Процессор – элемент архитектуры

Процессор-устройство, осуществляющее процесс обработки данных и программное управление этим процессом.

Структура команды определяется составом, назначением и расположением полей в коде.

Форматом команды называется заранее оговоренная структура полей ее кода с разметкой номеров разрядов (бит), определяющих границы отдельных полей команды, или с указанием числа разрядов (бит) в определенных полях, позволяющая ЭВМ распознавать составные части кода.

Пример формата команды процессора i486.

mod r/m - спецификатор режима адресации;

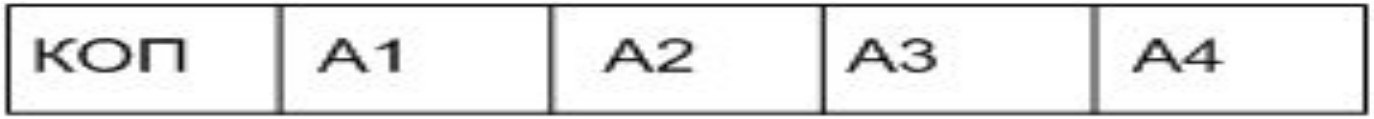
r/m - регистр памяти;

SS - масштабный множитель для режима масштабирования индексной адресации;

КОП - код операции;

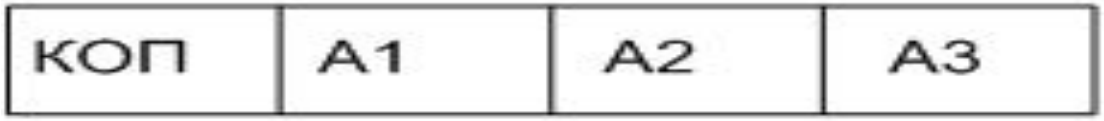
index - определяет индексный регистр;

base - определяет базовый регистр.

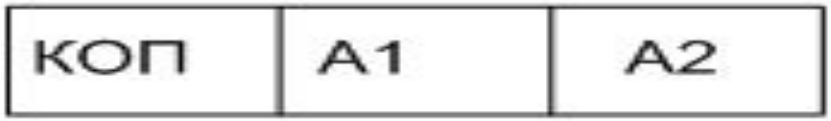


$n_{\text{коп}} \geq \log_2 M$

$n_{\text{адр}} \geq \log_2 S$



СчК



Роны, ПрТр

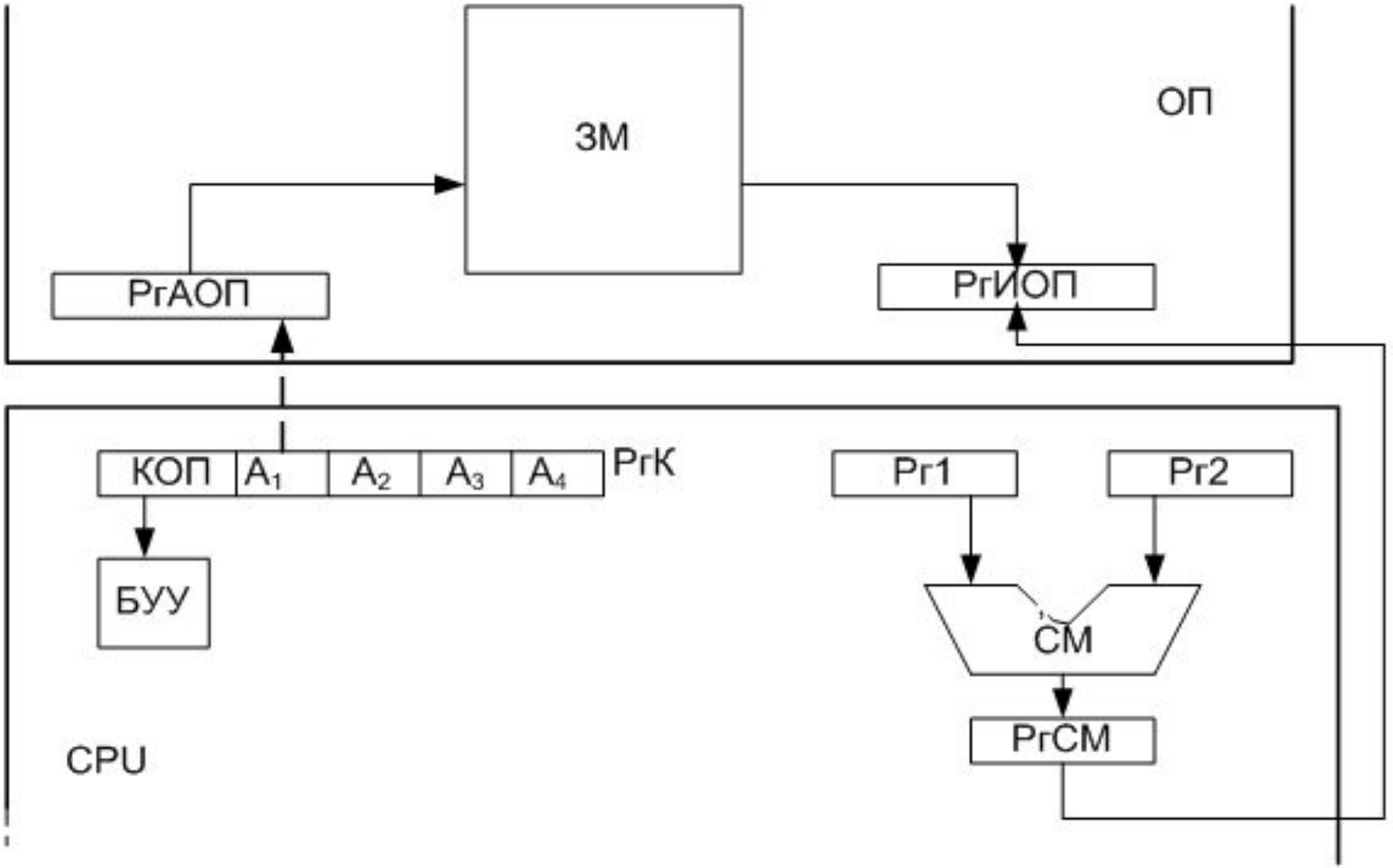
стек

Классификация команд

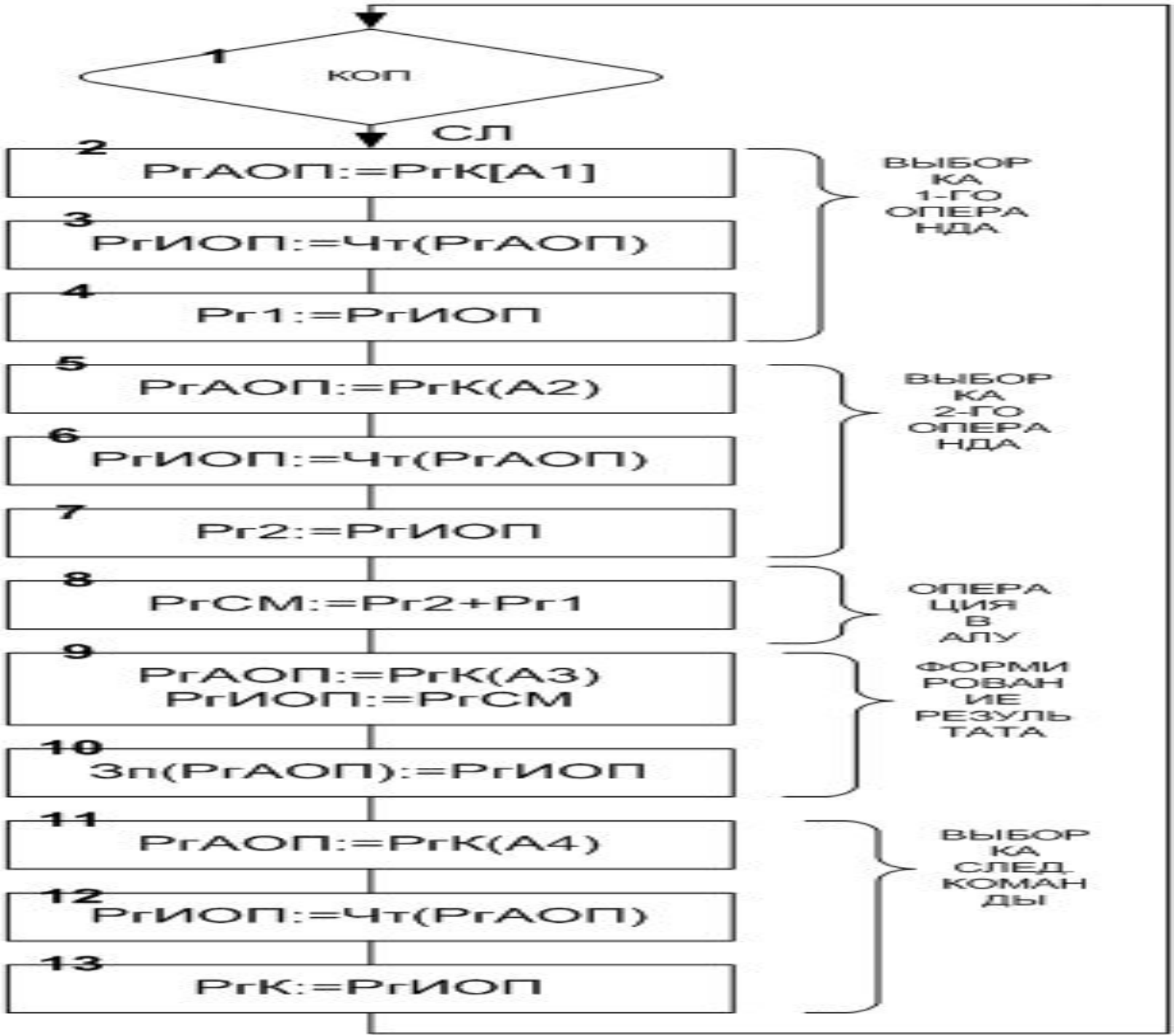
- По коду операции
  - ¨ команды арифметических операций для чисел с фиксированной или плавающей запятой;
  - ¨ команды десятичной арифметики;
  - ¨ команды передачи данных (MOV AX, 0FFFh);
  - ¨ команды операций ввода/вывода (IN, OUT);
  - ¨ команды логических операций (AND, OR, NOT);
  - ¨ команды передачи управления (управление циклом — LOOP, условные переходы — JAE, JB);
  - ¨ команды задания режима работы машины и др.
- По числу адресов в адресной части команды
  - а) безадресные (нульадресные);
  - б) одноадресные;
  - в) двухадресные;
  - г) трехадресные;
- по способу кодирования операции:
  - а) с фиксированным полем – в этом случае для кодирования F команд необходимо в поле КОП выделить int log2F + 1 двоичных разрядов;
  - б) с расширяющимся полем – в поле КОП ис-пользуется переменное число бит – пол-байта
- по длине команды:
  - для микропроцессоров –;
  - для мини-ЭВМ, супермини-ЭВМ и супер-ЭВМ –

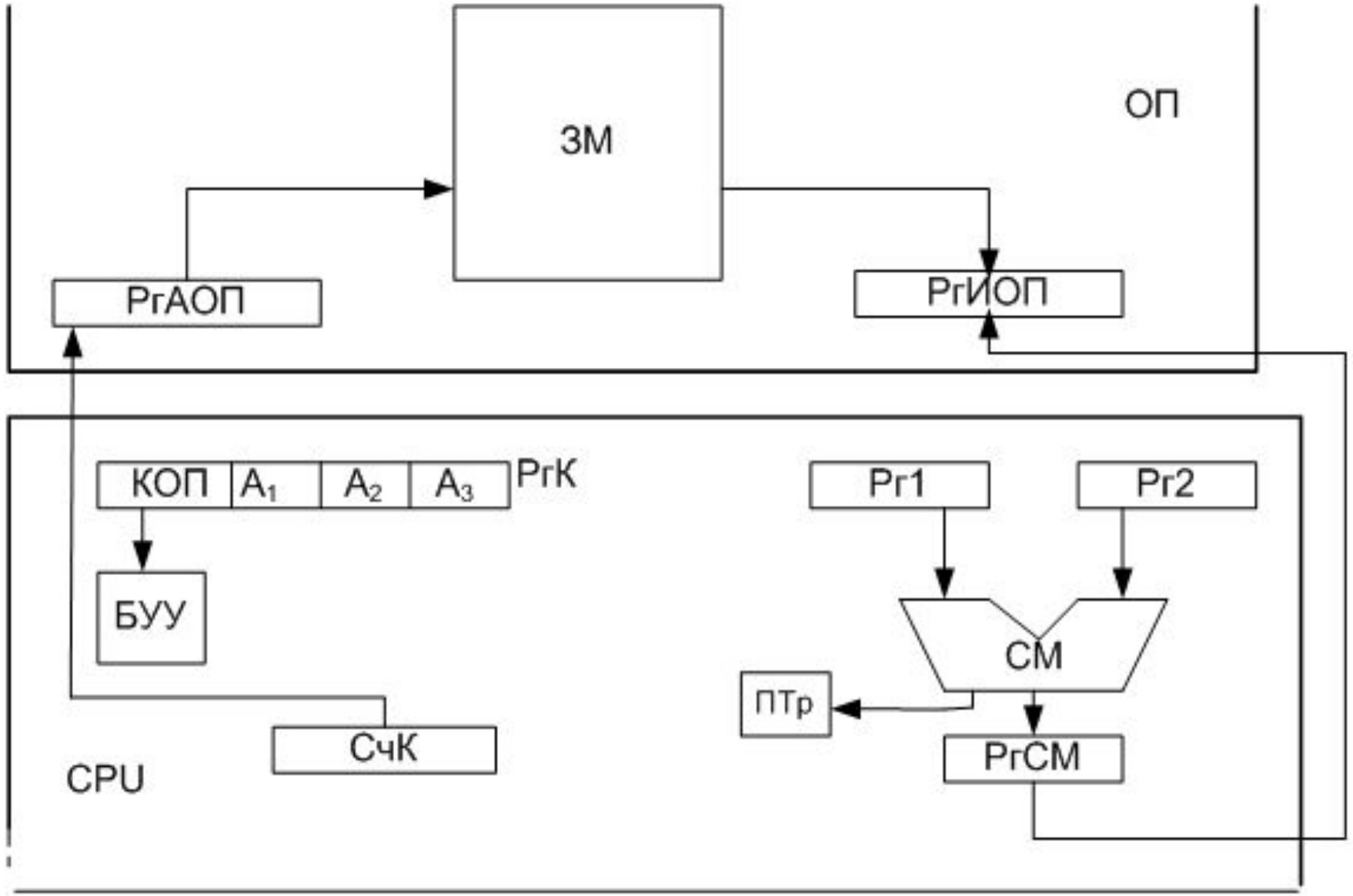
$$y = \frac{a + b}{c - d}$$

a=[101] [106,107]-вспомогательные ячейки
b=[102]
c=[103]
d=[104]
y=[105]
2000= СЛ 101,102,106, 2001
2001=Выч 103,104,107,2002
2002=Дел 106,107,105,2003

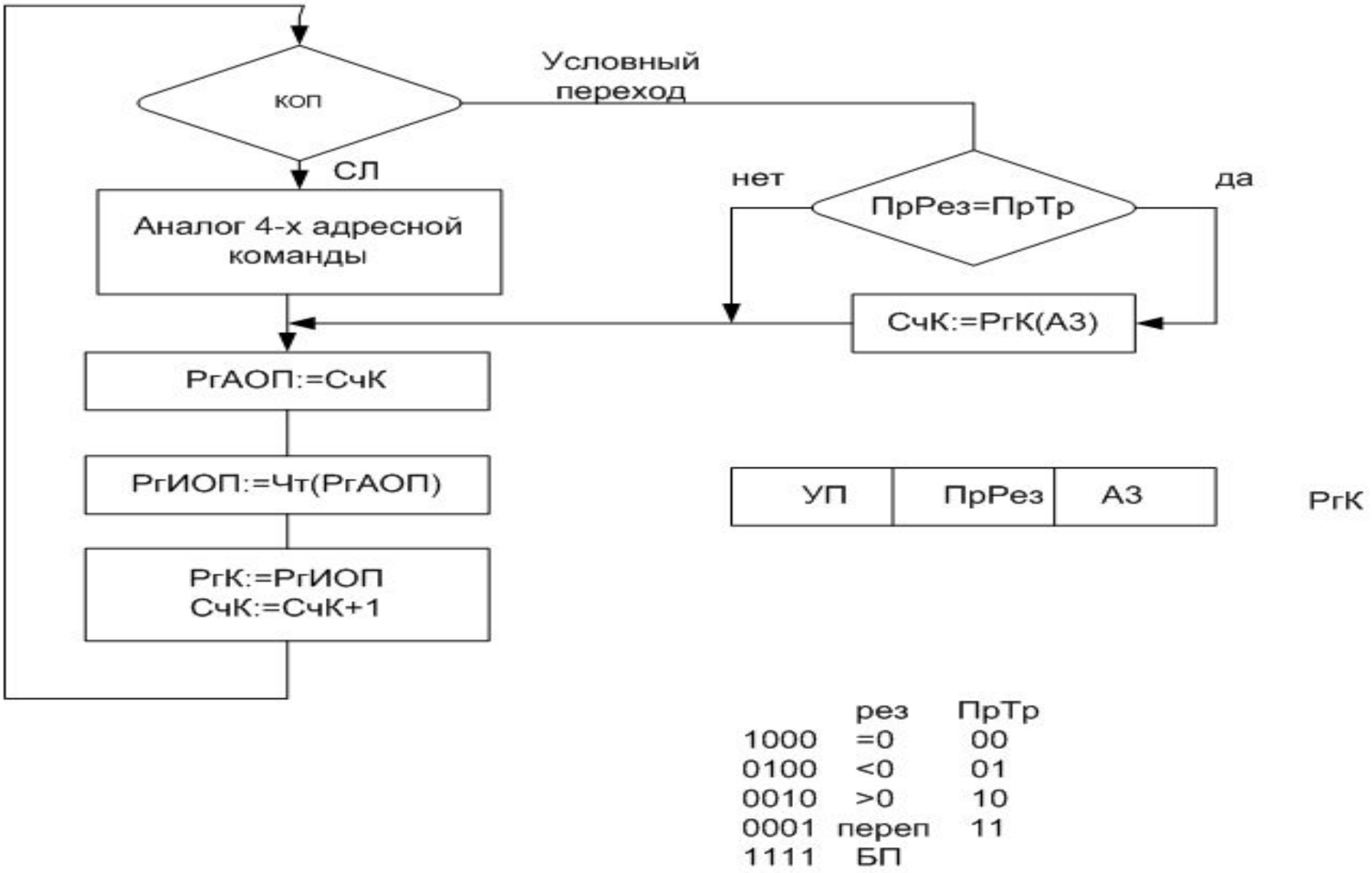


Функционирование машин с использованием 4-х адресной системы команд

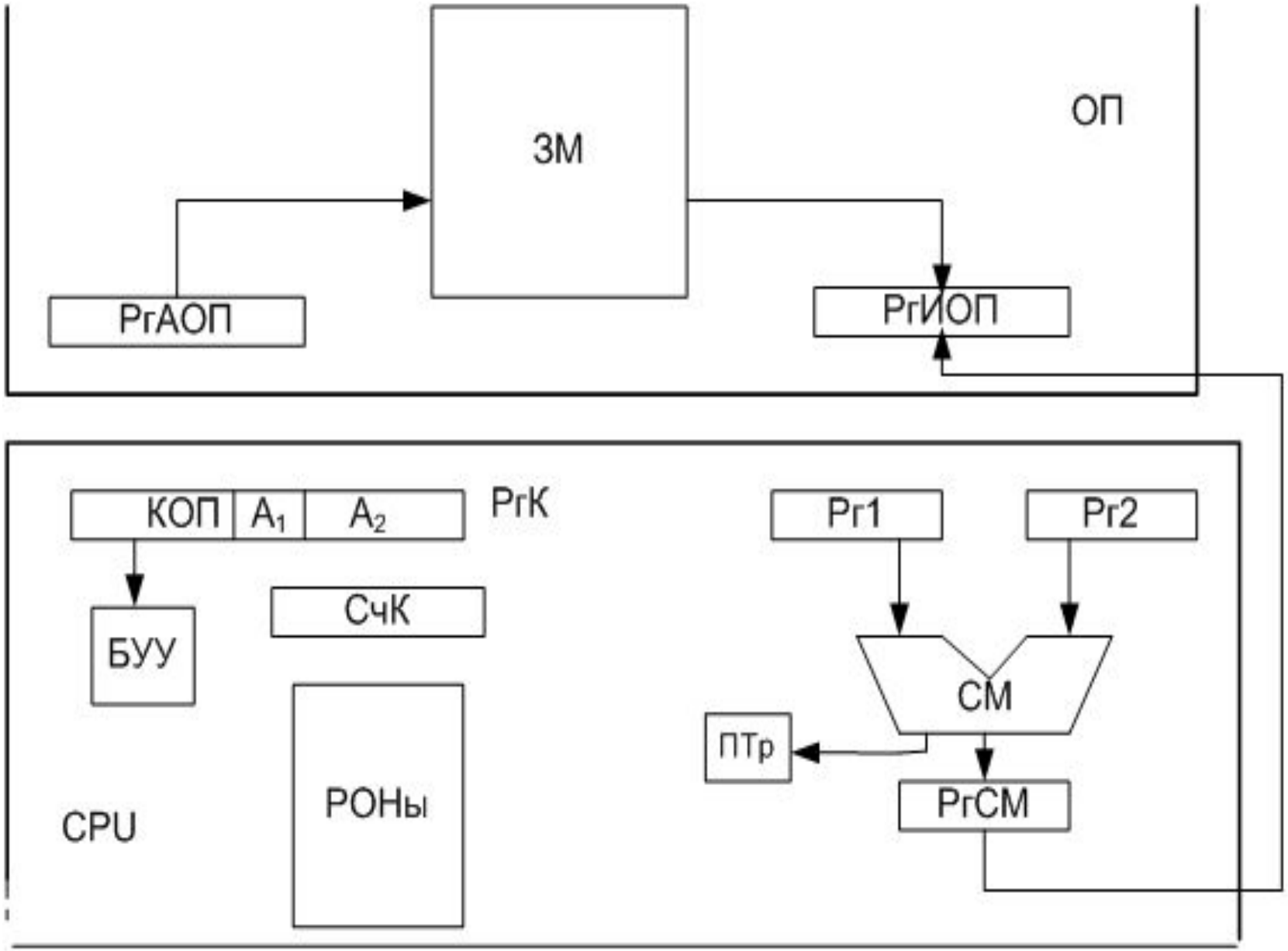

1
КОП
СЛ
2
РгАОП:=РгК[А1]
3
РгИОП:=Чт(РгАОП)
4
Рг1:=РгИОП
ВЫБОРКА 1-ГО ОПЕРАНДА
5
РгАОП:=РгК(А2)
6
РгИОП:=Чт(РгАОП)
7
Рг2:=РгИОП
ВЫБОРКА 2-ГО ОПЕРАНДА
8
РгСМ:=Рг2+Рг1
ОПЕРАЦИЯ В АЛУ
9
РгАОП:=РгК(А3)
РгИОП:=РгСМ
10
Зп(РгАОП):=РгИОП
ФОРМИРОВАНИЕ РЕЗУЛЬТАТА
11
РгАОП:=РгК(А4)
12
РгИОП:=Чт(РгАОП)
13
РгК:=РгИОП
ВЫБОРКА СЛЕД. КОМАНДЫ



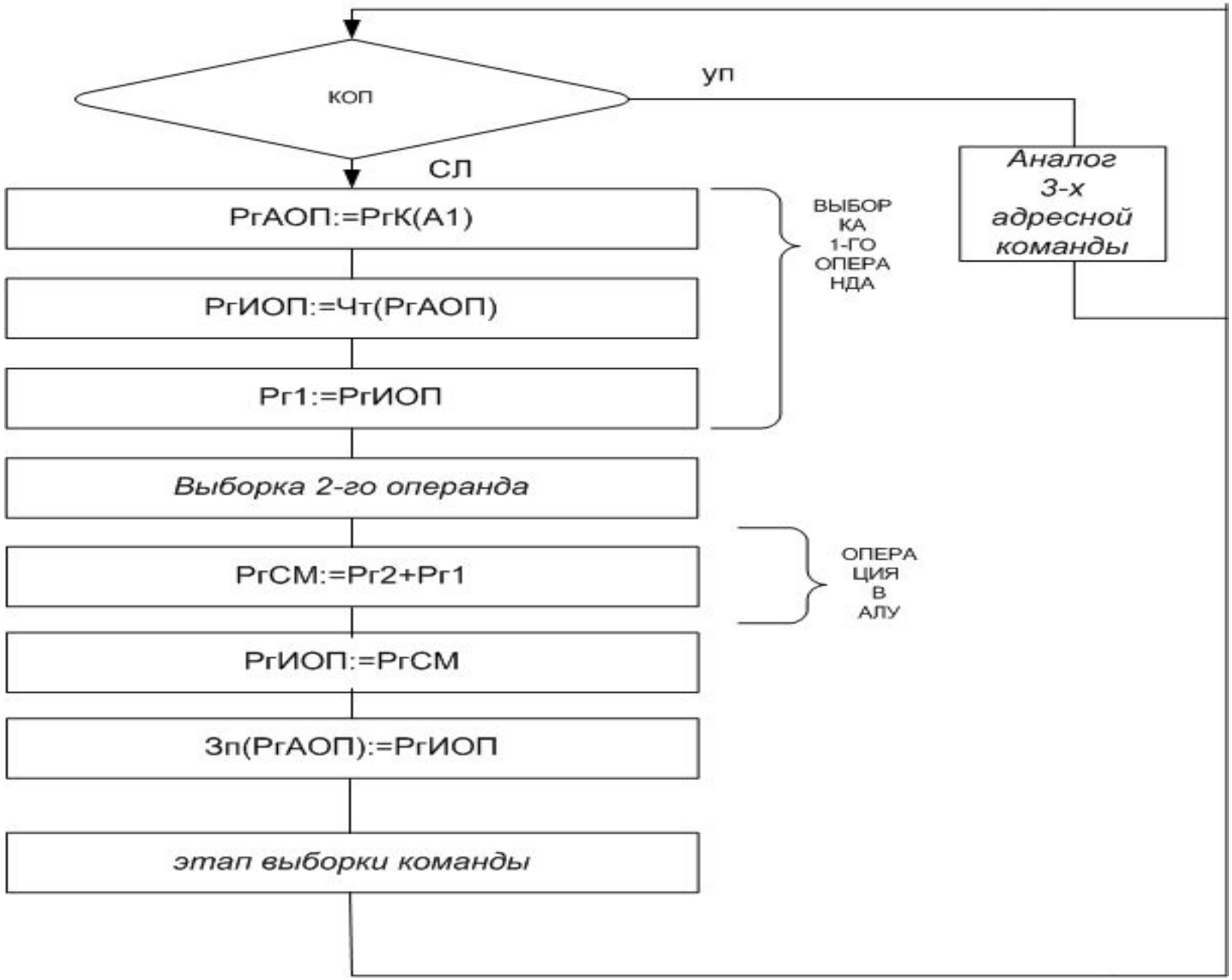
Функционирование машин с использованием 3-х адресной системы команд



	рез	ПрТр
1000	=0	00
0100	<0	01
0010	>0	10
0001	перен	11
1111	БП	



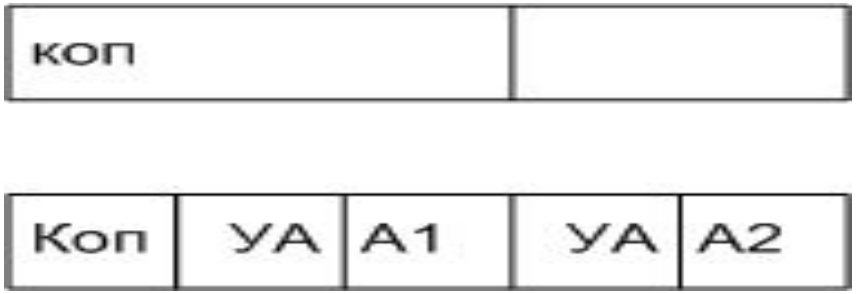
Функционирование машин с использованием 2-х адресной системы команд


КОП
УП
СЛ
Аналог 3-х адресной команды
РгАОП:=РгК(А1)
РгИОП:=Чт(РгАОП)
Рг1:=РгИОП
ВЫБОРКА 1-ГО ОПЕРАНДА
Выборка 2-го операнда
РгСМ:=Рг2+Рг1
ОПЕРАЦИЯ В АЛУ
РгИОП:=РгСМ
Зп(РгАОП):=РгИОП
этап выборки команды

Длина 1адр. Lкоп +Lадр
Длина 2адр. Lкоп +2 Lадр
Длина 3адр. Lкоп +3Lадр

P	**S1**	**S2**	**1адр**	**2адр**	**3адр**
$(1-p)^2$	$\overline{S1}$	$\overline{S2}$	1	1	1
(1-p)p	$\overline{S1}$	S2	2	1	1
(1-p)p	S1	$\overline{S2}$	2	1	1
P^2	S1	S2	3	2	1

Способы адресации

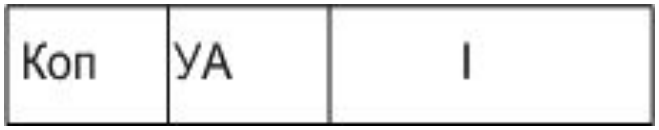


Эффективность способа адресации определяется :
-затратами оборудования С,
-затратами времени Т на доступ к адресуемым данным.

$\mathbf{C=C_{ВА}+C_{ЗУ},}$
$C_{ВА}$ -затраты аппаратных средств, обеспечивающих вычисление исполнительных адресов,
$C_{ЗУ}$ –затраты памяти на хранение адресных кодов команд.

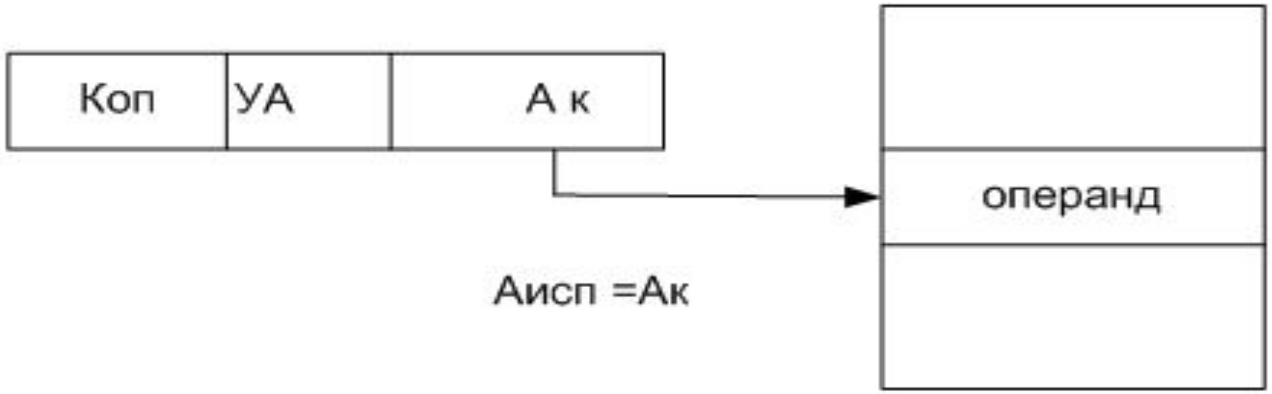
$\mathbf{T=T_{ФИА}+T_{ЗУ},}$
$T_{ФИА}$ – затраты времени на формировании исполнительного адреса,
$T_{ЗУ}$ -время выборки или записи операнда.

Непосредственная адресация



$$C_{НА} = 0, \quad T_{НА} = 0$$

Прямая адресация

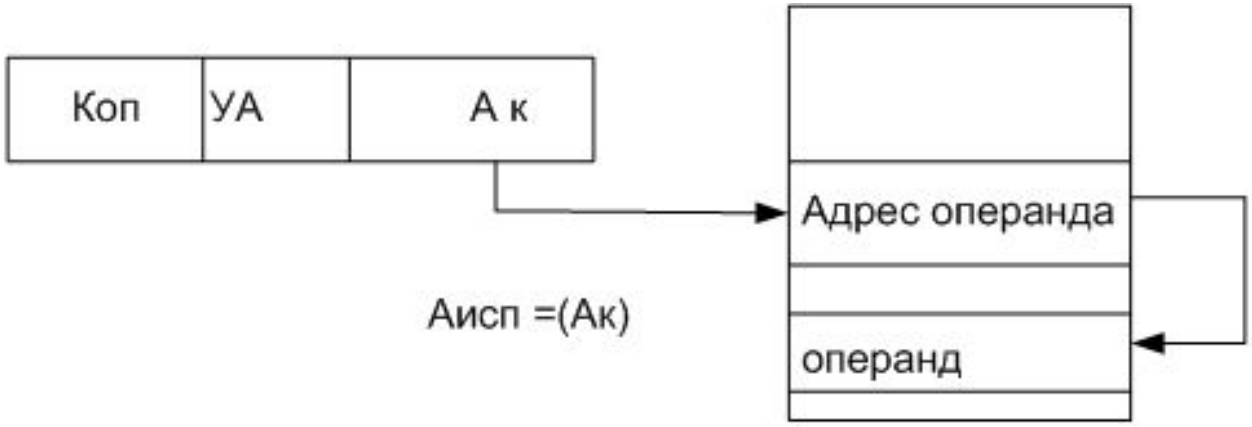


$C_{ПА}=int(log_2 Ni)$

$T_{ПА}=t_{ЗУ}$

Ni -Кол-во адресуемых операндов

Косвенная адресация



$$C_{КА}=R_{Яч}+ int(\log_2 N_A)=int(\log_2 (Ni+N_A))$$

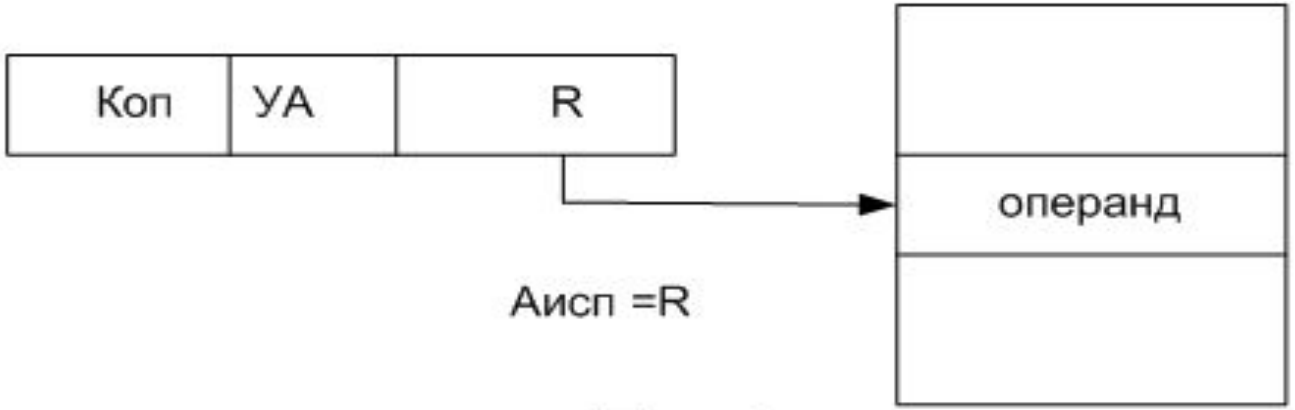
Ni -Кол-во адресуемых операндов

N_A- Кол-во ячеек для хранения исполнительных адресов

$R_{Яч}$ -разрядность ячейки памяти, хранящей исполнительный адрес.

$$Тк_A=2t_{зу}$$

Регистровая адресация



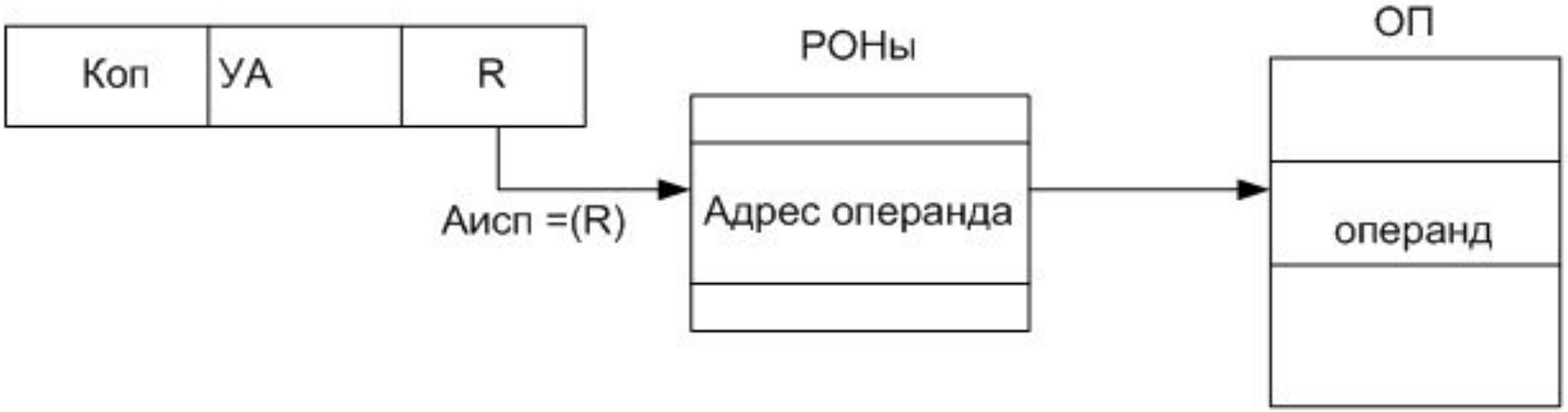
$T_{РА}=t_{РОН}$

$t_{РОН}$ -время выборки операнда из РОН,

$t_{РОН} << t_{ЗУ}$

$C_{РА}<<C_{ПА}$

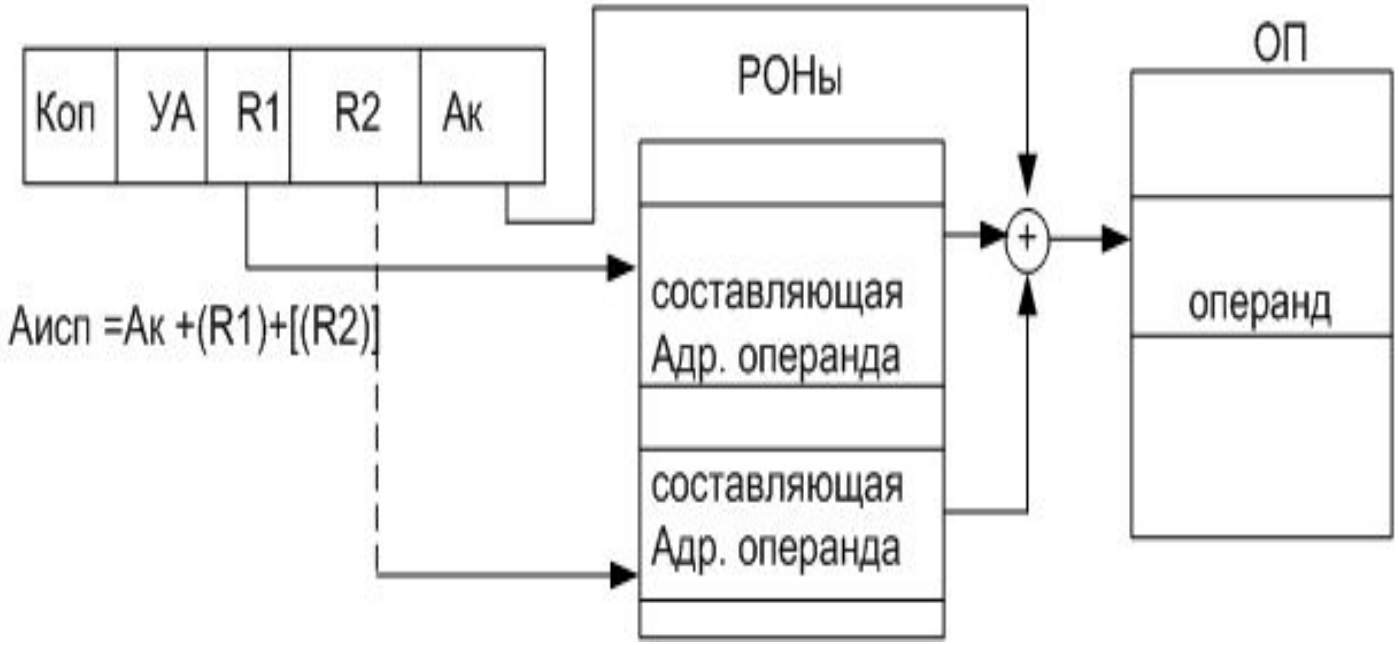
Косвенная регистровая адресация



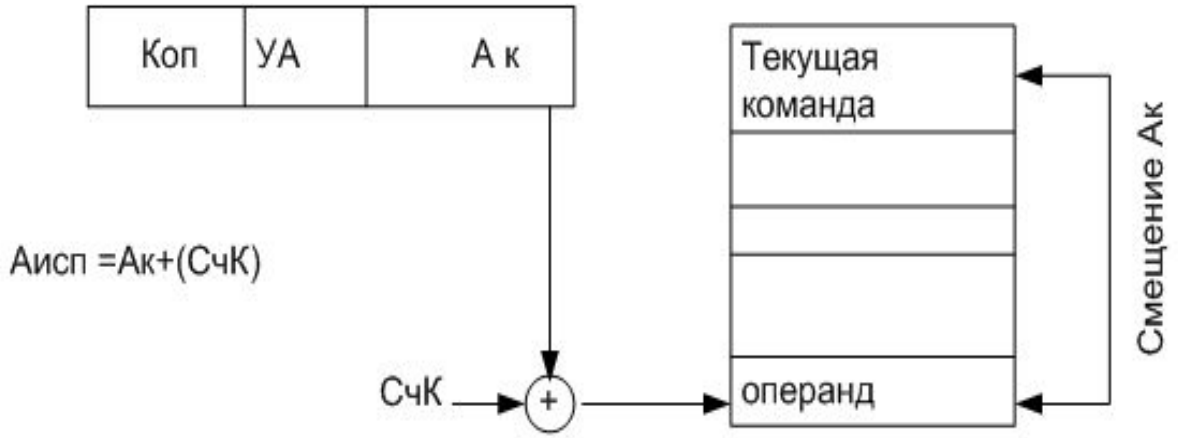
$$C_{КРА}=R_{РОН}+ int(\log_2 N_A)=int(\log_2 (Ni+N_A))$$

$$T_{КРА}=t_{Рон}+t_{ЗУ}$$

Адресация со смещением



Относительная адресация

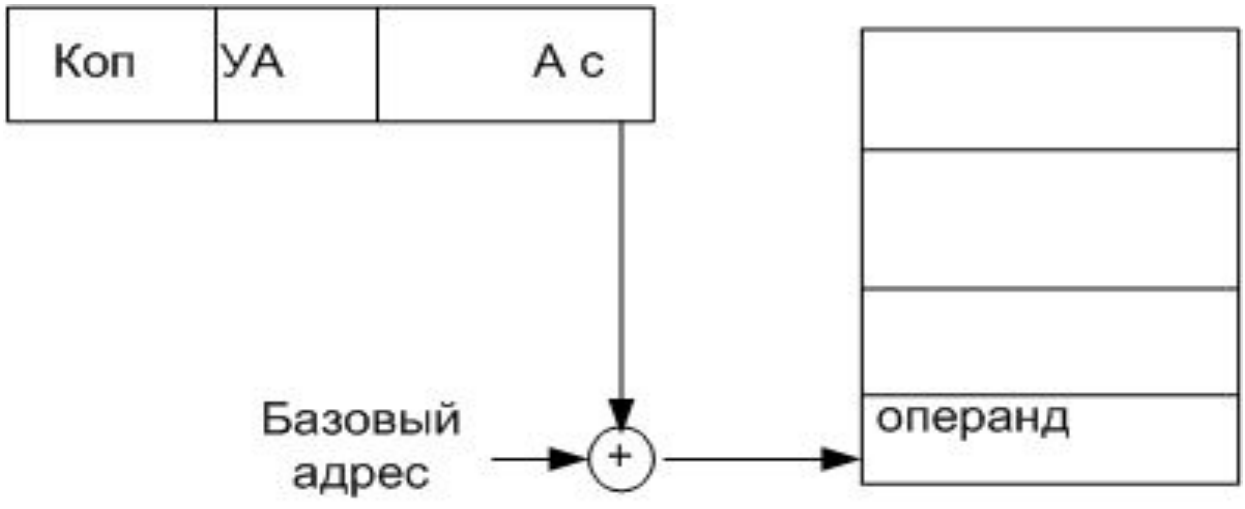


$$T_{CA}=t_{РОН}+t_{СЛ}+t_{ЗУ}$$

$$C_{CA}= int(\log_2 Ni- R_{CK})$$

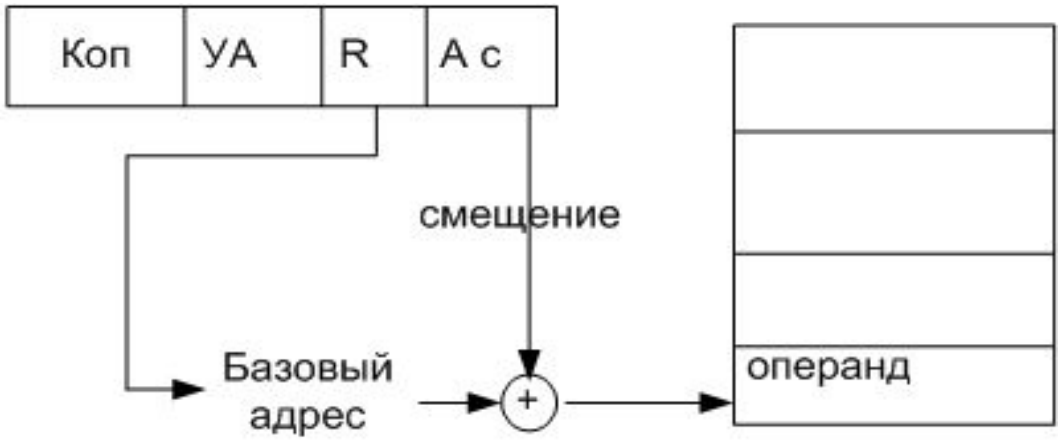
$R_{Ск}$ -Разрядность счетчика команд
$t_{СЛ}$-время сложения составляющих исполнительного алреса

Базовая регистровая адресация



Аисп =Ас+(БР)

Базовая регистровая адресация с базовым регистром



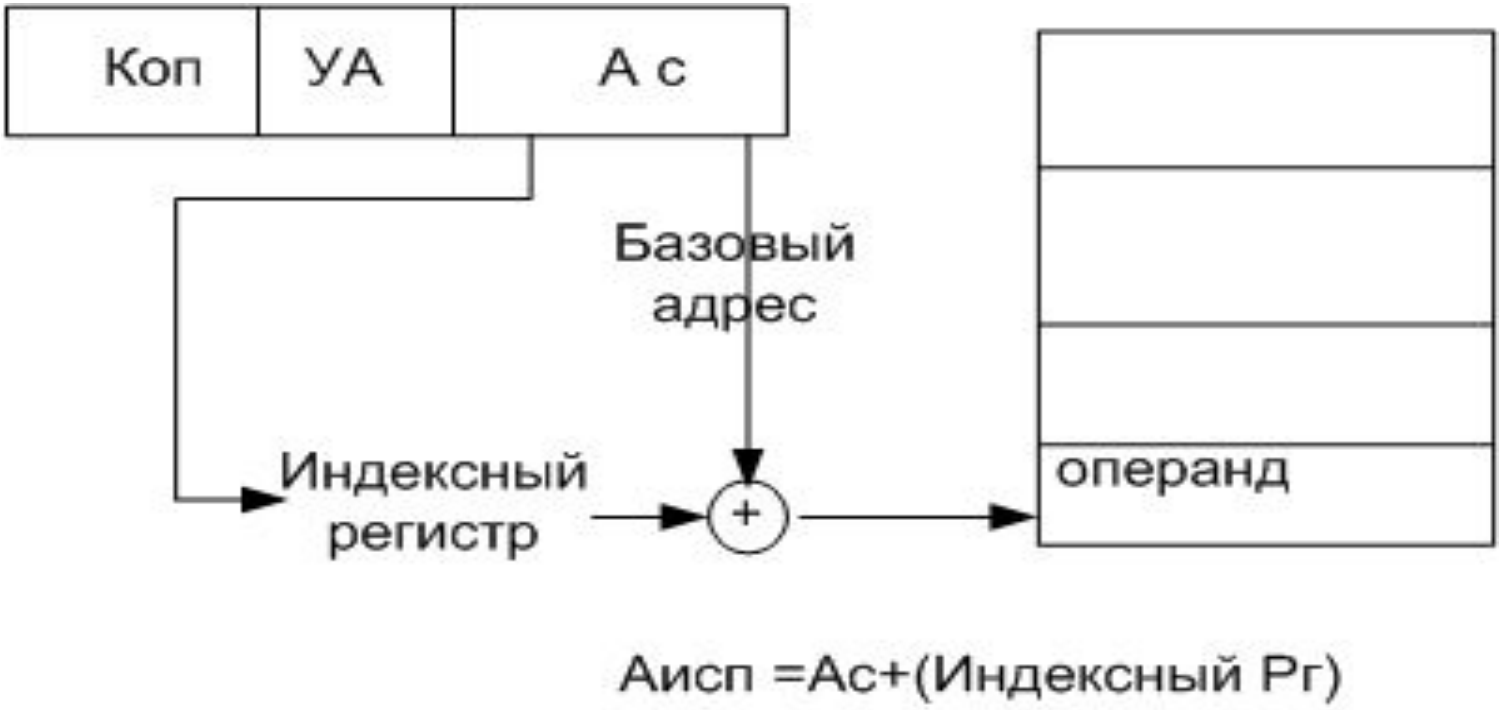
$$T_{БРА}=t_{РОН}+t_{СЛ}+t_{ЗУ}$$

Аисп =Ac+(R)

$$R_{СМ}=C_{БРА}= int(\log_2 (max(N_{Оп})))$$

Nоп -кол-во операндов i-й программы

Индексная адресация



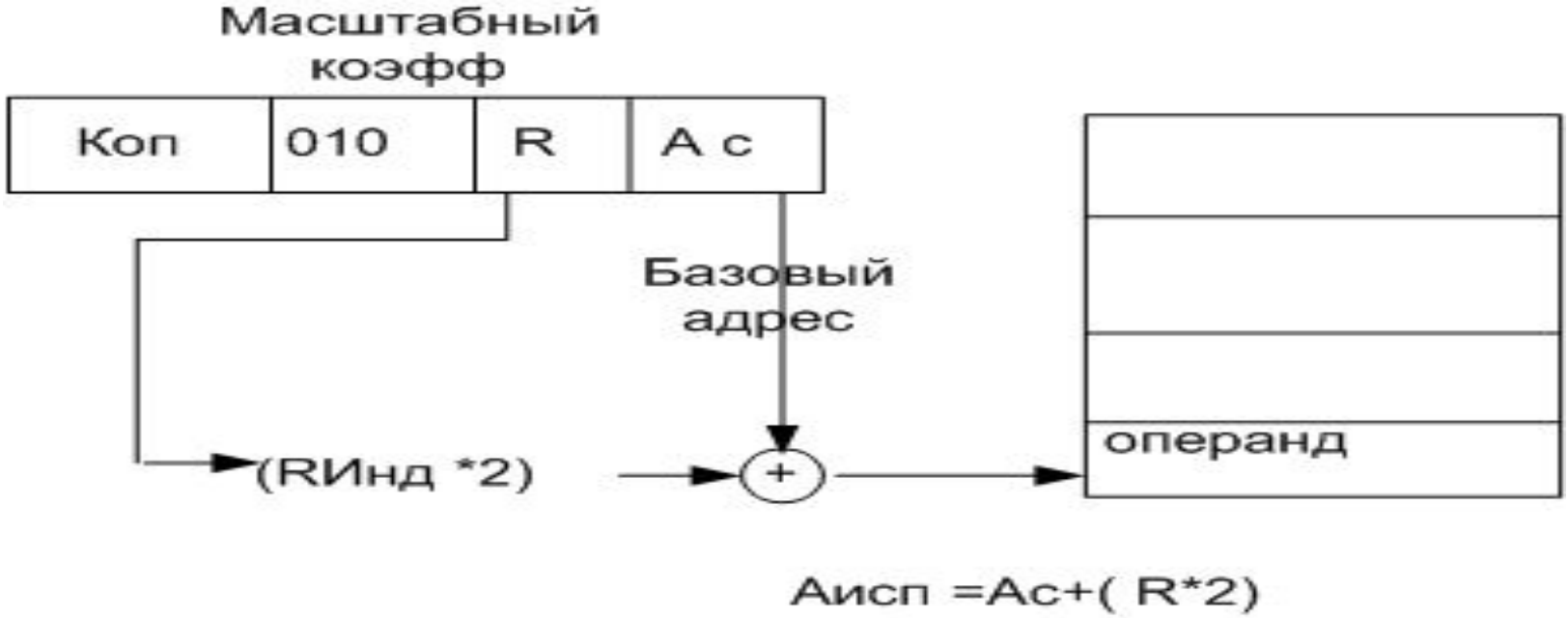
Автоинкрементная
Постинкрементное автоиндексирование
$A_{исп} = A_c + (R),\ R \leftarrow (R)+1$

преинкрементное

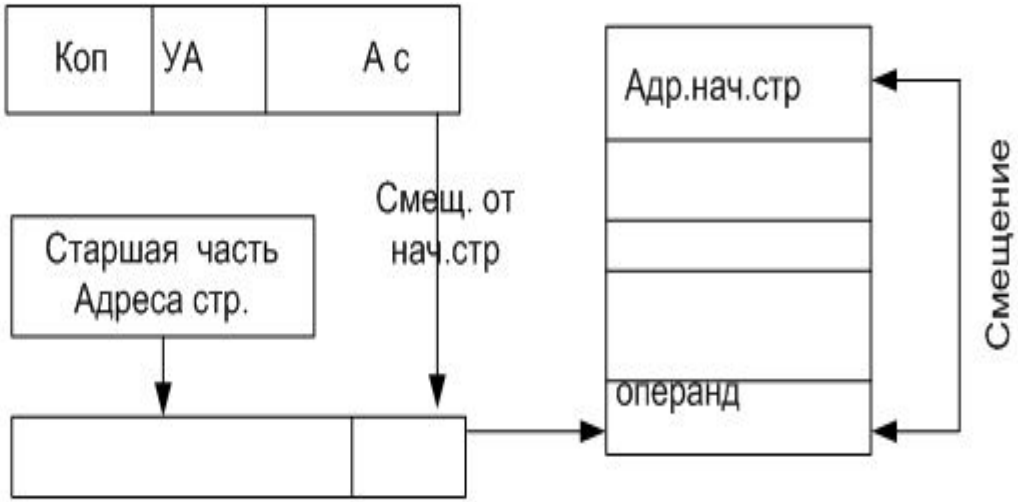
$R \leftarrow (R) + 1,\ A_{исп} = A_c + (R)$

Автодекрементная
пост….
$A_{исп} = A_c + (R),\ R \leftarrow (R)-1$
пре…..
$R \leftarrow (R) - 1,\ A_{исп} = A_c + (R)$

Индексная с масшабированием



Страничная адресация



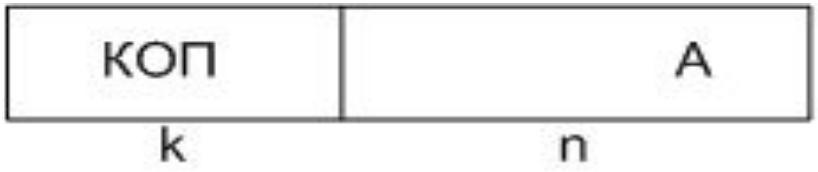
$$T_{СТА}=t_{РОН}+t_{ЗУ}$$

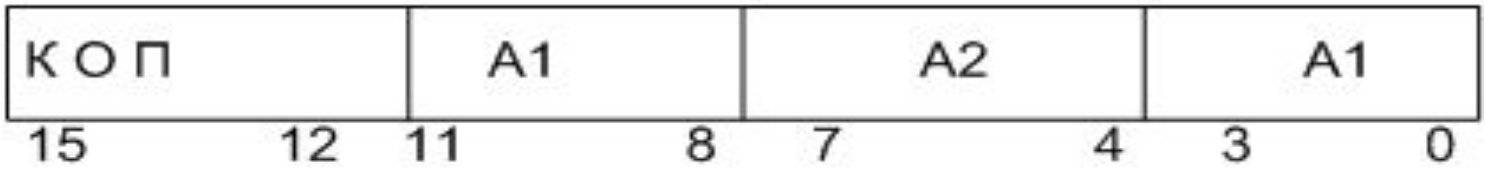
Аисп =А н.стр. II Ас

$$C_{СТА}= int(\log_2 Ni- \log_2 M)$$

М-кол-во страниц в памяти

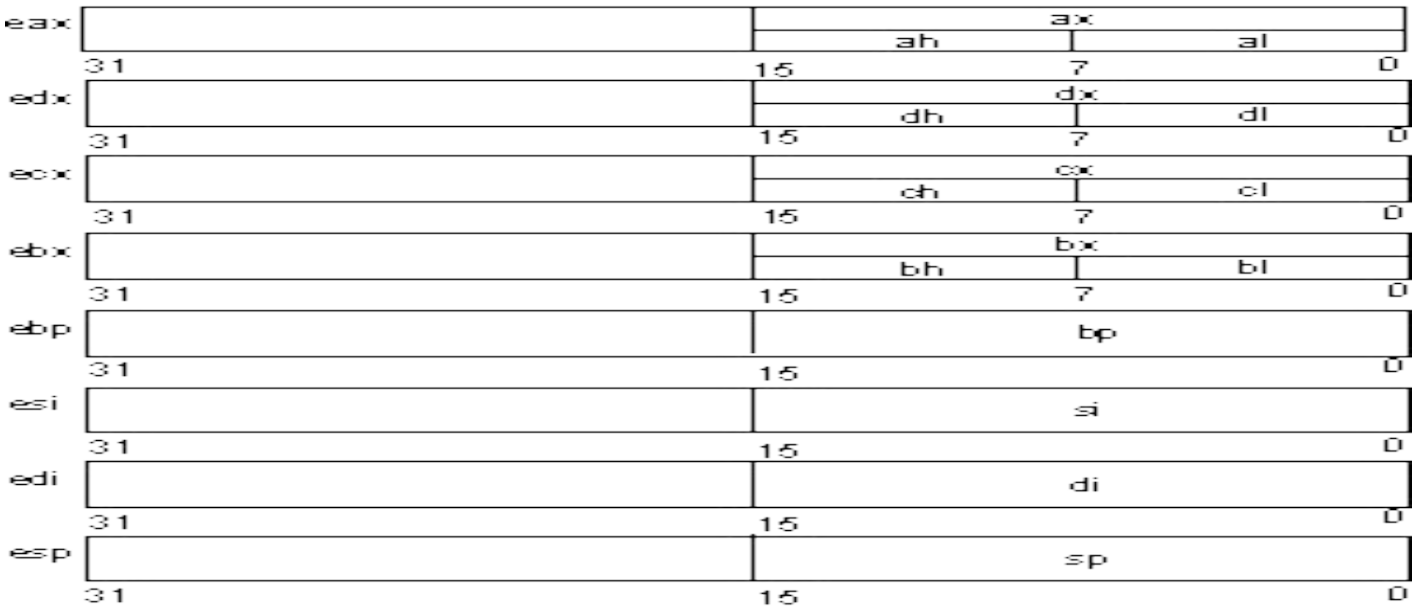
Расширение кода операции



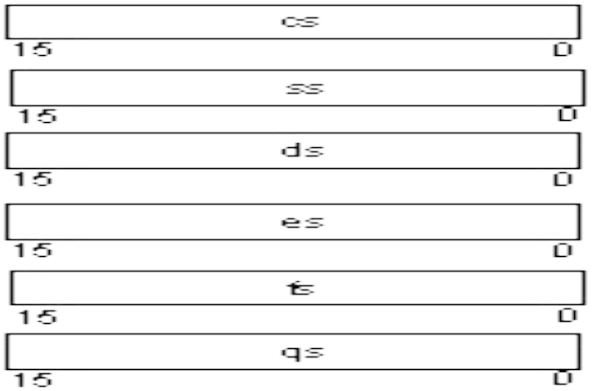


4 БИТНЫЙ КОП (15 трехадресных команд)															
0	**0**	**0**	**0**	**X**	**X**	**X**	**X**	**Y**	**Y**	**Y**	**Y**	**Z**	**Z**	**Z**	**Z**
0	**0**	**0**	**1**												
…..															
1	**1**	**1**	**0**	**X**	**X**	**X**	**X**	**Y**	**Y**	**Y**	**Y**	**Z**	**Z**	**Z**	**Z**
8 БИТНЫЙ КОП								***(14 двухадресных команд)***							
1	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**Y**	**Y**	**Y**	**Y**	**Z**	**Z**	**Z**	**Z**
1	**1**	**1**	**1**	**0**	**0**	**0**	**1**								
….															
1	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**Y**	**Y**	**Y**	**Y**	**Z**	**Z**	**Z**	**Z**
12 БИТНЫЙ КОП						***(31 одноадресная команда)***									
1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**Z**	**Z**	**Z**	**Z**
							0	**0**	**0**	**0**	**1**				
							….								
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**Z**	**Z**	**Z**	**Z**
16 БИТНЫЙ КОП (16 безадресных команд)															
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
…..															
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

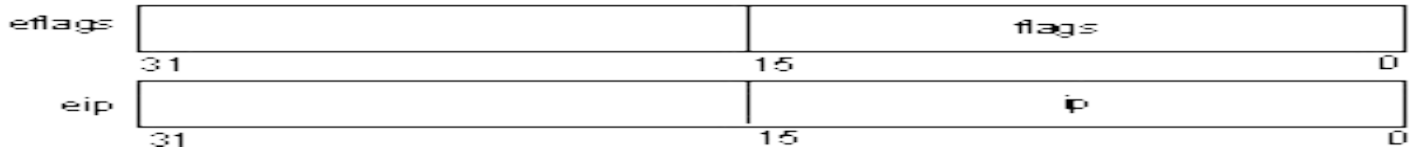
Регистры общего назначения:



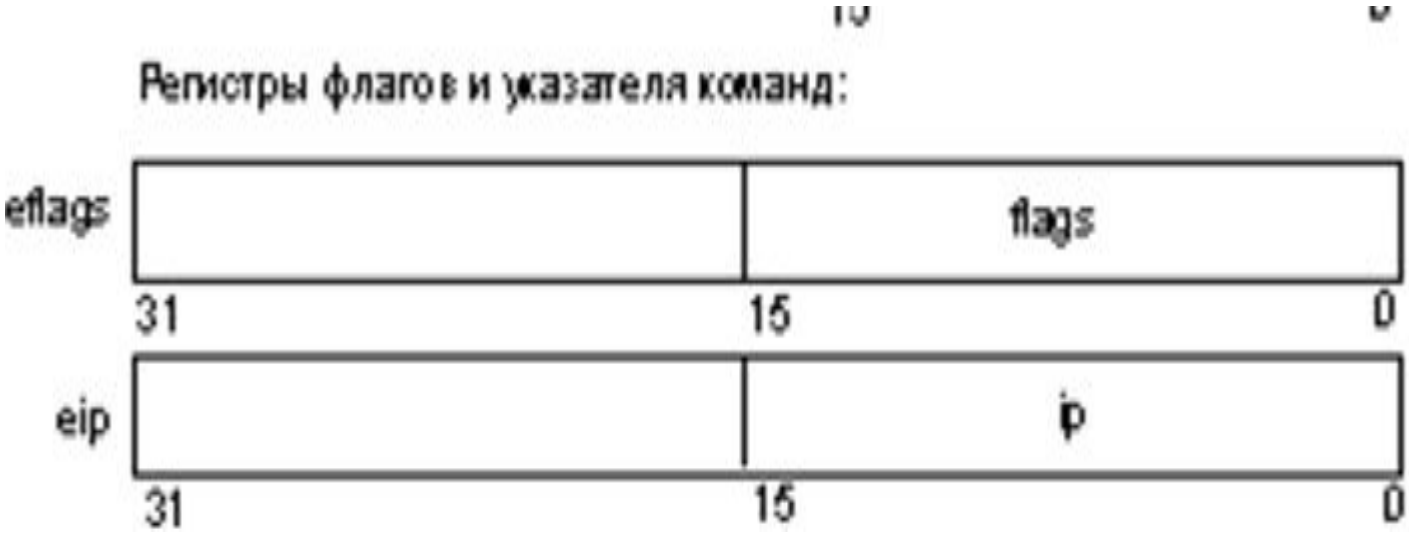
Сегментные регистры :



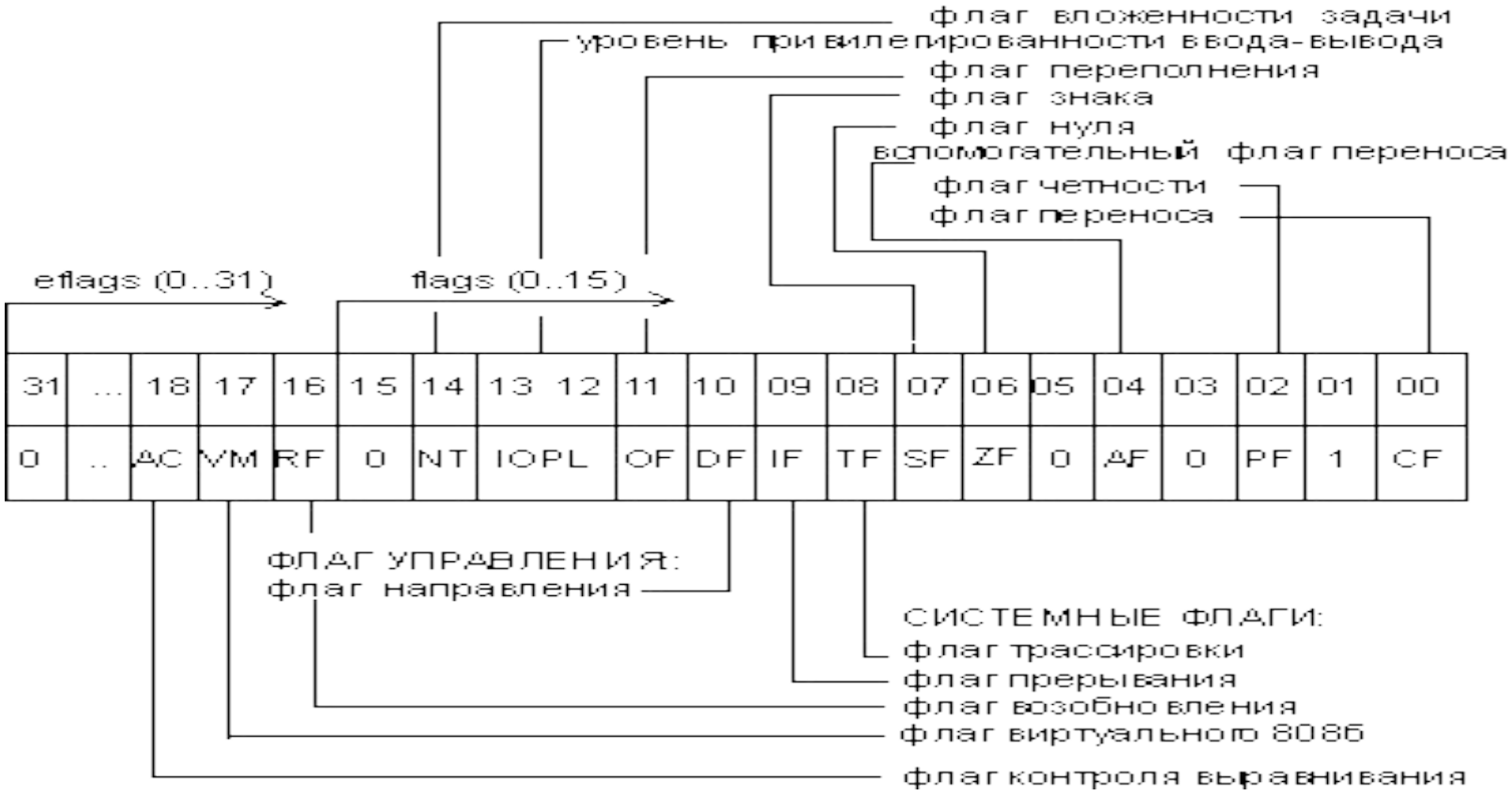
Регистры флагов и указателя команд:

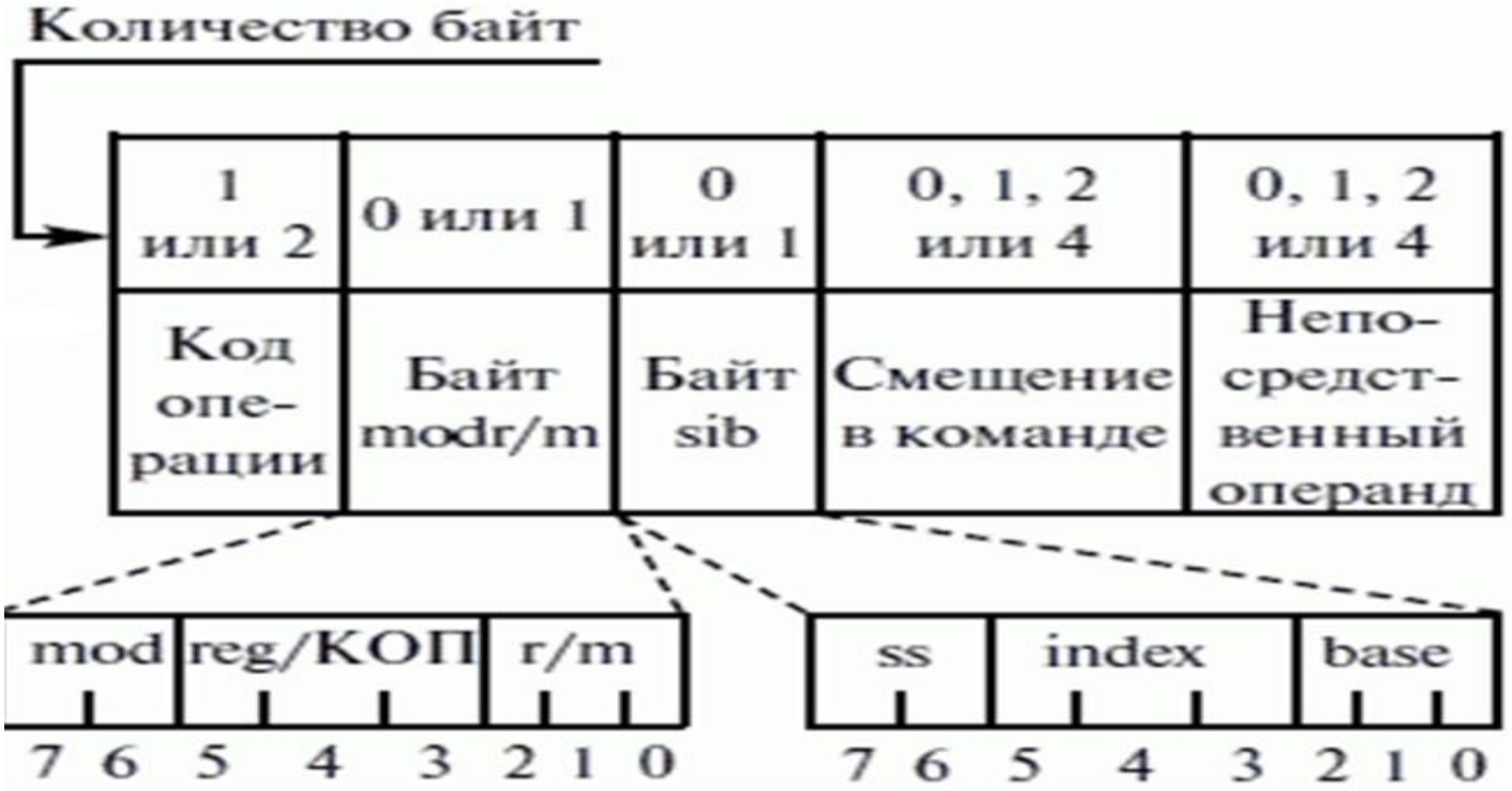


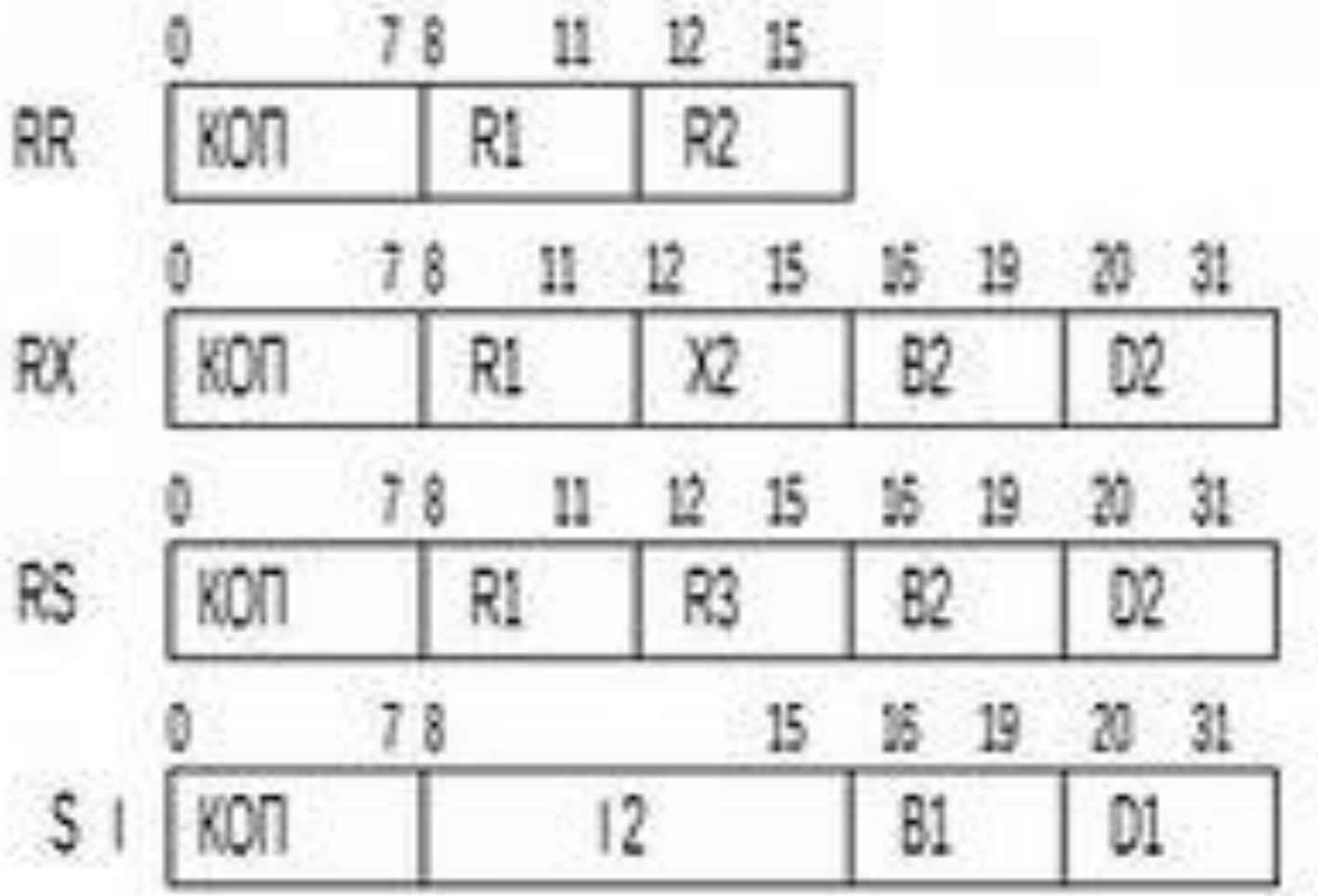
Вектор состояния



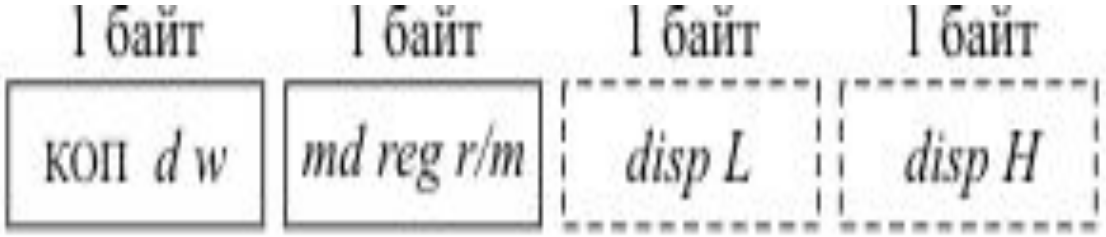
ФЛАГИ СОСТОЯНИЯ:



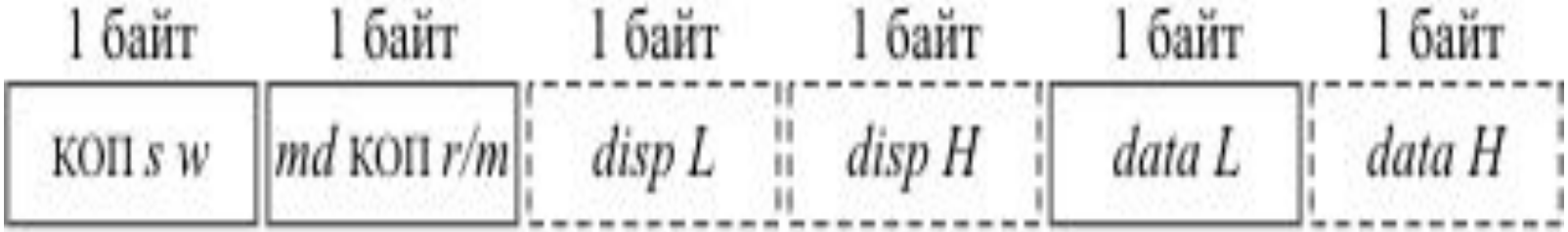

Количество байт
1 или 2
0 или 1
0 или 1
0, 1, 2 или 4
0, 1, 2 или 4
Код опе-рации
Байт modr/m
Байт sib
Смещение в команде
Непо-средст-венный операнд
mod
reg/КОП
r/m
ss
index
base
7 6 5 4 3 2 1 0
7 6 5 4 3 2 1 0


RR
0 7 8 11 12 15
КОП
R1
R2
RX
0 7 8 11 12 15 16 19 20 31
КОП
R1
X2
B2
D2
RS
0 7 8 11 12 15 16 19 20 31
КОП
R1
R3
B2
D2
SI
0 7 8 15 16 19 20 31
КОП
I2
B1
D1

Пример форматов команд процессоров intel



а) формат команд типа *RR* и *RS*



б) формат команды с непосредственным операндом

Форматы команд	Первое полуслово			Второе полуслово			Третье полуслово		
	байт 1		байт 2	байт 3		байт 4	байт 5		байт 6
	0 7		8 15	16 23		24 31	32 39		40 47
SS	КОП		L! L2	B1	D1		B2	D2	
SI	КОП		I2	B1	D1				
RS	КОП		R1 R3	B2	D2				
RX	КОП		R1 X2	B2	D2				
RR	КОП		R1 R2	-	-				

Регистровые структуры центрального процессора

Набор регистров и их структуры рассмотрим на примере процессоров Intel с CISC-архитектурой. Можно выделить следующие группы регистров:

1. Основные функциональные регистры (используются при выполнении прикладных программ**) :**

- регистры общего назначения (РОН);
- указатель команд;
- регистр флагов;
- регистры сегментов.

2. Регистры процессора (FPU) обработки чисел с плавающей точкой (используются при выполнении прикладных программ**):**

- регистры данных;
- регистр тегов;
- регистр состояния;
- регистр указателей команд и данных FPU;
- регистр управления FPU.

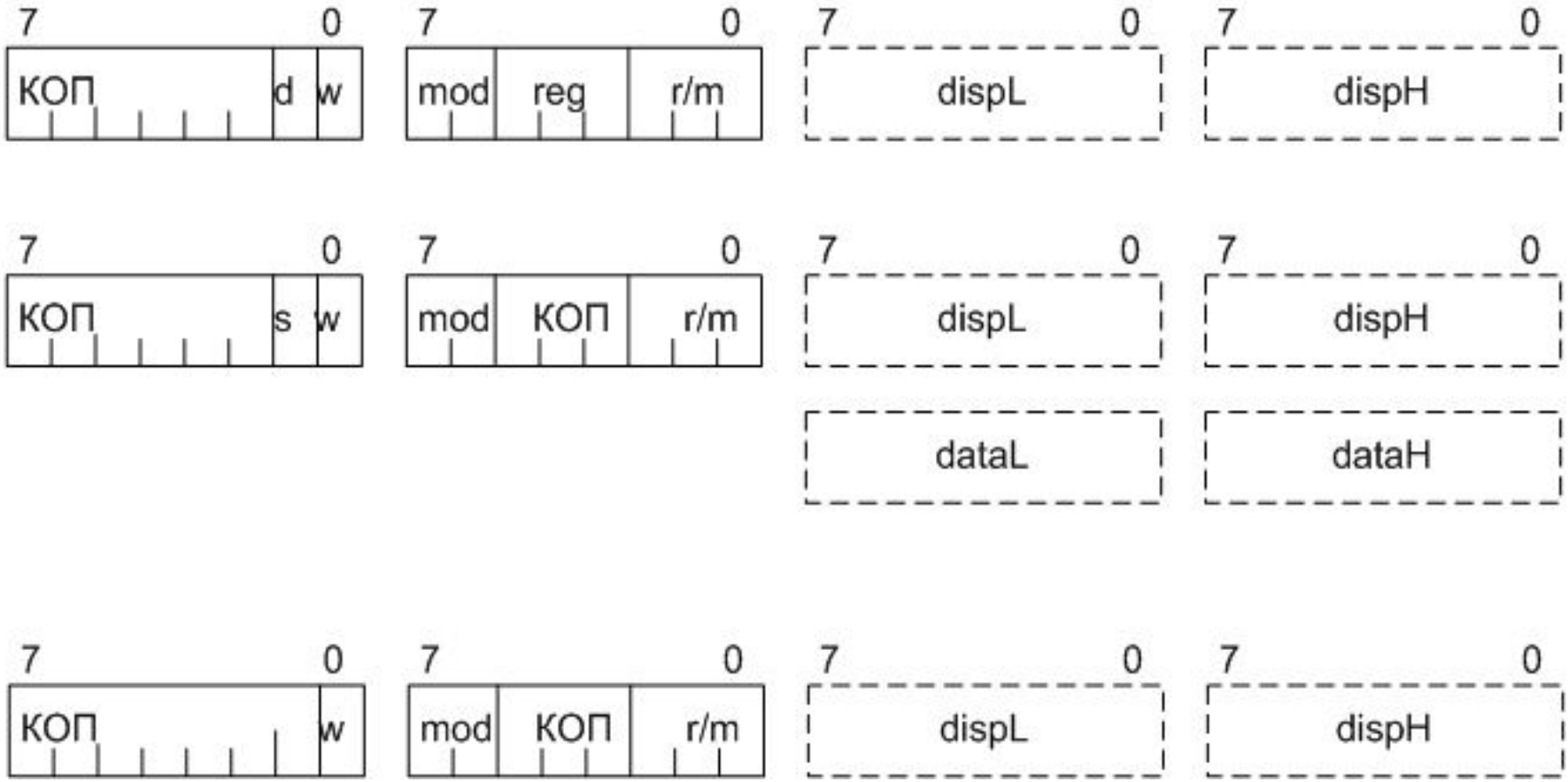
3. Системные регистры (используются при выполнении системных программ**):**

- регистры управления микропроцессора;
- регистры системных адресов.

4. Регистры отладки и тестирования (используются при отладке и тестировании**).**

Все 16-разрядные регистры микропроцессоров 8086, 80186, 80286 входят в состав набора 32-разрядных регистров.

Формат 2-х операндной команды



Описание команд

Григорьев В.Л. Микропроцессор i486. /В.Л.Григорьев.-М:Бином.1993г.-384с. том 2,3,4 стр.264

https://www.intel.ru/ Intel® 64 and IA-32 Architectures Software Developer's Manual

ADC - ADd with Carry
Сложение с переносом

Код	Команда	Такты	Описание
14 ib	ADC AL,imm8	1	Сложение с CF непоср. байта с AL
15 iw	ADC AX,imm16	1	Сложение с CF непоср. слова с AX
15 id	ADC EAX,imm32	1	Сложение с CF непоср. двойного слова с EAX
80 /2 ib	ADC r/m8,imm8	1/3	Сложение с CF непоср. байта с байтом r/m
81 /2 iw	ADC r/m16,imm16	1/3	Сложение с CF непоср. слова со словом r/m
81 /2 id	ADC r/m32,imm32	1/3	Сложение с CF непоср. двойного слова с двойным словом r/m
83 /2 ib	ADC r/m16,imm8	1/3	Сложение с CF непоср. байта со знаковым расширением со словом r/m
83 /2 ib	ADC r/m32,imm8	1/3	Сложение с CF непоср. байта со знаковым расширением с двойным словом r/m
10 /r	ADC r/m8,r8	1/3	Сложение с CF байтного рег. с байтом r/m
11 /r	ADC r/m16,r16	1/3	Сложение с CF словного рег. со словом r/m
11 /r	ADC r/m32,r32	1/3	Сложение с CF двухсловного регистра с двойным словом r/m
12 /r	ADC r8,r/m8	1/3	Сложение с CF байта r/m с байтным регистром
13 /r	ADC r16,r/m16	1/3	Сложение с CF слова r/m со словным регистром
13 /r	ADC r32,r/m32	1/3	Сложение с CF двойного слова r/m с двухсловным регистром

Операция

```
DEST <-- DEST + SRC + CF;
```

Описание

Команда ADC производит целочисленное сложение двух операндов DEST и SRC и флажка переноса CF. Результат сложения присваивается первому операнду (DEST) и соответственно устанавливаются флажки. Обычно команда ADC применяется как часть операции многобайтного или многословного сложения. Когда непосредственное байтное значение прибавляется к слову или двойному слову, оно вначале расширяется со знаком до размера операнда.

Воздействие на флажки

Флажки OF, SF, ZF, AF, CF и PF устанавливаются в соответствии с результатом.

Особые случаи P-режима

#GP(0), если результат в незаписываемом сегменте.

Далее см. **ПРИМЕЧАНИЕ**.

Intel® 64 and IA-32 Architectures Software Developer's Manual

Volume 2 (2A, 2B & 2C): Instruction Set Reference, A-Z