Базовые принципы организации вычислительных систем

1.1. Основные понятия и определения в области организации вычислительных систем

Рассмотрим базовые концепции, которые лежат в основе любой вычислительной системы, от простейшего микроконтроллера до сложного компьютера – базовая терминология вычислительной техники, принципы организации микропроцессорных систем, структура связей, режимы работы и основные типы вычислительной систем.

Введем несколько основных определений.

Электронная система *– в данном случае это любой электронный узел, блок, прибор или комплекс, производящий обработку информации.*

Задача *– это набор функций, выполнение которых требуется от электронной системы.*

Быстродействие *– это показатель скорости выполнения электронной системой ее функций.*

Гибкость *– это способность системы подстраиваться под различные задачи.*

Избыточность *– это показатель степени соответствия возможностей системы решаемой данной системой задаче.*

Интерфейс *– соглашение об обмене информацией, правила обмена информацией, подразумевающие электрическую, логическую и конструктивную совместимость устройств, участвующих в обмене. Другое название –* ***сопряжение****.*

Вычислительная система (микропроцессорная система (МПС)) *может рассматриваться как частный случай электронной системы, предназначенной для обработки входных сигналов и выдачи выходных сигналов* (рис. 1.1).

В качестве входных и выходных сигналов использоваться аналоговые сигналы, одиночные цифровые сигналы, цифровые коды, последовательности цифровых кодов. Внутри системы производиться хранение, накопление сигналов (или информации).

Рисунок 1.1 – Электронная система

1.2. Программируемые системы и системы на «жесткой логике»

Система на «жесткой логике» *– традиционная вычислительная система, особенность которой состоит в том, что алгоритмы обработки и хранения информации в ней жестко связаны со схемотехникой системы. Любая система на "жесткой логике" обязательно представляет собой специализированную систему, настроенную исключительно на одну задачу или (реже) на несколько близких, заранее известных задач.*

Изменение алгоритмов в системе на «жесткой логике» возможно только путем изменения структуры системы, замены электронных узлов, входящих в систему, и/или связей между ними. Например, если нам нужна дополнительная операция суммирования, то необходимо добавить в структуру системы лишний сумматор. Или если нужна дополнительная функция хранения кода в течение одного такта, то мы должны добавить в структуру еще один регистр. Естественно, это практически невозможно сделать в процессе эксплуатации, обязательно нужен новый производственный цикл проектирования, изготовления, отладки всей системы.

Преимущества системы на «жесткой логике»:

- *специализированная система* (в отличие от универсальной) *никогда не имеет аппаратурной избыточности*, то есть каждый ее элемент обязательно работает в полную силу (конечно, если эта система грамотно спроектирована).

- *специализированная система может обеспечить максимально высокое быстродействие*, так как скорость выполнения алгоритмов обработки информации определяется в ней только быстродействием отдельных логических элементов и выбранной схемой путей прохождения информации.

Основным большим недостатком цифровой системы на «жесткой логике» является *то, что для каждой новой задачи ее надо проектировать и изготавливать заново.* Это процесс длительный, дорогостоящий, требующий высокой квалификации исполнителей. А если решаемая задача вдруг изменяется, то вся аппаратура должна быть полностью заменена.

Программируемые (универсальные) системы *– универсальная вычислительная система, которая может адаптироваться под любую задачу, перестраиваться с одного алгоритма работы на другой без изменения аппаратуры. Задание нового алгоритма обеспечивается вводом в систему дополнительной управляющей информации,* ***программы работы системы*** (рис. 1.2). Тогда система становятся универсальной, или **программируемой**, с не жесткой, а гибкой логикой.

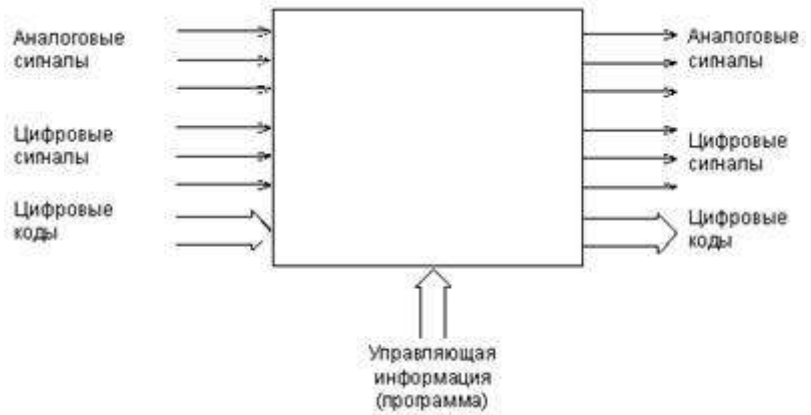

Рисунок 1.2 – Программируемая (универсальная) электронная система

Основным достоинством программируемой системы является *возможность решения различных задач, без изменения схемотехники системы.*

Основными недостатками программируемой системы является:

- *любая универсальность обязательно приводит к избыточности.* Чем проще решаемая задача, тем больше избыточность, и тем менее оправданной становится универсальность. Избыточность ведет к увеличению стоимости системы, снижению ее надежности, увеличению потребляемой мощности и т.д.

- *универсальность, как правило, приводит к существенному снижению быстродействия. Общее правило таково: чем больше универсальность, гибкость, тем меньше быстродействие.* Оптимизировать универсальную систему так, чтобы каждая новая задача решалась максимально быстро, попросту невозможно. Более того, для универсальных систем не существует таких задач (пусть даже и самых простых), которые бы они решали с максимально возможным быстродействием. За все приходится платить.

Таким образом, системы на "жесткой логике" хороши там, где решаемая задача не меняется длительное время, где требуется самое высокое быстродействие, где алгоритмы обработки информации предельно просты. А универсальные, программируемые системы хороши там, где часто меняются решаемые задачи, где высокое быстродействие не слишком важно, где алгоритмы обработки информации сложные.

1.3. Понятие о процессоре

Ядром любой вычислительной системы является процессор (от англ. processor). Синонимом на русском языке является слово "обработчик".

Процессор – *это тот узел, блок, который производит обработку информации внутри вычислительной системы.*

Процессор заменяет практически всю "жесткую логику", которая понадобилась бы в случае традиционной цифровой системы. Он выполняет:

- арифметические функции (сложение, умножение и т.д.),
- логические функции (сдвиг, сравнение, маскирование кодов и т.д.),
- временное хранение кодов (во внутренних регистрах),
- пересылку кодов между узлами микропроцессорной системы и многое другое.

Количество элементарных операций, выполняемых процессором, может достигать нескольких сотен.

Остальные узлы вычислительной системы выполняют вспомогательные функции:

- хранение информации (в том числе и управляющей информации, то есть программы),
- связь с внешними устройствами,
- связь с пользователем и т.д.

Надо учитывать, что все свои операции процессор выполняет ***последовательно****, то есть одну за другой, по очереди.* С одной стороны, *это несомненное достоинство, так как позволяет с помощью всего одного процессора выполнять любые, самые сложные*

алгоритмы обработки информации. Но, с другой стороны, *последовательное выполнение операций приводит к тому, что время выполнения алгоритма зависит от его сложности*.

Вычислительная система способна сделать все, но работает она не слишком быстро, ведь все информационные потоки приходится пропускать через один-единственный узел – микропроцессор (рис. 1.3). В традиционной цифровой системе можно легко организовать параллельную обработку всех потоков информации, за счет усложнения схемы обработки потоков.

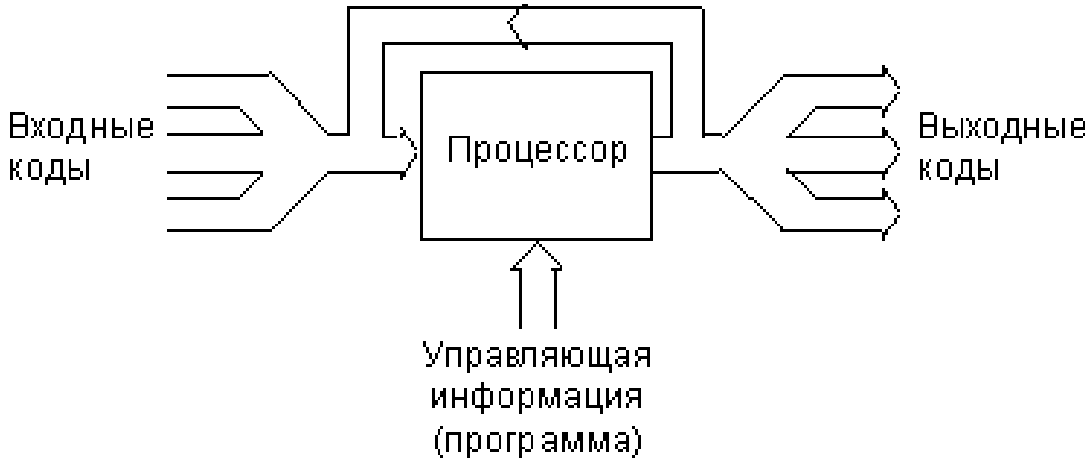

Рисунок 1.3 – Информационные потоки в микропроцессорной системе

Выполняемая в конкретный момент времени операция определяется управляющей информацией, программой.

Программа *представляет собой набор* ***команд (инструкций)****, составленный человеком* (*программистом*).

Команда (инструкция) *– цифровой код, расшифровав который, процессор узнает, что ему надо делать.* Каждая команда имеет свое время выполнения, поэтому время выполнения всей программы зависит не только от количества команд в программе, но и от того, какие именно команды используются.

Все команды, выполняемые процессором, образуют ***систему команд*** *процессора. Структура и объем системы команд процессора определяют его быстродействие, гибкость, удобство использования.*

Всего команд у процессора может быть от нескольких десятков до нескольких сотен. Система команд может быть рассчитана на узкий круг решаемых задач (у специализированных процессоров) или на максимально широкий круг задач (у универсальных процессоров). Коды команд могут иметь различное количество разрядов (занимать от одного до нескольких байт).

Для выполнения команд в структуру процессора входят:

- *внутренние регистры,*
- *арифметико-логическое устройство* (*АЛУ, ALU – Arithmetic Logic Unit*),
- *мультиплексоры, буферы, регистры и другие узлы.*

Работа всех узлов процессора синхронизируется общим внешним тактовым сигналом (*синхроимпульсом*).

То есть процессор представляет собой довольно сложное цифровое устройство (рис. 1.4).

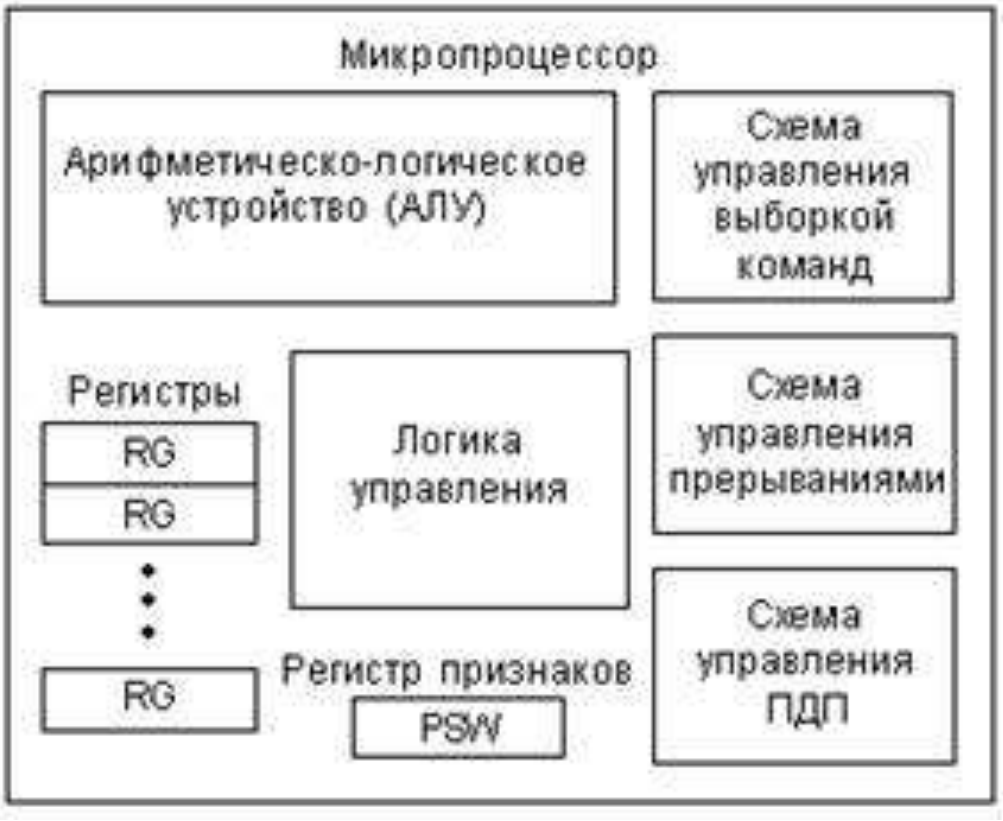

Рисунок 1.4 – Пример структуры простейшего процессора

Разработчик должен рассматривать процессор как "черный ящик", который в ответ на входные и управляющие коды производит ту или иную операцию и выдает выходные сигналы. Разработчику необходимо знать систему команд, режимы работы процессора, а также правила взаимодействия процессора – протоколы обмена информацией. О внутренней структуре процессора надо знать только то, что необходимо для выбора той или иной команды, того или иного режима работы.

1.4. Шинная структура связей

Для достижения максимальной универсальности и упрощения протоколов обмена информацией в вычислительных системах применяется так называемая шинная структура связей между отдельными устройствами, входящими в систему.

При классической структуре связей (*рис. 1.5*) *все сигналы и коды между устройствами передаются по отдельным линиям связи. Каждое устройство, входящее в систему, передает свои сигналы и коды независимо от других устройств.* При этом в системе получается очень много линий связи и разных протоколов обмена информацией.

При шинной структуре связей (*рис. 1.6*) *все сигналы между устройствами передаются по одним и тем же линиям связи, но в разное время (это называется мультиплексированной передачей). Причем передача по всем линиям связи может осуществляться в обоих направлениях (так называемая* ***двунаправленная передача****).* В результате количество линий связи существенно сокращается, а правила обмена (протоколы) упрощаются.

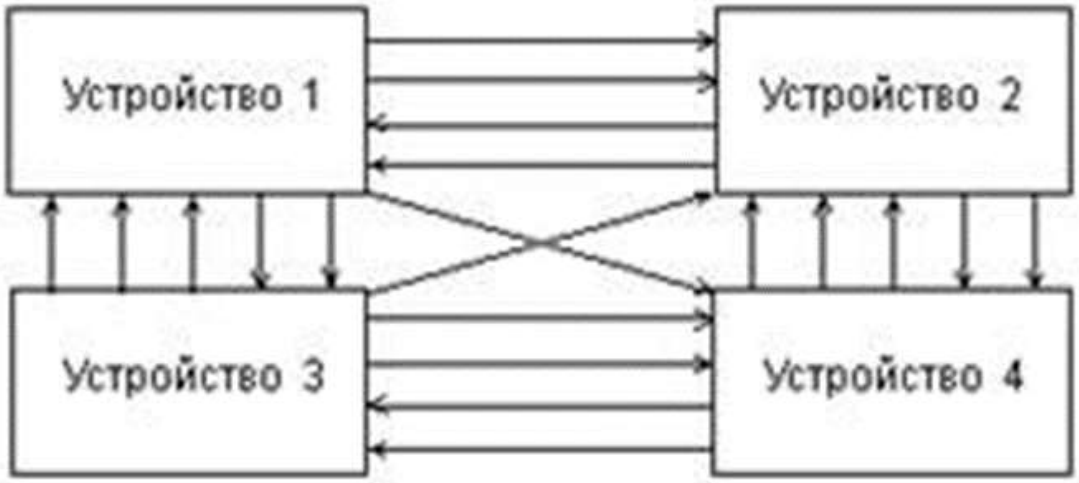

Рисунок 1.5 – Классическая структура связей

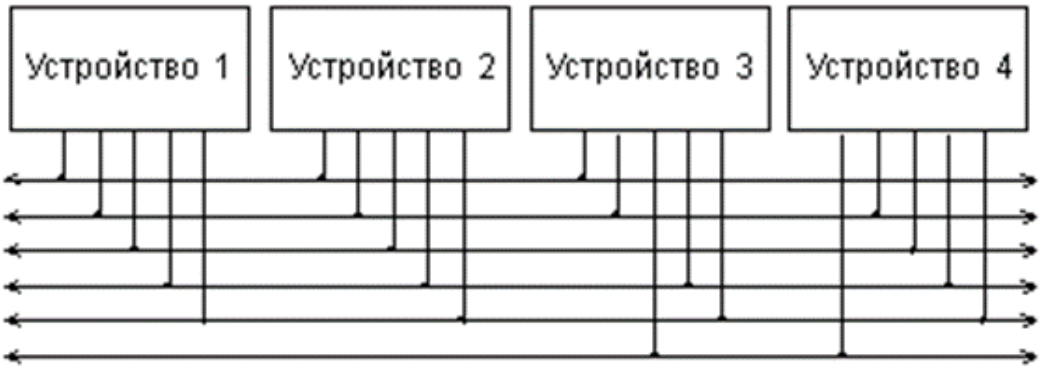

Рисунок 1.6 – Шинная структура связей

Группа линий связи, по которым передаются сигналы или коды как раз и называется ***шиной***(*англ. bus*).

Достоинством шинной структуры связей является

- *легкость осуществляется пересылка всех информационных потоков в нужном направлении*, например, их можно пропустить через один процессор, что очень важно для микропроцессорной системы.

- *все устройства, подключенные к шине, должны принимать и передавать информацию по одним и тем же правилам* (*протоколам обмена информацией по шине*). Соответственно, все узлы, отвечающие за обмен с шиной в этих устройствах, должны быть единообразны, унифицированы.

Недостатком шинной структуры связей является

- *вся информация передается по линиям связи последовательно во времени, по очереди, что снижает быстродействие системы по сравнению с классической структурой связей.*

- *все устройства подключаются к каждой линии связи параллельно. Поэтому любая неисправность любого устройства может вывести из строя всю систему*, если она портит линию связи. По этой же причине отладка системы с шинной структурой связей довольно сложна и обычно требует специального оборудования.

1.5. Структура вычислительной системы

Типичная структура вычислительной системы включает в себя три основных типа устройств (рис. 1.7):

- *процессор*;

- *память, включающую оперативную память* (ОЗУ, RAM – Random Access Memory) и постоянную память (ПЗУ, ROM – Read Only Memory), которая служит для хранения данных и программ;

- *устройства ввода/вывода* (УВВ, I/O – Input/Output Devices), служащие для связи микропроцессорной системы с внешними устройствами, для приема (ввода, чтения, Read) входных сигналов и выдачи (вывода, записи, Write) выходных сигналов.

*Все устройства вычислительной системы объединяются общей системной шиной (она же называется еще **системной магистралью** или **каналом**).*

Рисунок 1.7 – Структура вычислительной системы

Системная магистраль включает в себя *четыре основные шины нижнего уровня:*

- *шина адреса (Address Bus);*

- *шина данных (Data Bus);*

- *шина управления (Control Bus);*

- *шина питания (Power Bus).*

Шина адреса *служит для определения адреса (номера) устройства, с которым процессор обменивается информацией в данный момент.* Каждому устройству (кроме процессора), каждой ячейке памяти в микропроцессорной системе присваивается собственный адрес. Когда код какого-то адреса выставляется процессором на шине

адреса, устройство, которому этот адрес приписан, понимает, что ему предстоит обмен информацией. *Шина адреса может быть однонаправленной или двунаправленной.*

Шина данных *– это основная шина, которая используется для передачи информационных кодов между всеми устройствами микропроцессорной системы.* Обычно в пересылке информации участвует процессор, который передает код данных в какое-то устройство или в ячейку памяти или же принимает код данных из какого-то устройства или из ячейки памяти. Но возможна также и передача информации между устройствами без участия процессора. *Шина данных всегда двунаправленная.*

Шина управления *в отличие от шины адреса и шины данных состоит из отдельных управляющих сигналов, каждый из которых имеет свою функцию.* Некоторые сигналы служат для стробирования передаваемых или принимаемых данных (то есть определяют моменты времени, когда информационный код выставлен на шину данных). Другие управляющие сигналы могут использоваться для подтверждения приема данных, для сброса всех устройств в исходное состояние, для тактирования всех устройств и т.д. *Линии шины управления могут быть однонаправленными или двунаправленными.*

Шина питания *предназначена не для пересылки информационных сигналов, а для питания системы.* Каждому напряжению питания соответствует своя линия связи. Все устройства подключены к этим линиям параллельно.

Если в микропроцессорную систему надо ввести входной код (или входной сигнал), то процессор по шине адреса обращается к нужному устройству ввода/вывода и принимает по шине данных входную информацию. Если из микропроцессорной системы надо вывести выходной код (*или выходной сигнал*)*, то процессор обращается по шине адреса к нужному устройству ввода/вывода и передает ему по шине данных выходную информацию.*

Если информация должна пройти сложную многоступенчатую обработку, то процессор может хранить промежуточные результаты в системной оперативной памяти. Для обращения к любой ячейке памяти процессор выставляет ее адрес на шину адреса и передает в нее информационный код по шине данных или же принимает из нее информационный код по шине данных.

В памяти (*оперативной и постоянной*) *находятся также и управляющие коды* (*команды выполняемой процессором программы*)*, которые процессор также читает по шине данных с адресацией по шине адреса.* Постоянная память используется в основном для хранения программы начального пуска вычислительной системы, которая выполняется каждый раз после включения питания. Информация в нее заносится изготовителем раз и навсегда.

Таким образом, в вычислительной системе все информационные коды и коды команд передаются по шинам последовательно, по очереди. Это определяет сравнительно невысокое быстродействие микропроцессорной системы. Оно ограничено обычно даже не быстродействием процессора (которое тоже очень важно) и не скоростью обмена по системной шине (магистрали), а именно последовательным характером передачи информации по системной шине (магистрали).

Важно учитывать, что устройства ввода/вывода чаще всего представляют собой устройства на "жесткой логике". Иногда устройства ввода/вывода имеют в своем составе процессор, то есть представляют собой небольшую специализированную микропроцессорную систему. Это позволяет переложить часть программных функций на устройства ввода/вывода, разгрузив центральный процессор системы.

1.6. Режимы работы вычислительной системы

Как уже отмечалось, вычислительная система обеспечивает большую гибкость работы, она способна настраиваться на любую задачу. Гибкость эта обусловлена прежде всего тем, что функции, выполняемые системой, определяются программой (программным обеспечением, software), которую выполняет процессор. Но гибкость микропроцессорной системы определяется не только этим. Настраиваться на задачу помогает еще и ***выбор режима работы системы****, то есть режима обмена информацией по системной магистрали* (*шине*).

Практически любая развитая вычислительная (микропроцессорная) система поддерживает три основных режима обмена по системной магистрали:

- *программный обмен информацией;*
- *обмен с использованием прерываний* (*Interrupts*)*;*
- *обмен с использованием прямого доступа к памяти*(*ПДП, DMA – Direct Memory Access*).

Программный обмен информацией *является основным в любой микропроцессорной системе.* Он предусмотрен всегда, без него невозможны другие режимы обмена. *Все операции* (*циклы*) *обмена информацией в данном случае инициируются только процессором, все они выполняются строго в порядке, предписанном исполняемой программой. Процессор читает (выбирает) из памяти коды команд и исполняет их, читая данные из памяти или из устройства ввода/вывода, обрабатывая их, записывая данные в память или передавая их в устройство ввода/вывода.*

Путь процессора по программе может быть линейным, циклическим, может содержать переходы (прыжки), но он всегда непрерывен и полностью находится под контролем процессора. Ни на какие внешние события, не связанные с программой, процессор не реагирует (рис. 1.8).

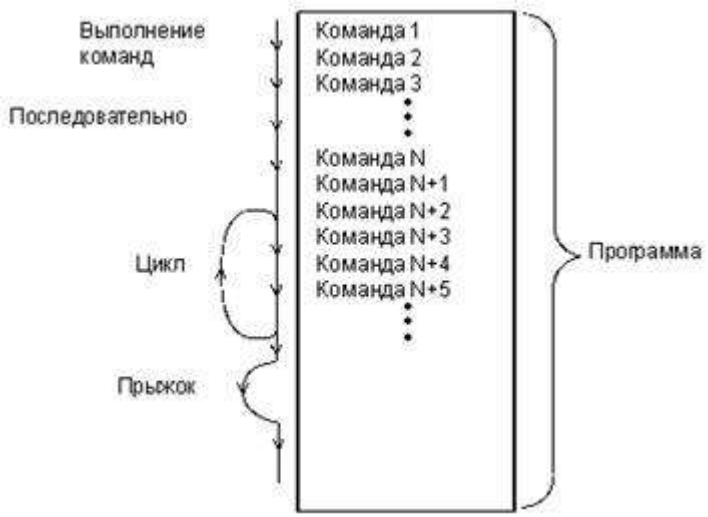

Рисунок 1.8 – Программный обмен информацией

Обмен по прерываниям *используется тогда, когда необходима реакция микропроцессорной системы на какое-то внешнее событие, на приход внешнего сигнала.* В случае компьютера внешним событием может быть, например, нажатие на клавишу клавиатуры или приход по локальной сети пакета данных. Компьютер должен

реагировать на это, соответственно, выводом символа на экран или же чтением и обработкой принятого по сети пакета.

В общем случае организовать реакцию на внешнее событие можно тремя различными путями:

- *с помощью постоянного программного контроля факта наступления события* (так называемый метод опроса флага или polling);

- *с помощью **прерывания**, то есть насильственного перевода процессора с выполнения текущей программы на выполнение экстренно необходимой программы*;

- *с помощью прямого доступа к памяти*, то есть без участия процессора при его отключении от системной магистрали.

Первый случай с опросом флага реализуется в микропроцессорной системе постоянным чтением информации процессором из устройства ввода/вывода, связанного с тем внешним устройством, на поведение которого необходимо срочно реагировать.

Во втором случае в режиме прерывания процессор, получив запрос прерывания от внешнего устройства (часто называемый IRQ – Interrupt ReQuest), заканчивает выполнение текущей команды и переходит к программе обработки прерывания. Закончив выполнение программы обработки прерывания, он возвращается к прерванной программе с той точки, где его прервали (рис. 1.9).

Рисунок 1.9 – Обслуживание прерывания

Прямой доступ к памяти (ПДП, DMA) – это режим, принципиально отличающийся от двух ранее рассмотренных режимов тем, что обмен по системной шине идет без участия процессора. *Внешнее устройство, требующее обслуживания, сигнализирует процессору, что режим ПДП необходим, в ответ на это процессор заканчивает выполнение текущей команды и отключается от всех шин, сигнализируя запросившему устройству, что обмен в режиме ПДП можно начинать. Операция ПДП сводится к пересылке информации из устройства ввода/вывода в память или же из памяти в устройство ввода/вывода. Когда пересылка информации будет закончена, процессор вновь возвращается к прерванной программе, продолжая ее с той точки, где его прервали* (рис. 1.10). Это похоже на режим обслуживания прерываний, но в данном случае процессор не участвует в обмене. Контроллер ПДП может считаться

специализированным процессором, который отличается тем, что сам не участвует в обмене, не принимает в себя информацию и не выдает ее (рис. 1.11).

Рисунок 1.10 – Обслуживание ПДП

Рисунок 1.11 – Информационные потоки в режиме ПДП

1.7. Архитектура микропроцессорных систем

До сих пор мы рассматривали только один тип архитектуры микропроцессорных систем – ***архитектуру с общей, единой шиной для данных и команд (одношинную, или принстонскую, фон-неймановскую архитектуру)***. *Соответственно, в составе системы в этом случае присутствует одна общая память, как для данных, так и для команд* (рис. 1.12).

Рисунок 1.12 – Архитектура с общей шиной данных и команд

Но существует также и альтернативный тип архитектуры микропроцессорной системы – это ***архитектура с раздельными шинами данных и команд (двухшинная, или гарвардская, архитектура)***. *Эта архитектура предполагает наличие в системе отдельной памяти для данных и отдельной памяти для команд* (рис. 1.13)*. Обмен процессора с каждым из двух типов памяти происходит по своей шине.*

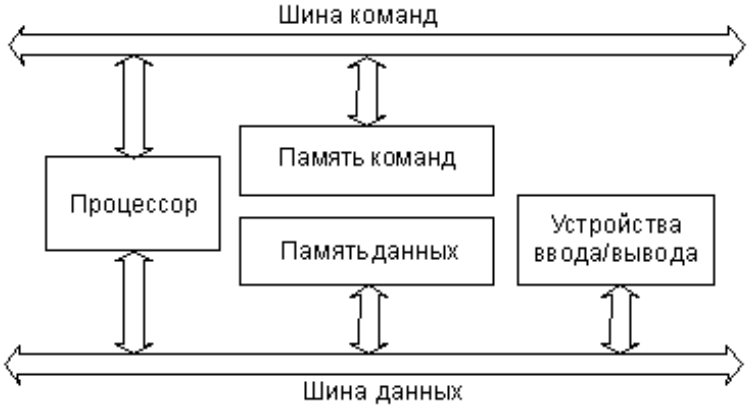

Рисунок 1.13 – Архитектура с раздельными шинами данных и команд

Архитектура с общей шиной распространена гораздо больше, она применяется, например, в персональных компьютерах и в сложных микрокомпьютерах. Архитектура с раздельными шинами применяется в основном в однокристальных микроконтроллерах.

Рассмотрим некоторые достоинства и недостатки обоих архитектурных решений.

Архитектура с общей шиной (принстонская, фон-неймановская) *проще, она не требует от процессора одновременного обслуживания двух шин, контроля обмена по двум шинам сразу. Наличие единой памяти данных и команд позволяет гибко распределять ее объем между кодами данных и команд.* Например, в некоторых случаях нужна большая и сложная программа, а данных в памяти надо хранить не слишком много. В других случаях, наоборот, программа требуется простая, но необходимы большие объемы хранимых данных. Перераспределение памяти не вызывает никаких проблем, главное – чтобы программа и данные вместе помещались в памяти системы. Как правило, в системах с такой архитектурой память бывает довольно большого объема (до десятков и сотен мегабайт). Это позволяет решать самые сложные задачи.

Архитектура с раздельными шинами данных и команд *сложнее, она заставляет процессор работать одновременно с двумя потоками кодов, обслуживать обмен по двум шинам одновременно.* Программа может размещаться только в памяти команд, данные – только в памяти данных. Такая узкая специализация ограничивает круг задач, решаемых системой, так как не дает возможности гибкого перераспределения памяти. *Память данных и память команд в этом случае имеют не слишком большой объем, поэтому применение систем с данной архитектурой ограничивается обычно не слишком сложными задачами.*

Преимущество архитектуры с двумя шинами(*гарвардской*) *заключается прежде всего в высоком быстродействии.* Дело в том, что при единственной шине команд и данных процессор вынужден по одной этой шине принимать данные (из памяти или устройства ввода/вывода) и передавать данные (в память или в устройство ввода/вывода), а также читать команды из памяти. Естественно, одновременно эти пересылки кодов по магистрали происходить не могут, они должны производиться по очереди. В случае двухшинной архитектуры обмен по обеим шинам может быть независимым, параллельным во времени. Соответственно, структуры шин (количество разрядов кода адреса и кода данных, порядок и скорость обмена информацией и т.д.) могут быть выбраны оптимально для той задачи, которая решается каждой шиной.