Конспект лекций по курсу

МИКРОПРОЦЕССОРЫ (МП) И МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ (МПС)

Конспект лекций по курсу «Микропроцессоры (МП) и микропроцессорные системы (МПС)» содержит краткие сведения о принципах организации, проектирования, а также использования аппаратных и программных средств микропроцессорных систем.

Для углубленного изучения указанных разделов данного курса (в объеме часов для самостоятельной работы, предусмотренных программой) и подготовки к экзамену, рекомендуется использовать методические указания к лабораторным работам и литературу, указанную ниже.

Литература

1. Вычислительные системы, сети и телекоммуникации: Учебник / Пятибратов А.П, Гудыно Л.П., Кириенко А.А. / Под ред. А.П. Пятибратова. –М.: Финансы и статистика, 2002.
2. Семененко В.А., Ступил Ю.В Справочник по электронной вычислительной технике. –М. Машиностроение, 1993.
3. Семененко В. А, Скуратович Э.К. Арифметико-логические основы компьютерной схемотехники. –М.: Изд-во МГИУ, 2004
4. Семененко В.А., Скуратович Э.К., Федоров Н.В. Функциональная и структурная организация вычислительных машин и систем. –М.: Изд-во МГИУ, 1999.
5. Микропроцессоры и микропроцессорные комплекты интегральных микросхем: Справочник / Под ред.В.А.Шахнова. – М.: Радио и связь, 1988.
6. Токхайм Р. Микропроцессоры: Курс и упражения / Пер.с англ. Под ред.

В.Н.Грасевича. – М.: Энергоатомиздат, 1987.

7. Григорьев В.Л. Программное обеспечение микропроцессорных систем. – М.: Энергоатомиздат, 1983. – 208 с.
8. Лю Ю-Чжен, Гибсон Г. Микропроцессоры семейства 8086/8088. Архитектура, программирование и проектирование микрокомпьютерных систем. / Пер.с англ. – М.: Радио и связь, 1987. – 512 с.
9. Бирюков С.А.Цифровые устройства на интегральных микросхемах - М.: Радио и связь, 1991
10. Щелкунов Н.Н., Дианов А.П. Микропроцессорные средства и системы- М.: Радио и связь, 1989
11. Микропроцессорные системы автоматического регулирования / В.В.Солодовников, В.Г.Коньков, В.А.Суханов, О.В.Шевяков; Под ред. В.В.Солодовникова- М.: Высшая школа, 1991

НАЗВАНИЯ РАЗДЕЛОВ КУРСА

1. Основные понятия микропроцессорной техники

Понятие о микропроцессорах (МП) и микропроцессорных системах (МПС). Техническая, программная и информационная совместимость микропроцессорных систем. Элементная база микропроцессорных систем.

2. Организация и структура типовой микропроцессорной системы

Основные компоненты микропроцессорной системы: центральный процессор, основная память, подсистема ввода/вывода. Их назначение. Системная магистраль (шина). Взаимодействие компонентов МПС.

3. Структура и функционирование однокристальных микропроцессоров

Состав и назначение узлов процессорного модуля. Компоненты микропроцессора, его регистровая и программная модель (на примере 8-разрядного МП КР580ВМ80). Байт состояния. Построение процессорного модуля на основе больших интегральных схем (БИС) микропроцессорного комплекта КР580.

4. Основы программирования на языке Ассемблер

Машинные и ассемблерные коды. Конструкция команды. Машинные циклы и такты.

5. Программно-аппаратная организация обмена данными в микропроцессорной системе

Использование адресных сигналов в МПС. Режимы и формы адресации данных на примере микропроцессора КР580ВМ80.

6. Архитектура, назначение и взаимодействие микропроцессора и периферийных БИС

6.1. Программируемый интервальный таймер КР580ВИ53. Его режимы работы и программирование.

6.2. Ввод/вывод данных по прерываниям. Программируемый контроллер прерываний КР580ВН59. Основные особенности ввода/вывода по прерываниям. Использование стека при обслуживании прерываний (переходах

к подпрограмме обслуживания и возвратах в основную программу). Варианты передачи управления для обслуживания прерываний. Общие принципы программирования при вводе/выводе по прерываниям. Программно-аппаратные особенности полинга при вводе/выводе по прерываниям. Маскирование прерываний и его реализация.

6.3. Прямой доступ к памяти. Основные особенности прямого доступа к памяти (ПДП). Программирование контроллера ПДП на примере контроллера КР580ВТ57. Его функционирование. Программно-аппаратная организация прерываний для обеспечения ПДП.

7. Основы построения и принцип работы современных БИС оперативных запоминающих устройств.

Быстродействие и производительность памяти. Достоверность хранения данных. Методы повышения скорости передачи данных (пакетный режим передачи данных, чередование памяти, кэширование памяти).

СТРУКТУРА МПС

Основой построения современных МПС является совокупность модулей, связанных общей системной шиной. Состав и назначение основных устройств МПС рассмотрим по структурной схеме, приведенной на рис.1.

Микропроцессор (МП) – предназначен для управления работой всех блоков МПС и выполнения, соответствующих программе машинных арифметических и логических операций. В состав микропроцессора входят:

1. *Устройство управления (УУ)* – получает от *генератора тактовых импульсов* опорную последовательность импульсов; формирует в нужные моменты времени и подает в соответствующие блоки МПС необходимые сигналы управления, обусловленные спецификой выполняемой операции и результатами предыдущих операций; формирует адреса ячеек памяти операндов, участвующих в этой операции, и передает эти адреса в соответствующие блоки МПС.

В общем случае УУ микропроцессора формирует управляющие сигналы для выполнения следующих основных процедур:

• выборки из регистра-счетчика адреса команды МПП адреса ячейки ОЗУ, где хранится очередная команда программы;

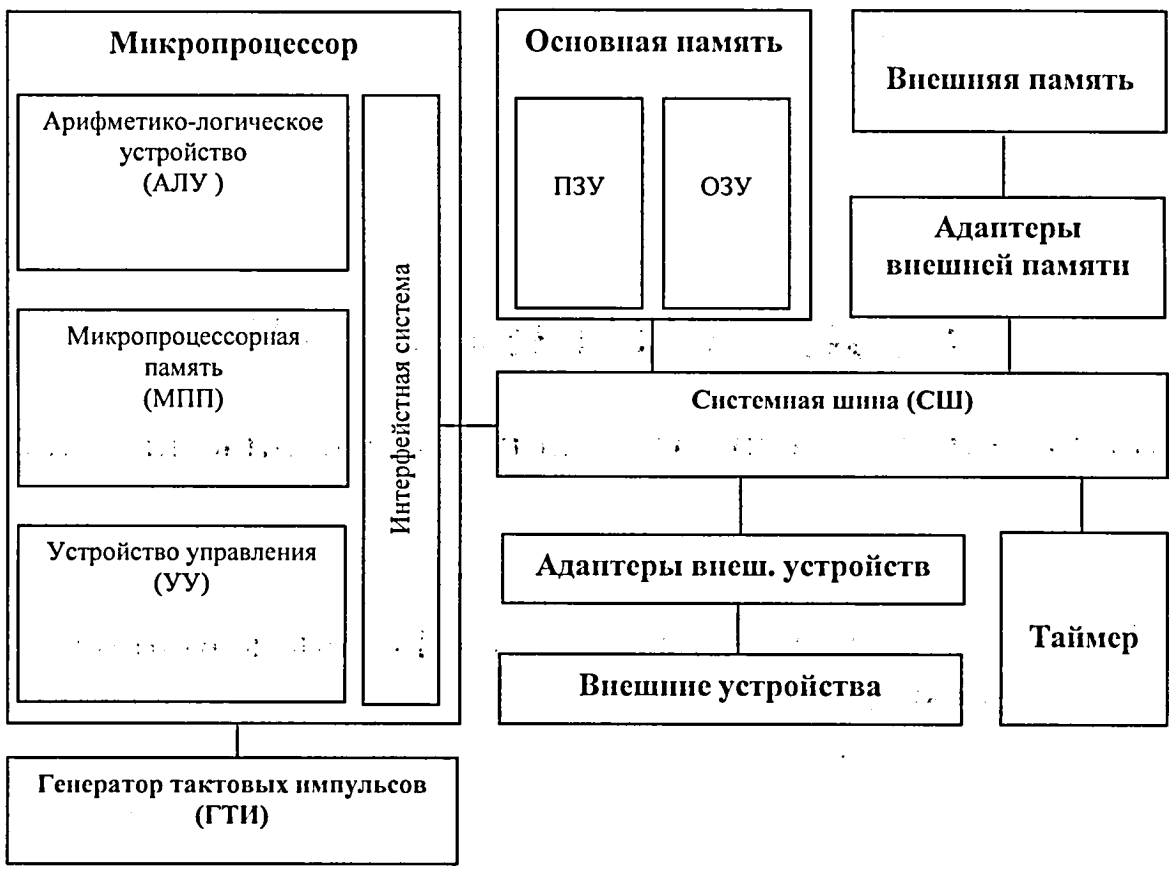


Рис 1. Структурная схема МПС

• выборки из ячеек ОЗУ кода очередной команды и приема считанной команды в регистр команд;

• расшифровки кода операции и признаков выбранной команды;

• считывания из соответствующих расшифрованному коду операции ячеек ПЗУ микропрограмм управляющих сигналов, определяющих для всех блоков машины процедуры выполнения заданной операции, и пересылки управляющих сигналов в эти блоки;

• считывания из регистра команд и регистров МПП отдельных составляющих адресов операндов, участвующих в вычислениях, и формирования

полных адресов операндов;

- выборки операндов (по сформированным адресам) и выполнения заданной операции их обработки;
- записи результатов операции в память;
- формирования адреса следующей команды программы.

2. *Арифметико-логическое устройство (АЛУ)* – предназначено для выполнения различных арифметических и логических операций (в некоторых моделях ПЭВМ для ускорения выполнения операций к АЛУ подключается дополнительный математический сопроцессор);

Сумматор – вычислительная схема, выполняющая сложение поступающих на ее вход двоичных кодов; сумматор имеет разрядность двойного машинного слова.

Регистры – быстродействующие ячейки памяти.

3. *Основная память (ОП)* – служит для хранения, записи и считывания данных. Память делится на постоянную и оперативную.

ПЗУ (ROM) хранит фиксированные программы и данные, оно является энергонезависимым и при выключении питания информацию не теряет,

ОЗУ (RAM) хранит оперативные данные (изменяемые программы, промежуточные результаты вычислений и др.), является энергозависимым и теряет информацию при выключении питания. ОЗУ реализуется на *регистрах,* которые обеспечивают высокую скорость записи и считывания информации, необходимую для эффективной работы быстродействующего микропроцессора;

Регистры общего назначения (РОН) являются универсальными и могут использоваться для хранения любой информации, а *специальные регистры* применяются для хранения определенной информации, например, адреса команды, признаков результата выполнения операций, режимов работы ПЭВМ и др.

4. *Интерфейсная система микропроцессора* – реализует сопряженную связь микропроцессора с другими устройствами ПЭВМ; включает в себя внутренний интерфейс МП, буферные запоминающие регистры, а также схемы

управления системной шиной и портами ввода-вывода (ПВВ), причем последними являются соответствующие регистры, через которые микропроцессор обменивается данными с другими, главным образом периферийными устройствами.

Кроме обозначенных на рис.1 блоков, в состав систем входят обычно и более сложные, чем адаптеры, блоки управления внешними устройствами – *контроллеры.* К их числу относятся, прежде всего, контроллеры прерываний и прямого доступа к памяти. Имеются также контроллеры клавиатуры, дисплея, дисковой памяти и т. д.

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

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

Микропроцессор представляет собой программно-управляемое устройство, обеспечивающее арифметическую и логическую обработку данных и управление вычислительным процессом. Конструктивно МП выполнен на большой интегральной схеме, представляющей собой полупроводниковый кристалл в пластиковом, керамическом или металлокерамическом корпусе, на котором расположены необходимые выводы для приема и выдачи электрических сигналов. Степень интеграции такой схемы определяется размером кристалла и количеством размещенных в нем транзисторов.

Системная шина. Это основная интерфейсная система компьютера, обеспечивающая сопряжение и связь всех его устройств между собой.

Все блоки, а точнее их порты ввода-вывода, через унифицированные разъемы подключаются к шине единообразно: непосредственно или через *контроллеры (адаптеры).* Управление системной шиной осуществляется микропроцессором либо непосредственно, либо, что чаще, через дополнительную микросхему – контроллер шины, формирующий основные сигналы управления.

Важнейшими функциональными характеристиками системной шины являются: *количество обслуживаемых устройств* и *пропускная способность,* т.е. максимально возможная скорость передачи данных Пропускная способность шины зависит от ее разрядности (8-, 16-, 32- и 64) и тактовой частоты, на которой шина работает.

Системная шина физически представляет собой параллельные проводники, расположенные на материнской плате и включающие в себя:

- *шину данных* (ШД), содержащую провода и схемы сопряжения для параллельной передачи всех разрядов числового кода (машинного слова) операнда;
- *шину адреса* (ША), включающую провода и схемы сопряжения для параллельной передачи всех разрядов кода адреса ячейки основной памяти или порта ввода-вывода внешнего устройства;
- *шину инструкций* (ШИ), содержащую провода и схемы сопряжения для передачи инструкций (управляющих сигналов) во все блоки машины;
- шину питания, имеющую провода и схемы сопряжения для подключения блоков ПЭВМ к системе электропитания.

По однонаправленной адресной шине МП посылает адреса, определяя объект, с которым будет обмен, по шине данных (двунаправленной) обменивается данными с модулями (блоками) системы, по шине управления идет обмен управляющей информацией.

Системная шина обеспечивает три направления передачи данных:

- между микропроцессором и основной памятью;
- между микропроцессором и портами ввода-вывода внешних устройств;

• между основной памятью и портами ввода-вывода внешних устройств (в режиме прямого доступа к памяти).

В качестве системной шины в разных ПЭВМ использовались и могут использоваться:

• шины *расширений* – шины общего назначения, позволяющие подключать большое число самых разнообразных устройств;

• локальные *шины,* специализирующиеся на обслуживании небольшого количества устройств определенного типа.

Основная память (ОП). Она предназначена для хранения и оперативного обмена данными с другими блоками машины. ОП содержит два вида запоминающих устройств – *постоянное запоминающее устройство (ПЗУ)* и *оперативное запоминающее устройство (ОЗУ).*

ПЗУ служит для хранения неизменяемой информации (постоянных программ, различных констант, справочных данных и др.) и позволяет ее оперативно только считывать (изменить информацию в ПЗУ нельзя)

ОЗУ предназначено для оперативной записи, хранения и считывания данных, непосредственно участвующих в вычислительном процессе, реализуемом ПЭВМ в текущий период времени. ОЗУ отличается достаточно высоким быстродействием, а также возможностью прямого адресного доступа к каждой ячейке памяти.

Внешняя помять. Она относится к внешним устройствам ПЭВМ и используется для долговременного хранения любой информации, которая может когда-либо потребоваться для решения задач. В ней, в частности, хранится все программное обеспечение компьютера. Наиболее распространенными видами устройств внешней памяти являются накопители на жестких (НЖМД) и гибких (НГМД) магнитных дисках, накопители на оптических дисках (CD-ROM – Compact Disk Read Only Memory – компакт-диск с только читаемой памятью) и др.

Внешние устройства (ВУ). Внешние устройства обеспечивают взаимодействие машины с окружающей средой: пользователями, объектами

управления, другими ЭВМ, вычислительными сетями

К системной шине и микропроцессору ПЭВМ наряду с *типовыми* внешними устройствами могут быть подключены и некоторые дополнительные платы, расширяющие функциональные возможности микропроцессора математический сопроцессор, контроллер прямого доступа к памяти, сопроцессор ввода-вывода, контроллер прерываний и др.

Математический *сопроцессор* используется для ускоренного выполнения операций над двоичными числами с плавающей точкой, над двоично-кодированными десятичными числами, для вычисления некоторых, в том числе тригонометрических функций. В отличие с CPU, он не управляет системой, а ждет команду CPU на выполнение арифметических вычислений и формирование результатов. Он имеет свою систему команд, работает параллельно с центральным МП, но под его управлением. Последние модели МП, начиная с 80486, включают сопроцессор в свою структуру.

Контроллер прямого доступа к памяти освобождает МП от прямого управления накопителями на магнитных дисках. При его использовании данные непосредственно передаются между внешней памятью и ОЗУ, минуя микропроцессор.

Сопроцессор ввода-вывода за счет параллельной работы с центральным МП значительно ускоряет выполнение процедур ввода-вывода при обслуживании нескольких внешних устройств (видеомонитор, принтер, НЖМД НГМД и др.).

Контроллер прерываний обслуживает процедуры прерывания, принимает запрос на прерывание от внешних устройств, определяет уровень приоритета этого запроса и выдает сигнал прерывания в микропроцессор. МП, получив этот сигнал, приостанавливает выполнение текущей программы и переходит к выполнению специальной программы, соответствующей данному типу прерывания. После завершения программы обслуживания прерывания восстанавливается выполнение прерванной программы.

Порты ввода-вывода – это пункты системного интерфейса ПЭВМ, через

которые микропроцессор обменивается данными с другими устройствами. Порт устройства содержит аппаратуру сопряжения и два регистра памяти – для обмена данными и обмена управляющей информацией. Всего портов у микропроцессора может быть 65536. Многие типовые устройства (НЖМД, НГМД клавиатура, принтер и др.) имеют постоянно закрепленные за ними порты ввода-вывода.

ОСНОВНЫЕ ХАРАКТЕРИСТИКИ МИКРОПРОЦЕССОРОВ

В состав процессора входят четыре основные блока:

- арифметико-логическое устройство (АЛУ);
- блок управления;
- блок памяти;
- устройства ввода/вывода.

Данная логическая организация процессора – архитектура – была определена еще в 1946 г. американским ученым Дж. фон Нейманом. Сегодня, больше половины столетия спустя, почти все процессоры имеют архитектуру «фон Неймана».

Производительность МП характеризуется следующими основными параметрами:

- степень интеграции
- внутренняя и внешняя разрядность обрабатываемых данных
- тактовая частота
- память, к которой может адресоваться МП
- объем установленной кэш-памяти

Кроме того, МП различаются по технологии производства, напряжению питания, форм-фактору и др.

Степень интеграции определяет количество транзисторов на кристалле.

Внутренняя разрядность данных. Одной из основных характеристик процессора является количество бит, которое может обрабатывать одновременно: 16, 32 или 64 бита.

Внешняя разрядность данных. Процессор управляет системой, обмениваясь данными с кэш-памятью, RAM и другими устройствами по специальным магистралям, называемым шинами.

Как уже было отмечено, важнейшими характеристиками шины являются разрядность и тактовая частота, потому что они определяют количество бит информации в секунду, которые теоретически можно передавать по шине, – *пропускная способность шины.*

Еще совсем недавно тактовая частота системной шины составляла 66, 100 133 МГц, в настоящее время тактовая частота увеличилась до 200, 400 и 533 МГц счет передачи за один такт 2 или 4 пакетов данных:

133 * 4 = 533 МГц.

Разрядность процессора определяется внутренней, а не внешней разрядностью данных. Например, хотя МП Pentium может одновременно пересылать/получать 64 бита данных он является 32-битным, потому что может обработать одновременно только 32 бита

Тактовая частота. Любой современный PC имеет тактовый генератор *(System Clock),* который синхронизирует работу различных его компонентов. Минимальный промежуток времени, определяемый тактовым генератором, еще называют *циклом.* Частота работы тактового генератора системной шины FSB (Front Side Bus) – внешней шиной процессора – измеряется в мегагерцах (миллион циклов в секунду).

Адресация памяти. Ширина адресной шины определяет количество ячеек, к которым может обратиться МП для чтения или записи. Ширина адресной шины и ширина шины данных не связаны, хотя эти шины работают с одинаковой тактовой частотой.

ОРГАНИЗАЦИЯ И СТРУКТУРА МИКРОПРОЦЕССОРНОЙ СИСТЕМЫ

Принцип построения МП-систем на основе трехшинной архитектуры показан на рис 1.1. Здесь:

DB, AB, CB (Data Bus, Address Bus, Control Bus) – шины данных, адреса и

управления;

CPU (Central Processing Unit) – центральное процессорное устройство, в состав которого входят однокристальный МП, шинные драйверы, приемопередатчики, адресные регистры другие вспомогательные устройста. *CPU* управляет выборкой (чтением) команд, из памяти и передачей данных в МП-системе;

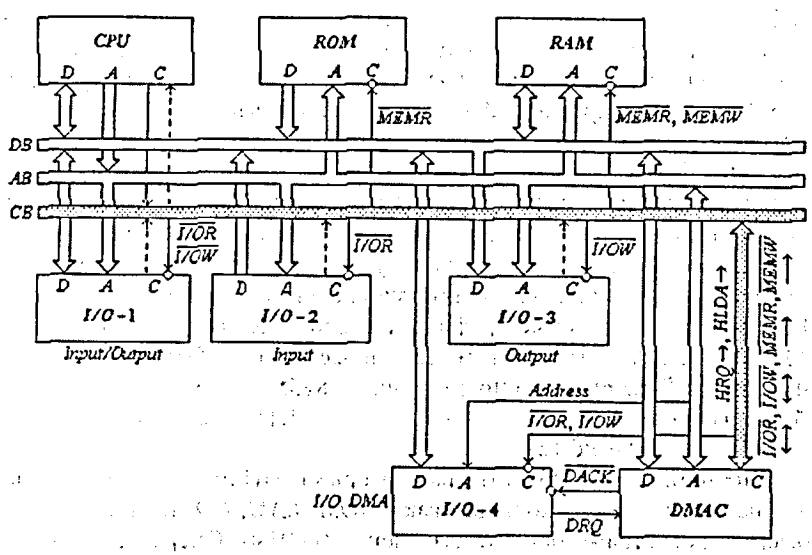


Рис.1.1. Принцип построения МП-систем на основе трехшинной архитектуры

ROM (Read Only Memory – память только для чтения) – ПЗУ;

RAM (Random Access Memory – память с произвольной выборкой) – ОЗУ;

I/O (Input/Output) – устройства ввода-вывода (внешние устройства), которые могут быть только устройствами ввода (I/0-2) или только устройствами вывода (I/0-3);

DMAC (Direct Memory Access Controller) – контроллер прямого доступа к памяти, представляющий собой специализированный процессор ввода-вывода *(DMAC,* как и МП, может управлять шинами). Если шинами управляет МП, то *DMAC* является ведомым системной шины (представляет для МП обычное устройство ввода-вывода – пассивный режим работы *MAC).* При переходе

DMAC в режим ведущего системной шины (активный режим работы МАС) МП предварительно должен перевести свои шины в *Z*-состояние. В активном режиме МАС генерирует адресные сигналы и сигналы управления, обеспечивающие непосредственную передачу данных между системной памятью и внешним устройством *I/0-4*.

В качестве примера рассмотрим МПС на базе микропроцессора 8080А. Графическое обозначение БИС 8080А (отечественный аналог 580ВМ80А) приведено на рис.1.2.

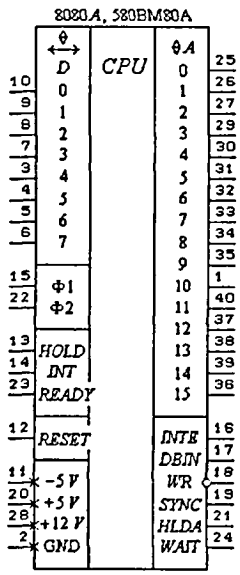


Рис. 1.2. Микропроцессор 580ВМ80А

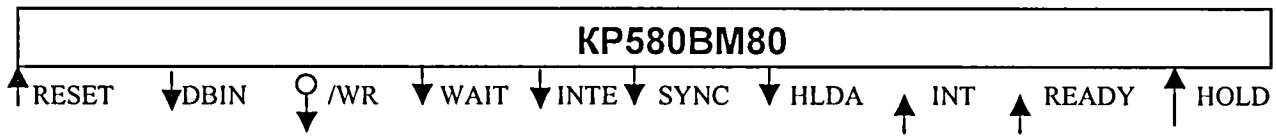
Системная шина управления *СВ,* как правило, формируется с помощью дополнительных аппаратных средств, специально спроектированных для выпускаемых семейств БИС (для серии 580 – системные контроллеры 580ВК28 и 580ВК38).

Центральное процессорное устройство *(CPU* на рис.1.1) при использовании серии 580 состоит из ИС 580ВМ80А (8080А), 580ГФ24 (генератор тактовых сигналов 8224), 580ВК38 (системный контроллер 8238) и 1533АП5*2 (драйверы шины адреса). В любой МП-системе с раздельной адресацией памяти и *I/O* системная шина управления содержит линии управления чтением *ROM* и *RAM,* записью данных в *RAM,* вводом данных *I/O* и

выводом данных в *I/O,* обеспечивающие операции пересылки операндов.

СИГНАЛЫ УПРАВЛЕНИЯ, ФОРМИРУЕМЫЕ МП КР580ВМ80.

На рис. 3.1 показаны сигналы управления формируемые непосредственно МП КР580ВМ80.



В микропроцессорной серии 580 для соответствующих системных сигналов управления приняты обозначения:

MEMR (Memory Read) – чтение команд программы или данных из памяти *(ROM* и *RAM),*

MEMW (Memory Write) – запись данных в память *(RAM),*

I/OR (I/O Read) – ввод данных из внешнего устройства *I/O* (чтение данных),

I/OW (I/O Write) – вывод данных во внешнее устройство *I/O* (запись данных).

Конечно, в МП-системах используются и другие сигналы управления операциями пересылки, например, сигнал *READY* (готовность), выдаваемый памятью или *I/O,* имеющими достаточное быстродействие (при их готовности к приему или выдаче операндов выдается чтение *READY* = 1).

Ввод-вывод по прямому доступу к памяти. Использование МП пересылки операндов между памятью и *I/O* осуществляется программным способом с помощью команд вывода (IN *port* и OUT *port)* и команд записи и чтения памяти. Для пересылки операнда требуется последовательное выполнение двух команд, осуществляющих пересылку операнда между *I/O,* МП и памятью по схеме:

I/O – > МП – > *Memory* или *Memory* – > МП – > *I/O.*

Такой способ пересылки требует большого времени, так как используется промежуточное звено (МП). Контроллер прямого доступа к памяти *DMAC*

используется для реализации пересылок вида

I/O –> Memory или *Memory –> I/O*

с целью их ускорения. Внешнее устройство *I/O*-4 на рис. 1.1 может пересылать операнды только под управлением *DMAC,* который является специализированным процессором ввода-вывода, осуществляющим непосредственную связь между *I/O* и системной памятью *(RAM* и *ROM).* Для аппаратной поддержки ввода-вывода по прямому доступу к памяти фирмой *Intel* была спроектированы БИС 8257 (580ВТ57 и 1810ВТ37А) – программируемые контроллеры прямого доступа к памяти.

В пассивном режиме *DMAC* программируется с помощью команд OUT *port* на выполнение операции записи или чтения памяти, т.е. *DMAC* указывается направление передачи данных типа *I/O*-4 –> *Memory* или *Memory* –> *I/O*-4. При программировании обязательно задается начальный адрес системной памяти *AB* (базовый адрес) и размер пересылаемого блока данных. Взаимодействие *DMAC* с МП осуществляется с помощью сигналов:

DRQ (DMA Request) – сигнал запроса *DMA,* поступающий от *I/O;*

HRQ = HOLD (Hold Request) – сигнал запроса *DMA* (захвата шин), подаваемый от *DMAC* на МП;

HLDA (Hold Acknowledge) – сигнал подтверждения захвата шин, выдаваемый МП на *DMAC с* одновременным переводом своих локальных шин данных, адреса и управления *в Z*-состояние;

DACK (DMA Acknowledge) – сигнал подтверждения *DMA.* Этот сигнал принимает активный уровень (0) при пересылке каждого байта по адресам от предварительно запрограммированного начального адреса A_B до конечного адреса, вычисляемого *DMAC* с помощью указанного при программировании объема пересылаемых данных АЛ.

Для МП 8080А взаимодействие *I/O, DMAC* и МП определяется схемой:

DRQ = 1 => *HRQ* = 1 => *HLDA* = 1 => $\overline{\text{DACK}} = 0$ ... $\overline{\text{DACK}} = 0$

до окончания заданного объема пересылок. После окончания пересылок *DMAC* устанавливает значение сигнала *HRQ* = 0, в ответ на которое МП выдает

значение сигнала *HLDA* = 0, указывающее завершение прямого доступа к памяти. В активном режиме *DMAC* вырабатывает активные уровни пар сигналов управления $\overline{MEMR}$ и $\overline{I/O}$ (при пересылках данных из памяти в *I/O)* или $\overline{MEMW}$ и $\overline{I/OR}$ (при пересылках данных в память из *I/O),* аналогичные системным сигналам управления *CPU.*

Описанная процедура *DMA*-пересылок инициируется запросом *DMA* сигналом *DRQ* = 1, поступающим от внешнего устройства, являющегося в этом смысле активным. Инициатором пересылок может быть и сам МП. В связи с этим все БИС контроллеров *DMA* выполняются так, что могут воспринимать специальные команды разрешения *DMA,* в ответ на которые *DMAC* выдает сигнал *HRQ* = 1 при *DRQ* = 1.

АРХИТЕКТУРА МИКРОПРОЦЕССОРА 8080

Первый в мире однокристальный 8-разрядный МП 8008 был изготовлен фирмой Intel 1972 г., а в 1973 г. эта фирма выпустила 8-разрядный МП 8080.

Структурная схема 8-разрядного МП 8080А представлена на рис.1.3. Основными узлами МП являются:

Data Bus Buffer/Latch – буфер шины данных с фиксацией данных, обеспечивай: направленную связь МП с памятью (ОЗУ, ПЗУ) и внешними устройствами *I/O* по 8-разрядной шине данных $O_{7\text{-}0}$ (приемопередатчик с *Z*-состоянием выходов). В течение первого такта каждого машинного цикла МП на шину данных выводит слово состояния *SW;*

Instruct. RG (Instruction Register) – 8-разрядный регистр памяти команд (инструкций), выбираемых МП из ПЗУ или ОЗУ при выполнении программы;

Instruction Decoder and Machine Cycle Encoding – дешифратор команд (инструкций) и шифратор машинных циклов, преобразующий код машинной команды в последовательность внутренних управляющих сигналов (обычно реализуется на программируемых логических матрицах). Машинный цикл – это число тактов сигнала синхронизации затрачиваемых на пересылку одного байта

данных по внешней шине данных МП D_{7-0}. Команды выполняются за 1 - 5 машинных циклов, а машинные циклы – за 3 - 5 тактов тактового сигнала. Самые «быстрые» команды выполняются за 4 такта (один машинный цикл), а самая медленная – за 18 тактов (пять машинных циклов);

Timing and Control – устройство синхронизации и управления, обеспечивающее как управление всеми внутренними узлами, так и связь с внешней средой;

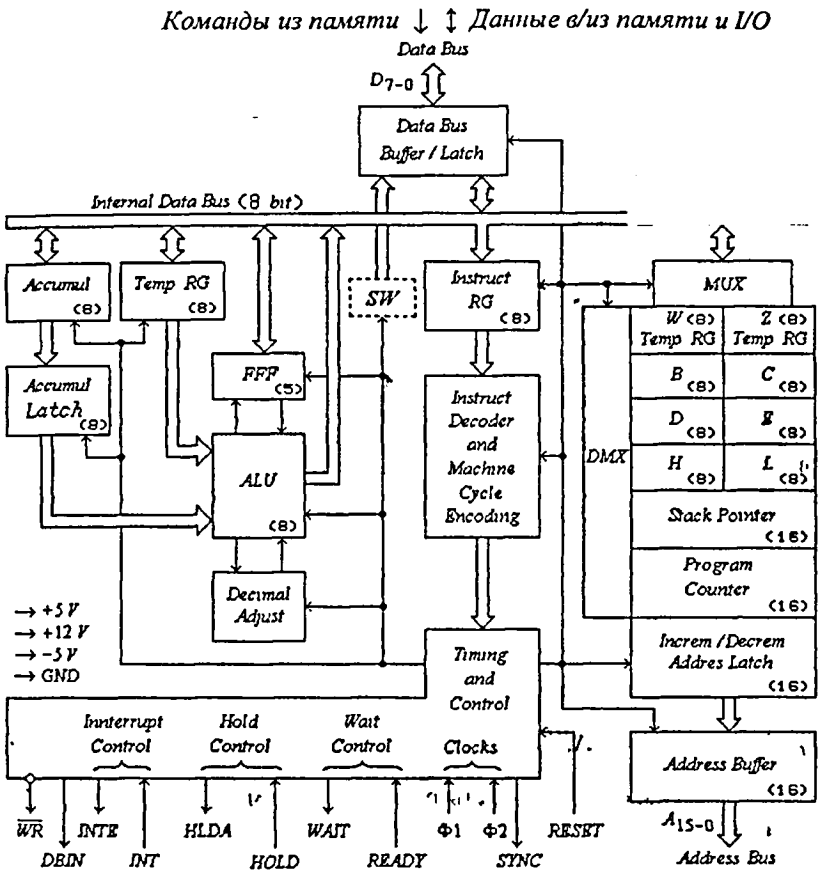


Рис. 1.3. Структурная схема микропроцессора 8080A

Internal Data Bus – 8-разрядная внутренняя шина данных, обеспечивающая связь между регистрами памяти МП;

B, C, D, E, H и L – регистры общего назначений (РОНы), предназначенные для хранения операндов в процессе выполнения программ и имеющие связь с шиной данных D_{7-0} (программно доступны). РОНы могут объединяться в регистровые пары *rp B, D* и H (регистры *B* и C, *D* и *E, H* и *L – Register Pairs)* для

хранения 16-разрядных двоичных чисел (слов) и выполнения операций над ними *(B, D* и *H* – старшие байты слов, а *C, E* и L – младшие байты слов);

W, Z и *Temp.RG (Temporary Register)* – регистры временного хранения операндов, используемые при внутренних пересылках операндов (программно недоступны);

PC (Program Counter) – 16-разрядный программный счетчик, задающий адреса команд программы, выбираемых из памяти;

SP (Stack Pointer) – 16-разрядный указатель стека, адресующий специальную область ЗУ, называемую *стеком* и используемую для хранения адресов возврата из подпрограмм и операндов при выполнении некоторых команд. Стек представляет собой магазинную память типа *LIFO (last-in, first-out* – последним вошел, первым вышел);

Incrementer / Decrementer, Address Latch – блок модификации адреса команды (+1/-1) с его фиксацией в регистре памяти *(Address Latch),* обеспечивающий параллельное выполнение операций вычисления следующего адреса команды программы, выбираемой из памяти.

Address Buffer – буфер адреса $A_{15\text{-}0}$ с *Z*-состоянием выходов, выдающий адреса устройств ввода-вывода при выполнении команд программы. Адреса памяти могут пост как из программного счетчика PC и указателя стека *SP,* так и из регистров общего назначения *B* и C, *D* и *E, H* и *L*;

MUX (Multiplexer) – мультиплексор, обеспечивающий под управлением устройства *Timing and Control* коммутацию регистров, выдающих операнды на внутреннюю шину данных; (мультиплексор производит выбор источника операнда, указанного в выполняемой команде);

DMX (Demultiplexer) – демультиплексор, производящий выбор одного из регистров, указанного в выполняемой команде в качестве устройства назначения (приемника операнда);

ALL/ (Arithmetic and Logic Unit) – арифметическо-логическое устройство, выполняющее арифметические и логические операции над операндами,

Accumulator – буфер аккумулятора, обеспечивающий связь регистра,

называемого аккумулятором, с внутренней шиной данных;

Accumulator Latch – аккумулятор, выдающий на *ALU* один из операндов при выполнении арифметических и логических операций и принимающий результат выполненной операции;

Decimal Adjust – схема десятичной коррекции, производящая учет переносов при выполнении команд десятичной арифметики;

FFF (Flag Flip Flops) – регистр признаков *F* (регистр флагов F), фиксирующий пять результатов выполнения операции в *ALU*;

SW (Status Word) – устройство формирования слова состояния МП, выдающее на шину данных код операции, которую будет выполнять МП в данном машинном цикле. По этому коду системный контроллер 580ВК28 формирует активный уровень одного из системных сигналов управления:

$\overline{MEMR}$ (чтение байта данных из памяти),

$\overline{MEMW}$ (запись байта данных в память),

$\overline{I/OR}$ (чтение байта данных из внешнего устройства),

$\overline{I/OW}$ (запись байта данных во внешнее устройство),

$\overline{INTA}$ *(Interrupt Acknowledge* – подтверждение прерывания).

Сигналы микропроцессора. Для синхронизации работы МП и обеспечения его взаимодействия с внешней средой используются сигналы (8 линий данных, 16 линий адреса и 12 линий управления):

$D_{7\text{-}0}$ *(Data Bus)* – двунаправленная шина данных, обеспечивающая связь МП с памятью *(RAM* и *ROM)* и внешними устройствами *(I/O)* при передаче команд программы и данных. Шина данных используется в мультиплексном режиме для последовательной во времени выдачи кода слова состояния *SW* (одновременно с выдачей адресов $A_{15\text{-}0}$) и затем для приема /передачи данных $D_{7\text{-}0}$, т. е. МП имеет совмещенную (мультиплексную) шину управления-данных. Слово состояния *SW* должно фиксироваться во внешнем регистре памяти (из-за

мультиплексного режима) для последующей его дешифрации, т.е. преобразования в системные сигналы управления памятью и внешними устройствами $\overline{MEMR}$, $\overline{MEMW}$, $\overline{I/OR}$, $\overline{I/OW}$ и $\overline{INTA}$. При выполнении команд программы на шину данных может выдаваться содержимое регистров A *(Accumulator), B, C, D ,E, H, L, PC* и регистра признаков *FFF (Flag Flip Flops);*

$A_{15\text{-}0}$ *(Address Bus)* – шина адреса, обеспечивающая адресацию всех устройств, подключенных к МП. Максимальный объем адресуемой памяти равен 64 Кбайт (2^{16}*8 бит) или 65536 байт в десятичном представлении. Адресация внешних устройств производится по шине адреса только одним байтом, т.е. в МП-системе можно использовать не более $2^8 = 256$ устройств ввода и 256 устройств вывода. При выполнении команд ввода IN *port* и вывода OUT *port* на шину адреса $A_{15\text{-}0}$ МП выдает два одинаковых байта

$$A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 = A_{15} A_{14} A_{13} A_{12} A_{11} A_{10} A_9 A_8,$$

т.е. $(A_i = A_{i+8})$, $i = 0,1,...7$, любой из которых можно использовать для адресации внешних устройств. При выполнении команд программы на шину адреса может выдаваться содержимое регистров *PC, SP, B* и *C, D* и *E, H* и *L;*

Ф1, Ф2 (Clock Phases) – сигналы двухфазной синхронизации (тактовые сигналы), подаваемые от генератора 580ГФ24;

SYNC (Synchronizing Signal) – сигнал синхронизации, указывающий начало каждого машинного цикла *(SYNC* = 1 при выдаче на шину данных слова состояния *SW);*

$\overline{WR}$ *(Write* – запись) и *DBIN (Data Bus In* - ввод с шины данных) – сигналы, указывающие внешней среде на выполнение микропроцессором операций записи/вывода $(\overline{WR} = 0)$ или чтения/ввода *(DBIN*=1). Полное описание этих операций приведено в табл.1.1. При значении сигнала $\overline{WR} = 0$ данные $D_{7\text{-}0}$, выданные микропроцессором, имеют истинное значение (данные устойчивы – переходные процессы закончились);

READY (готовность) и *WAIT* (ожидание, или подтверждение состояния ожидания) – сигналы квитирования чтения-записи и ввода-вывода, используемые для синхронизации операций передачи данных между МП и памятью или внешними устройствами, требующими большей длительности активных уровней сигналов управления записью/чтением ($\overline{WR}=0$/ *DBIN* =1), чем их стандартная длительность Значение сигнала *READY* = 0 должно поступать от памяти или внешних устройств, имеющих недостаточное быстродействие. Значение *READY* = 0 переводит МП в состояние ожидания до тех пор, пока не будет получено значение *READY* = 1 (МП в ответ на значение сигнала *READY* = 0 выдает значение сигнала *WAIT* = 1 – ввод-вывод по готовности;

INT (Interrupt Request – запрос прерывания) и *INTE (Interrupt Enable* – разрешение прерывания) – сигналы квитирования ввода-вывода по прерыванию. Значение сигнала *INTE* указывает состояние внутреннего триггера разрешения прерывания. Если *INTE* = 0, то запросы прерывания *INT* =1 не воспринимаются. Сброс триггера *INTE* в 0 производится как программным, так и аппаратным способами, а установка в 1 – только программным способом;

HOLD (запрос захвата шин) и *HLDA (Hold Acknowledge* – подтверждение захвата шин) – сигналы квитирования ввода-вывода по прямому доступу к памяти. В ответ на значение сигнала *HOLD*=1 МП переводит свои шины данных и адреса в Z-состояние (МП отключается от шин) и выдает значение *сигнала HLDA* = 1;

RESET – сигнал системного сброса, подаваемый при включении питания МПС *(RESET*=1) и производящий сброс в 0 программного счетчика *PC (*PC <-0 – старт программы производится с адреса 0), триггера *HLDA* и триггера *INTE (INTE* <-0 – прерывания запрещены), которым соответствуют внешние сигналы *HLDA* и *INTE.* Длительность значения сигнала *RESET* =1 должна быть не менее трех периодов тактового сигнала Φ_2.

Программный счетчик *PC.* Команды МП 8080А состоят из одного, двух

или трех байт. Байты команд, составляющих программу, располагаются в памяти последовательно в порядке возрастания адресов. После выполнения очередной команды программный счетчик *PC* указывает, где в памяти расположен первый байт следующей команды выполняемой программы. Устройство управления увеличивает содержимое счетчика *PC* на единицу (инкремент *PC)* всякий раз, когда очередной байт команды передается (читается) из памяти в МП. Если команда состоит из двух или трех байт, то ее выборка происходит за два и три машинных цикл соответственно (машинный цикл – от 3 до 5 тактов сигналов синхронизации Φ_1 и Φ_2, затрачиваемых на пересылку одного байта по внешней шине данных МП). Таким образом, перед началом выборки следующей команды счетчик *PC* уже содержит адрес ее первого байта.

Команды выполняются в той же последовательности, в которой они расположены в памяти. Изменить этот порядок выполнения программист может с помощью команд передачи управления – команд переходов JMP *addr* и команд вызова подпрограмм CALL *addr* и *RST n* (n = 0.. 7). Команды JMP *addr* и CALL *addr* во втором и третьем байтах содержат два байта адреса *addr,* который автоматически загружается в программный счетчик *PC* при выполнении этих команд:

PC <- *addr* = 0000h ... FFFFh,

где *h* – указатель 16-ричной системы счисления.

Принцип работы стека. Для аппаратной поддержки выполнения вложенных программ идеально подходит магазинная память типа *LIFO (last-in, first-out* - последним вошел, первым вышел), называемая стеком *(Stack).* Такая память автоматически обеспечивает последовательное сохранение нескольких адресов возврата из вложенных подпрограмм и последовательное их извлечение в обратном порядке (последний адрес возврата, включённый в стек, извлекается первым).

Адрес возврата – адрес первого байта команды, непосредственно следующей за командой CALL *addr.* В результате выполнения команды CALL

addr содержимое указателя стека *SP* уменьшится на 2. Если в вызванной подпрограмме имеется другая команда CALL *addr*1 (вложенные подпрограммы), то описанные выше действия будут повторены – в стек будет включен еще один адрес возврата. Число вложений подпрограмм ограничено только объемом памяти, который можно отвести в МП-системе под стек за счет уменьшения памяти данных.

Каждая подпрограмма должна заканчиваться командой возврата из подпрограммы RET, при выполнении которой автоматически последовательно производятся операции:

PCl <- *Stack, SP*<-*SP*+1, *PCh* <- *Stack, SP*<-*SP*+1 – извлечение из стека адреса возврата.

В результате выполнения команды RET содержимое указателя стека *SP* увеличится на 2 и выполнение основной программы продолжится с того места, в котором произошло обращение к подпрограмме. Вызов одной и той же подпрограммы может производиться любое число раз из разных мест основной программы. Автоматический декремент при включении байта в стек и инкремент при извлечении байта из стека избавляют программиста от задания адресов ячеек памяти в явном виде.

В МП используются только команды, которые включают и извлекают из стека слова – два байта адреса или данных. Для включения в стек и извлечения из стека слов данных предназначены команды PUSH и POP соответственно. Итак, стек используется только при выполнении команд CALL, RST, RET, PUSH и POP (имеются также команды условного вызова подпрограмм и условного возврата из подпрограмм, аналогичные по выполняемым операциям командам безусловного вызова подпрограмм CALL *addr* и безусловного возврата из подпрограмм RET). При выполнении любой из этих команд МП на шину адреса $A_{15\text{-}0}$ выдает содержимое указателя стека *SP*.

После включения питания МП-системы сразу же следует выполнить инициализацию указателя стека *SP*: командой LXI SP, *d*16 (*d16*=0000h... FFFFh) необходимо записать в указатель стека некоторый адрес, задающий начальную

вершину стека *(Top of Stack)*.

Указатель стека *SP* в любой момент времени указывает на вершину стека, т. е. на последний элемент, включенный в стек. Содержимое 8-разрядных ячеек памяти стека изменяется только, при включении в него новых слов. При извлечении же слов из стека содержимое ячеек памяти не изменяется, как и при чтении данных из ОЗУ.

Практическая реализация МПС. На рис.1.5 показана структурная схема МПС, построенной на основе МП 8080А. Центральное процессорное устройство *(CPU)* реализовано на трех кристаллах: 8080А – микропроцессор, 8224 (580ГФ24) – генератор тактовых сигналов, 8238 (580ВК38) – системный контроллер. 8238 по принятому коду слова состояния *SW* в определенные моменты времени вырабатывает активный уровень (0) только одного системного сигнала управления $\overline{MEMR}$, $\overline{MEMW}$, $\overline{I/OR}$, $\overline{I/OW}$ или $\overline{INTA}$. Системные шины *(System Bus)* содержат все сигналы, необходимые для подключения к нему памяти (обычно более одной БИС *ROM* и *RAM*) и внешних устройств (обычно более одной БИС *I/O*). У БИС *ROM* вход $\overline{WE}$ *(Write Enable* – разрешение записи) отсутствует.

Часть адресных сигналов из $A_{15\text{-}0}$ подается непосредственно на адресные входы A_i БИС *ROM* и *RAM* и *I/O* ($i = 0...m$, где $m<15$ для БИС *ROM* и *RAM* и $m<=3$ для большинства БИС *I/O*).

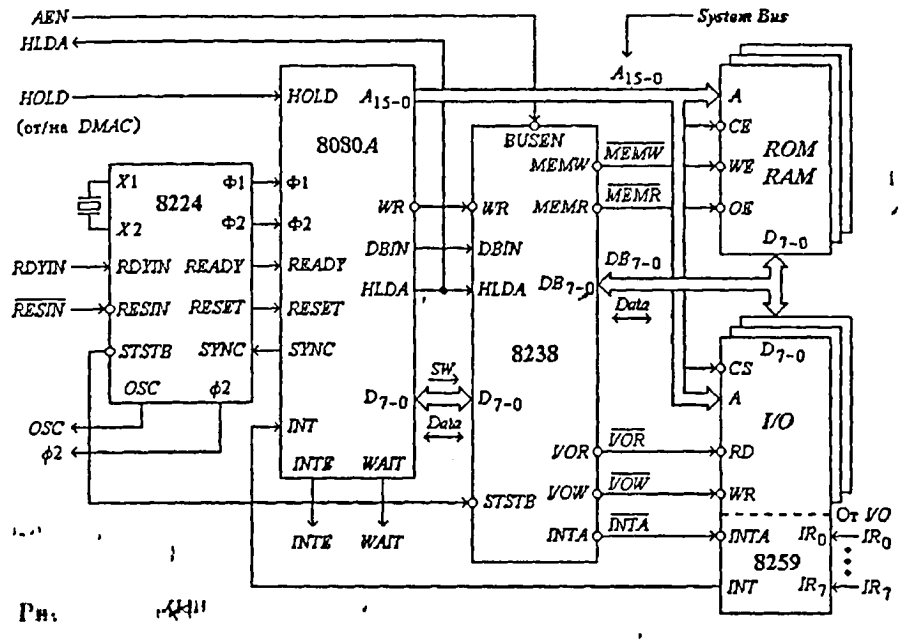


Рис. 1.5. Структурная схема МПС на основе МП 8080А

Эти адресные сигналы поступают на внутренние дешифраторы БИС для селекции (выбора) одной из 2^{m+1} ячеек памяти в *ROM* и *RAM* или одного из 2^{m+1} регистра памяти во внешнем устройстве *I/O*. Остальные адресные сигналы A_j ($j = 0...m+1...15$ для *ROM* и *RAM* и $m+1...7$ для *I/O*) подаются на внешние дешифраторы, выходные сигналы которых используются для селекции (выбора) одной из нескольких БИС *ROM, RAM* и *I/O*. Эти сигналы подаются на входы выбора кристалла $\overline{CE}$ и $\overline{CS}$ (*Chip Enable, Chip Select* – разрешение кристалла) для включения только одной БИС.

Передача данных между МП и памятью происходит только при совпадении во времени активных уровней (0) адресного сигнала $\overline{CE}$ и сигнала управления $\overline{MEMR}$ или $\overline{MEMW}$. Передача данных между МП и внешними устройствами *I/O* происходит только при совпадении во времени активных уровней (0) адресного сигнала $\overline{CS}$ и сигнала управления $\overline{I/OR}$ или $\overline{I/OW}$. Если активные уровни сигналов $\overline{CE}$ и $\overline{MEMR}$ какой-либо БИС памяти не совпадают во времени, то ее выходы данных D_{7-0} находятся в *Z*-состоянии.

Если активные уровни сигналов $\overline{CS}$ и $\overline{I/OR}$ какой-либо БИС *I/O* не совпадают во времени, то ее выходы данных $D_{7\text{-}0}$ находятся в *Z*-состоянии. При правильно выполненной дешифрации адресных сигналов в каждый момент времени может быть включена (выбрана) только одна БИС.

Если в МПС ввод-вывод по прямому доступу к памяти не используется, то следует положить *HOLD = 0* и *AEN* = 0, т. е. входы *HOLD* и $\overline{BUSEN}$ *(Bus Enable* – разрешение шины) необходимо подключить к земле.

Если в МПС использованы достаточно быстродействующие память и внешние устройства, то следует задать *READY* = 1, что обеспечивается подключением входа *RDYIN (Ready Input) к* источнику питания V_{cc} = +5 В.

Ввод-вывод по прерыванию в стандартных МПС обеспечивается контроллером прерываний 8257. Команды CALL *addr_m* поступают в МП из контроллера прерываний 8259 по шине данных $D_{7\text{-}0}$. Чтение трехбайтовых команд CALL *addr_m* из контроллера прерываний безадресное (адресный сигнал выбора кристалла $\overline{CS}$ может иметь произвольное значение), так как чтение команд CALL *addr_m* выполняется сигналом $\overline{INTA}$, который больше нигде не используется. Это единственный случай, когда команды в МП поступают не из памяти, а из внешнего устройства – в данном случае из БИС 8257.

МАШИННЫЕ ЦИКЛЫ

Команды, выбираемые МП из памяти, выполняются под воздействием тактовых сигналов Φ_1 и Φ_2, частота которых определяет производительность МПС, построенной на основе МП8080. В МП используется программное управление для реализации большого числа команд различной сложности. В зависимости от типа команды для ее выполнения требуется вполне определенное число тактов.

Идеальной с точки зрения производительности МП была бы реализация выполнения каждой команды за один такт. Однако, при большом числе

различных команд, выполняемых МП, это потребовало бы значительного увеличения площади кристалла, отводимой под схему управления (в настоящее время разработаны МП с сокращенной системой команд – RISC-процессоры, большинство команд которых выполняется за один такт). В традиционных МП для упрощения схемы управления при большом числе команд их выполнение разделено на микрооперации, каждая из которых выполняется за один такт. Определенное число тактов группируется в *машинный цикл*, в течение которого выполняется одно обращение к шине данных для приема или выдачи одного байта во внешнюю среду.

Число обращений к шине данных, необходимое для выборки команды из памяти и ее выполнения, определяет число машинных циклов и время выполнения команды. В МП используются 1-, 2- и 3-байтовые команды. В первом машинном цикле (*M*1) всегда производится выборка (чтение) из памяти первого байта команды, в котором содержится код операций (*Opcode).* Декодировав первый байт команды, МП «узнает», сколько байт она содержит. Выполнив чтение из памяти всех байт команды, МП исполняет ее. Если при исполнении требуется передача по системной шине данных одного, двух или четырех байт данных, то число машинных циклов выполнения команды будет больше числа байт в команде на 1, 2 или 4 цикла. Команды в зависимости от их типа выполняются за 1…5 машинных циклов, каждый из которых выполняется за 3…5 тактов. За один цикл могут выполняться только те однобайтные команды, код операций которых задает операции только над внутренними объектами МП. Простые команды выполняются за 4 такта, самая сложная команда – за 18 тактов.

Рассмотрим процессы, протекающие в МП при выполнении команды OUT *port.* По адресу, находящемуся в программном счетчике *PC,* МП после окончания выполнения предыдущей команды переходит к выборке из памяти первого байта команды OUT *port.* В первом такте машинного цикла *M*1 МП выдает на шину адреса $A_{15\text{-}0}$ адрес первого байта команды, находящейся в памяти. Одновременно с адресом МП по шине данных $D_{7\text{-}0}$ выдает слово

состояния *SW*= 10100010 – код выборки команды, которое значением сигнала $\overline{STSTB}=0$ записывается в регистр памяти системного контроллера 8238. Выдача слова состояния *SW* сопровождается значением сигнала *SYNC* = 1, по которому генератор тактовых сигналов 8224 и вырабатывает значение сигнала $\overline{STSTB}=0$. Системный контроллер дешифрирует код слова состояния *SW* и выдает активный уровень сигнала чтения памяти $\overline{MEMW}=0$, устанавливающий на шине данных значение $D_{7\text{-}0}$= *DI (Data Input)* – содержимое ячейки памяти. Сказанное описывается следующей схемой:

$SYNC=1 \Rightarrow \overline{STSTB}=0 \Rightarrow \overline{MEMW}=0$ (последовательность сигналов)

от МП на 8224 от 8224 на 8238 от 8238 на память

SW => Reg 8238 => МП <- КОП (последовательность передач по ШД)

из МП в 8238 фиксация SW в Reg из памяти в МП

Код первого байта команды, поступивший по шине данных $D_{7\text{-}0}$ в регистр *Instruct. RG*, подается на дешифратор команды и шифратор машинных циклов *(Instruction Decoder and Machine Cycle Encoding),* который сообщает схеме управления МП, сколько байт содержится в выбираемой команде. Далее МП выполняет машинный цикл M_2 для выборки из памяти второго байта команды OUT *port* (при этом также выдается слово состояния *SW* = 10100010

Для исполнения команды OUT *port* требуется один дополнительный машинный цикл M_3. В первом такте этого цикла МП выдает на шину адреса $A_{15\text{-}0}$ значения

$$A_7A_6A_5A_4A_3A_2A_1A_0 = A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}A_9A_8 = port$$

(port – число, прочитанное из памяти в машинном цикле M_2), а на шину данных – код слова состояния *SW*= 00010000 (см. табл. 1.15 в § 1.9), по которому системный контроллер 8238 формирует активный уровень сигнала управления $\overline{I/OW}=0$, поступающий на вход записи $\overline{WR}$ внешнего устройства (см. рис. 1.5) в момент времени, когда МП выводит на шину данных из аккумулятора *A*

байт данных *DO = A (DO – Data Output*). После завершения выполнения текущей команды МП автоматически переходит к выборке первого байта следующей команды. Этот процесс продолжается до выполнения всей программы.

Сигналы квитирования передачи данных *READY, INT* и *HOLD* анализируется микропроцессором в определенных тактах машинных циклов.

Сигнал готовности *READY.* Этот сигнал анализируется в такте T_2 каждого машинного цикла. Если будет обнаружено значение *READY* = 0, то МП между тактами T_2 и T_3 вводит целое число тактов ожидания T_w. Такты ожидания T_w вводятся до тех пор, пока сигнал *READY* не изменится с 0 на 1. В тактах ожидания T_w сигналы на всех шинах МП изменяют те же значения, что и в такте T_2 (изменяется только сигнал *WAIT* с 0 на 1). Этим достигается увеличение длительности активных уровней системных сигналов $\overline{MEMR}$, $\overline{MEMW}$, $\overline{I/OR}$ и $\overline{I/OW}$.

Сигнал запроса прерывания *INT.* Значение сигнала запроса прерывания *INT* = 1 записывается во внутренний триггер запроса прерывания в последнем такте последнего машинного цикла текущей команды при условии, что установлены значения сигналов *INTE* = 1 (прерывания разрешены) и *HOLD* = 0 (нет запроса прямого доступа к памяти). Микропроцессор, завершив выполнение текущей команды, переходит к выполнению машинного цикла "Подтверждение прерывания" – цикл M_1. В такте T_1 этого цикла МП сбрасывает в 0 внутренний триггер *INTE* (дальнейшие прерывания запрещаются) и на шину данных $D_{7\text{-}0}$ выдает код слова состояния *SW* = 00100011, по которому системный контроллер 8238 вырабатывает активный уровень сигнала $\overline{INTA}$ (рис.1.5). Внутренний триггер запроса прерывания сбрасывается в 0 в такте T_2 машинного цикла M_1.

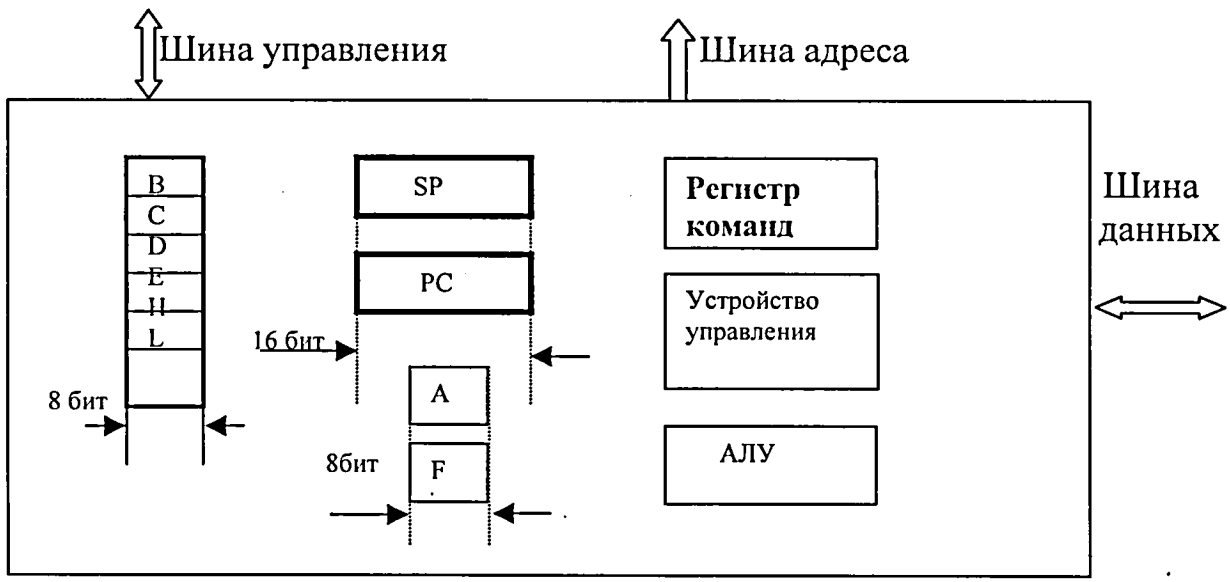
При значении сигнала *INTE* = 1 запрос прерывания *INT*= 1

воспринимается микро сором также в состоянии его останова, которое наступает после выполнения команды (при этом выдается код слова состояния *SW* = 00101011, по которому также вырабатывается активный уровень сигнала $\overline{INTA}$).

Сигнал запроса прямого доступа к памяти *HOLD*. Этот сигнал анализируется в тактах T_2 и T_w каждого машинного цикла. Если будет обнаружено значение *HOLD* = 1 при условии, что значение сигнала готовности *READY* = 1, то МП прерывает работу после завершения текущего машинного цикла выполняемой команды (шины данных и адреса МП переводятся в *Z*-состояние), выдав значение сигнала подтверждения захвата шин *HLDA* = 1. При выполнении операций чтения памяти ($\overline{MEMR} = 0$) или ввода (*I/OR* = 0) значение *HLDA* = 1 выдается в такте T_3 по положительному фронту тактового сигнала Φ_2. При выполнении же операции записи в память (*MEMW* = 0) или вывода ($\overline{I/OR} = 0$) значение *HLDA* = 1 выдается в такте, следующим за тактом T_3, также по положительному фронту тактового сигнала Φ_2.

Запрос прямого доступа к памяти *HOLD* = 1 воспринимается микропроцессором в состоянии его останова, которое наступает после выполнения команды *HLT* (при переходе в состояние останова МП выдает значение сигнала *WAIT* = 1 и переводит шины данных и адреса в *Z*-состояние).

РЕГИСТРОВАЯ МОДЕЛЬ МП КР580ИК80



B,C,D,E,H,L – регистры общего назначения микропроцессора. Емкость каждого из них 8 бит (или 1 байт). В них можно записать и считать (из них) любые данные.

A – аккумулятор, который также относится к регистрам общего назначения. Аккумулятор представляет важнейший регистр микропроцессора, т.к. большинство команд по умолчанию используют А.

PC – программный счетчик.

SP – указатель стека.

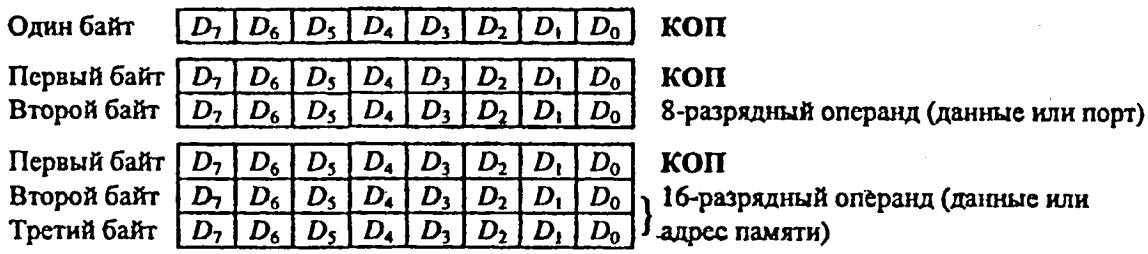
F – регистр флагов.

Флаги МП:

- ZF – флаг нуля (если предыдущий результат равен нулю, этот флаг равен 1);
- *SF* – флаг знака (соответствует старшему биту результата);
- CF – флаг переноса (устанавливается в 1 при переносе из старшего бита результата);
- PF – флаг паритета (устанавливается при четном количестве единиц в результате);

ФОРМАТ И СИСТЕМА КОМАНД МИКРОПРОЦЕССОРА ВМ80

Команды МП 8080А представляются одним, двумя и тремя байтами (см.рис.) Многобайтные команды размещаются в последовательных ячейках памяти (адрес первого байта команды является адресом всей команды). Чем больше байт в команде, тем больше времени затрачивается для ее выборки из памяти.



Код операции (КОП) всегда находится в первом байте команды. Второй и третий байт команды используются для задания операндов (данных, портов устройств ввода-вывода и адресов памяти) Так как КОП задается только одним байтом, то всего может быть не более $2^8 = 256$ различных команд. Все множество КОП по числу байт в команде делится на подмножества:

1 байт – 200, 2 байта – 18, 3 байта – 26

Биты 7-й и 6-й в КОП определяют код команды. Биты 5-й, 4-й и 3-й – регистр приемника. Биты 2-й, 1-й и 0-й определяют регистр источника.

Таблица X.X. Коды регистров

Разряды			Имя регистра
D5	*D4*	*D3*	
D2	*D1*	*D0*	
1	1	1	*A*
0	0	0	*B*
0	0	1	*C*
0	1	0	*D*
0	1	1	*E*
1	0	0	*H*
1	0	1	*L*
1	1	0	*M*

Пример. Команда *MOV C, B* осуществляет пересылку из регистра *B* в регистр *C*. Код данной команды 01 001 000. Данная команда не влияет на регистр флагов.

Для выполнения команды *MOV* требуется машинный цикл, если в команде не участвует ячейка памяти *M*. Если в команде участвует ячейка памяти *M*, то для выполнения команды требуется 2 машинных цикла.

Таблица X.XX. Коды регистровых пар

Разряды		Регистровая пара *(rp)*	Регистры
D_5	D_4		
0	0	*B*	*B* и *C*
0	1	*D*	*D* и *E*
1	0	*H*	*H* и *L*
1	1	*SP* или *PSW*	*PSW*= *A* и *F*

Пример. МК = $00D_5D_40001$ для команд *LXI rp, d16*

МК = $01D_5D_4\ D_3\ D_2\ D_1\ D_0$ для команд MOV *r1, r2*

МК= $11D_5D_40101$ для команд *PUSH rp*

МК = $11D_5D_40001$ для команд *POP rp*

Кроме адресации памяти и устройств ввода-вывода, выполняемой по шине адреса $A_{15\text{-}0}$, необходима адресация РОНов и регистровых пар. Эта адресация

производится частью разрядов КОП $D_5 - D_0$, коды которых приведены в табл. Х.Х. Из примера машинного кода (МК) команды *MOV r1, r2* видно, что 01 задает операцию *MOV* – пересылку содержимого регистра *r2* в регистр *r1,* т.е. машинные коды всех команд передачи данных типа *MOV rl, r2* задаются значениями 40h... 7Fh, за исключением МК = 76h, которому должна была бы соответствовать команда *MOV M, M.* Но команд, оперирующих с двумя операндами в памяти нет.

Пример. Найти машинный код команды *MOV E, C.*

r1 = 011 для регистра *E* и *r2* = 001 для регистра *C*, поэтому МК = 01 011001= 59h.

АДРЕСАЦИЯ ДАННЫХ И ПЕРЕХОДОВ

Микропроцессор в каждом машинном цикле может выдавать на шину адреса только одно из значений 0000.., FFFFh, которое адресует один из операндов, находящихся в памяти. Поэтому в двухоперандных командах второй операнд всегда должен находиться во внутренних регистрах МП, адресуемых частью разрядов КОП (см.тему «Формат и система команд микропроцессора ВМ80»). В МО 8080 используются четыре режима адресации данных:

- непосредственный,
- прямой,
- регистровый,
- косвенно-регистровый

и два метода адресации переходов:

- прямой
- косвенно-регистровый.

Непосредственная адресация данных. При такой адресации 8-разрядные (*d8*) или 16-разрядные (*dl6*) данные представляются вторым или вторым и третьим байтами команды.

Прямая адресация данных. В этом случае адрес операнда находится во втором и третьем байтах команды. Эти два байта задают адрес ячейки памяти, в

которой находится операнд. Формально к методу прямой адресации можно причислить и команды ввода и вывода IN *port* и OUT *port,* выполняющие операции A <– IN *port* и OUT *port* <– A, так как адрес операнда находится во втором байте команды, но, как правило, внешние устройства не используются для хранения оперативных данных с последующим их использованием в вычислительном процессе (для этого целесообразно использовать оперативную память).

Регистровая адресация данных. При такой адресации в коде команды адресуются регистры или регистровые пары, в которых хранятся операнды.

В двухоперандных командах для каждого операнда может использоваться свой режим адресации данных. Так, в последней команде имеют место регистровая и непосредственная адресации операндов. Достоинство регистровой адресации – команды однобайтовые (меньше места занимают в памяти программ).

Косвенно-регистровая адресация данных. В этом случае адрес операнда, расположенного в памяти *M,* определяется содержимым одной из 16-разрядных регистровых пар *B, D* или *H.* В командах LDAX и STAX (команды передачи данных) для адресации памяти можно использовать только регистровые пары *B* и *D,* причем вторым из операндов всегда является аккумулятор *A.* Регистровая пара *H* имеет более универсальное применение – используется в командах передачи данных, командах арифметических и логических операций для косвенно-регистровой адресации операндов, расположенных в памяти и обозначаемых в командах символом M.

Неявная и стековая адресации данных. Описанные выше четыре метода адресации операндов требуют указания в командах операнда или его адреса в явном виде. Неявная и стековая адресация не требуют этого и поэтому не входят в перечень режимов адресации данных – наличие в МП неявной и стековой адресации операндов подразумевается по умолчанию.

Примерами команд с неявной адресацией могут служить команды циклического сдвига (в неявном виде подразумевается, что операндом является

содержимое аккумулятора). Это же относится и к таким командам, как CMA, CMC, STC и др.

Стековая адресация означает, что содержимое указателя стека *SP* является адресом данных, причем этот адрес в неявном виде подразумевается в кодах операций однобайтовых команд. Оперируют эти команды с двумя байтами данных (словами). В командах *PUSH PSW* и *PUSH rp* источником операндов являются регистровые пары *A* и *F*, *B* и *C*, *D* и *E*, *H* и *L*, а получателем – ячейки памяти *M*(*SP* - 1) и *M*(*SP* - 2). Декремент указателя стека *SP* при выполнении этих команд производится автоматически. В командах *POP PSW* и *POP rp* источником операндов являются ячейки памяти *M*(*SP*) и *M*(*SP* + 1), а получателем – пары регистров *A* и *F*, *B* и *C*, *D* и *E*, *H* и *L*. При выполнении этих команд инкремент указателя стека *SP* производится автоматически с установкой окончательного его значения *SP* + 2.

Достоинство стековой адресации: команды однобайтовые с выполнением операции пересылки двух 8-разрядных операндов. Данные команды часто используются для временного сохранения содержимого регистровых пар в памяти при нехватке регистров общего назначения для оперативного хранения операндов в процессе решения задачи. Для инициализации стека используется команда LXI SP, *d16*. Первой операцией всегда должно быть включение слова в стек и только затем его извлечение. Необходимый объем памяти стека определяется программистом, исходя из решаемых задач.

Прямая адресация переходов. Если адрес перехода содержится в самой команде переходов, то такая адресация называется *прямой.* В командах переходов JMP *addr* и *Jcond addr* используется прямая адресация – адрес перехода содержится во втором и третьем байтах команд. При прямой адресации в программах адреса переходов указываются метками.

Косвенно-регистровая адресация переходов. При такой адресации в коде операции команды передачи управления указывается регистровая пара, содержимое которой загружается в программный счетчик *PC* с потерей предыдущего его значения. Имеется только одна команда с косвенно-

регистровой адресацией переходов PCHL, загружающая содержимое *гр H* в программный счетчик *PC*. Команду PCHL удобно использовать для организации системы переходов или вызова подпрограмм по фиксированным адресам, записанным предварительно в таблицу, хранящуюся в памяти, или при вычислении адресов переходов по какому-либо алгоритму.

ФУНКЦИОНИРОВАНИЕ КР580ВМ80

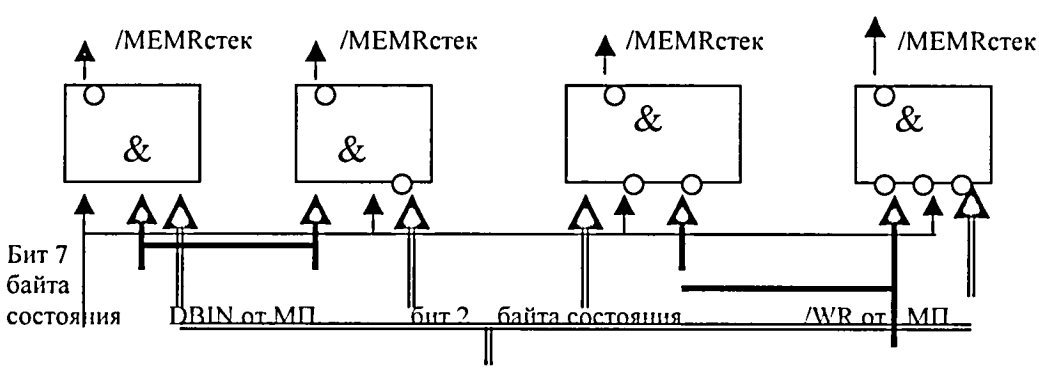
Команда МП является неким строительным кубиком при построении программ. В свою очередь команды МП выполняет по машинным циклам. Машинный цикл – это процесс обмена 1 байтом информации между МП и внешней подсистемой (памятью или портами ввода/вывода). Команды МП КР580ВМ80 могут содержать от одного до пяти машинных циклов (т.е. есть достаточно быстрые и медленные команды). В свою очередь машинные циклы делятся на машинные такты, а машинным тактом называется период синхросигнала Ф1. В машинном цикле может быть от трех до пяти машинных тактов.

В такте Т1 МП выдает на ША содержимое программного счетчика – РС (в программном счетчике адрес ячейки памяти в которой находится первый байт команды). Также в первом такте на ШД выдается байт состояния, который содержит служебную информацию о типе машинного цикла. В такте Т2 МП проверяет уровень сигнала READY и если он высокий (“1”), то в такте Т3 по ШД МП планирует считать данные с ШД. Для этого байт с ШД пропадает, а ШД подготавливается к вводу. Ввод байта производится в третьем такте, а именно спад DBIN защелкивает данные с ШД в МП (в регистр команд). После этого МП анализирует принятую команду. Это происходит в такте Т4 машинного цикла. Следует заметить, что первые три машинных такта всех машинных циклов унифицированы и представляют фазу обмена информацией. А такты Т4 и Т5 могут быть или не быть в зависимости от конкретного машинного цикла.

Поясним, что за информация располагается в байте состояния. Как мы

знаем у МП только два сигнала управления – чтение DBIN и запись /WR. Однако адреса портов ввода/вывода и ячеек памяти могут быть одинаковыми и непонятно, как МП в одном случае запишет байт в ячейку памяти с адресом 0, а в другом случае выведет байт в порт вывода с адресом 0. Для уточнения того, что производится в текущем машинном цикле, необходима дополнительная информация. И эта дополнительная информация выводится на ШД в первом машинном такте всех команд, а в начале второго машинного такта эта информация пропадает с ШД. Дополнительная информация называется байтом состояния, который показывает, какой именно тип машинного цикла выполняется. Всего у МП имеется 11 типов машинных циклов. Байт состояния находится на ШД только в первом машинном такте каждого машинного цикла, поэтому для того, что информацию байта состояния использовать во всем машинном цикле его нужно запомнить (защелкнуть в регистре-защелке).

Рассмотрим организацию МП-системы, когда по 64Кбайта отводится отдельно для памяти и стека. Для этого нельзя использовать микросхему КР580ВК28, т.к. машинные циклы чтения из памяти и чтения из стека приводят и одному сигналу управления - /MEMR. Аналогично машинные циклы записи в память и записи в стек приводят к одному машинному циклу - /MEMW. Для того, чтобы различить адресные пространства стека и памяти необходимо вместо двух сигналов управления /MEMR и /MEMW сформировать 4 сигнала: $/MEMR_{стек}$ $/MEMW_{стек}$ $/MEMW_{память}$ $/MEMR_{память}$. На рис. показана схема формирования данных сигналов.



У МП два адресных пространства – памяти и ввода-вывода. При этом адресное пространство памяти существенно больше (64 Кбайт против 256). Также команд МП работающих с памятью существенно больше чем команд работающих с портами ввода-вывода. Поэтому иногда реализуется ввод-вывод отображенный на память, который заключается в том, что часть адресного пространства память отводится непосредственно под память, а другая часть адресного пространства отводится под порты ввода-вывода. При этом получается, что команды МП работы с портами ввода-вывода и с память становятся одинаковыми.

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ АССЕМБЛЕР

МП может только выполнять команды, которые представляют последовательности нулей и единиц. Для человека такой способ программирования МП неудобен и для устранения этого неудобства используется язык ассемблера. Язык ассемблера обеспечивает соответствие каждому двоичному коду мнемонического (или символического) обозначения. А преобразование программы из мнемонических обозначений в двоичный код производится специальной программой которая называется компилятором. В строке программы на языке ассемблера можно выделить 4 поля. Рассмотрим их последовательно.

1. Поле метки. Это необязательное символическое имя, которое ассоциируется с 16-битным адресом той ячейки памяти в которую будет помещен первый байт очередной команды. Метки используются в качестве операндов команд передачи управления и они освобождают программиста от необходимости оперировать абсолютными адресами памяти. Поскольку метки ассоциируются с адресами памяти их нельзя делать одинаковыми у разных строк. При программировании можно, естественно, отмечать любую строку

программы, но из практических соображений следует вводить метки только для тех команд, которым следует передавать управление.

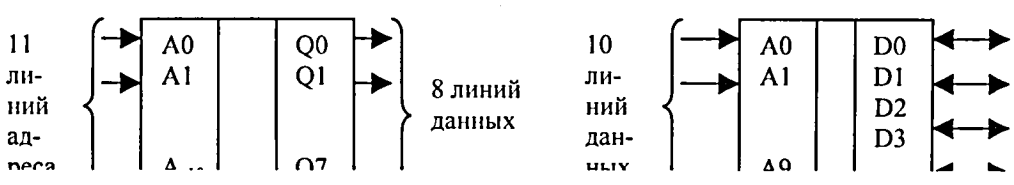
2. Поле кода. В поле кода содержится символическое описание выполняемой команды, которое заменяет числовое значение кода операции. Большинство мнемонических обозначений представляет аббревиатуру предложений, характеризующих основные функции команд. Например, MOV - от слова move (передать, переслать). Обычно длина поля кода не превышает четырех позиций, а между ним и соседним полем операнда справа необходим минимум один пробел. Мнемоники кодов операций являются ключевыми словами ассемблера и, если содержимое этого поля не входит в множество допустимых мнемоник, то ассемблер выдает сообщение о недействительной команде.

3. Поле операндов. В этом поле определяются данные являющиеся операндами команды. Следовательно, содержимое поля операнда должно интерпретироваться в соответствии с функцией команды. В качестве операндов могут фигурировать адреса памяти, внутренние регистры МП, адреса портов ввода/вывода, числовые и символьные данные. Некоторые команды, оперирующие с определенными внутренними регистрами, имеют пустое поле одно из операндов.

4. Поле комментария. Это поле начинается с разделителя – точка с запятой и полностью игнорируется компилятором, поэтому в это поле можно помещать любой текст.

ПРОГРАММНО-АППАРАТНАЯ ОРГАНИЗАЦИЯ ОБМЕНА ДАННЫМИ В МИКРОПРОЦЕССОРНОЙ СИСТЕМЕ

Рассмотрим типичные микросхемы ПЗУ и ОЗУ.В качестве микросхемы ПЗУ (постоянное запоминающее устройство) рассмотрим К573РФ5 которая показана на рис.



К573РФ5. К541РУ2.

/CE - вход выбора кристалла (при “0” на этом входе микросхема является подключенной к системе, а при при “1” микросхема отключена из МП-системы - микросхемы “как будто нет”). Дело в том, что в микропроцессорной системе много компонентов, но обмен информацией производится между микропроцессором и одной из микросхем. .И с помощью входного сигнала /CE (такой вход есть у всех микросхем причем у некоторых он называется /CS) можно одну из микросхем подключить, а другие отключить из микропроцессорной системы.

/OE – вход разрешения чтения (при “0” на этом входе информация из ячейки с адресом указанным на линиях A_{0-11} поступает на выходные линии Q_{0-7} , при “1” на этом входе выходы Q_{0-7} находятся в высокоимпедансном состоянии).

Количество адресных входов (11) определяет количество ячеек памяти - 2^{11} в микросхеме ПЗУ, а количество выходных линий данных Q_{0-7} определяет информационную емкость каждой ячейки.

На рис. приведена микросхема ОЗУ К541РУ2. В этой микросхеме 2^{10} ячеек (что определяется количеством адресных линий), а информационная емкость каждой ячейки 4 бита. Таким образом, данная микросхема может хранить 512 байт. Рассмотрим входы и выходы этой микросхемы.

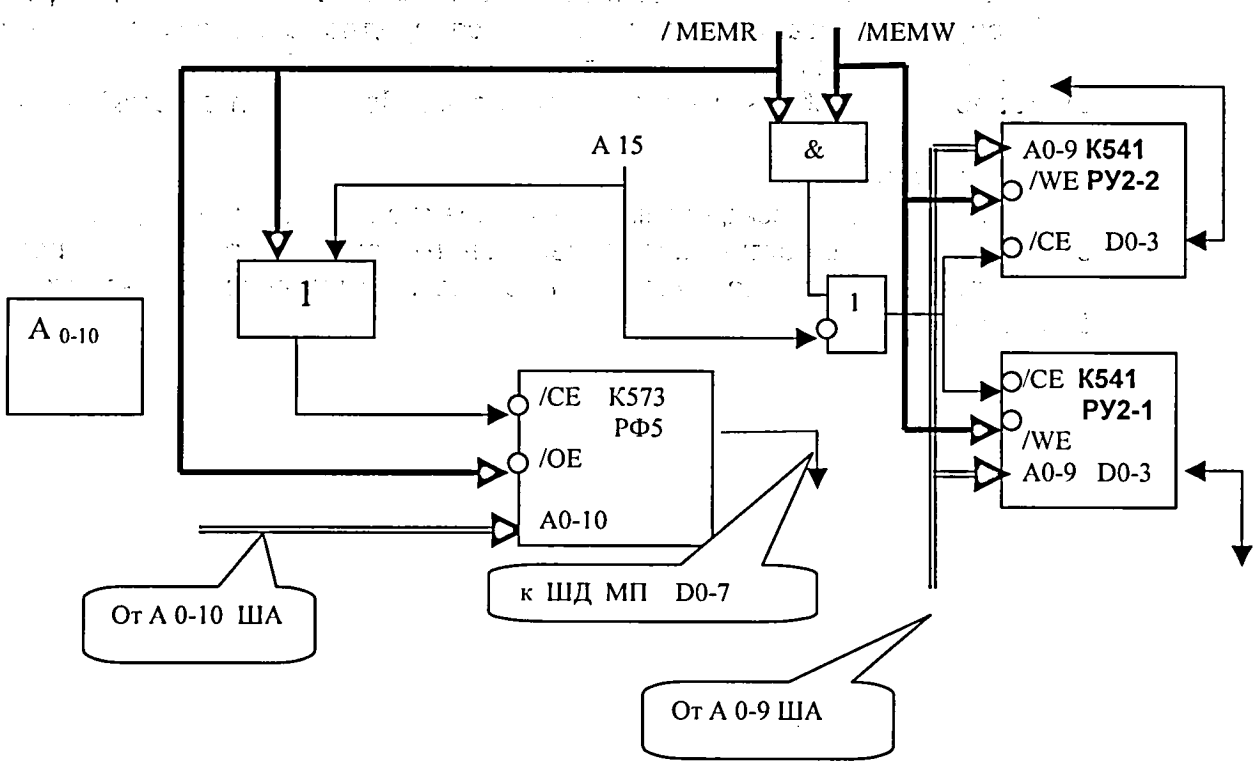
/CE - вход выбора кристалла (при “0” на этом входе микросхема является подключенной к системе, а при при “1” на этом входе микросхема отключена от системы).

/WE – сигнал разрешения записи (при /WE =”0” и /CE =”0” данные

записываются в микросхему, а при /WE = “1” и /CE =”0” данные читаются из микросхемы в микропроцессор).

В качестве примера рассмотрим подключение к МП системе 2Кбайта ПЗУ (начиная с адреса 0000Н) и 1 Кбайта ОЗУ (начиная с адреса 8000Н). Т.к. в этом случае реальная память занимает только часть адресного пространства, то схему дешифрации (ОЗУ и ПЗУ) можно выбирать исходя из ее максимальной простоты. По условию ПЗУ от ОЗУ отлисается страшим битом ША - А 15 (у ПЗУ - 0 , а у ОЗУ – 1). На следующем рисунке (рис.) показано подключение требуемой памяти к микропроцессорной системе. Если на А15 ША формируется логический “0” , то это обеспечивает низкий уровень на входе /CE К573РФ5. Таким образом, когда МП обращается к ячейкам памяти с адресами в интервале 0000Н – 7FFFH, то автоматически происходит обращение к ячейкам памяти из ПЗУ. Адресные разряды А0-А10 позволяют выбрать одну из ячеек внутри микросхемы ПЗУ.

Поясним теперь как подключается ОЗУ. Для формирования сигнала /CE К541РУ2 используются адресные линия А15 , а именно когда на этом проводе логическая единица - 1, то на входе /CE будет низкий уровень. При другом уровне А 15 на входе /CE микросхем К541РУ2 будет высокий уровень.



ОСНОВЫ ПОСТРОЕНИЯ И ПРИНЦИП РАБОТЫ СОВРЕМЕННЫХ МПС. ЧИПСЕТ

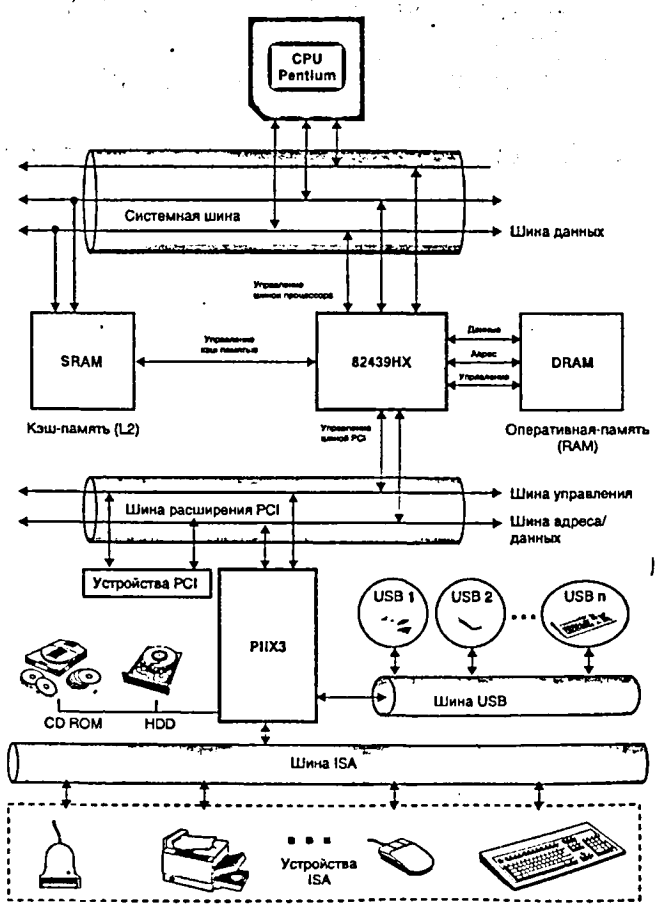
Chipset — это набор микросхем, установленных на материнской плате для обеспечения обмена данными между CPU, оперативной памятью и различными устройствами, подключенными к шинам (ISA, PCI, USB). Кроме того, Chipset управляет потоком данных к(от) жестким(х) дискам и другим устройствам, подключенным к каналам IDE. Именно Chipset определяет функциональные возможности материнской платы: тип устанавливаемых процессоров, тип и объем оперативной и кэш-памяти второго уровня, тактовую частоту системной шины, поддерживаемые шины и др.

В середине 1990-х гг. Chipset 430FX (Triton I) принес известность корпорации Intel как производителя лучших Chipset для CPU класса Pentium. Основным конкурентом Intel в области производства Chipset является тайваньская корпорация VIA Technologies Inc. Другие известные производители Chipset для недорогих систем — компании ALI (подразделения корпорации Aser Labs) и SiS. Однако самый главный конкурент Intel в производстве процессоров — фирма AMD — также производит Chipset для своих CPU.

Все Chipset имеют свои достоинства и недостатки. Однозначно классифицировать Chipset нельзя. Хорошие Chipset, прежде всего, отличаются

возможностью более гибкого конфигурирования через CMOS Setup. Это может стать для пользователей средством оптимизации работы системы.

Принцип функционирования чипсета в составе МПС на примере Chipset Triton 430HX приведен на рис.8.2.



Классификация ЗУ

Чтобы CPU мог выполнять программы, они должны быть загружены в оперативную *рабочую* память, т. е. в память, доступную для программ пользователя. К данным, находящимся в оперативной памяти *(Random Access Memory, RAM* – память с произвольным доступом), CPU имеет непосредственный доступ, а к периферийной, или внешней памяти (гибким и

жестким дискам) – через буфер, являющийся также разновидностью оперативной памяти, недоступной пользователю. Только после того как программа будет загружена в RAM с внешнего носителя данных, возможна дальнейшая ее работа.

Запоминающим элементом динамической памяти является полупроводниковый конденсатор, который может находиться в заряженном или разряженном состоянии. Если конденсатор заряжен, то в ячейку записана логическая 1, если разряжен – логический 0. В идеальном конденсаторе заряд может сохраняться неограниченное время. В реальном конденсаторе существует ток утечки, поэтому информация, записанная в динамическую память, со временем будет утрачена, т. к. конденсаторы запоминающих элементов полностью разрядятся.

Чтобы пояснить этот процесс, представим элемент памяти как ведро с водой, которое может быть либо пустым (состояние 0), либо полным (состояние 1). Однако в этом ведре имеются такие маленькие дырки, что вода (информация) вытекала бы по капле, если бы "водоносу" (специальной логической схеме) не было поручено компенсировать убыток воды (данных) так, чтобы уровень ее оставался неизменным. Этот процесс называется регенерацией памяти *(Refresh)*. Деятельность "водоноса" имеет огромное значение, поэтому ему нельзя мешать. Это означает, что CPU имеет доступ к данным, находящимся в RAM, только в течение циклов, свободных от регенерации. Единственным способом регенерации хранимой в памяти информации является выполнение операции чтения/записи данных. Если информация заносится в динамическую память, а затем в течение нескольких миллисекунд остается невостребованной, она будет утрачена, т. к. конденсаторы запоминающих элементов полностью разрядятся.

Регенерация памяти происходит при выполнении каждой операции чтения или записи. Однако нет гарантии, что при выполнении любой программы произойдет обращение ко всем ячейкам памяти, поэтому имеется специальная схема, которая через определенные промежутки времени (например, каждые 2

мс) будет осуществлять доступ (для считывания) ко всем строкам памяти. В эти моменты CPU находится в состоянии ожидания. За один цикл схема регенерирует все строки динамической памяти

Повышение скорости обмена данными

Для повышения скорости обмена данными между CPU и микросхемами памяти разработаны специальные режимы работы памяти и технологии

- Пакетный режим
- Чередование памяти
- Разбиение памяти на страницы
- Кэширование памяти

Пакетный режим

Как уже отмечалось, CPU запрашивает данные из памяти не побайтно, а в виде пакетов, состоящих из 32 или 64 бит. Такой порядок обмена данными с памятью был впервые реализован на PC с CPU 80486 и назван пакетным режимом *(Burst Mode)*

В этом режиме кроме одного слова CPU считывает еще три, расположенные рядом

Чередование памяти

Метод управления памятью с чередованием адресов *(Interleaving mode)* основан на том, что логически связанные байты чаще всего располагаются в памяти друг за другом Как уже отмечалось, в микросхеме памяти осуществляется периодическая регенерация данных, в процессе которой микросхема не доступна для записи и чтения. Чтобы не было пауз в работе памяти, осуществляется ее чередование, т е. помещение следующих друг за другом ячеек памяти в различные банки, из которых CPU должен считывать данные попеременно. Пока в одном из чипов памяти происходит регенерация данных, CPU может считывать следующий байт из другого банка.

Организацию и управление чередованием памяти осуществляет контроллер памяти, который логически объединяет два банка в один и распределяет адресное пространство так, чтобы соседние адреса находились в

разных банках.

При использовании SIMM-модулей чередование памяти возможно лишь в том случае, когда в различные банки установлены SIMM-модули одинаковой емкости. В микросхемах SDRAM этот режим реализуется аппаратно не контроллером памяти, а на уровне микросхемы.

Разбиение памяти на страницы

Метод разбиения памяти на страницы *(Pacing mode)* основан на том факте (уже упоминавшемся при описании чередования памяти), что каждый поступающий в CPU байт расположен рядом с байтом, уже считанным из памяти и логически связанным с ним. Следовательно, не нужно повторять сигнал RAS, если адреса строк выбираемых ячеек памяти находятся в пределах одной страницы, т. е. адрес строки неизменен.

Обычно память делится на страницы размером 512 байт и более.

Кэширование памяти

(Кэширование памяти используется для ускорения доступа к данным, находящимся в RAM. Это достигается за счет применения промежуточной (буферной между CPU и RAM) быстродействующей памяти небольшой емкости (256 Кбайт – 2 Мбайт) (кэш-памяти. Такая память работает на частоте CPU, поэтому при обращении к ней не требуются циклы ожидания.

ДИНАМИЧЕСКАЯ ПАМЯТЬ

Ниже рассмотрены характеристики различных типов элементов динамической памяти, начиная от тех, которые применялись в первых PC и до самых современных.

DRAM

Буква "D" в аббревиатуре "DRAM" означает "динамическая" (Dynamic), т.е. для сохранения данных, записанных в микросхеме памяти, необходима их периодическая регенерация. Все микросхемы DRAM имеют матричную организацию, причем каждый элемент матрицы (миниатюрный конденсатор) хранит один бит данных и адресуется с помощью следующих сигналов: RAS, адрес строки, CAS и адрес столбца.

В процессе совершенствования технологии изготовления DRAM были разработаны различные типы памяти PM, FPM, EDO и SDRAM

FPM DRAM

Память типа DRAM, реализующая быстрый страничный режим,

называется FPM DRAM *(Fast Page Mode DRAM)*

Память этого типа появилась в последних моделях PC с CPU 80486 и получила широкое распространение. Время доступа процессора к памяти при использовании микросхем FPM DRAM (60 нс) сократилось на 40% по сравнению с временем доступа к обычным DRAM. Время рабочего цикла последних чипов составило 35 нс. Тем не менее, микросхемам FPM DRAM не удавалось "угнаться" за процессором, если частота системной шины превышала 28 МГц.

EDO DRAM

В компьютерах на базе CPU Pentium применяется память типа EDO DRAM *(Extended Data Output).*

Структурная схема EDO DRAM похожа на схему FPM DRAM. Отличие состоит в том, что для FPM DRAM линии ввода/вывода данных отключались от системной шины, как только начиналось задание адреса следующего бита, а в режиме EDO линии остаются подключенными до окончания ввода нового адреса и, соответственно, начала вывода следующего бита. Вместо сигнала CAS для указания конца операции чтения используется сигнал OE *(Output Enable).* Таким образом, память EDO позволяет одновременно считывать данные и задавать адрес следующих данных, что, в свою очередь, сокращает длительность рабочего цикла

Модули памяти EDO работают на 10–15% быстрее, чем FPM DRAM. Они работают без задержки с системными шинами, работающими на тактовой частоте 50 МГц (1:20нс=50 МГц). Тем не менее, преимущество EDO перед FPM проявляется лишь при чтении данных – одновременное выполнение операций записи и адресации невозможно.

BEDO DRAM

Микросхемы BEDO DRAM *(Burst EDO)* – это разновидность EDO DRAM. В отличие от EDO в микросхему BEDO был добавлен специальный генератор номера столбца. После первого поступления на вход микросхемы адреса ячейки и сигналов RAS и CAS, для последующих четырех столбцов сигнал CAS

генерируется внутри микросхемы.

CDRAM, EDRAM

Микросхемы CDRAM *(Cache DRAM)* и EDRAM *(Enhanced DRAM)* содержат немного ячеек быстрой памяти SRAM, имеющих время доступа 10–15 нс. Например, на од-Ном кристалле могут находиться 4 Мбайт DRAM и 16 Кбайт SRAM. Интегрированные элементы SRAM в данном случае можно рассматривать как встроенную кэш-память.

SDRAM

До 1997 г. производились только асинхронные микросхемы DRAM, т. е. такие, работа которых не была синхронизирована с частотой работы системной шины.

Асинхронные элементы имеют только информационные входы и срабатывают непосредственно после изменения сигнала на входах Сигнал на выходе появляется через некоторое время Оно не регламентируется и может изменяться в зависимости от температуры и от старения полупроводниковых элементов Основным недостатком асинхронных элементов является их низкая помехоустойчивость, проявляющаяся в сбоях при работе PC

Для срабатывания синхронных элементов смены сигналов на входах недостаточно Необходим дополнительный тактирующий сигнал, который подается на соответствующий вход В качестве такого сигнала выбран тактовый сигнал системной шины Этот сигнал задает частоту смены информации в определенные моменты времени. В эти же моменты обновляется информация на выходах элементов Таким образом, процессы записи и считывания информации в память строго привязаны к тактам CPU или шины

Все рассмотренные выше элементы памяти (FPM, EDO, BEDO DRAM) работают асинхронно с тактовой частотой системной шины. Поэтому для передачи данных из памяти в CPU необходимо подтверждение связи между CPU и контроллером памяти

Процесс чтения данных организован следующим образом. CPU сообщает контроллеру памяти с помощью сигнала ADS, что ему необходимы данные,

располагающиеся по определенному адресу. Затем осуществляется цикл чтения данных из DRAM. Когда данные прочитаны и находятся на выходных линиях DRAM, контроллер памяти посылает процессору сигнал BRDY и только тогда данные поступают на CPU. Если данные еще не готовы, то CPU вынужден осуществлять холостые циклы *(Wait state)* в ожидании данных. Память типа BEDO DRAM может поставлять данные CPU и без цикла ожидания, но только при тактовой частоте системной шины до 66 МГц.

В 1997 г. для синхронизации работы памяти и системной шины использовалась микросхема синхронной динамической памяти SDRAM *(Sychronous DRAM)*. Метод доступа к строкам и столбцам данных в микросхемах SDRAM и стандартной DRAM реализован одинаково. Отличие заключается в следующем: все операции в микросхемах SDRAM синхронизированы с тактовой частотой CPU, т. е. память и CPU работают синхронно без циклов ожидания (рис.7.9).

Первоначально SDRAM была предложена в качестве более дешевой по стоимости альтернативы дорогой видеопамяти VRAM (Video RAM), используемой в видеоадаптерах. За счет исключения циклов ожидания сократилось время выполнения команд и передачи данных. Современные микросхемы SDRAM могут работать на тактовых частотах CPU 66, 75, 83, 100, 125 и 133 МГц.

Кроме того, для сокращения времени выборки данных в микросхеме SDRAM предусмотрено чередование адресов, а также пакетный режим; используется трехступенчатая конвейерная адресация, которая позволяет осуществить доступ к запрошенным данным до завершения обработки предыдущих.

Все это позволило сократить время рабочего цикла микросхемы до 8–10 нс (1:10 нс=100 МГц) и повысить скорость передачи данных до 800 Мбайт/с при тактовой частоте системной шины 100 МГц.

PC100/133 SDRAM

При тактовой частоте системной шины 100 МГц многие микросхемы

SDRAM работали неустойчиво, поэтому для такой системной шины корпорация Intel разработала спецификацию микросхем памяти, получившую название PC100.

С увеличением тактовой частоты системной шины до 133 МГц появились микросхемы SDRAM, поддерживающие данную тактовую частоту. Они получили название PC133 SDRAM.

ESDRAM

Микросхемы ESDRAM являются расширением микросхем SDRAM. В микросхеме интегрированы элементы SDRAM, позволяющие работать на частоте системной шины 66, 100 и 166 МГц. Время рабочего цикла сократилось до 8 не. Микросхемы полностью совместимы с PC100 SDRAM.

DDR SDRAM (SDRAM II)

Следующим этапом в развитии памяти SDRAM явилась память DDR SDRAM, которую также называют SDRAM 11.

В отличие от "обычной" памяти SDRAM (или SDRAM I), данные передаются по восходящему и ниспадающему фронтам синхросигнала, в результате чего пропускная способность шины данных увеличилась вдвое. Так, в случае 64-разрядной шины данных за один такт передается 16 байт данных (64/8*2).

DDR II

Дальнейшим развитием технологии DDR SDRAM является технология DDR И-В отличие от DDR SDRAM, в DDR II за один такт по каждой линии будет передаваться не 2, а четыре бита информации (при 64-разрядной шине – 32 байта), что вдвое увеличит пропускную способность шины памяти. Естественно, что данный подход требует более совершенной системы синхронизации. В основе микросхемы памяти остается 100 МГц ядро SDRAM.

Максимальная пропускная способность памяти DDR должна составить 3200, 6400 Мбайт/с.

Direct RDRAM

Небольшая американская фирма Rambus начала разработку нового типа

памяти в 1992 г. Уже спустя 3 года она анонсировала принципиально новую технологию – Rambus. Новая технология подразумевала наличие усовершенствованных микросхем памяти Base RDRAM и так называемого Rambus-канала, включающего высокоскоростную 8-разрядную шину (700 МГц) и специальный контроллер памяти. Пропускная способность памяти составила 600 Мбайт/с, что превосходит аналогичный показатель некоторых современных модулей памяти.

В 1995 г. фирма заключила соглашение с корпорацией Intel, которая поддержала перспективные разработки фирмы Rambus и приступила к разработке и выпуску Chipset, поддерживающих технологию Rambus.

В микросхеме Direct DRAM сохранились старые принципы записи и считывания Данных в ячейки матрицы, изменилась лишь организация банков выборки данных из памяти.

Технология Direct Rambus ориентирована на 16-разрядную шину данных, функционирующую на частотах до 400 МГц. Использование технологии DDR позволило увеличить скорость передачи данных до 1.6 Гбайт/с (16 бит*(400*2)/8) на один канал (или до 6.4 Гбайт/с на 4 каналах). Данные и служебные биты передаются по каждому фронту тактового сигнала с частотой 800 МГц (1.25 нс), что соответствует скорости передачи 800 Мбит/с на линию. Таким образом, пропускная способность шины составляет 1.6 Гбайт/с. Передача данных осуществляется только между контроллером и микросхемами памяти, обмен только между микросхемами невозможен.

Вся информация передается по Rambus-каналу пакетами (пакеты строк, столбцов и данных). Каждый пакет передается за четыре такта (10 нс).

SLDRAM

В январе 1997 г. несколько фирм объединились, чтобы создать более дешевую, чем Direct RDRAM, быстродействующую память, ориентированную, в первую очередь, на установку в PC стоимостью до 1000 USD. Память нового типа получила название SLDRAM *(SyncLmc DRAM).* Стандарт SLDRAM изначально планировался как открытый, что по замыслу членов консорциума

SLDRAM Inc. могло способствовать быстрому развитию стандарта и обеспечило бы ему долгую "жизнь".

Технология SLDRAM представляет собой следующий шаг в развитии DRAM от FPM DRAM до DDR DRAM. В микросхемах SLDRAM также используется классическое ЯДРО DRAM. Стандарт SLDRAM имеет преимущества стандартов SDRAM и DDR SDRAM. Кроме того, стандарт SLDRAM предусматривает протокол пакетной передачи адреса.

По заявлению разработчиков микросхемы типа SLDRAM совместимы с предыдущими элементами DRAM. В первом поколении SLDRAM (как и в Direct RDRAM) используется 16-разрядная шина данных, работающая на тактовой частоте 400 МГц. Подобно DDR SDRAM и RDRAM, передача данных осуществляется по фронту и спаду тактового сигнала. Таким образом, пропускная способность SLDRAM составляет 16 бит*400 МГц*2=1.6 Гбайт/с. В дальнейшем разработчики планируют увеличить пропускную способность SLDRAM за счет повышения тактовой частоты системной шины.

АРХИТЕКТУРА, НАЗНАЧЕНИЕ И ВЗАИМОДЕЙСТВИЕ МИКРОПРОЦЕССОРА И ПЕРИФЕРИЙНЫХ БИС. ПРОГРАММИРУЕМЫЙ ИНТЕРВАЛЬНЫЙ ТАЙМЕР КР580ВИ53

Микросхема КР580ВИ53 (рис.1) решает одну из наиболее общих проблем любой микропроцессорной системы – генерацию точных временных интервалов. Микросхема выполнена по n-МОП – технологии в 24-выводном корпусе и имеет единственный источник питания +5В.

В состав микросхемы входят три 16-разрядных вычитающих счетчика с частотой счета по входу CLK до 2 МГц. Каждый счетчик может работать в одном из шести режимов независимо от других. Все счетчики программно доступны для записи и чтения с помощью слов данных DW и могут работать как в двоичном коде, так и в двоично-десятичном. Управление режимами выполняется с помощью управляющих слов CW (рис.2), которые кроме режима определяют код счета и формат обмена данными с микропроцессором при

операциях со счетчиками : только старшим байтом, только младшим байтом или всем словом.

Связь таймера с микропроцессорной системой осуществляется через двунаправленную 8-разрядную шину данных $D_{0\text{-}7}$ под управлением 5-ти управляющих сигналов: A_0, A_1, /CS, /RD и /WR в соответствии с табл.1 . Здесь и в дальнейшем обозначение в виде наклонной черты представляет логическое отрицание. Например, /RD показывает, что активное действие производится при низком уровне (уровень “0”) на этом входе.

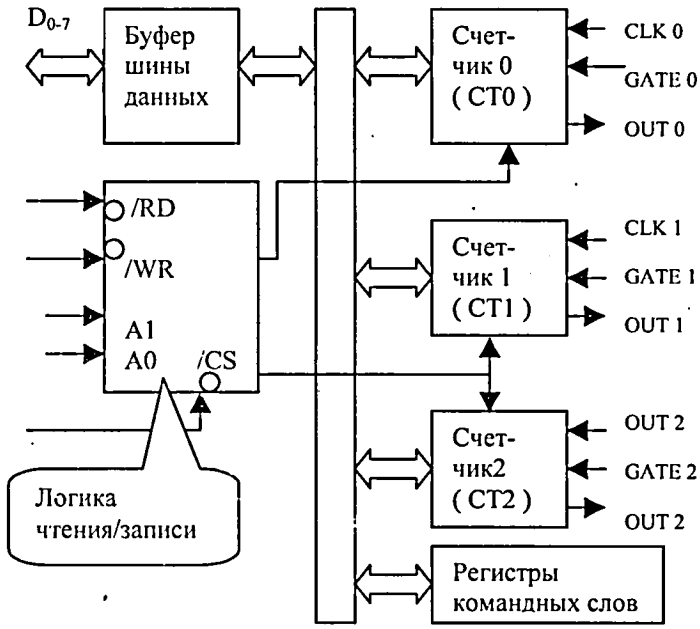


Рис.1, *а*. Структурная схема КР580ВИ53.

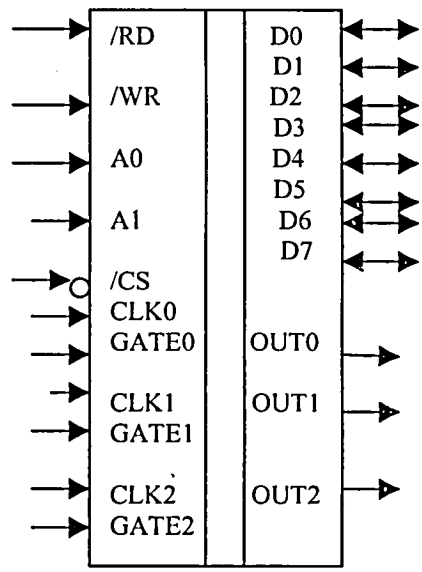


Рис.1, *б*. Условное обозначение

При двухбайтовом формате данных операция со счетчиками выполняется дважды: сначала записывается или считывается младший байт, а затем старший. Обслуживание счетчиков выполняется параллельно и независимо друг от друга. При подаче питания их состояния и режим работы оказываются неопределенными. Поэтому перед началом работы каждый счетчик должен быть инициализирован индивидуально посылкой соответствующего слова управления CW. Каждое слово управления CW (за исключением операции защелкивания) сопровождается 1-2 байтами слова данных DW начального состояния выбранного счетчика. Выполнение операции счета начинается после