**СОДЕРЖАНИЕ**

|  |  |
| --- | --- |
| Введение | 4 |
| 1. Разработка общей структуры микро-ЭВМ | 8 |
| 2. Разработка основных устройств микро-ЭВМ | 13 |
| 3. Принципиальная схема устройства управления. | 22 |
| Заключение | 25 |
| Литература | 26 |
| Приложение А. Принципиальная схема микро-ЭВМ | 27 |
| Приложение Б. Временная диаграмма работы микроЭВМ (команды) | 28 |
| Приложение В. Временная диаграмма работы ЭВМ (прерывание) | 29 |
| Приложение Г. Временная диаграмма работы микроЭВМ (ПДП) | 30 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**ВВЕДЕНИЕ**

Диапазон применения микропроцессорной техники сейчас очень широк, требования к микропроцессорным системам предъявляются самые разные. Поэтому сформировалось несколько типов микропроцессорных систем, различающихся мощностью, универсальностью, быстродействием и структурными отличиями. Основные типы следующие:

* микроконтроллеры — наиболее простой тип микропроцессорных систем, в которых все или большинство узлов системы выполнены в виде одной микросхемы;
* контроллеры — управляющие микропроцессорные системы, выполненные в виде отдельных модулей;
* микрокомпьютеры — более мощные микропроцессорные системы с развитыми средствами сопряжения с внешними устройствами.
* компьютеры (в том числе персональные) — самые мощные и наиболее универсальные микропроцессорные системы.

Четкую границу между этими типами иногда провести довольно сложно. Быстродействие всех типов микропроцессоров постоянно растет, и нередки ситуации, когда новый микроконтроллер оказывается быстрее, например, устаревшего персонального компьютера. Но кое-какие принципиальные отличия все-таки имеются. Микроконтроллеры представляют собой универсальные устройства, которые практически всегда используются не сами по себе, а в составе более сложных устройств, в том числе и контроллеров. Системная шина микроконтроллера скрыта от пользователя внутри микросхемы. Возможности подключения внешних устройств к микроконтроллеру ограничены. Устройства на микроконтроллерах обычно предназначены для решения одной задачи. Контроллеры, как правило, создаются для решения какой-то отдельной задачи или группы близких задач. Они обычно не имеют возможностей подключения дополнительных узлов и устройств, например, большой памяти, средств ввода/вывода. Их системная шина чаще всего недоступна пользователю. Структура контроллера проста и оптимизирована под максимальное быстродействие. В большинстве случаев выполняемые программы хранятся в постоянной памяти и не меняются. Конструктивно контроллеры выпускаются в одноплатном варианте. Микрокомпьютеры отличаются от контроллеров более открытой структурой, они допускают подключение к системной шине нескольких дополнительных устройств. Производятся микрокомпьютеры в каркасе, корпусе с разъемами системной магистрали, доступными пользователю. Микрокомпьютеры могут иметь средства хранения информации на магнитных носителях (например, магнитные диски) и довольно развитые средства связи с пользователем (видеомонитор, клавиатура). Микрокомпьютеры рассчитаны на широкий круг задач, но в отличие от контроллеров, к каждой новой задаче его надо приспосабливать заново. Выполняемые микрокомпьютером программы можно легко менять. Наконец, компьютеры и самые распространенные из них — персональные компьютеры — это самые универсальные из микропроцессорных систем. Они обязательно предусматривают возможность модернизации, а также широкие возможности подключения новых устройств. Их системная шина, конечно, доступна пользователю. Кроме того, внешние устройства могут подключаться к компьютеру через несколько встроенных портов связи (количество портов доходит иногда до 10). Компьютер всегда имеет сильно развитые средства связи с пользователем, средства длительного хранения информации большого объема, средства связи с другими компьютерами по информационным сетям. Области применения компьютеров могут быть самыми разными: математические расчеты, обслуживание доступа к базам данных, управление работой сложных электронных систем, компьютерные игры, подготовка документов и т.д. Существует несколько типов архитектур микропроцессорных систем. Одна из них — архитектура с общей, единой шиной для данных и команд (одношинная, или принстонская, фон-неймановская архитектура). Соответственно, в составе системы в этом случае присутствует одна общая память, как для данных, так и для команд ([рис. 1.1](http://www.excode.ru/art6446p1.html#image.1.15)).

  
**Рис. 1.1.** Архитектура с общей шиной данных и команд (принстонская).

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. 1.2](http://www.excode.ru/art6446p1.html#image.1.16)). Обмен процессора с каждым из двух типов памяти происходит по своей шине. 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.

  
**Рис. 1.2.** Архитектура с раздельными шинами данных и команд(гарвардская).

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. В чем же преимущество архитектуры с двумя шинами (гарвардской)? В первую очередь, в 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. Современные процессоры способны совместить во времени выполнение команд и проведение циклов обмена по системной шине. Использование конвейерных технологий и быстрой кэш-памяти позволяет им ускорить процесс взаимодействия со сравнительно медленной системной памятью. Повышение тактовой частоты и совершенствование структуры процессоров дают возможность сократить время выполнения команд. Но дальнейшее увеличение быстродействия системы возможно только при совмещении пересылки данных и чтения команд, то есть при переходе к архитектуре с двумя шинами. 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. Поэтому при прочих равных условиях переход на двухшинную архитектуру ускоряет работу микропроцессорной системы, хотя и требует дополнительных затрат на аппаратуру, усложнения структуры процессора. Память данных в этом случае имеет свое распределение адресов, а память команд — свое. Проще всего преимущества двухшинной архитектуры реализуются внутри одной микросхемы. В этом случае можно также существенно уменьшить влияние недостатков этой архитектуры. Поэтому основное ее применение — в микроконтроллерах, от которых не требуется решения слишком сложных задач, но зато необходимо максимальное быстродействие при заданной тактовой частоте. Для реализации данного проекта было принято решения использовать программное обеспечение MaxPlus фирмы Altera ввиду ее удобства и простоты по сравнению с конкурентом (пакетом WebPack фирмы Xilinx).

**1. РАЗРАБОТКА ОБШЕЙ СТРУКТУРЫ МИКРО-ЭВМ. 1.1 ФУНКЦИОНАЛЬНАЯ СХЕМА МИКРО-ЭВМ.**

Ядром любой микропроцессорной системы является микропроцессор или просто процессор. Микропроцессор — это тот узел, блок, который производит всю обработку информации внутри микропроцессорной системы. Остальные узлы выполняют всего лишь вспомогательные функции: хранение информации (в том числе и управляющей информации, то есть программы), связи с внешними устройствами, связи с пользователем и т.д. Процессор 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 таких элементарных операций, выполняемых процессором, может достигать нескольких сотен.



**Рис. 1.3** Структура и функционирование простейшего ЭВМ

Свои операции процессор выполняет последовательно, то есть одну за другой, по очереди. Существуют процессоры и с параллельным выполнением некоторых операций, встречаются также микропроцессорные системы, в которых несколько процессоров работают над одной задачей параллельно, но это редкие исключения. С одной стороны, последовательное выполнение операций — 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. Простые алгоритмы выполняются быстрее сложных. То есть микропроцессорная система способна сделать все, но работает она не слишком быстро, ведь все информационные потоки приходится пропускать через один-единственный узел — микропроцессор. 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). 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. Схема простейшего процессора представлена на рис.1.3.

**1.2. ОПИСАНИЕ ВЗАИМОДЕЙСТВИЯ ВСЕХ БЛОКОВ МИКРО- ЭВМ В ХОДЕ ВЫПОЛНЕНИЯ КОМАНД ПРОГРАММЫ.**

Классическая ЭВМ состоит из трех основных устройств: арифметико-логического устройства, устройства управления и запоминающего устройства. Рассмотрим особенности организации этих устройств. Блок выборки команд выполняет чтение команд из памяти и их дешифрацию. В первых микропроцессорах было невозможно одновременное выполнение предыдущей команды и выборка следующей команды, так как процессор не мог совмещать эти операции. Но уже в 16-разрядных процессорах появляется так называемый конвейер (очередь) команд, позволяющий выбирать несколько следующих команд, пока выполняется предыдущая. Два процесса идут параллельно, что ускоряет работу процессора. Конвейер представляет собой небольшую внутреннюю память процессора, в которую при малейшей возможности (при освобождении внешней шины) записывается несколько команд, следующих за исполняемой. Развитием идеи конвейера стало использование внутренней кэш-памяти процессора, которая заполняется командами, пока процессор занят выполнением предыдущих команд.. Для большего ускорения выборки команд в современных процессорах применяют совмещение выборки и дешифрации, одновременную дешифрацию нескольких команд, несколько параллельных конвейеров команд, предсказание команд переходов и некоторые другие методы. Арифметико-логическое устройство (или АЛУ, ALU) предназначено для обработки информации в соответствии с полученной процессором командой. Примерами обработки могут служить логические операции (типа логического "И", "ИЛИ", "Исключающего ИЛИ" и т.д.) то есть побитные операции над операндами, а также арифметические операции (типа сложения, вычитания, умножения, деления и т.д.). Над какими кодами производится операция, куда помещается ее результат — определяется выполняемой командой. Если команда сводится всего лишь к пересылке данных без их обработки, то АЛУ не участвует в ее выполнении. Быстродействие АЛУ во многом определяет производительность процессора. Причем важна не только частота тактового сигнала, которым тактируется АЛУ, но и количество тактов, необходимое для выполнения той или иной команды. Для повышения производительности разработчики стремятся довести время выполнения команды до одного такта, а также обеспечить работу АЛУ на возможно более высокой частоте. Один из путей решения этой задачи состоит в уменьшении количества выполняемых АЛУ команд, создание процессоров с уменьшенным набором команд (так называемые RISC-процессоры). Другой путь повышения производительности процессора — использование нескольких параллельно работающих АЛУ. Регистры процессора представляют собой по сути ячейки очень быстрой памяти и служат для временного хранения различных кодов: данных, адресов, служебных кодов. Операции с этими кодами выполняются предельно быстро, поэтому, в общем случае, чем больше внутренних регистров, тем лучше. Кроме того, на быстродействие процессора сильно влияет разрядность регистров. Именно разрядность регистров и АЛУ называется внутренней разрядностью процессора, которая может не совпадать с внешней разрядностью. По отношению к назначению внутренних регистров существует два основных подхода. Первого придерживается, например, компания Intel, которая каждому регистру отводит строго определенную функцию. С одной стороны, это упрощает организацию процессора и уменьшает время выполнения команды, но с другой — снижает гибкость, а иногда и замедляет работу программы. Например, некоторые арифметические операции и обмен с устройствами ввода/вывода проводятся только через один регистр — аккумулятор, в результате чего при выполнении некоторых процедур может потребоваться несколько дополнительных пересылок между регистрами. Второй подход состоит в том, чтобы все (или почти все) регистры сделать равноправными, как , например, в 16-разрядных процессорах Т-11 фирмы DEC. При этом достигается высокая гибкость, но необходимо усложнение структуры процессора. Существуют и промежуточные решения, в частности, в процессоре MC68000 фирмы Motorola половина регистров использовалась для данных, и они были взаимозаменяемы, а другая половина — для адресов, и они также взаимозаменяемы. Схема управления прерываниями обрабатывает поступающий на процессор запрос прерывания, определяет адрес начала программы обработки прерывания (адрес вектора прерывания), обеспечивает переход к этой программе после выполнения текущей команды и сохранения в памяти (в стеке) текущего состояния регистров процессора. По окончании программы обработки прерывания процессор возвращается к прерванной программе с восстановленными из памяти (из стека) значениями внутренних регистров(рис.1.4).

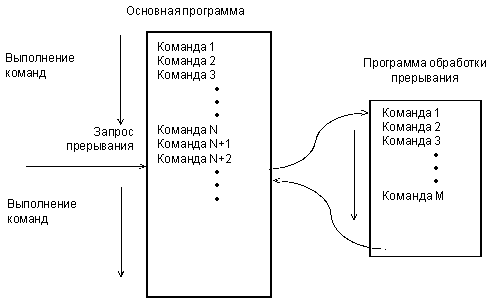
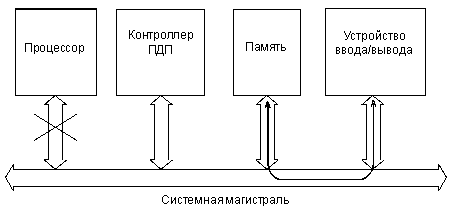
 **Рис.1.4.**  Обслуживание прерывания.

Схема управления прямым доступом к памяти служит для временного отключения процессора от внешних шин и приостановки работы процессора на время предоставления прямого доступа запросившему его устройству.

 **Рис. 1.5.**  Информационные потоки в режиме ПДП

Обмен по системной шине идет без участия процессора(рис. 1.5.). 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. Логика управления организует взаимодействие всех узлов процессора, перенаправляет данные, синхронизирует работу процессора с внешними сигналами, а также реализует процедуры ввода и вывода информации.

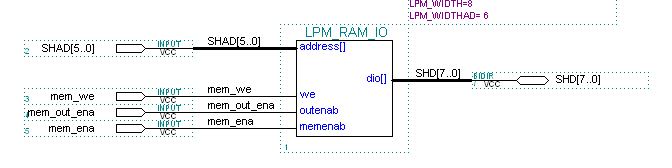
Таким образом, в ходе работы процессора схема выборки команд выбирает последовательно команды из памяти, затем эти команды выполняются, причем в случае необходимости обработки данных подключается АЛУ. На входы АЛУ могут подаваться обрабатываемые данные из памяти или из внутренних регистров. Во внутренних регистрах хранятся также коды адресов обрабатываемых данных, расположенных в памяти. Результат обработки в АЛУ записывается во внутренний регистр или в память (как источник, так и приемник данных указывается в составе кода команды). При необходимости информация может переписываться из памяти (или из устройства ввода/вывода) во внутренний регистр или из внутреннего регистра в память (или в устройство ввода/вывода). Внутренние регистры любого микропроцессора обязательно выполняют две служебные функции: •определяют адрес в памяти, где находится выполняемая в данный момент команда (функция счетчика команд или указателя команд); •определяют текущий адрес стека (функция указателя стека). В разных процессорах для каждой из этих функций может отводиться один или два внутренних регистра. Эти два указателя отличаются от других не только своим специфическим, служебным, системным назначением, но и особым способом изменения содержимого. Их содержимое программы могут менять только в случае крайней необходимости, так как любая ошибка при этом грозит нарушением работы компьютера, зависанием и порчей содержимого памяти. Содержимое указателя (счетчика) команд изменяется следующим образом. В начале работы системы (при включении питания) в него заносится раз и навсегда установленное значение. Это первый адрес программы начального запуска. Затем после выборки из памяти каждой следующей команды значение указателя команд автоматически увеличивается (инкрементируется) на единицу (или на два в зависимости от формата команд и типа процессора). То есть следующая команда будет выбираться из следующего по порядку адреса памяти. При выполнении команд перехода, нарушающих последовательный перебор адресов памяти, в указатель команд принудительно записывается новое значение — новый адрес в памяти, начиная с которого адреса команд опять же будут перебираться последовательно. Такая же смена содержимого указателя команд производится при вызове подпрограммы и возврате из нее или при начале обработки прерывания и после его окончания.

**2. РАЗРАБОТКА ОСНОВНЫХ УСТРОЙСТВ МИКРО-ЭВМ.**

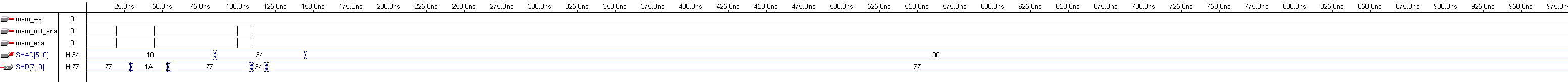
**2.1. ЗАПОМИНАЮЩИЕ УСТРОЙСТВА. ФУНКЦИОНАЛЬНЫЙ СОСТАВ И ВРЕМЕННЫЕ ДИАГРАММЫ РАБОТЫ ОЗУ.**

а Разработчики фирмы Altera предоставляют несколько разновидностей запоминающих устройств для своих ПЛИС: ПЗУ - lpm\_rom\_io, ОЗУ- lpm\_ram\_dp, lpm\_ram\_dq и lpm\_ram\_io. lpm\_ram\_dp – это ОЗУ двойного доступа, позволяющее одновременно проводить операции чтения и записи в память. lpm\_ram\_dq – это ОЗУ с разделением шин данных для чтения и записи. lpm\_ram\_io – обычное ОЗУ с общей двунаправленной шиной для записи и чтения данных из памяти.

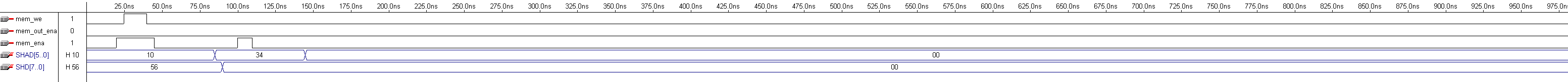
Наиболее просты в функционировании ЗУ - lpm\_rom\_io и lpm\_ram\_io. Рассмотрим работу асинхронных ПЗУ и ОЗУ этих типов. На рис. 2.1 изображена принципиальная схема ОЗУ lpm\_ram\_io, а на рис. 2.2 и 2.3 представлены временные диаграммы чтения и записи ОЗУ.



**Рис. 2.1.** Принципиальная схема ОЗУ.



**Рис. 2.2.** Временная диаграмма цикла чтения из ОЗУ



**Рис. 2.3.** Временная диаграмма цикла записи в ОЗУ.

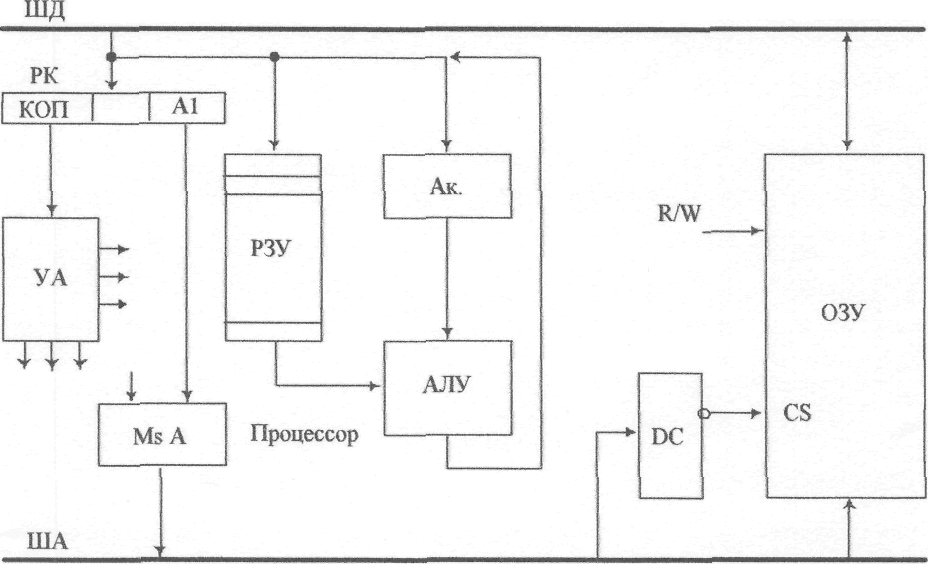
Как видно из рисунков 2.2 и 2.3 для ОЗУ требуется 3 управляющих сигнала: сигнал включения mem\_ena, сигнал выдачи данных на выход ОЗУ mem\_out\_ena и строб записи mem\_we.

**2.2. УСТРОЙСТВО УПРАВЛЕНИЯ.**

Устройство управления непрерывно выполняет циклическую последовательность действий: поочередное извлечение команд из памяти и их исполнение. В УУ наибольший интерес представляет регистр микрокоманды. Программа для микропроцессорной системы управления создается в виде последовательности команд. Каждая команда в общем случае содержит указание того, что должно быть сделано (код операции), и определение объекта действия, т.е. операнд. Способ задания операнда называется режимом адресации. Для микропроцессоров разработано около двух десятков режимов адресации, направленных на сокращение длины программы и времени ее выполнения. Наиболее простой, но и наиболее неэкономичной, является прямая адресация, при этом в поле операнда содержится полный адрес операнда в памяти. При использовании данного способа адресации второй операнд целесообразно адресовать косвенно или неявно. В противном случае хранение в структуре команды двух прямых

значений адресов приведёт к существенному увеличению длины формата, что в свою очередь ведёт к увеличению общего объёма оперативной памяти, используемой для хранения программ. Структурная схема, реализующая прямой способ адресации представлена на рис. 2.4.

При исполнении команд с данным способом адресации на первом этапе выполняется расшифровка поля КОП управляющим блоком. После расшифровки адрес А1 через мультиплексор адреса поступает на шину адреса. Дешифратор модуля ОЗУ активизирует на своём выходе соответствующую линию выборки. Далее операнд считывается в аккумулятор для последующей обработки в АЛУ.



**Рис. 2.4.** Структурная схема прямой адресации

Система команд.

Как было уже сказано, микропроцессор имеет индивидуальный набор команд. Но определенные закономерности присущи всем вычислительным системам, В зависимости от архитектуры конкретного микропроцессор построение слова, образующего команду, т.е. структура, которую принято называть форматом команды, может изменяться в широких пределах. Команды могут иметь фиксированную длину (например, одно машинное слово) или переменную длину (два, три и более слов); это определяется способом адресации к объекту действия, т.е. операнду. По формату набор команд можно разделить на две категории:

1. безадресные команды (без ссылки на данные). Команда содержит  
   только код операции и состоит из одного машинного слова.  
   Примерами являются команды ОСТАНОВ, ВОЗВРАТ и т.д.

2) команды с обращением к памяти (со ссылкой на данные).  
Команда должна содержать достаточно длинное слово, чтобы  
определить либо адрес данных в памяти, а в некоторых случаях  
промежуточный адрес, либо непосредственно сам операнд.

Диапазон адресации к ячейкам памяти и фактическая техника адресации, заложенная в архитектуре микропроцессора, определяют длину зоны операнда в команде. Для того чтобы можно было адресоваться к памяти емкостью 4К, требуется иметь зону адреса в 12бит, а для памяти емкостью 64К - 16бит. Соответственно приходится использовать команды, состоящие из двух, трех и более слов.

Классификация команд.

Все команды микропроцессора можно разделить на следующие основные типы или функциональные группы:

1. перемещения данных. Организуют передачу информации в  
   аккумулятор из памяти или, наоборот, из аккумулятора в память,  
   а также в аккумулятор из устройства ввода и из аккумулятора в  
   устройства вывода.
2. преобразования данных (арифметические и логические).  
   Используются для преобразования информации, т.е. выполнения  
   арифметических действий, например, СЛОЖИТЬ, ВЫЧЕСТЬ, и  
   логических операций, например, И, ИЛИ и т. д.
3. управления программой. Изменяют содержание программного  
   счетчика, являющегося адресом следующей команды.
4. ввода-вывода. Команды, организующие обмен информацией с  
   внешними устройствами системы.
5. специальные. Команды, которые непосредственно не выполняют  
   рассмотренные выше функции. Например, возможность  
   прерывания программы, запрет прерывания и т.д.

Рассмотрим формат команд для проектируемого ЭВМ:

Двухбайтная команда

1-ый байт:

|  |  |
| --- | --- |
| 7 4 | 3 0 |
| Код команды | № регистра |

2-ой байт

|  |
| --- |
| 7 0 |
| Адрес операнда |

В таблице 1 приведены коды команд.

Таблица 1

|  |  |
| --- | --- |
| Код команды | Команда |
| 0001 | MOV reg,adr |
| 0110 | MOV adr,reg |
| 1010 | JMP adr |
| 0101 | ADD reg,adr |
| 0100 | ADD adr,reg |
| 0010 | NOR reg,adr |
| 0011 | NOR adr,reg |
| 1100 | STI |
| 0101 | STI k |
| 1001 | CLI k |
| 1011 | HLT |

**2.3. АРИФМЕТИКО-ЛОГИЧЕСКОЕ УСТРОЙСТВО.**

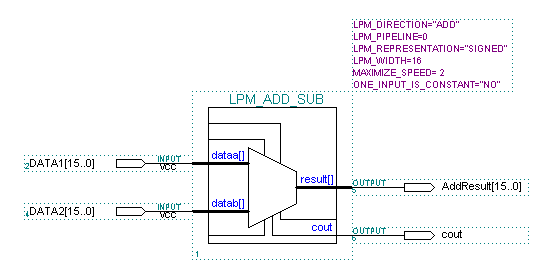
2.3.1Логическое построение АЛУ.

Этот блок микропроцессора представляет собой сложную двухвходовую комбинационную схему, которая сигналами от УУ настраивается на выполнение определенной операции. Свое название схема получила потому, что в ней комбинация сигналов на входе однозначно определяет сигнал на выходе.

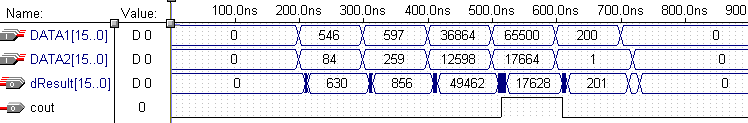
АЛУ представляет собой совокупность простейших электронных схем, способных хранить двоичные числа и выполнять над ними элементарные операции булевой алгебры. Данные, над которыми выполняются операции, поступают из регистров процессора, а результат также направляется в регистры. Эти регистры представляют собой элементы внутренней памяти процессора и связаны с АЛУ линиями передачи сигналов. В качестве вторичного результата АЛУ формирует значения набора двоичных флагов (признаков). Значения флагов также фиксируются в регистрах процессора. Блок управления передает в АЛУ сигналы, управляющие процессом выполнения операций, передачей операндов в АЛУ и считыванием результатов.

Среди стандартных элементов MAX+plus присутствует сумматор-вычитатель, который мы и будем использовать для реализации функции сложения.

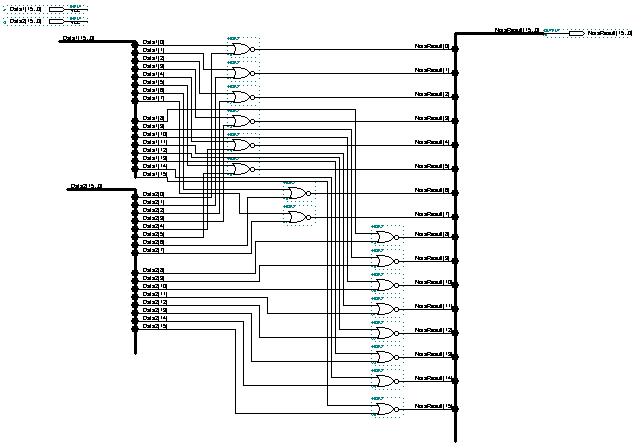
2.3.2.Реализация операции сложения.



Сигнал cout для формирования флага переполнения.



2.3.3. Реализация операции NOR



Блок РОНов вынесен за пределы АЛУ. Таким образом, АЛУ одним входом подключается к блоку РОНов, а другим к шине данных и через нее коммутируется с ОЗУ. Результат работы АЛУ может записываться как в блок РОНов, так и в ОЗУ.

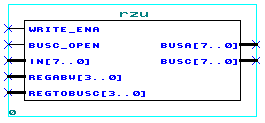
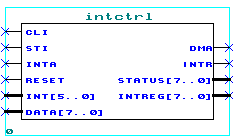


Схема блока РОНов представлена на рис. 2.6.

**Рис.2.6.** Схема блок РОНов.

**2.4. АППАРАТУРА И ФУНКЦИОНИРОВАНИЕ СИСТЕМЫ ПРЕРЫВАИЙ.**

Схема контроллера прерываний представлена на рис 2.9



**Рис.2.7.** Схема контроллера прерываний

Контроллер прерываний отвечает за взаимодействие процессора с устройствами ввода/вывода. Он включает в себя регистр масок, регистр статуса, регистр запроса прерываний. Если на вход контроллера пришел сигнал запроса на прерывание от устройства, контроллер прерываний проверяет, замаскировано ли оно. Если нет, то устанавливает сигнал INTR и ждет ответа от процессора.

Процессор может запретить некоторые или все прерывания для этого надо послать контроллеру соответствующую команду.

По стробу Reset во внутренний восьмиразрядный регистр маски IMR записывается число 0xFF (все прерывания разрешены). Шина INT[5..0] обеспечивает приём до 6 источников прерываний одновременно (приём рассчитан на импульсные сигналы). Запросы, пришедшие с этого входа, маскируются регистром IMR и сохраняются в регистре запросов IRR. Номер старшего прерывания дополнительно сохраняется в регистре статуса ISR. Тем самым реализуется приоритетность прерываний – прерывания со старшими номерами будут обслуживаться первыми. Прерывание INT4 интерпретируется как запрос от устройства, желающего использовать режим ПДП. Маскированный сигнал INT4 преобразуется в сигнал DMARequest. Прерывания INT0-INT5 интерпретируются как аппаратные. Если одно и более из этих прерываний демаскировано, то контроллер прерываний формирует на выходе INTR логическую единицу.

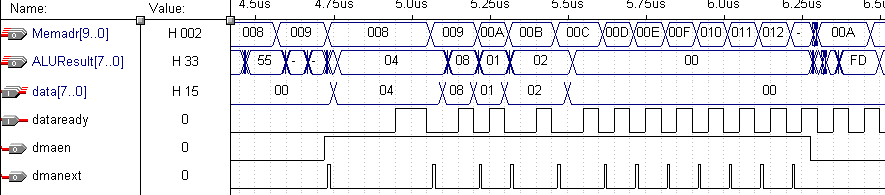
По единичному уровню сигнала INTA в регистре запросов IRR сбрасывается бит наиболее приоритетного прерывания, так как это прерывание считается взятым на обслуживание. Согласно принципу радиальной системы прерываний адреса обработчиков фиксированы и не могут быть изменены. Адрес вычисляется по следующему алгоритму: значение регистра ISR умножается на 2 (схемотехнически сдвигается на 1 бит).

Если в очереди больше нет запросов на обслуживание (IRR=0), то на выводе INTR формируется логический ноль.

По стробу на одном из входов cli или sti, контроллер прерываний выбирает байт с шины data[7..0] и интерпретирует его как номер прерывания, которое надо маскировать/демаскировать. Через внутреннюю схему управления irmodify производится сброс или установка соответствующего бита в регистре маски IMR.

**2.5. ПРЯМОЙ ДОСТУП К ПАМЯТИ. АППАРАТУРА И ФУНКЦИОНИРОВАНИЕ КОНТРОЛЛЕРА ПДП (КПДП).**

Контроллер ПДП служит для того, чтобы обеспечить взаимодействие внешнего устройства с ОЗУ без участия процессора. Для этого КПДП получив от контроллера прерываний сигнал, КПДП подключается к шинам и начинает вырабатывать управляющие сигналы и данные для ОЗУ.



**Рис.2.8.** Временная диаграмма работы КПДП.

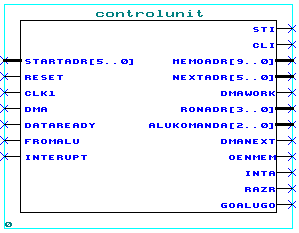
**2.6. МИКРО-ЭВМ. АППАРАТУРА И ФУНКЦИОНИРОВАНИЕ МИКРО-ЭВМ.**

Принципиальная схема микро-ЭВМ представлена в приложении. В данном устройстве была реализована шинная топология. Все блоки ЭВМ обращаются друг с другом посредством шин. Если одно устройство занимает шину, то другие отключены от нее и ждут необходимых управляющих сигналов. Рассмотрим работу микро-ЭВМ в целом. После прихода строба RESET, который поступает на устройство управления (УУ) процессор запускается. Внутри УУ блок выборки команды (УВК) инициализирует системный счетчик значением с шины StartAdr – с этого адреса начинается любая программа. УВК 2 байта из ОЗУ (значение программного счетчика каждый раз наращивается). Затем УВК дает сигнал о том, что команда считана и запускается счетчик выполнения команды. В течение не более 2-х тактов при подаче соответствующего адреса на ПЗУ выдается группа управляющих сигналов. После выполнения команды подается сигнал УВК о том, что надо выбирать следующую команду. И все повторяется заново. Если выполняемая команда арифметическо-логическая или пересылка данных, то сигналы подаются на АЛУ и ОЗУ. При выполнении операций маскирования/демаскирования УУ подает сигналы на контроллер прерываний. Это сигнал о приходе новой команды, код команды для контроллера КПДП, номер прерывания, которое надо замаскировать/демаскировать. Также на контроллер прерывания приходит сигнал разрешения прерывания. КПДП запускается по сигналу DMA. Он сразу выставляет запрос на прерывание и ждет разрешения записи в ОЗУ. После прихода разрешения прерывания КПДП выставляет строб записи в ОЗУ и каждый такт выдает на шины адреса и данных соответственно адрес, куда надо записать, и записываемое значение.

Временная диаграмма работы микро-ЭВМ представлена в приложении Г.

**3. ПРИНЦИПИАЛЬНАЯ СХЕМА УСТРОЙСТВА УПРАВЛЕНИЯ.**

Устройство управления предназначено для считывания очередной команды из ОЗУ. В соответствии с этой командой устройство управления вырабатывает управляющие сигналы для остальных блоков.



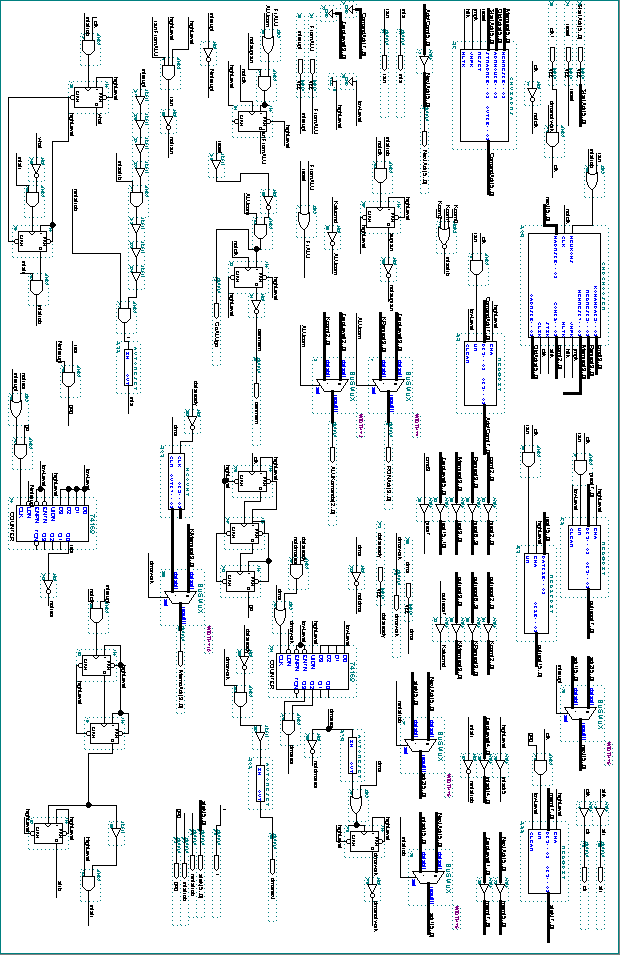
**Рис.3.1.** Схема УУ.

В таблице 2 представлены сигналы устройства управления.

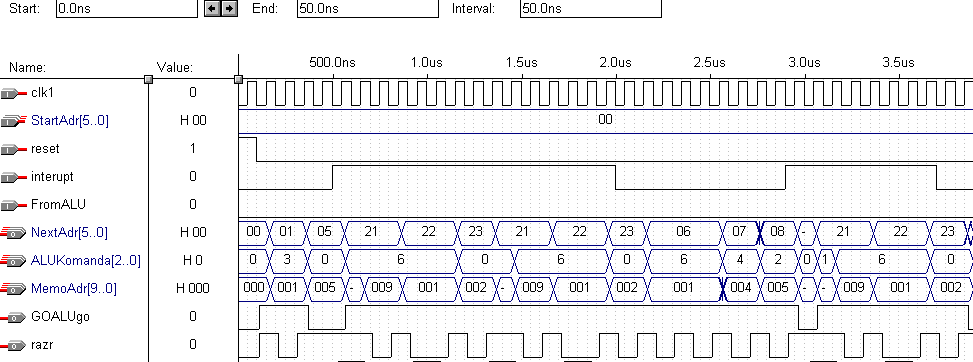
Таблица 2

|  |  |
| --- | --- |
| Входные сигналы | |
| CLK | Сигнал тактирования |
| STARTADR[5..0] | Данные с ШД |
| RESET | Запуск ЦП |
| FROMALU | Подтверждение от АЛУ |
| DMA | Запрос ПДП |
| DATAREADY | Готовность данных |
| INTRUPT | Запрос на прерывание |
| Выходные сигналы | |
| MEMOADR[9..0] | Шина адреса |
| OENMEM | Сигнал в ОЗУ |
| GOALUGO | Сигнал в АЛУ |
| RONADR[3..0] | Номер РОНа участвующего в арифметич. операции |
| ALUKOMANDA[2..0] | Команда АЛУ |
| NEXTADR[5..0] | Адрес след. команды |
| CLI | Команда CLI |
| STI | Команда STI |
| INTA | Завершение прерывания |
| DMANEXT | Запрос данных |

# Принципиальная схема УУ



Временная диаграмма работы АЛУ представлена на рисунке:



**Рис.3.2.** Временная диаграмма работы АЛУ.

**ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта была разработана схема микро-ЭВМ с заданными характеристиками на элементной базе ПЛИС FLEX 10KE. Моделирование микро-ЭВМ происходило в среде автоматизированного проектирования Max Plus II фирмы Altera. Спроектированы основные узлы микро-ЭВМ и проработано моделирование результатов их работы на временном симуляторе.

Полученное устройство построено на принципах принстонской архитектуры (раздельная память команд и данных), устройство управления ЭВМ воспринимает команды нефиксированной длины (на выполнение одной команды уходит от 3 до 4 тактов), устройство обладает ориентировочной тактовой частотой 50MHz.

**Литература**

1. Применение микропроцессоров для автоматизации технологических  
   процессов. Вершин О.Е. -Л: Энергоиздат-Линенград, 1986.
2. ПЛИС фирмы "ALTERA": элементная база, система проектирования и  
   языки описания аппаратуры. Стешенко В.Б. -М.: Издательский дом  
   "Додэка-ХХГ, 2002.
3. Структурная организация и архитектура компьютерных систем.  
   Проектирование и производство. У.Столлингс. - М.: Издательский дом  
   "Вильяме", 2002
4. Кобяк И П. Организация ввода-вывода в компьютерных системах:

Методическое пособие по курсам "Теория и проектирование ЭВМ"' и "Структурная и функциональная организация ЭВМ" для студентов специальности 22.01 "Вычислительные машины, системы и сети" факультета здочного обучения. -Мн.гБГУИР., 1996.

5. КобякИ П. Архитектура компьютерных систем ч1и ч2:

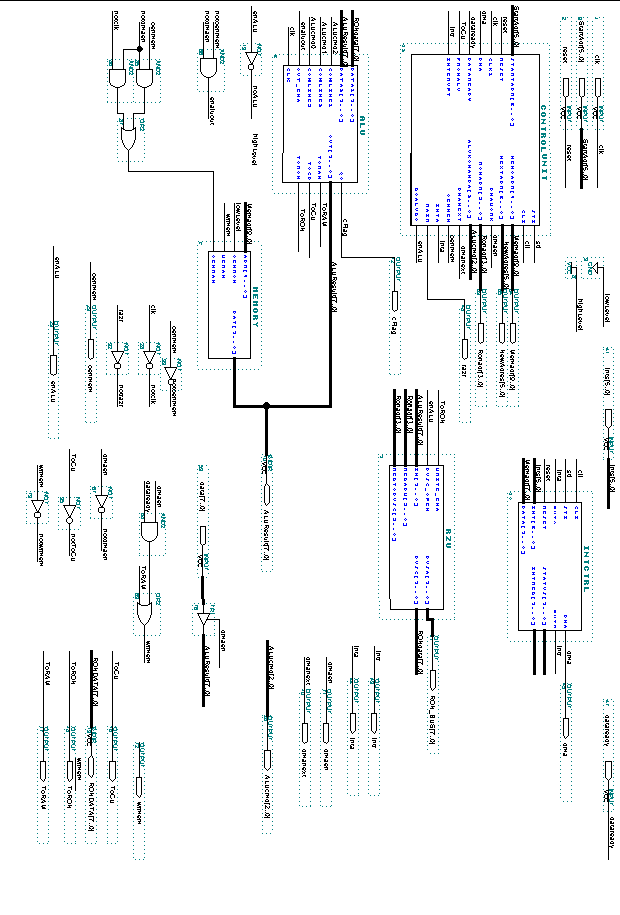
Методическое пособие по курсам "Теория и проектирование ЭВМ1" и "Структурная и функциональная организация ЭВМ" для студентов специальности 22.01 "Вычислительные машины, системы и сети" факультета здочного обучения. -Мн.:БГУИР, 2001.

6. КобякИ П. Организация памяти компьютерных систем:

Методическое пособие по курсам "Теория и проектирование ЭВМ1" и "Структурная и функциональная организация ЭВМ" для студентов специальности 22.01 "Вычислительные машины, системы и сети" факультета здочного обучения. -Мн.:БГУИР, 2000.

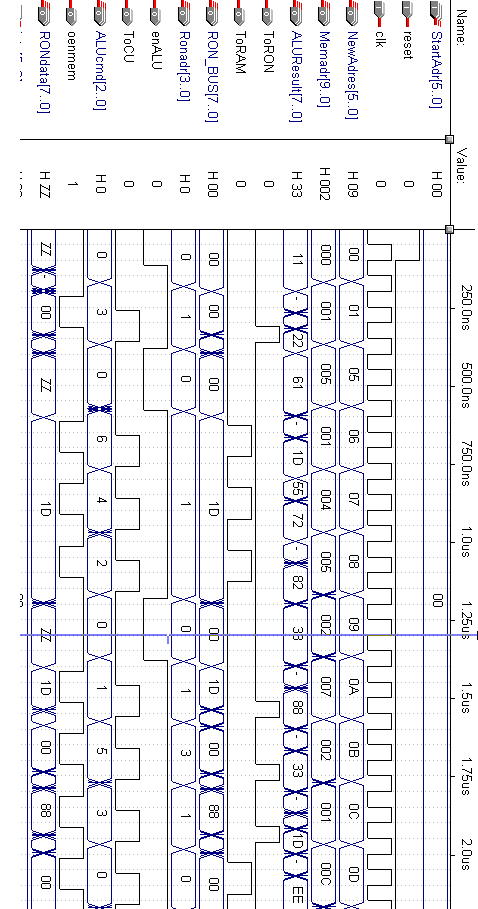
### Приложение А.

#### Принципиальная схема микро-ЭВМ



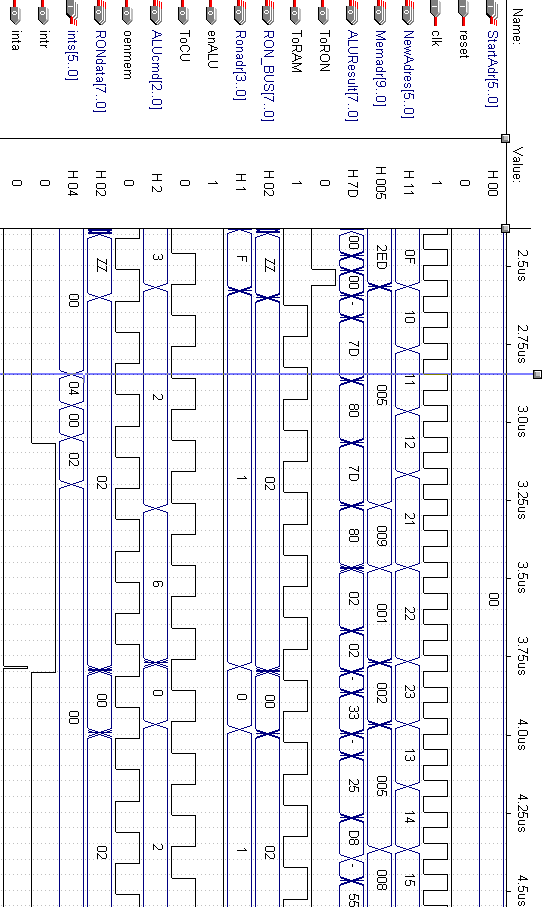
##### Приложение Б.

Временная диаграмма работы микро-ЭВМ (выполнение команд)



##### Приложение В.

Временная диаграмма работы микро-ЭВМ (прерывание)



##### Приложение Г.

Временная диаграмма работы микро-ЭВМ (работа КПДП)

