ОПД Лабораторная №3

Сдать 3 лабу довольно просто, особо трудные вопросы не спрашивают, вполне реально сдать за 1 подход. Для начала нужно изучить всю теорию.

1) Адресация

Код 11	10	9	8	Мнемоника	Описание	Реализация машинных циклов Address Fetch, *Operand Fetch*
0	M	M	M	**ADD 0ADDR** **ADD $L**	Прямая абсолютная	*DR → AR; MEM(AR) → DR*
1	0	0	0	**ADD (L)**	Косвенная относительная	**SXT_CR(0..7) → BR,** **BR + IP → AR, MEM(AR) → DR,** *DR → AR; MEM(AR) → DR*
1	0	0	1		Резерв	
1	0	1	0	**ADD (L)+**	Косвенная автоинкрементная (постинкремент)	**SXT_CR(0..7) → BR,** **BR + IP → AR, MEM(AR) → DR, DR + 1 → DR,** **DR → MEM(AR), DR - 1 → DR,** *DR → AR; MEM(AR) → DR*
1	0	1	1	**ADD -(L)**	Косвенная автодекрементная (предекремент)	**SXT_CR(0..7) → BR,** **BR + IP → AR, MEM(AR) → DR, DR - 1 → DR,** **DR → MEM(AR),** *DR → AR; MEM(AR) → DR*
1	1	0	0	**ADD &N** **ADD (SP+N)**	Косвенная относительная, со смещением (SP)	**SXT_CR(0..7) → BR,** **BR + SP → DR,** *DR → AR; MEM(AR) → DR*
1	1	0	1		Резерв	
1	1	1	0	**ADD L** **ADD (IP+N)**	Прямая относительная	**SXT_CR(0..7) → BR,** **BR + IP → DR,** *DR → AR; MEM(AR) → DR*
1	1	1	1	**ADD #N**	Прямая загрузка	**SXT_CR(0..7) → BR, BR → DR**

Рисунок В.10 Циклы выборки адреса и операнда для различных режимов адресации

Для понимания, нужно разобраться в этих фотографиях. На первой фотке показаны виды адресации, рассмотрим относительную.
Коп (код операции) - это зарезервированный символ (например, мы написали команду 6159, здесь коп 6, который означает SUB, таким образом БЭВМ понимает что мы хотим от нее. Если команда 215C, то коп 2 - логическое И)

Все КОП написаны в фото ниже. Там все команды. Необходимо знать все, кроме команд ввода-вывода.
После КОП идет бит, который говорит какая именно у нас адресация. Если 0 - абсолютная, 1 - относительная.

- Абсолютная -> указывает на ячейку, где хранятся необходимое нам значение. (абсолютная адресация -> необходимая нам ячейка)
- Относительная -> указывает на ячейку памяти относительно текущего адреса команды.
    - Прямая -> указывает на непосредственный адрес ячейки (так как мы и привыкли с абсолютной адресацией)
    - Косвенная -> указывает на ячейку, содержимое которой указывает на другую ячейку, в которой и хранится необходимое нам значение. Другими словами, появился некий посредник (косвенная адресация -> ячейка 1 -> необходимая нам ячейка)

Итак, первый 4 бИта - КОП (то, какая команда будет выполняться), 11 - бит отвечает за режим, если 0, то абсолютная прямая, если 1, то смотрим на вторую фотку и смотрим 11-8 бИты. Когда поняли какая у нас адресация, находим необходимую ячейку и уже работаем с ней. Для наглядности разберем пример.

Система команд базовой ЭВМ

Код	Команда	Признаки[1]				Описание
		N	Z	V	C	
0XXX	**Безадресные команды**					
0000	NOP	-	-	-	-	Нет операции
0100	HLT	-	-	-	-	Останов
0200	CLA	*	*	0	-	0 → AC
0280	NOT	*	*	0	-	(^AC) → AC
0300	CLC	-	-	-	0	0 → C
0380	CMC	-	-	-	*	(^C) → C
0400	ROL	*	*	*	*	AC и C сдвигается влево. AC15 → C, C → AC0
0480	ROR	*	*	*	*	AC и C сдвигается вправо. AC0 → C, C → AC15
0500	ASL	*	*	*	*	AC сдвигается влево. AC15 → C, 0 → AC0
0580	ASR	*	*	*	*	AC сдвигается вправо. AC0 → C, AC15 → AC14
0600	SXTB	*	*	0	-	Расширение знака мл. байта AC7 → AC15...AC8
0680	SWAB	*	*	0	-	Обмен ст. и мл. байта AC7...AC0 ↔ AC15...AC8
0700	INC	*	*	*	*	AC + 1 → AC
0740	DEC	*	*	*	*	AC - 1 → AC
0780	NEG	*	*	*	*	^AC + 1 → AC
0800	POP	*	*	0	-	(SP)+ → AC
0900	POPF	*	*	*	*	(SP)+ → PS
0A00	RET	-	-	-	-	(SP)+ → IP
0B00	IRET	*	*	*	*	(SP)+ → PS, (SP)+ → IP
0C00	PUSH	-	-	-	-	AC → -(SP)
0D00	PUSHF	-	-	-	-	PS → -(SP)
0E00	SWAP	*	*	0	-	Обмен A и вершины стека
1XXX	**Команды ввода-вывода**					
10XX	DI	-	-	-	-	Запрет прерывания
11XX	EI	-	-	-	-	Разрешение прерываний
12XX	IN REG	-	-	-	-	Чтение из регистров ВУ
13XX	OUT REG	-	-	-	-	Запись в регистры ВУ
18XX	INT NUM	*	*	*	*	Программное прерывание с заданным вектором
XXXX	**Адресные команды**					
2XXX	AND M	*	*	0	-	M & AC → AC
3XXX	OR M	*	*	0	-	M | AC → AC
4XXX	ADD M	*	*	*	*	M + AC → AC
5XXX	ADC M	*	*	*	*	M + AC + C → AC
6XXX	SUB M	*	*	*	*	AC - M → AC
7XXX	CMP M	*	*	*	*	Установить флаги по результату AC - M
8XXX	LOOP M	-	-	-	-	M - 1 → M; Если M <= 0, то IP + 1 → IP
9XXX						Резерв
AXXX	LD M	*	*	0	-	M → AC
BXXX	SWAM M	*	*	0	-	M ↔ AC
CXXX	JUMP M	-	-	-	-	M → IP
DXXX	CALL M	-	-	-	-	SP - 1 → SP, IP → (SP), M → IP
EXXX	ST M	-	-	-	-	AC → M
FXXX	**Команды ветвления**					
F0XX	BEQ(BZS)	-	-	-	-	Переход если равенство (Z==1)
F1XX	BNE(BZC)	-	-	-	-	Переход если неравенство (Z==0)
F2XX	BMI(BNS)	-	-	-	-	Переход если минус (N==1)
F3XX	BPL(BNC)	-	-	-	-	Переход если плюс (N==0)
F4XX	BHIS(BCS)	-	-	-	-	Переход если выше или равно/перенос (C==1)
F5XX	BLO(BCC)	-	-	-	-	Переход если ниже/нет переноса(C==0)
F6XX	BVS	-	-	-	-	Переход если переполнение (V==1)
F7XX	BVC	-	-	-	-	Переход если нет переполнения (V==0)
F8XX	BLT	-	-	-	-	Переход если меньше (N⊕V==1 / N!=V)
F9XX	BGE	-	-	-	-	Переход если больше или равно (N⊕V==0 / N==V)
FAXX						Резерв

Команда 4EF7

1. 4 -> Вспоминаем все КОП, либо смотрим на фотку выше. Это команда ADD.
2. E = 1110 -> смотрим на фотку с адресацией. Прямая относительная.

3. F7 -> сам адрес. В БЭВМ, но в БЭВМ хранится ведь по 4 байта, а тут только 2. Тогда БЭВМ просто добавляем слева F и получается F7 -> FFF7.
4. Окей, мы распарили всю команду. Теперь, чтобы понять, что делает БЭВМ просто смотрим на вторую фотку в правый столбец и понимаем)
5. Так как адресация относительная, то нужно наш адрес FFF7 прибавить с текущим адресом IP. Мы помним, что после того, как мы начали выполнять текущую команду IP прибавляется 1, и указывает на следующую команду. То есть IP - 1 = адрес текущей команды, которая сейчас выполняется. Это важно помнить, именно поэтому часто пишут не просто IP, а IP + 1.
6. Итак, к IP прибавляем FFF7. Очевидно, что скорее всего произойдет переполнение. Но полученное числа и будет указывать нужный нам адрес.

Со временем, я просто научился в голове просчитывать, сколько нужно прибавить к FFF7, чтобы получилось 0000. Нужно прибавить 9.
А адрес текущей ячейки допустим равен 11A. Мы помним, что IP + 1, значит чтобы найти нужную нам команду, нужно выполнить следующее действие
11A + 1 - 9 = 112. Это и есть наше искомая команда. Но это лишь тот способ, как считаю я, вы можете решать попроще.

Так же важно помнить, что коп + режим отнимают у нас 8 бит. Значит, на сам адрес остается всего 8. Значит максимально смещение, которые мы сможем указать - > 2^8. Это могут спросить на защите.

2) С адресацией разобрались, дальше нужно изучить команды ветвления.

Наименование	Мнемон.	Код	Описание
Переход, если равенство	**BEQ D**	F0XX	IF Z==1 THEN IP+D+1 → IP
Переход, если неравенство	**BNE D**	F1XX	IF Z==0 THEN IP+D+1 → IP
Переход, если минус	**BMI D**	F2XX	IF N==1 THEN IP+D+1 → IP
Переход, если плюс	**BPL D**	F3XX	IF N==0 THEN IP+D+1 → IP
Переход, если выше или равно /перенос	**BCS D** **BHIS D**	F4XX	IF C==1 THEN IP+D+1 → IP
Переход, если ниже/нет переноса	**BCC D** **BLO D**	F5XX	IF C==0 THEN IP+D+1 → IP
Переход, если переполнение	**BVS D**	F6XX	IF V==1 THEN IP+D+1 → IP
Переход, если нет переполнения	**BVC D**	F7XX	IF V==0 THEN IP+D+1 → IP
Переход, если меньше	**BLT D**	F8XX	IF N⊕V==1 THEN IP+D+1 → IP
Переход, если больше или равно	**BGE D**	F9XX	IF N⊕V==0 THEN IP+D+1 → IP
Безусловный переход	**BR D** **JUMP D**	CEXX	IP+D+1 → IP

Их нужно знать все. У некоторых по 2 команды. Но они выполняют одно и тоже.
Перед командами ветвления часто выполняется команда CMP. Она похожа на команду SUB, но единственное отличие, CMP не записывает результат в аккумулятор, но выставляет вспомогательные флаги NZV. А это чё нам как раз необходимо, ведь сравнение происходит именно за счет этих флагов.
Посмотрев на картинку, все становится ясно.

3) Необходимо изучить команду LOOP.

LOOP M	-	-	-	-	M - 1 → M; Если M <= 0, то IP + 1 → IP

Это вкратце, а вот подробнее

Итак, команда LOOP из указанного ячейки вычитает 1 и проверяет >= 0. Если да, то выполняет следующую команду (обычно там Jump или Br), а если нет, то перескакивает на 1 команду и работает дальше.
В целом это вся основная информация.

4) Массивы
Про них важно знать лишь то, что они идут друг за другом и хранят нужную нам информацию. Вот и все. Их мы перебираем с помощью LOOP и косвенной относительной адресации.

Знаю эту информацию, нужна просто практика. Порешайте свой вариант и будете полностью готовы к защите лабы. Сложные вопросы практик не спрашивает, потому что сама говорит, что лаба довольно простая.

Некоторые советы по лабе.

- Можно взять шаблон со второй лабы и заменить содержимое, не меняя при этом архитектуру файла. Потому что шаблон почти один и тот же.
- После того, как заполните первую таблицу и поймете что именно делает ваша программа, добавьте еще один столбик и опишите там именно ход программы.

Например, как сделал я - добавил столбик “описание программы”.

Адрес	Код команды	Мнемоника	Комментарии	Описание программы
58C	059F	A	Значение	Адрес первого элемента массива
58D	0200	B	Значение	Адрес на текущем элементе массива (отсчет идет от последнего элемента к начальному)
58E	E000	C	Значение	Количество элементов в массиве (5)
58F	0200	D	Значение	Результат программы (количество положительных чисел в массиве)
590	0200	CLA	Очистка AC AC=0	
591	EEFD	ST 58F	Прямая относительная ST IP+1-3 = ST 58F Обнуляем переменную D	Обнуляем переменную, которая будет хранить результат программы
592	AF05	LD #05	Прямая загрузка AC=0005	
593	EEFA	ST 58E	Прямая относительная ST IP+1-6 = ST 58E C = AC = 0005	Заполняем переменную, которая отвечает за размер массива
594	4EF7	ADD 58C	Прямая относительная ST IP+1-9 = ST 58C AC = 0005 + A	Складываем адрес первого элемента в массиве и количество элементов этого массива. Таким образом получаем адрес, который на 1 больше от адреса последнего элемента массива

- Добавьте все эти пункты, которые есть у меня и по какой-либо причине нет у вас. Они прописаны в методичке, и если мне не изменяет память, практик заставляла переделывать ребят, у которых не находила у них этих пунктов.

Описание программы

Программа считает количество положительных элементов в массиве.

Расположение в памяти БЭВМ программы, исходных данных и результатов:

58C-58E, 59F-5A3 – исходные данные

590-59E – инструкции

58F – результат

Адреса первой и последней выполняемой инструкции программы:

590 – адрес первой инструкции

59E– адрес последней инструкции

Область представления:

Элементы массива – знаковое 16-разрядное число.

A (адрес первого элемента массива) – беззнаковое 11 разрядное число.

B (адрес текущего элемента массива) – беззнаковое 11 разрядное число.

C (количество элементов в массиве) – знаковое 16-разрядное число.

D (результат программы) – знаковое 16-разрядное число.

Область допустимых значений

-2^{15} <= Элементы массива <= $2^{15} - 1$

0 <= A <= 2042 ($07FF_{16} – 5 = 07FA_{16}$)

5 <= B <= 2047 ($07FF_{16}$)

0 <= C <= 127 (т. к. прямая загрузка)

- ОДЗ пишется довольно просто, обычно ограничение упирается лишь в то, что во всей памяти БЭВМ может хранится всего 7FF значений и в ограничение ввода данных. Например, у косвенной и прямой адресации это от 0 до 127.
- Трассировку можно выполнить только после того, как получите числа от практика. Чтобы быстро сдать лабу, а не вводить каждый раз всю программу в эмулятор, то просто напишите свою программу на ассемблере и будет вам счастье. Вводите один раз в эмулятор и все готово. Пример моего ассемблера

```
ORG 0x58C

A: WORD 0x059F

B: WORD 0x0200

C: WORD 0xE000

D: WORD 0x0200

START: CLA

ST D

LD #05

ST C

ADD A

ST B

TOJUMP: LD -(B)

BMI ITER

        BEQ ITER

LD D

INC

ST D

ITER: LOOP $C

JUMP TOJUMP

HLT

X1: WORD 0xF900

X2: WORD 0xF800
```

X3: WORD 0x0D00

X4: WORD 0x0700

X5: WORD 0xB59B

- Если в ассемблере что-то непонятное, то откройте 51 страницу в методичке. Там будет небольшой гайд.
- Будьте готовы к тому, что практик попросит вас изменить программу. Например, у вас в программе у массива 5 элементов, а она даст всего 3 числа. Это значит, что нужно изменить массив до 3 элементов.
- Если подходите к практику впервый раз, то говорите, что показываете ей **ДОПУСК.** Иначе попадете в просак и она вас начнет троллить.
- Мои вопросы: где хранятся флаги NZVC (в регистре PC и рассказал что еще хранится в этом регистре), что делает коммутатор и как выставляются флаги NZVC, что делает команда SWAB, как устанавливаются флаги C и V.
- Как вы видите вопросы мало относятся к самой лабе. Поэтому перед защитой вспомните все устройство БЭВМ. Но мои вопросы скорее исключение.У других спрашивали иное..

Полезные ссылки:

- Лаба 3 опд статья от старших курсов (там в начале можно посмотреть примеры отчётов), советую разок прочитать, есть некоторые полезности
- Презентация про ветвления и циклы (там все очень доступно объясняется, я даже лекцию посмотрел и все понял)
- Книжка
- Методичка
- Разбор второй лабы, поможет вспомнить БЭВМ, если вы прям все забыли