

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  
(национальный исследовательский университет)**

О.М.БРЕХОВ, Г.А. ЗВОНАРЕВА, А.В. КОРНЕЕНКОВА, Е.А. ДАВЫДКИНА

**ОСНОВЫ ФУНКЦИОНИРОВАНИЯ ЯДРА ЭВМ**

Учебное пособие

Утверждено  
на заседании редсовета  
Протокол № \_\_\_\_  
« \_\_\_\_ » \_\_\_\_\_ 202\_\_ г.

Москва  
Издательство МАИ  
2021

**УДК 004.318**

**Брехов О.М., Звонарева Г.А., Корнеев А.В., Давыдкина Е.А.**  
Основы функционирования ядра ЭВМ: Учебное пособие. – М.: Изд-во МАИ, 2021. – 55 с.: ил.

Учебное пособие по основам функционирования ядра ЭВМ разработано на кафедре «Вычислительные машины, системы и сети» МАИ в соответствии с рабочими программами дисциплин «Организация ЭВМ», «Архитектура ЭВМ», «Вычислительные машины, системы и сети» по направлениям подготовки бакалавриата «Информатика и вычислительная техника», «Информационные системы и технологии», «Управление в технических системах».

В учебном пособии рассмотрены вопросы, связанные с организацией функционирования ядра ЭВМ, в том числе: вопросы по организации операционной и управляющей частей центрального процессора; описание систем счисления, используемых в ЭВМ; правила перевода из одной системы счисления в другую; кодирование команд и данных; формы представления чисел в ЭВМ. Предложенный материал необходим для проработки материала, связанного с организацией и архитектурой вычислительных машин, в том числе для выполнения практических и курсовых работ, предусмотренных рабочими программами указанных дисциплин. Теоретические сведения подкреплены примерами, а также представлены задания по вариантам для выполнения практических и курсовых работ. Работы рассчитаны на самостоятельное выполнение студентами под руководством преподавателя.

Учебное пособие предназначено для студентов, обучающихся по направлениям, связанным с разработкой и применением вычислительных машин, в том числе авиационного назначения.

Рецензенты:

ISBN

© Московский авиационный институт

(национальный исследовательский университет), 2021

## Оглавление

ВВЕДЕНИЕ.....	4
ТЕМА №1. ПРЕДСТАВЛЕНИЕ ЧИСЛОВОЙ ИНФОРМАЦИИ В ЭВМ.....	5
1.1. СИСТЕМЫ СЧИСЛЕНИЯ, ИСПОЛЬЗУЕМЫЕ В ЭВМ.....	5
1.2. ПРЯМОЙ, ОБРАТНЫЙ, ДОПОЛНИТЕЛЬНЫЙ КОДЫ .....	12
1.3. ФОРМЫ ПРЕДСТАВЛЕНИЯ ЧИСЕЛ В ЭВМ.....	19
ТЕМА №2. ОРГАНИЗАЦИЯ ОПЕРАЦИОННОЙ ЧАСТИ ЦЕНТРАЛЬНОГО ПРОЦЕССОРА. ....	22
ТЕМА № 3. ОРГАНИЗАЦИЯ УСТРОЙСТВА УПРАВЛЕНИЯ. ....	38
РАЗРАБОТКА МИКРОПРОГРАММНОГО УСТРОЙСТВА УПРАВЛЕНИЯ ОПЕРАЦИОННОЙ ЧАСТЬЮ ЦЕНТРАЛЬНОГО ПРОЦЕССОРА. КУРСОВОЕ ПРОЕКТИРОВАНИЕ .....	46
ЛИТЕРАТУРА.....	54

## ВВЕДЕНИЕ

Традиционно ядро ЭВМ – это процессор и память. Процессор – функциональная часть ЭВМ, выполняющая основные операции по обработке данных и управлению работой других блоков. В свою очередь процессор включает арифметико-логическое устройство (АЛУ) и устройство управления (УУ).

АЛУ предназначено для выполнения всех арифметических и логических операций над числовой и символьной информацией.

УУ формирует и подает во все блоки машины в нужные моменты времени определенные сигналы управления (управляющие импульсы), обусловленные спецификой выполняемой операции и результатами предыдущих операций; формирует адреса ячеек памяти, используемых выполняемой операцией, и передает эти адреса в соответствующие блоки компьютера; опорную последовательность импульсов устройство управления получает от генератора тактовых импульсов.

Для понимания организации работы АЛУ необходимо знать способы представления числовой информации в ЭВМ, вопросы по организации операционной части центрального процессора, системам счисления, используемым в ЭВМ, правилам перевода из одной системы счисления в другую, кодированию команд и данных, формам представления чисел в ЭВМ.

Для понимания организации работы УУ необходимо знать понятие микрооперации, микрокоманды и микропрограммы, организации микропрограммной памяти, связи операционной и управляющей частей центрального процессора.

## ТЕМА №1. ПРЕДСТАВЛЕНИЕ ЧИСЛОВОЙ ИНФОРМАЦИИ В ЭВМ

### 1.1. СИСТЕМЫ СЧИСЛЕНИЯ, ИСПОЛЬЗУЕМЫЕ В ЭВМ.

Системой счисления называется способ представления чисел посредством алфавита символов. Разделяются на позиционные и непозиционные системы счисления.

В ЭВМ используются позиционные системы счисления.

Основанием системы счисления называется количество цифр, использующееся в данной системе счисления.

В примерах рассматриваются следующие системы счисления:

0, 1, 2, 3, ..., 9 – десятичная система счисления

0, 1, 2, 3, 4, 5, 6, 7 – восьмеричная система счисления

0, 1 – двоичная система счисления

0, 1, 2, 3, ..., 9, A, B, C, D, E, F – шестнадцатеричная система счисления

В ЭВМ используются двоичная, восьмеричная, шестнадцатеричная системы счисления.

Пример. Сложение в двоичной системе

$$\begin{array}{r} + \quad 1 \ 0 \ 1 \ 0 \ 1 \ 1 \\ \quad 0 \ 0 \ 0 \ 1 \ 1 \ 1 \\ \hline 1 \ 1 \ 0 \ 0 \ 1 \ 0 \end{array}$$

Таблица соответствия чисел от 0 до 15, представленных в десятичной, двоичной и шестнадцатеричной системах счисления (табл. 1), где q – основание системы счисления.

Таблица 1.

Таблица соответствия чисел от 0 до 15, представленных в десятичной, двоичной и шестнадцатеричной системах счисления

q=10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
q=2	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
q=16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

## **Представление чисел в позиционной системе счисления**

Число в позиционной системе счисления представляется в виде полинома:

$$X = X_n X_{n-1} \dots X_1 X_0, X_{-1} \dots X_{-m} = X_n q^n + X_{n-1} q^{n-1} + \dots + X_1 q^1 + X_0 q^0 + X_{-1} q^{-1} + \dots + X_{-m} q^{-m}$$

Где:  $n + 1$  - число цифр в целой части числа,

$m$  – число цифр в дробной части числа.

### Пример:

Число 235,87 в десятичной системе счисления записывается в виде:

$$2 \cdot 10^2 + 3 \cdot 10^1 + 5 \cdot 10^0 + 8 \cdot 10^{-1} + 7 \cdot 10^{-2}$$

В настоящее время наибольшее распространение получили двоичная и шестнадцатеричная системы счисления, использование которых на различных примерах будет продемонстрировано ниже.

## **Правила перевода чисел из двоичной (шестнадцатеричной) системы счисления в десятичную систему счисления**

Для перевода числа из двоичной (шестнадцатеричной) системы счисления в десятичную систему счисления число представляется в виде полинома, где все цифры выражаются в десятичной системе счисления и действия выполняются в десятичной системе счисления.

Пример: перевести число 11000, выраженное в двоичной системе счисления, в десятичную систему счисления:

$$(11000)_2 = (?)_{10}$$

$$\begin{array}{r} 43210 \\ 11000 \end{array}$$

$$(11000)_2 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 16 + 8 = (24)_{10}$$

Пример: перевести число 18, выраженное в шестнадцатеричной системе счисления, в десятичную систему счисления

$$(18)_{16} = (?)_{10}$$

$$(18)_{16} = 1 \cdot 16^1 + 8 \cdot 16^0 = (24)_{10}$$

## **Правила перевода чисел из десятичной системы счисления в двоичную (шестнадцатеричную) систему счисления.**

Существуют разные правила для перевода целых и дробных чисел.

### **1. Правила перевода целых чисел.**

Чтобы перевести целое число из десятичной системы счисления в двоичную (шестнадцатеричную) систему счисления, необходимо исходное число разделить на основание новой системы счисления (два или шестнадцать). Полученное частное вновь делится на основание и так до тех пор, пока частное не станет меньше основания новой системы счисления. Последнее частное будет первой цифрой в числе новой системы счисления. Последующие цифры в новом числе будут соответствовать остаткам в обратном порядке их получения.

Пример: перевести целое число 24 из десятичной системы счисления в двоичную систему счисления.

$$(24)_{10} - (?)_2$$

$$\begin{array}{r}
 24 \overline{) 2} \\
 24 \overline{) 2} \\
 \hline
 0126 \overline{) 2} \\
 063 \overline{) 2} \\
 021 \\
 \hline
 1
 \end{array}$$

Таким образом получается число:  $(11000)_2$ .

Пример: перевести целое число 24 из десятичной системы счисления в шестнадцатеричную систему счисления.  $(24)_{10} - (?)_{16}$

$$\begin{array}{r}
 24 \overline{) 16} \\
 16 \overline{) 1} \\
 \hline
 8
 \end{array}$$

Получается число  $(18)_{16}$ .

## 2. Правила перевода дробной части числа.

Если дробь неправильная – целая и дробная часть переводится по своим правилам.

Чтобы перевести правильную дробь из десятичной системы счисления в двоичную (шестнадцатеричную) систему счисления, необходимо исходную дробь умножить на основание новой системы счисления. В полученном произведении выделяется целая и дробная части. Дробная часть умножается на основание новой системы.

Цифры в новой системе счисления соответствуют целым частям произведения в порядке их получения.

Пример: перевести правильную дробь 0,35 из десятичной системы счисления в двоичную систему счисления.  $(0,35)_{10} \rightarrow (?)_2$

$$0,35 \cdot 2 = \underline{0},7; \quad 0,7 \cdot 2 = \underline{1},4; \quad 0,4 \cdot 2 = \underline{0},8; \quad 0,8 \cdot 2 = \underline{1},6; \quad 0,6 \cdot 2 = \underline{1},2; \quad 0,2 \cdot 2 = \underline{0},4; \dots$$

Таким образом, получается число  $(0,010110\dots)_2$

Пример: перевести правильную дробь 0,35 из десятичной системы счисления в шестнадцатеричную систему счисления.  $(0,35)_{10} \rightarrow (?)_{16}$

0.35	0.6	0.6
* 16	* 16	* 16
<u>5.6</u>	<u>9.6</u>	<u>9.6</u>

Получается число  $(0,59\dots9)_{16}$

### **Правила перевода из двоичной системы счисления в шестнадцатеричную систему счисления.**

Чтобы перевести число из двоичной системы счисления в шестнадцатеричную систему счисления, необходимо, двигаясь вправо и влево от запятой, разбить число на тетрады. Каждую тетраду необходимо заменить



шестнадцатеричной цифрой (см.таблицу1), недостающие разряды добавляются нулями.

Пример: перевести число 11000, 01011 из двоичной системы в шестнадцатеричную систему счисления

$$(00011000,01011000)_2 - (?)_{16}$$

$$1 \quad 8 \quad , \quad 5 \quad 8$$

Получается число  $(18,58)_{16}$

### **Правила перевода из шестнадцатеричной системы счисления в двоичную систему счисления.**

Чтобы перевести число из шестнадцатеричной системы счисления в двоичную систему счисления необходимо каждую шестнадцатеричную цифру заменить эквивалентной тетрадой в двоичной системе счисления (см. табл. 1).

Пример: перевести число 18,59 из шестнадцатеричной системы счисления в двоичную систему счисления.

$$(18,59)_{16} - (?)_2$$

$$\begin{array}{cccc} 1 & 8 & , & 5 & 9 \\ 0001 & 1000, & 0101 & 1001 \end{array}$$

Получается число  $(11000,01011001)_2$ .

### Задание по разделу 1.1.

Перевести число А из десятичной системы счисления в двоичную и шестнадцатеричную систему счисления.

Полученные числа перевести в десятичную систему счисления.

Перевести число А из двоичной системы счисления в шестнадцатеричную систему счисления. И обратно, из шестнадцатеричной системы счисления в двоичную систему счисления.

Перевести число В из десятичной системы счисления в шестнадцатеричную систему счисления, а затем полученное число из шестнадцатеричной системы счисления в двоичную систему счисления. Числа А и В приведены в табл. 2.

Таблица 2.

Значение чисел А и В

Номер варианта	А	В
1	28	38
2	29	37
3	30	36
4	31	35
5	32	37
6	33	38
7	34	34
8	35	33
9	36	35
10	37	36
11	38	32
12	39	37
13	40	35
14	41	34
15	42	31
16	43	33
17	44	30
18	45	28
19	46	26
20	47	23
21	48	20
22	49	25
23	50	23
24	55	51
25	54	49
26	53	49
27	52	48
28	51	47
29	20	47
30	21	50
31	22	46
32	23	48

33	24	45
34	25	44
35	26	43
36	27	46
37	28	37
38	29	38
39	30	36
40	31	35
41	32	37
42	33	33
43	34	34
44	35	32
45	36	31

## 1.2. ПРЯМОЙ, ОБРАТНЫЙ, ДОПОЛНИТЕЛЬНЫЙ КОДЫ

### Прямой код числа

Прямым кодом целого числа  $x$  является число, образованное по формуле:

$$X = \begin{cases} 0X_{n-2} \dots X_1X_0 & , \quad X \geq 0 \\ 1X_{n-2} \dots X_1X_0 & , \quad X < 0 \end{cases}$$

где  $n-1$  – число значащих разрядов в числе

Для дробного числа:

$$X = \begin{cases} 0,X_{-1} \dots X_{-(n-1)} & , \quad X \geq 0 \\ 1,X_{-1} \dots X_{-(n-1)} & , \quad X < 0 \end{cases}$$

В дальнейшем, для наглядности, знаковый разряд отделяется точкой:

$$\begin{array}{c|c} 0 & 101 \\ \text{Знаковый разряд} & \text{Значащие разряды} \\ 1 & 101 \end{array}$$

### Обратный код числа

Обратным кодом целого числа  $x$  является число, образованное по формуле:

$$X = \begin{cases} 0X_{n-2} \dots X_1X_0 & , \quad X \geq 0 \\ 1\overline{X}_{n-2} \dots \overline{X}_1\overline{X}_0 & , \quad X < 0 \end{cases}$$

Для дробного числа:

$$X = \begin{cases} 0,X_{-1} \dots X_{-(n-1)} & , \quad X \geq 0 \\ 1,\overline{X}_{-1} \dots \overline{X}_{-(n-1)} & , \quad X < 0 \end{cases}$$

$\overline{X}_i$  – это дополнение до основания системы счисления.

Для двоичной системы счисления обратный код отрицательного числа получается путём инвертирования значащих разрядов, т.е. нули заменяются единицами и наоборот - единицы нулями, а в знаковый разряд ставится 1.

Пример:

0.101    +5 прямой код  
1.010    -5 обратный код

Для положительных чисел прямой и обратный код совпадают.

При использовании алгебраического суммирования отрицательные числа представляются в обратном коде, положительные – в прямом и производится поразрядное суммирование, включая знаковый разряд.

Если возникает единица переноса из знакового разряда, то она суммируется с младшим разрядом.

Если знаковый разряд суммы равняется “0”, то это означает, что результат положительный и представлен в прямом коде.

Если в знаковом разряде суммы единица, то это означает, что результат - отрицательный и представлен в обратном коде.

Пример:

A=5, B=4

0.0101	+5 прямой код	0.0100	+4 прямой код
1.1010	-5 обратный код	1.1011	-4 обратный код

<u>A+B</u>	
+ 0.0101	+5 прямой код
0.0100	+4 прямой код
<hr/> 0.1001	+9 прямой код
 <u>A-B</u>	
+ 0.0101	+5 прямой код
1.1011	-4 обратный код
<hr/> 1 0.0000	
↙ +1	
0.0001	прямой код

Знаковый разряд результата равен 0. Из этого следует, что результат положительный и представлен в прямом коде.

$$\begin{array}{r}
 \underline{B-A} \\
 0.0100 \quad +4 \text{ прямой код} \\
 +1.1010 \quad -5 \text{ обратный код} \\
 \hline
 1.1110
 \end{array}$$

Знаковый разряд результата равен 1. Следовательно, результат получается отрицательным и представлен в обратном коде. Прямой код результата равен:

$$1.0001 \quad \text{прямой код}$$

### Дополнительный код числа

Дополнительным кодом целого числа  $x$  является число, образованное по формуле:

$$X = \begin{cases} 0X_{n-2} \dots X_1X_0 & , \quad X \geq 0 \\ 1\bar{X}_{n-2} \dots \bar{X}_1\bar{X}_0 + 1 & , \quad X < 0 \end{cases}$$

Для дробного числа:

$$X = \begin{cases} 0, X_{-1} \dots X_{-(n-1)} & , \quad X \geq 0 \\ 1, \bar{X}_{-1} \dots \bar{X}_{-(n-1)} + 2^{-(n-1)} & , \quad X < 0 \end{cases}$$

Положительное число в прямом и дополнительном коде совпадают.

Чтобы получить дополнительный код целого отрицательного двоичного числа необходимо проинвертировать значащую часть и прибавить «1» к младшему разряду, в знаковый разряд поставить 1.

Пример:

$$A=5, B=4$$


$$\begin{array}{r}
 1.1010 \quad -5 \text{ обратный код} \\
 + \quad 1 \\
 \hline
 1.1011 \quad -5 \text{ дополнительный код}
 \end{array}$$

$$\begin{array}{r}
 1.1011 \quad -4 \text{ обратный код} \\
 + \quad 1 \\
 \hline
 1.1100 \quad -4 \text{ дополнительный код}
 \end{array}$$

При алгебраическом суммировании с использованием дополнительного кода, отрицательные числа представляются в дополнительном коде, а положительные – в прямом коде и производится суммирование кодов чисел, включая знаковый разряд.

При возникновении единицы переноса из знакового разряда, эта единица отбрасывается в отличие от обратного кода.

Пример:

<u>A-B</u>	
1.1100	-4 дополнительный код
+0.0101	+5 прямой код
0.0001	+1 прямой код
	

Если образовался “0” в знаковом разряде, то число получилось положительным и представлено в прямом коде, а если “1”, то это число отрицательное и представлено в дополнительном коде.

Пример:

<u>B-A</u>	
0.0100	+4 прямой код
+1.1011	-5 дополнительный код
1.1111	

Знаковый разряд результата равен 1. Из этого следует, что результат получился отрицательным и представлен в дополнительном коде. Прямой код результата равен:

+ 1.0000	
1	
1.0001	-1 прямой код

### **Переполнение разрядной сетки**

Переполнением разрядной сетки называется ситуация, когда при суммировании кодов чисел результат требует на один разряд больше, чем исходные операнды и не помещается в разрядной сетке ЭВМ [1].

#### Признаки переполнения разрядной сетки ЭВМ

*1. По наличию и отсутствию переноса в знаковый и из знакового разряда.*

Переполнение возникает в том случае, если существует только лишь один перенос, либо в знаковый разряд, либо из знакового разряда. Если существуют

оба переноса и в знаковый разряд, и из знакового разряда, либо переносы из знакового и в знаковый разряды отсутствуют, то переполнения разрядной сетки нет.

Пример:

A=5, B=4

0.101 +5 прямой код	0.100	+4 прямой код
1.010 -5 обратный код	1.011	-4 обратный код

A+B

1		
0.101	+5 прямой код	
+ 0.100	+4 прямой код	
<u>1.001</u>	переполнение!!!	

Имеется один перенос в знаковый разряд, перенос из знакового разряда отсутствует.

-A-B

1		
1.010	-5 обратный код	
+ 1.011	-4 обратный код	
<u>0.101</u>	переполнение!!!	

Имеется один перенос из знакового разряда, перенос в знаковый разряд отсутствует.

## 2. Модифицированное кодирование.

Модифицированным кодом называется код, в котором под знак числа отводится два или более разрядов.

При использовании двух разрядов для представления знака числа комбинация знаковых разрядов 00 соответствует положительным числам, комбинация 11 – отрицательным числам, комбинации 01 и 10 соответствуют переполнению разрядной сетки.

Пример:



### A+B

+ 00.101	+5 прямой модифицированный код
00.100	+4 прямой модифицированный код
01.001	переполнение!!!

Комбинация 01 в знаковом разряде соответствует переполнению разрядной сетки.

### (-A)+(-B)

+ 11.010	-5 обратный модифицированный код
11.011	-4 обратный модифицированный код
10.101	переполнение!!!

Комбинация 10 в знаковом разряде соответствует переполнению разрядной сетки.

Пример (для дополнительного модифицированного кода):

### A-B

00.101	+5 прямой модифицированный код
+ 11.100	-4 дополнительный модифицированный код
00.001	+1 прямой код

В знаковых разрядах результата комбинация 00. Из этого следует, что переполнение разрядной сетки отсутствует, результат положительный.

### B-A

00.100	+4 прямой модифицированный код
+ 11.011	-5 дополнительный модифицированный код
11.111	

В знаковых разрядах результата комбинация 11. Из этого следует, что переполнение разрядной сетки отсутствует и результат отрицательный, представлен в дополнительном коде.

+ 11.000	
1	
11.001	-1 прямой код

### (-A)+(-B)

+ 11.011	-5 дополнительный модифицированный код
<u>11.100</u>	- 4 дополнительный модифицированный код
10.111	переполнение!!!

В знаковых разрядах результата комбинация 10, что соответствует переполнению разрядной сетки.

### Задание по разделу 1.2.

Вычислить  $\underline{A-B}$  и  $\underline{B-A}$  с использованием:

- обратного модифицированного кода;
- дополнительного модифицированного кода.

Вычислить  $\underline{A+B}$  с помощью:

- модифицированного кодирования;
- определить переполнение разрядной сетки ЭВМ по наличию и отсутствию переноса в знаковый и из знакового разряда соответственно.

Число значащих разрядов в числе равно 6. Числа А и В приведены в табл. 1 (см. задание 1).

### 1.3. ФОРМЫ ПРЕДСТАВЛЕНИЯ ЧИСЕЛ В ЭВМ.

В данном разделе на примерах рассматриваются наиболее часто используемые формы представления чисел в ЭВМ [1].

#### **Формат представления чисел с фиксированной точкой**

Рассмотрим формат представления чисел с фиксированной запятой (рис. 1).

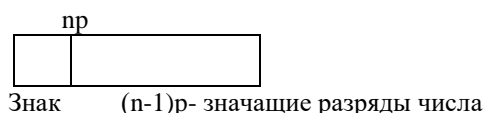


Рис. 1. Формат представления чисел с фиксированной запятой

Точка может фиксироваться либо перед старшим разрядом числа, либо после младшего. Если точка фиксируется перед старшим разрядом числа, то все числа по модулю меньше единицы. Если после младшего, то все числа по модулю больше единицы.

В современных ЭВМ для представления целых чисел используется формат чисел с фиксированной точкой, причем точка фиксируется после младшего разряда.

Отрицательные числа с фиксированной точкой хранятся в памяти в обратном или в дополнительном коде.

#### **Формат чисел с плавающей точкой**

Число  $X$  с плавающей точкой можно представить по формуле:

$$X = \pm M_x \cdot q^{\pm p_x}$$

Где:  $M_x$  – мантисса числа,

$q$  – основание системы счисления,

$p$  – порядок.

Мантисса – это нормализованная правильная дробь.

Формат чисел с плавающей точкой представлен на рис. 2.

Тогда:

$$R_{cm} = R_x + 2^L - \text{смещенный порядок,}$$

где  $L$  – количество разрядов, отводимое под порядок.

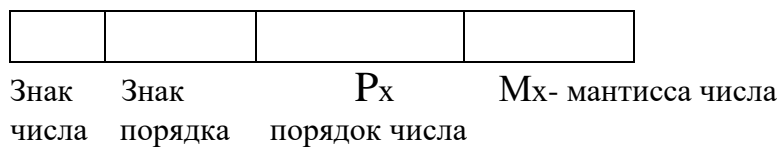


Рис. 2. Формат чисел с плавающей точкой

В современных ЭВМ часто для представления мантиссы используется либо двоичная, либо шестнадцатеричная системы счисления. Ниже для наглядности рассматриваются примеры использования шестнадцатеричной системы счисления для представления мантиссы числа.

### Примеры представления чисел в форме с плавающей точкой

Длина разрядной сетки  $n$  принимается равной 32 разряда, под смещенный порядок отводится 7 разрядов.

Пример: Число  $(24,35)_{10}$  представить в форме с плавающей точкой.

$$(24,35)_{10} - (18,59...9)_{16} \quad P_x = 2$$

$$P_{см} = (42)_{16} = (66)_{10}$$

0	100 0010	0001 1000 0101 1001 1001 1001
Знак числа	4    2	1    8    5    9    9    9

Пример: Число  $(38,0)_{10}$  представить в форме с плавающей точкой.

$$(38,0)_{10} - (26,0)_{16} \quad P_x = 2$$

$$P_{см} = (42)_{16} = (66)_{10}$$

0	100 0010	0010 0110 0000 0000 0000 0000
Знак числа	4    2	2    6

Пример: Число  $(0,38)_{10}$  представить в форме с плавающей точкой.

$$(0,38)_{10} - (0,6147AE)_{16} \quad P_x = 0$$

$$P_{CM} = P_X + 2L = 0 + 64 = (64)_{10} = (40)_{16}$$

0	100 0000	0110 0001 0100 0111 1010 1110
Знак числа	4      0	6      1      4      7      A      E

### Задание по разделу 1.3.

Числа А и -А представить в формате с фиксированной точкой. Разрядная сетка равна 16 разрядам.

Представить число в формате с плавающей точкой. Разрядная сетка равна 32 разряда, 7 разрядов – смещенный порядок, 24 разряда – мантисса числа. Числа А и В приведены в табл. 2 (см. задание 1).

- а) А,0
- б) 0,А
- в) А,В
- г) -А,В

## ТЕМА №2. ОРГАНИЗАЦИЯ ОПЕРАЦИОННОЙ ЧАСТИ ЦЕНТРАЛЬНОГО ПРОЦЕССОРА.

Центральный процессор (ЦП) – это устройство предназначается для непосредственной обработки данных и работает под управлением программ.

### **Основные узлы центрального процессора**

1) Арифметико-логическое устройство (АЛУ) – предназначается для выполнения арифметических и логических операций над данными.

В АЛУ выполняются операции с фиксированной точкой; с плавающей точкой; операции двоично – десятичной арифметики, логические операции, обработка алфавитно-цифровой информации.

2) Устройство управления (УУ) – предназначается для выработки управляющих сигналов, под воздействие которых выполняются машинные команды.

3) Регистры общего назначения (РОНы)

РОНы – это программно-адресуемые регистры, предназначаются для хранения операндов, результатов, а также для хранения индекса, базы, которые используются при вычислении адреса, и др.

4) Управляющие регистры

Счётчик команд (СчК) – хранит адрес следующей выполняемой команды;

Регистр команд (РК) – хранит текущую выполняемую команду.

5) Вспомогательные блоки – к ним относятся: блок прерывания; блок связи центрального процессора (ЦП) и оперативной памяти (ОП), блок контроля и диагностики и т.д.

Ниже рассматривается функционирование центрального процессора при выполнении арифметико-логической команды, включающей в себя основные этапы выполнения машинных команд.

### **Основные этапы выполнения арифметико-логической машинной команды**

1 этап: выбор машинной команды из памяти

2 этап: дешифрация кода операции

3 этап: формирование исполнительного адреса и выбор операндов

4 этап: непосредственное выполнение операции в АЛУ

5 этап: запись результата

### Команды ЭВМ

Машинная команда – это двоичный код, который включает в себя операционную часть и адресную часть.

В адресной части содержится информация об адресах операндов и результатов.

Различаются следующие команды:

- 4-х адресная,
- 3-х адресная,
- 2-х адресная,
- одноадресная,
- безадресная.

#### 4-х адресные команды

Формат 4-х адресной команды представлен на рис. 3, где:

Поле A1 – информация об адресе первого операнда;

Поле A2 – информация об адресе второго операнда;

Поле A3 – адрес результата;

Поле A4 – адрес следующей выполняемой команды.

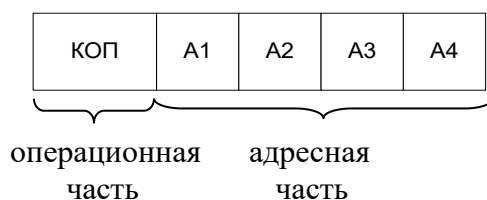


Рис. 3. Формат 4-х адресной команды

В целях обучения для простоты понимания в дальнейшем будем считать, что длина команды, операндов, результатов и др. равна ширине выборки из оперативной памяти и центральный процессор напрямую обращается к памяти.

Микропрограмма и структурная схема операционной части ЦП при  
выполнении 4-х адресной команды

Пусть A1 – адрес ячейки оперативной памяти, где хранится первый операнд;

A2 – адрес ячейки оперативной памяти, где хранится второй операнд;

A3 – адрес ячейки оперативной памяти, по которому необходимо записать результат;

A4 – адрес ячейки оперативной памяти, где хранится следующая выполняемая команда.

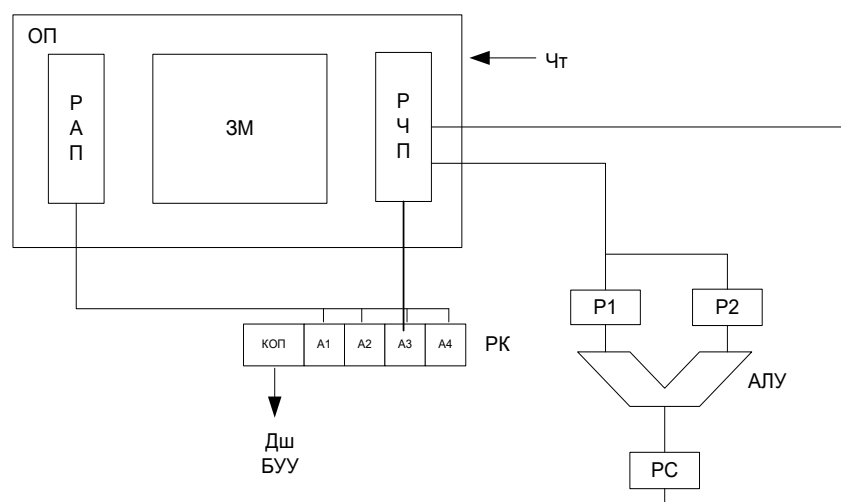


Рис. 4. Структурная схема операционной части центрального процессора при  
выполнении 4-х адресной команды

Рассмотрим структурную схему операционной части ЦП при выполнении 4-х адресной команды (рис. 4), где:

ЗМ – запоминающий массив;

РАП – регистр адреса памяти;

РЧП – регистр числа памяти;

РК – регистр команд;

АЛУ – арифметико-логическое устройство;

P1, P2 – входные регистры АЛУ, предназначаются для хранения операндов;

РС – выходной регистр, предназначается для хранения результата выполненной операции в АЛУ.



Рассмотрим микропрограмму выполнения 4-х адресной команды согласно структурной схеме (рис. 4).

**1 этап.** Выбирается команда из памяти

РАП:=РК (А4)

Адрес следующей выполняемой команды подается в память на регистр адреса памяти (РАП)

РЧП:=Чт (РАП) (Выбор команды на РЧП)

РК:=РЧП

Пересылается команда из памяти в ЦП на РК.

**2 этап.** Дешифрация кода операции.

**3 этап.** Формируется исполнительный адрес и выбираются операнды.

Выбор 1-го операнда	{	РАП:=РК (А1) – Адрес первого операнда подаётся в ОП на РАП
		РЧП:=Чт (РАП) – Чтение из ЗМ на РЧП первого операнда.
		Р1:=РЧП – Первый операнд подаётся на входной регистр АЛУ.
Выбор 2-го операнда	{	РАП:=РК (А2) – Адрес второго операнда подаётся в ОП на РАП
		РЧП:=Чт (РАП) – Чтение из ЗМ на РЧП второго операнда.
		Р2:=РЧП – Второй операнд подаётся на входной регистр АЛУ.

**4 этап.** Выполняется операция в АЛУ

РС:=Р1операция Р2

**5 этап.** Записывается результат

РАП:=РК (А3) – Адрес, по которому необходимо записать результат, подается в ОП на РАП

РЧП:=РС – Результат подается в ОП на РЧП

Зп (РАП) :=РЧП – Результат записывается в ЗМ

### 3-х адресные команды

Формат 3-х адресной команды представлен на рис. 5.



Рис. 5. Формат 3-х адресной команды

В 3-х адресной команде отсутствует А4. При использовании 3-х адресной команды добавляется дополнительное устройство – счётчик команд.

### Микропрограмма и структурная схема операционной части ЦП при выполнении 3-х адресной команды.

Рассмотрим структурную схему операционной части ЦП при выполнении 3-х адресной команды (рис. 6).

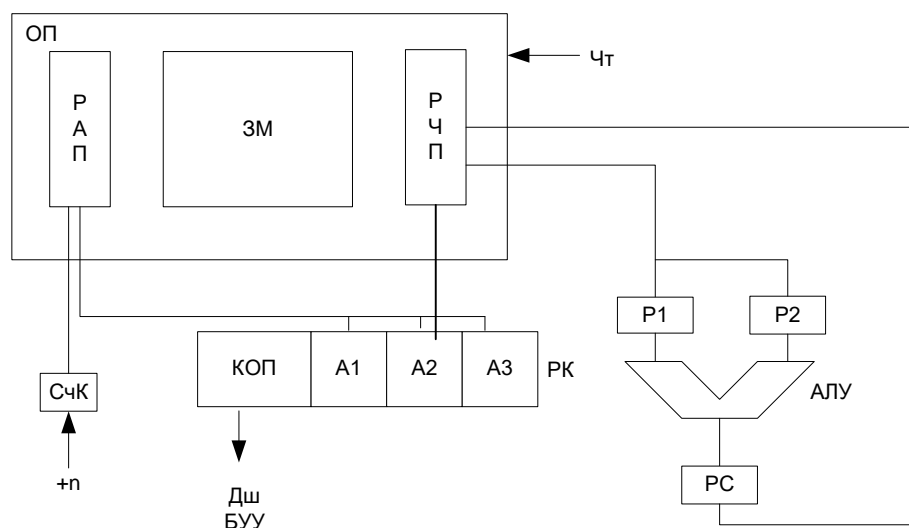


Рис. 6. Структурная схема операционной части ЦП при выполнении 3-х адресной команды

СчК – счётчик команд, хранит адрес следующей команды

Рассмотрим микропрограмму выполнения 3-х адресной команды согласно структурной схеме (рис. 6).

**1 этап.** РАП:=СчК

РЧП:=Чт (РАП)

РК:=РЧП

СчК:=СчК + n {n-длина команды}

**2,3,4,5 этапы** – аналогичны выполнению 4-х адресной команды.

### 2-х адресные команды

Формат 2-х адресной команды представлен на рис. 7.



Рис. 7. Формат 2-х адресной команды

Поле А3 – отсутствует. Результат записывается на место 1-го или 2-го операнда.

При этом операнд затирается и если его необходимо использовать в дальнейшем, он предварительно должен быть сохранён.

### Одноадресные команды

Формат одноадресной команды представлен на рис. 8.



Рис. 8. Формат одноадресной команды

Есть только поле A1.

В структуру ЦП вводится дополнительный регистр аккумулятора (РА), на который предварительно дополнительной командой засылается второй операнд.

Результат записывается на место либо первого операнда, либо в аккумулятор.

### Безадресные команды

Выполнение безадресной команды представлено на рис. 9.

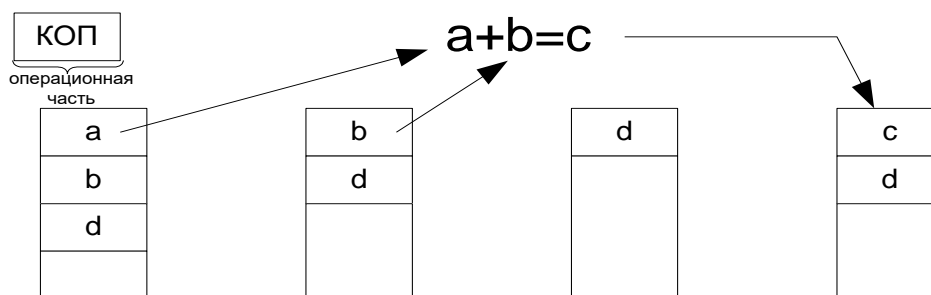


Рис. 9. Выполнение безадресной команды

Имеется только поле кода операции.

Стек используется для хранения операндов и записи результата.

### **Способы адресации**

Различают понятия: адресный код в команде и исполнительный адрес операнда.

Адресный код в команде – это информация об адресе.

Исполнительный адрес (ИА) операнда – это физический адрес ячейки памяти, в которой хранится операнд или в которую необходимо записать результат.

Рассмотрим способы адресации для операндов и результата [3].

### 1) Прямая адресация

В адресном поле команды  $A_i$  задаётся исполнительный адрес операнда (рис.10).

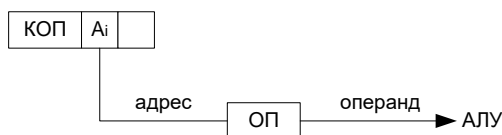


Рис. 10. Выбор операнда при прямой адресации

$$\text{Выбор операнда} \begin{cases} \text{РАП} := \text{РК} (A_i) \\ \text{РЧП} := \text{ЧТ} (\text{РАП}) \\ \text{Р1} := \text{РЧП} \end{cases}$$

Р1-входной регистр АЛУ/

### 2) Непосредственная адресация

В адресном поле команды задается непосредственно сам операнд (рис. 11).

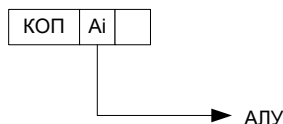


Рис. 11. Выбор операнда при непосредственной адресации

$$\text{Р1} := \text{РК} (A_i)$$

### 3) Косвенная адресация

В адресном поле команды задается адрес ячейки оперативной памяти (ОП), в которой хранится адрес операнда (адрес адреса) (рис. 12).

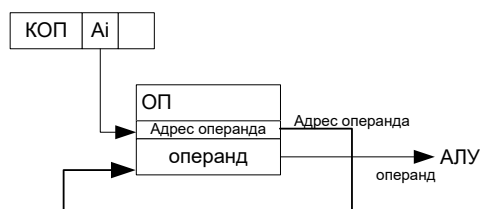


Рис. 12. Выбор операнда при косвенной адресации

$$\text{РАП} := \text{РК} (A_i)$$

$$\text{РЧП} := \text{ЧТ} (\text{РАП}) \quad - \text{выбор адреса операнда из ЗМ}$$

$$\text{РАП} := \text{РЧП}$$

$$\text{РЧП} := \text{ЧТ} (\text{РАП}) \quad - \text{выбор операнда из ЗМ}$$

$$\text{Р1} := \text{РЧП}$$

### 4) Регистровая адресация

В структуру ЦП дополнительно вводятся регистры общего назначения (РОНы), которые обеспечивают:

- а. Сокращение времени выбора операндов по сравнению с ОП;
- б. Сокращение длины команды  $A_i$  для кодирования номера РОНа.

Микропрограмма и структурная схема операционной части ЦП при выполнении двухадресной команды формата регистр-регистр (R-R).

При использовании регистровой адресации в адресном поле команд  $A_i$  задается номер регистра, где хранится операнд (рис. 13).

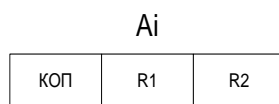


Рис. 13. Формат команды R-R

При написании микропрограммы принимается, что запись результата производится на место первого операнда.

Рассмотрим структурную схему операционной части ЦП при выполнении 2-х адресной команды (рис. 14), где РК – регистр команд – предназначается для хранения текущей выполняемой команды.

Рассмотрим микропрограмму выполнения 2-х адресной команды согласно структурной схеме (рис. 14).

**1 этап.** Выбор команды из памяти

$РАП := СчК$  – адрес следующей команды подается на РАП

$РЧП := Чт(РАП)$  – выбор команды из ЗМ

$РК := РЧП$  – машинная команда из памяти передается на РК в ЦП

$СчК := СчК + n$  –  $n$  – длина выполняемой машинной команды в байтах

**2 этап.** Дешифрация кода операции

**3 этап.** Формирование исполнительного адреса и выбор операндов.

Операнды находятся в РОНах

Выбор 1-го операнда	{	$РАП := РК(R1)$ – на РАП из РК подается номер регистра, где хранится 1-ый операнд $РЧРП := Чт(РАП)$ – выбор первого операнда из РОНов $Р1 := РЧРП$ – пересылка первого операнда на входной регистр АЛУ
Выбор 2-го операнда	{	$РАП := РК(R2)$ – на РАП из РК подается номер регистра, где хранится 2-ой операнд $РЧРП := Чт(РАП)$ – выбор второго операнда из РОНов $Р2 := РЧРП$ – пересылка второго операнда на входной регистр АЛУ

**4 этап.** Выполнение операции в АЛУ

PC := P1 операция P2

**5 этап.** Запись результата

РАРП := ПК (R1)

РЧРП := PC

Зп (РАРП) := РЧРП

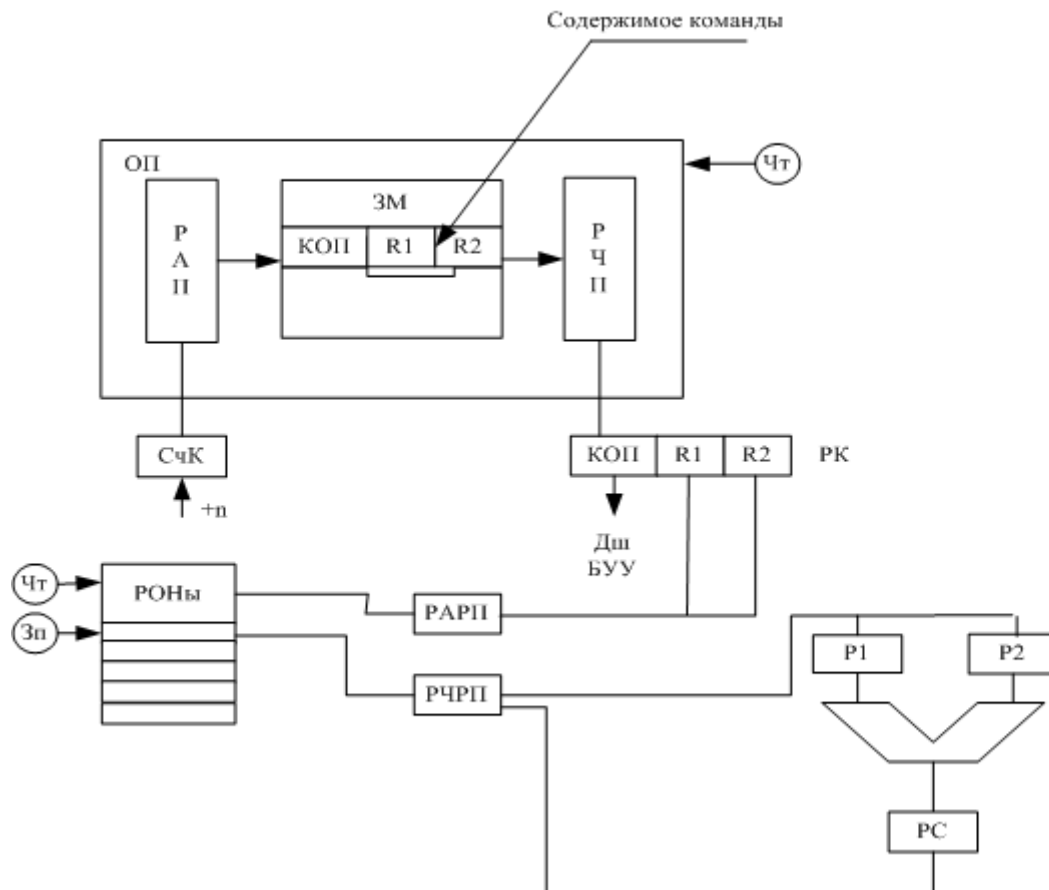


Рис. 14. Структурная схема операционной части центрального процессора при выполнении 2-х адресной команды формата регистр-регистр

#### б) Базовая адресация

Формат команды с базовой адресацией представлен на рис. 15, где:

$D_i$  – смещение;

$B_i$  – номер базового регистра.

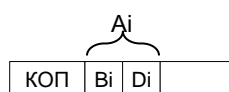


Рис. 15. Формат команды с базовой адресацией

Исполнительный адрес (ИА) - в адресном поле явно не задан и определяется по формуле:

$$ИА = (Bi) + Di,$$

Где (Bi) – база (содержимое базового регистра).

Базовая адресация – обеспечивает перемещаемость программ в памяти.

При базовой адресации операнды хранятся в ОП. Для вычисления исполнительного адреса операнда необходимо к содержимому базового регистра прибавить смещение, которое выбирается из адресного поля команды (рис. 16).

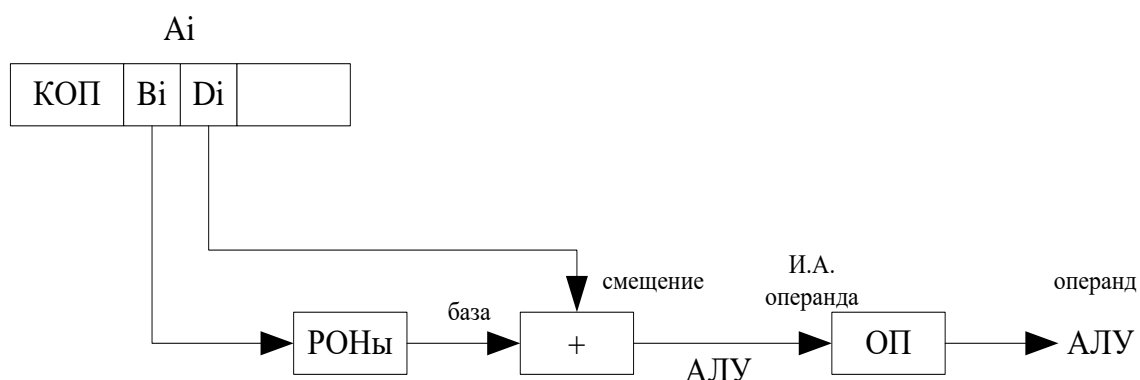


Рис. 16. Выбор операнда при базовой адресации

## 7) Индексная адресация

Формат команды с индексной адресацией представлен на рис. 17, где:

$X_i$  – номер индексного регистра.



Рис. 17. Формат команды с индексной адресацией

Исполнительный адрес операнда с индексной адресацией находится по формуле:

$$ИА = (X_i) + D_i,$$

где ( $X_i$ ) – индекс (содержимое индексного регистра).

Индексная адресация - используется при работе с массивами, таблицами.

Исполнительный адрес операнда вычисляется аналогично базовой адресации (рис. 18).

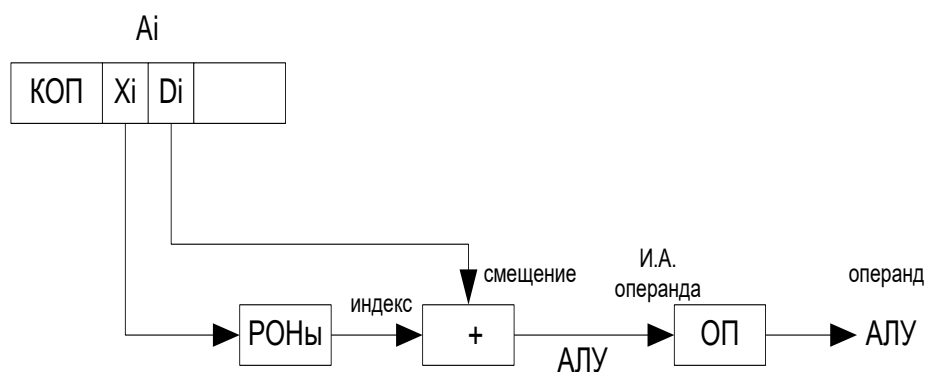


Рис. 18. Выбор операнда при индексной адресации

#### 8) Базово – индексная адресация

Формат команды с базово-индексной адресацией представлен на рис. 19.

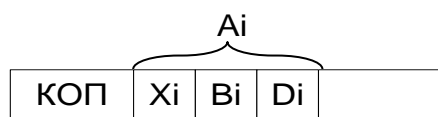


Рис. 19. Формат команды с базово-индексной адресацией

Исполнительный адрес операнда с базово-индексной адресацией находится по формуле:

$$\text{ИА} = (\text{Xi}) + (\text{Bi}) + \text{Di},$$

где:

(Xi) – индекс (содержимое индексного регистра);

(Bi) – база (содержимое базового регистра);

Di – смещение выбирается из команды и подаётся на АЛУ (рис. 20).



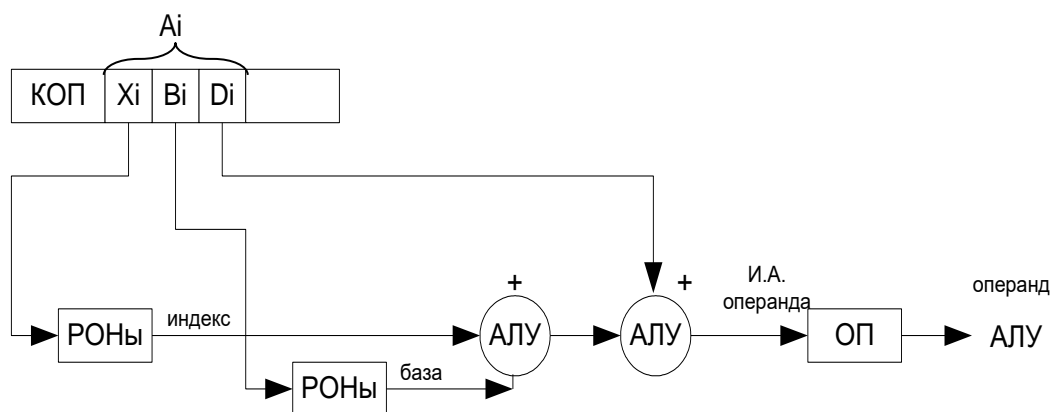


Рис. 20. Выбор операнда при базово-индексной адресации

Микропрограмма и структурная схема операционной части ЦП при выполнении двухадресной команды с записью результата на место первого операнда.

Примем, что для 1-ого операнда используется регистровая адресация, для 2-ого операнда – базово-индексная адресация. Структурная схема операционной части ЦП при выполнении 2-х адресной команды при использовании регистровой и базово-индексной адресации с записью результата на место первого операнда представлена на рис. 21.



$PC = R1 + R2$  - вычисление ИА  
 выбор 2-го операнда:  
 $РАП := PC$  - ИА подаётся на РАП  
 $РЧП := ЧТ(РАП)$  - выбор 2-го операнда  
 $R1 := РЧП$   
 б) выбор 1-го операнда:  
 $РАРП := РК(R1)$   
 $РЧРП := ЧТ(РАРП)$   
 $R2 := РЧРП$

**4 этап.** Выполнение операции в АЛУ

$PC := PC$  операция  $R2$

**5 этап.** Запись результатов

$РЧРП := PC$   
 $Зп(РАРП) := РЧРП$

## 9) Косвенно – регистровая адресация

Формат команды с косвенно-регистровой адресацией представлен на рис. 22.



Рис. 22. Формат команды с косвенно-регистровой адресацией

В адресном поле команды указывается номер РОНа, который содержит адрес операнда, хранящийся в ОП (оперативной памяти) (рис. 23).



Рис. 23. Выбор операнда при косвенно-регистровой адресации

$РАРП := РК(R1)$   
 $РЧРП := ЧТ(РАРП)$  - выбор адреса операнда из РОНов  
 $РАП := РЧРП$   
 $РЧП := ЧТ(РАП)$  - выбор операнда из ОП  
 $R1 := РЧП$

Если нужно записать результат на место операнда с косвенно – регистровой адресацией, то результат записывается в ОП, а не в РОНЫ.

## Задание по теме № 2.

Составить микропрограмму функционирования центрального процессора при выполнении 2-х адресной команды. Использовать способы адресации согласно заданию (табл. 3).

Начертить структурную схему операционной части центрального процессора.

Таблица 3.

### Способы адресации операндов и запись результата

п/п	1ый операнд	2ой операнд	Результат
1	Регистровая	Прямая	1ый операнд
2	Регистровая	Косвенная	1ый операнд
3	Регистровая	Косвенно-регистровая	1ый операнд
4	Регистровая	Базовая	1ый операнд
5	Прямая	Регистровая	1ый операнд
6	Прямая	Косвенная	1ый операнд
7	Прямая	Косвенно-регистровая	1ый операнд
8	Прямая	Базовая	1ый операнд
9	Косвенная	Регистровая	1ый операнд
10	Косвенная	Косвенно-регистровая	1ый операнд
11	Косвенная	Базовая	1ый операнд
12	Косвенная	Непосредственная	1ый операнд
13	Косвенная	Прямая	1ый операнд
14	Косвенно-регистровая	Регистровая	1ый операнд
15	Косвенно-регистровая	Прямая	1ый операнд
16	Косвенно-регистровая	Косвенная	1ый операнд
17	Косвенно-регистровая	Индексная	1ый операнд
18	Базовая	Регистровая	1ый операнд
19	Базовая	Прямая	1ый операнд
20	Базовая	Косвенная	1ый операнд
21	Базовая	Косвенно-регистровая	1ый операнд
22	Базовая	Непосредственная	1ый операнд
23	Прямая	Регистровая	2ой операнд
24	Косвенная	Регистровая	2ой операнд
25	Косвенно-регистровая	Регистровая	2ой операнд
26	Индексная	Регистровая	2ой операнд
27	Регистровая	Прямая	2ой операнд
28	Косвенная	Прямая	2ой операнд
29	Косвенно-регистровая	Прямая	2ой операнд
30	Базовая	Прямая	2ой операнд

31	Регистровая	Косвенная	2ой операнд
32	Косвенно-регистровая	Косвенная	2ой операнд
33	Базовая	Косвенная	2ой операнд
34	Прямая	Косвенная	2ой операнд
35	Непосредственная	Косвенно-регистровая	2ой операнд
36	Регистровая	Косвенно-регистровая	2ой операнд
37	Прямая	Косвенно-регистровая	2ой операнд
38	Косвенная	Косвенно-регистровая	2ой операнд
39	Индексная	Косвенно-регистровая	2ой операнд
40	Регистровая	Базовая	2ой операнд
41	Прямая	Базовая	2ой операнд
42	Косвенная	Базовая	2ой операнд
43	Косвенно-регистровая	Базовая	2ой операнд
44	Непосредственная	Базовая	2ой операнд
45	Прямая	Регистровая	1ый операнд
46	Непосредственная	Косвенно-регистровая	2ой операнд

### ТЕМА № 3. ОРГАНИЗАЦИЯ УСТРОЙСТВА УПРАВЛЕНИЯ.

Центральный процессор, как и любое другое устройство обработки цифровой информации, включает в себя две основные части:

- операционную часть (операционное устройство);
- управляющую часть (устройство управления).

Операционная часть состоит из регистров, счётчиков, сумматоров, дешифраторов и связей между ними. Операционная часть функционирует под воздействием управляющих сигналов, которые вырабатывает управляющее устройство. Операционная часть выполняет заданную микропрограмму, состоящую из микрокоманд.

Микрокоманда включает в себя одну или несколько микроопераций. Микрооперация – это элементарная функциональная операция, выполняемая под воздействием одного управляющего сигнала в течение одного такта. Если в течение одного такта выполняется несколько микроопераций под воздействием различных управляющих сигналов, то они могут быть объединены в одну микрокоманду.

Устройство управления (УУ) служит для выработки последовательности управляющих сигналов, под воздействием которых выполняются микрооперации. В зависимости от способа выработки управляющего сигнала различают 2 основных подхода к построению УУ:

- микропрограммная реализация УУ;
- аппаратная реализация УУ (схемная реализация или УУ с жёсткой логикой) [2].

#### **Микропрограммная реализация устройства управления.**

Микропрограммная реализация устройства управления операционной частью ЦП определяется схемой Уилкса (рис. 24), где:

РАМК – регистр адреса микрокоманд;

РМК – регистр микрокоманд;

Дш – дешифратор;

Тр.з – триггер задержки.

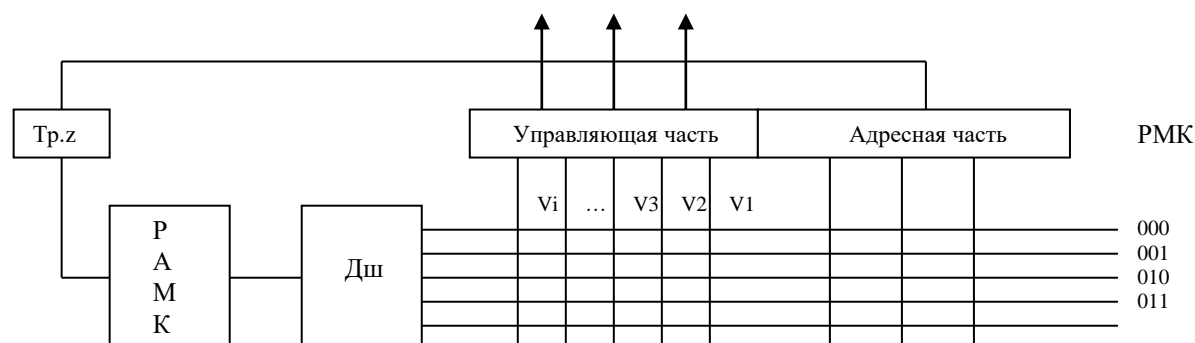


Рис. 24. Схема Уилкса

Микропрограмма, состоящая из микрокоманд, записывается в память микрокоманд. Каждая микрокоманда состоит из двух частей: управляющей части, где кодируются управляющие сигналы  $V_i$ , и адресной части.

В адресной части микрокоманды кодируется адрес ячейки памяти, где хранится следующая выполняемая микрокоманда. В начальный момент времени на РАМК подается адрес ячейки памяти, где хранится первая микрокоманда. По этому адресу из памяти микрокоманда считывается и подается на регистр микрокоманд.

Из управляющей части микрокоманды управляющие сигналы подаются на вентили в операционную часть ЦП, а из адресной части на регистр адреса микрокоманд в следующем такте заводится адрес следующей выполняемой микрокоманды.

Пример реализации устройства управления центральным процессором при выполнении двухадресной команды формата «регистр-регистр» (R-R).

Структурная схема операционной части ЦП при выполнении 2-х адресной команды формата R-R для построения устройства управления представлена на рис. 25, где:

Результат записывается на место первого операнда.

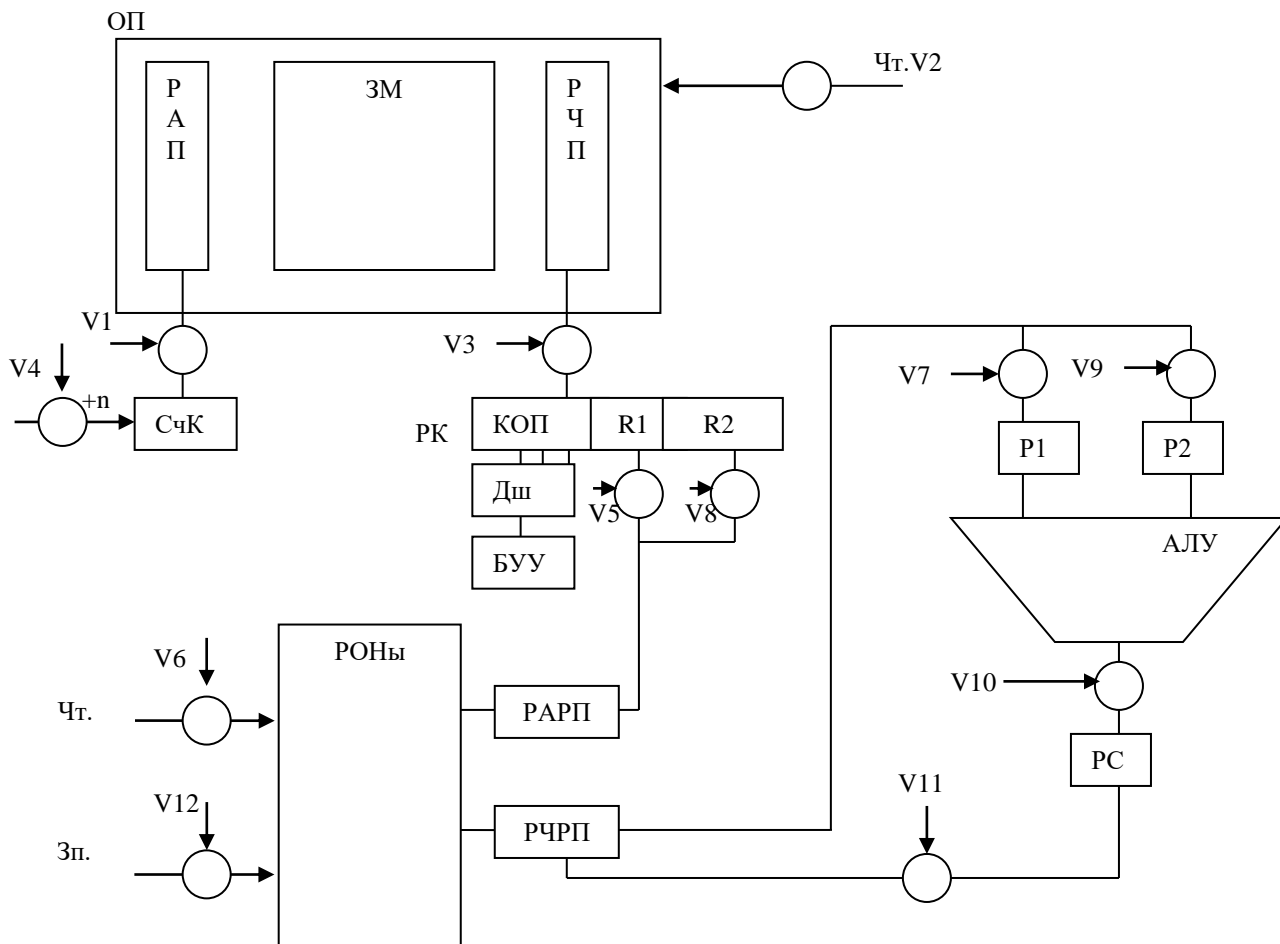


Рис. 25. Структурная схема операционной части центрального процессора при выполнении 2-х адресной команды формата R-R для построения устройства управления

Рассмотрим микропрограмму выполнения 2-х адресной команды формата R-R согласно структурной схеме (рис. 25).

Операнды находятся в РОНах.



Выбор 1-го операнда  $\left\{ \begin{array}{l} V5 \text{ РАРП} := \text{РК}(\text{R1}) - \text{на РАРП из РК подаётся номер регистра,} \\ \text{где хранится 1-ый операнд} \\ V6 \text{ РЧРП} := \text{ЧТ}(\text{РАРП}) - \text{выбор первого операнда из РОНов} \\ V7 \text{ Р1} := \text{РЧРП} - \text{пересылка первого операнда на входной регистр} \\ \text{АЛУ} \end{array} \right.$

Выбор 2-го операнда  $\left\{ \begin{array}{l} V8 \text{ РАРП} := \text{РК}(\text{R2}) - \text{на РАРП из РК подаётся номер регистра,} \\ \text{где хранится 2-ой операнд} \\ V6 \text{ РЧРП} := \text{ЧТ}(\text{РАРП}) - \text{выбор второго операнда из РОНов} \\ V9 \text{ Р2} := \text{РЧРП} - \text{пересылка второго операнда на входной регистр} \\ \text{АЛУ} \end{array} \right.$

**4 этап.** Выполнение операции в АЛУ.

V10  $\text{РС} := \text{Р1}$  операция Р2

**5 этап.** Запись результатов.

V5  $\text{РАРП} := \text{РК}(\text{R1})$  - адрес 1-го операнда, по которому будет записан результат, подаётся в РАРП

V11  $\text{РЧРП} := \text{РС}$  - пересылка результата в регистр РЧРП

V12  $\text{Зп}(\text{РАРП}) := \text{РЧРП}$  - запись результата в R1

Отметим, что 4 этап представлен условно 1 микрооперацией. При рассмотрении функционирования АЛУ 4 этап заменяется микропрограммой.

Совместим микрооперации во времени. Рассмотрим микропрограмму после совмещения микроопераций.

**1 этап.** Выбор команды из памяти.

V1  $\text{РАП} := \text{СчК}$

V2  $\text{РЧП} := \text{ЧТ}(\text{РАП})$

V3, V4  $\text{РК} := \text{РЧП}, \text{СчК} := \text{СчК} + n$

**2 этап.** Дешифрация кода операции.

**3 этап.** Формирование исполнительного адреса и выбор операндов.

V5  $\text{РАРП} := \text{РК}(\text{R1})$

V6  $\text{РЧРП} := \text{ЧТ}(\text{РАРП})$

V7, V8  $\text{Р1} := \text{РЧРП}, \text{РАРП} := \text{РК}(\text{R2})$

V6  $\text{РЧРП} := \text{ЧТ}(\text{РАРП})$

V9  $\text{Р2} := \text{РЧРП}$

**4 этап.** Выполнение операции в АЛУ.

V10  $\text{РС} := \text{Р1}$  операция Р2

**5 этап.** Запись результатов.

V5, V11  $\text{РАРП} := \text{РК}(\text{R1}), \text{РЧРП} := \text{РС}$

V12  $\text{Зп}(\text{РАРП}) := \text{РЧРП}$

### Горизонтальный подход к реализации микропрограммного устройства управления (УУ).

При горизонтальном микропрограммировании для каждого управляющего сигнала в управляющей части выделяется отдельный разряд. Если в некотором

такте управляющий сигнал должен быть равен 1, то в соответствующем разряде записывается 1. Таким образом, количество разрядов в управляющей части соответствует числу управляющих сигналов. При такой организации можно совмещать микрооперации во времени.

Достоинство: высокое быстродействие за счёт совмещения микроопераций во времени.

Недостатки: требуется большая ёмкость памяти для хранения микрокоманды, так как реально совместить большое количество микроопераций в одной микрокоманде не удаётся.

Пример: реализация горизонтального микропрограммного УУ при выполнении 2-х адресной команды формата R-R. Срабатывание вентилей по тактам работы ЦП представлено в табл. 4.

Таблица 4.

Срабатывание вентилей по тактам при горизонтальном подходе реализации микропрограммного УУ

Вентили	Такты
V1	1
V2	2
V3, V4	3
V5	4
V6	5
V7, V8	6
V6	7
V9	8
V10	9
V5, V11	10
V12	11

Пример реализации горизонтального микропрограммного УУ при выполнении 2-х адресной команды формата R-R представлен на рис. 26.

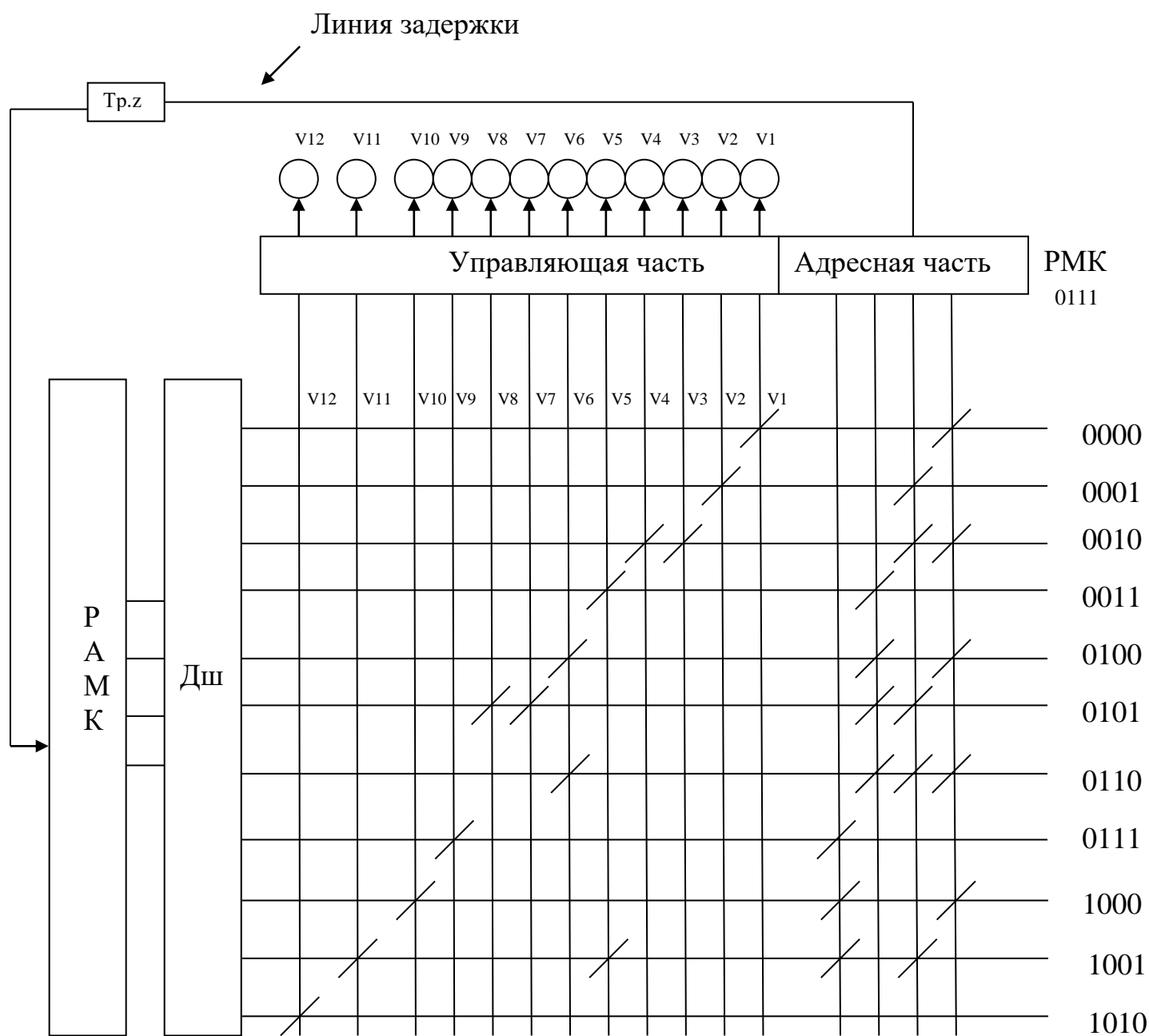


Рис. 26. Пример реализации горизонтального микропрограммного устройства управления

### Вертикальный подход к реализации микропрограммного устройства управления.

При вертикальном микропрограммировании в операционной части МК кодируется номер управляющего сигнала, поэтому на выходе регистра микрокоманд в управляющей части ставится дешифратор.

Достоинство: значительно сокращается емкость памяти микрокоманд.

Недостатки: увеличенное время выполнения микрокоманд из-за невозможности совмещения микроопераций в микрокоманде и появления дешифратора на выходе.

Пример: реализация вертикального микропрограммного УУ при выполнении 2-х адресной команды формата R-R. Срабатывание вентилей по тактам работы ЦП представлено в табл. 5.

Таблица 5.

Срабатывание вентилей по тактам при вертикальном подходе реализации микропрограммного УУ.

Вентили	Такты
V1	1
V2	2
V3	3
V4	4
V5	5
V6	6
V7	7
V8	8
V6	9
V9	10
V10	11
V5	12
V11	13
V12	14

Пример реализации вертикального микропрограммного УУ при выполнении 2-х адресной команды формата R-R представлен на рис. 27.

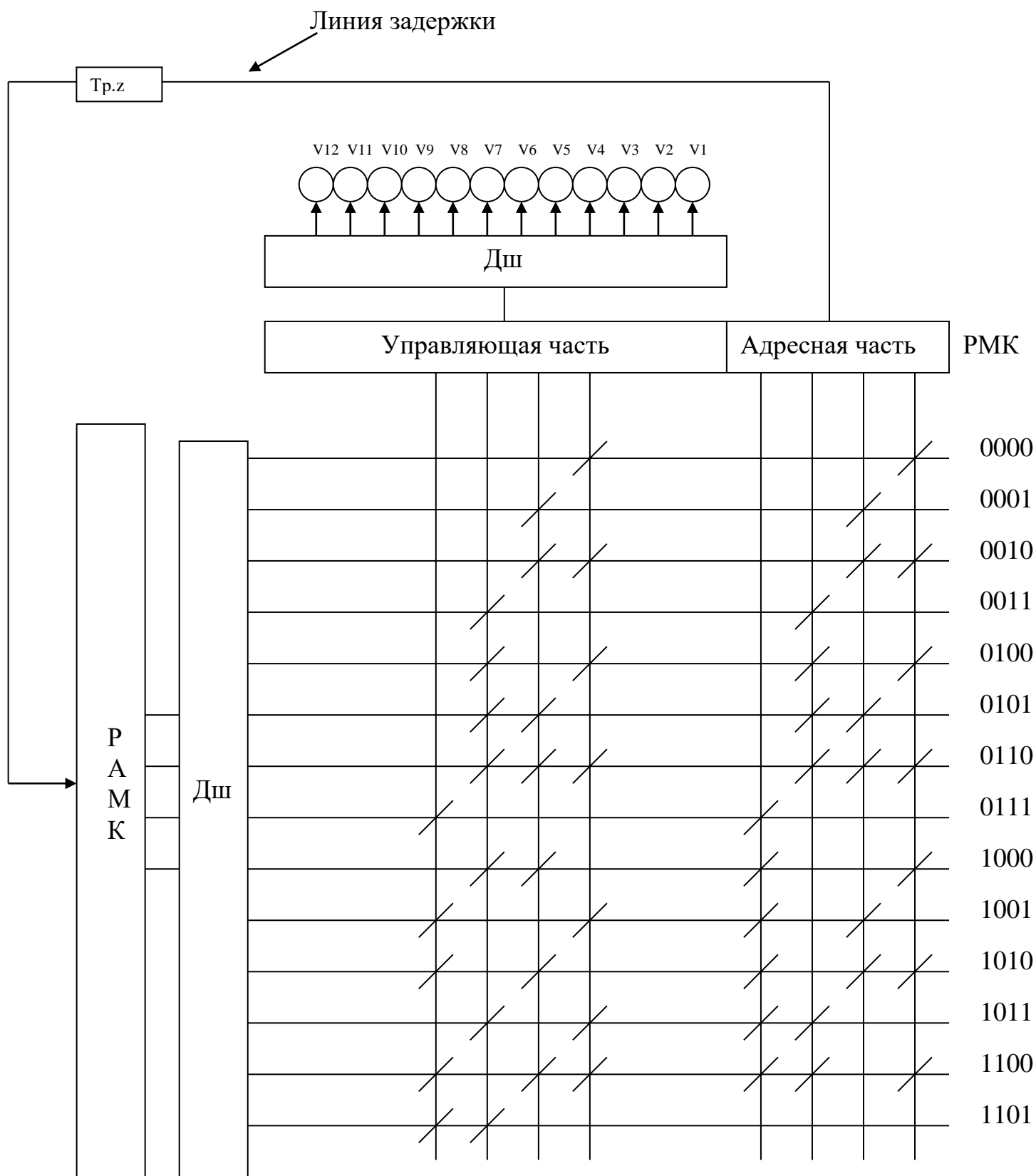


Рис. 27. Пример реализации вертикального микропрограммного устройства управления

### Задание по теме № 3.

Разработать микропрограммное устройство управления для операционной части ЦП в соответствии с заданием по теме № 2.

Использовать горизонтальный и вертикальный подход для реализации микропрограммного устройства управления.

## РАЗРАБОТКА МИКРОПРОГРАММНОГО УСТРОЙСТВА УПРАВЛЕНИЯ ОПЕРАЦИОННОЙ ЧАСТЬЮ ЦЕНТРАЛЬНОГО ПРОЦЕССОРА. КУРСОВОЕ ПРОЕКТИРОВАНИЕ

На базе рассмотренного теоретического материала и приведенных многочисленных выше примеров предлагается в рамках выполнения курсового проекта разработать микропрограммное устройство управления операционной частью центрального процессора при выполнении машинной команды с заданными способами адресации и выполняемой операцией в арифметико-логическом устройстве в соответствии с заданием.

Рассмотренные принципы разработки и функционирования устройства управления лежат в основе проектирования микропрограммного устройства управления операционной частью цифрового устройства обработки информации.

В отличие от ранее рассмотренных примеров разрабатываемое устройство управления включает команды перехода на микропрограммном уровне. Рассмотрим особенности выполнения команд перехода на микропрограммном уровне.

### Выполнение команды перехода на микропрограммном уровне.

При необходимости выполнения команды перехода на микропрограммном уровне адрес следующей выполняемой микрокоманды состоит из 2-х частей:

1. Основной (базовой) части (адреса микрокоманды), который выбирается (хранится) в адресном поле микрокоманды перехода;

2. Значений признаков триггеров, которые определяют младшую часть адреса микрокоманды. Значения признаков триггеров формируются в операционной части центрального процессора. Таким образом, при использовании одного признакового триггера, при выполнении операции перехода на микропрограммном уровне в зависимости от условия, которое содержится на признаковом триггере, будет сформировано два адреса, отличающихся младшими разрядами (см. пример выполнения операции умножения в АЛУ).

Адрес микрокоманды:

Основная часть (базовая)	Признаковый триггер (младшая часть адреса)
-----------------------------	---

Рассмотрим реализацию перехода на микропрограммном уровне на примере умножения чисел с фиксированной точкой, представленных в прямом коде.

#### Алгоритм умножения [1].

В каждом цикле выполнения операции умножения анализируется очередная цифра множителя. Если очередная цифра множителя равна 1, то к сумме частичных произведений прибавляется множимое, в противном случае прибавляется ноль. Цикл завершается сдвигом множимого относительно суммы частичных произведений, либо сдвигом суммы частичных произведений относительно неподвижного множимого. Таким образом, выполнение операции умножения в АЛУ сводится к последовательности операций сложения и сдвига.

В случае отрицательного операнда при умножении чисел, представленных в прямом коде, операция умножения сводится к выполнению следующих этапов:

- Определение знака произведения путем сложения по модулю 2 знаковых разрядов множимого и множителя;
- Обнуление знаковых разрядов отрицательных операндов;
- Выполнение операции умножения чисел, представленных в прямом коде.

Существует четыре способа умножения чисел с фиксированной точкой. При выполнении операции умножения можно сдвигать либо множимое, либо промежуточный результат и начинать анализ множителя либо с младших разрядов, либо со старших.

Рассмотрим способ умножения чисел, представленных в прямом коде, начиная с младших разрядов множителя, со сдвигом суммы частичных произведений вправо и при неподвижном множимом.

На рис. 28 приведен фрагмент микропрограммы умножения, связанный с анализом младшего разряда множителя.

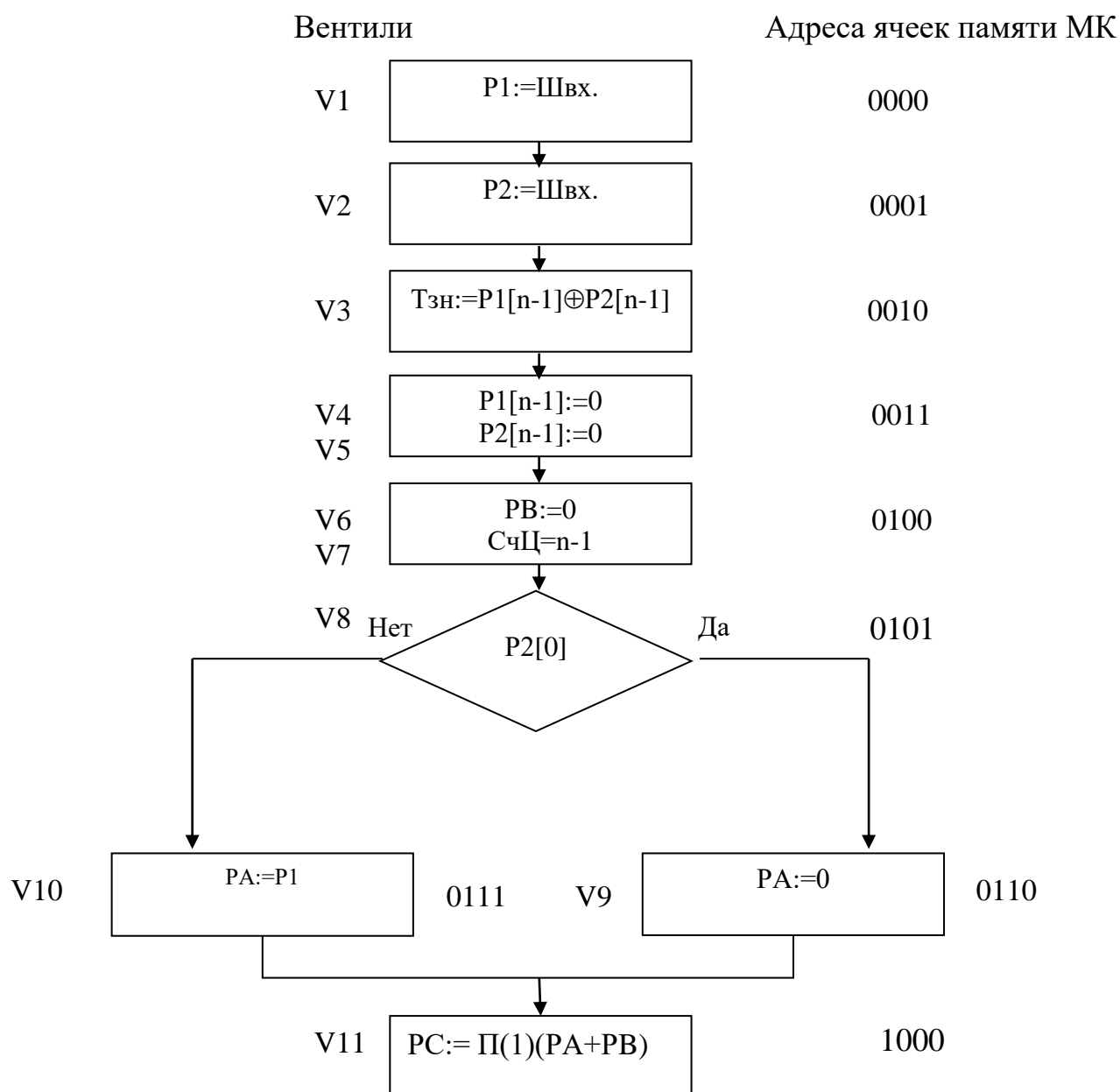




Рис. 28. Фрагмент схемы алгоритма выполнения операции умножения над числами с фиксированной точкой, представленных в прямом коде

Множимое записывается в P1, множитель в P2.

Формируется знак результата путем сложения по модулю два знаковых разрядов множимого и множителя.

Вначале выполнения операции умножения обнуляется регистр PB.

Счетчик циклов устанавливается равным количеству значащих разрядов множителя.

Анализируется младший разряд множителя: если равен 1, то в PA переписывается содержимое P1, если равен 0, то PA обнуляется.

Содержимое регистров PA и PB суммируется и записывается в регистр PC со сдвигом вправо.

В адресной части микрокоманды перехода содержится базовая часть. В качестве признакового триггера при выполнении операции выступает младший разряд множителя.

Базовая часть адреса в данном примере соответствует 011\_ (табл. 6).

Таблица 6.

Базовая часть	Признаковый триггер
011_	0 или 1, в зависимости от значения младшего разряда множителя

Тогда, если признаковый триггер равен 0, то адрес микрокоманды соответствует 0110, если признаковый триггер равен 1, то адрес микрокоманды соответствует 0111.

Если значение младшего разряда регистра P2 равно 0, то, в соответствии с алгоритмом, вырабатывается управляющий сигнал V9 - обнуление регистра PA (признаковый триггер в данном случае равен 0). Тогда микрокоманда, предназначенная для выработки сигнала V9, должна храниться в памяти по адресу 0110, так как в младший разряд адреса микрокоманды подставляется значение признакового триггера, т.е. 0.

Если же значение младшего разряда регистра P2 равно 1, то значение регистра P1 пересылается на РА и признаковый триггер будет равным 1. Это осуществляется под управлением сигнала V10. Поэтому эта микрокоманда должна храниться по адресу 0111.

После обнуления регистра РА под управлением V9 или передачи информации с регистра P1 на регистр РА под управлением V10 должна выполняться микрокоманда, формирующая управляющий сигнал V11, под управлением которого происходит сложение РА и РВ и сдвиг суммы на 1 разряд вправо. Микрокоманда, предназначенная для выработки сигнала V11, записана по адресу 1000, поэтому в микрокомандах, предназначенных для выработки управляющих сигналов V9 и V10, в адресном поле (рис. 29) содержится 1000.

РФА предназначен для формирования адреса с учётом команды перехода на микропрограммном уровне. При появлении команды перехода в микропрограмме старшая часть адреса выбирается из адресной части микрокоманды, а младшая часть соответствует признаковым триггерам в операционной части.

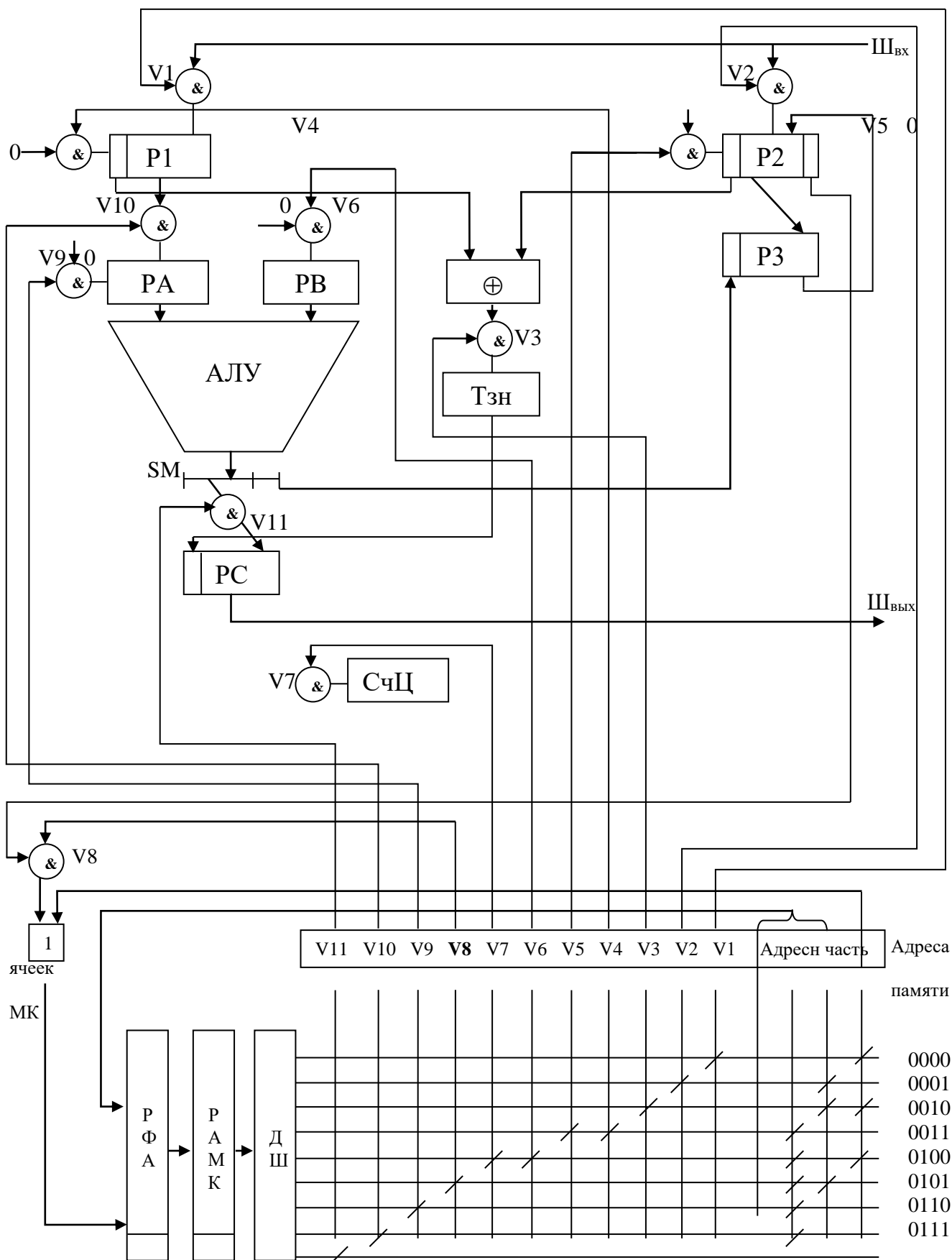


Рис. 29. Структурная схема АЛУ для выполнения операции умножения

### Задание на курсовое проектирование и порядок проведения работ

Разработать горизонтальное (вертикальное) микропрограммное устройство управления операционной частью ЦП при выполнении 2-х адресной команды с заданными способами адресации в соответствии с вариантом задания (см. задания по теме № 2). В АЛУ в зависимости от задания могут выполняться различные арифметико-логические операции, при использовании разных алгоритмов выполнения операций [1].

В процессе выполнения курсового проекта необходимо:

1. Разработать микропрограмму выполнения машинной команды с учетом заданных способов адресации и выполнения операции в АЛУ в соответствии с заданием без совмещения микроопераций во времени. Представить микропрограмму в виде структурной схемы алгоритма.
2. Разработать структурную схему операционной части ЦП.
3. Поставить управляющие сигналы в соответствии с микрооперациями на структурной схеме алгоритма и соответствующие вентили с указанием подведенных к ним управляющих сигналов на структурной схеме операционной части центрального процессора.
4. Для горизонтального микропрограммного устройства управления необходимо объединить независимые микрооперации во времени.
5. Разработать микропрограммный блок устройства управления в соответствии с заданием.
6. Разработать структурную схему центрального процессора на базе разработанной структурной схемы операционной части центрального процессора и блока устройства управления

### Требования по оформлению курсовой работы.

Курсовая работа представляется в печатном виде и должна содержать следующие разделы:

1. Задание по курсовой работе.
2. Введение (постановка задачи).
2. Краткое описание теории по заданной теме.

3. Разработанная микропрограмма операционной части устройства цифровой обработки данных без совмещения микроопераций, представленная в виде структурной схемы алгоритма.

4. Разработанная микропрограмма операционной части устройства цифровой обработки данных с учетом совмещения микроопераций во времени (при использовании горизонтального микропрограммного устройства управления), представленная в виде структурной схемы алгоритма.

6. Разработанная структурная схема операционной части устройства цифровой обработки данных.

5. Разработанный микропрограммный блок устройства управления (БУУ).

6. Разработанное устройство цифровой обработки данных на базе разработанной структурной схемы операционной части устройства цифровой обработки данных и БУУ.

7. Заключение (выводы).

8. Список использованных источников.

## ЛИТЕРАТУРА

1. Брехов О.М., Звонарева Г.А., Корнеевкова А.В., Клименко А.В. Организация центрального процессора : учеб. пособие / О.М. Брехов [и др.]; МАИ (Нац. исслед. ун-т). - Москва : МАИ, 2021. - 123 с. : ил. - (Учебное пособие). - Библиогр.: с.118 (3 назв.). - ISBN 978-5-4316-0806-3.

2. Орлов С.А. Цилькер Б.Я. Организация ЭВМ и систем. Фундаментальный курс по архитектуре и структуре современных компьютерных средств : учебник для вузов по направл. "Информатика и вычислит. техника" / С.А. Орлов, Б.Я. Цилькер. - 3-е изд. - СПб. : Питер, 2015. - 685 с. : ил. - (Учебник для вузов). - Библиогр.: с.663-671(191 назв.). - ISBN 978-5-496-01145-7.

3. Таненбаум Э. Архитектура компьютера / Э. Таненбаум. - Изд.4-е. - СПб. : Питер, 2006. - 698 с. : ил. - (Сер."Классика computer science"). - Библиогр.:с.647-664 . - ISBN 5-318-00298-6.