IP -> BR, AR

Стрелка означает простое копирование значения слева в регистр(ы) справа

MEM(AR)

означает данные из памяти по адресу, который в данном случае находится в AR

IF CR(X) = 1 или 0

Если скобки стоят рядом после регистра, то они означают получение определенного бита из регистра, в данном случае получение бита из CR под номером X (биты нумеруются справа налево с нуля)

Регистр не обязательно CR, может быть любой

GOTO {метка} @XX

Означает переход на следующую микрокоманду под номером XX

AR + CR или 1 + IP

Простое сложение значений регистров и/или чисел, проще считать в двоичном коде, чтоб не ошибиться

~0

Знак ~ означает инверсию, биты равные нулю, становятся единицей и наоборот, равные единице – нулем

~0 = $FFFF_{16}$ = -1 в привычном виде

AC & DR

Знак & означает логическое умножение, побитовое умножение – бит под номером i из первого регистра и бит под номером i из второго умножаются

A	B	A & B
0	0	0
0	1	0
1	0	0
1	1	1

PS(C) PS(Z) PS(N) PS(V)

Получение бита переноса, нуля, отрицательного, переполнения соответственно из регистра состояния

Все биты регистра состояния

Содержимое регистра состояния

Бит	Мнемоника	Содержимое
0	C	Перенос
1	V	Переполнение
2	Z	Нуль
3	N	Знак
4	0	0 - используется для организации безусловных переходов в МПУ
5	EI	Разрешение прерываний
6	INT	Прерывание
7	W	Состояние тумблеров "РАБОТА/ОСТАНОВ" (1 - "РАБОТА")
8	P	Программа

Бит переноса

Если считать два значения в столбик, то можно записывать единицы сверху (держим в уме со школы), значение над 16 битом (которого нет, он слева от 15го, мнимый) означает бит переноса

В беззнаковых числах, если прибавить 1 к 15 или вычесть из 0 единицу произойдет потеря значения числа, которая сопровождается возникновением переноса. Этот перенос в конструкции ЭВМ учитывается в бите С (Carry), который служит сигналом программисту, что произошла ситуация потеря значения, и ее необходимо обработать отдельно.

В знаковых числах точка возникновения ошибки расположена между представлением чисел -8 и 7. При вычитании из -8 единицы или прибавления единицы к 7 возникает переполнение (OVerflow), которое в ЭВМ контролируется битом переполнения. Напомним, что данный бит обозначается в БЭВМ - V, а в разных современных архитектурах процессоров O, V или OV. АЛУ определяет переполнение по следующему правилу: если поразрядные переносы в знаковом и старшем разряде одновременно отсутствуют или присутствуют - значит переполнения нет, если присутствует только в одном – значит переполнение знаковой разрядной сетки есть.

```
1101 ← биты совпадают   1000  ← биты различны   0111  ← биты различны
+1101 -3                +1000 -8                +0111 7
 0101 +5                 1111 -1                 0001 1
---------               ---------               ---------
10010  2                10111 7(≠-9) ←ошибка    01000 -8(≠8) ←ошибка
C=1 V=0                 C=1 V=1                 C=0 V=1
```

Если стрелка указывает на N, Z, V, C (Могут быть разные комбинации), то нужно найти их значения и заполнить в PS

- Бит признака отрицательного числа (N - Negative)
- Признак того, что аккумулятор содержит 0 (Z — Zero). Данный бит содержит единицу , когда все биты аккумулятора = 0.
- Бит переполнения знаковых чисел (V — oVerflow)
- Бит переноса С беззнаковых чисел (Carry)

Особые команды

- ROL – Циклический сдвиг влево, 15 бит регистра перемещается в С (бит переноса), С (бит переноса) перемещается в 0 бит, каждый бит перемещается на **i + 1** позицию

- ROR - Циклический сдвиг вправо, 0 бит регистра перемещается в С (бит переноса), С (бит переноса) перемещается в 15 бит, каждый бит перемещается на **i - 1** позицию

- ASR - Арифметический сдвиг вправо, 0 бит регистра перемещается в С (бит переноса), С (бит переноса) НИКУДА не перемещается, каждый бит перемещается на **i - 1** позицию; внимание! **AC_{15} -> AC_{15}, AC_{14}** т.к. Сдвиг арифметический, знак меняться не должен. Примеры: ASR(A000)=D000 ~ ASR(1010 0000 0000 0000)= 1101 0000 0000 0000 ASR(4400)=2200 ~ ASR(0100 0100 0000 0000) = 0010 0010 0000 0000

- Extend sign - расширение знака – копирование 7го бита в биты с 15 по 8,
для упрощения, если 7й бит = 0, то значение будет 00XX, если 7й бит = 1, то FFXX, где XX те же цифры(буквы), что были до, то есть если было 008A, то будет FF8A

- SWAB – биты с 7 по 0 позицию меняются местами с битами с 15 по 8 позицию

- HALT – конец выполнения, таблица закончена, ты молодец

84	0000200000	CLC	0 → C
85	80C4101040		GOTO INT @ C4
86	8184011040	CMC	if PS(C) = 1 then GOTO CLC @ 84
87	0000208300		HTOH(~0 + ~0) → C
88	80C4101040		GOTO INT @ C4

- ШТОШ, СМС – это инверсия бита переноса; в 87 микрокоманде (НТОН берет СТАРШИЙ БАЙТ, в данном случае от ~0 +~0, которое при сложении получит перенос) заполняет бит переноса С единицей, подробней про НТОН мне лень расписывать, на экзамен, может, выучите

Вероятность этого стремится к нулю

C2	81C7084002	IO	if CR(11) = 1 then GOTO IRQ @ C7
C3	0400000000	DOIO	IO
C4	80DE801040	INT	if PS(W) = 0 then GOTO STOP @ DE
C5	8001401040		if PS(INT) = 0 then GOTO INFETCH @ 01
C6	0800000000		INTS
C7	0088009208	IRQ	~0 + SP → SP, AR
C8	0001009004		IP → DR
C9	0200000000		DR → MEM(AR)
CA	0088009208		~0 + SP → SP, AR
CB	0001009040		PS → DR
CC	0220001002		LTOL(CR) → BR; DR → MEM(AR)
CD	00A0020020		SHL(BR) → BR, AR
CE	0100000000		MEM(AR) → DR
CF	0004009001		DR → IP
D0	0080001420		LTOL(BR + 1) → AR
D1	0100000000		MEM(AR) → DR
D2	0040009001		DR → PS
D3	8001101040		GOTO INFETCH @ 01

IO – выполнение ввода/вывода значения из Внешнего Устройства

INT – бит прерывания в регистре состояния

INTS – выполнение прерывания

С большой вероятностью не будет, т.к. данных из ВУ у нас нет, и мне немного лень разбираться, что такое SHL и как конкретно работает LTOL

Кажется, рассмотрел всё, что есть в действиях, донаты принимаются, деньги пойдут куда-нибудь