Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ВолгГТУ)

Факультет электроники и вычислительной техники

Кафедра ЭВМ и С

Контрольная работа

по дисциплине Архитектура вычислительных систем

Вариант 52

|  |  |
| --- | --- |
|  | Выполнил:  студент гр. ИВТ-261  Сидский Н.А.  Проверил:  к.т.н., Андреев А.Е. |

Волгоград, 2023 г.

Постановка задачи:

Синтезировать по варианту структуру простого магистрального процессора с одним или двумя АЛУ (возможно наличие отдельного адресного АЛУ), выполняющего от 8-и заданных команд. Разработать форматы команд, кодировку команд. Разработать структурную схему процессора, функциональные схемы всех блоков процессора, функциональную схему процессора в целом с указанием всех шин и управляющих сигналов. Разработать формат микрокоманд, организацию управления всеми устройствами процессора, микрокод для каждой из заданных команд. Привести примеры выполнения каждой команды с указанием значения всех основных сигналов и содержимого основных регистров на каждом такте. Привести 2 примера небольших программ с указанием значения основных сигналов и содержимого основных регистров на каждом такте.

Определить максимальную тактовую частоту процессора. Определить производительность процессора в операциях в секунду (IPS), а также выраженную в числе выполняемых тестовых программ в секунду. Указать способы повышения производительности процессора.

Характеристика процессора:

Простой процессор магистрального типа с одноблочным универсальным АЛУ (и возможно – с дополнительным адресным АЛУ по вариантам). Разрядность регистров РОН и АЛУ процессора – 8 бит. Число РОН – не менее 4. Адресуемая память - от 256 слов. Устройство управления – микропрограммное с памятью микропрограмм. Способ выполнения команд – последовательное выполнение.

Особенности варианта №52

Особенности АЛУ:

1) Тип сдвигателя в АЛУ: Комбинационный на любое число разрядов в

обе стороны

2) Операции АЛУ у всех : ADD + SRL (вправо логич.), NAND, INC

3) Адресность команд: Двухадресные (типа add r1, r2)

4) Адресная арифметика: На отдельном АЛУ

Особенности УУ:

1) Используемые варианты адресации (у всех : регистровая и прямая) + косвенная регистровая, непосредственная

2) Логика обработки разных форматов одних и тех же команд: Микропрограммная

3) Состав команд – у всех: LD, ST, ADD, SHL + JE, NOR, SHRA, XCH

Кодировка команд:

|  |  |
| --- | --- |
| Тип адресации | Код типа адресации - addrType |
| Регистровая (только не LD, ST)  ADD R1, R2 | 0 |
| Прямая (только LD, ST)  LD R1, addr | 0 |
| Косвенная регистровая (только LD, ST)  LD R1, [R2]; R1 = Mem[R2] | 1 |
| Непосредственная (только не LD, ST)  ADD R1, const; | 1 |

|  |  |  |
| --- | --- | --- |
| Код операции | Команда | Описание |
| 000 | ADD R1, R2; | //R1 = R1 + R2 |
| 000 | ADD R1, Imm; | //R1 = R1 + Imm |
| 001 | SHRA R1, R2; | // R1 = R1 >>>R2 |
| 001 | SHRA R1, Imm; | // R1 = R1 >>>Imm |
| 010 | SHL R1, R2; | // R1 = R1 <<R2 |
| 010 | SHL R1, Imm; | // R1 = R1<<Imm |
| 011 | LD R1, addr; | // R1 = Mem(addr) |
| 011 | LD R1, [R2]; | // R1 = Mem[R2]; |
| 100 | ST R1, addr; | // Mem(addr) = R1 |
| 100 | ST R1, [R2]; | // Mem[R2]=R1; |
| 101 | JE R1, Imm; | // jmp on equal |
| 110 | NOR R1, R2; | // R1 = ~(R2 | R2) |
| 110 | NOR R1, Imm; | // R1 = ~(R1 | Imm) |
| 111 | XCH R1, R2; | // temp=R1; R1=R2; R2=temp |

Формат микрокоманд:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ADD R1, R2 | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 0 | 0 | 0 | 0 | x | x | y | y |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ADD R1, Imm | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | Не исп. | |
| 0 | 0 | 0 | 1 | x | x | 0 | 0 |
| Imm (8 бит) – 2ой байт команды | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| SHRA R1, R2 | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 0 | 0 | 1 | 0 | x | x | y | y |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| SHRA R1, Imm | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | Не исп. | |
| 0 | 0 | 1 | 1 | x | x | 0 | 0 |
| Imm (8 бит) – 2ой байт команды | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| SHL R1, R2 | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 0 | 1 | 0 | 0 | x | x | y | y |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| SHL R1, Imm | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | Не исп. | |
| 0 | 1 | 0 | 1 | x | x | 0 | 0 |
| Imm (8 бит) – 2ой байт команды | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| LD R1, [R2] | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 0 | 1 | 1 | 1 | x | x | y | y |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| LD R1, addr | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | Не исп. | |
| 0 | 1 | 1 | 0 | x | x | 0 | 0 |
| addr (8 бит) – 2ой байт команды | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ST R1, [R2] | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 1 | 0 | 0 | 1 | x | x | y | y |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ST R1, addr | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | Не исп. | |
| 1 | 0 | 0 | 0 | x | x | 0 | 0 |
| addr (8 бит) – 2ой байт команды | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| JE Imm | | | | | | | |
| КОД(КОП) | | | addrType | Не исп. | | | |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| Imm (8 бит) – 2ой байт команды | | | | | | | |

Выполняет переход на адрес Imm при флаге Е = 1.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| NOR R1, R2 | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 1 | 1 | 0 | 0 | x | x | y | y |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| NOR R1, Imm | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | Не исп. | |
| 1 | 1 | 0 | 1 | x | x | 0 | 0 |
| Imm (8 бит) – 2ой байт команды | | | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| XCH R1, R2 | | | | | | | |
| КОД(КОП) | | | addrType | R1 | | R2 | |
| 1 | 1 | 1 | 0 | x | x | y | y |

Операционное устройство

Операционное устройство состоит из блока регистров, общего и адресного АЛУ. Адресное используется, когда приемник или хотя бы один из источников является адресным регистром (IR1, IR2, PC). Т.к. в адресном АЛУ всего 2 операции, то для кода операции адресного АЛУ достаточно 1 бита, в то время как для общего нужно 3 бита; в адресное АЛУ подаются младшие два бита кода действия.

Операции общего АЛУ:

|  |  |  |
| --- | --- | --- |
| Код | Операция общего АЛУ | Описание |
| 000 | ADD | Сложение |
| 001 | INC | Инкремент (+1) |
| 010 | NAND | НЕ-И |
| 011 | SRL | Логический сдвиг вправо |
| 100 | SHRA | Арифметический сдвиг вправо |

Операции адресного АЛУ:

|  |  |  |
| --- | --- | --- |
| Код | Операция адресного АЛУ | Описание |
| 0 | NOP | Выводит свой второй аргумент |
| 1 | ADD | Сложение |

Схема общего АЛУ:

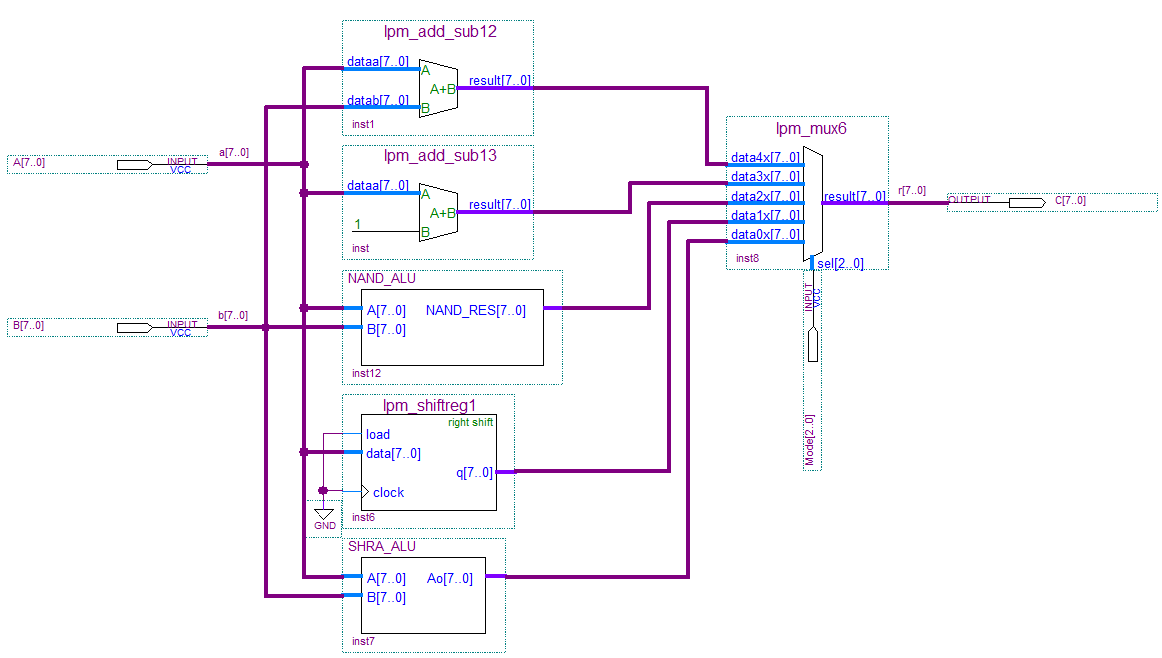


Диаграмма работы АЛУ:

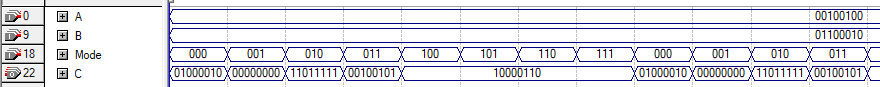


Схема адресного АЛУ:

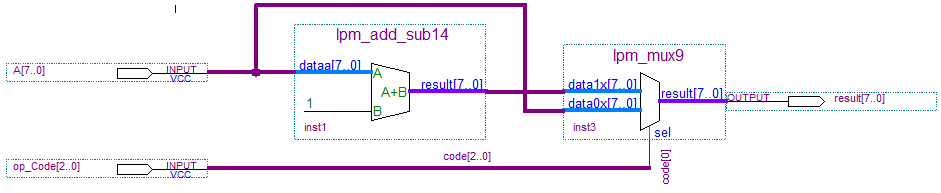
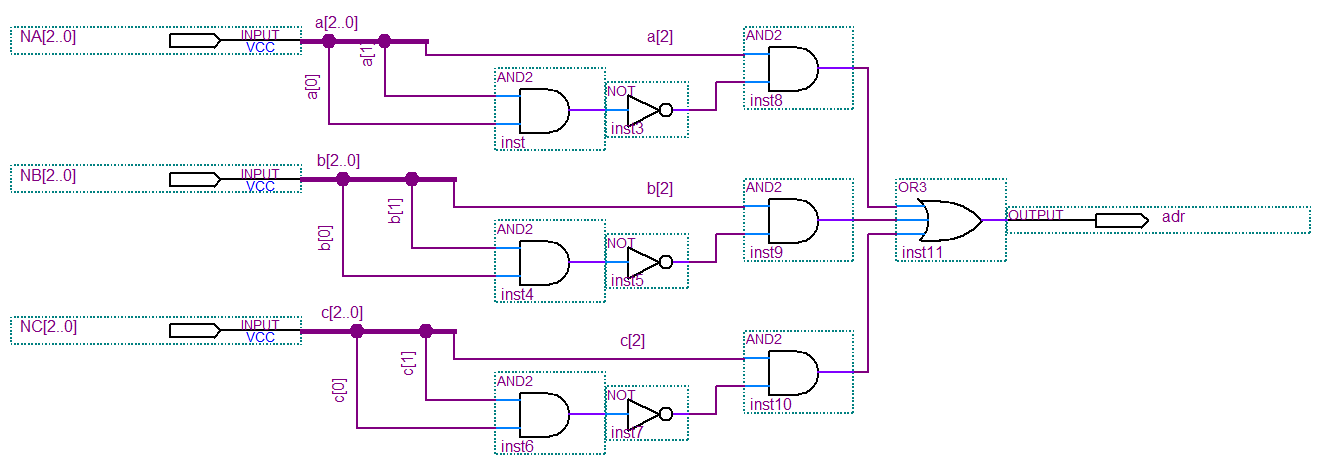


Диаграмма адресного АЛУ:



Схема выбора одного из двух АЛУ:



Выдает 1, когда нужно выбрать адресное АЛУ.

Блок регистров состоит из четырех программно-доступных регистров общего назначения Reg0, Reg1, Reg2, Reg3 и программно-недоступных регистров для внутренних данных: счетчик команд PC, два регистра инструкций IR1и IR2, временный регистр temp. IR2 также используется для хранения временных данных, когда регистра temp недостаточно.

Таблица адресов регистров:

|  |  |
| --- | --- |
| Регистр | Адрес |
| Reg0 | 000 |
| Reg1 | 001 |
| Reg2 | 010 |
| Reg3 | 011 |
| PC | 100 |
| IR1 | 101 |
| IR2 | 110 |
| temp | 111 |

Кроме того, в блоке регистров отдельно хранится флаг Z.

Схема блока регистров:

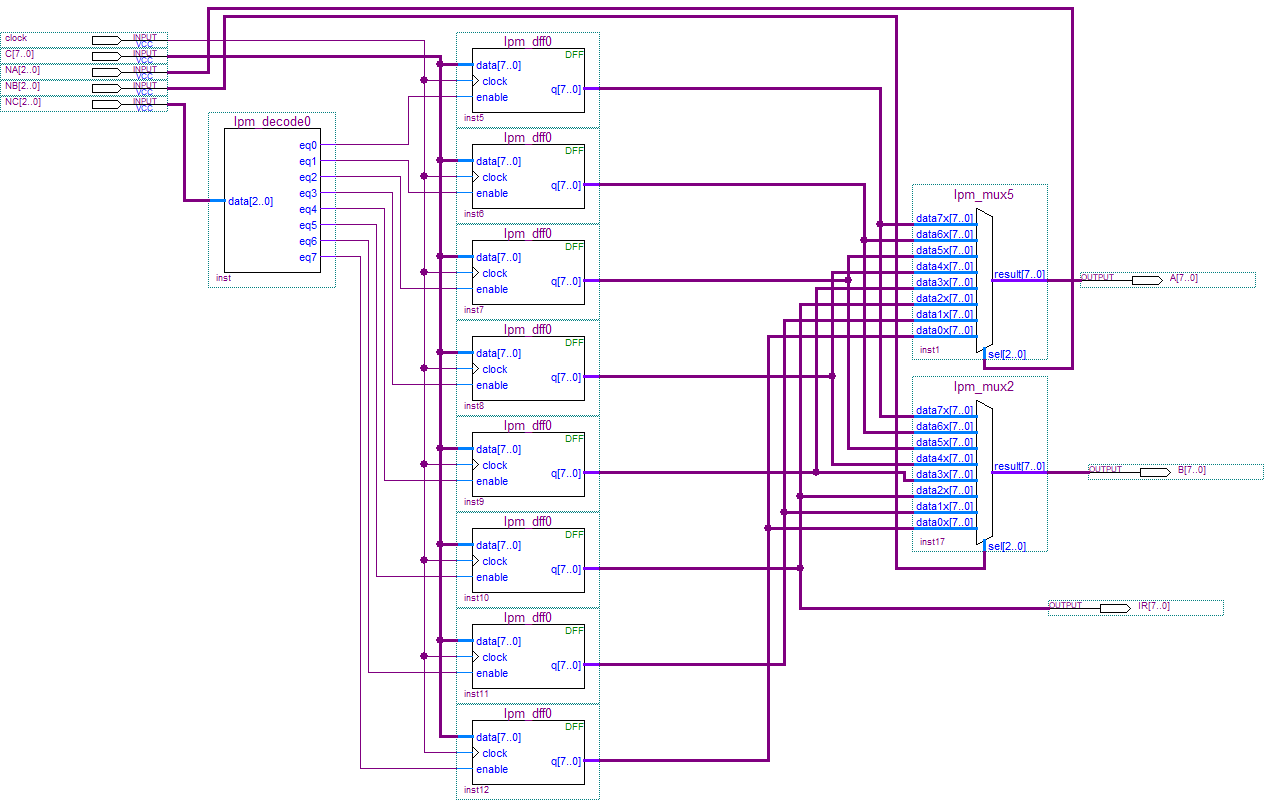


Диаграмма блока регистров:

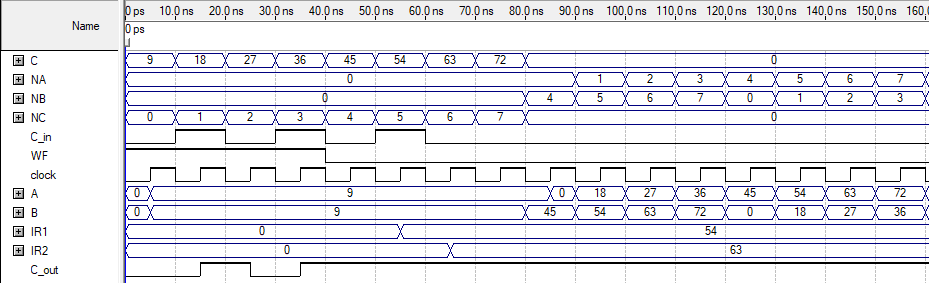


Схема операционного устройства в сборе:

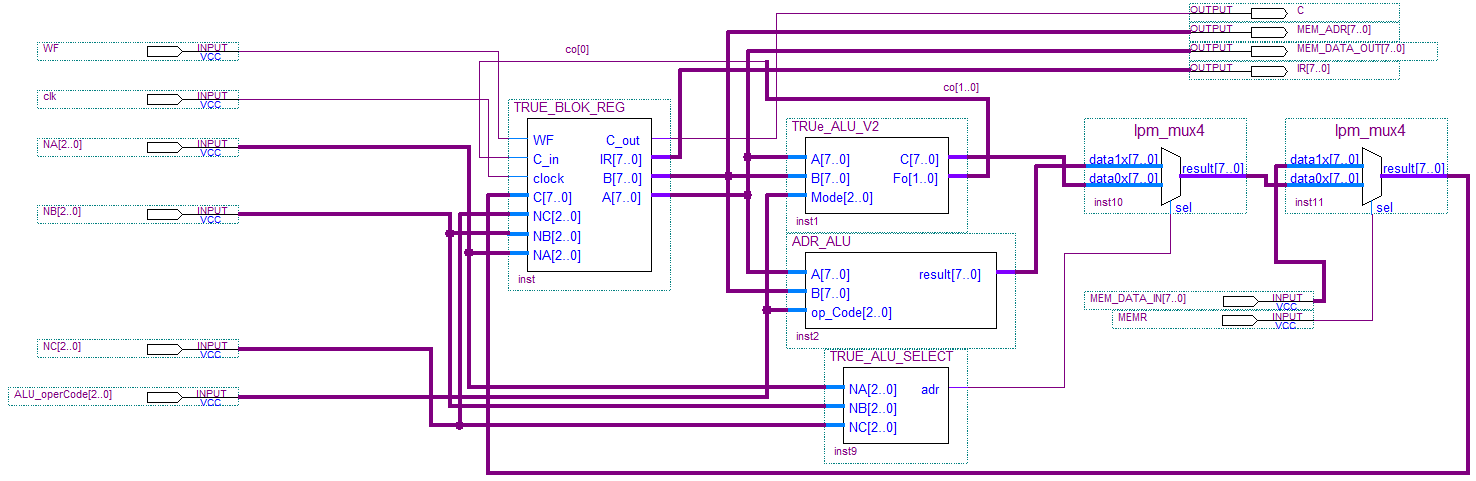
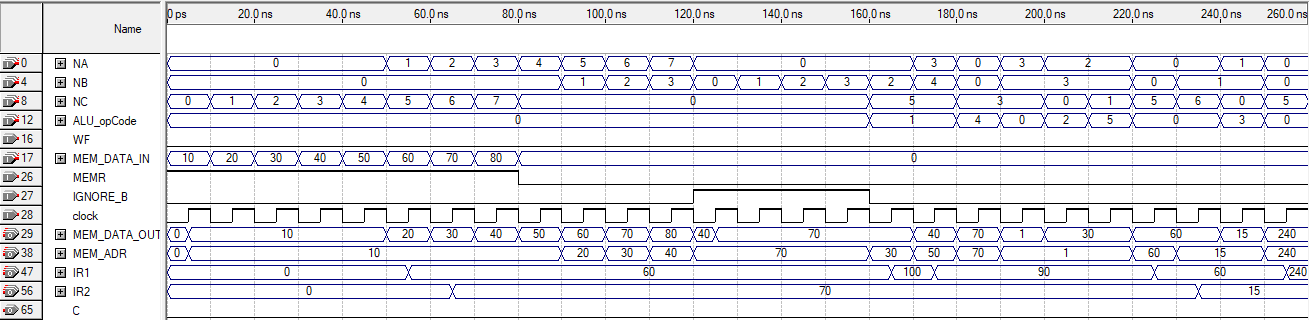


Диаграмма операционного устройства:



Управляющее устройство

Варианты формирования адреса:

|  |  |  |
| --- | --- | --- |
| Имя | Код | Описание |
| NEXT | 000 | Переход на следующий по порядку адрес |
| JMP | 001 | Безусловный переход на заданный адрес |
| JE | 010 | Переход при равных значениях операндов |
| DECODE | 101 | Декодирование; справа приписывается 011 |
| JNB | 100 | Условный переход при Addr, равным нулю |

Схема КС:

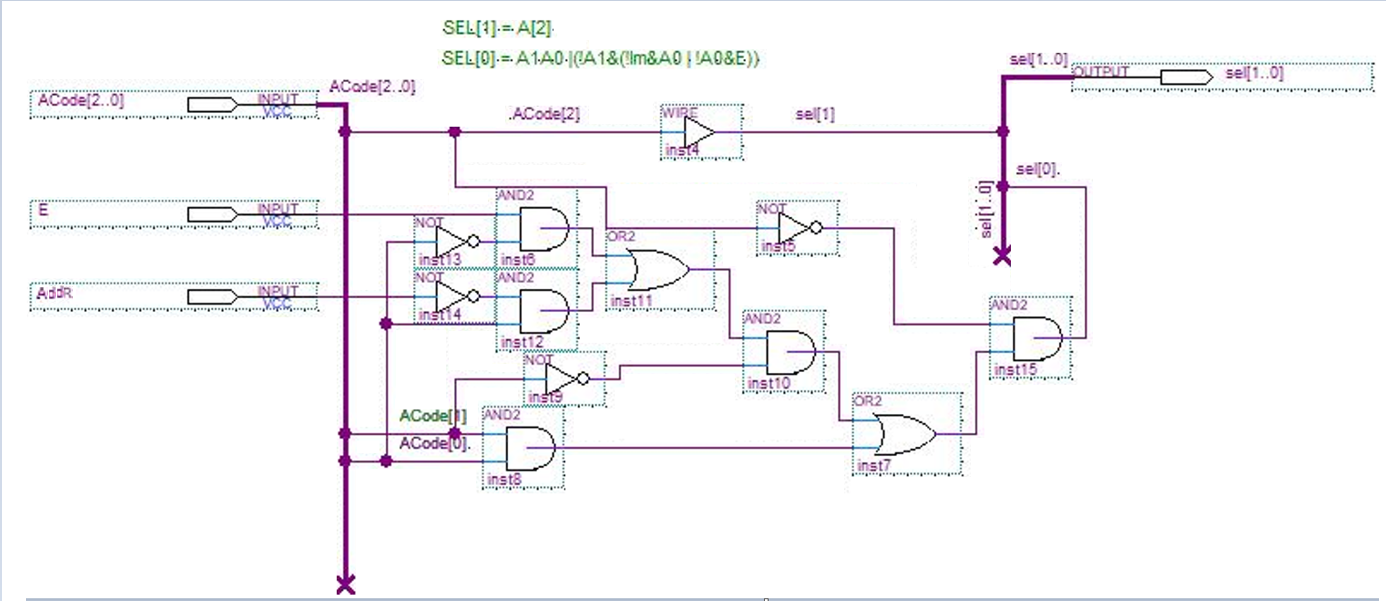


Таблица истинности:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Acode[2] | Acode[1] | Acode[0] | E | Addr | Sel1 | Sel0 |
| NEXT | 0 | 0 | 0 | \* | \* | 0 | 0 |
| DECODE | 1 | 0 | 1 | \* | \* | 1 | 0 |
| JMP | 0 | 0 | 1 | \* | \* | 0 | 1 |
| JE | 0 | 1 | 0 | 0 | \* | 0 | 0 |
| JE | 0 | 1 | 0 | 1 | \* | 0 | 1 |
| JNB | 1 | 0 | 0 | \* | 0 | 1 | 0 |
| JNB | 1 | 0 | 0 | \* | 1 | 0 | 0 |

Схема формирования адреса:

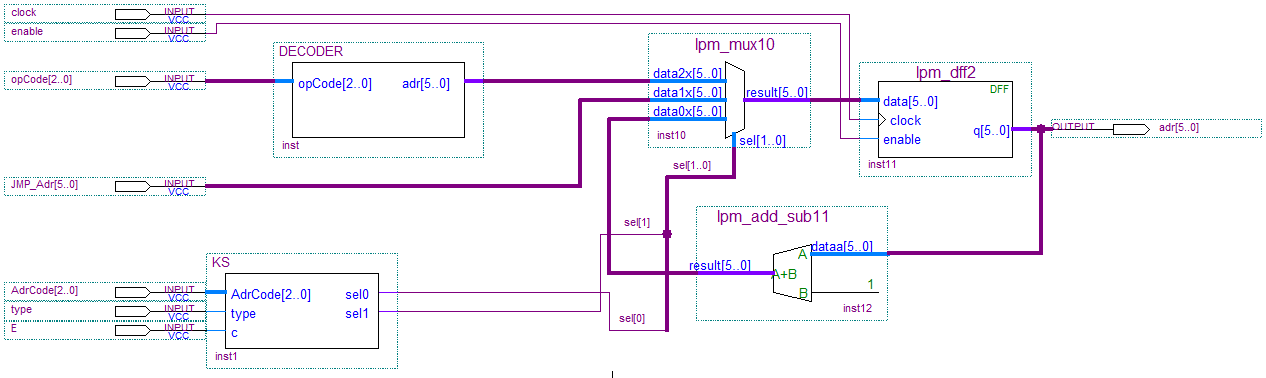


Диаграмма формирования адреса:

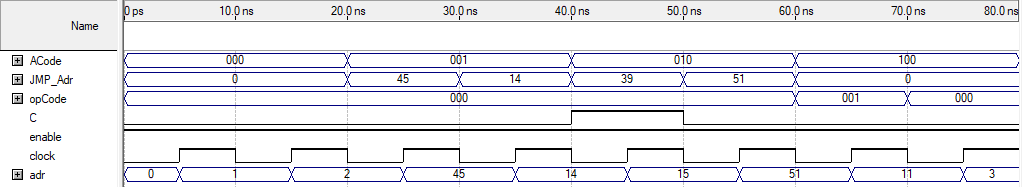
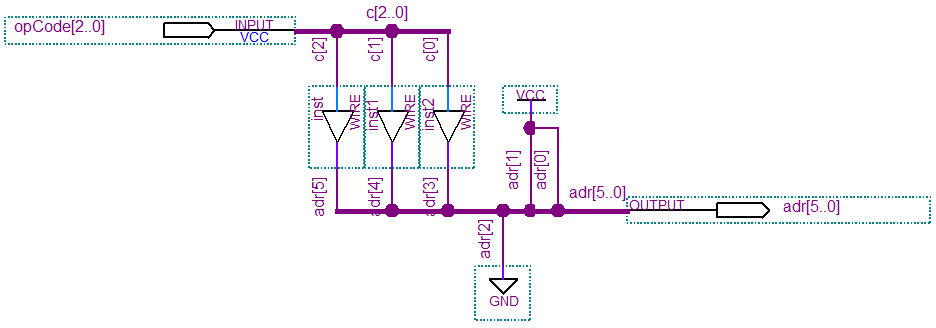


Схема декодирования:



Формат микрокоманды:

A, AMX, B, BMX: операнды; если AMX равен 1, то первый операнд выбирается из команды. Аналогично для BMX и CMX;

C, CMX: приемник;

ALU: действие АЛУ;

WF: если равен 0, то флаги не будут обновляться;

MEMR: если равен 1, то по адресу C будет записано значение из памяти, а не результат действия в АЛУ;

MEMW: если равен 1, то будет производиться запись в память;

ACode: вариант формирования адреса;

JMPA: адрес перехода;

Микропрограмма:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Адрес | AMX | A | BMX | B | CMX | C | ALU | WF | MEMR | MEMW | ACode | JMPA |
| 000000 (START) | 0 | A | 0 | PC | 0 | IR1 | NOP | 0 | 1 | 0 | NEXT |  |
| 000001 | 0 | PC | 0 | A | 0 | PC | INC | 0 | 0 | 0 | NEXT |  |
| 000010 | 0 | A | 0 | PC | 0 | IR2 | NOP | 0 | 1 | 0 | DECODE |  |
| 000011 (LD) | 1 | 001 | 1 | 010 | 0 | IR2 | ADD | 0 | 0 | 0 | NEXT |  |
| 000100 | 0 | temp | 1 | 001 | 1 | 000 | NOP | 0 | 1 | 0 | JMP | END |
| 001011 (ST) | 1 | 001 | 1 | 010 | 0 | IR2 | ADD | 0 | 0 | 0 | NEXT |  |
| 001100 | 1 | 000 | 1 | 001 | 0 | temp | NOP | 0 | 0 | 1 | JMP | END |
| 010011 (ADD) | 0 | A | 1 | 010 | 0 | temp | NOP | 0 | 0 | 0 | NEXT |  |
| 010100 | 1 | 001 | 0 | temp | 1 | 000 | ADD | 1 | 0 | 0 | JMP | END |
| 011011 (SHRA) | 0 | A | 1 | 010 | 0 | temp | NOP | 0 | 0 | 0 | NEXT |  |
| 011100 | 1 | 001 | 0 | temp | 1 | 000 | SHRA | 0 | 0 | 0 | JMP | END |
| 100011 (JC) | 0 | temp | 0 | temp | 0 | temp | NOP | 0 | 0 | 0 | JNC | END |
| 100100 | 0 | IR2 | 0 | A | 0 | PC | NOP | 0 | 0 | 0 | JMP | START |
| 101011 (NOR) | 0 | A | 1 | 010 | 0 | temp | NOP | 0 | 0 | 0 | NEXT |  |
| 101100 | 0 | temp | 0 | temp | 0 | temp | NAND | 0 | 0 | 0 | NEXT |  |
| 101101 | 0 | A | 0 | temp | 1 | 000 | NOP | 0 | 0 | 0 | NEXT |  |
| 101110 | 1 | 001 | 1 | 001 | 0 | temp | NAND | 0 | 0 | 0 | NEXT |  |
| 101111 | 1 | 000 | 0 | temp | 1 | 000 | NAND | 0 | 0 | 0 | NEXT |  |
| 110000 | 1 | 000 | 1 | 000 | 1 | 000 | NAND | 0 | 0 | 0 | JMP | END |
| 110011 (INC) | 0 | A | 0 | A | 0 | temp | SET | 0 | 0 | 0 | NEXT |  |
| 110100 | 1 | 000 | 0 | A | 1 | 000 | ADD | 1 | 0 | 0 | JMP | END |
| 111011 (SLC) | 0 | A | 1 | 010 | 0 | temp | NOP | 0 | 0 | 0 | NEXT |  |
| 111100 | 1 | 001 | 1 | 010 | 1 | 000 | SLC | 1 | 0 | 0 | JMP | END |
| 111101 (END) | 0 | PC | 0 | A | 0 | PC | INC | 0 | 0 | 0 | JMP | START |

Схема управляющего устройства:

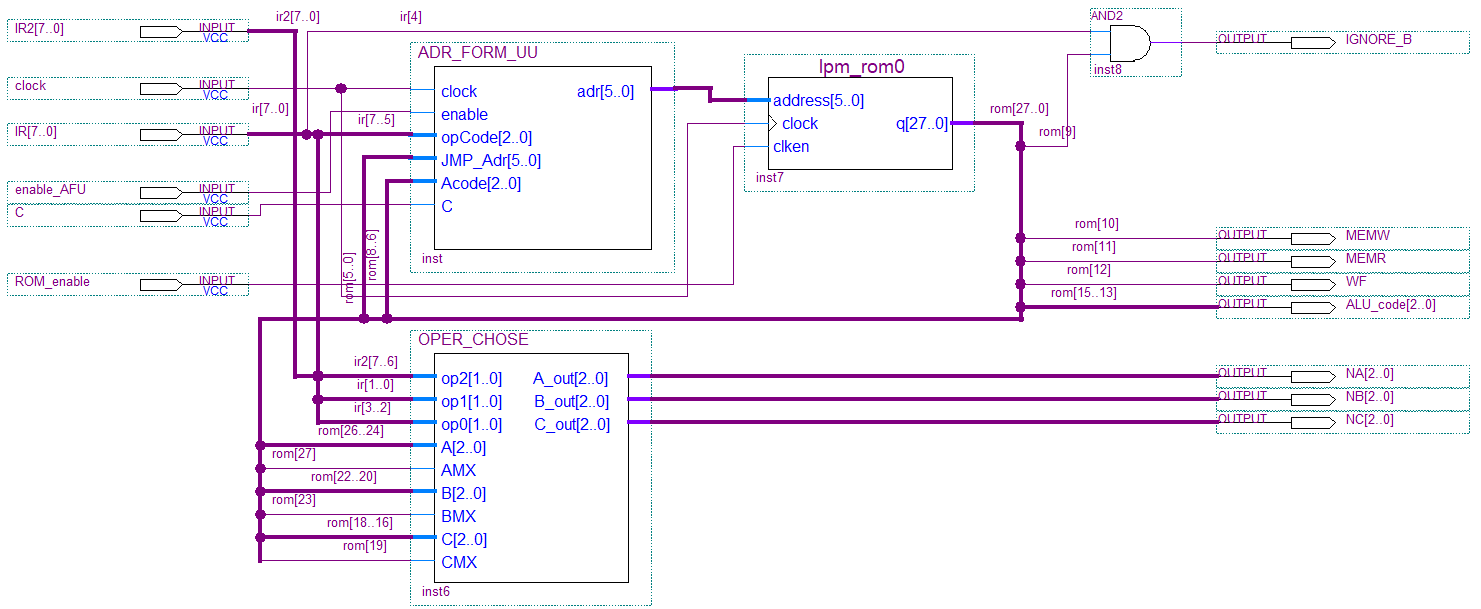
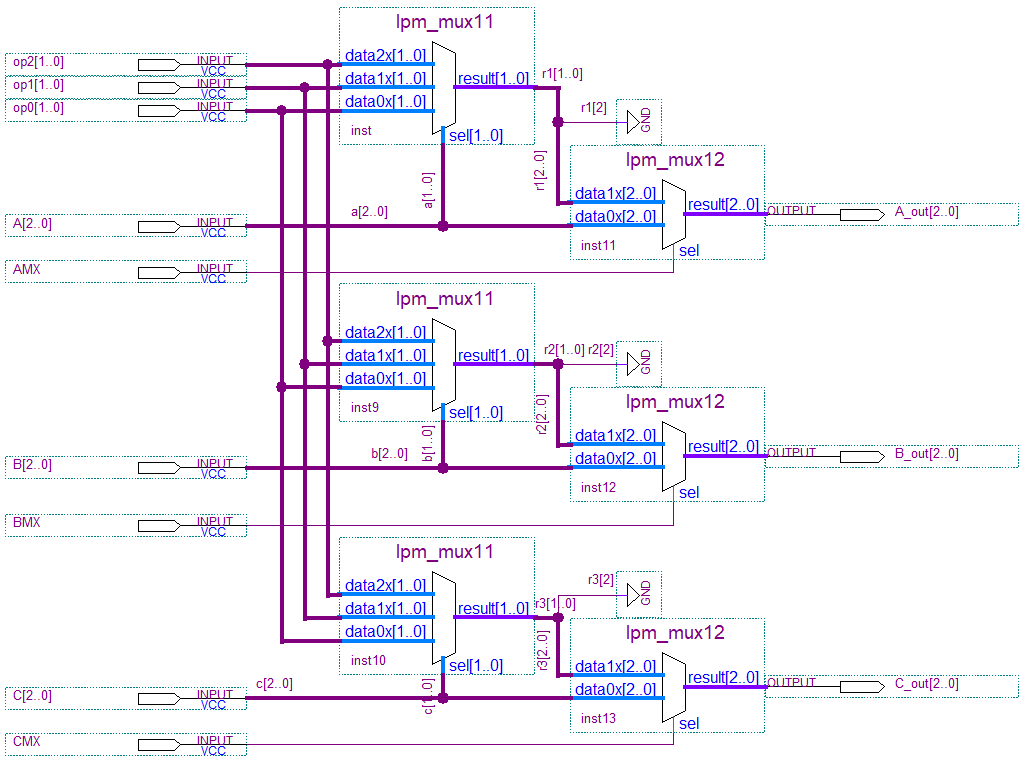
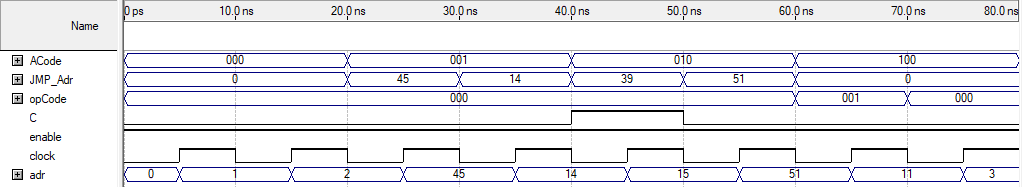


Схема выбора операндов:



Функциональные диаграммы

Схема формирования адреса:



Устройство управления:

На данный момент нахожусь в поисках ошибки, схема не компилируется.

Схема процессора в сборе:

