**СОДЕРЖАНИЕ**

Введение 3

1. РАЗРАБОТКА ОБШЕЙ СТРУКТУРЫ МИКРО-ЭВМ 4

2. РАЗРАБОТКА ОСНОВНЫХ УСТРОЙСТВ МИКРО-ЭВМ 10

3. Анализ и оптимизация разработанной микро-ЭВМ 13

Заключение 14

Список литературы 15

**Введение**

В наше время компьютерные технологии используются во всех сферах жизни. Электронную начинку можно встретить как в космической отрасли, так и в детских игрушках и даже в тостере. Конечно, разные устройства используются для разных нужд, в них используются различные микропроцессорные устройства.

Наиболее простые устройства оснащаются микроконтроллерами, вычислительное ядро которых обычно выполнено из одной микросхемы.

Более сложные системы основаны на контроллерах, вычислительные части которых зачастую делаются в виде отдельных модулей.

Многим устройствам необходимо обмениваться данными с удаленными (внешними) устройствами. Такие системы основывают на микрокомпьютерах — более мощных микропроцессорных системах со средствами сопряжения с внешними устройствами.

Самой мощной и универсальной микропроцессорной системой является компьютер.

Существует несколько типов архитектур микропроцессорных систем. В данном курсом проекте реализована архитектура с общей шиной для данных и команд (принстонская, фон-неймановская архитектура). Соответственно, в составе системы в этом случае присутствует одна общая память, как для данных, так и для команд (рис. 1).

  
**Рис. 1.** Архитектура с общей шиной данных и команд.

Архитектура с общей шиной проще, она не требует от процессора одновременного обслуживания двух или более шин. Наличие единой памяти данных и команд позволяет использовать системы для широкого ряда задач. 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.

С точки зрения архитектуры, система не различает данные и команды, а значит команды могут подгружаться из внешних источников как обычные данные, а затем выполняться на процессорных элементах.

**1. Разработка общей структуры микро-ЭВМ.**

**1.1 Функциональная схема микро-ЭВМ.**

Сердцем любой микропроцессорной системы является микропроцессор или просто процессор. Микропроцессор — это блок, который производит всю обработку информации внутри микропроцессорной системы. Остальные узлы выполняют вспомогательные функции: хранение информации (данных, текущей команды, текущего состояния), приема и передачи данных по опредленным протоколам, синхронизация работы процессора. 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, контроль за состоянием системы и предотвращением коллизий.

Все команды, выполняемые процессором, образуют архитектуру системы команд процессора. Структура и объем системы команд процессора определяют его быстродействие, гибкость, удобство использования. Система команд может быть рассчитана на узкий круг решаемых задач (у специализированных процессоров) или на максимально широкий круг задач (у универсальных процессоров). В системах, рассчитанных на узкий круг исполняемых программ, на производительность сильно влияет расположение отдельных блоков системы. Так, в некоторых системах с интенсивной работой с устройствами памяти скорость доступа к данным будет играть важнейшую роль, нежели скорость выполнения арифметических действий с данными.

Архитектура данного курсового проекта построена для интенсивных вычислений с использованием АЛУ и стека, а также для взаимодействия с внешними устройствами. Время доступа к данных регистров и стека мало (2 такта), в сравнении со временем доступа к памяти (при «cache hit» 2 такта, при «cache miss» 5 тактов, без учета задержек на ожидания доступа к памяти) или к памяти внешнего устройства.

**1.2. Разработка системы команд.**

В ходе проектирования курсового проекта, основываясь на архитектурных особенностях заданного варианта, а именно: ширина шины данных, ширина шина адреса, количество регистров и количество команд, обязательных к исполнению, заданная архитектура, была разработана система команд из 14 команд, включающая в себя команды работы с устройствами памяти, арифметико-логическим модулем и управления потоком команд.

В данной архитектуре команд роль устройства управления передана трехэтапному конвейеру, на каждом этапе выполняется отдельная часть, присущая работе каждого этапа. В ходе проектирования системы команд для каждого этапа конвейера выяснил, что на одном этапе для нескольких команд алгоритм действий совпадает.

Все команды имеют одинаковый размер, равный разрядности шины данных, а именно 16 бит.

Команда 1. Синтаксис «Pop R». Является командой выдачи данных из стека, единственным входным параметром команды адрес регистра R, в который будет записан результат выполнения команды. На первом этапе конвейера команда и адрес регистра пересылается во второй блок, где выполняется извлечение данных из стека. На третьем этапе данные сохраняются в регистр.

Команда 2. Синтаксис «HLT». Команда остановки микропроцессорной системы. Обнуляет все блоки обработки, промежуточные данные во временных регистрах конвейера и останавливает счетчик тактов. На первых двух тактах конвейера команда пересылается в следующий блок, на третьем этапе подается сигнал об остановке всех операций на всех блоках микропроцессорной системы.

Команда 3. Синтаксис «Push P». Команда записи данных в стек, единственным входным параметром команды является адрес регистра R, из которого будут записаны данные. На первом этапе конвейера данные из регистра R поступают во второй блок, где ведется запись данных в стек. Преимущество этой команды в том, что она выполняется на двух блоках конвейера без обращений к блоку памяти, что позволяет третьему блоку работать с этим блоком без коллизий.

Команды 4-7. Синтаксис «Команда R1,R2,R3». Команды с использованием арифметико-логического блока. Имеют три входных параметра – адреса двух регистров – операндов(R1 и R2) и адрес регистра сохранения результата(R3). На первом этапе конвейера операнды достаются из регистров и передаются во второй блок, где происходит выполнение операции на блоке АЛУ. В третий блок передается адрес регистра сохранения результата, результат выполнения команды, а также флаг переполнения для команд сдвига и вычитания. Команда 4 выполняет операцию вычитания (R3 = R1-R2), команда 5 выполняет операцию побитного «ИЛИ» (R3 = R1|R2), команда 6 выполняет операцию «НЕ исключающее ИЛИ» (R3 = NOT(R1 NOR R2)), команда 7 выполняет операцию кругового смещения влево регистра R1 на количество разрядов, хранящихся в R2.

Команда 8. Синтаксис «Mov R,Short\_Addr». Команда копирования 16 бит из регистра в память. На первом этапе конвейера данные достаются из регистра R, далее команда, адрес в памяти и данные передаются третьему блоку через второй для записи данных в память.

Команда 9. Синтаксис «Mov Short\_Addr, R». Команда копирования 16 бит из памяти в регистр. На первом этапе конвейера данные достаются из памяти с адресом High\_addr + Short\_Addr, далее команда, адрес регистра и данные передаются третьему блоку через второй для записи данных в регистр. Старшая часть адреса High\_Addr хранится в служебном регистре в первом блоке конвейера и передается всем остальным блокам конвейера.

Команда 10. Синтаксис «JMP Short\_Addr». Команда безусловного перехода. Изменяет указатель следующей команды в памяти. Команда выполняется только на первом этапе конвейера.

Команда 11. Синтаксис «Set Short\_Addr». Команда установки старшей части адреса следующей команды. Команда выполняется только на первом этапе конвейера.

Команда 12. Синтаксис «JMS Short\_Addr». Команда условного перехода. Изменяет указатель следующей команды в памяти. Команда и адрес передаются в третий блок, также из второго блока передается значение регистра флагов. Если флаг переполнения поднят, то осуществляется переход, иначе действий не производится. При переходе подается сигнал на изменение указателя следующей команды, а также устанавливаются первых два этапа конвейера в исходное состояние.

Команды 13-14. Синтаксис «SetDMA Direction,DST\_SHORT\_Addr| SRC\_SHORT\_ADR». Установка конфигурации адресов контроллера прямого доступа к памяти. Входные параметры направление (Direction) и адреса передаются в третий блок. В третьем блоке подаются сигналы на конфигурирование контроллера прямого доступа к памяти.

Данные о всех командах для всех этапов конвейера и общая система команд сведены в таблицы(см. Приложение). Как видно из таблиц, не все команды проходят все стадии конвейера. Данный факт позволяет на уровне написания программ оптимизировать работу микропроцессора.

**1.3. 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. В данном курсовом проекте роль устройства управления передана трехэтапному конвейеру, а так же реализовано несколько видов запоминающих устройств, отличающихся различными алгоритмами взаимодействия. Рассмотрим особенности взаимодействия этих устройств(см. структурную схему в приложении 1).

Основной идеей данной архитектуры – это перенос функций устройства управления на блоки конвейера. Все три блока конвейера не управляются извне. Блоки работают автономно и общаются друг с другом только двумя сигналами: сигнал готовности передать очередную порцию данных следующему блоку и сигнал готовности принять данные следующим блоком. Это означает что первый блок работает тогда, когда выходные данные отсутствуют. Второй блок работает тогда, когда первый блок готов отдать данные и если третий блок забрал готовые данные. Третий блок работает только тогда, когда второй блок сформировал для него выходные данные. Данный подход позволяет реализовывать команды, которые выполняются не на всех блоках без внесения аппаратных изменений.

Первоначально весь код программы записан в ПЗУ блока памяти. Размер ПЗУ составляет 8кб и имеет адресацию 000h-1FFFh, что позволяет записать 4096 команд. Возможно также использование данных из ОЗУ в качестве команд, предварительно загрузив команды из внешнего устройства посредством КПДП. Объем ОЗУ так же равен 8кб, адреса 2000h-3FFFh, что позволяет удвоить максимальный размер программы.

После старта микропроцессора, первый блок конвейера считывает первую команду с адресом 000h. Все данные, которые приходят из блока памяти попадают в кэш, для более быстрого повторного использования. На основе этой команды принимается решение о дальнейшей работе остальных блоков конвейера. Исходя из системы команд, не все команды требуют выполнения на всех трех этапах конвейера.

Команды могут использовать в качестве хранилища данных четыре типа устройств – ОЗУ, ОЗУ внешнего устройства, регистры общего назначения и стек. Многие команды используют либо стек, либо регистры общего назначения, так как задержка доступа к этим устройствам много меньше чем доступ к данным из ОЗУ при «промахе» на кэше. Для доступа к данным из внешнего устройства необходимо загрузить данные в ОЗУ в блоке памяти, из которого данные достаются по алгоритму доступа к блоку памяти. В конвейере два блока имеют доступ к шине блока памяти. Для разрешения конфликта доступа используется статический централизованный параллельный арбитр. Блок записи (третий) имеет высший приоритет, что позволяет разрешить конфликт чтение перед записью для команд, следующих через одну после команды записи результата в память.

Блоки стека и АЛУ имеют выходы только к блоку выполнения команд конвейера. При такой организации нет конфликтов доступа к блокам, что ускоряет работу всего этапа конвейера.

**2. Разработка основных устройств микро-ЭВМ.**

**2.1. Запоминающие устройства. Функциональный состав временные диаграммы работы ОЗУ и ПЗУ.**

В данном курсом проекте использовалось синхронный блок ОЗУ (LPM\_RAM\_IO) и асинхронный блок ПЗУ (LPM\_ROM). Оба устройства расположены на одной шине, поэтому было принято решение объединить два блока в один блок памяти. Так как в асинхронном режиме подается сигнал готовности блока ПЗУ, то для упрощения интерфейса доступа к блоку данных эмулируется сигнал готовности для синхронной памяти ОЗУ. Реализация данного эмулятора представлена в приложении 2. В качестве ядра блока памяти была использована наработка блока памяти из лабораторного курса СИФО ЭВМ. Ядро представлено в приложении 3. Блок памяти имеет пять входных сигналов (шина данных, шина адреса, сигнал записи, сигнал чтения, сигналы тактовых импульсов) и два выходных сигнала (шина данных и сигнал готовности устройства)

Временная диаграмма представлена в приложении 12. Из этой диаграммы видно, что запись в память ОЗУ происходит за два такта, при чтении из памяти ОЗУ правильные сигналы устанавливаются на шине данных во время фронта второго такта, правильные данные могут быть считаны либо по спаду второго такта, либо по фронту третьего. Установление сигнала на шине данных при чтении из памяти ПЗУ происходит во время фронта первого такта, что позволяет на втором такте считать верные значения.

**2.2. Устройство управления.**

В данной архитектуре функцию устройства управления берет на себя конвейер. Каждый этап конвейера отвечает за отдельные этапы выполнения команды.

**2.2.1. Блок выборки команд и операндов.**

Первый блок конвейера отвечает за передачу остальным блокам команды и операндов. Команда хранится в ПЗУ блока памяти, операнды хранятся в ОЗУ блока памяти либо в блоке регистров общего назначения.

Первый блок конвейера имеет указатель адреса команды, а так же устройство изменения указателя на следующую команду, либо на адрес, указанный в командах изменения адреса следующей команды.

Блок имеет выводы на блок памяти, блок регистров общего назначения, а также систему сигналов доступа к этим блокам посредством централизованного статичного арбитра. Блок также считывает тактовые импульсы, а также сигнал общего выключения системы. Блок выборки команд и операндов связан с блоком выполнения команды шинами для передачи команды, операндов и адресов результата. Так же блок сообщает второму блоку конвейера о готовности к передаче данных, считывает сигнал принятия данных от блока выполнения команды.

С блоком записи результата данный блок связывает сигнал изменения адреса указателя текущей команды, используемый в команде условного перехода.

Временная диаграмма блока выборки команд и операндов представлена в приложении 13. В памяти ПЗУ хранится программа.

|  |  |  |
| --- | --- | --- |
| адрес | код команды | команда |
| 0h | 3500 | Push 5 |
| 2h | 4512 | Sub 5,1,2 |
| 4h | 9400 | MOV 0,4 |
| 6h | A00A | JMS A |
| 8h | 0000 | 0 |
| Ah | 2000 | HLT |

На диаграмме показаны выходные сигнала на каждом такте выполнения данных команд, команда готовности блока (DataGetBlockReady) сбрасывается эмуляцией входного сигнала чтения данных вторым блоком(ProcessBlockReady).

**2.2.2. Блок выполнения команды.**

Данный блок выполняет операции с использованием блока АЛУ, а так же операции с блоком памяти типа стек.

Временная диаграмма для блока выполнения команды представленная в приложении 14. На диаграмме продемонстрированы основные команды, которые выполняются на этом блоке. А именно: команда загрузки в стек (команда 3) данных, выгрузки данных из стека(команда 1), команда пересылки команды и операндов третьему блоку (команда 2), команда вычитания в качестве демонстрации работы блока АЛУ(команда 4).

**2.2.3. Блок записи результата.**

Третий блок конвейера выполняет операции по записи операндов в ОЗУ блока памяти либо в блок регистров общего назначения. Также этот блок обрабатывает команду условного перехода – на основе полученных данных от второго блока о регистре флагов может быть выставлены сигналы на изменение указателя текущей команды с обнулением результатов выполнения первых двух блоков конвейера. Блок записи результата также выполняет команду остановки всей системы (HLT).

Как и блок выборки команд и операндов, этот блок имеет доступ к блоку памяти и блоку регистров общего назначения, а также систему сигналов доступа к этим блокам. Со вторым этапом блок связан сигналом о готовности второго блока и сигналом чтения данных. На первый блок подается сигнал об изменении адреса указателя текущей команды.

Временная диаграмма представлена в приложении 15. На диаграмме показаны разные режимы работы при различных входных сигналах.

**2.3. Арифметико-логическое устройство.**

АЛУ представляет из себя устройство выполнения арифметико-логических операций над двумя операндами. Схема устройства представлена в приложении 6.

Данный блок выполняет 4 операции:

NXOR – побитная инверсная логическая сумма двух 16-битных операндов.

OR – побитное логическое «ИЛИ» двух 16-битных операндов.

ROL – круговой сдвиг второго операнда влево на количество разрядов, указанных в первом операнде. Первый сдвинутый бит является выходным флагом. Схема устройства данного блока представлена в приложении 8. Временная диаграмма представлена в приложении 16

Sub – вычитание второго операнда из первого. Возможных выход за пределы разрядной сетки отмечается поднятием флага.

Блок АЛУ включает в себя сигналы загрузки операндов, шину данных, сигнал выдачи результата конкретной команды.

Временная диаграмма представлена в приложении 17. На диаграмме продемонстрированы результаты выполнения всех арифметико-логических команд.

**2.4. Блок регистров общего назначения**

Регистры общего назначения – блок памяти, имеющий наименьшую задержку на чтение/запись данных. Схема блока РОН представлена в приложении 5. Блок имеет сигнал записи, чтения и сброса данных, шину данных и шину тактовых импульсов.

Временная диаграмма представлена в приложении 18. На диаграмме показаны разные режимы работы при различных входных сигналах.

**2.5. Блок памяти типа стек**

Стек представляет собой устройство памяти, работающее по принципу FILO(First In Last Out). Схема устройства представлена в приложении 4. Блок имеет сигналы ввода и вывода данных, обнуления памяти, шину данных.

Временная диаграмма представлена в приложении 19. На диаграмме отображены последовательные записи трех элементов с последующей вычиткой трех элементов. При попытке вычитки из пустого устройства система не выдает сигналов на шину.

**2.6. Блок статического централизованного параллельного арбитра.**

Представляет собой устройство, которое разрешает только одному устройству доступ к конкретному блоку. Схема устройства представлена в приложении 7. Устройство имеет входные сигналы запросов на доступ к шине, входной сигнал занятости шины устройством, разрешающие сигналы для каждого устройства.

Временная диаграмма представлена в приложении 20. На диаграмме отображены режимы работы арбитра при различных входных сигналах.

**3. Анализ и оптимизация разработанной микро-ЭВМ**

В ходе разработки и проектирования микропроцессорной системы, основной идеей было создать систему с максимальной скоростью вычисления.

Для этих целей в архитектуру был добавлен конвейер. Он позволяет распараллелить выполнение команд, тем самым ускорить среднее время выполнения микропроцессорной программой.

Для ускорения системы функции устройства управления были переданы блокам конвейера, создана система независимой обработки этапов команд на блоках конвейера. Это позволят ускорить выполнение программы, так как не все команды исполняются на всех трех этапах конвейера.

Большинство команд ориентированно на работу с блоком регистров общего назначения – самого быстрого блока памяти в системе. Подобная архитектура команд позволяет также уменьшить количество запросов к блоку памяти, что уменьшает время ожидания доступа к блоку памяти для первого и третьего этапов конвейера.

Вынесение блока АЛУ и блока стека во второй блок конвейера предотвращает коллизии доступа к этим устройствам, время ожидания доступа сводится к нулю.

В входе разработки блоков конвейера было найдено решение, позволяющее ускорить выполнение команд первым блоком на один такт, что составляет от 15 до 30 процентов от времени выполнения команды на этом блоке. Ускорение стало доступно после выполнения микрокоманды перехода счетчика адреса на адрес следующей команды в параллельном режиме сразу после поступления команды из блока памяти.

В блоке выборки команд и операндов все команды, не требующие операндов были приведены под одно время выполнения, однако в некоторых командах требовались дополнительные такты для дешифрации команды. Разбиение выполнения таких команд приведет к ускорению выполнения некоторых команд, но и к усложнению архитектуры первого блока конвейера.

Еще одним способом ускорения выполнения программы является правильное расположение команд в программе. Две команды доступа к памяти, идущие друг за другом по алгоритму программы, вызывают временные задержки на ожидании доступа к блоку памяти.

**Заключение**

В результате выполнения курсового проекта была разработана схема микро-ЭВМ с заданными характеристиками. Моделирование данной микро-ЭВМ происходило в среде автоматизированного проектирования Altera Quartus II 9.1. Спроектированы основные узлы микро-ЭВМ и проработано моделирование результатов их работы на временном симуляторе.

Полученное устройство построено на принципе принстонской архитектуры (нет разделения информации на команды и данные), устройство имеет конвейер для параллельного выполнения команд. Коллизии доступа к блокам памяти решаются статическими централизованными параллельными арбитрами.**Список литературы**

1. Самаль Д.И., Колонов В.В. Структурная и функциональная организация ЭВМ. Повышение производительности центрального процессора. - Минск БГУИР, 2010.
2. Конспект лекций по курсу «Структурная и функциональная организация ЭВМ». - 2012.
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. 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.

1. Глецевич И.И., Прытков В.А., Отвагин А.В. Методические указания по дипломному проектированию для студентов специальности 40 02 01 «Вычислительные машины, системы и сети». – Минск БГУИР, 2009, 99 с.