ВВЕДЕНИЕ

Современный специалист в области создания программного обеспечения для вычислительной техники и автоматизированных систем должен обладать достаточными знаниями по использованию средств вычислительной техники в организации и управлении процессами разработки программного обеспечения. Низкоуровневое программирование позволяет четко усвоить принципы работы вычислительных машин и систем, а также их функциональных блоков, более рационально использовать их вычислительную мощность при разработке конкретного вида программного обеспечения, с учётом его особенностей.

Целью проведения лабораторных и практических работ является изучение студентами организации и принципов функционирования памяти, микропроцессора, организации ввода – вывода, а также приобретение навыков низкоуровневого программирования на языке ассемблера.

Для выполнения работ требуются IBM совместимый персональный компьютер, операционная система Windows XP и выше, программный пакет ассемблера TASM.

ОРГАНИЗАЦИЯ МИКРОСИСТЕМ НА БАЗЕ МИКРОПРОЦЕССОРОВ I8086

1.1. Цель работы

Знакомство с принципами организации микросистем на базе МП i8086/80286 (далее МП86). Изучение архитектуры и программирования в машинных кодах МП86. Отработка навыков работы с турбоотладчиком TD.

1.2. Принципы организации микросистем на базе МП i8086 (К1810ВМ86)

На рис. 1.1 приведена типовая структура микропроцессорных систем и микрокомпьютеров на базе 16-битного микропроцессора i8086 или К1810ВМ86. Микросистема содержит центральный процессор на основе МП i8086, память, подсистему ввода-вывода и логику управления системной шиной, которая преобразует центральную магистраль Адрес/Данные (A/D) МП в раздельные шины адреса и данных. Такую структуру имеют 16-разрядные персональные компьютеры типа IBM PC/XT, Искра 1030, СМ1910, Mazovia CM1914 и др.

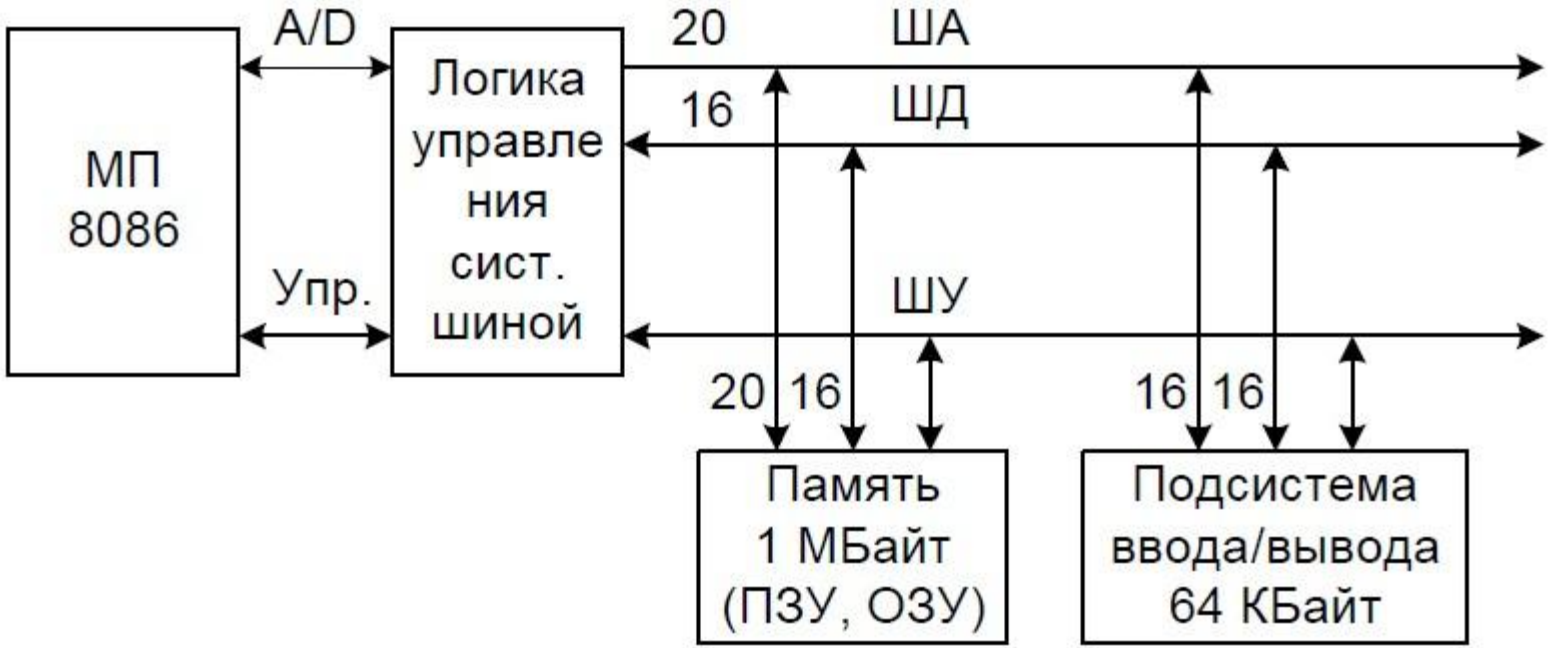

Рис. 1.1. Структура микросистемы на базе МП i8086

1.3. Структурная схема микропроцессора i8086 (рис. 1.2)

В МП i8086 основные этапы выполнения команды распределены внутри МП между двух сравнительно независимых устройств (рис. 1.2): между операционным (ОУ) и устройством сопряжения (УС).

ОУ содержит 16-битные регистры данных AX, BX, CX, DX, указатели памяти SP, BP, SI, DI, арифметико-логическое устройство АЛУ и регистр признаков F. Когда ОУ занято выполнением текущей команды, устройство сопряжения УС осуществляет опережающую выборку из памяти очередных команд. Команды хранятся во внутренней регистровой памяти, называемой очередью (буфером) команд. Очередь команд по существу выполняет функцию регистра команд процессора. Длина очереди составляет 6 байт.

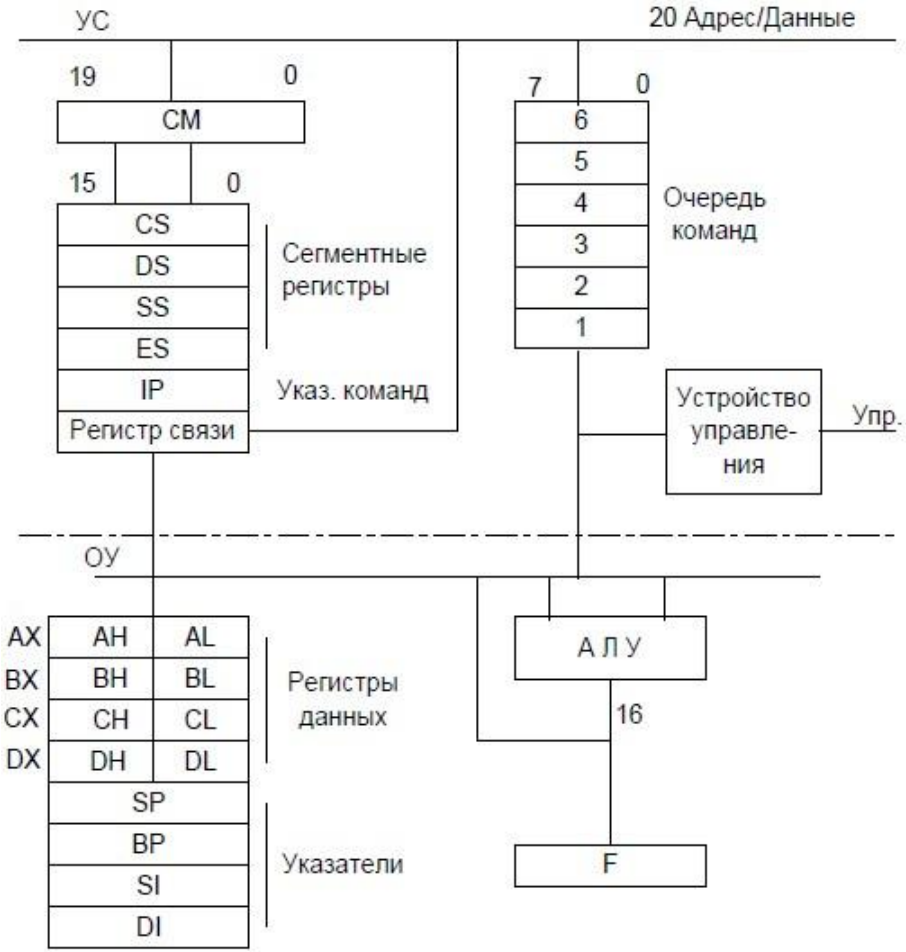

Рис. 1.2. Структурная схема МП i8086

В состав УС входят 16-битные сегментные регистры CS, DS, SS и ES и сумматор СМ, который формирует 20-битный физический адрес сегмента (базы) и смещения, называемого также эффективным (исполнительным) адресом ЕА. Это делается путём суммирования ЕА с содержимым сегментного регистра, сдвинутого относительно ЕА на 4 бита, как показано на рис. 1.3.

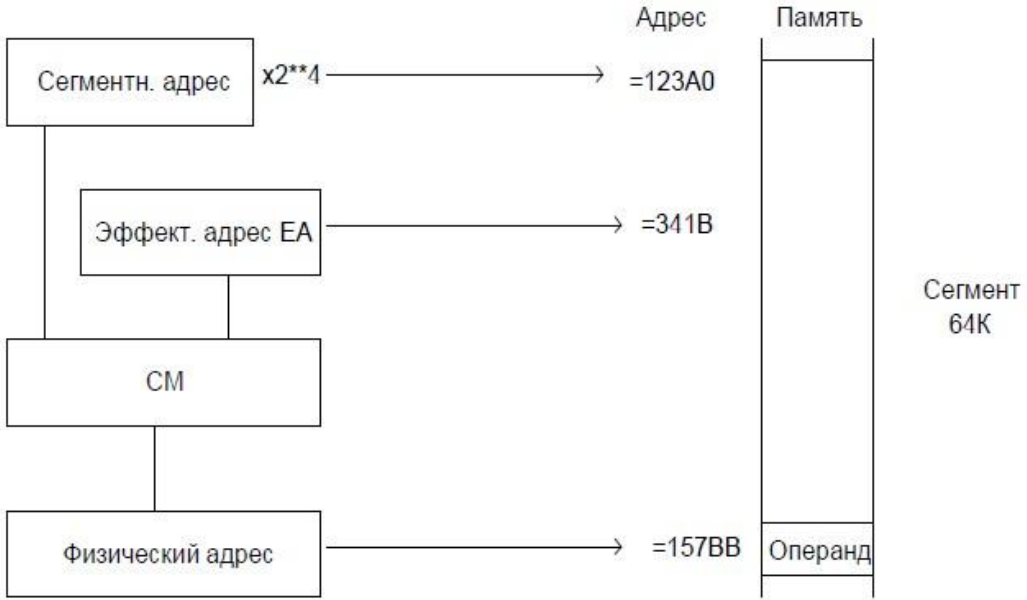

Рис. 1.3. Схема вычисления физического адреса

Если, например, содержимое сегментного регистра данных DS=123A, а указатель памяти SI=341B, то физический адрес операнда в памяти будет равен 157BB.

Пример 1.1

+	1	2	3	A	0	– начальный адрес сегмента данных $DS \cdot 2^4$
		3	4	1	B	– эффективный адрес в SI
	1	5	7	B	B	– физический адрес операнда в памяти

1.4. Программная модель МП (рис. 1.4.)

В программной модели МП (рис. 1.4) можно выделить следующие 4 группы регистров.

1. Регистры данных

В зависимости от того, чем оперирует команда – словами или байтами, регистры данных можно рассматривать как четыре 16-битных (AX, CX, BX, DX) или как восемь 8-битных регистров (AL, AH, CL, CH, BL, BH, DL, DH). Символы L и H означают младшие и старшие байты 16-битных регистров. Каждый из этих регистров помимо общих выполняет и специализированные функции: AX, AL – аккумулятор; BX – базовый регистр (для косвенной адресации памяти); CX – счетчик в операциях с цепочками; DX – регистр данных или указатель при косвенной адресации регистров (портов) ввода/вывода.

Рис. 1.4. Программная модель МП i8086

2. Регистры сегментов CS, DS, SS, ES

Как уже отмечалось ранее, память микросистемы на базе МП86 содержит сегменты памяти по 64 Кбайт. МП может иметь дело одновременно с четырьмя типами сегментов: кода (команд), стека, данных и дополнительного сегмента данных.

Сегментные регистры выполняют следующие функции.

Регистр сегмента команд CS указывает на сегмент, содержащий текущую выполняемую программу. Для вычисления адреса следующей (с учётом очереди команд) исполняемой команды к содержимому CS, умноженному на 2^4, добавляется содержимое указателя команд IP.

Регистр сегмента стека SS указывает на текущий сегмент стека.

Регистр сегмента данных DS указывает на текущий сегмент данных, обычно содержащий используемые программой данные.

Регистр дополнительного сегмента ES указывает на текущий дополнительный сегмент, который используется для выполнения операций над строками.

3. Регистры указателей SP, BP и индексов DI, SI

Регистры предназначены для хранения внутрисегментных смещений и обеспечивают косвенную адресацию данных в пределах текущего сегмента. Эти же регистры могут участвовать в выполнении арифметических и логических операций над двухбайтными данными, т.е. используются при этом как регистры данных. Поэтому данную группу регистров и регистры

данных относят к регистрам общего назначения (РОН). Указатели стека SP и базы BP предназначены для доступа к данным в текущем сегменте стека. Индексные регистры DI (приёмника), SI (источника), а также регистр BX содержат смещение, которое по умолчанию относятся к сегменту данных (DS). В операциях с цепочками данных указатель DI по умолчанию относится к дополнительному сегменту (ES). Указатель команд IP адресует следующую команду программы (разумеется, с учётом очереди команд) в сегменте кодов (CS). Этот регистр ведёт себя как программный счётчик PC в 8-битном МП К580ВМ80 (i8080).

4. Регистр признаков F

Формат 16-битного регистра признаков F показан на рис. 1.5.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	OF	DF	IF	TF	SF	ZF	X	AF	X	PF	X	CF

Признаки (флажки):

OF – переполнение,
DF, IF, TF – признаки управления МП,
SF – знак, ZF – нуль,
AF – вспомогательный перенос,
PF – четность, CF – перенос.

Рис. 1.5. Формат регистра признаков

Младший байт регистра полностью соответствует регистру признаков МП К580ВМ80. Кроме того, в старшем байте F содержится признак арифметического переполнения OF. В дополнение к этому в регистре признаков F фиксируются некоторые признаки, предназначенные для управления работой МП. DF – признак направления, указывающий на то, что обрабатываются цепочки: либо от меньших адресов к большим (DF=0), либо наоборот (DF=1). IF – признак прерывания: при IF=1 прерывание разрешено. TF – признак трассировки, разрешающий при TF=1 выполнять программу по командам (пошаговый режим).

1.5 Адресная организация памяти, представление данных

МП i8086 обеспечивает адресацию памяти ёмкостью 2^{20}=1 Мбайт. На программном уровне память представляют как линейную последовательность из 2^{20} байт =1 Мбайт (рис. 1.6).

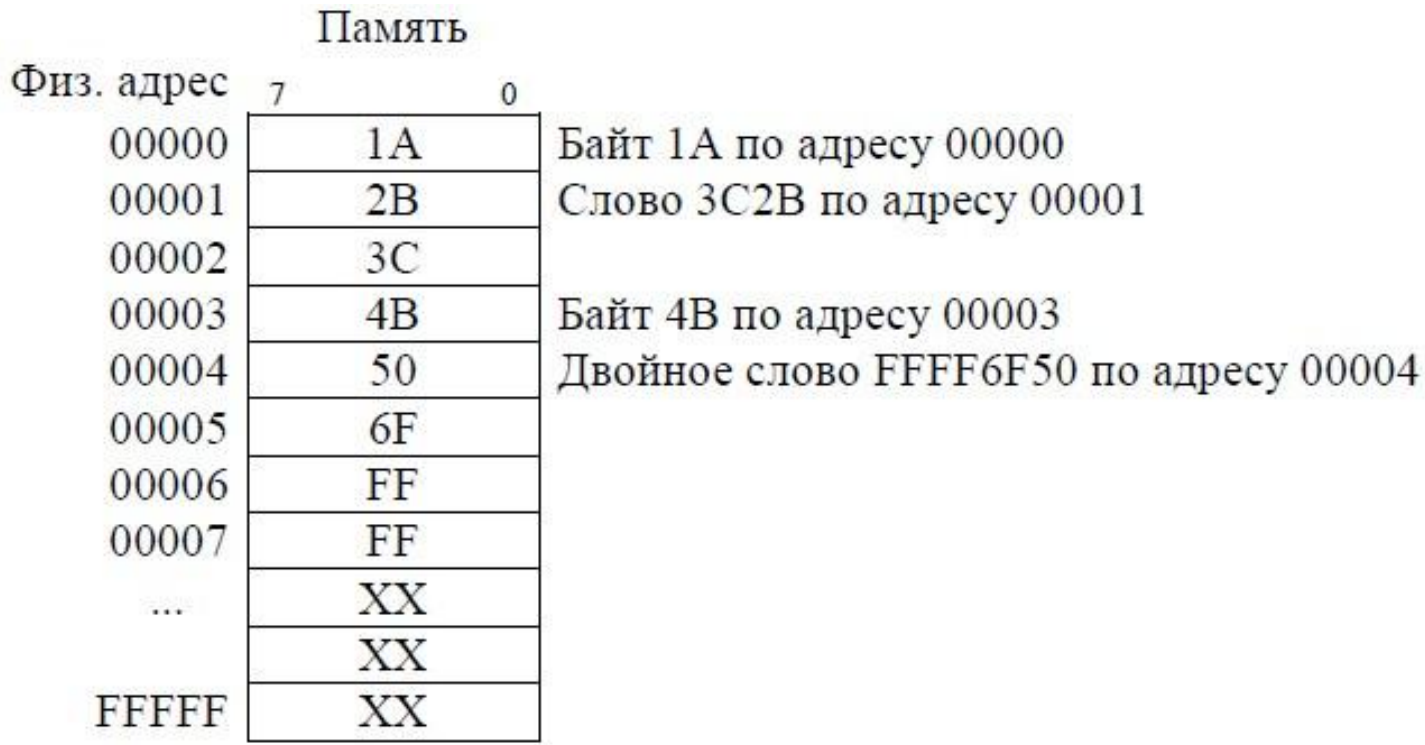

Рис. 1.6. Адресная организация памяти

Два смежных байта образуют слово. Слово может храниться по четному или нечетному адресу. В первом случае слово передается за один цикл шины МП, а во втором – за два.

Следовательно, для достижения наивысшей производительности МП необходимо слова размещать по четным адресам памяти. При этом адресом слова считается адрес его младшего байта, а старший байт размещается по более старшему адресу. Принцип “Младшее по младшему адресу” сохраняется и для представления других единиц данных: многобайтных команд, двойных слов и т.д.

Отметим, что выравнивание команд по четным адресам практически не влияет на производительность МП, т. к. он выбирает их в очередь команд с опережением. Двойное слова содержит 4 байта, а квадрослово – 8.

1.6. Примеры форматов команд МП i8086

Примеры форматов команд с регистровой (1, 5, 6) , непосредственной (2, 7, 8) и прямой (3, 4) адресацией приведены на рис. 1.7.

В обобщенном представлении команд, например ADD reg1,reg2, после мнемоники команды указывается приемник, а затем через запятую – источник операнда. В байтах data L, data H команды указываются младшая и старшая части (если W=1) константы соответственно.

Во всех командах содержится однобитное поле W: если W=1, то команда оперирует словом, если W=0, то – байтом.

В поле reg (или reg1, reg2, r/m) команд указываются трехбитные адреса РОНов МП. В табл. 1.1 приведены адреса регистров МП для различных значений W.

1. Передача регистр – регистр MOV reg1,reg2

1-й байт	2-й байт	3-й байт
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	
1 0 0 0 1 0 1 w	1 1 reg1 reg2	

2. Передача константы в регистр MOV reg,data

1 0 1 1 w reg	Data L	Data H	,если w=1

3. Передача из памяти в аккумулятор MOV acc,[EA]

1 0 1 0 0 0 0 w	Мл. часть EA	Ст. часть EA

4. Передача из аккумулятора в память MOV [EA],acc

1 0 1 0 0 0 1 w	Мл. часть EA	Ст. часть EA

5. Сложить регистр – регистр ADD reg1,reg2

0 0 0 0 0 0 1 w	1 1 reg1 reg2

6. Вычесть регистр – регистр SUB reg1,reg2

0 0 1 0 1 0 1 w	1 1 reg1 reg2

7. Сложить константу с аккумулятором ADD acc,data

0 0 0 0 0 1 0 w	Data L	Data H	,если w=1

8. Вычесть константу из аккумулятора SUB acc,data

0 0 1 0 1 1 0 w	Data L	Data H	,если w=1

Рис 1.7. Примеры форматов команд с регистровой, непосредственной и прямой адресацией

Таблица 1.1. Адреса регистров МП

Адрес регистра reg, r/m	Регистр	
	W=1	W=0
000	AX	AL
001	CX	CL
010	DX	DL
011	BX	BL
100	SP	AH
101	BP	CH
110	SI	DH
111	DI	BH

Рассмотрим примеры кодирования команд.

Пример 1.2. Команды передачи

Код команды	Мнемоника	Операция
10001011 11000011	MOV AX,BX	AX←BX
10001010 11100111	MOV AH,BH	AH←BH
BA A7 02	MOV DX,02A7H	DX←02A7H
A1 27 10	MOV AX,[1027H]	AX← [1027H]

1.7. Пример разработки программы в машинных кодах

При программировании на машинном языке адреса, коды команд и данные записываются в 16-ричной системе счисления.

Разработаем программу для вычисления выражения

$$M=K+N-R+120, \qquad (1.1)$$

в котором все операнды и результат – двухбайтные данные. K, R, M размещены в сегменте данных, а N – в регистре DX. Числа со знаком представлены в дополнительном коде (ДК). Например, ДК чисел K=+60, R=-10, N=+30 соответственно равны $[K]_{ДК}$=0060, $[R]_{ДК}$=FFF0, $[N]_{ДК}$=0030. Результат M=+1C0, $[M]_{ДК}$=01C0.

Правило образования ДК.

ДК положительного числа есть само число, представленное в формате байта или слова. ДК отрицательного числа есть инверсия (not) числа плюс единица, например в формате слова $[-30]_{ДК}$=not(0030)+1=FFCF+1=FFD0. Инверсия 16-ричной цифры равна: not(α)=F – α.

В табл. 1.2 приведена программа вычисления выражения (1.1). Начальный эффективный адрес программы равен 100. Этому адресу соответствует физический адрес CS·2^4+100. Далее его будем обозначать как CS:0100. Эффективные адреса данных лежат в диапазоне 0500-0505, а физические – в диапазоне DS:0500 - DS:0505.

Таблица 1.2. Программа вычисления выражения M=K+N-R+120

Адрес	Код	Мнемоника	Операция
CS:0100	A1	MOV AX,[0502H]	AX←R
0101	02		
0102	05		
0103	2B	SUB DX,AX	DX←N-R
0104	D0		
0105	A1	MOV AX,[0500H]	AX←K
0106	00		
0107	05		
0108	03	ADD AX,DX	AX←K+N-R
0109	C2		
010A	05	ADD AX,0120H	AX←AX+120
010B	20		
010C	01		
010D	A3	MOV [0504H],AX	[0504]←M
010E	04		
010F	05		
0110	90	NOP	Пустая операция
Адреса данных	Данные в ДК		
DS:0500	60	K=0060	Операнд K=+60
0501	00		
0502	F0	R=-0010	Операнд R=-10
0503	FF		
0504	XX	M=XXXX	Результат M
0505	XX		
РОН	Данные		
DX	0030	N=0030	Операнд N=+30

1.8. Отладчик TD

Отладчик TD – это системная программа, находящаяся на диске ПЭВМ. Отладчик позволяет управлять процессом исполнения пользовательской программы. Команды TD, вводимые с клавиатуры ПЭВМ, позволяют выводить на экран и изменять содержимое памяти и регистров МП, исполнять программу по шагам (командам) и др. Команды отладчика будут рассмотрены при выполнении работ.

1.9. Варианты заданий

Таблица 1.3. Варианты заданий

№ варианта задания	Размещение операндов				Длина операнда	Нач. адр. прогр.
	M	K	R	N		
1.M=K+R+2N-3	Память			BH	Байт	100
2.M=K-R-N+50	Память		DX	Пам.	Слово	100
3.M=2K-N+R-6	Пам.	CL	Память		Байт	100
4.M=K+3N-R+20	Память			DX	Слово	100
5.M=K-R-N+8	Память		CL	Пам.	Байт	100
6.M=2K-R+N+18	Пам.	DX	Память		Слово	100
7.M=2K-2R+N-10	Память		DL	Пам.	Байт	100
8.M=120-K-R+N	Пам.	CX	Память		Слово	100
9.M=20+K-2R+N	Память			CH	Байт	100
10.M=500+K-2R+N	Пам.	BX	Память		Слово	100

1.10. Порядок выполнения работы

Ниже изучаются команды программы Norton Commander и отладчика TD персональной ЭВМ, с помощью которых осуществляются ввод и отладка программ в машинных кодах микропроцессора МП86. В качестве примера используется программа вычисления выражения M=K-R+N+120, приведенная в табл. 1.2.

Создание исходного файла программы

1. Для создания в системе Norton Commander (NC) нового файла программы последовательно выполните пп. 2-4, при корректировке файла – пп. 5-7. Вызов системы производится командой nc.
2. При нажатой клавише Shift нажмите клавишу F4 – программа NC выдаст запрос. Введите в нем имя файла и его расширение txt и нажмите Enter (↵). Например, petrov.txt↵. Еще раз нажмите Enter и попадете в NC.
3. С помощью команд встроенного редактора NC наберите программу (табл. 1.2) в машинных кодах:

```
a1 0205
2bd0
a1 0005
03c2
05 2001
a3 0405
90
```

4. Чтобы сохранить созданный файл на диске, нажмите F2, а затем F10 – вы выйдете из редактора.

5. Если потребуется скорректировать или дополнить известный файл, то в панели системы NC курсором выделите имя этого файла и нажмите F4 – вы попадете в редактор с вызванным файлом.

6. С помощью команд редактора NC скорректируйте исходный файл. Например, перед машинной командой 90 введите еще несколько кодов 90 (пустая операция NOP).

7. Чтобы сохранить созданный файл на диске, нажмите F2, а затем F10 – вы выйдите из редактора.

Выделенный в панели NC файл можно распечатать на принтере, если нажать F5 – копировать, а затем при ответе на запрос ввести PRN (принтер) и нажать ↵.

Преобразование исходного модуля программы в машинных кодах в исполняемый модуль типа COM

8. С помощью команды

trans <имя файла>.txt↵

запустите программу трансляции TRANS, которая преобразует коды команд МП86, представленные в виде ASCII-символов, в последовательность двоичных кодов команд, образующих COM-файл программы.

Выполнение и отладка исполняемого модуля программы с помощью турбоотладчика TD

9. С помощью команды

td <имя файла>.com↵

запустите отладчик TD для работы с созданным COM-файлом программы. Отладчик загрузит в память исполняемый модуль типа COM с адреса 100h, причем коды программы и данных разместятся в одном сегменте кода емкостью 64Кбайт. После загрузки отладчик выдаст на экран монитора окно процессора CPU.

10. Нажав ENTER, снимите марку отладчика. На экране изобразится окно отладчика CPU, состоящее из 5 подокон: кодового сегмента, содержащего коды программы; регистров микропроцессора; регистров флажков; сегмента стека и сегмента данных. Клавишей F5 увеличите/уменьшите окно CPU. Двойной рамкой и находящимся в подокне маркером выделено активное подокно (или окно). Переход из одного подокна в другое производится нажатием клавиши Tab или Shift-Tab. Можно перейти в подокно, нажимая Shift и одну из клавиш перемещения курсора.

Находясь в подокне, можно, нажав Alt-F10, войти в локальное подменю и с помощью его команд изменить содержимое регистров памяти. Нажав F10, можно войти в главное меню отладчика и воспользоваться его командами для управления выполнением программы. Выход из меню производится клавишей Esc.

Ниже рассматриваются основные команды отладчика для отладки программ с помощью окна процессора CPU.

11. Нажав клавишу Tab, перейдите в подокно регистров. Подведите маркер к регистру DX, введите код 30 и нажмите ENTER . Тем самым вы ввели в DX операнд N=0030h. (При вводе 16-ричных кодов необходимо, чтобы 1-й символ начинался с цифры, например 0FFFF.)

12. Перейдите в подокно сегмента данных. Нажав Alt-F10, вызовите для данного подокна локальное меню. Выберите в нем команду GOTO и нажмите ENTER. Далее введите с клавиатуры начальный адрес данных ds:500↵. Убедитесь в том, что маркер выделяет в подокне памяти байт с адресом 500, а затем введите с клавиатуры коды 60 00 0F0 0FF↵. Тем самым вы ввели в память операнды K=+60 и R=-10.

Можно вводить данные в память в формате слова. Для этого с помощью команды Ctrl-D войдите в подменю локального меню и задайте в нем формат WORD – слово. Затем введите с клавиатуры необходимый операнд, например 60 по адресу 500.

Отметим, что Ctrl-D – это активная клавиша команд Alt-F10; Display as. Далее при работе с окнами отладчика будем в основном использовать активные клавиши, нажимая Ctrl в сочетании с первой буквой команды локального меню.

13. Перейдите в подокно кода. В нем изображаются дизассемблированные команды выполняемой программы, причем текущая команда помечается стрелкой. Для управления выполнением программы используют следующие основные команды, вызываемые активными клавишами:

F7	Выполнение одной команды
F8	Выполнение одной команды с пропуском вызовов
F9	Запуск программы в автоматическом режиме
F4	Выполнение команд до точки останова
Ctrl-F2	Установка программы в исходное состояние
F2	Установка/отмена точки останова

Нажимая последовательно F7, выполните программу по шагам до команды NOP. При этом следите за содержимым регистров МП и памяти, убедитесь в правильности полученного результата.

14. (Данный пункт выполнять необязательно). Нажав Ctrl-F2, установите программу в исходное состояние. Обратите внимание, что указатель отмечает первую команду программы. Перезагрузите исходные данные (см. пп. 11-12). С помощью маркера выделите 3-ю команду программы и нажмите F2 – таким образом вы задали точку останова. Аналогичным образом установите точку останова на команде NOP. Для выполнения программы до 1-й точки останова нажмите F9 (или F4). Далее выполните программу по шагам, последовательно нажимая F7.

15. Нажав Ctrl-F2, установите программу снова в исходное состояние. Перезагрузите исходные данные (см. пп. 11-12). С помощью маркера и команды F2 отметьте точки останова. Маркером выделите команду программы ADD AX,0120 и замените ее с клавиатуры на команду ADD AX,0420 и нажмите ENTER. Тем самым вы установили так называемую “заплату”. Выполните модифицированную программу в пошаговом режиме.

Работа с главным меню и окнами отладчика

Ранее были рассмотрены основные команды отладчика, вызываемые активными клавишами. В общем случае управление отладкой можно производить через главное меню отладчика. Это меню содержит позиции, указанные в верхней части экрана: File (файлы), View (просмотр), Run (выполнение) и другие. Вход в главное меню производится с помощью клавиши F10 и выбора необходимой позиции. Вызов новых окон, например памяти (DUMP) или регистров (REGISTERS), производится через позицию View.

16. Выведите на экран окно памяти, для чего выполните команду F10; View; Dump; ENTER. Работа в этом окне аналогична подокну данных окна CPU.

17. Нажав Ctrl-F5, установите режим перемещения и изменения размеров текущего окна. Нажимая клавиши перемещения курсора, переместите окно. Этими же клавишами можно изменить размеры окна, если держать нажатой клавишу Shift. Завершается этот режим нажатием клавиши ENTER.

18. Командой F10; View; Registers на экран окно регистров Registers.

19. Нажимая последовательно F6, можно перейти из одного окна в другое. Двойной рамкой или находящимся в ней маркером выделяется активное окно. Командой Alt-F3 текущее окно удаляется. Ошибочно удаленное окно можно восстановить командой Alt-F6. Удалите все окна, кроме окна CPU.

20. Для выхода из отладчика нажмите Alt-X.

21. Разработайте для заданного варианта программу вычисления выражения в машинных кодах МП86 и по аналогии с пп. 1-20 введите ее в ЭВМ и выполните. Исходные данные и результат представьте в ДК.

1.11. Содержание отчета

1. Программа вычисления заданного выражения с представлением исходных данных в ДК. Распределение ячеек памяти для данных и программы.

2. Трасса отлаженной программы.