Оглавление

Введение

Дисциплина «Электронные вычислительные машины и периферийные устройства» относится к базовой части профессионального цикла дисциплин основной образовательной программы по направлениям подготовки 09.03.01 Информатика и вычислительная техника, профиль «Системотехника и автоматизация проектирования и управления в строительстве», 09.03.02 Информационные системы и технологии, профиль «Системотехника и информационные технологии управления в строительстве». Дисциплина является обязательной к изучению.

В ходе изучения дисциплины, согласно требованиям ФГОС, у обучающихся должны быть сформированы три основные компетенции (табл. 1), отражающие знания, навыки и умения в части теоретического и практического освоения изучаемого материала.

Данное пособие содержит материал для проведения практических занятий по темам, связанным с изучением основ построения современных ЭВМ, а также с применением современных средств повышения производительности ЭВМ.

Таблица 1

Перечень планируемых результатов обучения, соотнесенных с планируемыми результатами освоения образовательной программы

Компетенция по ФГОС	Код компетенции по ФГОС	Основные показатели освоения (показатели достижения результата)	Код показателя освоения
Обладает способностью выбирать и оценивать способ реализации информационных систем и устройств (программно, аппаратно или программно-аппаратно) для решения поставленной задачи	ОПК-6	Знает способы реализации современных информационных систем и устройств	З1
		Умеет выбирать способы реализации информационных систем и устройств для решения поставленной задачи	У1
		Имеет навыки оценивать способ реализации информационных систем и устройств для решения поставленной задачи	Н1
Обладает способностью сопрягать аппаратные и программные средства в составе информационных и автоматизированных систем	ПК-5	Умеет определять совместимость аппаратных и программных средств в составе информационных и автоматизированных систем и при необходимости устранять возникающие коллизии	У2
Обладает способностью поддерживать работоспособность информационных систем и технологий в заданных функциональных характеристиках и соответствии критериям качества	ПК-30	Знает структуру информационных систем, принципы их организации и взаимодействия, методы и средства поддерживать работоспособность информационных систем и технологий в заданных функциональных характеристиках и соответствии критериям	З3
		Умеет разрабатывать процедуры поддержки работоспособности информационных систем, применять современные технологические методики организации функционирования информационных систем	У3
		Имеет навыки использования инструментальных и программных средств организации функционирования информационных систем	Н3

ОБЩИЕ ПОЛОЖЕНИЯ

Основные принципы *электронных вычислительных машин* (ЭВМ) были заложены в середине прошлого века, несмотря на то, что теория и практика ЭВМ динамично менялись, некоторые положения сохранили свою актуальность и в настоящее время, поэтому изучение материала следует начинать именно с основополагающих принципов: принципа программного управления, принципа адресности и т.д. В данном пособии эта тематика рассмотрена как с теоретической точки зрения, так и с практической.

Практическая сторона состоит в следующем: современные программные средства позволяют эмулировать на современных ЭВМ (в том числе на ЭВМ, установленных в учебном классе) работу классического микропроцессора i86 (отечественный вариант — К1810). Используя программное средство «отладчик» и команды ассемблера, обучающиеся в пошаговом режиме изучают принцип программного управления, принцип однородности памяти и др. При этом обучающиеся видят конкретные значения машинного кода, размер команды в байтах, изменения от шага к шагу счетчика команд, значения ячеек памяти и т.д. Также в пошаговом режиме студенты изучают механизм взаимодействия процессора с оперативной памятью.

Затем теоретически и практически в пошаговом режиме изучается работа регистра признаков (флагов) процессора, аппаратная реализация механизма условных переходов и циклов.

В работе сформулированы направления повышения эффективности работы ЭВМ: использование различных вариантов распараллеливания, сокращение времени «доставки» необходимой информации вычислительным ядрам, оптимизация алгоритмов обработки информации и др.

Далее обучающиеся изучают ряд конкретных методов повышения эффективности работы: конвейер команд, стек, очередь, механизм прерывания, механизм прямого доступа к памяти, пакетная передача информации и др. При практических упражнениях используются табличный процессор и отладчик программ, написанных на языке ассемблер.

В разделе «Теоретический и практический материал для выполнения курсовой работы» дан материал, важный как с точки зрения выполнения курсовой работы, так и с точки зрения лучшего понимания лекционного материала. Рассмотрены две важнейшие тенденции:

1) тенденция перехода от синхронных параллельных шин к высокочастотным последовательным. В рамках этого вопроса изучается кардинальное изменение архитектуры ЭВМ — переход от архитектуры «северный мост — южный мост», которая была заложена в основу ЭВМ на протяжении многих лет, к современной архитектуре «ГиперТранспорт»;

2) тенденция к повышению надежности передачи по интерфейсу. Наиболее перспективные интерфейсы периферийных устройств содержат инструменты повышения надежности, в частности обнаружение и коррекция кода с помощью специальных кодов и контрольных сумм. Таким, в частности, является интерфейс CAN, который можно успешно использовать в области строительства. Дается теория и практические задания для наиболее востребованного метода коррекции — метода кодов Хемминга.

ФОРМА ВЫПОЛНЕНИЯ, ПРЕДСТАВЛЕНИЕ РЕЗУЛЬТАТА, УСЛОВИЯ, ПРИ КОТОРЫХ РАБОТА СЧИТАЕТСЯ ПРИНЯТОЙ

Работы выполняются по индивидуальному заданию, на отдельном ПК.

Процесс сдачи работы обучающимися состоит из следующих этапов:

1) выполнить работу на компьютере. При возникновении вопросов обратиться к преподавателю;

2) после получения необходимого результата сдать задание на компьютере, отвечая на вопросы преподавателя;

3) получив положительную оценку на своем рабочем месте, студент должен перейти к ответам на вопросы в письменном виде у стола преподавателя. Преподаватель может опрашивать группу из 3–4 человек. Положительная оценка означает факт сдачи работы, который преподаватель фиксирует в своем журнале.

ПРАКТИЧЕСКИЕ РАБОТЫ

Практическая работа 1
ИЗУЧЕНИЕ ОСНОВОПОЛАГАЮЩИХ ПРИНЦИПОВ ПОСТРОЕНИЯ ЭВМ

Принцип однородности памяти. Команды и данные хранятся в одной и той же памяти и внешне в памяти неразличимы. Распознать их можно только по способу использования: т.е. одно и то же значение в ячейке памяти может использоваться и как данные, и как команда, и как адрес в зависимости лишь от способа обращения к нему. Это позволяет производить над командами те же операции, что и над числами.

Принцип адресности. Структурно основная память состоит из пронумерованных ячеек, причем процессору в произвольный момент доступна любая ячейка. Двоичные коды команд и данных разделяются на единицы информации, называемые словами, и хранятся в ячейках памяти, а для доступа к ним используются номера соответствующих ячеек — адреса.

Принцип программного управления. Все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последовательности управляющих слов — команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, т.е. в порядке их положения в программе. При выполнении каждой команды счетчик команд увеличивается на количество байт, содержащихся в команде.

Изучение проводится на основе схемы *микропроцессора* (МП) (рис. 1), собранного по архитектуре i86 (по такой архитектуре собран отечественный микропроцессор К1810). Данная архитектура является «классической» в области ЭВМ, в ней заложены многие концепции и принципы, которые в видоизмененном виде используются в современных микропроцессорах. Работа этой схемы может быть эмулирована на современных компьютерах, в частности компьютерах, установленных в компьютерных классах НИУ МГСУ

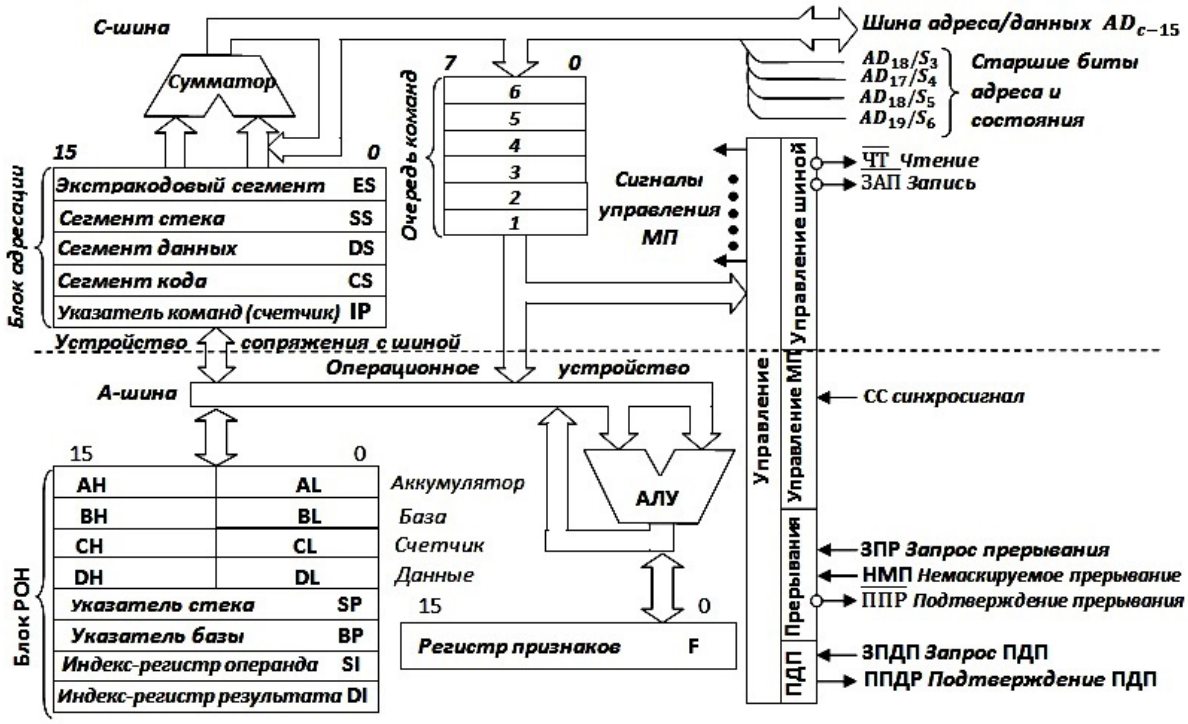

Рис. 1. Схема микропроцессора

Схему МП можно условно поделить на две части — *устройство сопряжения с шиной* (УСШ) (см. рис. 1 выше пунктирной линии) и операционное устройство (ОУ) (см. рис. 1 ниже пунктирной линии). Эти части работают в конвейерном режиме. В УСШ входит дву-

направленная шина адреса и данных (работает в мультиплексном режиме: вначале подаются 16 бит адреса, затем по этим же шинам передаются 16 данных. Для передачи четырех старших бит данных (всего 20 шин адреса и 16 шин данных) служат отдельные шины. Кроме того, в УСШ входят сегментные регистры, очередь команд и верхняя часть устройства управления (отвечает за чтение и запись).

В ОУ входят: двухбайтные регистры общего назначения — AX, BX, CX, DX, SP, BP, SI, DI, причем регистры AX, BX, CX, DX поделены пополам — верхняя H (high) и нижняя L (low); *арифметически-логическое устройство* (АЛУ); регистр признаков (флагов) F; нижняя часть устройства управления (отвечает за синхронизацию, прерывания и прямой доступ к памяти).

Работа схемы и изучение принципа программного управления производится с помощью отладчика AFD PRO.

Интерфейс имеет вид, как показано на рис. 2.

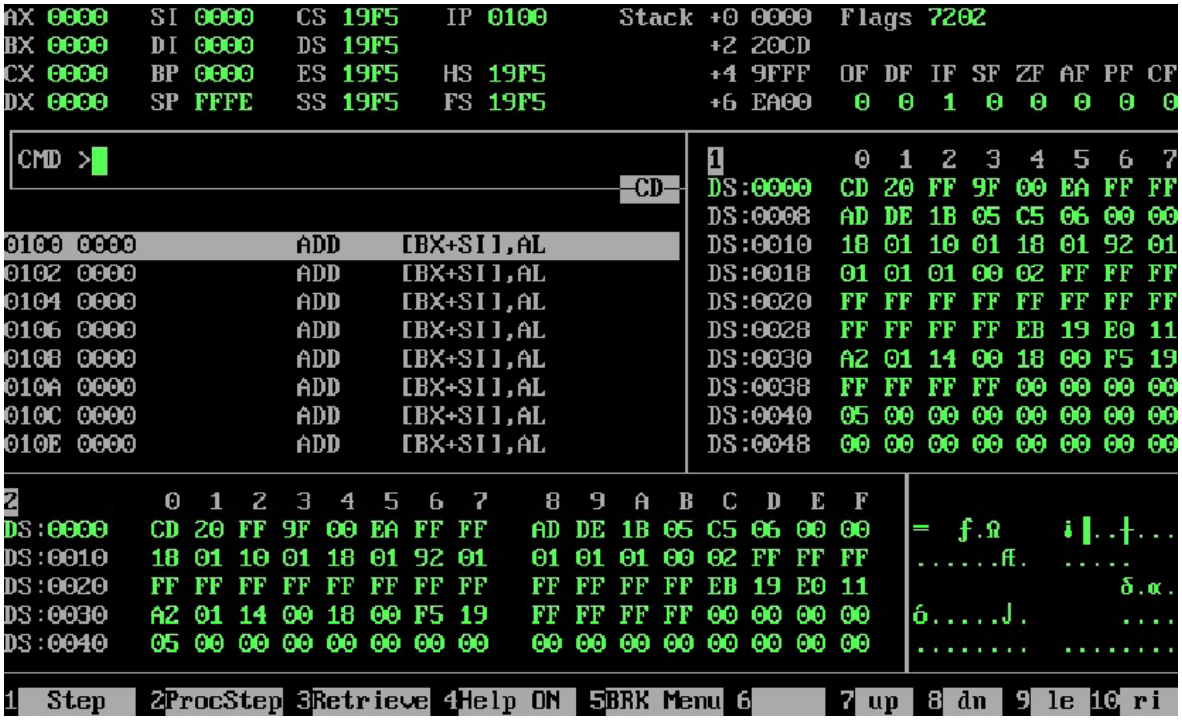

Рис. 2. Интерфейс отладчика команд

В рамке с надписью CMD> находится командная строка, выше — значения регистров процессора общего назначения AX, BX, CX, DX,SI, DI, BP, SP, правее их — сегментные регистры CS, DS, ES, SS, а еще правее — счетчик команд IP (HS и DF в работе не используются). В правом верхнем углу — 4 ячейки стека (двухбайтные) и регистр признаков (флагов), состоящий из флагов OF DF…CF. Слева и справа от командной строки расположены ячейки памяти.

При изучении принципов работы ЭВМ используется программирование на низком уровне — на языке ассемблер. Знание этого языка является очень полезным. Например, оно используется при программировании микроконтроллеров, которые в последнее время исключительно широко используются в цифровых технологиях вообще и в строительной отрасли в частности. Данный язык много лет преподается на факультете вычислительной математики и кибернетики МГУ.

Основные операторы языка ассемблер

Безоперандные команды:

STI — разрешить прерывания;

CLI — запретить прерывания;

NOP — задержка.

Арифметические однооперандные команды:

Inc CL — увеличить на 1 содержимое CL;

Dec DL — уменьшить на 1 содержимое DL.

Арифметические двухоперандные команды:

Mov CL,CH — переместить содержимое из регистра CH в регистр CL.

Отметим здесь три момента:

1) рассматривается регистровый способ адресации;

2) результат сохраняется в *левом* операнде (есть редкие исключения — команда xchg);

3) оба регистра должны быть *одной размерности* — 8- или 16-битные;

ADD CL,BH — сложить содержимое регистров CL и BH, результат поместить в регистр CL, команды ADD CL,BX или ADD CX,BH *не пройдут*, так как разные размеры регистров);

SUB CL,BH — вычесть из содержимого CL содержимое BH;

CMP CH,DL — сравнить содержимое регистров CH и DL;

XCHG CH,DL — поменять местами содержимое регистров CH и DL.

Команды сдвига:

SHL CL,i — сдвинуть влево содержимое CL на i разрядов, равносильно *умножению* содержимого CL на 2^i;

SHR DL,i — сдвинуть вправо содержимое DL на i разрядов, равносильно *делению* содержимого DL на 2^i.

Команды циклического сдвига («выдвинутый» бит не исчезает, записывается с противоположной стороны, умножения или деления не происходит!):

ROL CL,i — циклически сдвинуть влево содержимое CL на i разрядов;

ROR DL,i — циклически сдвинуть вправо содержимое DL на i разрядов.

Логические операторы:

NOT CL — логическое отрицание содержимого CL;

OR CL,CH — побитовая операция «ИЛИ»;

AND CL,CH — побитовая операция «И»;

XOR CL,CH — побитовая операция исключающего «ИЛИ».

Команды работы со стеком:

PUSH DX — поместить в стек содержимое DX;

POP DI — восстановить данные из стека в регистр DI.

Команды перехода:

JUMP 121 — перейти к метке 121. В отладчике AFD указывается номер метки, в «классическом» языке ассемблер используется имя метки.

Команды безусловного перехода (поясним на примере оператора):

JL 125 — перейти к метке 125, если «меньше»: т.е. до этого оператора есть оператор вычитания SUB CL,CH или сравнения CMP CL,CH. Если CL < CH, то происходит переход на метку 125, если нет — по выполняется следующий оператор. Кроме оператора JL есть операторы JG переход, JZ — если результат равен 0, JNZ — если результат не равен 0, есть и другие переходы. Для выполнения работы приведенных вариантов вполне достаточно, причем все они выполняются по тому же принципу, что и оператор JL. При выполнении переходов следует следить за соответствующими признаками (флагами): при выполнении перехода JL — флаг знака SF, при переходах JZ и JNZ — флаг нуля ZF.

Теперь о методах адресации. До сих пор мы рассматривали только один из них — регистровую адресацию. На самом деле их значительно больше, но в работе используются 3:

1) регистровая (примеры выше);

2) непосредственная, суть этого метода поясним примерами:

Mov CL,5 — в регистр CL загружается число 5.

Оператор Mov CL,5B5 *не пройдет*, поскольку регистр однобайтовый.

Оператор Mov CX,5B5 пройдет, а оператор Mov CX,5B5D3 *не пройдет* (в работе обучающийся должен объяснить, почему не пройдет).

Sub DX,8 — из регистра DX вычитается число 8.

В этих примерах результат в регистрах CL и DX;

3) косвенная. Суть метода поясним примером:

Mov BX,20.

Mov [BX],DDDD.

В первой строке в регистр BX заносится номер ячейки памяти (20). Во второй строке число DDDD заносится в ячейку памяти, на которую «указывает» регистр BX. В результате работы этих двух операторов в ячейку памяти 20 занесется число DDDD.

Важный момент — косвенная адресация возможна при использовании только трех регистров: BX, SI, DI.

Вернемся к выполнению практической работы. Приведем пример конкретного фрагмента (*фрагмент 1*) программы на языке ассемблер (программа состоит из 12...14 операторов, здесь приведем только первые):

Mov Cl,3 — поместить в регистр Cl число 3.

Mov Ch,5 — поместить в регистр Ch число 5.

Mov Dl,5 — поместить в регистр Dl число 4.

Add Ch,Cl — сложить содержимое регистров Ch и Cl, результат в Ch.

Sub Ch,Dl — вычесть из содержимого Ch содержимое Dl, результат в Ch.

Dec Dl — уменьшить на 1 содержимое Dl.

…….. .

Программа заносится в компьютер, начиная с ячейки 101, экран имеет вид, как показано на рис. 3.

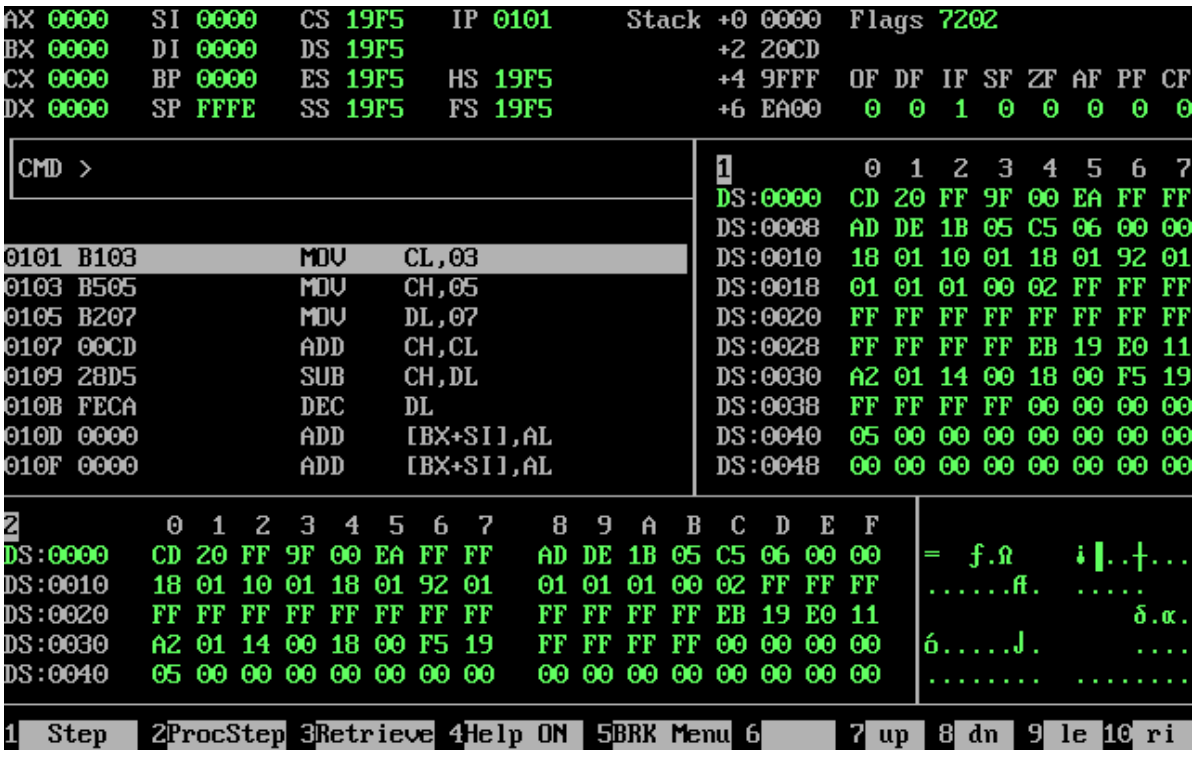

Рис. 3. Вид экрана до запуска программы на исполнение

Из рисунка видно, что программа начинается командой Mov Cl,03 с ячейки 101 (первый 0 учитывать не будем), машинный код команды в 16-ричном исчислении B103 (ему соответствует двоичный код 1011 0001 0000 0011). Машинный код занимает 2 байта, из кода хорошо виден принцип однородности — команды и данные хранятся одинаково. Поскольку первая команда занимает 2 байта, вторая команда начинается с ячейки 103, эта команда — B505 также занимает два байта и т.д. Отметим, что в счетчике команд IP находится адрес 1 команды 0101.

Далее команда запускается на исполнение в пошаговом режиме. Вид экрана компьютера после выполнения первой команды показан на рис. 4.

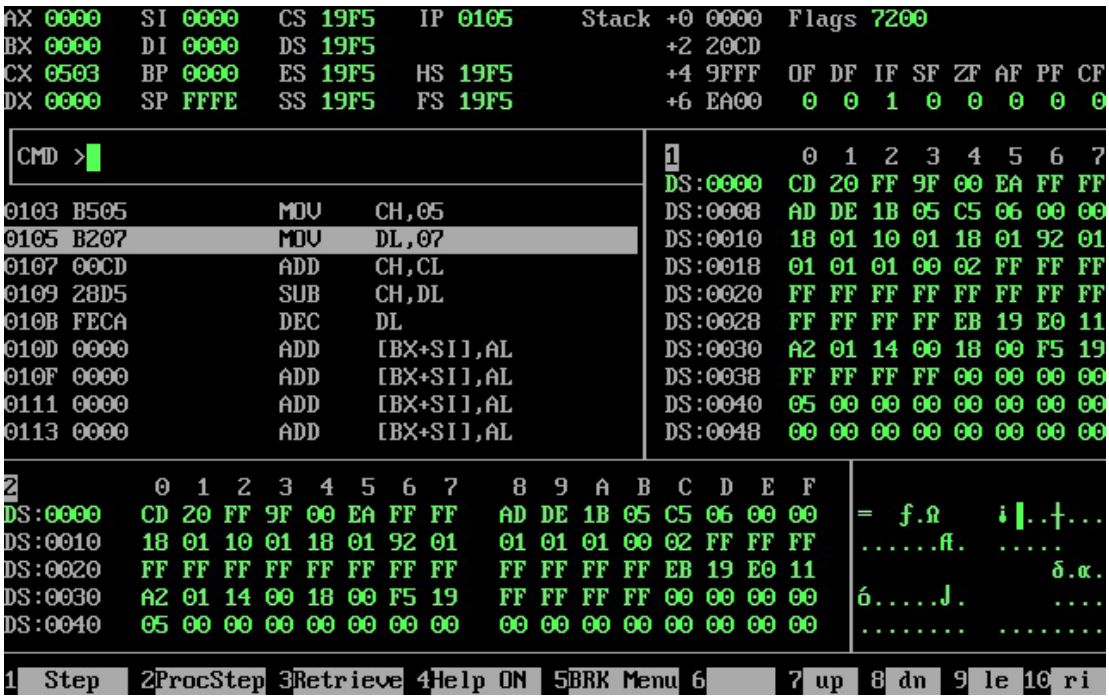

Рис. 4. Вид экрана после выполнения первой команды

Из рисунка видно, что в регистр CL записалось число 03, значение счетчика команд IP согласно принципам фон Неймана увеличилось на 2, поскольку 1 команда занимает 2 байта, и стало равным 0103.

Вид экрана компьютера после выполнения второй команды показан на рис. 5:

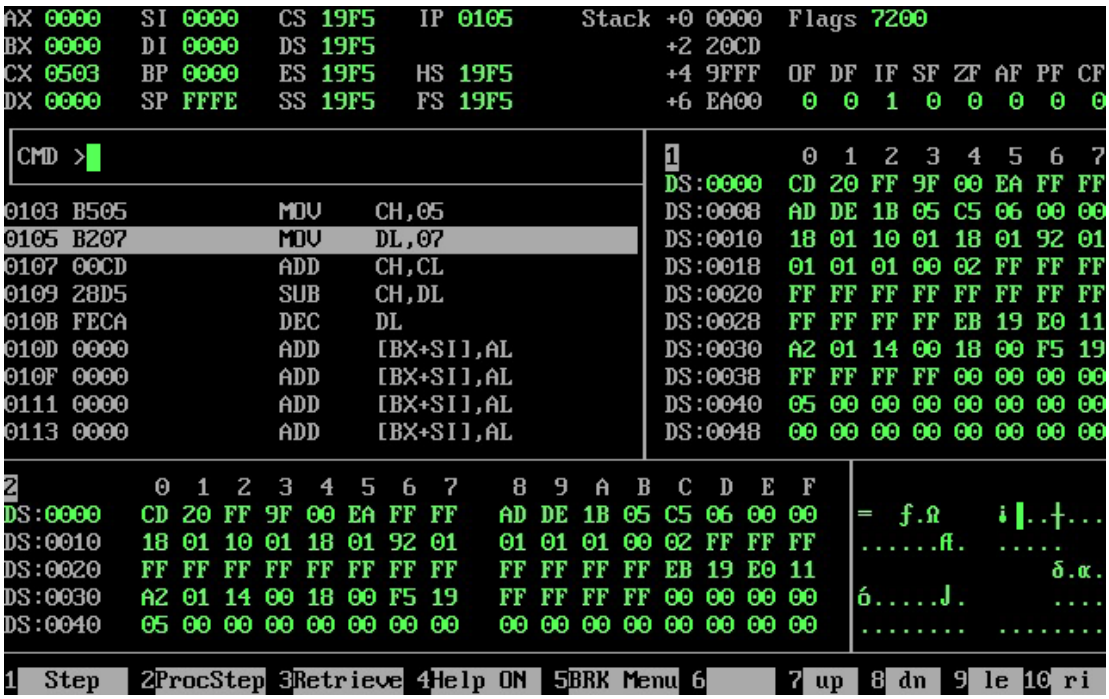

Рис. 5. Вид экрана после выполнения второй команды

Из рисунка видно, что в регистр CH записалось число 05, значение счетчика команд IP согласно принципам фон Неймана увеличилось на 2, поскольку 1 команда занимает 2 байта, и стало равным 0105.

Далее изучается *принцип записи данных из процессора в оперативную память*. Для этого используется механизм косвенной адресации, о котором говорилось ранее. Напомним, что суть этого

приема заключается в том, что вначале в регистр BX, SI или DI записывается адрес ячейки, куда мы хотим записать число, а затем с помощью косвенной адресации по этому адресу записывается само число. В ранее приведенном примере для записи в память использовался оператор типа mov [BX], DDDD, который по адресу, указанному в BX, записывает число. Чаще бывает удобным сначала записать число в регистр, а затем из регистра в память. Поясним конкретным примером:

Mov Cl,25 — в регистр Cl записывается число 25.

Mov DI,10 — в регистр DI записывается адрес «10».

Mov [DI],Cl — в ячейку памяти по адресу 10 записывается число 25.

Механизм взаимодействия процессора с оперативной памятью будем изучать с помощью следующего фрагмента (*фрагмент 2*):

```
Mov AX,1D1D.
Mov DI,10.
Mov [DI],AX.
Mov BX,2D2D.
Mov DI,12.
Mov [DI],BX.
Mov CX,3D3D.
Mov DI,14.
Mov [DI],CX.
Mov DX,4D4D.
Mov DI,16.
Mov [DI],DX.
```

В регистры AX, BX, CX и DX записываются числа 1D1D, 2D2D, 3D3D и 4D4D (и регистры, и числа одной размерности — 2 байта). С помощью косвенной адресации эти числа записываются в оперативную память по адресам 10, 12, 14, 16 (адреса с интервалом 2, поскольку числа 2-байтные).

Вид экрана после выполнения программы показан на рис. 6.

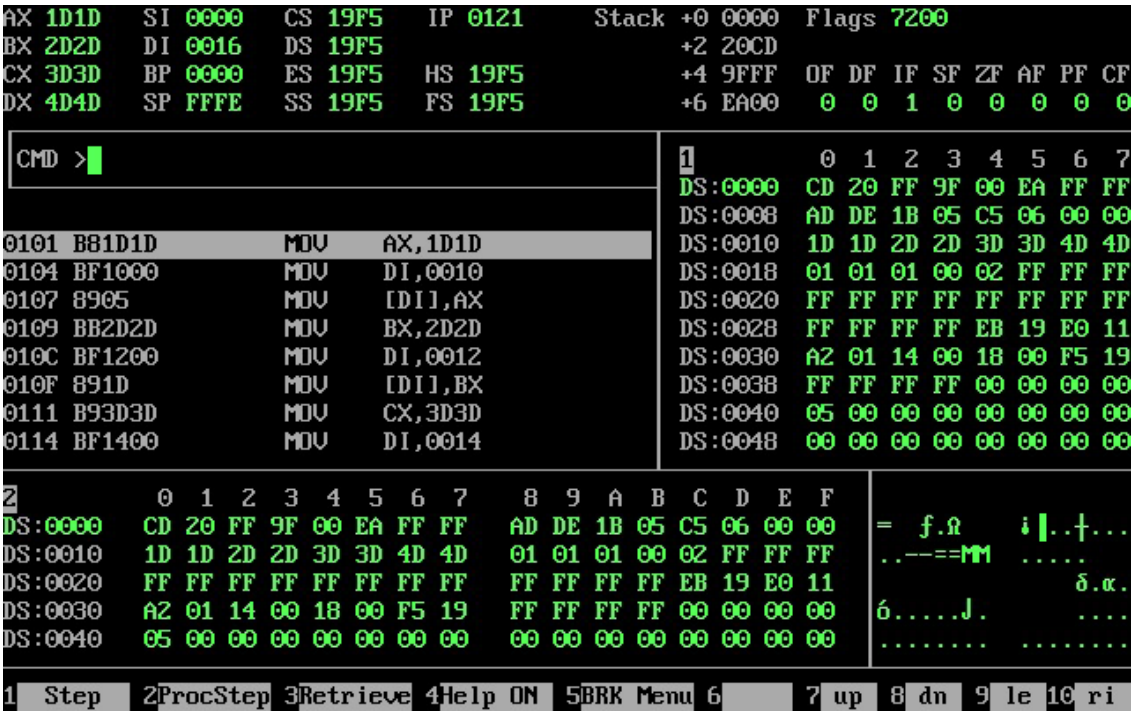

Рис. 6. Вид экрана после выполнения фрагмента 2

Из рисунка видно, что в регистрах AX, BX, CX, DX записаны числа 1D1D, 2D2D, 3D3D, 4D4D. Эти числа из *регистров процессора* записаны в *оперативную память* в ячейки 10-16 (0010-0016).

Порядок выполнения работы

1. Студент получает вариант задания типа:
– поместить число 7 в регистр DL;
– поместить число 8 в регистр DH;
– поместить число 4 в регистр CL;
– сложить регистры DH и DL;
…….. ;
– поместить число из регистра DH в ячейку памяти 20;
– поместить число из регистра DL в ячейку памяти 22;
…….. .

2. Необходимо набрать фрагмент программы на компьютере.

3. Запустить на выполнение в пошаговом режиме, понять и объяснить смысл информации на экране на каждом шаге.

4. В пошаговом режиме продемонстрировать работу преподавателю и объяснить:
– смысл изменений на каждом шаге значений регистров процессора, счетчика команд, ячеек памяти;
– механизм конкретной реализации принципа программного управления;
– размерность (в байтах) команд и регистров;
– принцип однородной памяти;
– механизм взаимодействия процессора и оперативной памяти.

Практическая работа 2
РЕГИСТР ПРИЗНАКОВ ПРОЦЕССОРА, АППАРАТНЫЕ МЕТОДЫ РЕАЛИАЗАЦИИ УСЛОВНЫХ ПЕРЕХОДОВ И ЦИКЛОВ

Регистр признаков (флагов) является важной составляющей процессора и участвует во многих механизмах работы: механизме прерывания, аппаратной реализации условных переходов, организации циклов и др. Изучение процессов с участием этого регистра (в частности механизма прерывания) важно не только для дисциплины «Электронные вычислительные машины и периферийные устройства», но и для дисциплины «Операционные системы».

В изучаемом процессоре регистр признаков (флагов) связан двусторонней шиной с арифметически-логическим устройством (см. схему микропроцессора, практическая работа 1). Регистр состоит из 16 отдельных битов (триггеров, флагов), каждый из которых отвечает за определенные события в процессоре (некоторые биты зарезервированы):

OF — флаг переполнения;

IF — флаг разрешения прерываний;

ZF — флаг нуля (имеется в виду результат операции);

SF — флаг знака (также имеется в виду результат операции);

…….. .

Флаги регистра признаков активно участвуют во многих механизмах процессора: так, флаги OF, IF участвуют в важнейшем механизме процессора — механизме прерываний (о нем пойдет речь далее), а флаги ZF, SF участвуют в аппаратном обеспечении и механизмов цикла, и условного перехода.

В данной работе рассмотрено использование в работе процессора двух флагов — ZF и SF, которые прежде всего нужны для аппаратной реализации условных переходов и циклов.

Рассмотрим вначале флаг нуля ZF. На основе этого флага работает ряд условных переходов, в частности переход JZ.

Действие этого перехода поясним на фрагменте программы:

```
101   mov  ch,7     (8)
103   mov  cl,7
105   cmp  ch,cl
107   jz   10f
109   mov  dh,0
10b   sub  bx,bx
10d   sub  bx,bx
10f   mov  dh,1
```

В процессе изучения программа запускается на выполнение дважды (для двух ветвей условного перехода). Первый раз в первый оператор mov ch,7 в регистр ch заносится значение 7, в следующем операторе в регистр cl также заносится число 7. Третий оператор cmp ch,cl сравнивает значения регистров ch, cl и, поскольку они равны, фиксируется результат «0» и флаг ZF устанавливается в 1. В операторе условного перехода jz 10f опрашивается значение флага ZF. Поскольку оно равно 1, то производится переход по адресу 10f . Во второй раз в оператор mov ch,7 в регистр ch заносится значение 8, в следующем операторе в регистр cl по-прежнему заносится число 7. Третий оператор — cmp ch,cl сравнивает значения регистров ch, cl и, поскольку они не равны, флаг ZF устанавливается в 0. В операторе условного перехода jz 10f опрашивается значение флага ZF, а поскольку оно не равно 1, то производится переход к следующему оператору mov dh,0 (заметим, что операторы sub bx,bx на результат не влияют и введены для того, чтобы лучше выделить точки перехода).

Далее этот вопрос исследуют экспериментально. На компьютере в отладчике набирают вышеприведенный фрагмент программы и запускают в пошаговом режиме. В первом случае (когда значения в регистрах cl и ch равны) экран должен выглядеть следующим образом (рис. 7).

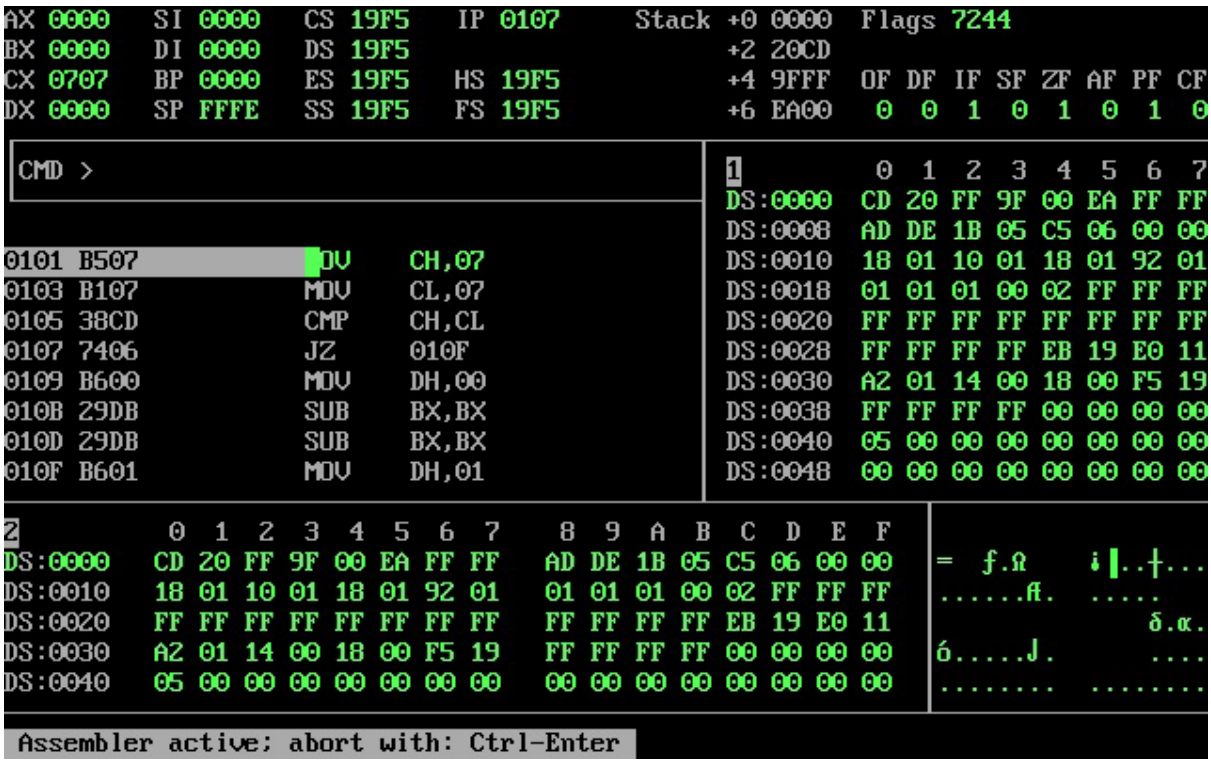

Рис. 7. Вид экрана, когда значения в регистрах cl и ch равны

Из рисунка видно, что CL = CH = 7 , значение флага ZF = 1. Поэтому в следующем шаге производится переход по адресу 010F. При этом экран имеет вид, как на рис. 8.

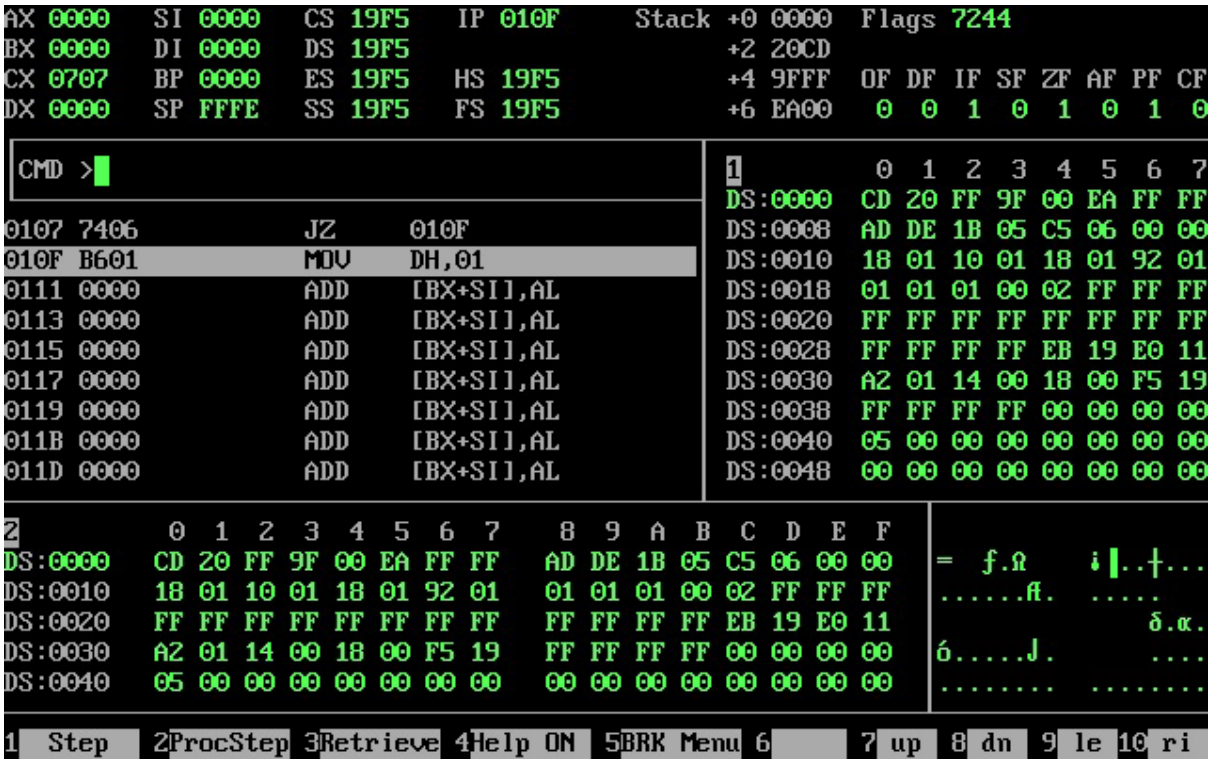

Рис. 8. Вид экрана после перехода по адресу 010F

Во втором случае (когда значения в регистрах cl = 8, ch = 7) экран после выполнения оператора cmp ch,cl должен выглядеть следующим образом (рис. 9).

Конец ознакомительного фрагмента.
Приобрести книгу можно
в интернет-магазине
«Электронный универс»
e-Univers.ru