

Проектирование процессор ЭВМ

1 слайд

Добрый день! Я, студент группы Б17-503, Яковенко Иван представляю вам курсовой проект на тему "Проектирование процессора ЭВМ"

Курсовой проект «Проектирование процессора ЭВМ»

Студент группы Б17-503 Яковенко И. А.
Руководитель Ядыкин И. М.

2 слайд

Работа ведется с четырехразрядными числами в дополнительном коде со знаком Спроектированный процессор выполняет 4 операции: УМНОЖЕНИЕ чисел со знаком с младших разрядов ПЕРЕСЫЛКУ ОТРИЦАТЕЛЬНУЮ – то есть взятие отрицательного значения модуля числа. Так же устанавливается признак результата равный знаку результата (1 если число меньше нуля, 0 если число равно нулю) И два перехода:

- Условный переход, если признак результата равен 1
- Безусловный переход

Адрес перехода задается в виде смещения в дополнительном коде.

Техническое задание № 19-15

Оперативная память – **16x8**

Регистровая память – **8x4**

Операнды – **дробные числа** в дополнительном коде

Слово = **4 разряда**

Формат команд:

Первый операнд команды хранится в **РП**. Адресация прямая

Второй операнд хранится в **ОП** (РА2=0 - прямая адресация, РА2=1 - постиндексная косвенная вариант 2)

Результат операции **УМНОЖЕНИЕ** записывается по адресу **второго операнда**.

Результат **ПЕРЕСЫЛКИ ОТРИЦАТЕЛЬНОЙ** по адресу **первого операнда**

Операции:

УМНОЖЕНИЕ – алгоритм умножения чисел в дополнительном коде с младших разрядов множителя

ПЕРЕСЫЛКА ОТРИЦАТЕЛЬНАЯ – дополнительный код абсолютного значения **второго операнда**

пишется по **адресу первого операнда**. То есть, модуль второго операнда берем со знаком минус (исключение 0). Устанавливается признак результата: 0 – (результат = 0), 1 – (результат < 0)

ПЕРЕХОД, ЕСЛИ 1 – продвинутый адрес в счетчике команд замещается адресом перехода, если значение PR = 1. Используется относительная адресация (в команде – смещение со знаком)

БЕЗУСЛОВНЫЙ ПЕРЕХОД – продвинутый адрес в счетчике команд замещается адресом перехода.

Используется относительная адресация (в команде указывается смещение со знаком)

2

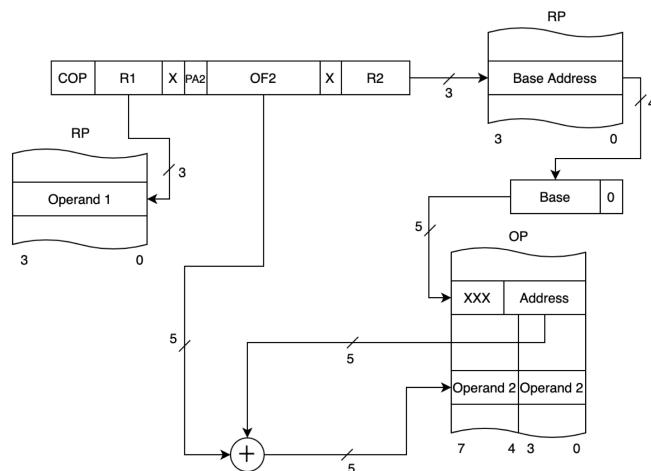
3 слайд

Для размещения команд в памяти были разработаны представленные форматы. Первый операнд команды УМНОЖЕНИЕ и ПЕРЕСЫЛКА ОТРИЦАТЕЛЬНАЯ задается всегда при помощи прямой регистровой адресации. Второй операнд, в зависимости от признака адресации РА2, указывается либо с помощью прямой адресации, либо с помощью постиндексной косвенной. Для экономного использования памяти расположение полей команд было размещено в 2, 3 или 4 словах. Для команд перехода используется относительная адресация и соответственно смещение со знаком, которое указывается в команде. Если прямые и относительные адресации, думаю всем понятны, остановимся на постиндексной косвенной адресации. Её схема показана справа на слайде. В команде указывается номер регистра, в котором хранится базовый адрес, используемый в качестве косвенного, и непосредственно смещение. При этом, так как базовый адрес занимает пять разрядов, его старший бит должен храниться в ячейке с четным адресом. Так как логический адрес в ОП должен содержать 5 разрядов, смещение, полученное из регистра, расширяется нулем справа до 5 разрядов. Сумма смещения, полученного из кода команды, и базового адреса, полученного из ОП, является исполнительным адресом операнда.

Форматы команд и способы адресации

	RK1	RK2	RK3	RK4
Переход	2 1	5		
	COP X	SM		
	15 14 13 12	8		
PA2 = 1 Умножение / Пересылка	2 3 1 1	5 1 3		
	COP R1 X PA2 OF2 X R2			
	15 14 13 11 10 9 8	4 3 2 0		
PA2 = 0 Умножение / Пересылка	2 3 1 1	5		
	COP R1 X PA2 A2			
	15 14 13 11 10 9 8	4		

Форматы команд



Постиндексная косвенная адресация

3

4 слайд

Центральным управляющим блоком спроектированного процессора является блок выработки микрокоманд. БМК был спроектирован на основе алгоритма микропрограммы. Для организации выполнения микропрограммы была использована принудительная адресация. Для хранения адреса микрокоманды - регистр адреса микрокоманды. В микрокоманде, кроме вырабатываемых управляющих сигналов, указываются так же адрес следующей микрокоманды и признаки, проверяемые в случае ветвления. Если не указан ни один признак, тогда ветвления не происходит, и в регистр адреса микрокоманды записывается адрес, указанный в микрокоманде. Если указан хотя бы один признак, тогда в микропрограмме возможно ветвление на 2 направления. В случае равенства единице хотя бы ондго из признаков, указанных в микрокоманде, в регистр адреса микрокоманды загружается адрес, идущий следующим за адресом из адресного поля микрокоманды. В противном случае записывается сам адрес, указанный в адресном поле микрокоманды. При этом, в случае ветвления, адрес перехода всегда должен быть четным. Процесс ветвления основан на изменении младшего разряда адреса следующей микрокоманды нулем или единицей, в случае когда указан хоть один признак.

Блок выработки микрокоманд

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A3	A2	A1	A0	COP1	COP0	PA2	PR	YC15	YC14	YC13	YC12	YC11	YC10	YC9	YC8	YC7	YC6	YC5	YC4	YC3	YC2	YC1	SNO

Формат команды

Принудительная адресация

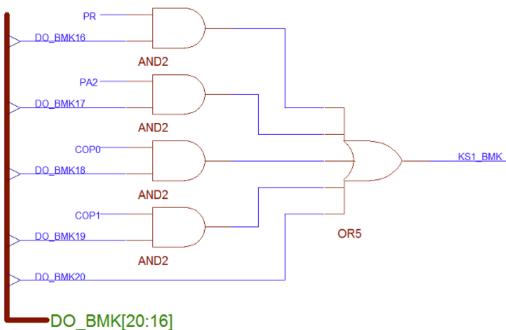
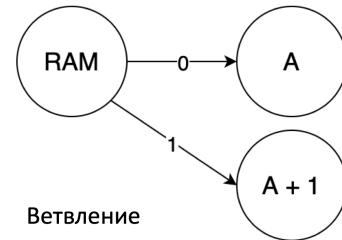
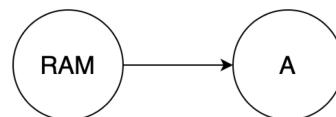


Схема формирования младшего разряда



Ветвление



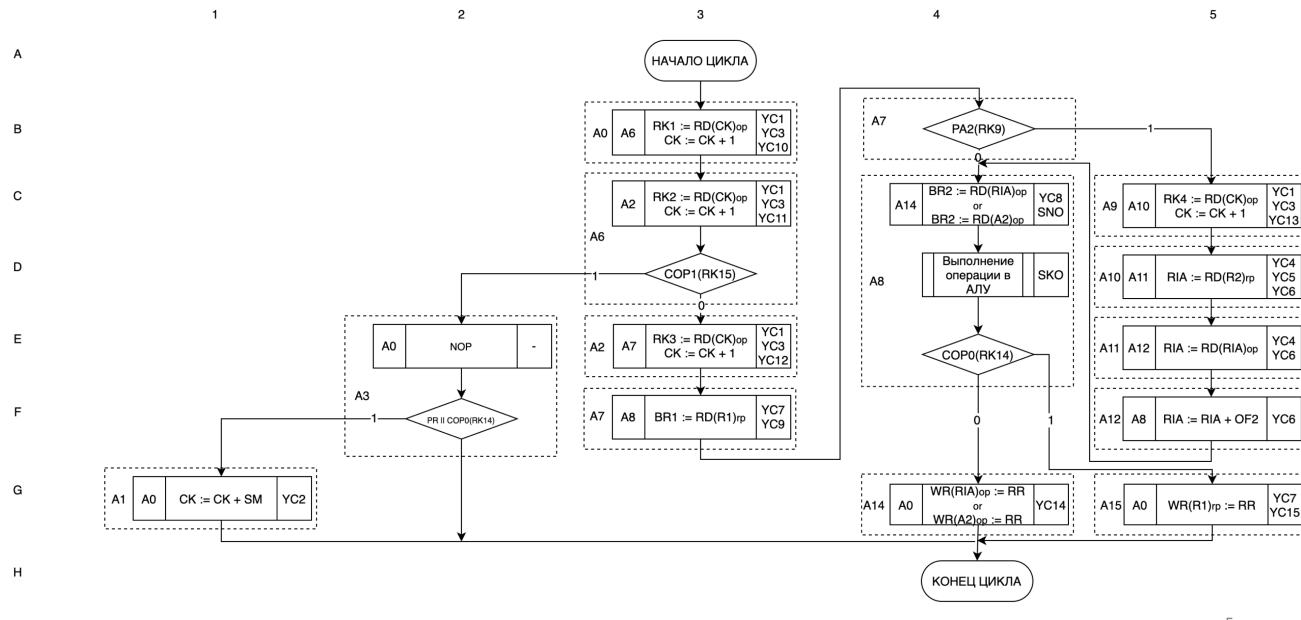
Последовательное выполнение

4

5 слайд

Была разработана блок-схема алгоритма выполнения последовательности команд, который лег в основу спроектированного БМК. Серьезным требованием при разработке алгоритма явилось соблюдение правильной расстановки адресов, так как адрес переход всегда должен быть четным. В алгоритме присутствует необходимость ветвления более чем на 2 направления, для этой цели были добавлены микрокоманды, не вырабатывающие управляющих сигналов (NOP). Каждой представленной на блок-схеме микрокоманде поставлен в соответствие набор сигналов, управляющих работой блока управления командами, функциональная схема которого представлена на следующем слайде. Для выполнения любой микрокомандычитываются минимум два слова. Далее определяется тип команды – если команда нелинейная, то уходим на выполнение условного или безусловного перехода. В ином случае считывается еще одно слово и определяется тип адресации по признаку PA2. Для постиндексной косвенной нужна будет прочитать 4 слова. Далее выполняется считывание второго операнда и выработка сигнала начала работы АЛУ. После окончания работы АЛУ результат записывается в соответствующую ячейку соответствующей памяти и выполнение команды заканчивается.

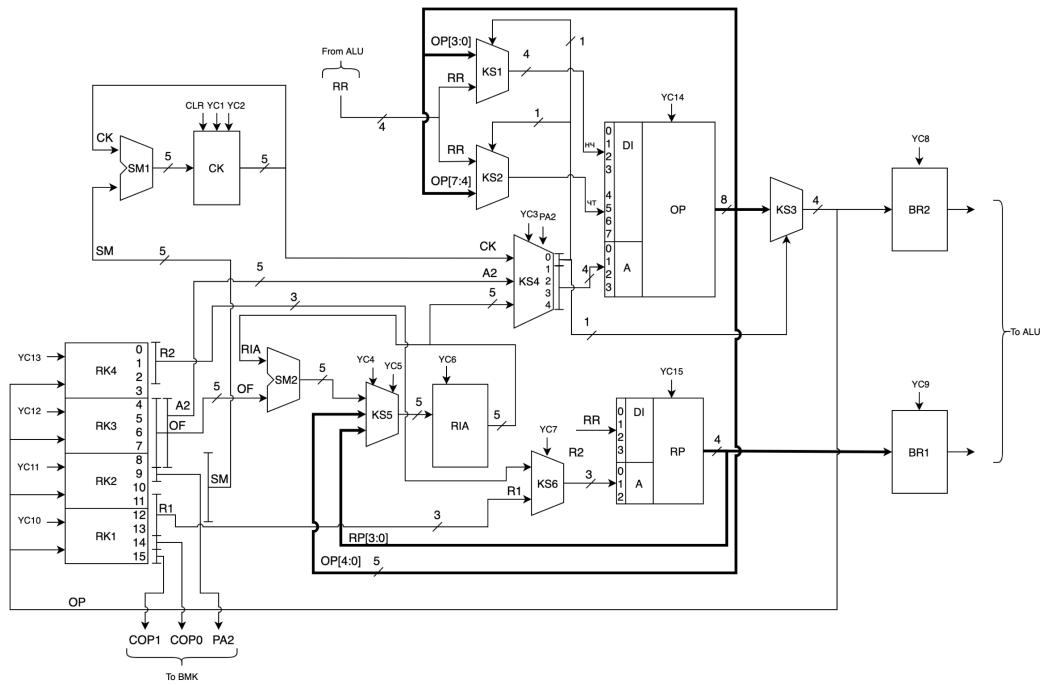
Алгоритм выполнения команд



6 слайд

Основными задачами БУК являются хранение и формирование адреса следующей выполняемой команды, считывание операндов и передача их в АЛУ и запись результата вычисления по окончанию работы АЛУ. Регистр команд состоит из 4 четырех-словных регистров хранения. Для реализации постиндексной косвенной адресации был введен регистр исполнительного адреса RIA. В него последовательно записывается адрес из регистровой памяти, расширенный нулем, базовый адрес из оперативной памяти и наконец сумма базы и смещения для обращения к оперативной памяти по исполнительному адресу. Для выбора четного или нечетного слова из оперативной памяти используется младший разряд адреса. На адресный вход подаются оставшиеся 4 разряда адреса. Для записи в ОП только одного слова используется два коммутатора, управляемых младшим разрядом исполнительного адреса. В случае равенства этого разряда единице в ячейку ОП записываются старшие 4 разряда текущего содержимого этой ячейки и результат операции. В случае равенства нулю - результат выполнения операции и младшие 4 разряда текущего содержимого.

Функциональная схема БУК

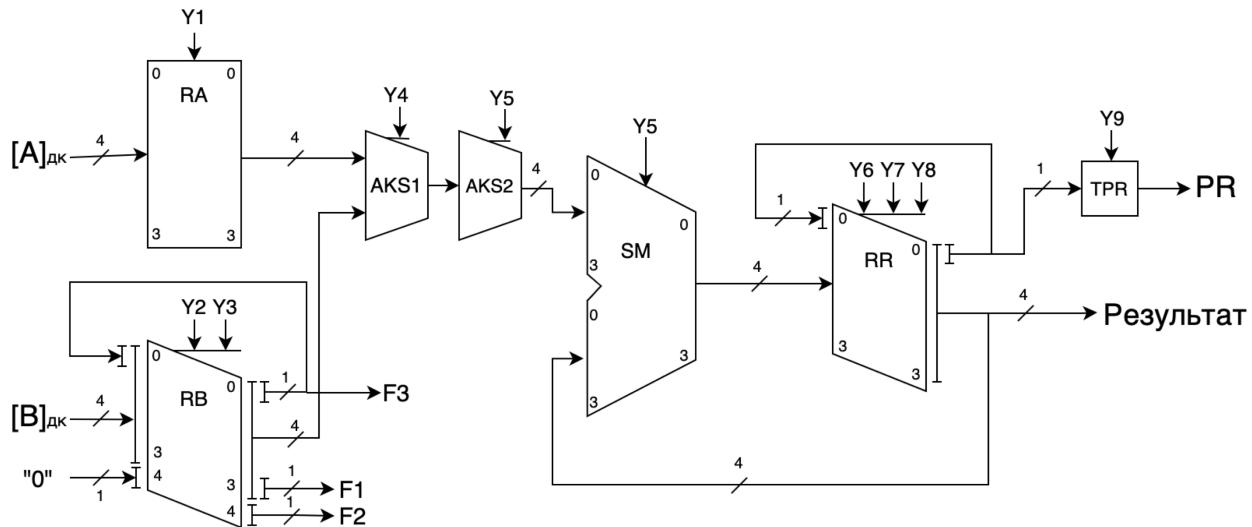


6

7 слайд

Основная логика работы АЛУ обусловлена блоком операций. Данный блок был спроектирован для выполнения двух, указанных в тех задании операций. Так как операции выполняются в дополнительном коде и при инвертировании необходимо прибавлять 1, управляющий сигнал Y5 второй комбинационной схемы, отвечающей за инвертирование числа так же заведен на сумматор в качестве входа переноса. Для реализации алгоритма умножения чисел с младших разрядов множителя и сдвигом частичных произведений вправо путем последовательного преобразования множителя в качестве регистра второго операнда и регистра результата были взяты сдвиговые регистры на 5 и 4 разряда. В регистре второго операнда выдигаемый младший разряд остается в 5 разряде, позволяя тем самым проще анализировать последние два разряда (на схеме F1 и F2). Для второй операции анализируется знак второго операнда и далее в регистр результата записывается прямое или обратное значение числа. Знак результата сохраняется в качестве признака в триггере признака результата (TPR). Управление БО производится при помощи МУУ, вместе они образуют АЛУ.

Функциональная схема БО

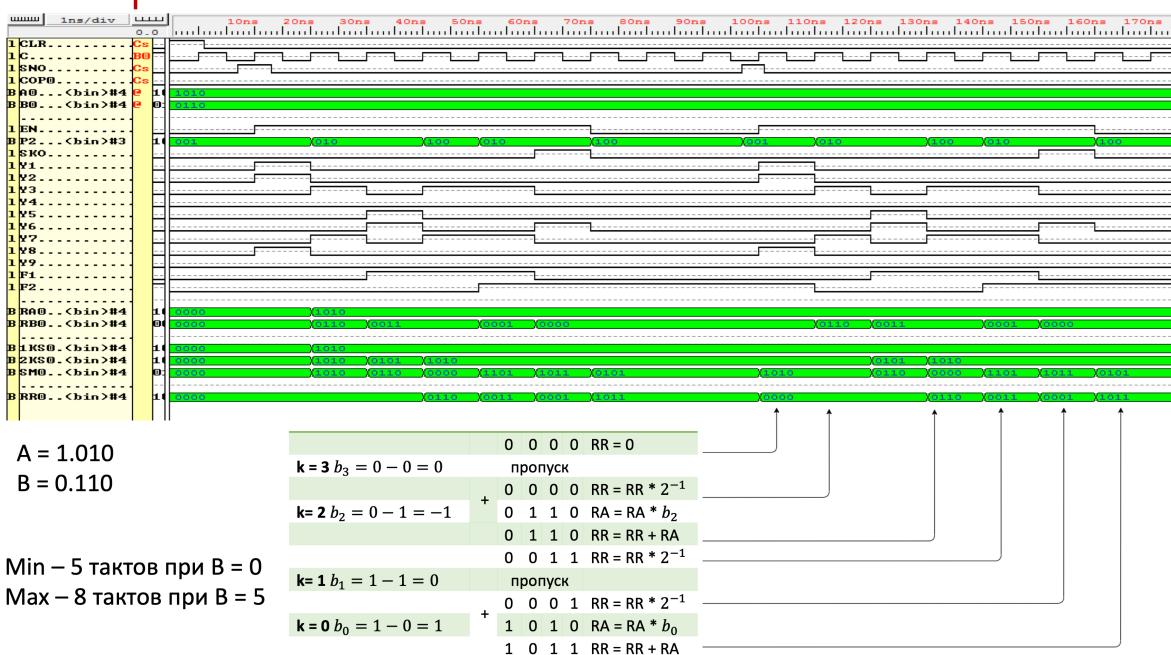


7

8 слайд

На слайде показан пример работы АЛУ на функциональном моделировании. Операнд A = 1.010 или $-6/8$, операнд B = 0.110 или $6/8$. При перемножении ожидаемый результат должен быть $-36/64$ или при обрезке до 4 битов $-5/8$ то есть 1.011. Этапы вычисления проиллюстрированы с помощью примера вычисленного вручную. Для проверки старта МУУ не с нулевого состояния тест был запущен дважды. В процессе тестирования было выяснено, что минимальное число тактов достигается при операнде B = 0. Максимальное число тактов при B = 5.

Тестирование АЛУ



8

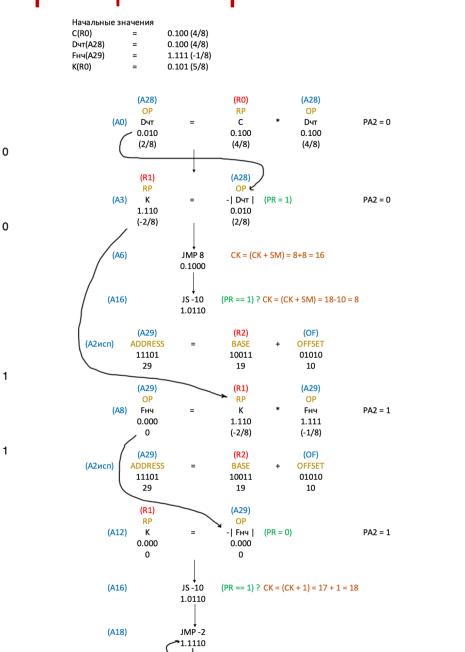
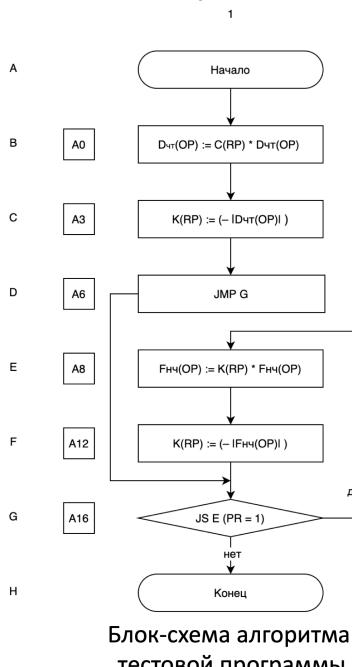
9 слайд

Для отладки процессора было проведено его тестирование. Данный тест обеспечивает:

- Проверку всех микрокоманд алгоритма (проход по всем веткам алгоритма)
- Проверку считывания команд разного формата, расположенныхных по четным и нечетным адресам в оперативной памяти (напр., первое умножение и пересылка)
- Проверку считывания операндов из оперативной памяти, расположенныхных по четным и нечетным адресам (D и F)
- Проверку записи результата в оперативную память по четным и нечетным адресам (D и F, в операциях пересылки)
- Проверку считывания операндов из регистровой памяти (C/K)
- Проверку записи операндов в регистровую память (K)
- Проверку всех используемых способов адресации (все)
- Проверку правильности вычисления признака перехода (есть или нет переход на A16)
- Проверку правильности работы АЛУ (значения совпадают с рассчитанными вручную)

Размещение команды в памяти представлено на рисунке справа. Значения, вырабатываемые процессором полностью совпадают со значениями, рассчитанными вручную. Соответственно на рассчитанном примере показан ход выполнения программы. Изогнутыми стрелками показана связь между записанными и считанными значениями. Изображен процесс вычисления исполнительного адреса для operandов с использованием постиндексной косвенной адресации.

Тестирование процессора



Пример выполнения тестовой программы

Адрес ячейки	Адрес слова	Команды и данные	Двоичный код	HEX
0	A0; A1	MUL, R0, 0, PA2, A28	0000 001	0 1
1	A2; A3	A28, SND, R1	1100 0100	C 4
2	A4; A5	R1, 0, PA2, A28	1001 1100	9 C
3	A6; A7	JMP, 0, 8	1100 1000	C 8
4	A8; A9	MUL, R1, 0, PA2, OF	0000 1010	0 A
5	A10; A11	OF, 0, R2	1010 0010	A 2
6	A12; A13	SND, R1, 0, PA2, OF	0100 1010	4 A
7	A14; A15	OF, 0, R2	1010 0010	A 2
8	A16; A17	JS, 0, -10	1001 0110	9 6
9	A18; A19	JMP, 0, -2	1101 1100	D E
10	A20; A21	0000, 0000	0000 0000	0 0
11	A22; A23	0000, 0000	0000 0000	0 0
12	A24; A25	0000, 0000	0000 0000	0 0
13	A26; A27	000, 19	0001 0011	1 3
14	A28; A29	[D] = +4/8, [F] = -1/8	0100 1111	4 F
15	A30; A31	0000, 0000	0000 0000	0 0

Размещение программы в ОП

Адрес ячейки	Адрес слова	Команды и данные	Двоичный код	HEX
0	R0	[C] = +4/8	0100	4
1	R1	[K] = +5/8	0101	5
2	R2	A13	1101	D
3	R3	0000	0000	0
4	R4	0000	0000	0
5	R5	0000	0000	0
6	R6	0000	0000	0
7	R7	0000	0000	0

Размещение программы в РП

Заключение

Таким образом, для реализации процессора было спроектировано 4 основных блока:

- БО – для выполнения операций
- МУУ – для задания последовательности микроопераций БО
- БУК – для обработки команд

- БМК - для задания последовательности микрокоманд

Для каждого из блоков было выполнено моделирование и тестирование. Блоки были соединены вместе, образуя процессор. В качестве дальнейшего развития проекта предлагается включение новых команд для процессора таких как СТОП, возможно написание своего рода транслятора из более понятного человеку кода в машинные слова.