Этап 2.
Разработка форматов команд, исполняемых процессором (согласно варианту задания на КП).

Поле КОП

Мы считаем, что проектируемое устройство, реализующее пять определенных вариантом задания команд, является фрагментом процессора, реализующего от 128 до 256 различных команд (т.е. не более 256).
Поэтому каждая команда должна содержать 8 битное поле кода операции (КОП):

$$\log_2 256 = 8$$

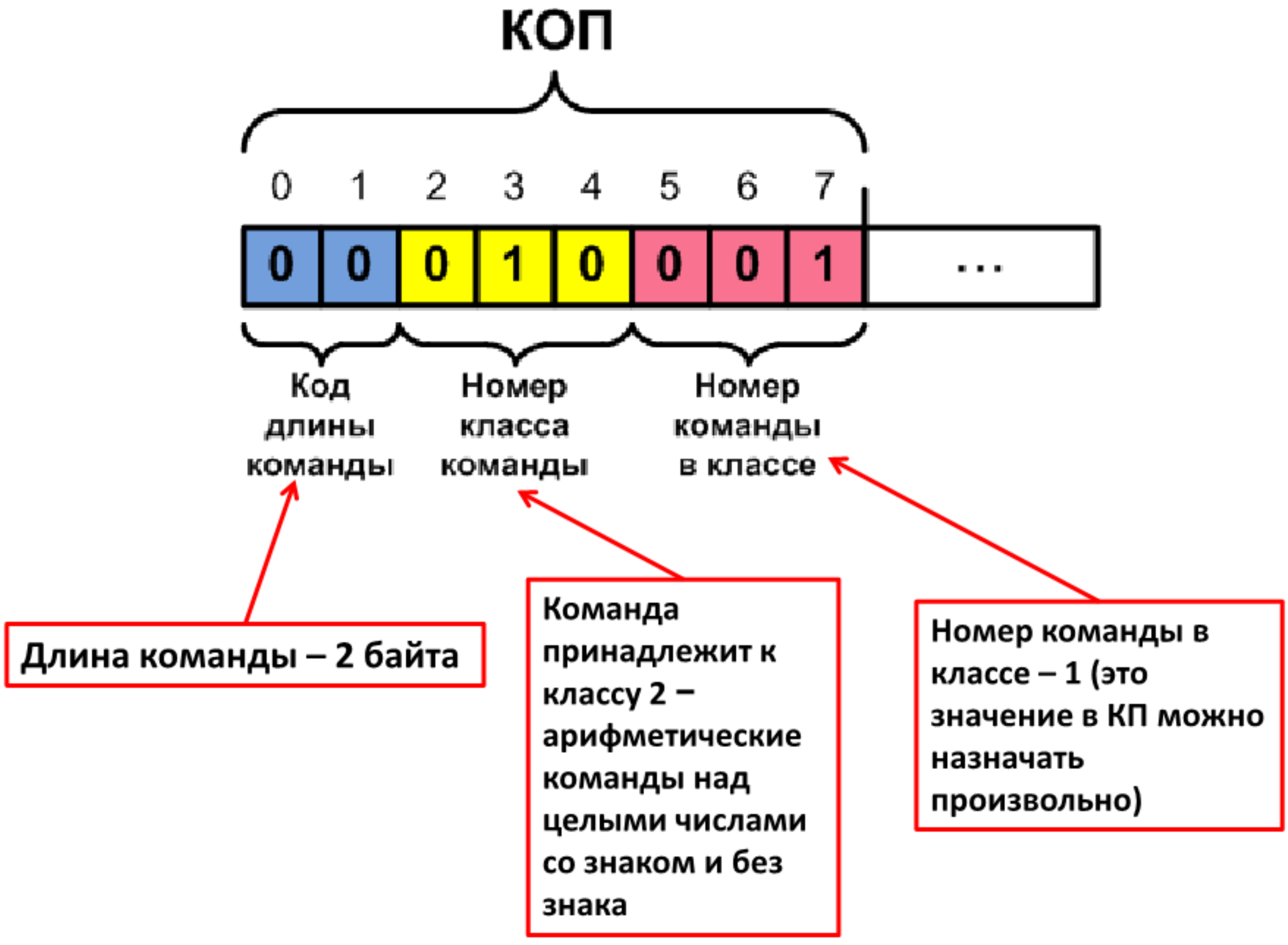
Исходя из вышесказанного, для кодирования КОП команды можно использовать целые беззнаковые числа в диапазоне от 0 до 255 (0 до 0xFF в шестнадцатеричной системе счисления).

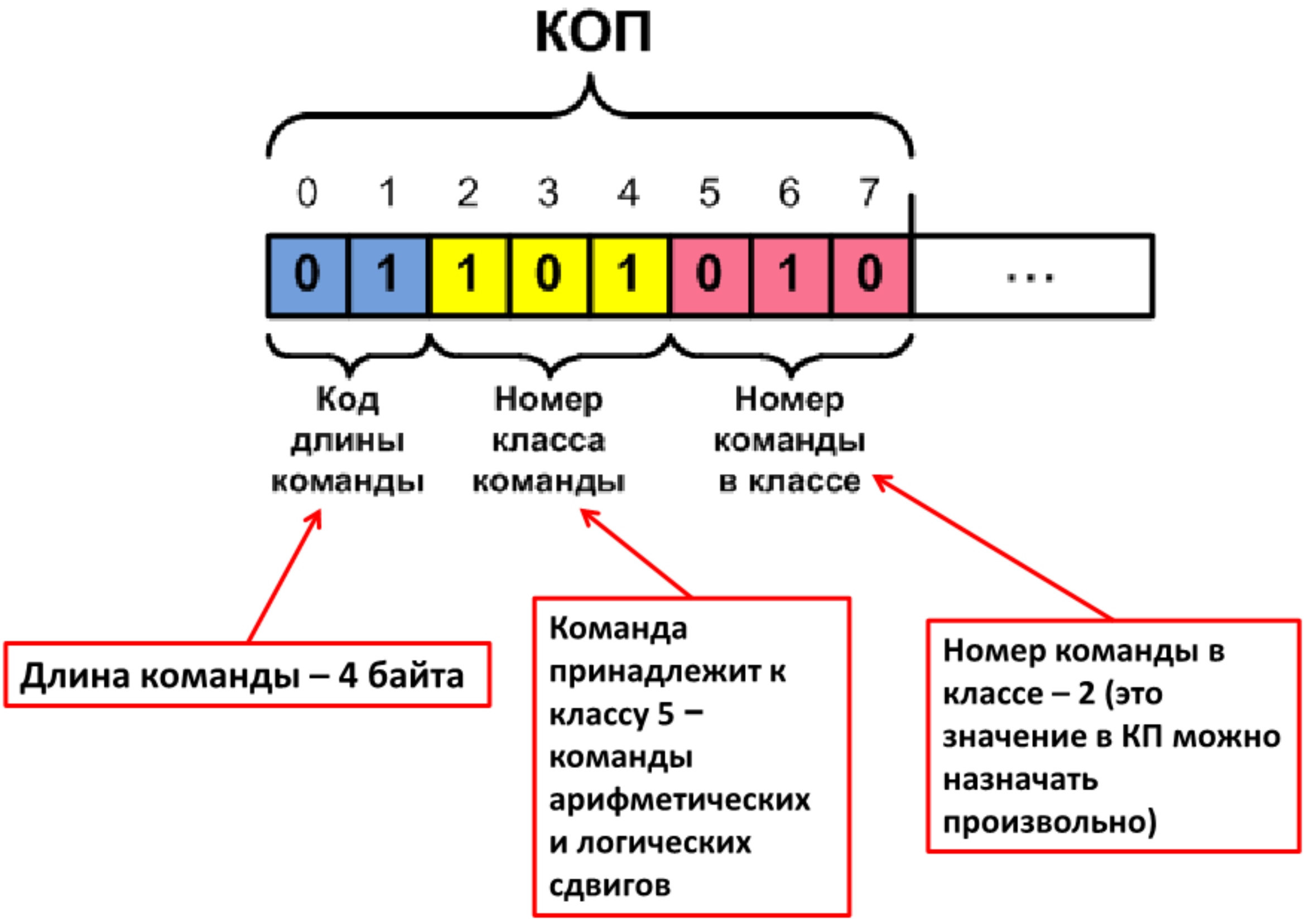
В поле КОП команды будем выделять три поля: КОП(0:1) – для кода формата команды (или кода длины команды), КОП(2:4) – для кода класса команды, КОП(5:7) – для номера команды в списке класса.

Примем, что для команд длиной два байта КОП(0:1)=00, для команд длиной четыре байта КОП(0:1)=01 (комбинации 10 и 11 – в резерве – вдруг у кого-то получатся команды длиной 6 или 8 байтов;-)

Номера классов команд

1) команды обращения к памяти по чтению и записи;
2) арифметические команды над целыми числами со знаком и без знака (сложение, вычитание, умножение, деление, сравнение);
3) арифметические команды над числами с плавающей точкой (сложение, вычитание, умножение, деление, сравнение);
4)логические команды (поразрядное «И», «ИЛИ», «Исключающее ИЛИ», инверсия),
5)команды арифметических и логических сдвигов (операндов одинарной (L4) и двойной (L8) длины) на заданное число разрядов (константа сдвига);
6)команды передачи управления (условных и безусловных переходов);
7)команды ввода/вывода;
0)специальные команды (HALT или STOP – останов).


КОП
0 1 2 3 4 5 6 7
0 0 0 1 0 0 0 1
...
Код длины команды
Номер класса команды
Номер команды в классе
Длина команды – 2 байта
Команда принадлежит к классу 2 – арифметические команды над целыми числами со знаком и без знака
Номер команды в классе – 1 (это значение в КП можно назначать произвольно)


КОП
0 1 2 3 4 5 6 7
0 1 1 0 1 0 1 0
...
Код длины команды
Номер класса команды
Номер команды в классе
Длина команды – 4 байта
Команда принадлежит к классу 5 – команды арифметических и логических сдвигов
Номер команды в классе – 2 (это значение в КП можно назначать произвольно)

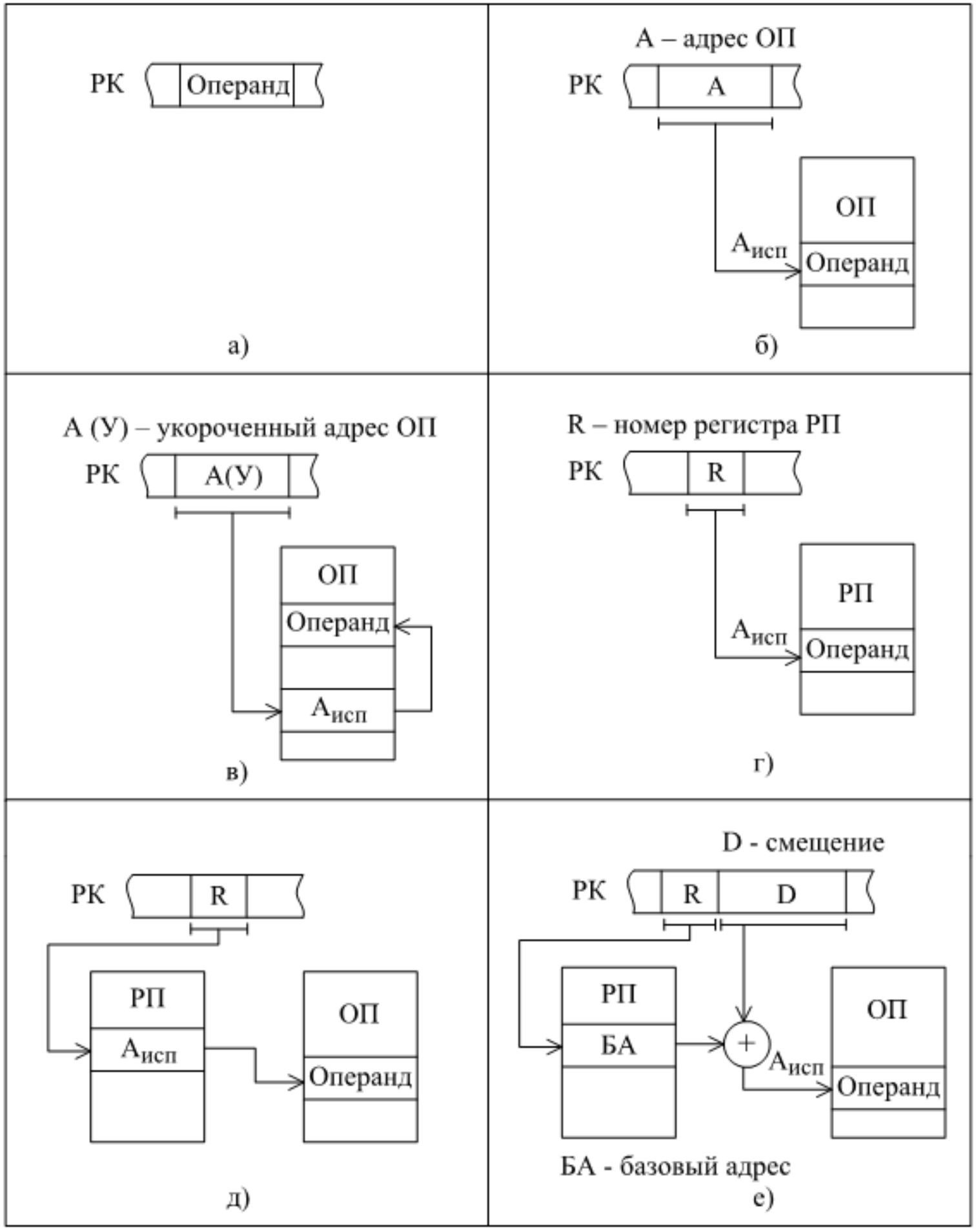


Рисунок 7.2 (МУ КП) –

Способы адресации операндов в команде:

а) непосредственный (Н),
б) прямой (П),
в) косвенный (К),
г) регистровый (Р),
д) косвенный через регистр (КР),
е) относительный (О),
ж) неявный – на рисунке не изображен.

Подробно способы адресации операндов в машинной команде рассмотрены в лекции 8 по дисциплине АКС.

Выбираем способы адресации операндов в разрабатываемых командах в соответствии с требованием задания на КП:

каждый из заданных способов адресации должен быть использован хотя бы один раз (т.е. хотя бы в одной из пяти разрабатываемых команд), если способ адресации не задан вариантом, его использовать нельзя.

Для упрощения алгоритма выборки команды, длину команды мы договорились делать кратной двум байтам.

1. Примеры разработки форматов арифметических команд.

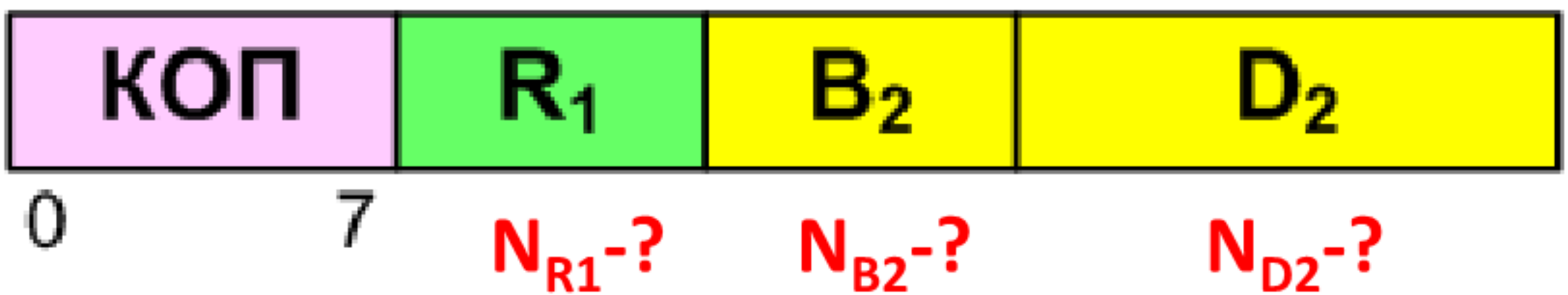
1.1. Сложение чисел формата I4 (целочисленное сложение).

Первый операнд в регистре (используется регистровая адресация, в поле R_1 задается номер регистра).

Второй операнд в ОП (используется относительная адресация: в поле B_2 задается номер регистра, который содержит базовый адрес блока памяти, в поле D_2 задается смещение внутри блока).

Результат помещается в регистр (на место первого операнда).

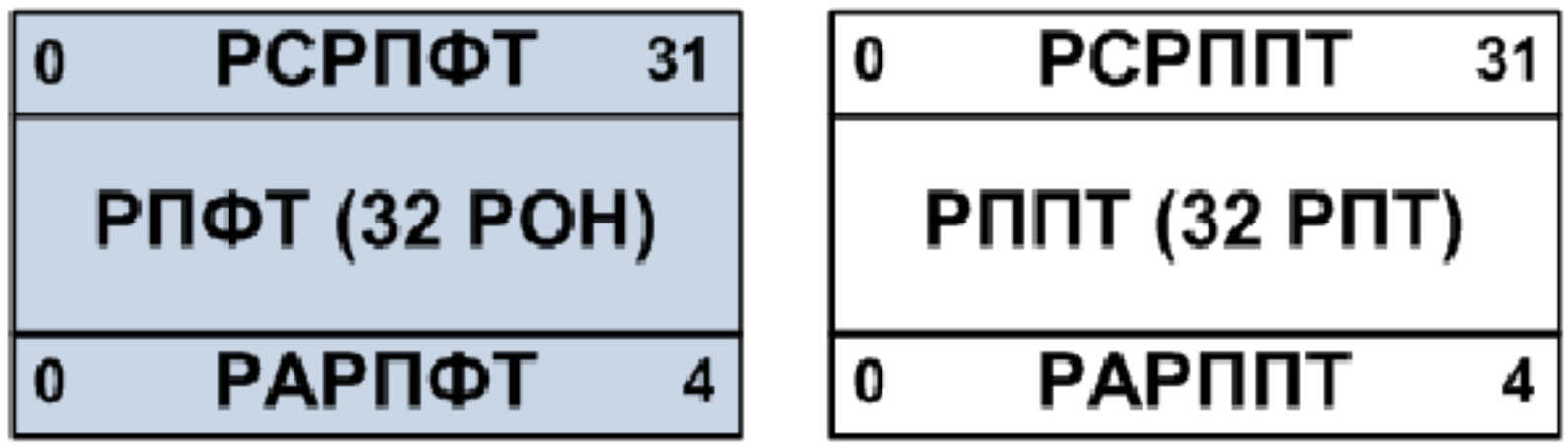
1.1.1. Нужно определить длину полей R_1, B_2 и D_2



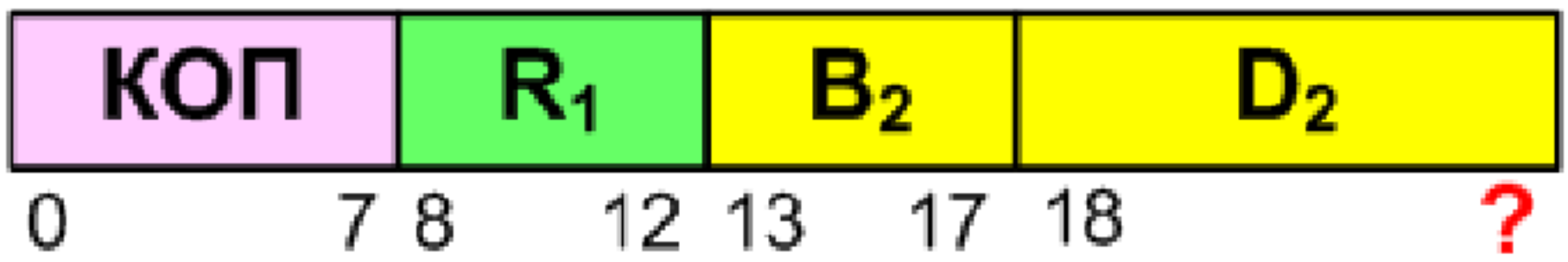
1) Длину полей R_1 и B_2 определяем исходя из объема (числа регистров) и типа регистровой памяти (РП).

Первый случай – раздельная регистровая память

Допустим, $E_{РП}$=32, Тип – Р (раздельная)



В данной команде обращения будут осуществляться к РПФТ (т.к. регистр с номером R_1 хранит целый операнд (I4), а регистр с номером B_2 – адрес (целое беззнаковое число). Чтобы адресоваться к одному из 32 РОН нужен 5-разрядный адрес. Таким образом, длина полей R1 и B2 совпадает с длиной регистра адреса регистровой памяти РАРПФТ: $N_{R_1} = N_{B_2} = N_{РАРПФТ} = \log_2 32 = 5$



2) Определяем размер поля D2.

Мы договорились, что в системе команд будут команды длиной 2 байта и 4 байта (длина команды кратна 2 байтам для упрощения алгоритма выборки команды).

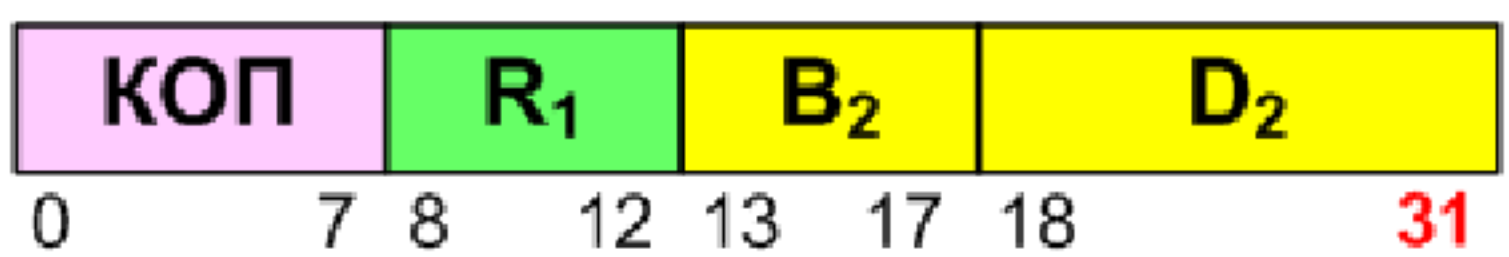
2 байта мы уже превысили, поэтому продлеваем команду до 4 байтов.

Под смещение можем отвести 14 битов (с 18 по 31).

Тогда вся ОП будет считаться разбитой на блоки объемом 2^{14}=16384 байта.

Если $E_{ОП}$=256 Мбайт, то всего таких блоков будет

$$2^{28} : 2^{14} = 2^{14} = 16384$$



Модель ОП при относительной адресации рассмотрена в лекции 8 по дисциплине «Архитектура компьютерных систем.

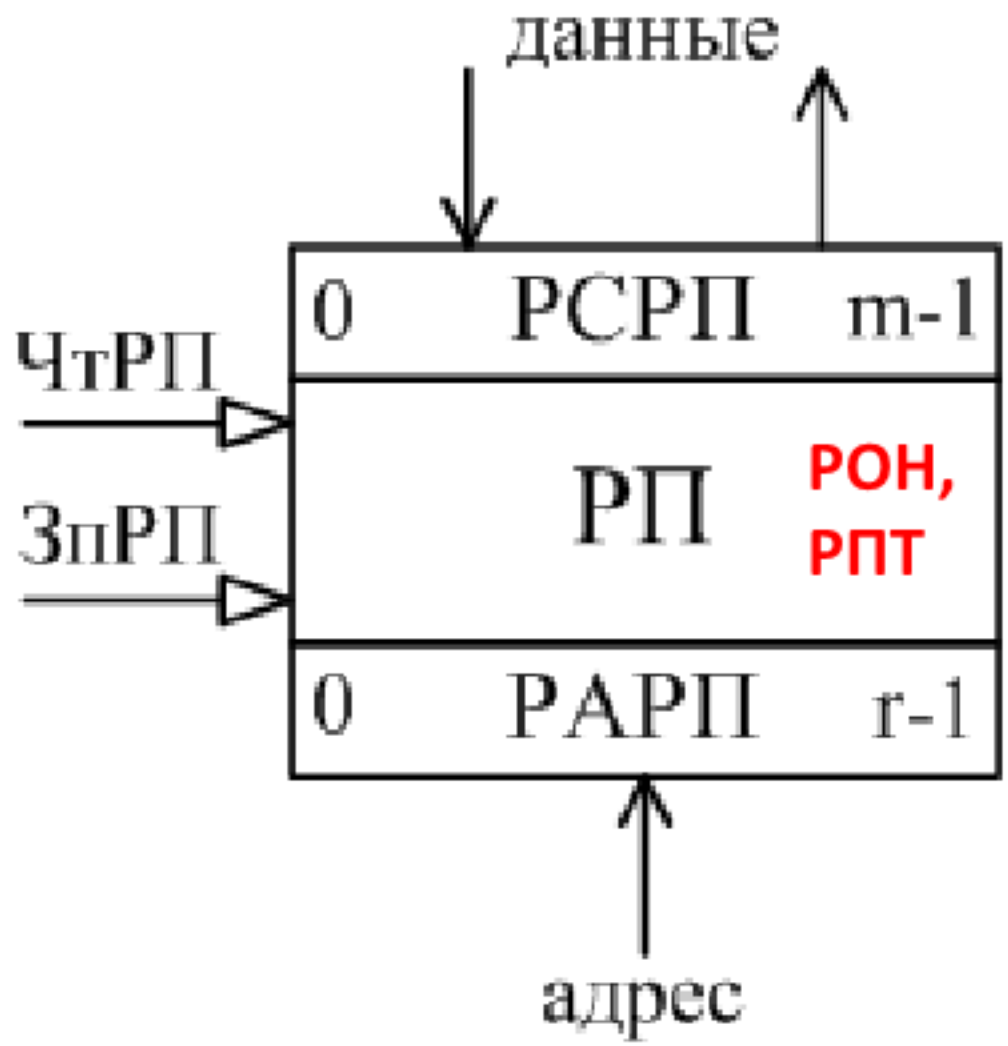
Внимание! Если еще в какой-нибудь из разрабатываемых команд вы решите использовать относительную адресацию, имейте в виду, что номера разрядов для полей B и D должны быть такими же (унификация форматов команд).

Второй случай – универсальная регистровая память

Пусть $E_{РП}$=32, тип - У (универсальная).

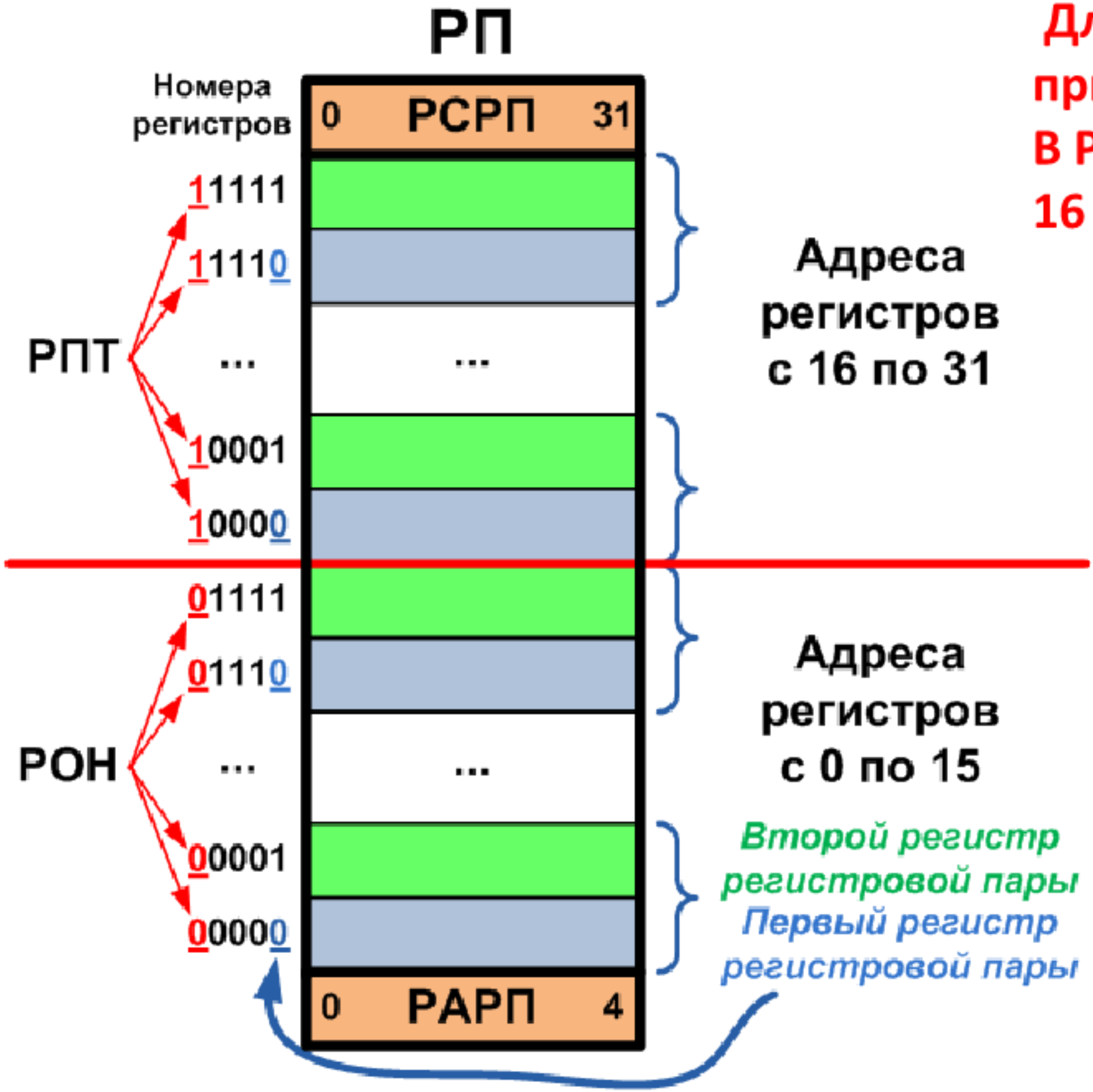
Примем следующие соглашения для универсальной РП:

1) **Первая половина регистров, находящихся в универсальной регистровой памяти будет использоваться в качестве РОН (для хранения операндов с фиксированной точкой), вторая половина регистров в РП будет использоваться в качестве РПТ (для хранения операндов с плавающей точкой).**



2) **Операнд логической операции – двоичный вектор двойной длины (L8) – должен храниться в паре РОН с соседними адресами (первый регистр регистровой пары должен иметь четный номер, иначе – «нарушение спецификации»).**

3) Операнд с плавающей точкой двойной длины (F8) должен храниться в паре РПТ с соседними адресами (первый регистр регистровой пары должен иметь четный номер, иначе – «нарушение спецификации»).



Для нашего примера: $E_{РП}$=32. В РП 16 РОН и 16 РПТ

У всех РОНов старший бит адреса равен нулю, а у всех РПТ – 1 (в команде используем адрес регистра без старшего бита, т.к. тип операнда подразумевается КОП.

Регистровая пара (для хранения L8 или F8) должна начинаться с регистра с четным номером.

Из 32 регистров, находящихся в универсальной РП, регистры с номерами 0 – 15 будут использоваться как РОНы (для операций над данными с фиксированной точкой), регистры с номерами 16-31 будут использоваться как РПТ.

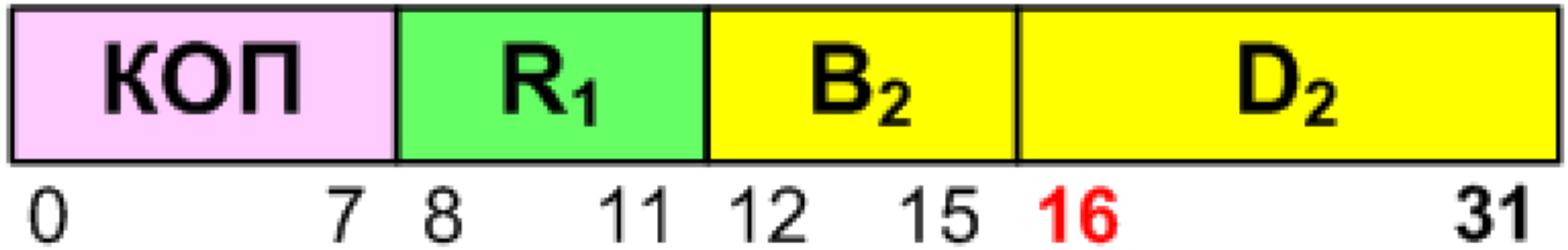
РАРП содержит 5 разрядов для адресации к 32 регистрам. У всех РОНов старший бит адреса равен 0, у всех РПТ – 1. Нет смысла указывать этот бит в команде, т.к. у команд, работающие с операндами с фиксированной точкой, и у команд, работающих с операндами с плавающей точкой, коды в поле КОП – разные. Если КОП подразумевает операцию над числами с фиксированной точкой, то обращение будет к РОН, а если над данными с плавающей точкой, то обращение будет к РПТ.

В формате команды под номер регистра отводится 4-разрядное поле, т.е. указываются номера регистров в диапазоне от 0 до 15 (экономим 1 разряд по сравнению с предыдущим случаем, когда использовалась раздельная РП).

При обращении к РП в микропрограмме выполнения команды, содержимое этого поля РК помещается в младшие разряды РАРП. В старший разряд РАРП заносится 0, если обращение производится к РОН, и 1, если обращение производится к РПТ. Микрооперация формирования адреса РОН выглядит следующим образом: РАРП(0:4):=0.РК(8:11), а адреса РПТ: РАРП(0:4):=1.РК(8:11).

Операнды двойной длины (L8, F8) считываются или записываются в РП за два обращения к ней. При этом операнды должны быть записаны в пару регистров с соседними номерами (считаны из пары регистров). Для контроля правильности считывания таких операндов принято, что первый регистр регистровой пары должен иметь четный номер (в команде – 0, 2, 4, 6, 8, 10, 12 или 14). Если в команде, работающей с операндами двойной длины указан нечетный номер регистра, то такая ситуация (исключительная) должна быть квалифицирована как «нарушение спецификации» с установкой соответствующего флага в РФ.

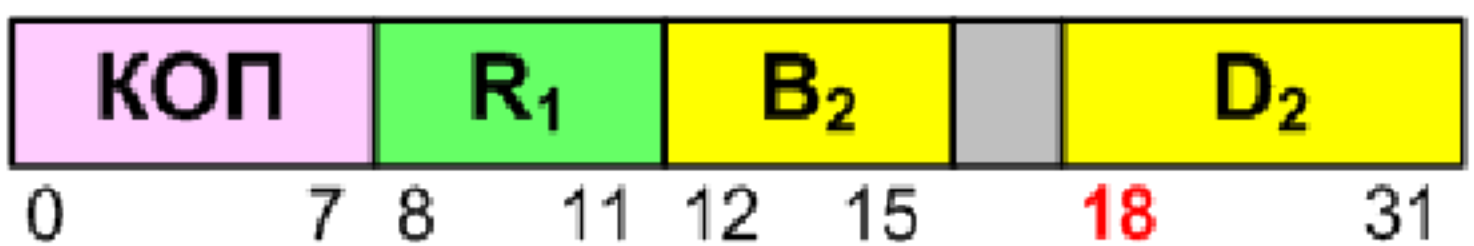
Формат разрабатываемой команды для рассмотренного выше примера будет выглядеть так.

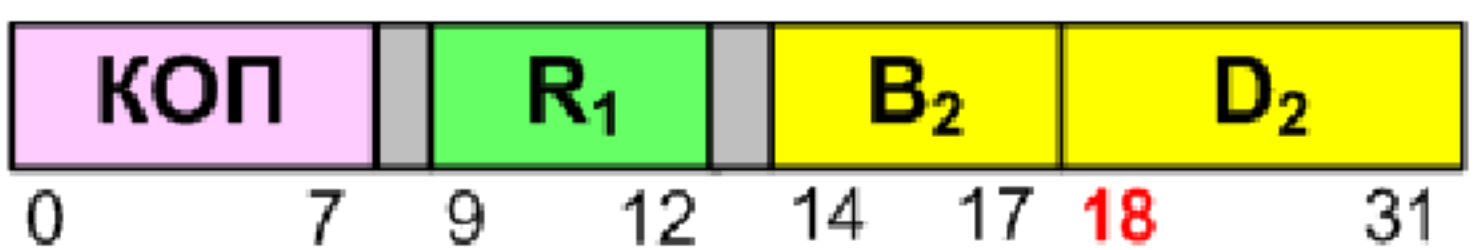


Заметим, что экономия двух битов в полях, где указывается номер регистра, позволила нам увеличить на два бита поле смещения второго операнда D_2, а, значит, разбить память на блоки большей величины. Объем блока составит 2^{16} (вместо 2^{14} в примере с раздельной РП), т.е. 65536 байтов. Но самих блоков будет меньше. При $E_{ОП}$=256 Мбайт блоков будет

$$2^{28} : 2^{16} = 2^{12} = 4096$$ **вместо 16384.**

Если разработчик хочет оставить в модели ОП при относительной адресации 16384 блока по 16384 байта, то является допустимыми следующие форматы команды (на выбор):





Биты, закрашенные серым цветом, не используются (доопределяются нулями при использовании команды в процессе программирования в машинных кодах).

Команды, разработанные согласно варианту задания, должны быть сведены в таблицу следующего вида:

Номер класса	Номер команды	Название	Содержание	ПР	Флаги	Код			
						Двоичный			16-ричный
						01	234	567	
1	1	Чтение из ОП в РП	$(R_1):=((B_2)+D_2)$	нет	A, S	10	001	001	89
…	…	…	…	…	…	…	…	…	…
2	1	Сложение с ФТ	$(R_1):=(R_1)+(R_2)$	=0 <0 >0	ППФТ	00	010	001	11
…	…	…	…	…	…	…	…	…	…
4	2	Логическое «ИЛИ»	$((B_1)+D_1):=((B_1)+D_1) \vee Im_2$	=0 ≠0	A, S	01	100	010	62
…	…	…	…	…	…	…	…	…	…

Рисунок 7.4 (МУ КП) – Структура таблицы, описывающей систему команд ЦОУ (ПР – признак результата, ППФТ – флаг переполнения с фиксированной точкой, A – флаг нарушения адресации, S – флаг нарушения спецификации, Im_2 – непосредственный (immediate) операнд)

1.1.2. Описание и код команды.

Команда сложения целых чисел со знаком (I4) относится к классу 2 (в нашей классификации).

Номер команды в классе назначим равным 1.

Действия, производимые командой, символически можно описать так:

$$(R_1) = (R_1) + ((B_2) + D_2)$$

Используются следующие обозначения:

(R_1) – содержимое регистра с номером R_1 ;

$((B_2) + D_2)$ – содержимое ячейки памяти, адрес которой получен как сумма содержимого регистра B_2 (базового адреса блока памяти) и смещения внутри блока, заданного полем D_2 команды (команда находится в регистре команд РК);

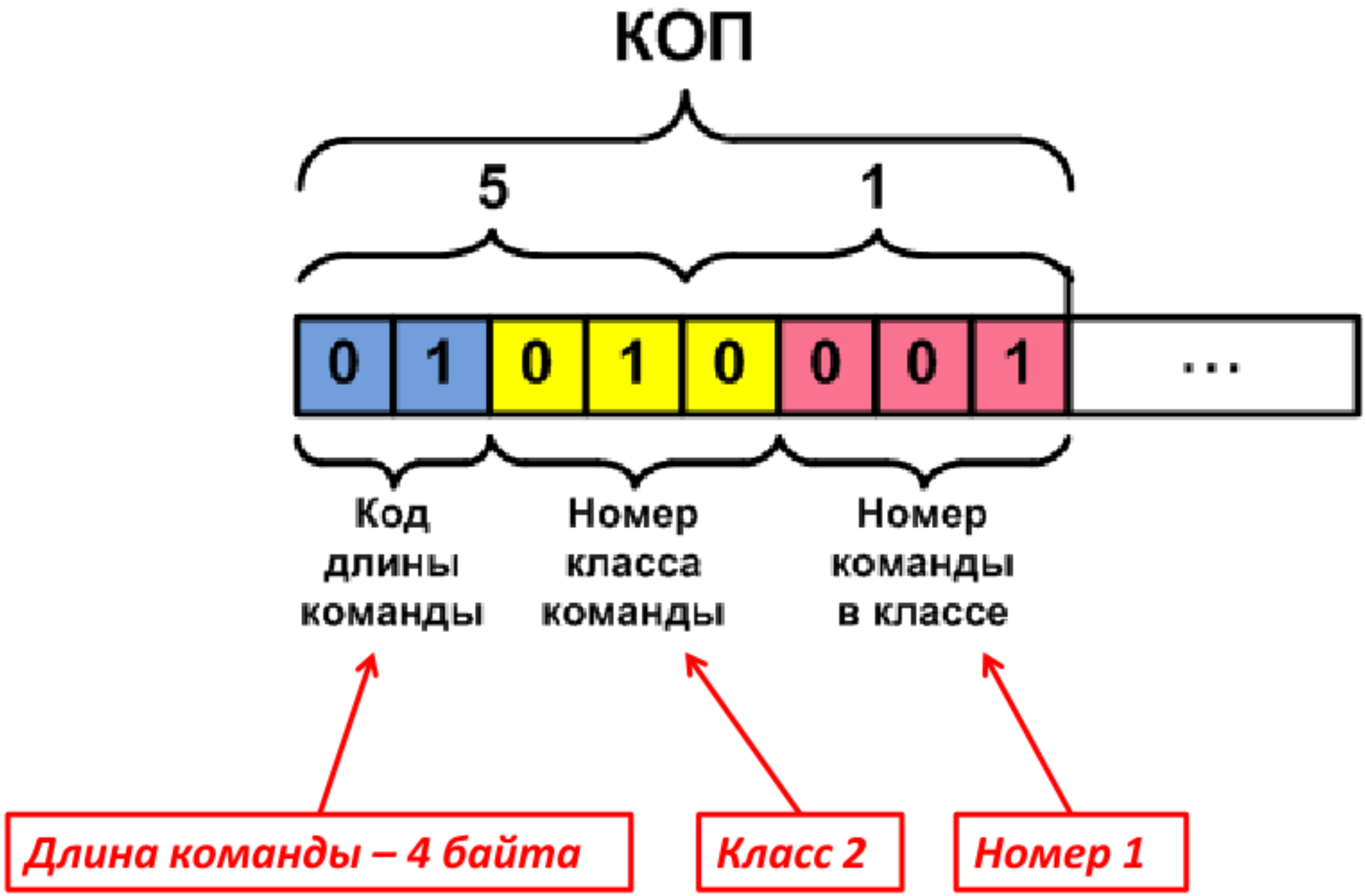
Команда сложения изменяет признак результата в регистре признаков РПР: 00 – результат равен нулю, 01 – результат меньше нуля, 10 – результат больше нуля.

При выполнении команды в АЛУ может возникнуть исключительная ситуация «Переполнение с фиксированной точкой» - ППФТ.

Еще две возможные исключительные ситуации связаны с тем, что один из операндов должен считываться из ОП.
При относительной адресации адрес второго операнда вычисляется путем сложения (на сумматоре адреса СмА) базового адреса и смещения. Если в результате сложения произошло переполнение, значит, команда хочет обратиться к не существующей памяти. Такую исключительную ситуацию обозначим A – «неправильная адресация».

Вторая ошибка возникает, если нарушен принцип целочисленности границ: вычисленный на СмА адрес второго операнда не кратен числу байтов в операнде (для рассматриваемой команды – не кратен 4, т.е. не заканчивается на два нуля). Такую ситуацию будем обозначать S – «неправильная спецификация».

Определим в соответствии с принятыми правилами значение поля КОП для проектируемой команды:



Фрагмент результирующей таблицы, соответствующий рассмотренной команде

Номер класса	Номер команды	Название	Содержание	ПР	Флаги	Код	
						Двоичный 01 234 567	16-ричный
2	1	Сложение с ФТ	$(R_1):=(R1)+((B_2)+D_2)$	=0 <0 >0	A, S, ППФТ	01 | 010 | 001	51
…	…	…	…	…	…	… | … | …	…

Для команды вычитания можно использовать тот же формат. Тогда содержание команды можно символически представить так:

$$(R1)=(R1) - ((B2)+D2)$$

Продолжаем заполнять таблицу «Система команд»

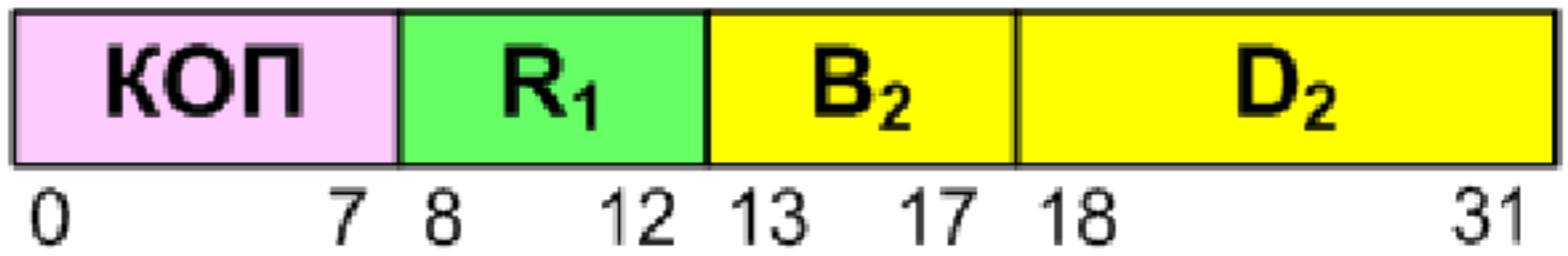
По-мер клас са	Номер коман-ды	Название	Содержание	ПР	Флаги	Код	
						Двоичный 01 234 567	16-рич ный
2	1	Сложение с ФТ	$(R_1):=(R_1)+((B_2)+D_2)$	=0 <0 >0	A, S, ППФТ	01 ¦ 010 ¦ 001	51
2	2	Вычитание с ФТ	$(R_1):=(R_1)-((B_2)+D_2)$	=0 <0 >0	A, S, ППФТ	01 ¦ 010 ¦ 010	52

...

Примечание: вариантом задания на КП предусмотрена реализация только одной команды каждого класса, поэтому таблица будет содержать по одной команде каждого класса.

1.2 Целочисленное деление (формат I4).

Можно использовать тот же формат команды (с учетом типа РП).
Например, для случая с раздельной РП:



Первый операнд – делимое: 64-разрядное целое число (в данном варианте требуется тип данных I8) находится в регистровой паре. Номер первого регистра пары задан в поле R_1 (д.б.четным). Второй регистр пары подразумевается неявно (регистр со следующим (нечетным) номером).

Второй операнд – делитель: 32-разрядное целое число (I4) находится в ОП. Базовый адрес блока ОП задан в поле B_2. Смещение внутри блока задано полем D_2.

Результат: 32 разрядное целое частное (I4) и 32-разрядный целый остаток от деления записываются соответственно в первый и второй регистр вышеупомянутой регистровой пары (на место делимого).

Команда деления не меняет признак результата в РПР.

Исключительные ситуации: «деление на ноль» (от АЛУ), «неправильная адресация» (от СмА), «неправильная спецификация» (если в поле R1 задан нечетный номер регистра или вычисленный на СмА адрес второго операнда не заканчивается на 00).

Символически работу команды целочисленного деления можно описать так:

$$(R_1):=(R_1).(R_1+1) / ((B_2)+D_2);$$
$$(R_1+1):=(R_1).(R_1+1) \% ((B_2)+D_2)$$

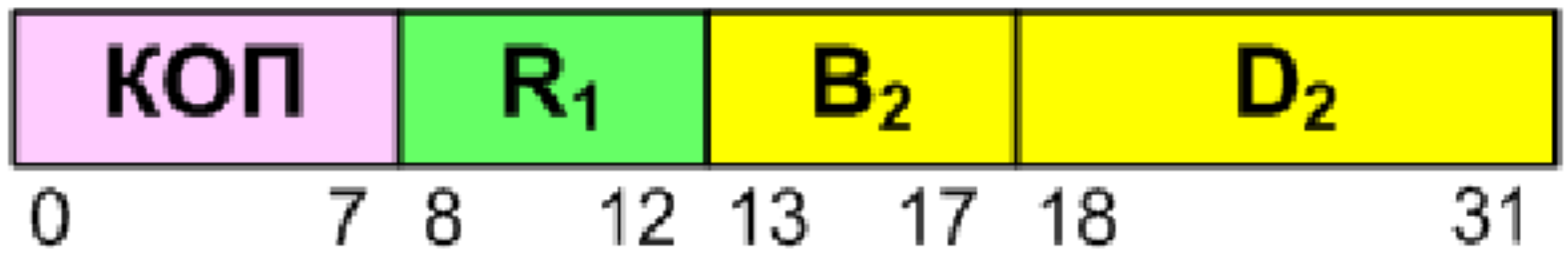
Продолжим формировать таблицу «Система команд»

Номер класса	Номер команды	Название	Содержание	ПР	Флаги	Код			
						Двоичный 01	234	567	16-ричный
2	1	Сложение с ФТ	$(R_1):=(R_1)+((B_2)+D_2)$	=0 <0 >0	Λ, S, ПФТ	01	010	001	51
2	2	Вычитание с ФТ	$(R_1):-(R_1) - ((B_2)+D_2)$	=0 <0 >0	A, S, ПФТ	01	010	010	52
2	3	Деление с ФТ	$(R_1):=(R_1).(R_1+1)/((B_2)+D_2);$ $(R_1+1):=(R_1).(R_1+1)\%((B_2)+D_2)$	нет	A, S, Дел0	01	010	011	53

…

1.3 Целочисленное умножение (формат I4).

Можно использовать тот же формат команды (с учетом типа РП).
Например, для случая с раздельной РП:



Первый операнд – множимое: 32-разрядное целое число (I4) находится в регистре, номер которого задан в поле R_1.

Второй операнд – множитель: 32-разрядное целое число (I4) находится в ОП. Базовый адрес блока ОП задан в поле B_2. Смещение внутри блока задано полем D_2.

Результат: 64 разрядное произведение (в данном варианте требуется тип данных I8) помещается в регистровую пару, начинающуюся с регистра, номер которого задан полем R_1. Отсюда следует, что этот номер должен быть четным.

Команда умножения не меняет признак результата в РПР.

Исключительные ситуации: «неправильная адресация» (старший бит СмА равен 1), «неправильная спецификация» (в поле R_1 задан нечетный номер регистра или вычисленный на СмА адрес второго операнда не заканчивается на 00).

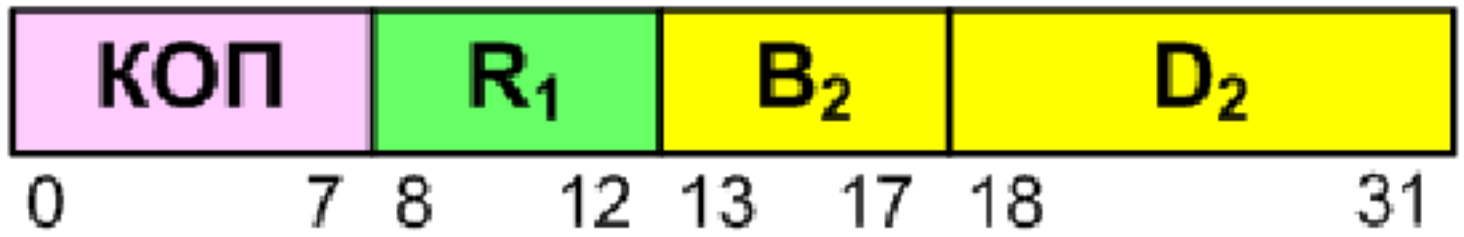
Символически работу команды целочисленного умножения можно описать так:

$$(R_1).(R_1+1):=(R_1)*((B_2)+D_2)$$

Номер класса	Номер команды	Название	Содержание	ПР	Флаги	Код			
						Двоичный 01	234	567	16-ричный
2	1	Сложение с ФТ	$(R_1):=(R_1)+((B_2)+D_2)$	=0 <0 >0	A, S, ППФТ	01	010	001	51
2	2	Вычитание с ФТ	$(R_1):=(R_1)-((B_2)+D_2)$	=0 <0 >0	A, S, ППФТ	01	010	010	52
2	3	Деление с ФТ	$(R_1):=(R_1).(R_1+1)/((B_2)+D_2);$ $(R_1+1):=(R_1).(R_1+1)\%((B_2)+D_2)$	нет	A, S, Дел0	01	010	011	53
2	4	Умножение с ФТ	$(R_1).(R_1+1):=(R_1)*((B_2)+D_2);$	нет	A, S	01	010	100	54

1.4. Арифметические команды, инициирующие операции над числами с плавающей точкой.

Можно использовать тот же формат команды (с учетом типа и размера РП). Например, для случая с раздельной РП объемом 32 регистра:



Если задан формат F4, операнды и результат – 32-разрядные, результат помещается на место первого операнда (номер соответствующего регистра задается полем R_1.

Если задан формат F8, операнды и результат – 64-разрядные, результат помещается в регистровую пару, в которой сначала находился первый операнд (номер первого регистра пары задается в поле R1 – должен быть четным).

При умножении и делении используется округление мантисс (поэтому длина операндов и результата совпадает).

Команды сложения и вычитания меняют признак результата в РПР, команды умножения и деления – нет.

Исключительные ситуации: «исчезновение порядка», «переполнение порядка», «потеря значимости» (от АЛУ), «неправильная адресация» (старший бит СмА равен 1), «неправильная спецификация» (в поле R_1 задан нечетный номер регистра (для F8) или вычисленный на СмА адрес второго операнда не заканчивается на 00 (для F4) или на 000 (для F8)).

Номер класса	Номер команды	Название	Содержание	ПР	Флаги	Код			
						Двоичный 01	234	567	16-ричный
3	1	Сложение с ПТ **Для F4**	$(R_1):=(R_1)+((B_2)+D_2)$	=0 <0 >0	A, S, ППор, ИПор, ПЗнач	01	011	001	59
3	2	Вычитание с ПТ **Для F4**	$(R_1):=(R_1)-((B_2)+D_2)$	=0 <0 >0	A, S, ППор, ИПор, ПЗнач	01	011	010	5A
3	3	Вычитание с ПТ **Для F8**	$(R_1).(R_1+1):=(R_1).(R_1+1)-((B_2)+D_2);$	=0 <0 >0	A, S, ППор, ИПор, ПЗнач	01	011	011	5B
3	4	Деление с ПТ **Для F4**	$(R_1):=(R_1)/((B_2)+D_2);$	нет	A, S, ППор, ИПор, ПЗнач, Дел0	01	011	100	5C
3	5	Умножение с ПТ **Для F8**	$(R_1).(R_1+1):=(R_1).(R_1+1)*((B_2)+D_2);$	нет	A, S, ППор, ИПор, ПЗнач	01	011	101	5D
3	6	Умножениес ПТ **Для F8**	$(R_1).(R_1+1):=(R_1).(R_1+1)*((R_2));$	нет	A, S, ППор, ИПор, ПЗнач	00	011	110	1E

Перечень возможных команд для работы с вещественными числами (далеко не полный)

?

Чем шестая команда отличается от пятой?

3	5	Умноже-ние с ПТ Для F8	$(R_1).(R_1+1):=$ $(R_1).(R_1+1)$ * $((B_2)+D_2);$	нет	A, S, ППор, ИПор, ПЗнач	01	011	101	5D
3	6	Умноже-ниес ПТ Для F8	$(R_1).(R_1+1):=$ $(R_1).(R_1+1)$ * $((R_2));$	нет	S, ППор, ИПор, ПЗнач	00	011	110	1E

1) **Способом адресации второго операнда. В некоторых вариантах на КП относительная адресация не предусмотрена. Можно использовать, например, косвенно-регистровую адресацию. Тогда формат команды будет выглядеть так:**

Первый операнд (регистровая адресация) находится в двух РПТ с номерами R_1 и R_1+1 (R_1 – четный). Второй операнд (косвенно-регистровая адресация) находится в ОП, а его адрес – в РОН с номером R_2.

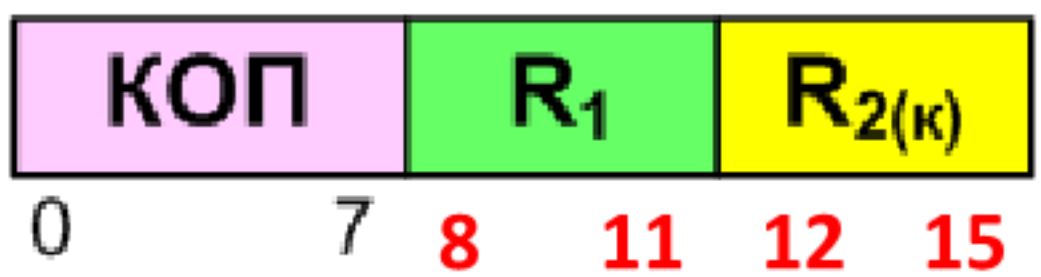
С помощью 32-разрядного адреса (разрядность регистра – 32 бита) можно адресоваться к 2^{32} байтам памяти. При объеме памяти 2^{28} байт (256Мбайт для нашего примера), старшие четыре (32-28=4) бита РОН использоваться не будут. В микропрограмме при получении адреса из регистра будем их просто обрезать, поэтому ситуации «нарушение адресации в этом случае не будет (СмА при косвенно-регистровой адресации не используется).

3	5	Умноже-ние с ПТ **Для F8**	$(R_1).(R_1+1):=$ $(R_1).(R_1+1)$ * $((B_2)+D_2)$;	нет	A, S, ППор, ИПор, ПЗнач	01	011	101	5D
3	6	Умноже-ниес ПТ **Для F8**	$(R_1).(R_1+1):=$ $(R_1).(R_1+1)$ * $((R_2))$;	нет	S, ППор, ИПор, ПЗнач	00	011	110	1E

Исключительная ситуация «неправильная спецификация» остается (если номер регистра, указанный в поле R_1, – нечетный, или адрес операнда, извлеченный из регистра R_2, не заканчивается на 000).

Чем еще отличаются эти две команды?

3	5	Умноже-ние с ПТ **Для F8**	$(R_1).(R_1+1):=$ $(R_1).(R_1+1)$ * $((B_2)+D_2)$;	нет	A, S, ППор, ИПор, ПЗнач	01	011	101	5D
3	6	Умноже-ниес ПТ **Для F8**	$(R_1).(R_1+1):=$ $(R_1).(R_1+1)$ * $((R_2))$;	нет	S, ППор, ИПор, ПЗнач	00	011	110	1E



Очевидно, что шестая команда занимает 2 байта (ее код начинается на 00).

В каких случаях это возможно?

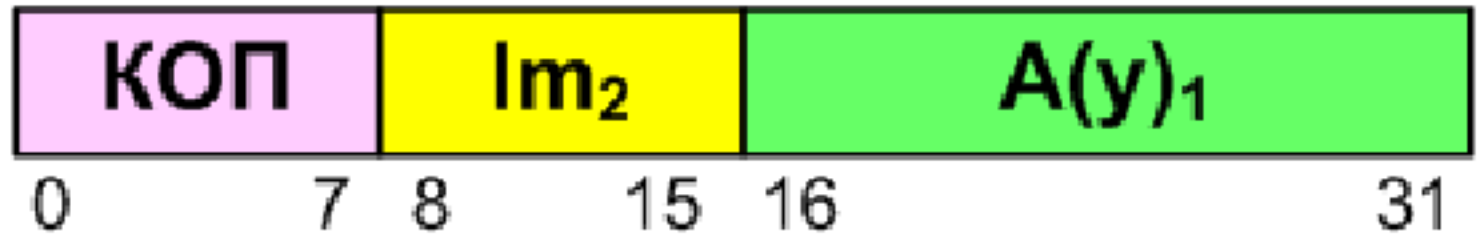
1) **$E_{РП}$=32, Тип – У;**
2) **$E_{РП}$<=16, Тип – У или Р.**

2. Примеры разработки форматов логических команд.

2.1. Поразрядное И, ИЛИ, Исключающее ИЛИ. Операнды и результат – двоичные вектора длиной 1 байт (L1).

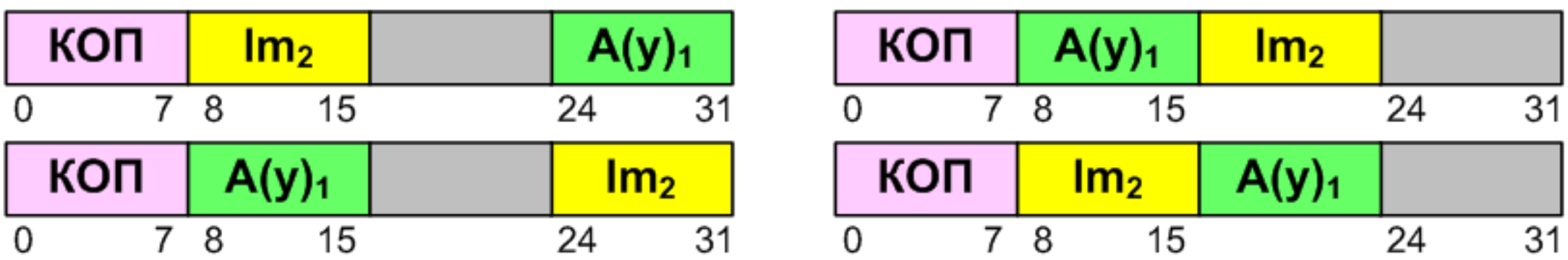
При таком малом размере операндов для второго операнда удобно использовать непосредственную адресацию. Для первого операнда и результата можно использовать регистр (его младший байт) или байт памяти, адрес которого можно задать в соответствии с принципом относительной, косвенно-регистровой или косвенной адресации (в КП должны быть использованы все заданные способы адресации).

Возможный формат:



Как видно из рисунка, второй операнд хранится непосредственно в команде в поле Im_2 (от immediate – непосредственный). Для первого операнда (и результата) используется косвенная адресация. В поле $A(y)_1$ задается адрес слова (!) ОП, в котором находится адрес первого операнда (байта) в ОП. Поскольку длину команды все равно придется продлить до 32 битов, под *адрес адреса первого операнда* (мы назвали его укороченным – A(y)) можно отвести 16 битов (с 16 по 31). Это значит, размер начального блока ОП (с относительно небольшими длинами адресов), в котором программист сможет использовать слова для задания *адреса адреса* составит 2^{16} байтов=65536 байтов=16384 слова (модель ОП при косвенной адресации рассмотрена в лекции 8 по дисциплине АКС).

Обычно блоки памяти такого большого размера для подобных служебных целей не используют (память нужна для хранения данных и программ!).
Поэтому представляется целесообразным выделить для задания укороченных адресов блок памяти гораздо меньшей величины (64, 32, 16 или даже 8 слов программисту вполне хватит). Если проектировщик принял решение выделить под косвенную адресацию блок в начале ОП длиной 64 слова (256 байтов), то предыдущий формат команды нужно изменить (выберите один из четырех вариантов, объясните, почему под укороченный адрес теперь отводится не 16, а всего 8 разрядов):



Внимание! Если еще в какой-то команде вы будете использовать косвенную адресацию, укороченный адрес должен задаваться в тех же разрядах команды – унификация форматов команд, то же справедливо и для непосредственного операнда).

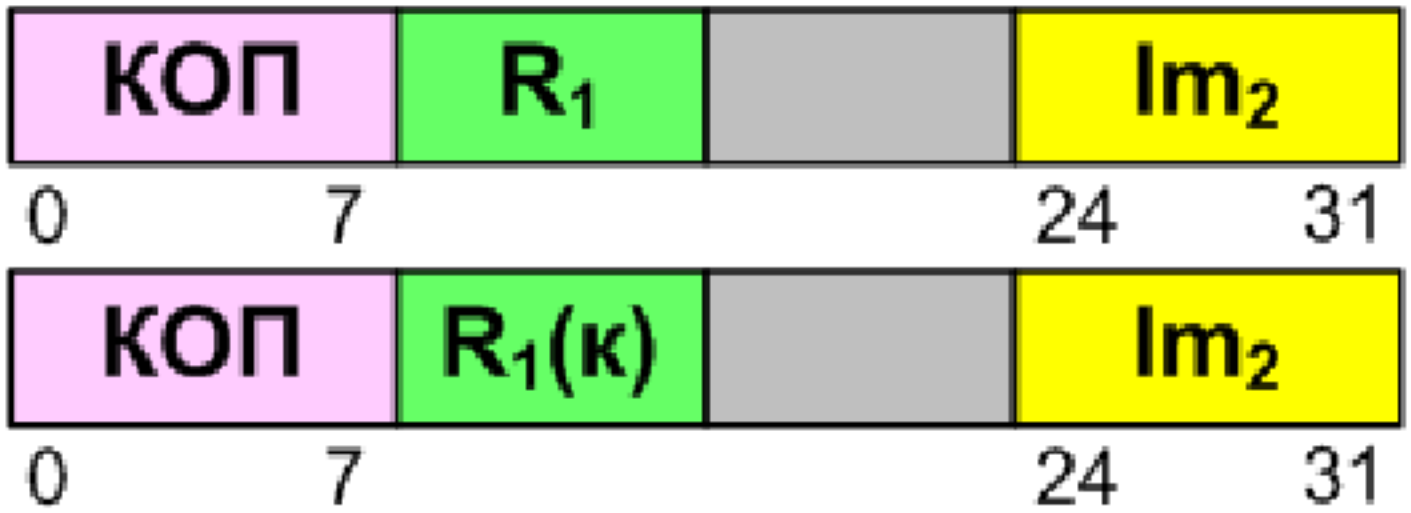
Логические поразрядные операции меняют признак результата в регистре признаков (00 – =0; 01 – ≠0).

Исключительная ситуация для команды приведенного формата может возникнуть только одна: «нарушение спецификации» (если укороченный адрес в поле $A(y)_1$ не заканчивается на 00 (это адрес слова).

Примечание. Адрес операнда, считанный из слова, на которое указывает укороченный адрес, в данном случае, может быть любым, даже нечетным, т.к. это адрес байта (L1).

Символически действия команды можно описать так: $((Ay_1)) = ((Ay_1)) \vee Im2$

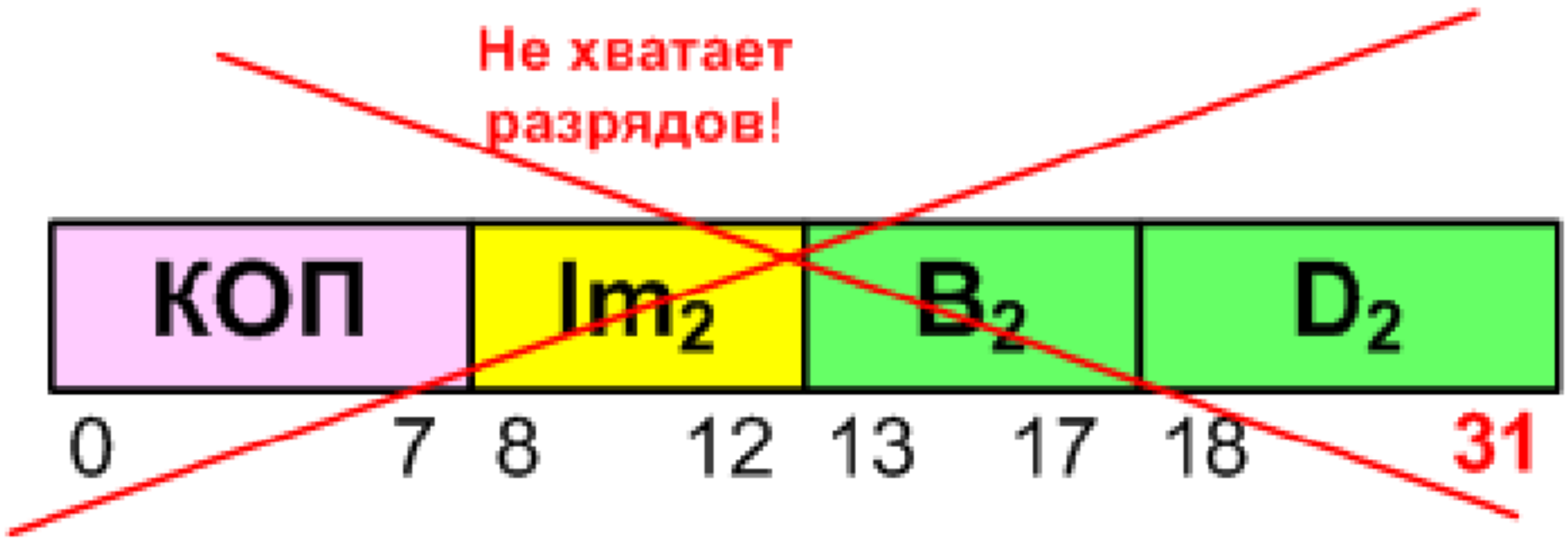
Если вариант задания не предусматривает косвенной адресации, можно использовать следующие форматы команд:



Для тренировки самостоятельно опишите эти форматы.

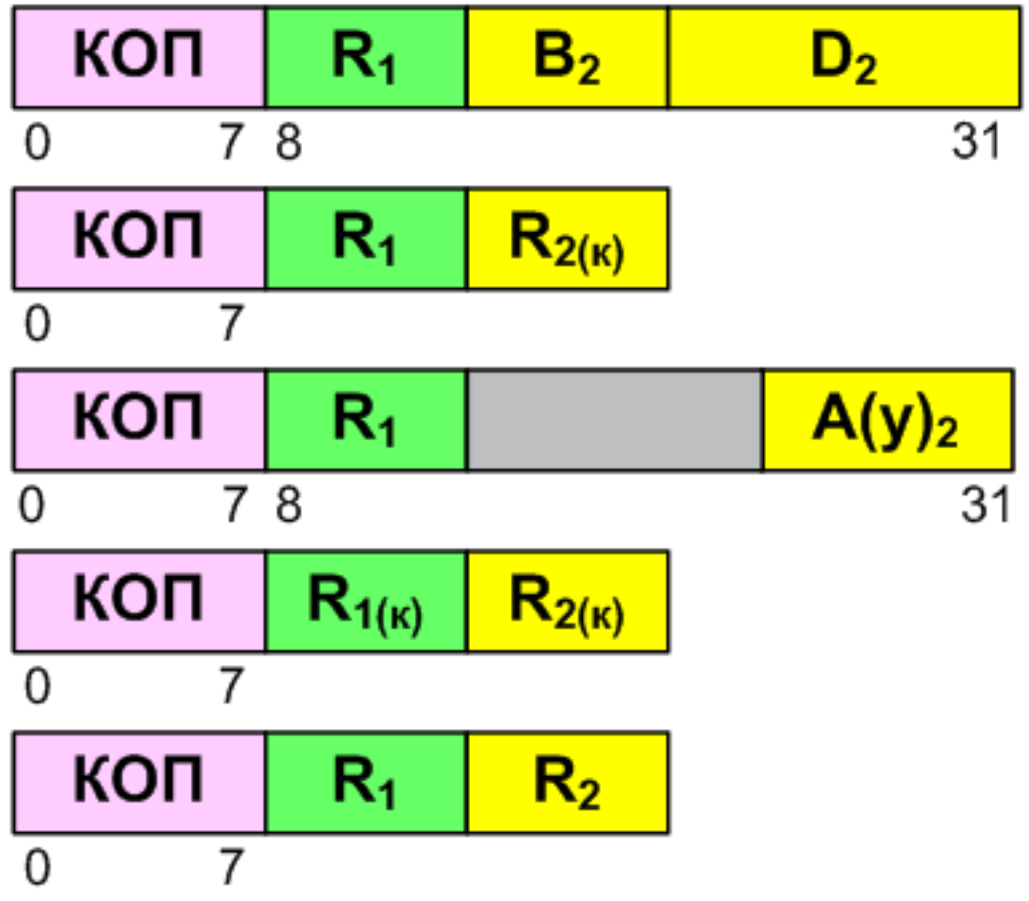
Число разрядов в поле номера регистра зависит от объема и типа РП.

А вот относительную адресацию использовать не удастся, т.к. принято ограничение: максимальный размер команды равен 4 байтам.



Если в команда, инициирующая поразрядную логическую операцию, должна работать с двоичными векторами, длина которых больше одного байта, то непосредственную адресацию использовать нецелесообразно (не уложимся в 4 байта ;-). Кроме того, есть варианты КП, в которых непосредственная адресация не предусмотрена.

Тогда можно использовать, например, следующие форматы:



Обоснуйте выбор одного из этих форматов или предложите свой.

Проставьте номера разрядов для полей команды (согласно своему варианту задания).

Опишите (для тренировки) все вышеприведенные команды по следующему плану:

1) **какую операцию инициирует команда и к какому типу она принадлежит;**
2) **типы и форматы данных, обрабатываемых командой;**
2) **где находятся операнды и какие способы адресации для них используются;**
3) **куда будет помещен результат выполнения операции;**
4) **меняет ли команда признак результата в РПР;**
5) **какие исключительные ситуации могут возникнуть в процессе выполнения команды (адреса операндов, находящихся в памяти должны быть проверены в микропрограмме исполнения команды на кратность числу байтов в операнде, а при относительной адресации – еще и на вхождение в диапазон существующих адресов ОП);**
6) **представьте действия, выполняемые командой в символическом виде;**
7) **закодируйте поле КОП команды в двоичном и шестнадцатеричном виде и запишите команду в таблицу «Система команд».**

2.2. Команды сдвигов

Команда сдвига имеет два операнда: первый операнд – данное, которое сдвигается, второй операнд – константа сдвига (задает на сколько разрядов нужно сдвинуть первый операнд).

В вариантах задания используются следующие обозначения

←1 – логический сдвиг влево двоичного вектора длиной в слово (L4);

←2 – логический сдвиг влево двоичного вектора длиной в двойное слово (L8);

1→ – логический сдвиг вправо двоичного вектора длиной в слово (L4);

2→ – логический сдвиг вправо двоичного вектора длиной в двойное слово (L8);

Возникает вопрос: чему равна максимальная константа сдвига и, соответственно, сколько разрядов она будет занимать?

При логическом сдвиге разряды, выдвигаемые за разрядную сетку теряются, а освобождающиеся разряды доопределяются нулями.

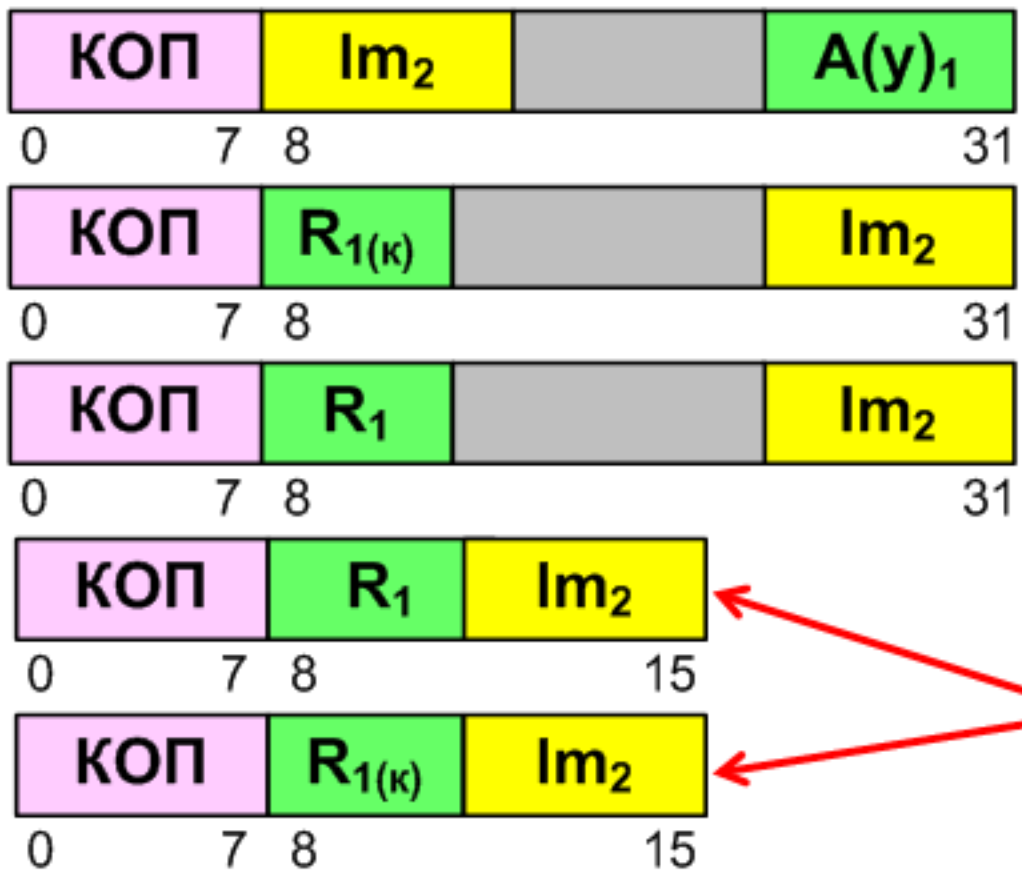
Если мы сдвинем слово (L4) на 32 разряда, оно полностью обнулится (такая операция будет эквивалентна операции присвоения нуля этому слову). Поэтому есть смысл производить сдвиг слова, только если константа сдвига находится в диапазоне от 1 до 31. В данном случае под константу сдвига целесообразно отвести 5 разрядов ($\log_2 32=5$).

В операциях сдвига двойного слова (F8) константа сдвига должна находиться в диапазоне от 1 до 63, а, значит ее длина – 6 разрядов ($\log_2 64=6$).

Поскольку константа сдвига занимает меньше 1 байта, целесообразно хранить ее непосредственно в команде (тогда под ее хранение не нужно тратить РОН или байт ОП, не нужно также тратить время на ее извлечение из ОП – будем брать ее прямо из регистра команд РК).

Сдвигаемый операнд может храниться в одном РОН (при длине L4) или в паре РОН (при длине F8). При использовании пары РОН в команде должен быть задан номер первого регистра пары (четный) . Другой вариант: хранить сдвигаемый операнд в ОП и для задания его адреса использовать косвенную или косвенно-регистровую адресацию.

Некоторые из возможных форматов команд:



Обоснуйте выбор одного из этих форматов или предложите свой.

Проставьте номера разрядов для полей команды (согласно своему варианту задания).

При каких типах операндов, объемах и типах РП можно использовать короткие (двухбайтные) команды?

Будем считать, что команда сдвига меняет признак результата в РПР ((=0; ≠0).

Исключительные ситуации: «неправильная спецификация» при указании номера первого регистра регистровой пары (для F8), при указании адреса памяти (не оканчивается на соответствующее длине операнда число нулей).

Примеры символических описаний действий, выполняемых командой сдвига:

$$(R_1) := L_{\mathbf{Im}_2}(R_1);$$

$$(R_1).(R_1+1) = R_{\mathbf{Im}_2}(R_1).(R_1+1);$$

$$((R_1)) := L_{\mathbf{Im}_2}((R_1));$$

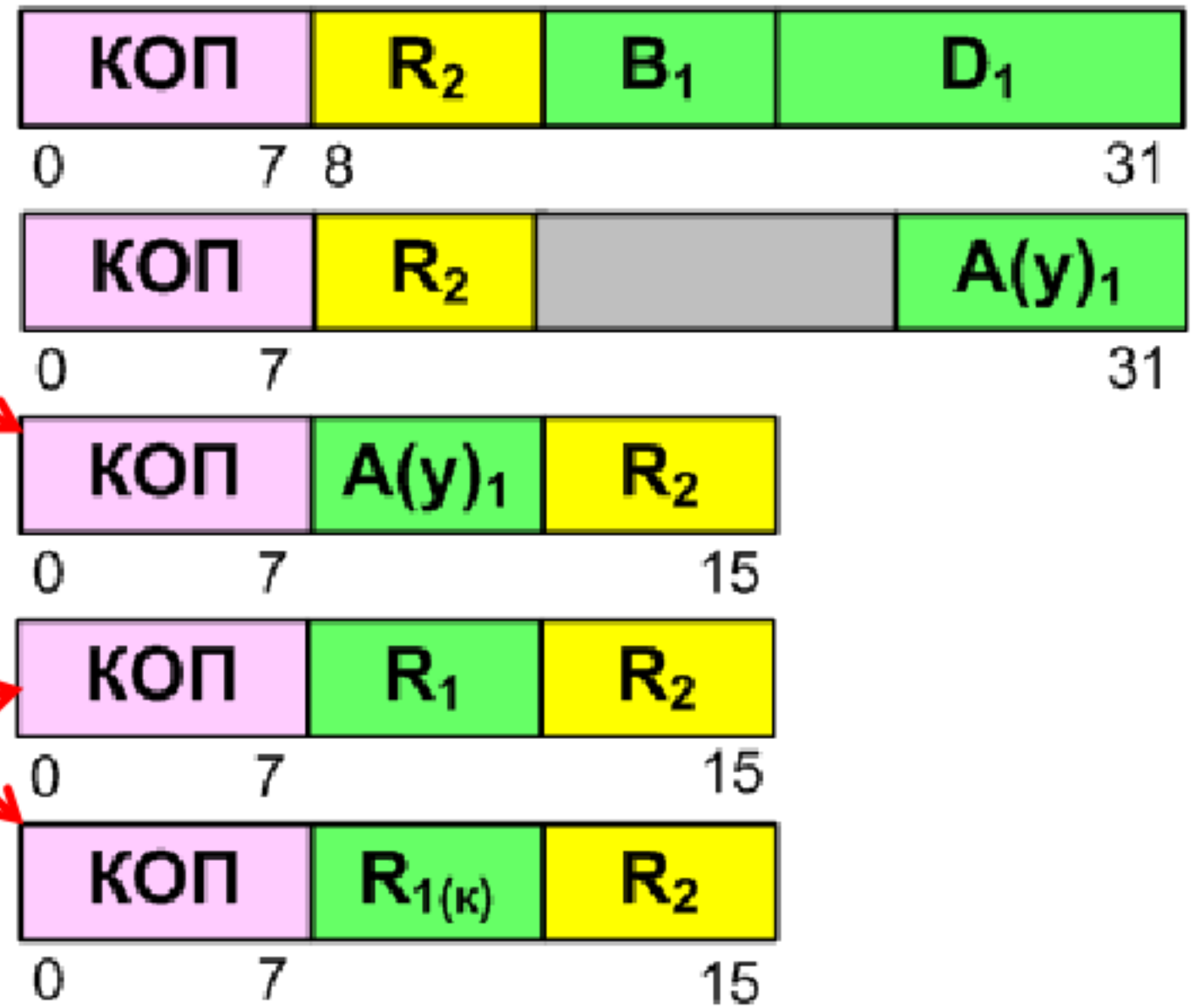
$$((Ay_1)) := R_{\mathbf{Im}_2}((Ay_1)).$$

Поставьте в соответствие эти описания приведенным на предыдущем слайде форматам команд.
Закодируйте поле КОП для выбранной команды сдвига.

В ряде вариантов задания непосредственная адресация не предусмотрена. В этом случае константу сдвига лучше всего хранить в РОН. Было бы странно хранить эти 5 или 6 битов (даже не байт) в ОП. Сдвигаемый операнд как и в предыдущем случае, можно хранить как в РОН, так и в ОП, используя при этом любой из подходящих видов адресации (косвенную, косвенно-регистровую или относительную). **Примеры форматов:**

Какой объем начального блока ОП (для задания адресов операндов) нужно выбрать, чтобы использовать короткий формат. Какой еще параметр здесь будет влиять на длину команды?

При каких объемах и типах регистровой памяти мы сможем использовать короткие (двухбайтные) команды, а при каких придется продлить подобную команду до 4 байтов?



Система команд. Примеры логических команд

Но-мер клас са	Номер коман-ды	Название	Содержание	ПР	Флаги	Код			
						Двоичный 01	234	567	16-рич ный
4	1	Поразрядное ИЛИ	$(R_1):=(R_1) \vee Im_2$	=0 ≠0		01	100	001	61
4	2	Поразрядное ИЛИ	$((R_1)):=((R_1)) \vee Im_2$	=0 ≠0		01	100	010	62
4	3	Поразрядное И	$(R1):= (R_1) \wedge ((Ay_1))$	=0 ≠0	S	01	100	011	63
4	4	Поразрядное И	$(R1):= (R_1) \wedge ((R_2))$	=0 ≠0	S	00	100	100	24
4	5	Поразрядное исключающее ИЛИ	$(R_1):=(R_1) \oplus ((B_2)+D_2)$	=0 ≠0	A, S	01	100	101	65

Как вы думаете, почему здесь нет флага S ?

Прокомментируйте использованные в командах способы адресации.

Система команд. Примеры команд сдвига

Номер класса	Номер команды	Название	Содержание	ПР	Флаги	Код			
						Двоичный 01 234 567			16-ричный
5	1	Логический сдвиг влево слова	$(R_1) = L_{\mathrm{Im}_2}(R_1)$	=0 ≠0		01	101	001	69
5	2	Логический сдвиг влево слова	$((R_1)) = L_{\mathrm{Im}_2}((R_1))$	=0 ≠0	**S**	01	101	010	6A
5	3	Логический сдвиг влево двойного слова	$((Ay_1)).((Ay_1)+4) =$ $L_{\mathrm{Im}_2}((Ay_1)).((Ay_1)+4)$	=0 ≠0	**S**	01	101	011	6B
4	4	Логический сдвиг вправо слова	$(R_1) = R_{\mathrm{Im}_2}(R_1)$	=0 ≠0		01	101	100	6C
4	5	Логический сдвиг вправо двойного слова	$(R_1).(R_1+1) =$ $R_{R_2}(R_1).(R_1+1)$	=0 ≠0	**S**	00	101	101	2D

Прокомментируйте использованные способы адресации операндов, причины присутствия или отсутствия флага S в командах, длины команд.

Нужно различать процесс проектирования команд (микропрограммирование) и процесс использования команд (программирование).
Когда ЭВМ вместе со своей системой команд спроектирована и воплощена в «железо», мы можем писать для нее программу, используя машинные команды.

Но нашей задачей в данном КП является именно «микропрограммирование», используемое для создания системы команд процессора.