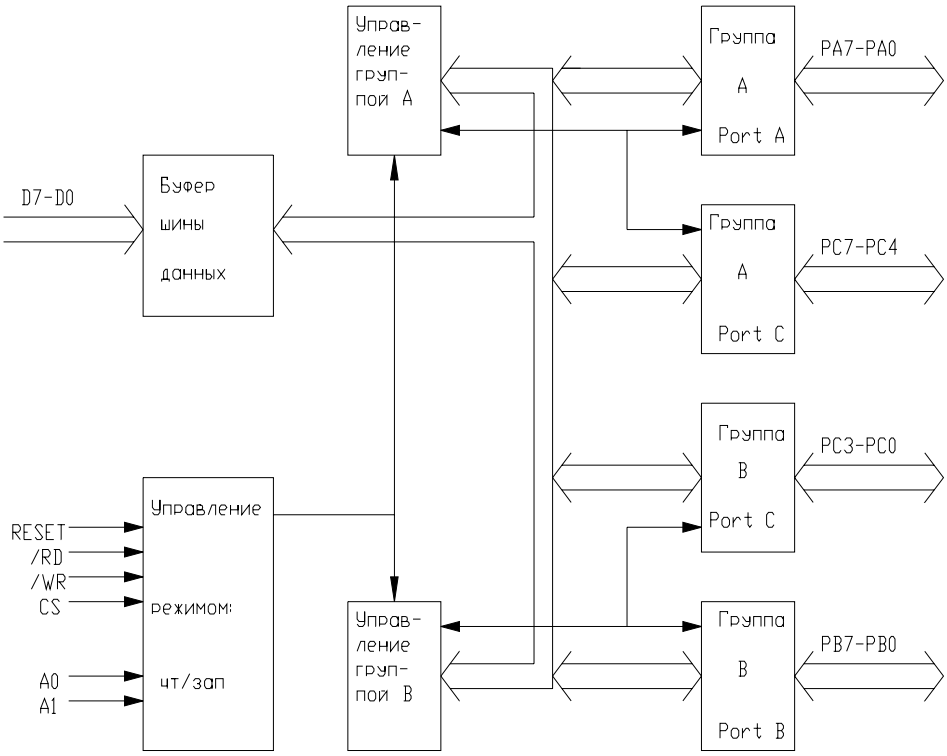


Рис.3.

Режим работы каждого из каналов ППА программируется при помощи управляющего слова, формат которого показан на рис.4.

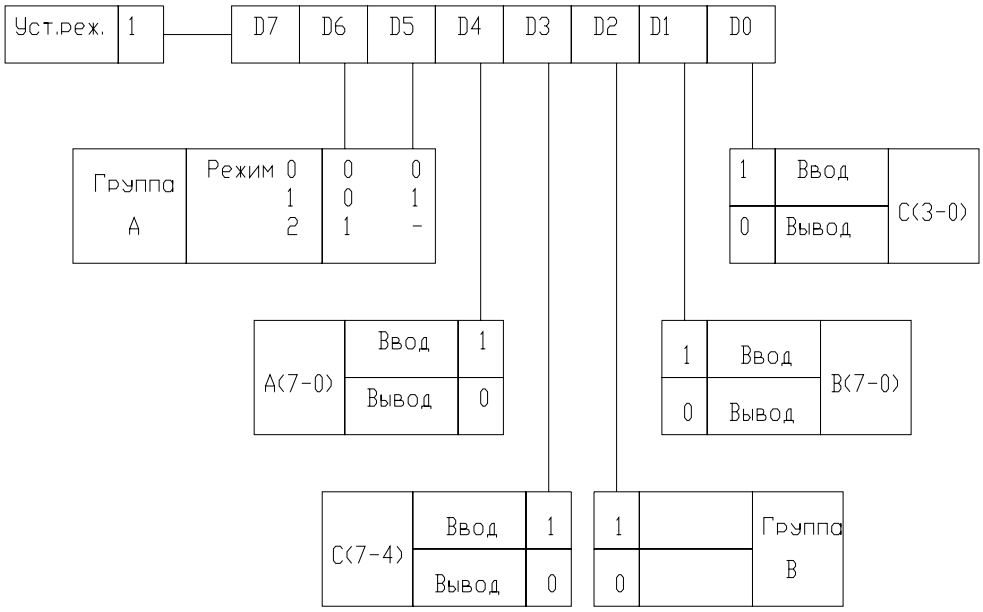


Рис.4.а

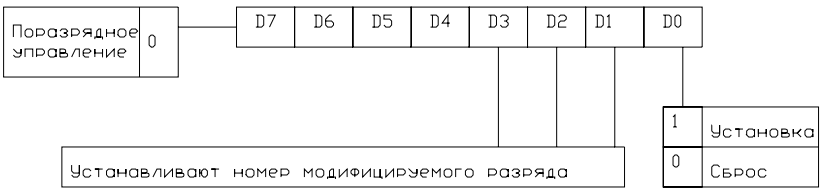


Рис.4.б

Разряд D7 управляющего слова определяет либо установку режимов работы его каналов (D7=1, рис.4.а), либо работу ППА в режиме установки/сброса отдельных разрядов порта С (D7=0 , рис.4.б).

ПРИМЕР записи кода управляющего слова для установки ППА (порт А - вывод, порт В - ввод, порт С - вывод, режим 0):

10000010B = 82H.

Вид операции, выполняемой БИС ППА, задается сигналами управления /RD, /WR, RESET, CS совместно с адресными линиями (таблица 2).

Таблица 2

A1	A0	адресуемый элемент
1	1	запись уравляющего слова из МП
0	0	- порт А
0	1	- порт В
1	0	- порт С

III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

В работе используется персональный компьютер, на котором установлена интегрированная среда Турбо Паскаль 7.0.

IV. ЗАДАНИЕ НА РАБОТУ

1. Изучить функциональную схему интерфейсного модуля.
2. Разработать алгоритм работы драйвера для управления интерфейсным модулем.
3. Написать и отладить драйвер для управления интерфейсным модулем. Оформить драйвер в виде ассемблерной вставки.
4. Составить отчет по лабораторной работе.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. изучение схемы и принципа работы интерфейсной платы;
2. разработка алгоритма и составление программы, основной модуль которой написан на языке Паскаль, а процедуры/функции содержат ассемблерные вставки, для программирования интерфейсной платы;
3. ввод и отладка программы;
4. оформление отчета по лабораторной работе.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать принципиальную схему и краткое описание интерфейсной платы, функциональную схему и описание ППА КР580ВВ55, текст программы с функцией, инициализирующей ППА на интерфейсной плате.

VII. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что понимается под интерфейсной платой?
2. Назовите области свободного адресного пространства ПК, которые могут быть использованы драйверами для управления интерфейсным модулем.
3. Назовите режимы работы ППА.
4. С какой целью в схему интерфейсного модуля включен шинный формирователь?
5. Зачем нужна оптронная развязка?

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.
2. Микропроцессорные автоматические системы регулирования. Основы теории и элементы./ Под ред. В.В. Солодовникова,- М., 1991.
3. Интегрированная среда Турбо Паскаль 7.0.

Лабораторная работа № 5. Организация управления объектом в реальном масштабе времени с помощью МПС.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Приобретение практических навыков разработки и отладки простейших драйверов для управления в режиме реального времени нестандартным периферийным оборудованием.

Разработка программы, содержащей функции с ассемблерными вставками для выполнения операций инициализации интерфейсной платы и ввода/вывода.

Написание функции с ассемблерной вставкой, выполняющей функцию управления объектом в реальном времени в соответствии с алгоритмом, заданным преподавателем.

а. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Средства цифровой вычислительной техники (СВТ) в настоящее время широко применяются для управления самыми различными техническими объектами. Это является следствием преимуществ цифровой обработки информации и достижений современной микроэлектроники.

Для реализации функций управления могут быть использованы либо специализированные цифровые вычислительные устройства, либо универсальные вычислительные машины, к которым в первую очередь относятся персональные компьютеры. Причем использование последних особенно удобно на этапе экспериментальной отработки синтезированных алгоритмов управления, поскольку персональные компьютеры имеют развитые средства для запоминания и отображения вводимой и выводимой информации, позволяют достаточно легко реализовать интерфейс пользователя.

Особенностями использования персональных компьютеров для реализации функций управления техническими объектами являются: необходимость реализации большого количество операций обмена данными с внешними устройствами, в том числе и нестандартными, операций предварительной обработки, сортировки данных и т.п., а также работа в режиме реального времени.

Работа в **режиме реального времени** означает, что информация, полученная от внешних устройств, должна быть не только правильно обработана, но и своевременно выдана на выходные устройства для воздействия на управляемый процесс. Интервал времени между моментами ввода информации называется **тактом квантования**. Величина его зависит от инерционности управляемого процесса и может составлять от нескольких десятков микросекунд до сотен миллисекунд.

Перечисленные выше особенности применения СВТ для управления процессами определяют необходимость составления компактных

управляющих программ с минимальным количеством операций, позволяющих работать с нестандартным периферийным оборудованием. Программирование таких задач на языках высокого уровня, как правило, не эффективно, а зачастую и невозможно. В данном случае наиболее рациональным является программирование в машинных кодах или на языке Ассемблера.

Машинно-ориентированный язык Ассемблера является специфичным для каждой ЭВМ и, вследствие этого, наиболее соответствующим ее архитектуре. Языки Ассемблера, хотя и уступают языкам программирования высокого уровня в наглядности и "прозрачности" описания алгоритмов, однако программы, полученные после трансляции с языков программирования высокого уровня на машинный язык, уступают по эффективности (время выполнения, требуемый объем памяти) аналогичным программам, полученным после трансляции с языка Ассемблера. Кроме того, ни один язык программирования высокого уровня не дает возможности управлять всеми аппаратными ресурсами вычислительной системы, что особенно необходимо при разработке управляющих программ, работающих в реальном времени.

И так, для отработки управляющих программ в режиме реального времени целесообразно использовать автоматизированный комплекс, в состав которого входят персональный компьютер, выполняющий функции универсального вычислителя и объект управления, принимающий управляющую информацию от вычислителя (ПК) и формирующий сигналы обратных связей в цифровой форме о своем состоянии.

В качестве объекта управления в данной работе используется лабораторный стенд, который позволяет:

- формировать байт информации и осуществлять его индикацию и ввод в порт ввода ПК с адресом 301Н;
- осуществлять индикацию байта, выводимого из ПК через порт с адресом 300Н;
- осуществлять преобразование байта, выведенного из ПК, в ШИМ-сигнал;
- контролировать с помощью осциллографа значения разрядов С0 и С7 порта С (адрес 302Н);
- проверять с помошью осциллографа работу преобразователя цифрового кода в ШИМ-сигнал.

Лабораторный стенд подключается при помощи соединительного кабеля к интерфейсной плате в ПК.

Вводимый байт формируется при помощи D-триггеров включенных по схеме со счетным входом. Индикация значений разрядов вводимого и выводимого байта выполняется при помощи светодиодов.

Вид передней панели стенда приведен на рис.1. На передней панели отображаются при помощи светодиодов вводимый в ПК и выводимый из ПК двоичные коды (разряды портов В и А БИС ППА, соответственно). Разряды 0 и 7 порта С подключены к гнездам С0 и С7 на лицевой панели, что позволяет контролировать их значения при помощи осциллографа. На гнезда

панели выведены также сигнал с модуля преобразователя цифрового кода в ШИМ-сигнал и “общий провод”.

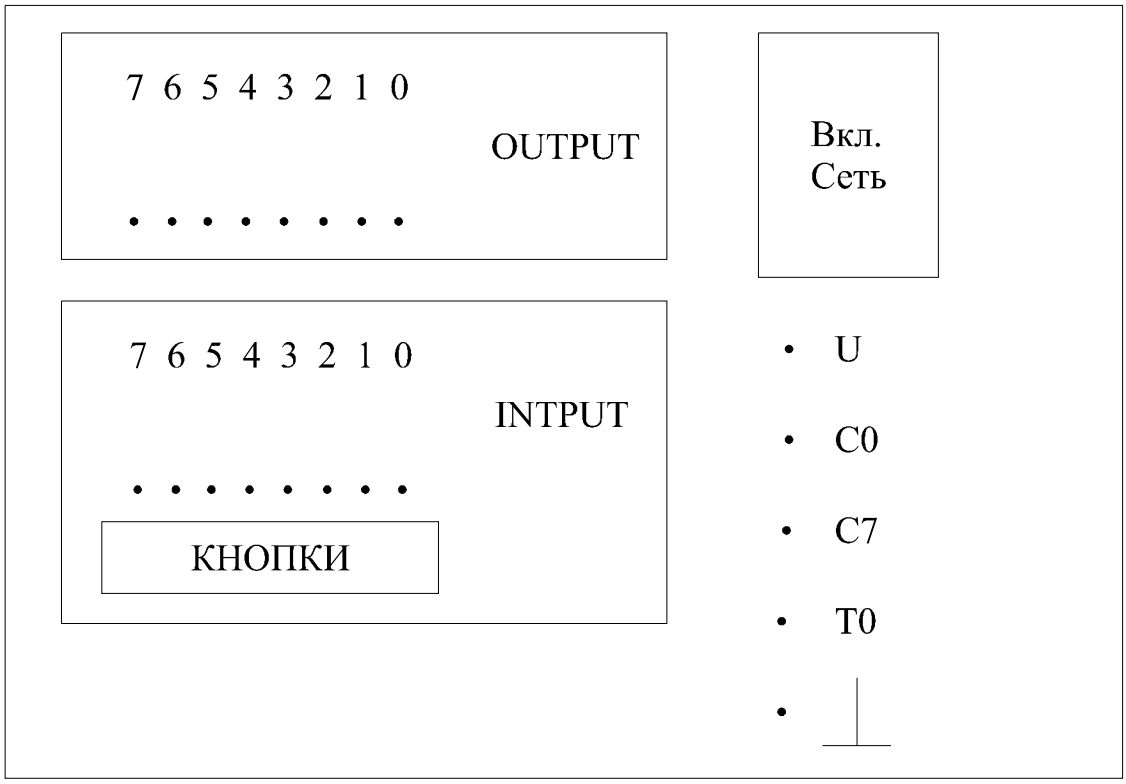


Рис.1.

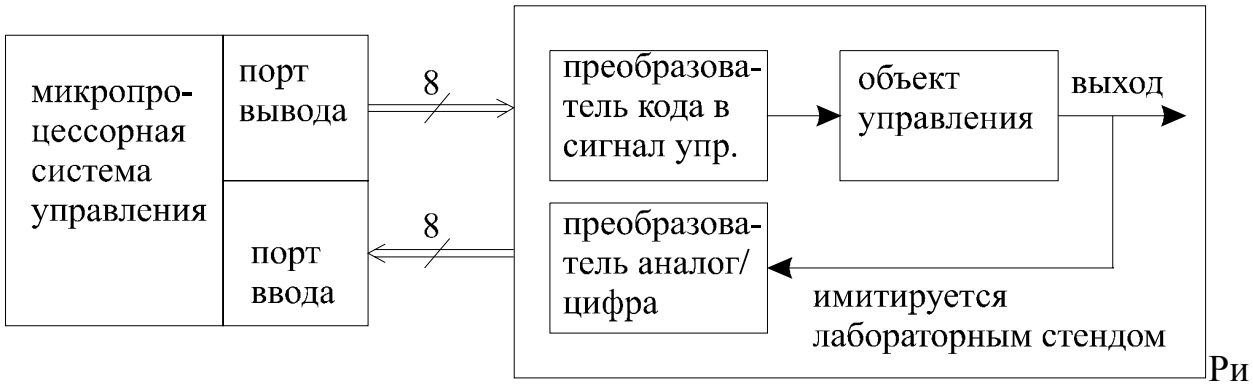
III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

В работе используется:

1. персональный компьютер, на котором установлена интегрированная среда Турбо Паскаль 7.0;
2. лабораторный стенд;
3. осциллограф С1-95.

IV. ЗАДАНИЕ НА РАБОТУ

Допустим, имеем микропроцессорную систему управления, функциональная схема которой представлена на рис.2.



Ри с.2.

Требуется:

1. Разработать подрограмму-функцию инициализирующую интерфейсную плату (по материалам лабораторной работы 3).

2. Разработать алгоритм и составить процедуру выполняющую в непрерывном цикле следующие действия:

- установку разряда C0 порта C в "1";
- установку разряда C0 порта C в "0";
- ввод байта из порта с адресом 301H ;
- инвертирование байта;
- вывод инвертированного байта в порт 300H.

Варианты условий выхода из цикла управления ко второму заданию:

- равенство контрольному значению значения 7-ого бита введенного кода. Контрольное значение (0 или 1) передается из вызывающей Паскаль-программы;

- равенство 1 значения контролируемого бита. Номер контролируемо бита передается из вызывающей Паскаль-программы;
- нажатие клавиши <1> на клавиатуре терминала (ASCII код клавиши <1> - 49).

3. Разработать алгоритм и составить программу выполняющую в непрерывном цикле управление объектом, признаком останова должно быть нажатие на клавишу <Esc> (ASCII код клавиши <Esc> - 27).

Функции программы:

- установку разряда C0 порта C в "1";
- установку разряда C0 порта C в "0";
- ввод байта из порта с адресом 301H;
- расчет управляющего воздействия в соответствии с заданным алгоритмом;
- вывод управляющего кода инвертированного байта в порт 300H.
- проверка условия выхода из цикла.

Варианты зависимости выходного кода цифрового управляющего устройства от входного кода представлены на рис.3, а соответствующие им алгоритмы расчета на рис.4.

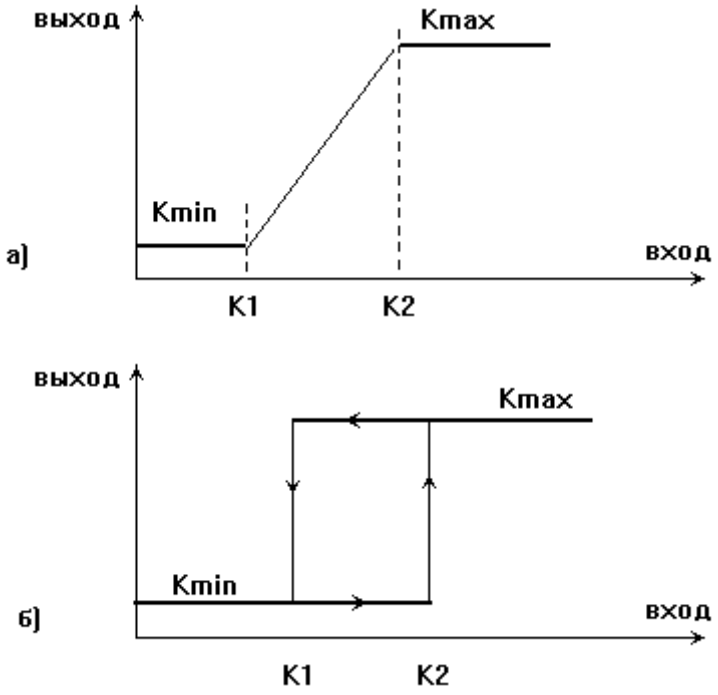
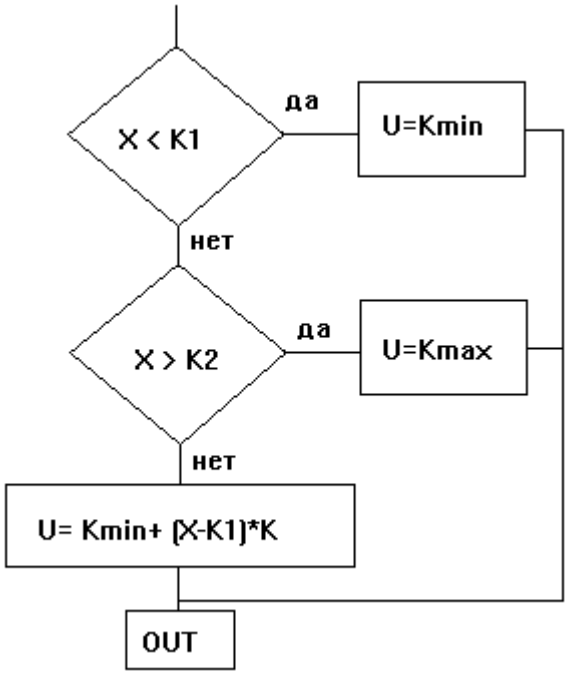


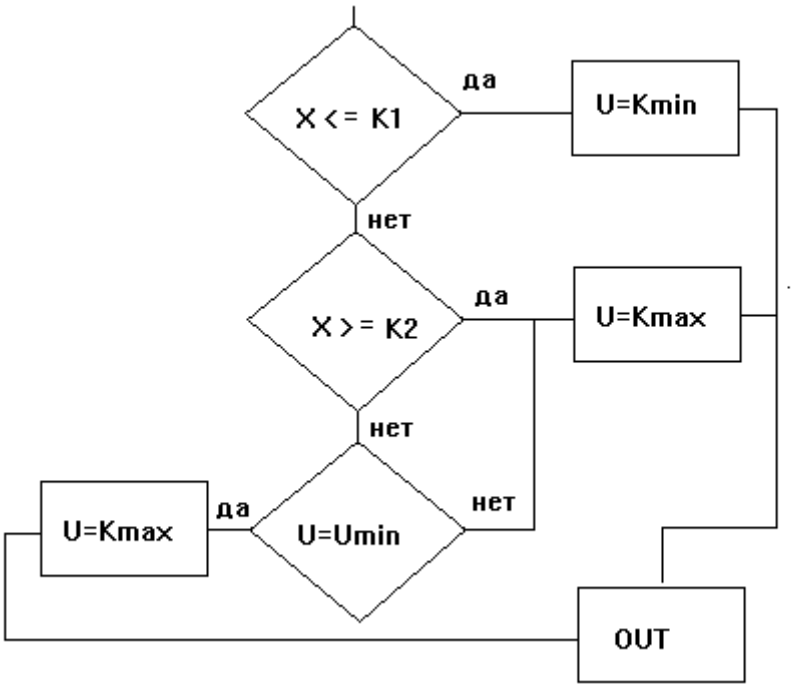
Рис.3.

Проверка работы программы.

Перед запуском программы необходимо подключить лабораторный стенд, имитирующий объект управления к ПК. Задавая код вводимого байта при помощи соответствующих кнопок и контролируя значения введенного и выведенного кода, убедиться в работоспособности управляющей программы . Подключить осциллограф к гнездам “C0” и “общий провод”. Изучить диаграмму сигналов на экране осциллографа. Определить период следования импульсов на выходе гнезда “C0” -



а)



б)

Рис.4.

1.Написать и отладить программу на языке Паскаль 7 с ассемблерной вставкой рассмотренную в примере 1.

Написать процедуру на языке Паскаль с ассемблерной вставкой выполняющей над двухбайтовыми переменными X1, X2, X3, X4 следующие действия:

1. Умножение двухбайтовых слов X1*X2;
   Сохранение результата в регистровой паре BX:CX;
   Пересылка третьего двухбайтового слова (X3) в аккумулятор;
   Расширение знака аккумулятора на регистр DX;

Сложение полученного двойного слова с содержимым регистровой пары
BX:CX;

Вычесть из результата, хранимого в регистровой паре DX:AX; четвертую переменную X4;

Поделить полученный результат на 10;

Передать результат в вызывающую программу.

В среде Турбо-Паскаля ввести и отладить программу. Получить результат для следующих значений переменных:

X1= 300 , X2= 200 , X3=-32000 , X4=8000.

3. Написать процедуру на языке Паскаль с ассемблерной вставкой выполняющей над байтовой переменной X1 следующие логические действия:

Установка 0 и 2 бита в "1";
Инвертирование 1 и 6 бита;
Сбросить 2 и 3 биты в "0";
Инвертировать результат;
Передать результат в вызывающую программу.

В среде Турбо-Паскаля ввести и отладить программу. Получить результат для конкретного значения переменной.

4. Написать процедуру на языке Паскаль с ассемблерной вставкой выполняющей поиск наибольшего элемента массива целых переменных. Размерность массива - 10.

В среде Турбо-Паскаля ввести и отладить программу. Получить результат для конкретнх значений элементов массива.

5. Написать процедуру с ассемблерной вставкой, реализующей бесконечный цикл. Выход из цикла осуществляется при нажатии определенной клавиши, ASCII -код которой передается в процедуру из вызывающей программы.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. разработка алгоритма и составление программы, основной модуль которой написан на языке Паскаль, а процедуры/функции содержат ассемблерные вставки;
2. ввод и отладка программы;
3. проверка работы программы;
4. оформление отчета по лабораторной работе.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать:

1. краткое описание лабораторного стенда;
2. алгоритм и программу управления объектом (по заданию преподавателя).
3. диаграммы сигналов снимаемых с клемм “С0” и “С7”.

VII. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите режимы работы портов БИС 580ВВ55.
2. Приведите пример команд реализующих побитные операции с цифровым кодом.
3. Что понимается под работой системы в режиме реального времени?
4. Каким образом в системе реализована шина управления?
5. Перечислите способы обмена информацией в цифровых системах.

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.
2. Абель П. Язык ассемблера для IBM PC и программирования: Пер с англ. Ю.В.Сальникова. М.:Высшая школа, 1992.-447с.
3. Интегрированная среда Турбо Паскаль 7.0.

Лабораторная работа № 6. Практикум прикладного программирования на языке ассемблер процессора AVR.

I. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Изучение правил написания программ и практическое знакомство с основными операторами языка Ассемблер для процессоров с ядром AVR, с целью освоения практических навыков составления и отладки простейших программ для управления объектами и оборудованием в реальном масштабе времени.

II. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Основные операторы языка AVR Ассемблер для МП AVR изучаются на аудиторных занятиях. Дополнительную информацию можно получить в работах [1, 2].

Среда программирования и отладки AVR Studio предназначена для написания и отладки программ на языках Ассемблер и Си для МП AVR, а также для осуществления загрузки кода программы в кристаллы микроконтроллеров семейства AVR и выполнения отладки программ на физическом объекте (в конечной системе).

Для осуществления операций в конечных системах в комплексе средств разработки имеются физические модули – программатор-модель конечной системы STK500 и отладчик – JTAG ICE.

Схематический вид программатора-модели конечной системы STK500 приведен на рис. 1.

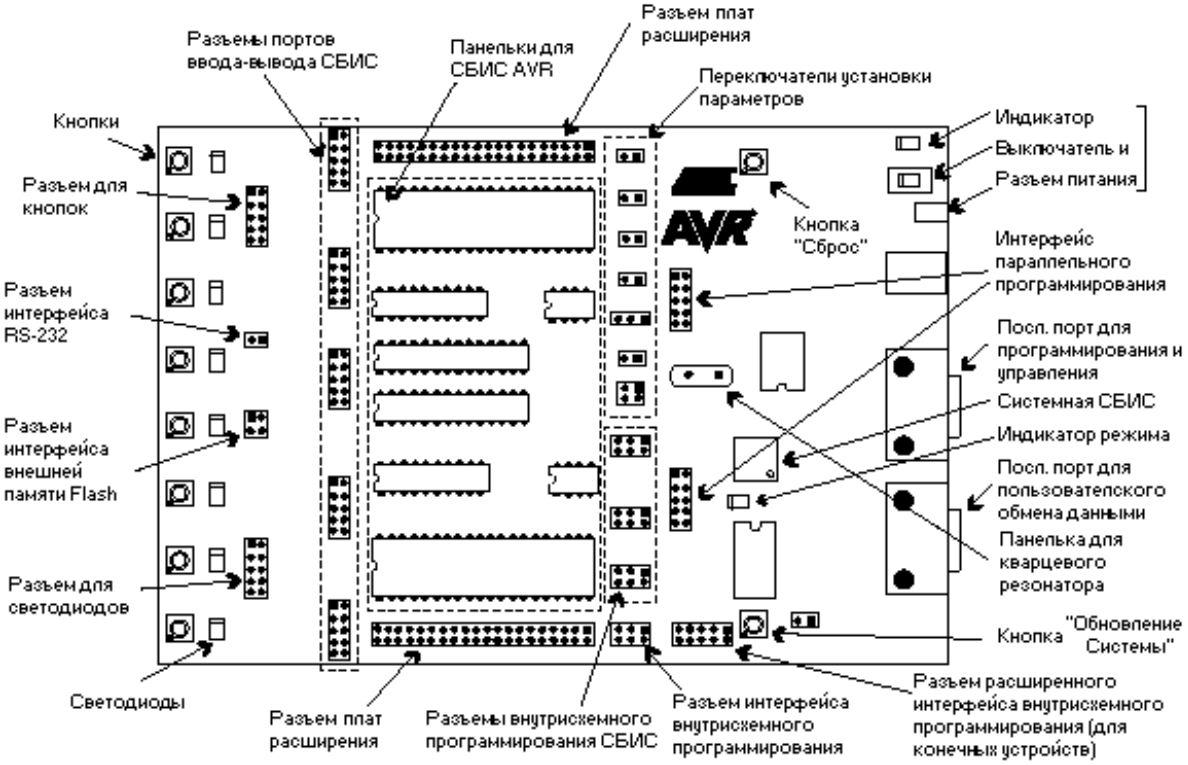


Рис. 1. Основные элементы STK500.

Структурная схема программатора-модели конечной системы STK500 приведен на рис. 2.

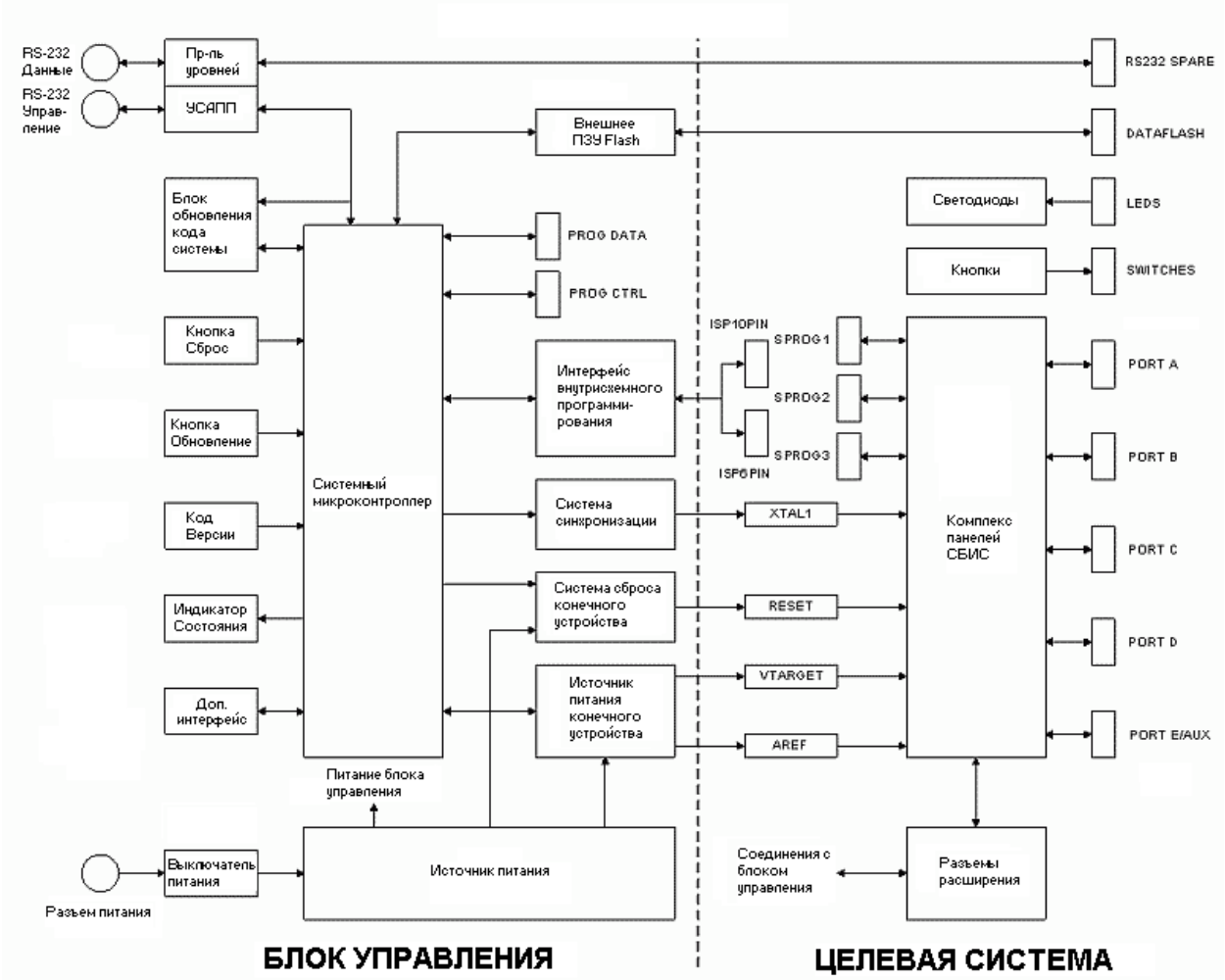


Рис. 2. Структурная схема STK500.

Целевая система STK500 содержит набор из восьми кнопок и восьми светодиодных индикаторов общего назначения, при помощи которых можно имитировать подсистемы ввода (кнопки) и вывода (индикаторы) реальных целевых устройств.

Схемное решение одного канала индикаторов показано на рис. 3.

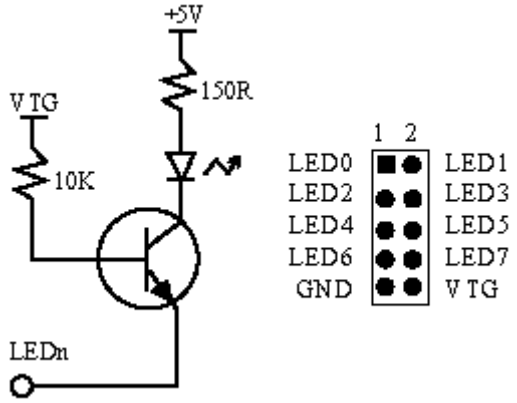


Рис.3. Схема канала индикации STK500.

Схемное решение одного кнопочного выключателя показано на рис. 4.

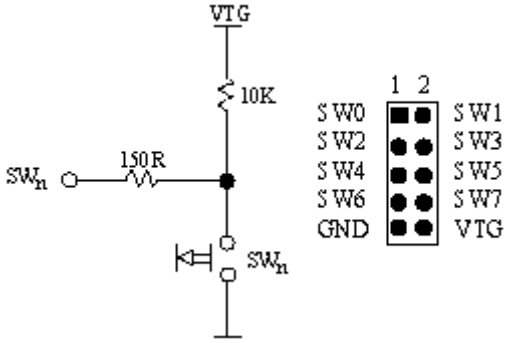


Рис.4. Схема канала кнопок STK500.

Структура целевой системы, используемая в дальнейшем для написания программ, приведена на рис.5.

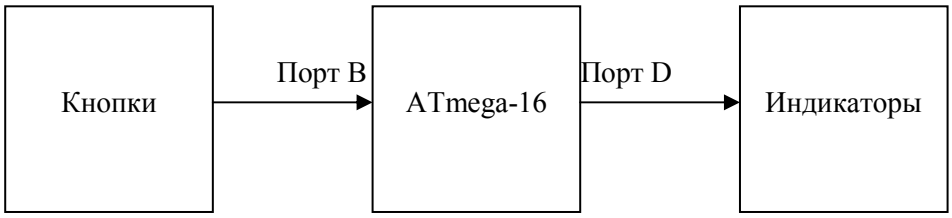


Рис. 5. Структура целевой системы.

Имеется в виду, что выводы порта D микроконтроллера подсоединены к группе индикаторов, а выводы порта B – к группе кнопок.

Листинг программы на языке Ассемблера в среде AVR Studio приведен на рис. 6.

```
;***** Пример 1 *****
;*******************

.include "m16def.inc"    ; Подключение определений для ATmega16

.dseg                    ; Пространство данных

Data:     .db       0
Keys:     .db       0

.cseg                    ; Пространство кода

          .org      0    ;Таблица адресов прерываний
          rjmp Start
          .org INT0addr; External Interrupt Request 0
          .org INT1addr; External Interrupt Request 1
          .org OC2addr; Timer/Counter2 Compare Match
          .org OVF2addr; Timer/Counter2 Overflow
          .org ICP1addr; Timer/Counter1 Capture Event
          .org OC1Aaddr; Timer/Counter1 Compare Match A
          .org OC1Baddr; Timer/Counter1 Compare Match B
          .org OVF1addr; Timer/Counter1 Overflow
```

```
          .org OVF0addr; Timer/Counter0 Overflow
          .org SPIaddr; Serial Transfer Complete
          .org URXCaddr; USART, Rx Complete
          .org UDREaddr; USART Data Register Empty
          .org UTXCaddr; USART, Tx Complete
          .org ADCCaddr; ADC Conversion Complete
          .org ERDYaddr; EEPROM Ready
          .org ACIaddr; Analog Comparator
          .org TWIaddr; 2-wire Serial Interface
          .org INT2addr; External Interrupt Request 2
          .org OC0addr; Timer/Counter0 Compare Match
          .org SPMRaddr; Store Program Memory Ready

          .org 0x100

Start:
          ldi       r16, low(RamEnd)    ;* –Действие 1
          ldi       r17, high(RamEnd)   ;*
          out       SPL, r16            ;*
          out       SPH, r17            ;*

          ldi       r16, 0xFF           ;* –Действие 2
          out       DDRD, r16           ;*

          clr       r16                 ;* –Действие 3
          out       DDRB, r16           ;*

          clr       r16                 ;* –Действие 4
          com       r16                 ;*
          sts       Data, r16           ;*

Label1:
          in        r16, PINB           ;* –Действие 5
          sts       Keys, r16           ;*

          lds       r16, Keys           ;* –Действие 6
          out       PORTD, r16          ;*

          rjmp Label1                   ;* –Действие 7

.exit
```

Рис. 6.

Приведенная программа выполняет пересылку данных из порта B в порт D.

Написание процедуры-обработчика прерывания.

Листинг программы на языке Ассемблера в среде AVR Studio приведен на рис. 7.

```
;***** Пример 2 *****
;********************
```

```
.include "m16def.inc"



.dseg

Data:     .db       0
Keys:     .db       0
Data2:    .dw       0

.cseg

          .org      0
          rjmp Start
          .org INT0addr; External Interrupt Request 0
          .org INT1addr; External Interrupt Request 1
          .org OC2addr; Timer/Counter2 Compare Match
          .org OVF2addr; Timer/Counter2 Overflow
          rjmp Timer2Owf
          .org ICP1addr; Timer/Counter1 Capture Event
          .org OC1Aaddr; Timer/Counter1 Compare Match A
          .org OC1Baddr; Timer/Counter1 Compare Match B
          .org OVF1addr; Timer/Counter1 Overflow
          rjmp Timer1Owf
          .org OVF0addr; Timer/Counter0 Overflow
          .org SPIaddr; Serial Transfer Complete
          .org URXCaddr; USART, Rx Complete
          .org UDREaddr; USART Data Register Empty
          .org UTXCaddr; USART, Tx Complete
          .org ADCCaddr; ADC Conversion Complete
          .org ERDYaddr; EEPROM Ready
          .org ACIaddr; Analog Comparator
          .org TWIaddr; 2-wire Serial Interface
          .org INT2addr; External Interrupt Request 2
          .org OC0addr; Timer/Counter0 Compare Match
          .org SPMRaddr; Store Program Memory Ready

          .org 0x100

Timer1Owf:
          in        r15, SREG

          in        r18, PINB
          lds       r19, Keys
          cp        r18, r19
          breq t_1
          sts       Keys, r18
          lds       r19, Data
          com       r18
          eor       r19, r18
          sts       Data, r19
```

```
t_1:
          lds       r19, Data
          mov       r18, r19
          ror       r18
          ror       r19
          sts       Data, r19

          out       SREG, r15
          reti

Timer2Owf:
          in        r15, SREG

          lds       r18, Data2
          lds       r19, Data2+1
          clr       r0
          ldi       r20, 22
          sbrs r18, 0
          add       r18, r20
          sbrs r18, 0
          adc       r19, r0
          sbrc r18, 0
          sub       r18, r20
          sbrc r18, 0
          sbc       r19, r0
          cpi       r19, 0
          brne t2_1
          cbr       r18, 1
t2_1:
          cpi       r19, 0xff
          brne t2_2
          sbr       r18, 1
;         lds       r0, Data
;         ldi       r20, 0x80
;         eor       r0, r20
;         sts       Data, r0
t2_2:
          sts       Data2, r18
          sts       Data2+1, r19
          out       OCR2, r19

          out       SREG, r15
          reti

Start:
          ldi       r16, low(RamEnd)    ;* — Действие 1
          ldi       r17, high(RamEnd)   ;*
          out       SPL, r16            ;*
          out       SPH, r17            ;*

          ldi       r16, 0xFF           ;* — Действие 2
          out       DDRD, r16           ;*
```

```
          clr       r16                 ;* – Действие 3
          out       DDRB, r16           ;*

          ldi       r16, 0              ;* – Действие 4
          out       TCCR1A, r16         ;*
          ldi       r16, 3              ;*
          out       TCCR1B, r16         ;*
          ldi       r16, 0              ;*
          out       OCR1AL, r16         ;*
          ldi       r16, 128            ;*
          out       OCR1AH, r16         ;*
          in        r16, TIMSK          ;*
          andi r16, 0xC3                ;*
          ori       r16, 0x4            ;*
          out       TIMSK, r16          ;*

          ldi       r16, 0x7a           ;* – Действие 4a
          out       TCCR2, r16          ;*
          ldi       r16, 0              ;*
          out       OCR2, r16           ;*
          ldi       r16, 0              ;*
          out       ASSR, r16           ;*
          in        r16, TIMSK          ;*
          andi r16, 0x3F                ;*
          ori       r16, 0x40           ;*
          out       TIMSK, r16          ;*

          clr       r16                 ;* – Действие 5
          com       r16                 ;*
          sts       Data, r16           ;*

          in        r16, PINB           ;* – Действие 6
          sts       Keys, r16
          sei

Label1:
          lds       r16, Data
          out       PORTD, r16          ;* – Действие 7

          rjmp Label1                   ;* – Действие 8

.exit
```

Рис. 7.

Данная программа осуществляет преобразование введенных данных с блока переключателей и отображает данные комплектом индикаторов.

III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

В работе используются:

- Персональный компьютер, на котором установлена среда программирования и отладки AVR Studio 4.0.

- Набор для программирования и отладки семейства микроконтроллеров AVR – STK500.
- СБИС микроконтроллера серии AVR – ATmega-16.
- Соединительные кабели.

IV. ЗАДАНИЕ НА РАБОТУ

1. Написать и отладить программу на языке ассемблер, рассмотренную в примере 1.

1.1. Модифицировать программу в примере 1, таким образом, чтобы инвертировать все биты выводимых данных.

1.2. Модифицировать программу в примере 1, таким образом, чтобы инвертировать биты 0-3 выводимых данных.

1.3. Модифицировать программу в примере 1, таким образом, чтобы инвертировать биты 0, 2, 4, 6 выводимых данных.

1.4. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные содержали дополнительный код входных данных.

1.5. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные содержали в битах 0-4 сумму бит 0-3 и 4-7 входных данных.

1.6. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные содержали в битах 0-7 произведение бит 0-3 и 4-7 входных данных.

1.7. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные сменяли друг-друга с периодом 0.25 или 0.5 или 1 с.

1.8. Прокомментировать построчно текст программы.

2. Написать и отладить программу на языке ассемблер, рассмотренную в примере 2.

2.1. Разобраться с назначением структурных единиц кода в примере 2.

2.2. Прокомментировать построчно текст программы.

2.3. Модифицировать программу в примере 2, таким образом, чтобы инвертировать все биты выводимых данных.

2.4. Модифицировать программу в примере 2, таким образом, чтобы сменить направление вращения бит выводимых данных.

2.5. Модифицировать программу в примере 2, таким образом, чтобы удвоить период вращения бит выводимых данных.

2.6. Модифицировать программу в примере 2, таким образом, чтобы удвоить период мерцания индикатора LED7.

3. Написать программу на языке ассемблер, выполняющую над байтовой переменной, введенной из порта B следующие логические действия:

Установка 0 и 2 бита в "1";
Инвертирование 1 и 6 бита;
Сбросить 2 и 3 биты в "0";

Инвертировать результат;
Передать результат в порт D.

4. Написать на языке ассемблер процедуру, выполняющую поиск наибольшего элемента в массиве целых, адрес которого передается в нее в регистровой паре Z. Размерность массива - 100. Элемент массива – байт. Результат возвращается в регистре R0

4.1. Отладить программу и выполнить ее для конкретных значений массива.

4.2. Модифицировать программу в по п.4. для элемента массива размером два байта. Результат возвращать в регистровой паре R1:R0.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Разработка алгоритма и составление программы, на языке AVR Ассемблер;
2. Ввод и отладка программы;
3. Проверка работы программы;
4. Модификация программы согласно заданию;
5. Оформление отчета по лабораторной работе.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать:

- тексты программ на языке Ассемблер для МП AVR;
- описание используемых операторов языка Ассемблер;
- описание способов изменения такта квантования в системе;
- структуру целевой системы, для которой составлялись программы;
- построчный комментарий всех приведенных программ.

VII. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите основные группы команд языка AVR Ассемблер.
2. Работа с битами средствами языка AVR Ассемблер.
3. Что такое программные и аппаратные прерывания?
4. Какими способами можно изменять такт квантования сигналов в системе?

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.
2. Система команд МП AVR — Atmel: Doc.Rev. 0856D-AVR-08/02.- 148с.

3. Микроконтроллер ATmega-16 — Atmel: Doc. Rev. 2466H-AVR-12/03. - 349с.

Лабораторная работа № 7. Изучение архитектуры вычислительной сети.

I. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Знакомство с архитектурой современных вычислительных сетей. Получение практических навыков по организации, настройке и мониторингу функционирования локальной вычислительной сети (ЛВС) на примере ЛВС дисплейного класса лаборатории.

II. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1. Компоненты сети и виртуальные соединения.

Компьютерная сеть состоит из двух основных компонентов: прикладных программ и подсистем передачи данных между ними. Все компьютеры общаются между собой используя двоичное кодирование. При написании программ двоичный код не используется. Язык программирования является своего рода абстракцией от излишних подробностей двоичного кода. Точно также при взаимодействии компьютеров понятие протокола является абстракцией, защищающей программиста от излишних подробностей в компьютерной сети.

Когда два компьютера в сети общаются друг с другом, можно считать, что происходит обмен данными определенного уровня во взаимодействующих процессах:

прикладной	←——→	прикладной
представления	←——→	представления
сеансовый	←——→	сеансовый
транспортный	←——→	транспортный
сетевой	←——→	сетевой
канальный	←——→	канальный
физический	←——→	физический

Такие уровни принято называть равноправными. Сообщения между равноправными процессами виртуальны по своей природе. На самом деле реальная передача данных происходит только на физическом уровне. Концепция виртуального соединения создана для облегчения программирования в сети.

2. Сетевые службы и протоколы.

Каждый сетевой уровень подчиняется определенному сетевому протоколу, определяющему набор сетевых служб, присущих данному уровню.

Сетевая служба - это набор функций, которые уровень выполняет для выше лежащего уровня.

Протокол уровня определяет структуру данных и формат пакетов для выполнения запрашиваемой сетевой службы.

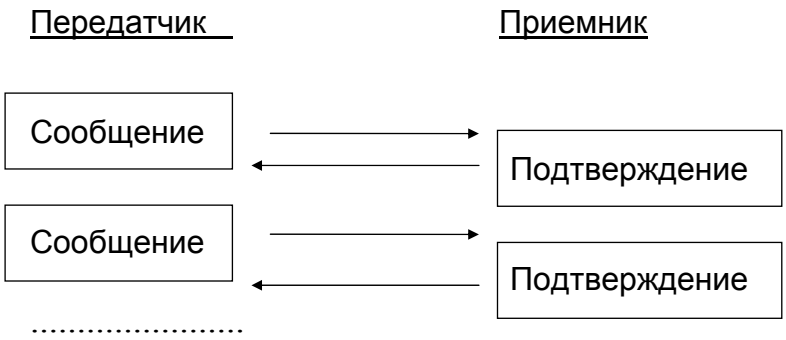
В рамках одной и той же сети для обеспечения одной и той же сетевой службы могут применяться различные методы передачи данных. Эти методы называются режимами (коррекция ошибок, подтверждение получения пакетов и т.д.). Если требуется, чтобы сетевая служба выполняла некоторую функцию, нужно выбрать соответствующий протокол или разработать собственный. Сеть обеспечивает сетевую службу, если требуемый для службы протокол доступен.

Различают сетевые службы, ориентированные и не ориентированные на соединение.

Любое сетевое соединение подразумевает наличие пути между двумя устройствами и наличие самих этих устройств. Двухточечное соединение - это непрерывная цепь между двумя устройствами. Ориентированная на соединение служба устанавливает виртуальное двухточечное соединение (например, телефон). Не ориентированная на соединение служба двухточечного соединения не устанавливает (доставка письма).

Процесс передачи данных по сети не застрахован от ошибок. Поэтому в сетях используется виртуальная коррекция ошибок. Ошибочные пакеты достигают сетевых модулей, но не достигают приложений.

Система контроля ошибок обязана обнаруживать и обрабатывать два вида повреждения данных: искажение и исчезновение. Для этого используются методы CRC и подтверждение получения пакета:



Некоторые протоколы коррекции ошибок могут посылать сообщения о недоставке пакета (например, если получен ошибочный пакет). Эти функции реализованы в протоколах Internet.

Межсетевое соединение двух и более сетей называют internet, что означает технологию взаимодействия сетей.

Термин Internet означает самое большое и самое популярное межсетевое объединение в мире. Оно объединяет более 20000 сетей, расположенных более чем в 130 странах мира.

Сообщения, посылаемые по сети, можно рассматривать как поток данных. Для управления непрерывным потоком данных в сети без ограничения размеров этого потока имеется сетевая служба flow control. Она обеспечивает синхронизацию работы передатчика и приемника данных.

Протоколы, обеспечивающие сетевые службы, могут действовать как на концах соединения, так и на промежуточных пунктах..

Передача данных в сетях может выполняться по двум принципам - с коммутацией каналов или коммутацией пакетов. Internet является сетью с коммутацией пакетов. Коммутатор пакетов называется hop (скачок). Служба промежуточной доставки (hop-by-hop service) выполняет определенные функции в месте каждой остановки пакета на его пути. Служба конечной доставки (end-to-end service) работает только в пунктах передачи и приема сообщений.

Одной из проблем, возникающих при передаче данных на каждом уровне сетевой модели OSI, является переполнение данных. Проблема может возникнуть как между равноправными процессами в соединенных компьютерах, так и при передаче данных между уровнями в одном и том же компьютере.

Сетевые протоколы делят данные на пакеты, но их длина может оказаться слишком большой для некоторого участка сети. Поэтому сеть должна уметь делить данные на пакеты еще меньшей длины. Процесс деления пакета данных на пакеты меньшей длины называется фрагментацией. Это приводит к ухудшению работы сети. Проблемы фрагментации и дефрагментации данных решаются на сетевом уровне и прикладная программа может не знать об этих процессах.

3. Сетевые уровни модели взаимодействия открытых систем.

Для построения модели взаимодействия открытых систем определены пять основных принципов:

1. 1. Новый уровень вводится, если программное обеспечение требует нового уровня абстракции.
2. 2. Каждый уровень должен выполнять строго определенную функцию.
3. 3. Набор функций, выполняемых сетевым уровнем, приводится в соответствие с общепринятыми международными стандартами.
4. 4. Границы уровня выбираются таким образом, чтобы сделать поток данных между уровнями минимальным.
5. 5. Количество сетевых уровней выбирается достаточно большим, чтобы не заставлять разработчиков размещать различные функции на одном уровне. Но слишком большое число уровней может привести к усложнению архитектуры сети.

В модели OSI уровни 7,6,5,4 в качестве единицы измерения информации используют «сообщение». Сетевой уровень трактует данные как пакеты. Канальный уровень оперирует с кадрами. Физический уровень имеет дело с байтами.

Сеть Internet состоит из сетей, которые легче всего описывать в терминах модели OSI/ISO.

Физический уровень - посредник в передаче данных по каналам связи. Он определяет тип кабеля, топологию сети, количество контактов на разъемах, параметры электрических сигналов и т.д.).

Канальный уровень (уровень соединения) передает данные от физического уровня к сетевому, превращая их в кадры и наоборот.

Заголовок	Адрес получ.	Адрес ист.	Тип кадра	Данные	CRC
64 бит	48 бит	48 бит	16 бит	368...1200	32 бит

Кадр Ethernet

Сетевой уровень определяет путь следования данных посети, позволяя им найти получателя. Это значит, что он определяет скорость передачи по сети и контроль целостности данных. Этот уровень можно рассматривать как службу доставки. Данные доставляются в виде отдельных пакетов, каждый из которых содержит адреса отправителя и получателя (в целях маршрутизации пакета).

Сетевой уровень служит интерфейсом между компьютерами и коммутаторами пакетов. Кроме того он следит за правильностью полученных пакетов.

Для маршрутизации данных в сети используется таблица маршрутизации. Это база данных, где описывается местонахождение возможных получателей пакетов. Используя такую таблицу, маршрутизатор в состоянии найти путь пакета для любого получателя в сети.

Таблица маршрутизации может быть статической или динамической. В статической таблице информация обновляется оператором. В динамической - различными программами при запуске каждого нового сеанса или появлении нового пакета маршрутизации.

Подключение новых компьютеров к сети приводит к возрастанию потока пакетов данных - трафика (traffic) через нее.

Сетевой уровень контролирует поток данных при маршрутизации пакетов.

В современном мире сети часто пересекают геополитические границы. При этом возникает необходимость учитывать трафик на разных участках сети для решения вопроса оплаты. Информация о трафике выдается сетевым уровнем.

Транспортный уровень передает данные между компьютерами, доставляя их прикладному процессу. На рис.3 показано взаимодействие нескольких прикладных процессов П1, П2, П3, запущенных на двух компьютерах.

Если обмен данными по сети на сетевом уровне выполняется скачками, то на транспортном уровне по виртуальному соединению данные передаются по цепи компьютер - компьютер.

В сети с коммутацией пакетов транспортный уровень должен фрагментировать пакеты, поступающие с сеансового уровня, чтобы передать их дальше на сетевой уровень. От транспортного уровня зависит количество

пакетов в сети. Т.е. транспортный уровень генерирует трафик пакетов данных, которым должен управлять сетевой уровень.

На одном компьютере может быть запущено несколько процессов. Для увеличения пропускной способности сети транспортный уровень открывает несколько сетевых соединений. С этой целью выполняются соответственно операции мультиплексирования и демультиплексирования.

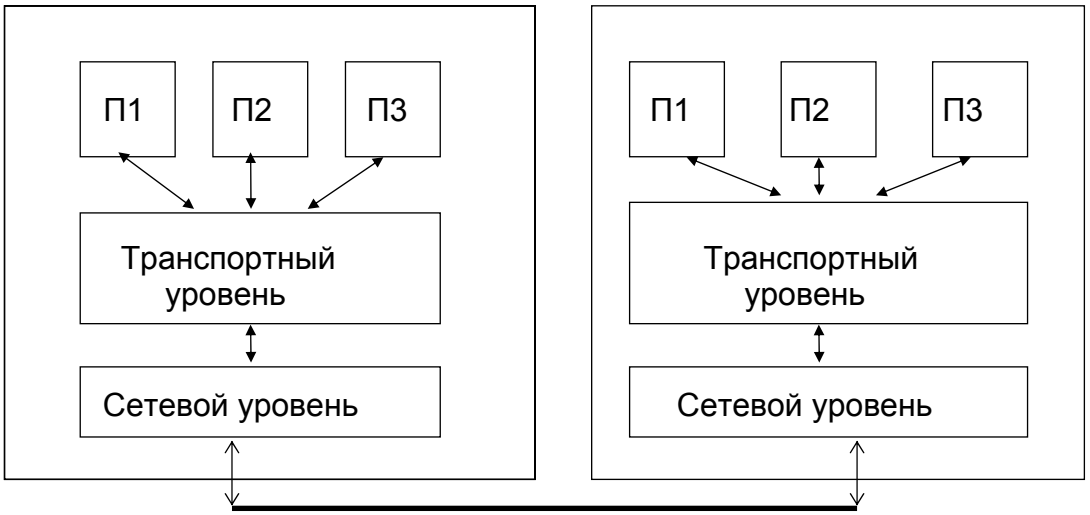


Рис. 3. Взаимодействие прикладных процессов

Сеансовый уровень в качестве пользовательского сетевого интерфейса решает задачи по обработке соединений между процессами и приложениями на различных компьютерах (обработка паролей, прав доступа и т.д.). Протоколы транспортного уровня Internet выполняют функции сеансового уровня модели OSI.

Уровень представления объединяет общие функции, которые Сеть неоднократно использует для сетевых соединений. Этот уровень образует интерфейс между сетью и устройствами компьютера. Т.е. он представляет, как сеть выглядит с точки зрения программного обеспечения и аппаратуры. Уровень может выполнять сжатие и шифрование данных.

Уровень приложения выполняет функции, относящиеся к общесетевым приложениям. Структура и функции я прикладной программы определяют структуру и функции прикладного уровня.

В соответствии с принципами OSI/ISO каждый сетевой уровень должен выполнять четко определенную функцию.

4. Понятие системы клиент - сервер.

Модель клиент - сервер предполагает двустороннее соединение по сети. Одно и то же приложение выполняется двумя сторонами по-разному и состоит из части-клиента и части-сервера. Модель существенно упрощает разработку сетевого приложения.

Практически все сетевые приложения Internet построены по принципу клиент - сервер.

5. Компоненты TCP/IP

Различные сервисы, включаемые TCP/IP, и их функции могут быть классифицированы по типу выполняемых задач.

1) Транспортные протоколы управляют передачей данных между двумя машинами:

- TCP (Transmission Control Protocol) - поддерживает передачу данных, основанную на виртуальном соединении посылающего и принимающего компьютеров.
- UDP (User Datagram Protocol) - протокол пользовательских дейтаграмм. Он поддерживает передачу данных без установления виртуального соединения. Нет гарантии доставки сообщения.

2) Протоколы маршрутизации обрабатывают адресацию данных и определяют наилучшие пути до адресата.

- IP (Internet Protocol) - Обеспечивает фактическую передачу данных.
- ICMP (Internet Control Message Protocol) - обрабатывает сообщения состояния для IP (ошибки, изменения в аппаратных средствах, которые влияют на маршрутизацию)
- RIP (Routing Information Protocol) - определяет наилучший маршрут доставки сообщения
- OSPF (Open Shortest Path First) - альтернативный протокол определения маршрутов

3) Поддержка сетевого адреса - способ идентификации машины с уникальными номером и именем.

- ARP (Address Resolution Protocol) - определяет уникальные цифровые номера компьютеров
- DNS (Domain Name System) - определяет числовые адреса по именам компьютеров и имена по числовым IP-адресам.
- RARP (Reverse Address Resolution Protocol) - протокол, выполняющий действие, обратное ARP.

4) Прикладные сервисы - это программы, которые пользователь применяет для получения доступа и различным услугам Internet.

- BOOTP (Boot Protocol) - загрузка операционной системы компьютера с сетевого сервера.
- FTP (File Transfen Protocol) - передача данных между компьютерами.
- TELNET - удаленный терминальный доступ к системе.

5) Шлюзовые протоколы - помогают передавать по сети сообщения о маршрутизации и информацию о состоянии сети, а так же обрабатывать данные для ЛВС.

- EGP (Exterior Gateway Protocol)-передача маршрутизационной информации для внешних сетей.
- GGP (Gateway - to-Gateway Protocol)- передача маршрутизационной информации между шлюзами.
- IGP (Interior Gateway Protocol) - передача маршрутизационной информации для внутренних сетей.

6) Другие протоколы:

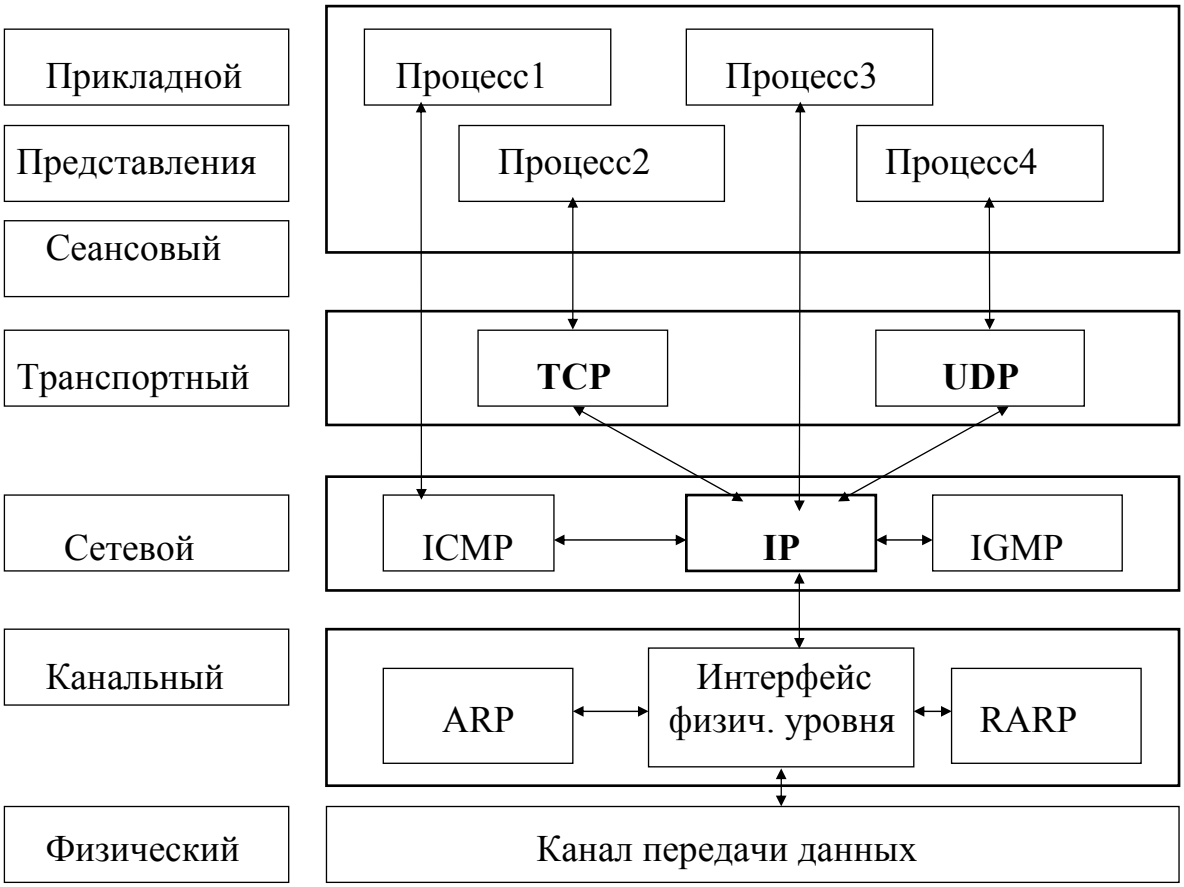
- NFS (Network File System) - использование каталогов и файлов другого компьютера как локальных ресурсов
- NIS (Network Information Service) - информация о пользователях, что упрощает вход в систему
- RPC (Remote Procedure Call) – буквально: «Удаленный вызов подпрограмм» протокол связи удаленных программ простым и эффективным способом
- SMTP (Simple Mail Transfer Protocol) - передача сообщения электронной почты.
- SNMP (Simple Network Management Protocol) - протокол администрирования сети. Посылает сообщение о состоянии сети и подключении к ней устройств.
- IGMP - протокол групповых сообщений

Согласно терминологии, используемой в литературе между следующими терминами принята полная однозначность:

TCP/IP = “набор протоколов Internet” = “набор протоколов TCP/IP” = “технология Internet”

6. Стэк протоколов.

Согласно модели OSI каждому уровню соответствует набор протоколов, определяющих его функциональность. Эта модель представляет сеть в виде вертикального стека протоколов. Это относится к любой комбинации сетевых уровней и соответствующих протоколов. Стек TCP/IP - один из стеков, соответствующих рекомендациям ISO.



Поток данных передается от одного уровня (протокола) к другому. Три шага передачи данных в сети Internet:

1. Путь от приложения до сети через стек протоколов
2. Определение адреса получателя данных
3. Физическое перемещение данных к месту назначения с использованием маршрутизации.

Стек протоколов TCP/IP состоит из пяти уровней. Независимо от выбранного маршрута данные должны пройти через модуль IP. В большинстве случаев данные проходят все уровни сетевой модели, но это не обязательно.

Канальный уровень (уровень соединения) состоит из интерфейса аппаратного обеспечения и двух модулей протоколов:

ARP (Address Resolution Protocol) - протокол преобразования адреса;

RARP (Reverse APR) - протокол обратного преобразования адреса

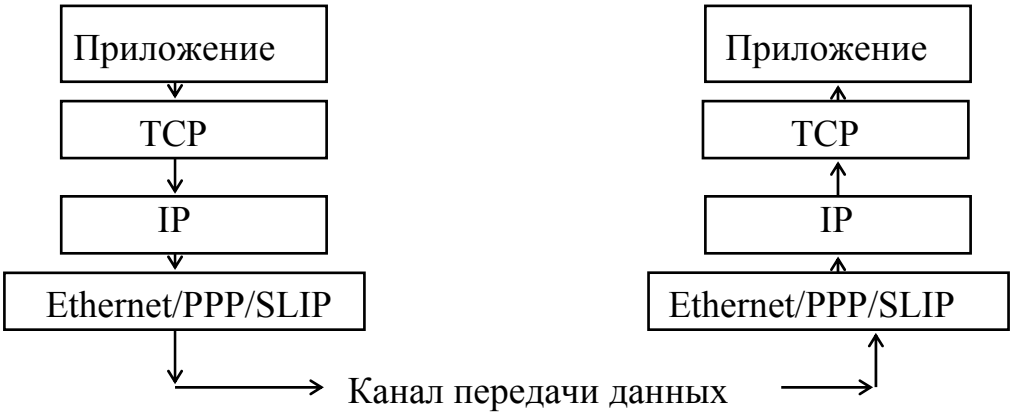
Канальный уровень делает так, что протоколам сетевого уровня все равно, на какой сети работать: Ethernet, Toker Ring и т.д.

7. Сетевой протокол Internet. Спецификация IP.

Сетевой уровень TCP/IP выполняет диспетчерские функции с использованием протокола IP. Уровень включает: протокол Internet (IP), протокол управляющих сообщений (ICMP) и протокол групповых сообщений (IGMP). ICMP и IGMP - являются дополнениями к IP, который выполняет основную работу. Практически все протоколы (TCP, UDP) пользуются услугами IP.

Работая в сети TCP/IP протокол IP инкапсулирует (размещает в своих пакетах) данные практически всех основных протоколов, за исключением ARP и RARP.

При передаче данные перемещаются по стеку протоколов от приложения к каналу передачи данных, а при приеме - в обратном направлении.



Инкапсуляция - это процесс преобразования данных прикладного процесса в форму, требуемых более низким уровнем стека сетевых протоколов. Каждый уровень инкапсулирует данные по-своему.

Прикладные данные могут иметь заголовок прикладного сообщения и поле прикладных данных. После прохождения сетевого уровня (IP) сегмент TCP превращается в IP-пакет (дэйтаграмму). Драйвер Ethernet инкапсулирует IP-пакет в кадр протокола Ethernet, PPP или SLIP - в зависимости от исапользуемого канала передачи данных и протокола низкого уровня.

Формат и поле заголовка IP

Версия Vers	Длина загол.	Тип сервиса TOS	Общая длина	
Идентификация			Флаги	Смещ. Фрагмента
TTL	Протокол		Контрольная сумма	
Адрес отправителя				
Адрес получателя				
Опции			Заполнитель	
ДАННЫЕ				

0 4 8 16 19 31

Длина IP-заголовка всегда кратна 32 битам.

Версия - значение версии IP-протокола. В настоящее время это поле **Vers=4**. Все программное обеспечение обязано считывать информации этого поля. Модуль, не примет пакет, который он не сможет обработать. При этом выдается сообщение о возникших проблемах.

Hlen - длина заголовка в 32-битовых словах (!). Hlen=20, т.е. 5h.

TOS - тип службы - приоритет IP-пакета. Это поле позволяет службе IP принять решение по поводу передачи пакета. Поле TOS имеет следующую структуру:

Приоритет	Задержка	Производит.	Надежность	Стоимость	Резерв
от 0 до 7	min задержка	Широкая полоса	надежность	Высокая затраты	min

Общая длина пакета (байт) - максимально 65525. Если приложение желает передать пакет большей длины, то происходит его фрагментация.

Идентификатор используется для идентификации пакетов и их принадлежности к конкретной дейтаграмме при фрагментации пакета.

Флаги помогают собирать фрагментированный пакет. Это поле действует вместе со смещением.

TTL - (Time-to-Live) - время жизни пакета в сети. Большинство систем устанавливает TTL=30. Маршрутизаторы вычитают время задержки пакета. Если TTL=0, то пакет уничтожается, о чем извещается источник пакета.

Протокол = 6 - данные из сегмента TCP
= 17 - UDP - дейтаграмма
= 1 - ICMP
= 2 - IGMP

Контрольная сумма определяется только для IP-заголовка.

Адрес источника и приемника информации определяет IP-адреса отправителя и приемника пакета.

Опции - используются для отладки и тестирования подготовленных компьютерах (Например указание маршрутизатору записывать пути движения пакета.)

8.1. Понятие IP-адреса.

Адрес в сети Internet называется IP-адресом. В литературе по Internet часто предполагается, что IP-адрес соответствует конкретному компьютеру. На самом деле в сети TCP/IP IP-адрес соответствует каждому интерфейсу, аналогично, как адрес Ethernet соответствует сетевой карте. Один компьютер может иметь несколько сетевых адресов.

Длина адреса Internet - 32 бита (4 байта). В языке Си IP-адрес можно представить типом long integer. Но удобнее записывать его в форме десятичных чисел через точку. Эта форма записи представляет собой четыре разделенных точками десятичных числа, следующих друг за другом:

194.226.20.177
255.255.255.255
0.0.0.0.

Это правило облегчает чтение и запись адреса. 32-битный IP-адрес скрывает за собой номера сети и сетевого компьютера (точнее интерфейса). Internet состоит из тысяч взаимосвязанных сетей. Чтобы отличить одну сеть от другой, информационный центр сети Интернет (Inter NIC, Network Information Centre) следит за тем, чтобы номер каждой, входящей в Internet сети, нигде больше не повторялся.

IP-адрес состоит из двух частей: адреса сети и номера хоста. Под хостом понимается любое устройство, которое имеет свой сетевой интерфейс (компьютер, принтер, терминал и т.п.). Существует 5 классов IP-адресов.

8.2. Классификация IP-адресов.

Классы адресов A,B,C,D, и E отличаются друг от друга количеством битов, отведенных на адрес сети и адрес хоста в сети, а соответственно, числом сетей и количеством компьютеров (хостов) в сети каждого класса.

Класс сети	Колич сетей	Количество хостов
A	126	16777214
B	16382	65534
C	2097150	254
D	-	228
E	-	227

0 1 2 3 4 5 6 7 8... 16 23 24 31

0	номер сети	н о м е р х о с т а
1 0	номер сети	н о м е р х о с т а
1 1 0	номер сети	номер хоста
1 1 1 0	г р у п п о в о й а д р е с	
1 1 1 1 0	з а р е з е р в и р о в а н о	

Адреса класса A предназначены для использования в больших сетях общего пользователя. Число таких сетей невелико. Реально адреса класса A используются небольшим количеством сетей, в состав которых входит более 65536 компьютеров.

Адреса класса B предназначены для использования в сетях среднего размера (сети больших кампаний, НИИ, университетов (>255 узлов).

Адреса класса C предназначены для использования в сетях с небольшим числом компьютеров (до 254).

Адреса класса D используется для обращения к группам компьютеров (более, чем 1).

Адреса класса E зарезервированы и вряд ли будут использоваться в ближайшее время.

В первоначальном проекте Internet предлагалось представление адреса в виде:

Адрес сети	Адрес хоста

0 7 8 31

Тогда интернет мог бы вместить > 4 млрд. компьютеров, принадлежащих всего 256 сетям. Рассмотренная схема классификации адресов уменьшает количество компьютеров в Internet примерно на 10 %, но увеличивает при этом количество сетей до более чем на 2 млн. То есть за счет уменьшения индивидуальных адресов увеличивается количество сетевых адресов. Это существенно облегчает сетевое администрирование, т.к. управление 4 млрд. или 16 млн. компьютеров - практически неразрешимая задача. Только 126 сетей имеют в своем составе 16 млн. компьютеров (класс А).

8.3. Процедура выделения IP-адресов

Количество возможных сетевых адресов ограничено. Распределением адресов занимается информационный центр Internet (InterNIC). Он следит, чтобы адрес не выдавался повторно. Адреса сетевых интерфейсов присваиваются администраторами внутри самостоятельных сетей. Повторите присвоение адресов недопустима.

Среди всех IP-адресов имеется несколько зарегистрированных под специальные нужды.

IP-адрес	Назначение.
0.0.0.0	Данный узел сети.
Номер_сети. Все нули	Данная IP-сеть.
Все_нули. Номер_узла	Узел в данной (локальной сети).
FF.FF.FF.FF	Все узлы в данной локальной IP-сети.
127.0.0.1	Все узлы указанной IP-сети (“ПЕТЛЯ”).

“ПЕТЛЯ” предназначена для тестирования программ и взаимодействия процессов в рамках одного компьютера. В большинстве случаев в файлах настройки этот адрес обязательно должен быть указан, иначе при запуске система может зависнуть.

IP-адреса проставляются InterNIC бесплатно. В целях увеличения эффективности одна сеть может разделяться на несколько подсетей путем деления адресного пространства. Например, условно можно выделить в сети класса В из 16 битов для нумерации компьютеров 8 битов для нумерации подсетей и остальные 8 битов - для нумерации компьютеров в подсетях. То есть исходная сеть разделяется сетевым администратором на несколько подсетей.

Подсеть - это подмножество сети, не пересекающееся с другими подсетями. Эти адреса - произвольное деление администратора и не имеют отношения к действующей классификации сетей Internet. Теоретическая сеть класса В можно разделить на 256 подсетей по 256 компьютеров в каждой.

Внешний источник данных ничего не знает о делении сети на подсети и шлет пакеты по действительному адресу Internet. То есть концепция деления сети на подсети работает внутри конкретной сети.

Для разбиения сети на подсети администратор может использовать маску, с помощью которой “маскируется” часть IP-адреса, используемая для получения номера подсети.

Маска подсети - это 4 байта, которые накладываются на IP-адрес для получения номера подсети. С помощью маски формируется адрес подсети. При указании маски нельзя использовать номера “все нули” и “все единицы”. Например, маска 255.255.255.0 позволяет разбить сеть класса B на 254 подсети по 254 узла в каждом.

Пример:

Имеется сеть класса B с адресом 144.206.0.0.

10010000 11001110

11111111 11111111 111 00000 00000000

номер сети — номер подсети — адрес узла

Эта маска позволяет разбить сеть на 6 подсетей, т.к. для нумерации нельзя использовать подсети 000 и 111.

IP-адрес	144.206.160.32	1001 0000	1100 1110	1010 0000	0010 0000
Маска	255.255.224.0	1111 1111	1111 1111	1110 0000	0000 0000
IP-адрес подсети номер 5		1001 0000	1100 1110	1010 0000	0000 0000

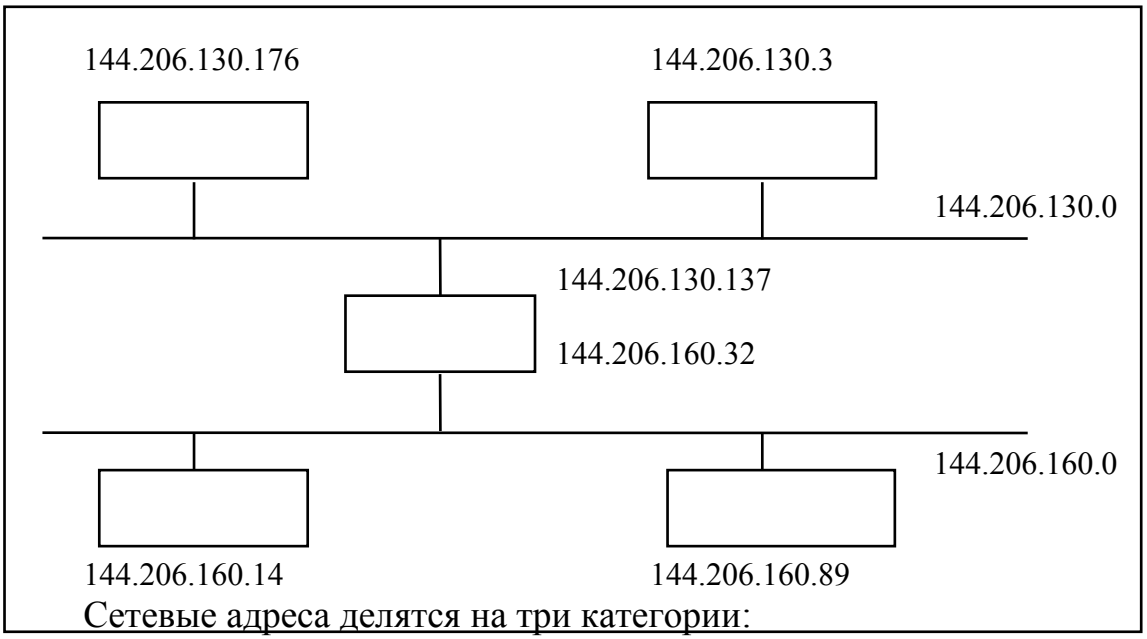
Для нумерации машин в подсети можно использовать оставшиеся 13 бит.

Вообще говоря подсети придуманы для того, чтобы обойти ограничения физических сетей на число узлов или длину кабеля в сегменте сети.

Ethernet (тонкий)	Сеть класса C
185 м и 32 узла	254 узла

Например, сеть 144.206.0.0. может состоять из двух подсетей 144.206.130.0 и 144.206.160.0

Сеть 144.206.130.176



Сетевые адреса делятся на три категории:

- обычные:
- групповые;
- широковещательные.

Адреса классов A, B, C - обычные, т.к. служат для обращения к одиночному сетевому компьютеру. Указывая широковещательный адрес, мы сразу задаем обращение ко всем компьютерам сети. Групповой адрес указывает определенную группу компьютеров. Компьютеры такой группы могут принадлежать разным сетям. Число компьютеров в группе не ограничено. Хосты и маршрутизаторы группового вещания используют протокол IGMP. Центр InterNIC определил набор официально известных групповых адресов, которые представляют группу постоянно существующих компьютеров. Такими адресами, например, являются:

224.0.1.1. - протокол сетевого времени.

224.0.1.2. - протокол игры “dogfight” фирмы “Силикон графикс”.

8.4. IP - маршрутизация

TCP/IP представляет собой стек протоколов.

IP - подсистема доставки данных в Internet. Посредством доставки дейтаграмм данные приложений и информация различных протоколов перемещаются по сети Internet. Ключевым моментом в доставке IP-дейтаграмм является IP-маршрутизация. Таблица IP-маршрутизации - это то место, где хранятся адреса некоторых получателей Internet. Содержимым и самим таблицам управляют протоколы маршрутизации.

Прикладной программист не имеет возможность управлять маршрутизацией данных своих приложений. Прикладные программы широко используют IP-адреса в своей работе.

Важный момент: у каждого сетевого интерфейса, принадлежащего одной физической сети, должен быть уникальный собственный номер.

Таблицы маршрутизации основывают работу на том факте, что все хосты на одной и той же сети имеют один и тот же сетевой номер. В таблицах маршрутизации хранятся только номера сетей.

Указание путей при пересылке пакетов в локальной сети называется прямой маршрутизацией, а при пересылке пакетов через шлюз - косвенной.

Струтктура записи в таблице маршрутизации

Network	Gateway	Flags

Íîìåð ñåòè.	Íîìåð ñåòè.	Ïðèçíàê ïðÿìîãî ñîåäèíåíèÿ с адресуемой сетью
Список идент.	Инф. о маршрутизаторе, через который передаются пакеты от сети.	

Когда компьютер получает IP-пакет, модуль IP проверяет идентификатор сети назначения, размещенный в заголовке пакета. Далее IP-модуль запрашивает запись для сети с этим идентификатором из таблицы маршрутизации. Если запись найдена, исследуется содержимое поля флагов. Если в поле флагов указано прямое соединение, значит пакет можно доставить, используя формат кадра данных низкого уровня (Ethernet). Если устройства с данным адресом не соединяется напрямую с сетью, в которой адресуется пакет, то необходимо осуществить процесс промежуточной доставки.

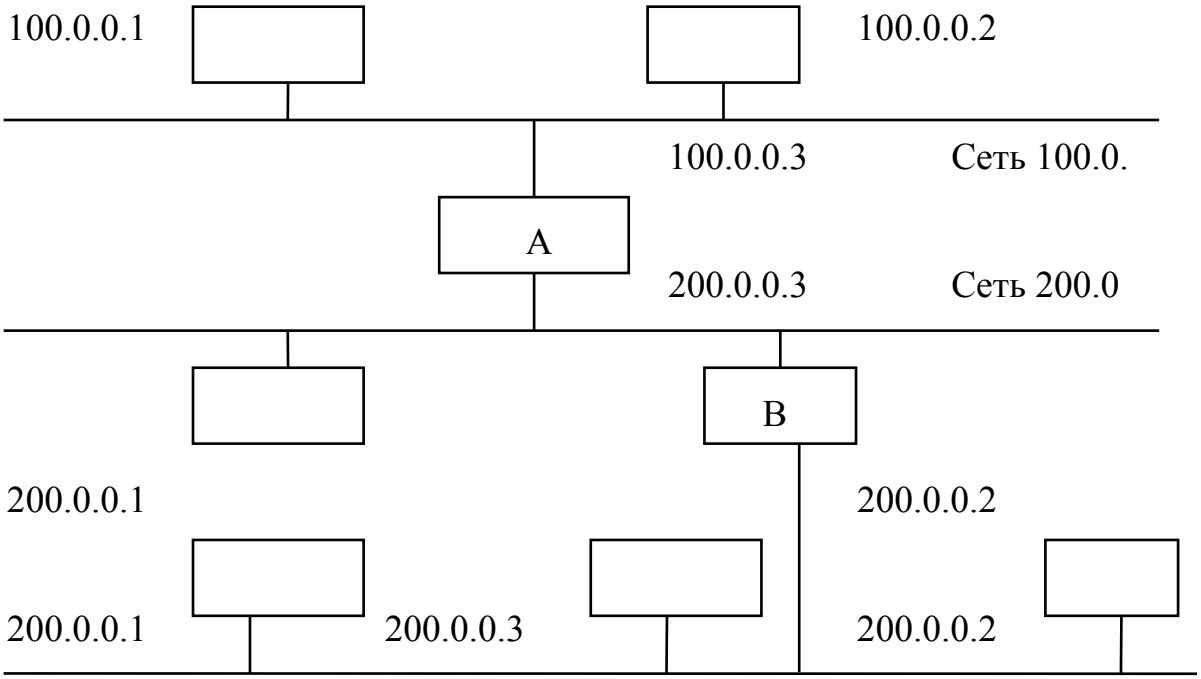


Таблица маршрутизации определяемой сети содержит данные только о маршрутизаторах,, связанных с ней напрямую. Это значит, что сеть может передать пакет, пользуясь непосредственной доставкой, любому из них.

Таблица маршрутизатора А.

Сеть	Шлюз	Флаг
100.0.0.	-	прямая
200.0.0.	-	прямая
200.0.0.	200.0.0.2	промежуточная
127.0.0.1.	-	прямая

Маршрутизатор никогда не будет пытаться прислать пакет через тот же интерфейс, через который пакет пришел.

Обычно таблица маршрутов на каждом хосте Internet формируется в момент включения. Информацией для ее формирования служат команды из файла конфигурации.

Например, route add default 144.206.160.32

Это значит, что все пакеты с компьютера будут передаваться на указанный шлюз, если они не смогут быть переданы в рамках данной сети.

route add default 144.206.160.40 127.0.0.1.

Данная запись зацикливает пакеты, не выпуская их из подсети.

Важным моментом при управлении маршрутизации является протокол RIP. В соответствии с протоколом RIP любая машина может быть маршрутизатором. Все маршрутизаторы делятся на активные и неактивные. Активные маршрутизаторы сообщают о маршрутах, которые они поддерживают в сети. Пассивные маршрутизаторы читают эти широковещательные сообщения и корректируют свои таблицы. Обычно таблица маршрутизации обновляется каждые 30 с. В таблицу не вносятся маршруты более 15 (20, 30) скачков. Если с некоторого шлюза в течении времени обновления (примерно 180 с) не поступает информация, то все маршруты, связанные с эти шлюзом, из таблицы удаляются. Если через шлюз за определенный момент времени не прошел ни один пакет, то этот шлюз может быть приведен в разряд пассивных маршрутизаторов. Это процедура routed.

Для запроса подсети в InterNIC необходимо направить заявку, в которой, в частности, указываются следующие сведения:

Тип запроса адресов:
request-type: 1 class C number
Сколько ПК подключено сразу:
machine-0: 40
Сколько через год:
machine-1: 180
Через 2 года:
***
Сколько подсетей:
subnet-0: 4
...
Когда и через кого планируется физическое подключение
Страна расположения сети

RU
Addening plan:

#	subnet #	submask	max New	Descriptor
1	15	255.255.255.224	30	Организация

9. Доменные имена

Всякий раз, когда посылается сообщение какому-либо хост-компьютеру Internet, IP-адрес используется для указания адреса отправителя и адреса получателя. Чтобы не запоминать числовые адреса, существует специальный сервис TCP/IP, называемый Domail Name System (Доменная система имен).

Домен - это уникальный идентификатор, являющийся символьным адресом абонента Internet. Для получения доменного имени необходимо обратиться в InterNIC. После регистрации это имя заносится в базу данных имен. все имена в ней уникальны, что обеспечивает отсутствие коллизий. Доменное имя состоит из нескольких доменов, разделенных точкой:

домен1.домен2. домен3

Система доменных имен строится по иерархическому принципу. Правая часть имени называется доменом верхнего уровня. InterNIC утвердил 6 имен верхнего уровня:

.arpa - идентификатор сети arpa;
.com - коммерческие компании;
.edu - образовательные учреждения;
.gov - правительственные учреждения;
.mil - военные организации;
.org - прочие организации.

Кроме того каждая строка имеет свой двухсимвольный идентификатор
au - Австралия
...
ca - Канада
...
su - СССР (для совместимости с ранее выделенными адресами)
....
ru - Россия
...
uk - Англия
...
us - США
...
zw - Зимбабве

На основе доменных имен строятся адреса электронной почты как в различных сетях, так и в сети Internet, а также обеспечивается доступ ко многим ресурсам Internet.

Вслед за доменами верхнего уровня следует домены, определяющие либо регионы, либо организации:

msk -Москва

msu - Московский государственный университет

tula

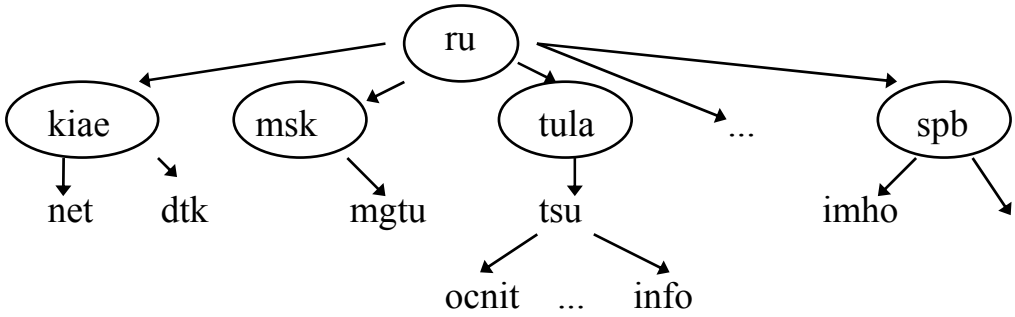
kiae - Курчатовский институт атомной энергии

Далее следуют домены, определяющие организации, подразделения, отделы пользователей и т.д. Обычный синтаксис адреса пользователя e-mail:

username@domain_name

Например: anatol@ocnit.tsu.tula.ru

Систему доменной адресации можно представить в виде графа.

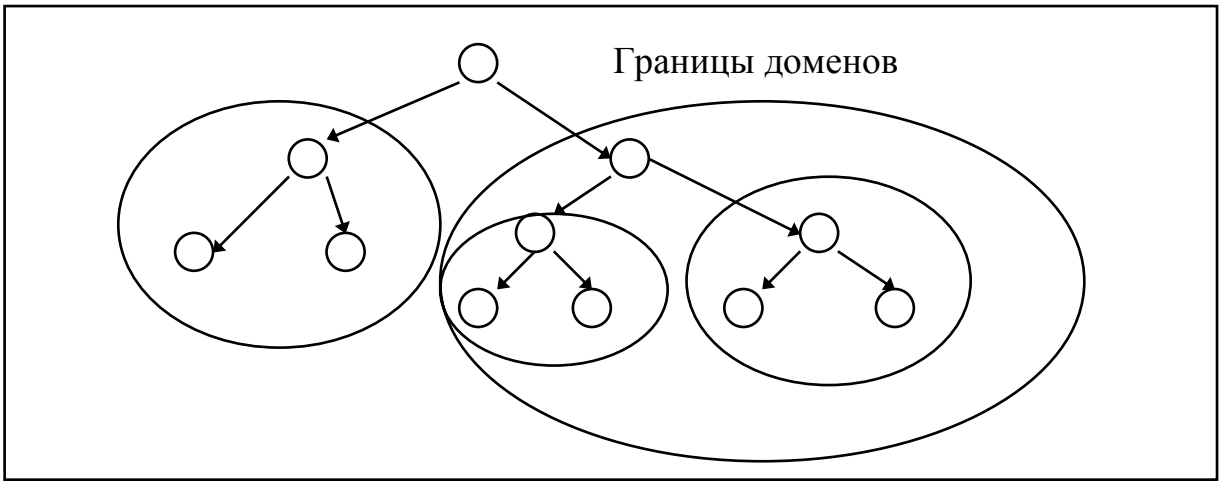


При регистрации доменного имени это имя заносится в сервер доменных имен (DNS), где в таблицу заносится и соответствующий IP-адрес. DNS является сервером сети TCP/IP и поддерживается специальным программным обеспечением. Одной из таких программ является BIND- Berklley Internet Name Domain/

BIND обеспечивает поиск доменных имен и IP-адресов для любого узла Сети Internet.

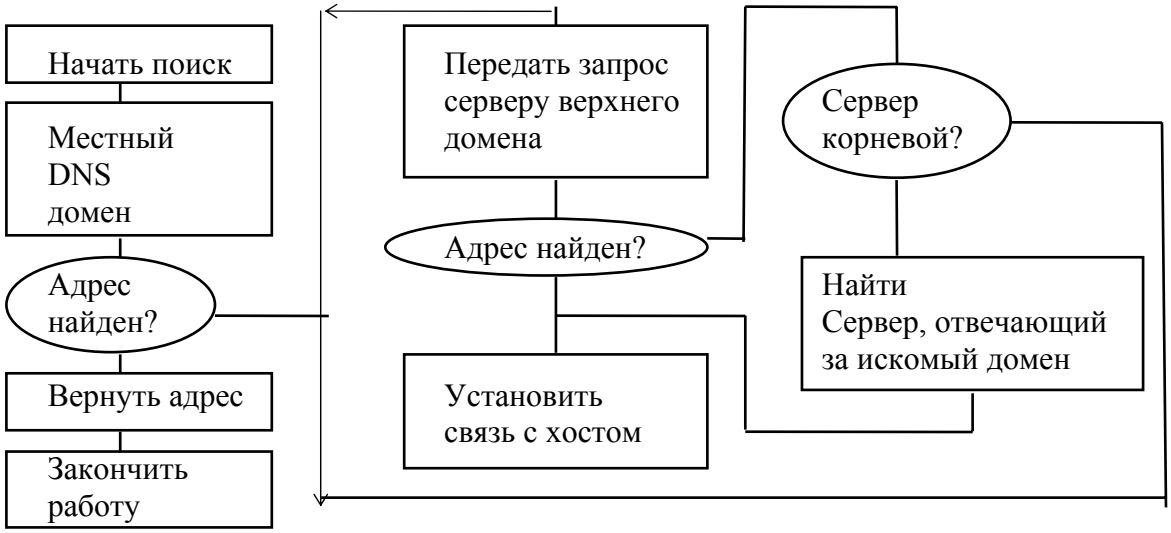
BIND реализована по схеме “клиент-сервер”. Собственно BIND-это сервер, который обеспечивает поиск доменного адреса машины, который адресован пакет, и определяет IP-адрес.

Доменную иерархию можно представить в виде схемы, в которой DNS имеет свою зону ответственности.



Чтобы получить адреса или имена машин, не входящих в зону ответственности DNS, сервер обращается к другим серверам имен. При этом

запрос может пройти путь до корня домена и спуститься по ветви вниз. Запрос формирует специальные программы resolver. Алгоритм поиска адреса:



В случае составного имени выполняется проверка на местном сервере, а затем выполняет переход к вышестоящему домену.

Имеется четыре вида серверов:

1. primary master server - сервер, поддерживающий свою базу данных имен и обслуживающий местный домен.

2. secondary master server - сервер, обслуживающий свой домен, но не хранящий базу данных, а обращающийся к другому серверу.

3. cashing server -не имеет своего домена, но получает данные либо от master -servera, либо из буфера.

4. remote server - сервер, установленный на удаленной машине. Программы обращаются к нему по сети.

Серверы типа 1 и 2 устанавливаются на компьютерах-шлюзах для локальный вычислительных сетей. Такой сервер должен быть один на зону.

Cashing server - ускоряет процесс поиска.

Remote server устанавливается на персональные компьютеры, не выполняющие роль шлюзов.

Любой сервер имен настраивается своими программами.

10.1 Понятие порта в сетевом приложении.

Приложение, работающее в Internet, как правило, общается с одними из протоколов транспортного уровня:

- протоколом управления транспортировки TCP или протоколом пользовательских дейтаграмм UDP. Приложение строит свою работу на взаимодействии с одним из этих протоколов. Конечная цель сетевого взаимодействия заключается в передаче информации между приложением-клиентом и приложением-сервером.

IP-модуль - служба доставки. Он обеспечивает транспортировку пакетов между сетевыми интерфейсами. Транспортный уровень и его протоколы передают сообщение между приложениями.

Ориентированный на соединение протокол TCP использует надежный поточно-байтовый способ доставки. Сетевое соединение устанавливается в виде виртуальной цепи. Протокол UDP ненадежен и передает данные при помощи дейтаграммы без установления соединения.

В терминологии TCP/IP на уровне прикладных программ используется понятие порт, которым обозначается конкретное приложение. Порт в TCP/IP - это то же, что и IP-адрес, но адрес обозначает конкретный сетевой интерфейс, а порт - приложение.

IP-дейтаграммы содержат адреса получателей и отправителей пакетов, а TCP хранит номера портов получателя и отправителя.

В персональном компьютере принято обозначать порты именами: LPT1, 2, COM1, 2, 3 и т.д. В Internet номера портов нумеруются. Прикладные программы ПК знают, что через LPT1: осуществляется вывод на печать. Точно так же в Internet существуют порты, ассоциируемые приложениями с вполне конкретными функциями.

номер порта	**Сетевая служба**
7	Служба ЭХО (Echo Protocol)
13	Время суток
21	FTP
23	Telnet Protocol (уд*** терминал)
25	Протокол передачи e-mail
37	Точное время
43	Кто-есть-кто
69	Протокол простой передачи файлов
79	Информация о пользователях

Internet включает протоколы для некоторых часто используемых приложений (FTP, Telnet, SMTP и т.п.). Этим приложениям присвоены “официальные” номера портов, которые могут использоваться всеми компьютерами Сети.

Порты служат для установления связи прикладной программы с определенным сервером. Номер порта определяет тип сервиса, запрашиваемого одним компьютером у другого.

Каждый компьютер в сети поддерживает небольшую таблицу, которая содержит описание использования всех портов. Вносить изменения в эту таблицу не следует.

Большинство приложений Internet ориентированы на использование и протокола UDP и протокола TCP.

Использование UDP. Протокол IP доставляет пакеты к сетевым компьютерам. UDP сортирует пакеты по портам. Клиенты должны

периодически сами проверять свои порты, т.к. UDP не извещает клиентов о поступлении пакетов.

Использование TCP. Этот протокол ориентирован на соединение. Прикладные протоколы, работающие на сетевом компьютере, пользуются услугами IP. Каждому прикладному процессу присвоен свой номер порта. IP доставляет служебный пакет в конкретный номер порта для установления соединения с вызвавшим протоколом. Далее происходит обмен служебными пакетами для уточнения вида соединения (ПП может переключить соединение на другой порт, чтобы не занимать официальный порт, может открыть еще одно соединение по этому же порту).

Любое приложение - клиент или сервер - имеет уникальный номер порта, с помощью которого UDP и TCP могут распознать нужное приложение.

10.2. Протокол UDP и его спецификация.

Структура UDP-заголовка имеет вид:

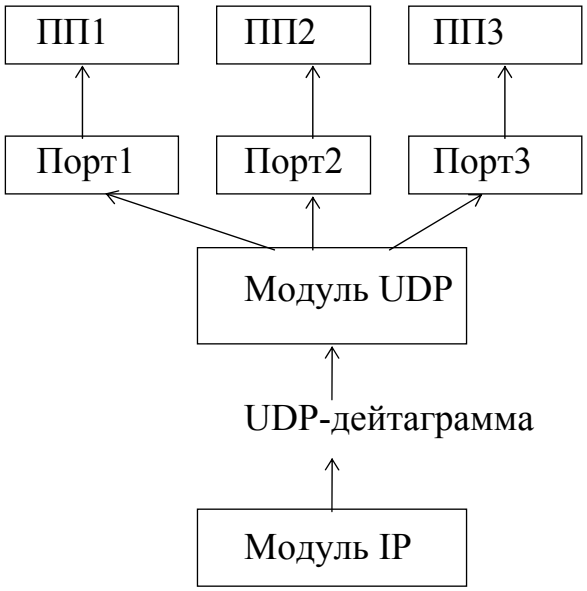
Порт получателя UDP	Порт источника UDP
Длина сообщения UDP	Контрольная сумма
Данные	

Порт источника - это номер порта, выполняющего запрашиваемый сетевой сервис.

Порт получателя - это номер порта приложения получателя данных.

Длина сообщения - это заголовок +данные

Контрольная сумма это - результат суммирования всей UDP-дейтаграммы (т.е. заголовок+занные). Это поле заполнять не обязательно!



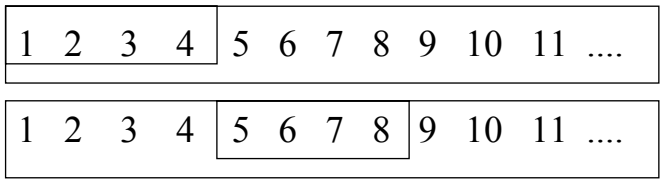
Модуль UDP отслеживает появление вновь прибывших дейтаграмм, сортирует их и распределяет в соответствии с портами назначения.

10.3. Протокол TCP.

TCP - наиболее распространенный протокол. Как и UDP, он служит для передачи данных между сетевым и транспортным уровнями. Этот протокол следит за правильностью доставки данных. Кроме того TCP оптимизирует пропускную способность сети, увеличивает ее производительность, например, При переполнении данных на приеме, он снижает скорость передачи данных.

TCP пользуется услугами IP и считается надежным протоколом.

Обеспечение надежности. Для выполнения этой функции используются подтверждения. Как только абонент получает блок данных, он передает подтверждение источнику. Механизм таймера и счетчика повторов передачи обеспечивают надежную доставку пакетов. Ожидание подтверждения после посылки каждого пакета снижает производительность сети. В TCP используется принцип “скользящего окна”. Этот принцип позволяет послать несколько сообщений, а потом ждать подтверждений.



TCP передает все данные, попавшие в окно и ждет подтверждений для этих данных.

Приняв подтверждение о приеме всех данных, TCP смещает окно дальше по потоку и передает следующие данные. TCP регулирует полосу пропускания сети, договариваясь с другой стороной о параметрах потока данных. Процесс регулирования происходит на протяжении соединения.

TCP может изменять ширину окна. Его размер задается в байтах. По умолчанию ширина окна составляет 4096 байт (8192/16384 байтов).

Блок данных TCP называется сегментом:

Сегмент_TCP:= ЗАГОЛОВОК+[ОПЦИИ]+[Заполнение]+[Данные]

Сегмент является последовательным потоком данных минимум из 20 байт.

Сегмент TCP:

Порт-передатчик	Порт-премник
Номер последовательности	
Номер подтверждения	
Lзаг. Резерв U A P R S F R C S S Y I G K H T N N	Размер окна
Контрольная сумма TCP	Указатель на данные для неотложенной обработки
ОПЦИИ	Заполнитель
ДАННЫЕ	

Порты источника и получателя - идентифицируют прикладные процессы или протоколы. Номера портов источника и получателя в совокупности с IP-адресами сетевых компьютеров (в IP-заголовке) однозначно определяют любое IP-соединение. Каждая из сторон TCP-соединения называется сокетом.

Номер последовательности - это первый байт данных из области данных. Этот номер указывает смещение начала данных относительно начала потока данных.

Номер подтверждения - обозначает байт данных, который принимающая сторона рассчитывает получить следующим в потоке. (Если был принят байт номер 500, то следующий должен быть 501)...

Длина заголовка передается в словах. Обычно имеет длину 20 байтов.

Флаги - шесть однобитовых полей

URG - сообщает принимающему модулю TCP о том, что указатель на данные, требующие неотложной обработки, установлен.

ACK - “номер подтверждения” содержит верные данные.

PSH - требование немедленно выслать принятый сегмент данных приложения получателю. Как правило, TCP буферизует данные, а затем передает приложению. Флаг PSH позволяет исключить задержки, например, при работе с Telnet для выдачи эха на экран.

RST - запрос у принимающего модуля TCP на сброс соединения. Большинство приложений завершает работу при получении этого флага (сброс в сети, в оборудовании и т.д.)

SYN - синхронизация номера последовательности. Сообщает, что источник готовится передать новый поток данных.

FIN - источник закончил передавать данные. Чтобы полностью закончить соединение, принимающий компьютер должен послать также пакет с установленным FIN.

Размер окна - сообщает количество байт, которое будет передавать источник.

Контрольная сумма TCP - это сумма, вычисленная в области данных. Протокол требует, чтобы источник включил эту сумму в поле, а приемник - вычислил ее вновь и сравнил результат.

Указатель неотложенных данных - указатель и флаг UGR извещают принимающий модуль, что в сегменте находятся данные, требующие немедленной обработки.

В TCP нет обработки этих данных. В настоящее время от обработки таких данных следует воздержаться.

Опции - TCP могут договариваться о максимальной длине сегмента (MSS) и устанавливать соответствующие опции. TCP оптимизирует пропускную способность сети. Эта опция указывает на максимальный размер сегмента, который можно передать. По умолчанию MSS=536 байт.

10.4. Установление и окончание TCP-соединения

Клиент посылает серверу запрос на установку соединения. Сервер, приемник сообщения, получив запрос на соединение, посылает обратно сообщение, указывая свой собственный начальный номер. Все TCP соединения являются дуплексными. Данные в обоих направлениях следуют одновременно. Поток в одном направлении не зависит от противоположного потока

Клиент		Сервер
Сегмент1, байты 1...500 Номер последовательности 1251	→	Готовит ответ с подтверждением и ожиданием последовательности
	←	1751
Сегмент2, байты 501 ...1000 номер последовательности 1751	→	Номер последовательности
	←	2251
Сегмент3, байты 1000 ... 1501	→	Номер последовательности
	←	2751

Сервер еще не знает об окончании передачи данных.

Окончание соединения TCP.

Соединения TCP заканчивается обменом пакетами, состоящими из двух стадий.

Инициатор обмена посылает пакет с указателем FIN. Принимающая строка также посылает пакет с флагом FIN. Соединение считается закрытым, если обе стороны послали друг другу сообщение с флагом FIN.

Может случиться, что поток данных в одну сторону будет закончен, но в другую сторону может быть сохранен (передаются пакеты без FIN). Такое завершение называется закрытым на половину. При этом пассивная сторона может принимать данные, не комментируя их содержание.

III. ОБЪЕКТ ИССЛЕДОВАНИЯ, ПРИБОРЫ, ОБОРУДОВАНИЕ

В работе используются:

- Персональный компьютер, на котором установлена среда программирования и отладки AVR Studio 4.0.
- Набор для программирования и отладки семейства микроконтроллеров AVR – STK500.
- СБИС микроконтроллера серии AVR – ATmega-16.
- Соединительные кабели.

IV. ЗАДАНИЕ НА РАБОТУ

1. Написать и отладить программу на языке ассемблер, рассмотренную в примере 1.

1.1. Модифицировать программу в примере 1, таким образом, чтобы инвертировать все биты выводимых данных.

1.2. Модифицировать программу в примере 1, таким образом, чтобы инвертировать биты 0-3 выводимых данных.

1.3. Модифицировать программу в примере 1, таким образом, чтобы инвертировать биты 0, 2, 4, 6 выводимых данных.

1.4. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные содержали дополнительный код входных данных.

1.5. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные содержали в битах 0-4 сумму бит 0-3 и 4-7 входных данных.

1.6. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные содержали в битах 0-7 произведение бит 0-3 и 4-7 входных данных.

1.7. Модифицировать программу в примере 1, таким образом, чтобы выводимые данные сменяли друг-друга с периодом 0.25 или 0.5 или 1 с.

1.8. Прокомментировать построчно текст программы.

2. Написать и отладить программу на языке ассемблер, рассмотренную в примере 2.

2.1. Разобраться с назначением структурных единиц кода в примере 2.

2.2. Прокомментировать построчно текст программы.

2.3. Модифицировать программу в примере 2, таким образом, чтобы инвертировать все биты выводимых данных.

2.4. Модифицировать программу в примере 2, таким образом, чтобы сменить направление вращения бит выводимых данных.

2.5. Модифицировать программу в примере 2, таким образом, чтобы удвоить период вращения бит выводимых данных.

2.6. Модифицировать программу в примере 2, таким образом, чтобы удвоить период мерцания индикатора LED7.

3. Написать программу на языке ассемблер, выполняющую над байтовой переменной, введенной из порта B следующие логические действия:

Установка 0 и 2 бита в "1";
Инвертирование 1 и 6 бита;
Сбросить 2 и 3 биты в "0";
Инвертировать результат;
Передать результат в порт D.

4. Написать на языке ассемблер процедуру, выполняющую поиск наибольшего элемента в массиве целых, адрес которого передается в нее в регистровой паре Z. Размерность массива - 100. Элемент массива – байт. Результат возвращается в регистре R0

4.1. Отладить программу и выполнить ее для конкретных значений массива.

4.2. Модифицировать программу в по п.4. для элемента массива размером два байта. Результат возвращать в регистровой паре R1:R0.

V. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

6. Разработка алгоритма и составление программы, на языке AVR Ассемблер;
7. Ввод и отладка программы;
8. Проверка работы программы;
9. Модификация программы согласно заданию;
10. Оформление отчета по лабораторной работе.

VI. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет должен содержать:

- тексты программ на языке Ассемблер для МП AVR;
- описание используемых операторов языка Ассемблер;
- описание способов изменения такта квантования в системе;
- структуру целевой системы, для которой составлялись программы;
- построчный комментарий всех приведенных программ.

VII. КОНТРОЛЬНЫЕ ВОПРОСЫ

5. Назовите основные группы команд языка AVR Ассемблер.
6. Работа с битами средствами языка AVR Ассемблер.
7. Что такое программные и аппаратные прерывания?

8. Какими способами можно изменять такт квантования сигналов в системе?

VIII. БИБЛИОГРАФИЧЕСКИЙ СПИСОК

5. Горячев О.В. Микропроцессоры и микропроцессорные вычислительные системы. – Тула, ТулГУ, 1998.
6. Система команд МП AVR — Atmel: Doc.Rev. 0856D-AVR-08/02.- 148с.
7. Микроконтроллер ATmega-16 — Atmel: Doc. Rev. 2466H-AVR-12/03. - 349с.