СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Микро-ЭВМ является устройством, предназначенным для обработки данных. Данные хранятся в специальных устройствах памяти, как внутри микро-ЭВМ – регистрах, так и на внешних устройствах. В данном курсовом проекте внешними устройствами будут выступать блоки RAM, ROM. Исходя из определения Гарвардской архитектуры команды и данные разделены между собой, поэтому блоки данных и команд будут иметь различные адресные пространства.

Кроме устройства управления и блока памяти будут также реализованы стек и АЛУ. Разработанный блок АЛУ будет обеспечивать выполнение 4 команд: INC, XOR, OR, ROL. Команды АЛУ будут реализованы для двух типов адресации: прямой регистровой и косвенной регистровой.

Для разработки курсового проекта использовалась программа Altera Quartus 9.1, которая позволяет составлять схемы различной сложности на логических элементах, эмуляции работы системы по средствам построения временных диаграмм.

1 РАЗРАБОТКА ОБЩЕЙ СТРУКТУРЫ МИКРО-ЭВМ

Структурная схема устройства представлена в приложении Б. Разработка микро-ЭВМ на ПЛИС приведены на схеме. Структурная схема состоит из:

- ПЗУ;
- ОЗУ;
- Устройство управления;
- Арифметико-логическое устройство;
- Стек;
- Регистры общего назначения.

Устройство управления взаимодействует со всеми блоками микро-ЭВМ. В ПЗУ отправляется адрес, по которому нужно считать данные и сигнал чтения. В ОЗУ отправляется адрес, по которому нужно считать или записать данные и сигнал чтения или записи. Из ПЗУ в УУ поступают слова для реализации команд. РОН принимает из УУ сигналы управления и адрес регистра, который нужно прочитать или в который нужно записать данные. РОН взаимодействует с ОЗУ при командах mov, со стеком при командах push, pop, с АЛУ при командах АЛУ. При реализации косвенной адресации РОН отправляет данные в УУ, где они преобразуются в адрес для ОЗУ. В АЛУ данные для команд поступают из РОН и ОЗУ, результат выполнения команд записывается в ОЗУ при косвенной регистровой адресации и РОН при прямой регистровой адресации.

1.1 Функциональный состав микро-ЭВМ

Главным блоком микро-ЭВМ является устройство управления. Данный блок вырабатывает сигналы управления блоком памяти, регистрами общего назначения, стеком и АЛУ. Считывание команд осуществляется с помощью подачи на адресный вход блока памяти значения счетчика команд. Значение этого счетчика инкрементируется с каждым считанным значением либо устанавливается определенное значение, если выполняется команда JMP или JS. Декодирование команды происходит путем считывания четырех старших разрядов первого считанного байта команды. Устройство управление также отвечает за получение всех необходимых для выполнения команды данных.

Блок памяти содержит в себе ПЗУ и ОЗУ. ПЗУ используется для хранения команд, а ОЗУ для данных, необходимых для выполнения команды. По варианту задания шина данных равна 8 бит, следовательно, и размер слов, хранимых в блоке памяти, равен 8 бит.

Арифметико-логическое устройство используется для выполнения арифметических, логических и сдвиговых команд. Выполняемая команда определяется заданным значением операции (0 – INC, 1 – XOR, 2 – OR, 3 – ROL). На выходную шину подается результат выполненной команды. Значения, которые участвуют в операциях АЛУ, записываются в регистры

АЛУ перед проводимой операцией.

Блок регистров общего назначения состоит из 12 регистров, которые используются для хранения данных.

Стек состоит из 13 регистров. По заданному варианту стек растет вниз, что означает уменьшение значения указателя при занесении данных с стек. Ограничение записи данных в стек организованно с помощью счетчика, который инкрементирует свое значение при команде PUSH или декрементирует при команде POP.

1.2 Разработка системы команд

В разрабатываемом устройстве применяется строго фиксированная длина команды, что увеличивает количество памяти, необходимое для хранения кода программы, но повышает быстродействие системы, т.к. не требуется вычисление адреса памяти, по которому хранится команда, и упрощает представление архитектуры системы команд. Архитектура системы команд (далее АСК) представленная в таблице 1.1.

Таблица 1.1 Структура команды микро-ЭВМ

Первое слово команды	
7-4	3-0
Код операции	Адрес регистра 1
Второе слово команды	
7-0	
Адрес памяти	
Третье слово команды	
7-4	3-0
Адрес памяти	Адрес регистра 2

В команде mov adr reg используется три слова команды, старшие 4 бита первого слова содержат код операции, биты с 3 по 0 содержат адрес регистра, который является источником данных, 8 бит второго слова и старшие 4 бита третьего слова содержат адрес ячейки памяти, которая является приемником данных. 4 младших бита третьего слова содержат безразличные биты.

В командах xor reg1 reg2, or reg1 reg2, rol reg1 reg2 используется три слова команды, старшие 4 бита первого слова содержат код операции, биты с 3 по 0 содержат адрес регистра, который является источником данных первого операнда АЛУ и приемником результата, 4 младших бита третьего слова содержат адрес регистра, который является источником данных второго операнда АЛУ. Второе слово и 4 старших бита третьего слова содержат безразличные биты.

В команде mov reg, adr используется три слова команды, старшие 4 бита первого слова содержат код операции, биты с 3 по 0 содержат адрес регистра, который является приемником данных, 8 бит второго слова и старшие 4 бита

третьего слова содержат адрес ячейки памяти, которая является источником данных. 4 младших бита третьего слова содержат безразличные биты.

В команде inc reg используется одно слово команды, старшие 4 бита первого слова содержат код операции, биты с 3 по 0 содержат адрес регистра, который является источником данных операнда АЛУ и приемником результата. Второе и третье слова содержат безразличные биты.

В команде push reg используется первое слова команды, старшие 4 бита первого слова содержат код операции, младшие 4 бита содержат адрес регистра, который является источником данных. Второе и третье слова содержат безразличные биты.

В команде pop reg используется первое слова команды, старшие 4 бита первого слова содержат код операции, младшие 4 бита содержат адрес регистра, который является приемником данных. Второе и третье слова содержат безразличные биты.

В команде jmp adr и js adr используется три слова команды, старшие 4 бита первого слова содержат код операции. 8 бит второго слова и старшие 4 бита третьего слова содержат адрес, в который будет произведен прыжок. 4 младших бита третьего слова содержат безразличные биты.

В команде mov [reg1] reg2 используется два слова команды, старшие 4 бита первого слова содержат код операции, младшие 4 бита содержат адрес регистра, который хранит адреса ячейки ОЗУ, которая является приемником данных. 4 младших бита третьего слова содержат адрес регистра, который является источником данных. Второе слово и 4 старших бита третьего слова содержат безразличные биты.

В командах xor [reg1] reg2, or [reg1] reg2, rol [reg1] reg2 используется два слова команды, старшие 4 бита первого слова содержат код операции, младшие 4 бита содержат адрес регистра, который хранит адрес ячейки ОЗУ, которая является источников данных первого операнда и приемником данных. 4 младших бита третьего слова содержат адрес регистра, который является источником данных второго операнда. Второе слово и 4 старших бита третьего слова содержат безразличные биты.

В команде inc [reg] используется одно слово команды, старшие 4 бита первого слова содержат код операции, младшие 4 бита содержат адрес регистра, который хранит адрес ячейки ОЗУ, которая является источников данных операнда и приемником данных. Второе и третье слова содержат безразличные биты.

В команде hlt 4 старшие бита первого слова содержат код операции, остальные биты безразличные.

Таким образом, система команд имеет следующий вид (см. таблицу 1.2).

Таблица 1.2 Система команд микро-ЭВМ

Мнемоническая запись	Слово 1		Слово 2	Слово 3	
MOV(M-R)	7-4	3-0	7-0	7-4	3-0

	0000	Adr reg	Adr ram		x
MOV(R-M)	7-4	3-0	7-0	7-4	3-0
	0001	Adr reg	Adr ram		x
PUSH reg	7-4	3-0	7-0	7-4	3-0
	0010	Adr reg	x	x	
POP reg	7-4	3-0	7-0	7-0	
	0011	Adr reg	x	x	
JMP operand	7-4	3-0	7-0	7-4	3-0
	0100	x	operand		x
JS operand	7-4	3-0	7-0	7-4	3-0
	0101	x	operand		x
INC reg	7-4	3-0	7-0	7-4	3-0
	0110	Adr reg	x		x
XOR reg1, reg2	7-4	3-0	7-0	7-4	3-0
	0111	Adr reg1	x		Adr reg2
OR reg1, reg2	7-4	3-0	7-0	7-4	3-0
	1000	Adr reg1	x		Adr reg2
ROL reg1, reg2	7-4	3-0	7-0	7-4	3-0
	1001	Adr reg1	x		Adr reg2
INC (косвенная регистровая) [reg]	7-4	3-0	7-0	7-4	3-0
	1010	Adr reg	x	x	x
XOR (косвенная регистровая) [reg1] reg2	7-4	3-0	7-0	7-4	3-0
	1011	Adr reg1	x		Adr reg2
OR (косвенная регистровая) [reg1] reg2	7-4	3-0	7-0	7-4	3-0
	1100	Adr reg 1	x		Adr reg 2
ROL (косвенная регистровая) [reg1] reg2	7-4	3-0	7-0	7-4	3-0
	1101	Adr reg1	x		Adr reg 2
MOV (косвенная регистровая) [reg1] reg2	7-4	3-0	7-0	7-4	3-0
	1110	Adr reg1	x		Adr reg 2
HLT	7-4	3-0	7-0	7-0	
	1111	x	x	x	

1.3. Описание взаимодействия всех блоков микро-ЭВМ при выполнении команд программы

Выполнение команды начинается с ее считывания из блока ПЗУ.

Сразу после считывания инструкция поступает в устройство управления УУ, где первым делом анализируется КОП. В зависимости от КОП, разделяет команду на адреса и операнды. Операнды под собой подразумевают или адрес регистра, или адрес ячейки в памяти, или константу для непосредственной адресации. В зависимости от КОП операнды поступают на входы АЛУ, регистров общего назначения или стека вместе со соответствующими управляющими сигналами.

2 РАЗРАБОТКА ОСНОВНЫХ УСТРОЙСТВ МИКРО-ЭВМ

В данном разделе приводится доскональное описание состава и принципов работы главных блоков разрабатываемой микро-ЭВМ. Общая

функциональная схема устройства представлена в приложении В, а схема центрального процессора отображена в приложении Г.

2.1 Запоминающие устройства

Тип разрабатываемой архитектуры подразумевает разбиение адресного пространства на память команд и данных, что обусловило проектирование отдельного блока памяти для данных, представленного асинхронным ОЗУ, и блока памяти для команд – асинхронным ПЗУ. Организация данных блоков была произведена при помощи стандартных модулей Quartus - lpm_rom и lpm_ram_io.

Считывание данных из блока памяти осуществляется при генерировании соответствующих управляющих сигналов и подаче тактовых импульсов. После подачи на адресный вход одного из блоков памяти адреса ячейки для считывания данных, выставление полученных данных осуществляется на следующем такте после подачи адреса.

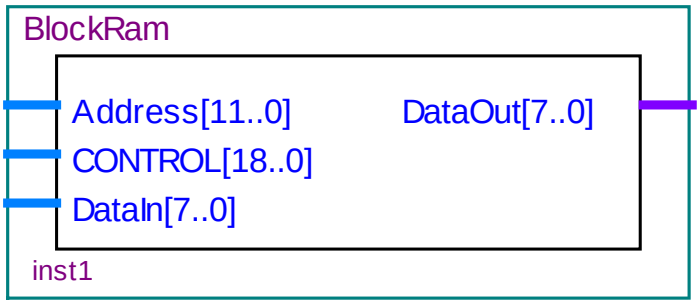

Рисунок 2.1 – Условно-графическое обозначение блока памяти ОЗУ

Блок памяти ОЗУ (рисунок 2.1) имеет следующие входы и выходы:
Входы:
– Address[11..0] – шина адреса данных;
– CONTROL[18..0] – шина управления;
– DataIn[7..0] – входная шина данных.
Выход:
– DataOut[7..0] – выходная шина данных.

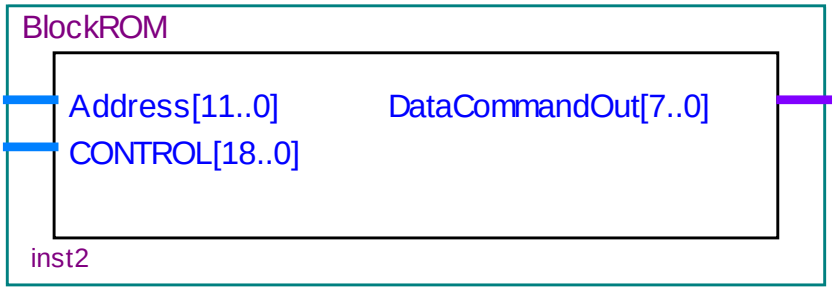

Рисунок 2.2 – Условно-графическое обозначение блока памяти ПЗУ

Блок памяти ПЗУ (рисунок 2.2) имеет следующие входы и выходы:
Входы:
– Address[11..0] – шина адреса команд;
– CONTROL[18..0] – шина управления.
Выходы:
– DataCommandOut[7..0] – выходная шина команд.
Схемы каждого из модулей памяти представлены в приложениях Д и Е для ОЗУ и ПЗУ соответственно.

2.2 Устройство управления

Устройство управления используется для выборки, декодирования считываемой инструкции, разбиения ее на составные части. Производится разбиение выбранной команды на различные этапы выполнения, посредством тактирования и выставления набора управляющих сигналов, определяющего дальнейшую последовательность и взаимодействие функциональных блоков.

Определение порядка выполнения инструкции осуществляется при помощи шины управления, передающей команду и адреса операндов во время реализации инструкции. Подача синхроимпульса обуславливает тактирование управляющего устройства. На каждом такте осуществляется выставление определенного управляющего сигнала. Данные выходные сигналы управляют работой основных модулей, проектируемой микро-ЭВМ, а также помогают избежать коллизий при доступе к шинам адреса и данных.

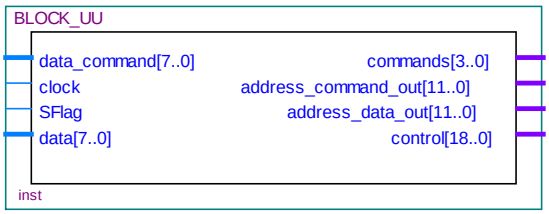

Рисунок 2.3 – Условно-графическое обозначение устройства управления

Ниже представлены входные и выходные пины, характерные для устройства управления (рисунок 2.3).

Входные:

- data_command[7..0] – шина данных команд;
- clock – тактовый сигнал;
- SFlag – значение Sign флага;
- data[7..0] – шина данных.

Выходные:

- commands[3..0] – команды;
- address_command_out[11..0] – шина адреса команд;
- address_data_out[11..0] – шина адреса данных;
- control[18..0] – шина управления.

Блок центрального процессора состоит из блоков АЛУ, РОН и стека. К модулям управления относятся - декодер инструкций из ПЗУ, блок выборки команды, блок записи адреса и блок записи данных. Функциональная схема блока управления, отображена в приложении Ж.

Блок работы с управляющими сигналами представляет собой набор отдельных модулей, каждый из которых отвечает за генерацию определенного импульса, инициирующего работу одного из внутренних устройств ЦП. К таким модулям относятся: блок записи в память, блок записи в регистр, блок чтения из памяти, блок чтения из регистра. На входы подаются значения каждой из команд, которые в процессе своей реализации работают с одним из внутренних устройств, а также определенный тактовый сигнал, в течении которого выполняется обращение к одному из функциональных блоков ЦП.

В блоке устройства управления генерируется шина управления, которая содержит 19 бит:

- 0 бит – тактовый сигнал;
- 1 бит – чтение из ПЗУ;
- 2 бит – чтение из ОЗУ;
- 3 бит – запись в ОЗУ;
- 4 бит – запись первого операнда для операций АЛУ;
- 5 бит – запись второго операнда для операций АЛУ;
- 6 бит – сигнал операции INC;
- 7 бит – сигнал операции XOR;
- 8 бит – сигнал операции OR;
- 9 бит – сигнал операции ROL;
- 10 бит – сигнал вывода результата операции АЛУ на шину данных;
- 11 бит – сигнал PUSH;
- 12 бит – сигнал POP;
- 13 бит – сигнал чтения РОН;
- 14 бит – сигнал записи в РОН;
- 15-18 – адрес регистра.

Для выполнения операции, сигнал которой выставляет представленный выше модуль, необходимы данные: адрес, определяющий номер ячейки или РОН с которыми ведется работа, и числовые значения, которые записываются

или считываются из памяти, или данные для выполнения операций в АЛУ. Выставлением необходимых значений на системные шины занимаются блоки записи адреса и записи данных. В зависимости от текущего входного тактового сигнала и типа операции, на шину, в случае блока записи адресов, выставляется часть инструкции, полученная при ее декодировании, а блок работы с данными выставляет на канал передачи, поступившие в центральный процессор в текущий момент, числовые значения. На входы данных модулей поступают определенные тактовые импульсы, линии, используемых команд, а также значения, которые передаются по системным шинам.

Работоспособность декодера команд полностью обусловлена сигналами управляющего устройства. Во время потактового считывания команды из ПЗУ устройство управления генерирует сигналы в блок декодирования. По этим импульсам производится заполнение внутренних регистров декодера, в зависимости от считанной инструкции. После завершения чтения и заполнения внутренних регистров декодера, производится полное считывание команды из него, посредством сигнала set_operands. Данные передаются по шине управления в течении полного периода выполнения команды.

Блок выборки команды, определяющий адрес инструкции представлен на рисунке 2.4.

Функциональная схема блока выборки команды показана в приложении З.

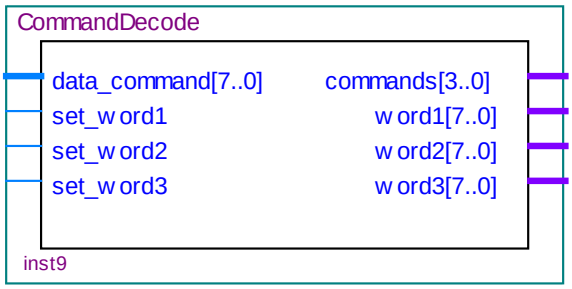

Рисунок 2.4 – Условно-графическое обозначение блока выборки команды

Входы и выходы блока выборки команды представлены ниже.

Входы:

– data_command[7..0] – шина данных команд;

– set_word1, set_word2, set_word3 – сигнал чтения первого, второго, третьего слова соответственно;

Выходы:

– commands[3..0] – текущая команда;

– word1[7..0], word2[7..0], word3[7..0] – значения первого, второго и третьего слов соответственно.

Выборка следующей инструкции осуществляется после завершения текущей команды. После ее выполнения генерируется управляющий сигнал, который позволяет через входную адресную шину поступить в блок выборки

команды начальному адресу реализуемой инструкции. При выполнении команд условного или безусловного переходов (JB, JMP) выходное адресное значение изменяется в зависимости от операндов данных команд.

В устройстве управления был внедрен блок операндов, представленный на рисунке 2.5.

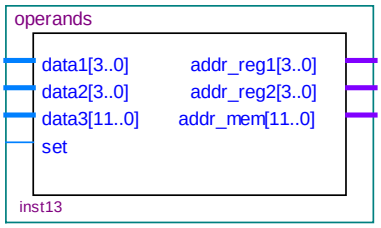

Рисунок 2.5 – Условно-графическое обозначение блока операндов

Входы и выходы данного блока представлены ниже.

Входы:

– data1[3..0] – шина данных для первого операнда;
– data2[3..0] – шина данных для второго операнда;
– data3[11..0] – шина данных для третьего операнда;
– set – сигнал обновления операндов;

Выходы:

– addr_reg1[3..0] – значение адреса первого регистра;
– addr_reg2[3..0] – значение адреса второго регистра;
– addr_mem[11..0] – значение адреса памяти.

Блок операндов по сигналу set записывает данные из слов команды в блоки. В дальнейшем данные из этих блоков будут служить адресом для регистров, ОЗУ или ПЗУ. Реализация данного блока представлена на рисунке 2.6.

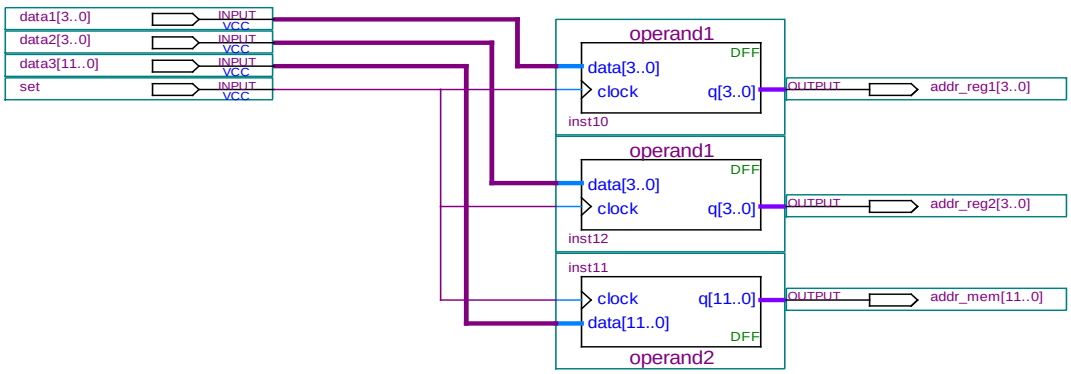

Рис. 2.6 – Реализация блока операндов

Также в УУ присутствует счетчик адреса ПЗУ, представленный на рисунке 2.7. Реализация данного блока представлена на рисунке 2.8.

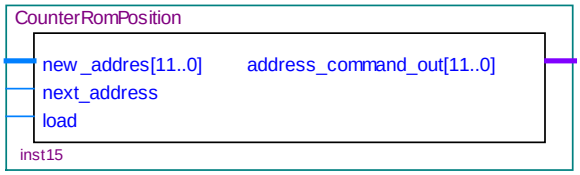

Рисунок 2.7 – Условно-графическое обозначение блока счетчика адреса ПЗУ

Входы и выходы данного блока представлены ниже.

Входы:

– new_address[11..0] – значение адреса, который будет загружен в счетчик при операциях JMP и JS.

– next_address – сигнал увеличения адреса ПЗУ;

– load – сигнал загрузки нового адреса в ПЗУ.

Выход:

– address_command_out[11..0] – шина адреса ПЗУ.

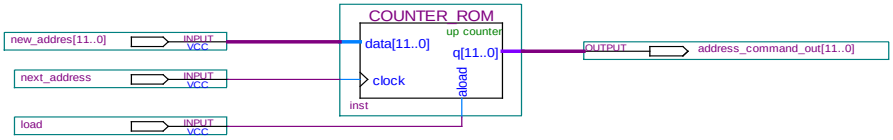

Рисунок 2.8 – Реализация счетчика адреса ПЗУ

Ключевую роль в микро-ЭВМ играет шина управления. Генерируется шина управления в блоке генерации управляющих сигналов. Условно-графическое обозначение блока представлено на рисунке 2.9. Реализация данного блока представлена в приложении И.

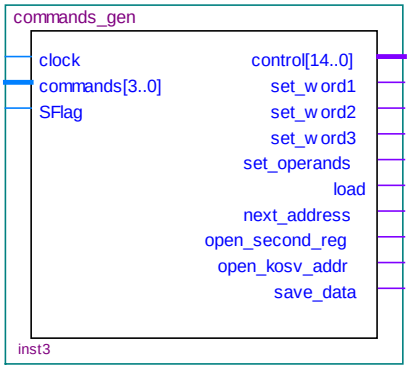

Рисунок 2.9 – Условно-графическое обозначение блока генерации управляющих сигналов

Входы и выходы данного блока представлены ниже.

Входы:

– clk – тактовый сигнал;

– commands[3..0] – код операции;
– SFlag – значение флага знака.
Выходы:
– control[14..0] – шина управления;
– set_word1 – сигнал записи первого слова;
– set_word2 – сигнал записи второго слова;
– set_word3 – сигнал записи третьего слова;
– set_operands – сигнал записи значений адресов;
– load – сигнал загрузки нового адреса в ПЗУ;
– next_address – сигнал увеличения адреса ПЗУ;
– open_kosv_addr – сигнал подачи косвенного адреса на гину адреса ОЗУ;
– save_data – сигнал сохранения данных из регистра для генерации нового адреса ОЗУ.

2.3 Стек

Данный блок является дополнительным типом памяти, позволяющим считывать записанные в него данные в обратном для операции записи порядке. В разрабатываемой микро-ЭВМ, в соответствии с вариантом, размерность стека составляет 13, а алгоритм роста – вниз. Также стек содержит указатель на вершину, изменяющуюся при записи данных в регистры данного блока.

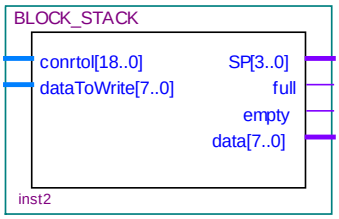

Рисунок 2.10 – Условно-графическое обозначение блока стека

Блок стека (рисунок 2.5) имеет следующие входы и выходы:
Входы:
– control[18..0] – шина управления;
– dataRoWrite[7..0] – шина данных.
Выходы:
– SP – значение указателя вершины стека;
– full – сигнал заполненного стека;
– empty – сигнал пустого стека;
– data[7..0] – шина данных.
Блок стека работает в зависимости от входных управляющих сигналов – POP, PUSH. При поступлении сигнала PUSH, производится запись данных в один из регистров стека, на который указывает значение вершины стека.

После завершения записи данных значение вершины декрементируется, что позволяет при повторной записи поместить данные другой регистр стека. При поступлении сигнала POP, что соответствует сигналу для чтения данных из стека, числовые значения считываются из регистра, номер которого соответствует величине флага вершины. Схема стека представлена в приложении К.

2.4 Арифметико-логическое устройство

Блок АЛУ выполняет арифметические, логические операции и операции сдвига, определенные в задании к курсовому проекту. Подача сигнала выполнения одной из операции инициирует работу данного функционального блока. Данные, необходимые для выполнения команд АЛУ, передаются последовательно по шине и сохраняются во внутренних регистрах арифметико-логического устройства. Последний такт выполнения команды реализует необходимую операцию над данными, хранящимися в памяти АЛУ.

Схема арифметико-логического устройства представлена в приложении Л.

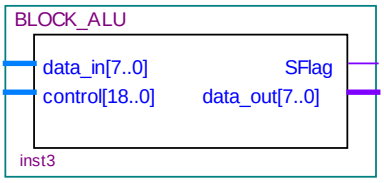

Рисунок 2.11 – Условно-графическое обозначение блока АЛУ

Характеристика входов и выходов блока АЛУ представлена ниже:

Входы:

– data_in[7..0] – шина данных;
– control[18..0] – шина управления.

Выходы:

– SFlag – значение Sign флага;
– data_out[7..0] – шина данных.

Числовые значения записываются в регистры под управлением 4 и 5 сигналов шины управления. В зависимости от вида реализуемой операции, сохраняемые данные применяются далее для различных целей. Например, при выполнении операции логическое ИЛИ, в регистры АЛУ сохраняются значения двух чисел, а при реализации операции сдвига, элементы памяти будут содержать значение числа и количества битов для сдвига соответственно.

Операция INC – инкрементация, пришедший операнд увеличивает свой значение на 1.

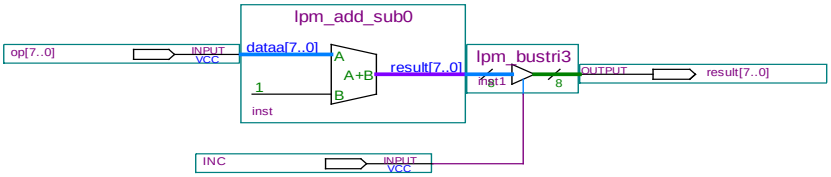

Рисунок 2.12 – Реализация блока INC

Операция XOR – логическое исключающее ИЛИ. Таблица истинности представлена на рисунке 2.13.

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Рисунок 2.13 – Таблица истинности XOR

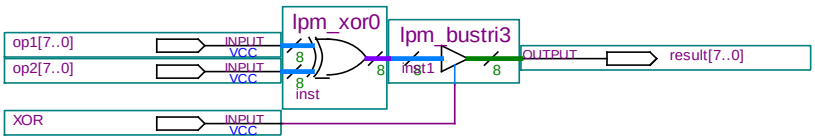

Рисунок 2.14 – Реализация блока XOR

Операция OR – логическое ИЛИ. Таблица истинности представлена на рисунке 2.15.

a	b	$a \vee b$
0	0	0
0	1	1
1	0	1
1	1	1

Рисунок 2.15 – Таблица истинности OR

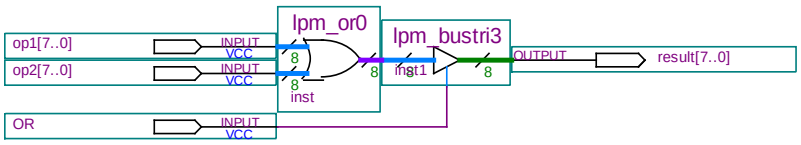

Рисунок 2.16 – Реализация блока OR

Операция ROL – циклический сдвиг влево.

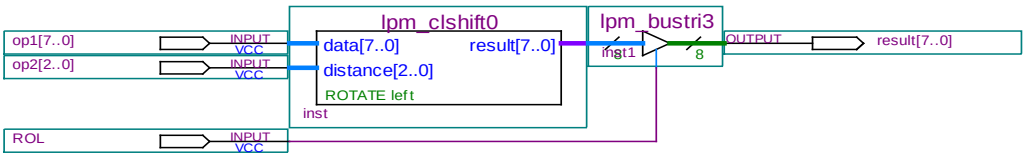

Рисунок 2.17 – Реализация блока ROL

2.5 Регистры общего назначения

Блок регистров общего назначения, используемый для временного хранения числовых данных, представляет собой набор элементов памяти общей разрядностью 12. Объем каждого отдельного РОН эквивалентен разрядности шины, по которой в этот модуль записываются или выводятся числовые значения - 8 бит. Работа с регистрами осуществляется посредством управляющих сигналов, генерируемых устройством управления.

Условно-графическое обозначение арбитра представлено на рисунке 2.9.

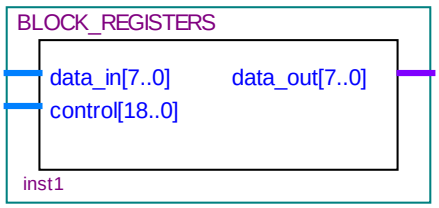

Рис. 2.18 – Условно-графическое обозначение блока регистров

Входные пины:

– data_in[7..0] – шина данных;

– control[18..0] – шина управления.

Выходной пин:

– data_out[7..0] – шина данных.

При поступлении сигнала управления, декодер по последним четырем битам шины управления определяет номер регистра, задействованного в выполнении операции с данными. Числовые значения с входа данных записываются либо считываются на выход данных с приходом тактирующего импульса.

Функциональная схема блока регистров представлена в приложении М.

3 ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ

3.1 Моделирование памяти

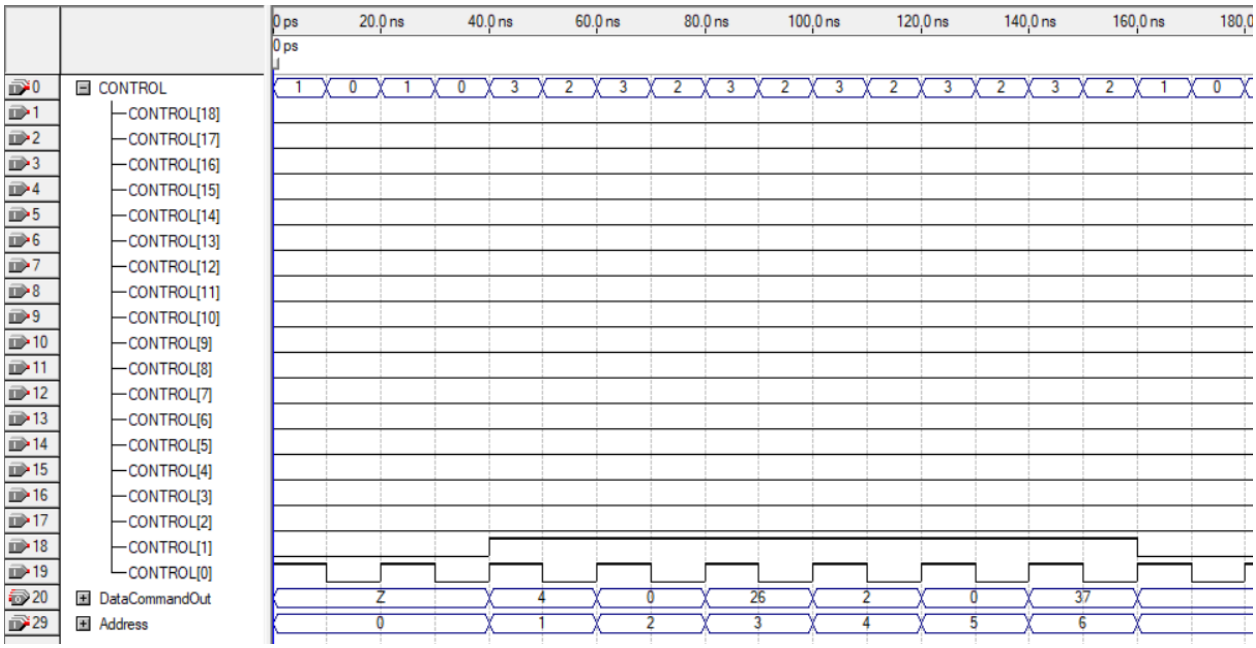

Рисунок 3.1 – Моделирование блока памяти ПЗУ

На рисунке 3.1 отображено моделирование блока памяти ПЗУ. С 40 по 160 ns происходит чтение из ПЗУ. Выходные данные поступают на шину DataCommandOut. На рисунке 3.2 представлен дамп памяти ПЗУ.

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	5	4	0	26	2	0	37	0
8	0	59	0	0	64	16	0	0
16	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0

Рисунок 3.2 – Дамп памяти ПЗУ

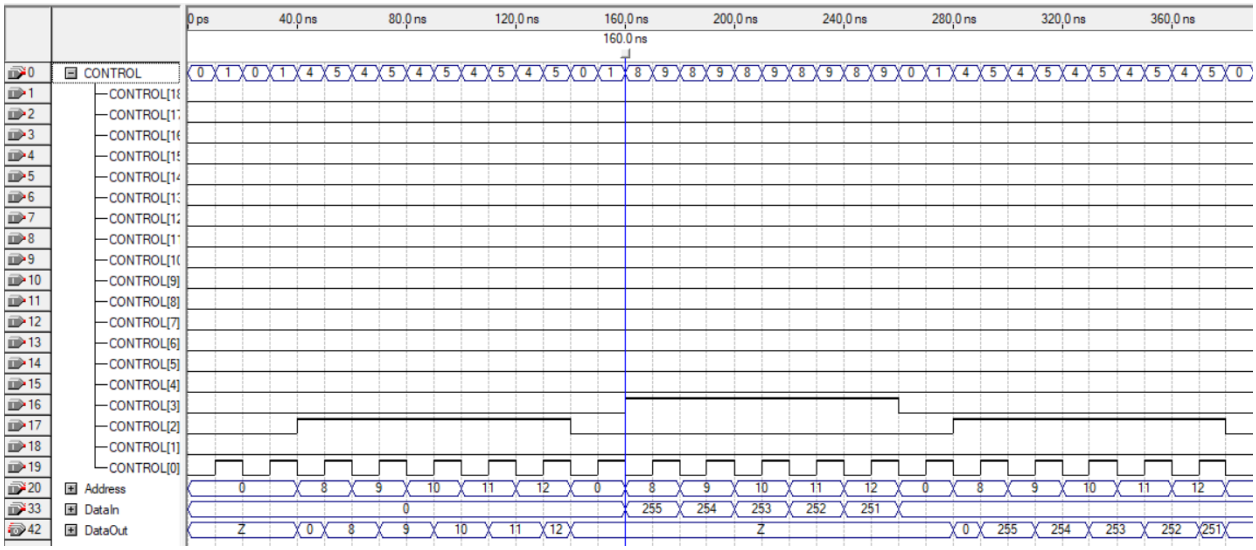

Рисунок 3.3 – Моделирование блока памяти ОЗУ

На рисунке 3.3 отображено моделирование блока памяти ОЗУ. С 40 по 140 ns и с 280 по 380 ns происходит чтение из ОЗУ. С 160 по 260 ns происходит запись в ОЗУ. Выходные данные поступают на шину DataOut. На рисунках 3.4 и 3.5 представлены дампы памяти до и после моделирования.

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	0	1	2	3	4	5	6	7
8	8	9	10	11	12	13	14	15
16	16	17	18	19	20	21	22	23
24	24	25	26	27	28	29	30	31
32	32	33	34	35	36	37	38	39
40	40	41	42	43	44	45	46	47
48	48	49	50	51	52	53	54	55
56	56	57	58	59	60	61	62	63
64	64	65	66	67	68	69	70	71
72	72	73	74	75	76	77	78	79
80	80	81	82	83	84	85	86	87
88	88	89	90	91	92	93	94	95
96	96	97	98	99	100	101	102	103
104	104	105	106	107	108	109	110	111
112	112	113	114	115	116	117	118	119
120	120	121	122	123	124	125	126	127
128	128	129	130	131	132	133	134	135
136	136	137	138	139	140	141	142	143
144	144	145	146	147	148	149	150	151
152	152	153	154	155	156	157	158	159
160	160	161	162	163	164	165	166	167

Рисунок 3.4 – Дамп памяти до моделирования

|BlockRam\|lpm_ram_io:inst\|altram:sram\|altsyncram:ram_bloc

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	0	1	2	3	4	5	6	7
8	255	254	253	252	251	13	14	15
16	16	17	18	19	20	21	22	23
24	24	25	26	27	28	29	30	31
32	32	33	34	35	36	37	38	39
40	40	41	42	43	44	45	46	47
48	48	49	50	51	52	53	54	55
56	56	57	58	59	60	61	62	63
64	64	65	66	67	68	69	70	71
72	72	73	74	75	76	77	78	79
80	80	81	82	83	84	85	86	87
88	88	89	90	91	92	93	94	95
96	96	97	98	99	100	101	102	103
104	104	105	106	107	108	109	110	111
112	112	113	114	115	116	117	118	119
120	120	121	122	123	124	125	126	127
128	128	129	130	131	132	133	134	135
136	136	137	138	139	140	141	142	143
144	144	145	146	147	148	149	150	151
152	152	153	154	155	156	157	158	159
160	160	161	162	163	164	165	166	167

Рисунок 3.5 – Дамп памяти после моделирования

3.2 Моделирование стека

Моделирование стека представлено на рисунке 3.6.

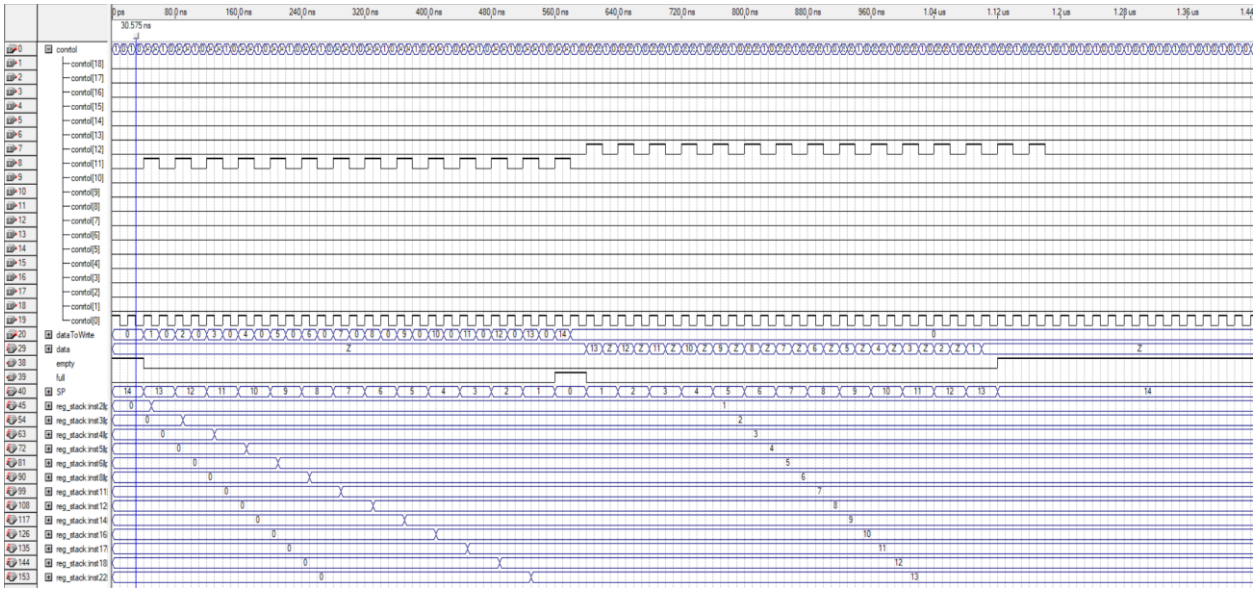

Рисунок 3.6 – Моделирование стека

Моделирование блока стека отображает выполнение 14 операций PUSH реализуемых последовательно и записывающих значения от 1 до 14. При 14 операции PUSH происходит переполнение стека, поднимается сигнал full. Запись данных в стек происходит сверху вниз, обусловлена условием организации роста стека вниз, то есть, по мере записи данных в регистры стека, флаг, указывающий на вершину будет сдвигаться вниз по стеку от 13 к 1 регистру. При выполнении операции POP данные остаются в своих регистрах, однако флаг вершины изменяет свое положение.

3.3 Моделирование блока регистров

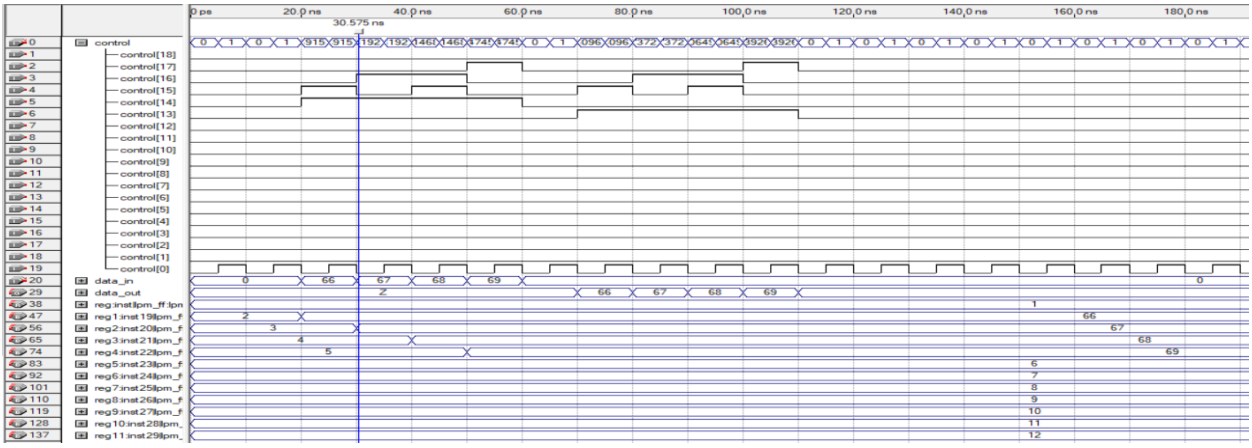

Рисунок 3.7 – Моделирование блока РОН

На рисунке 3.7 представлена диаграмма моделирования работы блока регистров общего назначения. На старте программы происходит автоматическое заполнение регистров начальными данными для дальнейшей удобной работы с ними. При поступлении 13 сигнала шины управления происходит чтение, при поступлении 14 – запись в регистры. Адрес для регистров передается в четырех старших битах шины управления.

3.4 Моделирование блока АЛУ

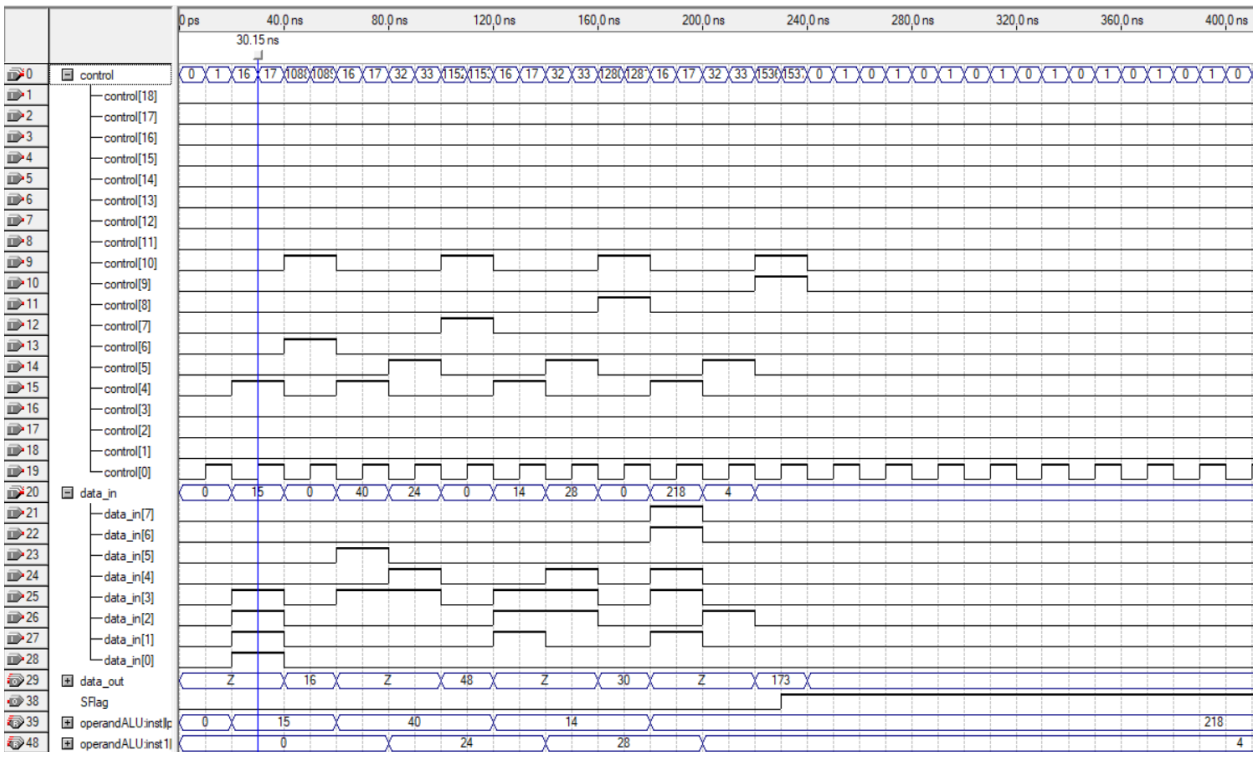

Рисунок 3.8 – Моделирование блока АЛУ

Моделирование, представленное на рисунке 3.8 отображает выполнение

блоком арифметико-логического устройства операций INC, XOR, OR, ROL соответственно. При поступлении 4 и 5 сигналов шины управления происходит запись первого и второго операнда соответственно. С 6 по 9 сигналы – сигналы операций АЛУ. При поступлении 10 сигнала происходит вывод результата на шину данных.

3.5 Общее моделирование системы

Общее моделирование системы представляет собой анализ реализации микрокода программы, описывающего порядок выполнения инструкций разработанной микро-ЭВМ. Микропрограмма для общего моделирования системы с представлением реализуемых команд в символьном и бинарном виде представлена в таблице 3.1.

Таблица 3.1 – Микрокод программы.

Номер ячейки ПЗУ	Запись на языке ассемблера	Запись в бинарном виде	Запись в шестнадцатеричном виде
1	2	3	4
0	Mov adr64 r5	00000101	05
1		00000100	04
2		00000000	00
3	Mov r10 adr128	00011010	1A
4		00001000	08
5		00000000	00
6	Push r5	00100101	25
7		00000000	00
8		00000000	00
9	Pop r11	00111011	3B
10		00000000	00
11		00000000	00
12	Jmp 256	01000000	40
13		00010000	10
14		00000000	00
256	Inc r4	01100100	64
257		00000000	00
258		00000000	00
259	Xor r6 r5	01110110	76
260		00000000	00
261		00000101	05
262	Or r10 r11	10001010	8A
263		00000000	00

Продолжение таблицы 3.1

1	2	3	4
264		00001011	0B
265	Js 128	01010000	50
266		00001000	08
267		00000000	00
128	Rol r2 r5	10010010	92
129		00000000	00
130		00000101	05
131	Inc [r9] (adr10)	10101001	A9
132		00000000	00
133		00000000	00
134	Xor [r8] (adr9) r3	10111000	B8
135		00000000	00
136		00000011	03
137	Or [r7] (adr8) r7	11000111	C7
138		00000000	00
139		00000111	07
140	Rol [r4] (adr6) r10	11010100	D4
141		00000000	00
142		00001010	0A
143	Mov [r1] (adr2) reg10	11100001	E1
144		00000000	00
145		00001001	09
146	Hlt	11110000	F0
147		00000000	00
148		00000000	00

Общее моделирование программы представлено на рисунках 3.9 – 3.11.

Рисунок 3.9 – Общее моделирование системы: команды mov adr64 r5, mov r10 adr128, push r5, pop r11, jmp256

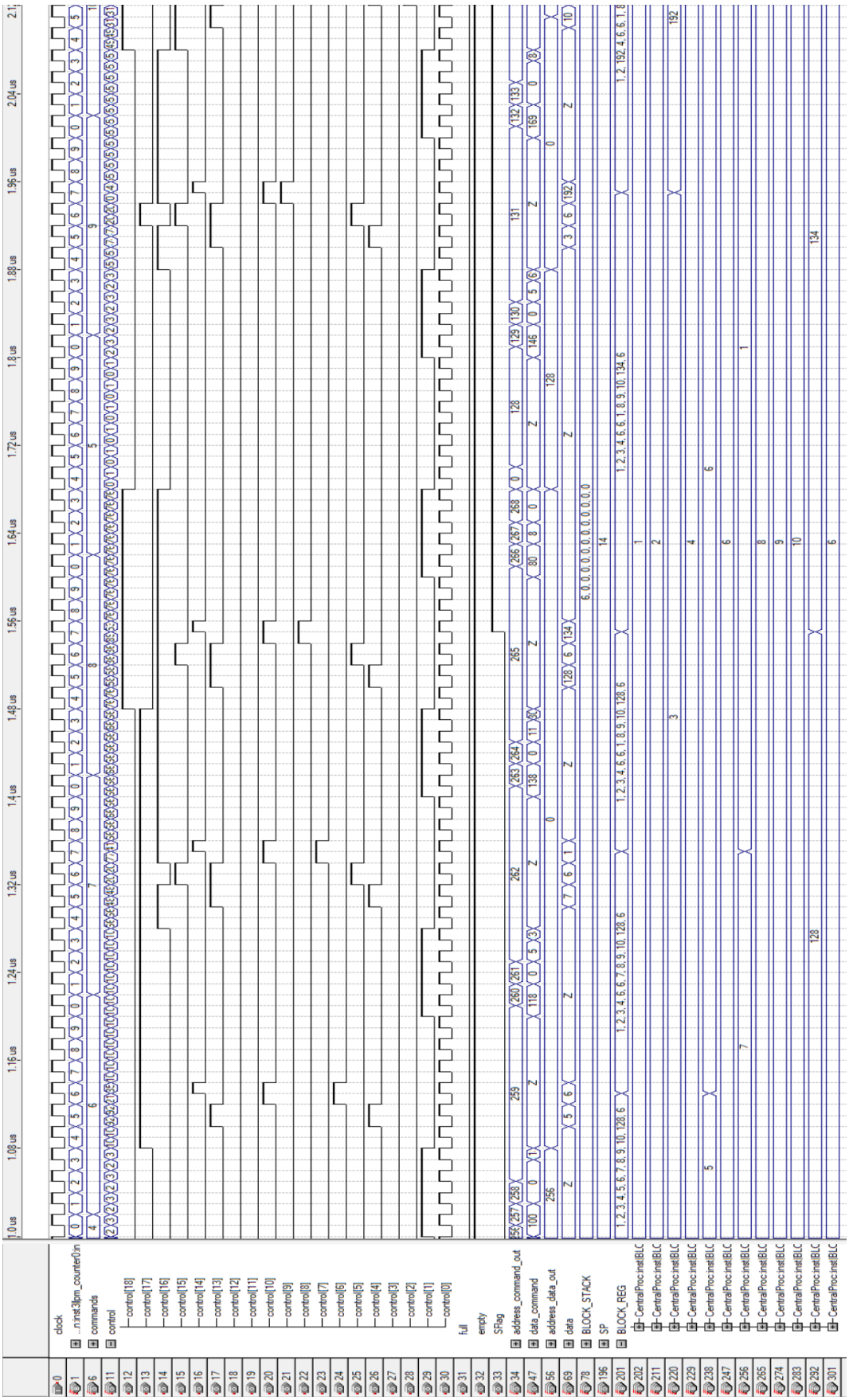

Рисунок 3.10 – Общее моделирование системы: команды inc r4, xor r6 r5, or r10 r11, js 128, rol r2 r5

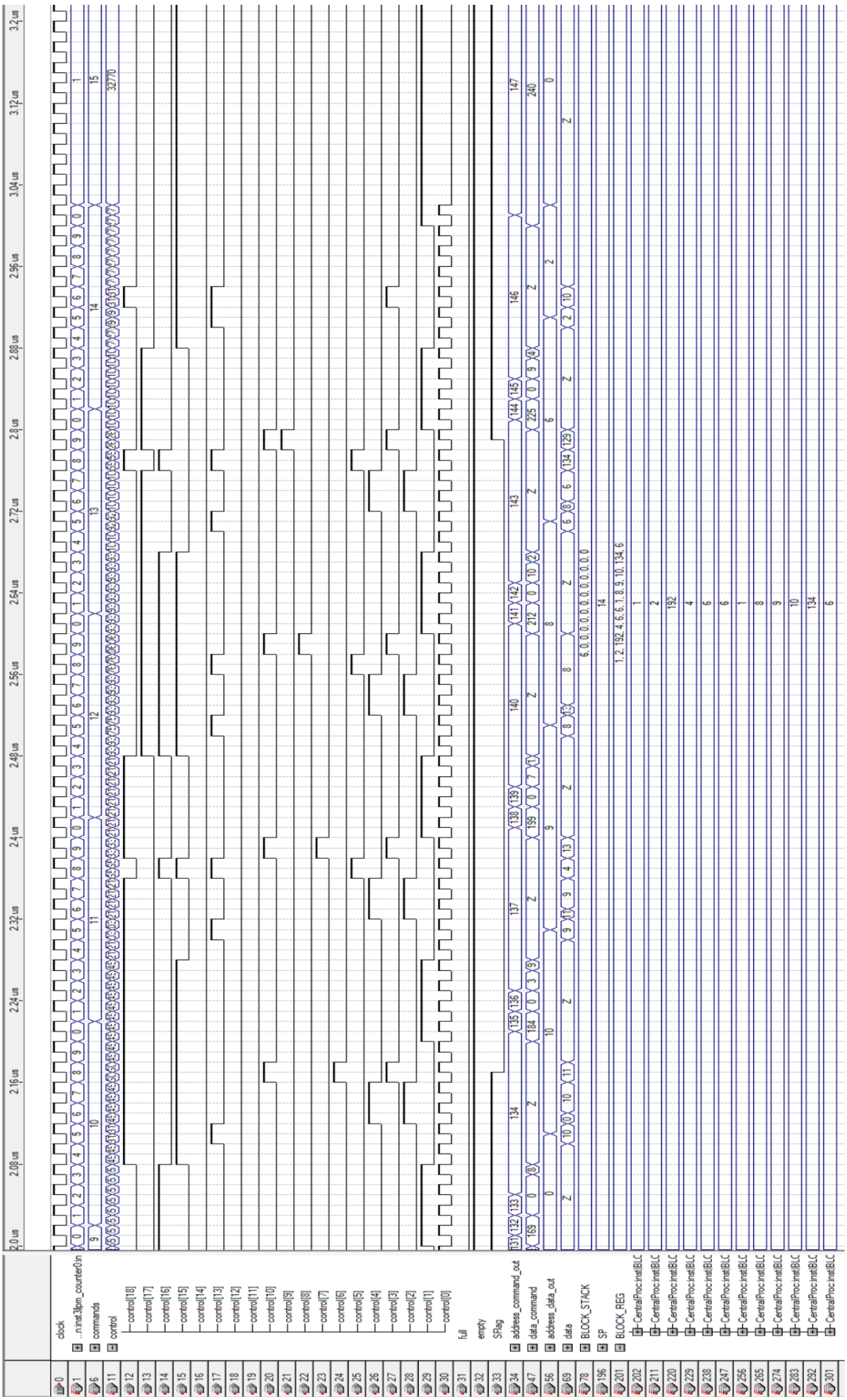

Рисунок 3.11 – Общее моделирование системы: команды inc [r9] (adr10), xor [r8] (adr9) r3, or [r7] (adr8) r7, rol [r4] (adr6) r10, mov [r1] (adr2) r9, hlt

3.6 Дампы памяти

Дамп памяти команд представлен на рисунке 3.12. Он содержит набор инструкций, формирующих собой микрокод разрабатываемого ЭВМ.

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	00000101	00000100	00000000	00011010	00001000	00000000	00100101	00000000
8	00000000	00111011	00000000	00000000	01000000	00010000	00000000	00000000
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
64	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
72	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
80	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
88	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
96	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
104	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
112	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
120	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
128	10010010	00000000	00000101	10101001	00000000	00000000	10111000	00000000
136	00000011	11000111	00000000	00000111	11010100	00000000	00001010	11100001
144	00000000	00001001	11110000	00000000	00000000	00000000	00000000	00000000
152	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
160	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
168	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
176	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
184	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
192	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
200	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
208	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
216	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
224	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
232	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
240	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
248	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
256	01100100	00000000	00000000	01110110	00000000	00000101	10001010	00000000
264	00001011	01010000	00001000	00000000	00000000	00000000	00000000	00000000
272	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000

Рисунок 3.12 – Дамп памяти команд

Исходный дамп памяти данных представлен на рисунке 3.13:

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	0	1	2	3	4	5	6	7
8	8	9	10	11	12	13	14	15
16	16	17	18	19	20	21	22	23
24	24	25	26	27	28	29	30	31
32	32	33	34	35	36	37	38	39
40	40	41	42	43	44	45	46	47
48	48	49	50	51	52	53	54	55
56	56	57	58	59	60	61	62	63
64	64	65	66	67	68	69	70	71
72	72	73	74	75	76	77	78	79
80	80	81	82	83	84	85	86	87
88	88	89	90	91	92	93	94	95
96	96	97	98	99	100	101	102	103
104	104	105	106	107	108	109	110	111
112	112	113	114	115	116	117	118	119
120	120	121	122	123	124	125	126	127
128	128	129	130	131	132	133	134	135
136	136	137	138	139	140	141	142	143
144	144	145	146	147	148	149	150	151
152	152	153	154	155	156	157	158	159
160	160	161	162	163	164	165	166	167
168	168	169	170	171	172	173	174	175
176	176	177	178	179	180	181	182	183
184	184	185	186	187	188	189	190	191
192	192	193	194	195	196	197	198	199
200	200	201	202	203	204	205	206	207
208	208	209	210	211	212	213	214	215
216	216	217	218	219	220	221	222	223
224	224	225	226	227	228	229	230	231
232	232	233	234	235	236	237	238	239
240	240	241	242	243	244	245	246	247
248	248	249	250	251	252	253	254	255
256	255	254	253	252	251	250	249	248
264	247	246	245	244	243	242	241	240

Рисунок 3.13 – Исходный дамп памяти данных

Дамп памяти данных после моделирования системы представлена на рисунке 3.14.

|main|BlockRam:inst1|lpm_ram_io:inst|altram:sram|altsyncram

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	0	1	10	3	4	5	129	7
8	8	13	11	11	12	13	14	15
16	16	17	18	19	20	21	22	23
24	24	25	26	27	28	29	30	31
32	32	33	34	35	36	37	38	39
40	40	41	42	43	44	45	46	47
48	48	49	50	51	52	53	54	55
56	56	57	58	59	60	61	62	63
64	6	65	66	67	68	69	70	71
72	72	73	74	75	76	77	78	79
80	80	81	82	83	84	85	86	87
88	88	89	90	91	92	93	94	95
96	96	97	98	99	100	101	102	103
104	104	105	106	107	108	109	110	111
112	112	113	114	115	116	117	118	119
120	120	121	122	123	124	125	126	127
128	128	129	130	131	132	133	134	135
136	136	137	138	139	140	141	142	143
144	144	145	146	147	148	149	150	151
152	152	153	154	155	156	157	158	159
160	160	161	162	163	164	165	166	167
168	168	169	170	171	172	173	174	175
176	176	177	178	179	180	181	182	183
184	184	185	186	187	188	189	190	191
192	192	193	194	195	196	197	198	199
200	200	201	202	203	204	205	206	207
208	208	209	210	211	212	213	214	215
216	216	217	218	219	220	221	222	223
224	224	225	226	227	228	229	230	231
232	232	233	234	235	236	237	238	239
240	240	241	242	243	244	245	246	247
248	248	249	250	251	252	253	254	255
256	255	254	253	252	251	250	249	248

Рисунок 3.14 – Дамп памяти данных после моделирования

ЗАКЛЮЧЕНИЕ

В данном курсовом проекте была реализована микро-ЭВМ с заданными характеристиками на элементной базе ПЛИС в среде автоматизированного проектирования Quartus II 9.1.

В соответствии с заданием на курсовое проектирование была реализована микро-ЭВМ по принципу гарвардской архитектуры.

Разрядность шины адреса разработанной микро-ЭВМ составляет 12 бит, шина данных - 8 бит. Количество регистров общего назначения - 12. ОЗУ является асинхронным, ПЗУ - асинхронным. Арифметическая команда – инкрементация (INC). Логические команды – логическое исключающее ИЛИ (XOR) и логическое “ИЛИ” (OR). Сдвиговые команды – циклический сдвиг влево (ROL). Виды адресации – прямая, косвенная регистровая и непосредственная. Объем стека – 13 регистров. Алгоритм роста стека – вниз. Команды условного и безусловного перехода – JMP, JS.

Реализованная микро-ЭВМ, выполняет полный набор представленных выше команд со всеми типами адресации. Разработанную микро-ЭВМ можно оптимизировать путем сокращения длины команды. Добавление кэш-памяти, конвейера и предсказателя переходов позволит улучшить структуру ЭВМ.

Таким образом, в ходе выполнения курсового проекта были получены знания по организации гарвардской архитектуры, изучены принципы построения и функционирования ЭВМ в целом, а также ее отдельных блоков, разработано устройство управления программного типа, получены знания по кодированию команд и разработке ПЛИС в среде автоматизированного проектирования САПР Quartus II.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] У. Столлингс, Структурная организация и архитектура компьютерных систем/ У. Столлингс. 5-е изд. – М.: "Вильямс ", 2001. Пер. с англ. – 892 с.

[2] Э. Таненбаум, Архитектура компьютерных систем / Э. Таненбаум. 4-е изд. – М.: " ПИТЕР ", 2002. Пер. с англ. – 698 с.

[3] Р. Грушвицкий, Проектирование систем на микросхемах программируемой логики / Р. Грушвицкий. – Спб.: "Питер", 2002. – 608 с.

[4] Е. Угрюмов, Цифровая схемотехника. - М.: " С – Петербург ", 2001 - 518 стр.

[5] Самаль Д.И – Структурная и функциональная организация ЭВМ. - БГУиР, кафедра ЭВМ, 2013.

[6] Зотов_В.Ю. – Проектирование Встраиваемых Микропроцессорных Систем На Основе ПЛИС Фирмы XILINX, 2006.

[7] Шагурин И.И. Процессоры семейства Intel P6. Архитектура, программирование, интерфейс. – М.: "Телеком", 2000. – 248 с.

[8] Рудометов Е. Материнские платы и чипсеты. – СПб.: "Питер", 2000. – 256 с.

[9] Бибило П.Н. Синтез логических схем с использованием языка VHDL. М.: СОЛОН-Р, 2002.

[10] Антонов А. П. Язык описания цифровых устройств AlteraHDL. - М.: РадиоСофт, 2001.

[11] 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.

ПРИЛОЖЕНИЕ А
(обязательное)

Ведомость документов

ПРИЛОЖЕНИЕ Б
(обязательное)

Микро-ЭВМ. Схема структурная

ПРИЛОЖЕНИЕ В
(обязательное)

Общая схема устройства. Схема функциональная

ПРИЛОЖЕНИЕ Г
(обязательное)

Блок центрального процессора. Схема функциональная.

ПРИЛОЖЕНИЕ Д
(обязательное)

Блок ОЗУ. Схема функциональная

ПРИЛОЖЕНИЕ Е
(обязательное)

Блок ПЗУ. Схема функциональная

ПРИЛОЖЕНИЕ Ж
(обязательное)

Блок устройства управления. Схема функциональная

ПРИЛОЖЕНИЕ З
(обязательное)

Блок выборки команды. Схема функциональная

ПРИЛОЖЕНИЕ И
(обязательное)

Блок генерации управляющих сигналов. Схема функциональная

ПРИЛОЖЕНИЕ К
(обязательное)

Блок стека. Схема функциональная

ПРИЛОЖЕНИЕ Л
(обязательное)

Блок АЛУ. Схема функциональная

ПРИЛОЖЕНИЕ М
(обязательное)

Блок РОН. Схема функциональная