Проект «Инженерные кадры Зауралья»

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Курганский государственный университет»

Кафедра автоматизации производственных процессов

ПОДГОТОВКА, ТРАНСЛЯЦИЯ И ПРОГРАММНОЕ ТЕСТИРОВАНИЕ СИСТЕМЫ КОМАНД

Методические указания
к выполнению лабораторной работы по дисциплине
«Микроконтроллеры и микропроцессоры в системах управления»
для студентов очной и заочной форм обучения направления 220700.62
«Автоматизация технологических процессов и производств»

Курган 2015

Кафедра: «Автоматизация производственных процессов»

Дисциплина: «Микроконтроллеры и микропроцессоры в системах управления» (направление 220700.62).

Составили: доцент В.В. Тактаев, канд. техн. наук, доцент О.В. Дмитриева.

Утверждены на заседании кафедры 27 ноября 2014 г.

Рекомендованы методическим советом университета в рамках проекта «Инженерные кадры Зауралья» 20 декабря 2013 г.

Содержание

Введение

Цель работы: изучение особенностей подготовки, трансляции и программного тестирования системы команд на лабораторном комплексе с программируемым контроллером Z-181.

1. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ КОНТРОЛЛЕРА Z80181

Контроллер Z80181 фирмы Zilog имеет следующие основные характеристики:

- Z80180 совместимый микропроцессор с 1-канальным контроллером последовательной связи Z80C30 SCC (Serial Communication Controller), контроллером счетчика-таймера Z80 CTC (Counter Timer Controller), двумя 8-разрядными универсальными параллельными портами, и двумя сигналами Chip Select;
- Высокая производительность (10 МГц);
- Малое энергопотребление в режиме прогона и режиме останова;
- Широкий диапазон напряжения питания (5 В ± 10%);
- ТТЛ совместимая CMOS;
- Генератор времени;
- 1-канальный Z85C30 контроллер последовательной связи;
- Z180-совместимое ядро включает: расширенное Z80 ядро; блок управления (диспетчер) памяти (Memory Management Unit) предоставляет доступ к 1 Мб памяти; два асинхронных канала; два канала прямого доступа к памяти; два 16-разрядных таймера; синхронизированные последовательные порты ввода – вывода; 100-pin упаковка.

Z181 центральный процессор имеет 100% совместимость программного обеспечения с Z180 центральным процессором. Кроме того, Z181 центральный процессор имеет следующие характеристики:

- Более высокое быстродействие;
- Производительность Z181 на 10-20% выше Z80;
- Расширенная схема обновления DRAM: Z181 схема регенерации DRAM обеспечивает периодическую регенерацию и генерирует 8-разрядный адрес регенерации. Она может быть заблокировано или период регенерации может быть скорректирован программно;
- Расширенная система команд: Z181 имеет двенадцать дополнительных команд к системе команд Z80 центрального процессора, куда входит команда умножения MLT;
- Останов и режимы малой мощности: Z181 центральный процессор имеет режим останова и режимы работы малой мощности, которые являются идеальными для приложений, требующих потребление малой мощности подобно батарее в переносных терминалах;

- Режим останова системы: когда Z181 SAC находится в режиме останова системы, при этом только Z181 MPU находится в режиме останова. Схемы CTC и SCC продолжают нормальную работу.

Система команд Z181 центрального процессора идентична Z180. Функциональная блок-схема контроллера Z80181представлена на рисунке 1.

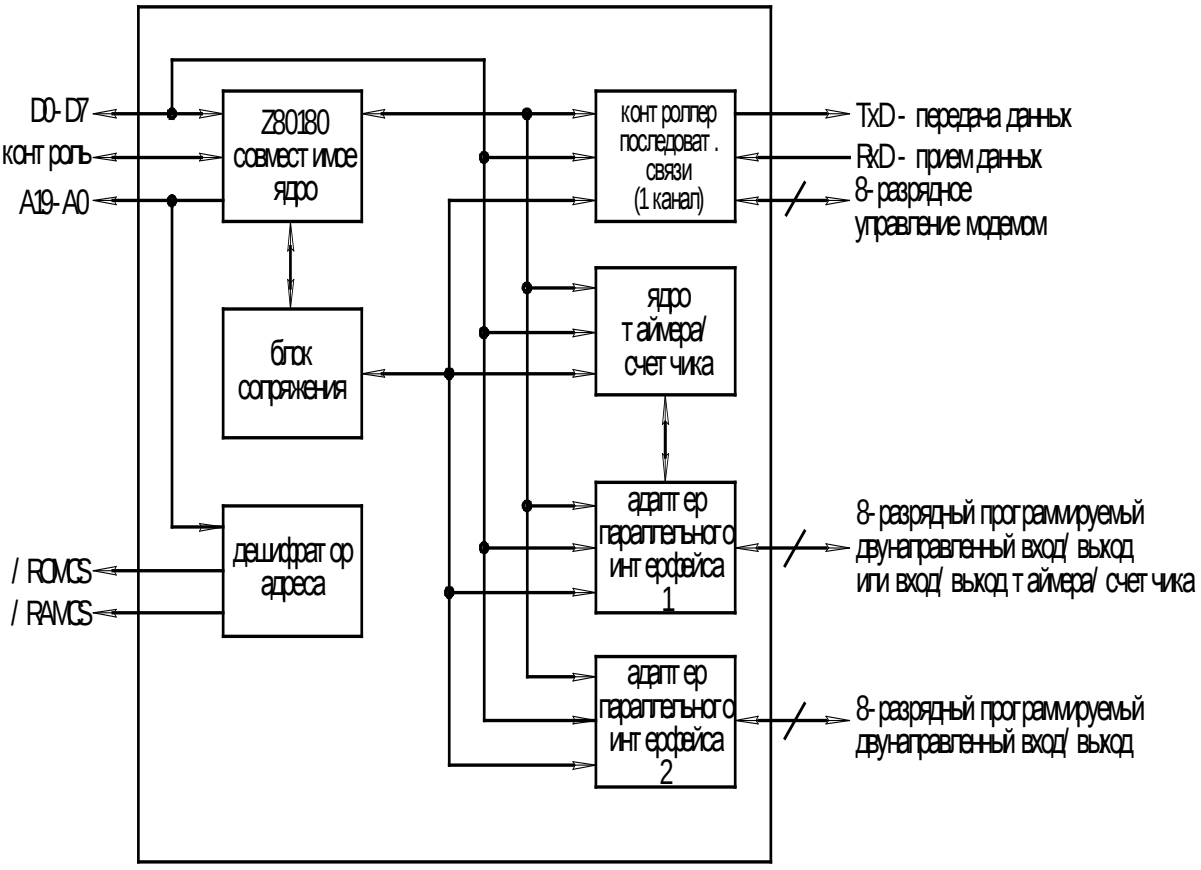

Рисунок 1 – Функциональная блок-схема контроллера Z80181

2. РЕГИСТРЫ КОНТРОЛЛЕРА

2.1. Регистры модуля микропроцессора

SAC имеет 78 внутренних 8-разрядных регистра управления встроенными устройствами. Шестьдесят четыре регистра управления Z181 MPU; два регистра управления SCC, четыре регистра управления PIA, четыре для счетчика / таймера, три для конфигурации ОЗУ / ПЗУ (границы адреса ЗУ) и один регистр системного контроля. Адреса ввода/вывода перечислены в таблице 1. Эти регистры назначены в адресном пространстве ввода/вывода, и адреса ввода/вывода полностью кодируются из диапазона A7-A0 и не имеют никакого изображения.

Таблица 1 – Адреса ввода/вывода и регистры контроллера

Адрес	Регистр
00H – 3FH	Z181 MPU регистры управления (переместимы в адресном пространстве на 040h – 07Fh или на 080h – 0BFh)
E0H E1H E2H E3H	PIA1 регистр направления передачи данных (P1DDR) PIA1 порт данных (P1DP) PIA2 регистр направления передачи данных (P2DDR) PIA2 порт данных (P2DP)
E4H E5H E6H E7H	CTC регистр управления – канал 0 (CTC0) CTC регистр управления – канал 1 (CTC1) CTC регистр управления – канал 2 (CTC2) CTC регистр управления – канал 3 (CTC3)
E8H E9H	SCC регистр управления (SCCCR) SCC регистр данных (SCCDR)
EAH EBH ECH	Регистр адреса верхней границы RAM (RAMUBR) Регистр адреса нижней границы RAM (RAMLBR) Регистр адреса границы ROM (ROMBR)
EDH	Регистр системной конфигурации (SCR)
EEH EFH	Резерв Резерв

2.2. Регистры центрального процессора

Регистры центрального процессора состоят из набора регистров GR (General Registers), набора регистров GR' и специальных регистров. Набор регистров GR состоит из 8-разрядного аккумулятора (A), 8-разрядного регистра флагов (F), и трех универсальных регистра (BC, DE, и HL), которые могут обрабатываться как 16-разрядные регистры (BC, DE, и HL) или как отдельные 8-разрядные регистры (B, C, D, E, H, и L) в зависимости от выполняемой команды. Набор регистров GR' является альтернативным набору регистров GR и также содержит аккумулятор (A'), регистр флагов (F') и три универсальных регистра (BC', DE', и HL'). В то время как содержимое альтернативного набора регистров GR' непосредственно не доступно, оно может быть программно заменено на содержимое соответствующих регистров набора GL. Специальные регистраторы включают 8-разрядный регистр вектора прерываний (I), 8-разрядный R счетчик (R), два 16-разрядных индексных регистра (IX и IY), 16-разрядный указатель вершины стека (SP), и 16-разрядный счетчик команд PC (Program Counter).

Аккумулятор (A, A') - первичный регистр, используемый для многих арифметических, логических и команд ввода/вывода.

Регистры флагов (F, F') хранят биты состояния, полученные при выполнении команд.

Универсальные регистры (BC, BC', DE, DE', HL, HL') используются и для адресации и для передачи данных. В зависимости от команды, каждая

половина (8 бит) этих регистров (B, C, D, E, H, и I) может использоваться отдельно.

Регистр вектора прерываний (I) для прерываний, которые требуют адрес таблицы векторов, чтобы быть вычисленной (INT0 режим 2, INT1, INT2 и внутренние прерывания), Регистр вектора прерываний (I) обеспечивает старший байт векторного адреса таблицы. При сбросе устанавливается в 00h.
Самые младшие семь битов **R счетчика (R)** считают число команд, выполненных Z80181. R увеличивается для каждого цикл выборки кода операции центрального процессора (каждый цикл M1). При сбросе устанавливается в 00h.

Индексные регистры (IX, и IY) используются и для адресации и для передачи данных. Для адресации, содержание смещения, определенного в команде добавляется или вычитается из индексного регистра, чтобы определить эффективный адрес операнда.

Регистр флагов (F) сохраняет логическое состояние, отражающее результаты выполнения команды. Содержимое регистра флагов используется для контроля выполнения программы и операции команды (таблица 2).

Таблица 2 – Регистр флагов

Позиция разряда	Бит / поле	R/W	Значение	Описание
7	S (sign)	R/W	0	**Знак.** S сохраняет состояние старшего значащего бита (бит 7) результата. Это полезно для действий со знаковыми числами, в которых значение бита 7=1 интерпретируется как “–”.
6	Z (zero)	R/W	0	**Нуль.** Z установлен в 1, когда выполнение команды дает 0 результат. Иначе Z сброшен в 0.
5	–	?	?	не используется
4	H (half carry)	R/W	0	**Половина переноса.** H используется командой десятичного корректирующего суммирования DAA (Decimal Adjust Accumulator), чтобы отразить заем или перенос из 4-х младших значащих битов и вследствие этого корректирует результаты двоично-десятичного сложения BCD и вычитания.
3	–	?	?	не используется
2	P/V (parity / overflow)	R/W	0	**Контроль по четности / переполнение.** P/V имеет двойное назначение. Для логических операций: P/V установлен в 1, если значение 1 бита в результате четно, и P/V сброшен в 0, если значение 1 бита в результате нечетно. Для двух арифметики дополнения: P/V установлен в 1, если результат операции выходит за пределы допустимого диапазона (от

Продолжение таблицы 2

				+127 до –128 для 8-разрядных операций, от +32767 до -32768 для 16-разрядных операций).
1	N (negative)	R/W	0	**Негатив.** N устанавливается в 1, если последней арифметической командой была операция вычитания (SUB, DEC, CP, и т.д.) и N сбрасывается в 0, если последней арифметической командой была операция сложения (ADD, INC, и т.д.).
0	C (carry)	R/W	0	**Перенос.** C установлен в 1, когда происходит перенос (сложение) или заем (вычитание) из старшего значащего бита результата. На C также воздействуют логические операции аккумулятора типа сдвигов и чередования.

R – чтение; W – запись; – - не используется; ? – не применимо

Указатель вершины стека (SP) содержит адрес ячейки памяти основанный, В МАГАЗИННОМ ПОРЯДКЕ располагают в стеке. При сбросе устанавливается в 0000h .

Счетчик команд (PC) содержит адрес команды, которая будет выполнена и автоматически обновляется после каждой выборки команды. При сбросе устанавливается в 0000h .

В контроллер Z80181 можно записывать файлы с расширением bin. Для этого программы, написанные на языке ассемблер, необходимо откомпилировать с помощью следующих процедур, описанных ниже.

3. ПОДГОТОВКА ПРОГРАММ

Файлы, которые необходимы для корректной работы программы на контроллере.

1) avlink.exe
2) avlib.exe
3) avmz80.exe
4) hexbin2.exe
5) zlibc.lib
   zlibf.lib
6) Term.exe
7) Z80.exe
8) Config.Z80
9) *.as
10) *.m8
11) *.bat

В процессе работы должны получить файлы:

12) *.bin
13) *.hex

14) *.o

15) *.prn

16) *.map

(*) или (ИМЯ) – название вашей программы, должно быть одинаковым для всех файлов. Все эти 16 файлов для удобства пользователя должны лежать в одной папке!

Создаем необходимые файлы:

1) Пишем программу на Ассемблере, сохраняем с расширением *.as

2) создаем файл с расширением *.bat

Содержит в себе следующие строки:

avmz80 ИМЯ.as OJ(ИМЯ.o) ch=HD64180 NOME QU
pause
avlink @ИМЯ. m8
pause
hexbin2 ИМЯ.hex ИМЯ.bin i E00

3) создаем файл с расширением * .m8

Содержит в себе следующие строки:

ИМЯ.com=ИМЯ.o
zlibf.lib,
zlibc.lib
-sp
-sm
-ds(.tos,faffh)
-st(m,0E000H)

4) Компилируем:

в командной строке вводим:

ИМЯ.bat ИМЯ.as

Во время компиляции должны на экране появиться следующие надписи:

C:\{Compilator ASM>avmz80 primer 1. as OJ(primer, o) ch =HD64180 NOME QU
C:\ Compilator ASM>pause Нажмите любую клавишу
C: \Compilator ASM>avlink @primerl.m8

AVL1NK V2.0 Copyright (c) 1986,1987, Avocel Systems, Inc. All right reserved.

Reading symbols from: PRIMER 1.0
Reading symbols from: ZLIBF.LIB
Reading symbols from: ZLIBC.LIB
Reading text from PRIMER1.0
Reading text from ZLIBF.LIB
Reading text from ZLIBC.LIB
C: \Compilator ASM>pause Нажмите любую клавишу
C: \Compilator ASM>hexbin2 primer 1.hex primer 1. b

После этого появятся файлы с расширениями :

*.bin

*.hex
*.o
*.prn
*.map

Нас интересует файл *.bin ,который мы будем записывать в контроллер и его же можно пошагово прогнать в программе Z80.exe

Файл *.prn может содержать сведения об ошибках.

4. ПОРЯДОК ВКЛЮЧЕНИЯ КОНТРОЛЛЕРА

Для включения контроллера, надо выключить компьютер.

Контроллер подсоединить к COM-порт A или B, включить его и компьютер в сеть. Повернуть переключатель на блоке питания в положение «вкл.». На блоке питания должны загореться красные лампочки.

Для использования возможностей стартового модуля существует утилита TERM, связывающая консоль персонального компьютера с контроллером. При первом запуске, утилита запросит параметры связи с контроллером, номер коммуникационного порта к которому присоединили, скорость соединения в бодах.

Для корректной работы с контроллером необходимо ввести номер коммуникационного порта и скорость соединения, равную 9600 бод. Перед каждым запуском необходимо удалить файл TERM.CFG, располагающийся в одном каталоге с утилитой TERM.EXE

После успешной настройки, утилита выдает знак готовности: »

Затем при одновременном нажатии на кнопки Reset и NML(синие кнопочки на плате контроллера, под крышкой) стартовый модуль выдает на экран компьютера приглашение:

Introtest
Z181 Evaluation Board Monitor
Version 1.0
ForAm29F002

5. ЗАПИСЬ ФАЙЛА В КОНТРОЛЛЕР Z80181

Для корректной работы ваш созданный bin-файл должен лежать в одной папке с файлом TERM.EXE.

Когда контроллер включён, и готов bin файл, можно начать его передачу в контроллер. Процедура передачи следующая:

1) Запускаем TERM.EXE (перед запуском удалить файл TERM.CFG)

2) Нажимаем H - выведется помощь Z180

3) Нажимаем W - запись файла ИМЯ.bin

Выведется: Z180» Write Data From:

4) Пишем E000 указываем адрес, с которого начнётся запись программы

Выведется: Write Data From: EOOO Load File For Write (L filename) »

5) Пишем L<Enter>ИМЯ.bin указываем имя файла с расширение bin

Выведется: Write Data From: E000 Load File For Write (L filename) » L<Enter>ИМЯ.bin

6) Нажимаем G (Выполнение загруженной программы* осуществляется командой G - (G)o from logical address: старт логического адреса.)

Выведется: Z180» Go from address:

7) Пишем E000

Выведется: Z1 80» Go from address: E000 <Enter> Программа начнет выполняться.

Общение с модулем осуществляется посредством команд. Команда представляет собой букву латинского алфавита, введенную с клавиатуры. После ввода одной из команд, модуль выдаст запрос на ввод параметров команды, если они требуются.

Список команд, представляющий собой переработанный ответ модуля на команду H (help), приведен ниже.

A - (A)lter memory from logical address (изменить содержимое ячеек памяти с заданного логического адреса)

B - set a (B)reakpoint (установить точку прерывания)

C - (C)ompare logical addresses (сравнить данные логических адресов)

D - (D)isplay logical memory (выдать содержимое памяти на экран)

E - (E)rase flash memory (очистить flash память)

F - (F)ill logical memory (заполнить ячейку памяти)

G - (G)o from logical address (старт с заданного логического адреса)

H - (H)elp screen (выдать подсказку на экран (помощь))

I - (l)nput byte from 16-bit I/O address (входной байт от 16-разрядного адреса ввода-вывода)

L - (L)oad an intel hex from the host (загрузить шестнадцатеричиый файл с высшего уровня)

M - (M)ove memory blocks (переместить блок памяти)

O - (O)utput byte to 16-bit I/O address (выходной байт от 16-разрядного адреса ввода-вывода)

P - (P)rogram flash memory (flash память программы)

R - display/modify cpu Registers (отобразить/изменить регистры процессора)

X - e(X)amine the memory map (посмотреть карту памяти)

V - (V)erify memory (проверить память)

W - (W)rite binary file from the host (записать двоичный файл с высшего уровня).

6. ПРИМЕР ПОДГОТОВКИ, ТРАНСЛЯЦИИ И ПРОГРАММНОГО ТЕСТИРОВАНИЯ КОМАНДЫ ADD (СУММА)

***.AS**

```
ld      de, (0e030h); указываем адрес для регистра de, в котором будет сохраняться значение;
ld      hl,(0e032h); указываем адрес для регистра hl, в котором будет сохраняться значение;
add     hl,de
ld      (0e020h), hl; указываем адрес для регистра, в котором будет сохраняться сумма
end
```

В примере пропущены команды сохранения регистра флагов до и после выполнения команды. Необходимо выбрать значения операндов с изменением признаков результата (нуля, переноса, четности и т.д.). В отчете выполнить анализ и соответствие результатов техническому описанию

***.BAT (здесь условно название ИМЯ** на стр 11**,** или выше ***.**AS заменено на 22.AS)

```
avmz80 22.as OJ(22.o) ch=HD64180 NOME QU
pause
avlink @22.m8
pause
hexbin2 22.hex 22.bin i E00
```

***.m8**

```
22.com=22.o
zlibf.lib,
zlibc.lib
-sp
-sm
-ds(.tos,faffh)
-st(m,0E000H)
```

***. PRN**

```
1;    сложение 2 двухбайтных чисел из ячеек E030/31 и E032/33
2       ld      hl,(0e030h);  первое слагаемое        0001' ED 5B E030
3       ld      de,(0e032h);  второе слагаемое        0004'2A E032
4       add     hl,de (0e032h);  сложение             0007' 19
5       ld      (0e020h), hl    ;  сохранить сумму    0008' 22 E020
7       end
```

Для проверки и пошаговой прогонки загружаем ваш файл *.BIN в программу Z80.exe:

1. Запускаем программу Z80.exe
2. Откроется окно, ALT+D
3. Если у вас все лежит в одной папке, то можно стереть тот путь, который предлагает программа и написать (.\ ) и вы сразу окажетесь там, где лежит ваш файл
4. Выбираем и загружаем файл *.BIN
5. Вы увидите текст своей программы (рисунок 2)

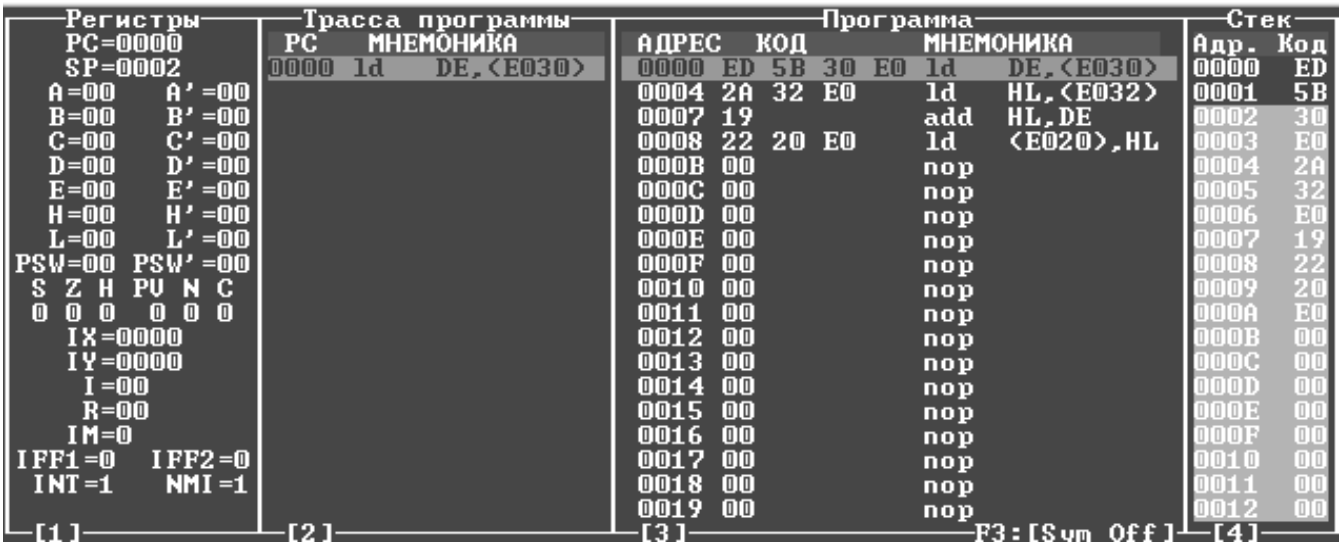

Рисунок 2 – Окно программы

6. Клавишей TAB переходим в окно ПРОГРАММА
7. Меняем в столбце все E коды на 0 (рисунок 3)

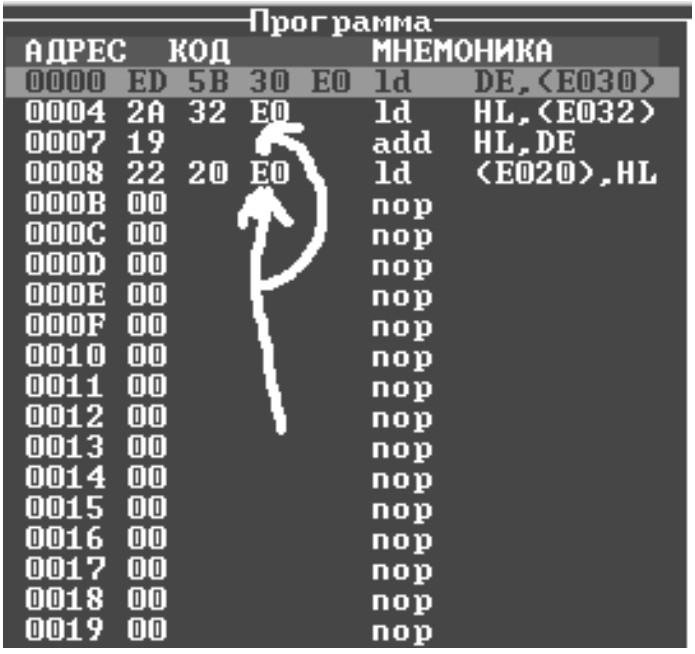

Рисунок 3 – Замена кодов программы

Окно после изменений представлено на рисунке 4.

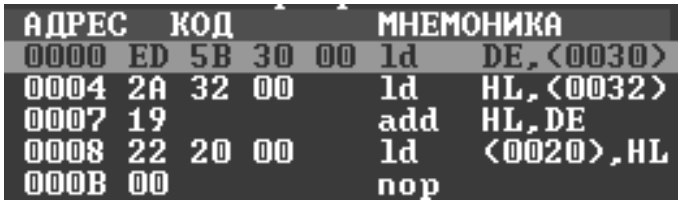

Рисунок 4 – Фрагмент окна программы после изменений

Последнее действие необходимо, т.к. контроллер Z80181 и программа Z80 воспринимают разные коды.

8. В зависимости от того из каких ячеек вы грузили регистры de и hl (в нашем случае это E030 и E032), надо прописать значения, которые вы хотите положить в эти адреса.

9. Курсором спускаемся к этим адресам. В нашем примере мы для регистра de записали первое слагаемое - число 0AAA, а для hl – второе слагаемое - число 5555. При записи значений меняем порядок записи символов (если для de, например, надо занести число 0AAA, то сначала в первую ячейку мы вносим AA, а во вторую 0A), т.е. в младшие адреса младшие разряды (рисунок 5).

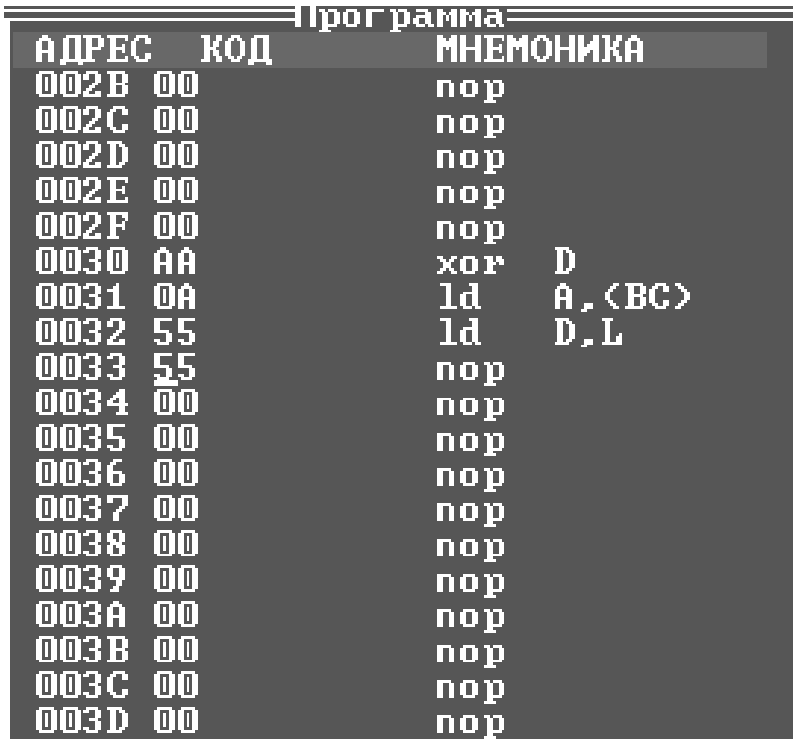

Рисунок 5 – Запись значений в регистрах

10. Теперь поднимаемся вверх и пошагово клавишей F7 запускам каждую строчку программы.

11. В нашем примере после прогонки команды в столбце регистров наблюдаются изменения, которые показаны на рисунке 6.

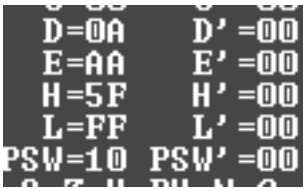

Рисунок 6 – Фрагмент окна программы с изменениями в столбце регистров

12. В столбце «Программа» спускаемся ниже, (в нашем примере до адреса E020, куда мы изначально в программе прописали сумму) и смотрим результат (рисунок 7).

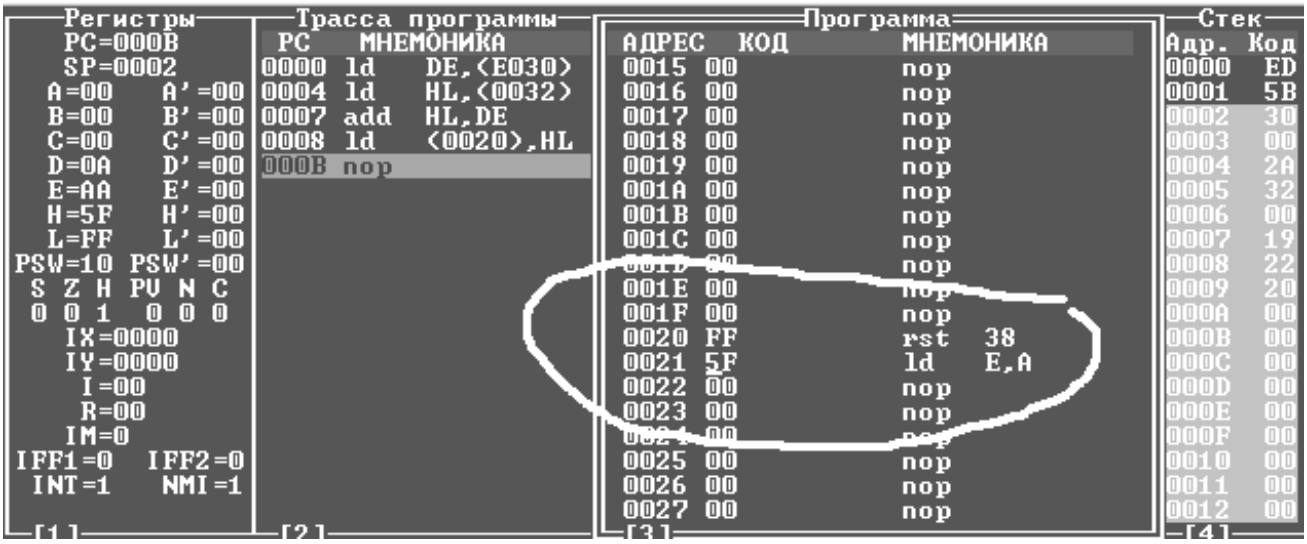

Рисунок 7 – Окно программы с результатами проверки программы

Если проверка прошла успешно, далее проверяем работу программы на контролере.

Загрузка файла *.BIN в контроллер осуществляется в следующей последовательности:

1. Подключаем контроллер и запускаем программу TERM.exe (Ваш файл и программа должны быть в одной папке). Не забудьте стереть файл TERM.CFG!

2. Выполняем последовательно действия, описанные в разделе «Запись файла в контроллер Z80181»

Нажимаем W - запись файла ИМЯ.bin

Выведется: Z180» Write Data From:

Пишем E000 указываем адрес с которого начнётся запись программы

Выведется: Write Data From: EOOO Load File For Write (L filename) »

Пишем L<Enter>ИМЯ.bin указываем имя файла с расширение bin

Выведется: Write Data From: E000 Load File For Write (L filename) » L<Enter>ИМЯ.bin

3. Вручную записываем значения тех регистров в те адреса, которые мы писали в программе:

а) нажимаем клавишу F

выведется: Fill logical memory:

Там же надо указать адрес той ячейки куда сохраняемся
Fill logical memory: E030 (это в нашем примере)
Выведется: How many bites? Пишем: 1
Выведется: Data write? Пишем: AA

б) нажимаем клавишу F

выведется: Fill logical memory:
Там же надо указать адрес той ячейки, куда сохраняем.
Fill logical memory: E031 (это в нашем примере)
Выведется: How many bites? Пишем: 1
Выведется: Data write? Пишем : 0A

в) нажимаем клавишу F

выведется: Fill logical memory :
Там же надо указать адрес той ячейки, куда сохраняем
Fill logical memory: E032 (это в нашем примере)
Выведется: How many bites? Пишем: 2
Выведется: Data write? Пишем: 55

4. Проверяем запись:

нажимаем клавишу D
выведется: Display logical memory:
надо написать начальный адрес E000
выведется: Display logical memory: E000
Выведется: How many bites?
Пишем 40 (т.к. чтобы увидеть то, что мы записали в 30-ые адреса)

5. Выведется 3 или 4 строки

В строке E030 в начале должны быть те значения которые мы записывали : 0AAA 5F FF

6. Проверяем в адресе, где должна быть сумма:

нажимаем клавишу D
выведется: Display logical memory:
надо написать начальный адрес E020(это в нашем случае)
выведется : Display logical memory: E020
выведется: How many bites?
пишем 2

7. Должен получиться ответ : FF 5F (такой же как и в программе Z80)

7. Для примера 6 - программы, написанной на языке C, сначала необходимо откомпилировать в ассемблерский код, а затем ассемблер необходимо откомпилировать в bin файл. Для компилирования из C в ассемблер необходимо использовать следующую процедуру.

Компилятор состоит из следующих файлов:

cc.bat – файл запуска
cpp.exe – из *.c делает *.pp
p1.exe – из *.pp делает *.p1
cgen.exe – из *.p1 делает *.cg

optim.exe – из *.cg делает *.as
avmz80.exe – создаёт объектный файл
Файл cc.bat включает следующие надписи:
cpp %1.c %1.pp
p1 %1.pp %1.p1
cgen %1.p1 %1.cg
optim %1.cg %1.as
avmz80 %1.as OJ(%1.o) NOME QU
Компиляция:
В командной строке вводим (расширение компилируемого файла нужно стереть): cc.bat primer1

После компиляции должны создаться файлы с расширениями: *.as, *.cg, *.o, *.p1, *.pp, *.prn. Файл *.prn содержит сведения об ошибках. Далее созданный файл *.as необходимо откомпилировать в bin-файл.

7. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какова типичная разрядность таймера/счетчика в составе микроконтроллера? (32, 64, 8 или 16, 4)
2. Какие различия векторов прерываний микропроцессоров I8080 и Z80181?
3. Каковы основные отличия микропроцессора I8080 (КР 580) от Z80181?
4. Имеется программная совместимость от I8080 к Z80181? А наоборот? Почему?
5. Как формируется физический адрес памяти из адреса смещения и содержимого регистров специальных функций?
6. Чем проверяется правильность хода компиляции?
7. Какие ошибки исходного теста выявляются транслятором с языка Ассемблер, как они отмечаются в файле *.PRN?

8. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

Работа выполняется группой из 2 человек. Контроль подготовки к выполнению осуществляется преподавателем, возможно с уточнением задания для отдельного студента. В качестве индивидуальных заданий могут быть использованы следующие примеры или аналогичные им.

ПРИМЕР 1: Проверка команды SBC(разность)
***.AS**

```
ld      de,(0e030h);  уменьшаемое
ld       hl,(0e032h);  вычитаемое
sbc      hl,de;            разность
ld       (0e035h), hl; сохранить для анализа
end
```

ПРИМЕР 2: команда EX

***.AS**

```
ld      hl,(0e030h); исходное содержимое HL
ex      de,hl          ; обменять с DE
ld      a,de           ; читать ячейку по адресу в DE
ld      (hl+1),a       ; сохранить по адресу HL+1
ld      (0e032h),de; в E032 копия E030
push  psw;  сохранить аккумулятор в стеке
ex      de,hl;  восстановить HL
end
```

ПРИМЕР 3: команда ADC

***.AS**

```
ld      a, (0e030h); предварительно занести первое слагаемое в ячейку E030
ld      hl, (0e033h);предварительно занести второе слагаемое в ячейку E033
adc     a,(hl); выполнить сложение с установкой признака C в 0 или 1
ld      (0e036h), a; сохранить результат в E036
end
```

ПРИМЕР 4: команда OR

***.AS**

```
ld      a,0055h;      первый операнд
ld      hl,0022h; предварительно в E022 занести второй операнд
or      m;      выполнить поразрядное логическое сложение
ld      (0e035h),a; результат сохранить в E035
end
```

ПРИМЕР 5: проверка символа шестнадцатеричного кода

Код программы, Файл ishex.c: (компиляция см п.7)

```
CC.bat ishex
main (c)
unsigned char c;
{
if ((c>='0' && c<='9')||
(c>='a'&& c<='f')||
(c>='A'&& c<='F'))return(1);
return(0); возврат 0, если символ не относится к знакам
}               шестнадцатиричного кода
end;
```

Сначала в папку Compilator C нужно положить файл с программой и откомпилировать её в ASM-файл, по технологии описанной выше. В результате созданный ASM-файл должен содержать следующие строки:

```
        defseg      c_text,class=CODE
        defseg      c_data,class=CODE
```

```
        defseg      c_bss,class=DATA
        globalncsv, cret, indir
        global_main
        globalbrelop
        globalwrelop
        globalbrelop
globalwrelop
        globalbrelop
        globalwrelop
        seg   c_text
_main:
        global csv
        call csv
        ld    b,48
        ld    a,(ix+6)
        call  brelop
        jp    c,u231            ; если байт больше 48 – код буквы F
        ld    e,(ix+6)
        ld    d,0
        ld    hl,57
        call  wrelop
        jp    nc,u630
u231:
        ld    b,97
        ld    a,(ix+6)
        call  brelop
        jp    c,u431            ; если байт больше 97 – код буквы f
        ld    e,(ix+6)
        ld    d,0
        ld    hl,102
        call  wrelop
        jp    nc,u630
u431:
        ld    b,65
        ld    a,(ix+6)
        call  brelop
        jp    c,l2          ; если байт больше 65 - код буквы a
        ld    e,(ix+6)
        ld    d,0
        ld    hl,70
        call  wrelop
        jp    c,l2
u630:
        ld    hl,1          ;возврат 1
```

```
jp      cret            ;символ относится к шестнадцатиричным
l2:
        ld      hl,0            ;возврат 0
jp      cret            ;символ не относится к шестнадцатиричным
        global_end
        seg     c_bss
_end:
        defs    2
        seg     c_text
        end
```

Далее, если нет ошибок (проверка файла ishex.prn), необходимо созданный файл ishex.as перенести в папку Compilator ASM, для дальнейшего его преобразования в bin-файл. Не забудьте сменить названия в файлах ASM_compiler.bat и ishex.m8. После этого, используя выше описанную методику, файл с ассемблерским кодом компилируется в bin-файл.

Созданный bin-файл должен содержать следующие данные:

```
00000000: CD 7D E0 06 30 DD7E 06  | CD 5A E0 DA 1C E0 DD 5E  =}p♠0▌ ~♠=Zp ┌∟p▌ ^
00000010: 06 16 00 21 39 00 CD 6E | E0 D2 4E E0 06 61 DD 7E  ♠▬ !9 =np╥Np♠a▌ ~
00000020: 06 CD 5A E0 DA 35 E0 DD | 5E 06 16 00 21 66 00 CD  ♠=Zp ┌5p▌ ^♠▬ !f =
00000030: 6E E0 D2 4E E0 06 41 DD | 7E 06 CD 5A E0 DA 54 E0  np╥Np♠A▌ ~♠=Zp ┌Tp
00000040: DD 5E 06 16 00 21 46 00 | CD 6E E0 DA 54 E0 21 01  ▌ ^♠▬ !F =np ┌Tp!☺
00000050: 00 C3 89 E0 21 00 00 C3 | 89 E0 D5 5F A8 FA 64 E0  ├Йр!  ├Йр ╒_и·dp
00000060: 7B 98 D1 C9 7B E6 80 57 | 7B 98 7A 3C D1 C9 7C AA  {Ш╤╔{цAW{Шz<╤╔|к
00000070: FA 76 E0 ED 52 C9 7C E6 | 80 ED 52 3C C9 E1 FD E5  ·vpэR╔|цAэR<╔c¤x
00000080: DD E5 DD 21 00 00 DD 39 | E9 DD F9 DD E1 FD E1 C9  ▌ x▌ !  ▌ 9щ▌ ·▌ c¤c╔
00000090: E9 E1 FD E5 DD E5 DD 21 | 00 00 DD 39 5E 23 56 23  щc¤x▌ x▌ !  ▌ 9^#V#
000000A0: EB 39 F9 EB E9          |                          ы9·ыщ
```

Примечания:

1. При выборе команд, нужно учитывать какие регистры работают с этими командами.
2. При записи адресов: перед адресом всегда ставить 0E и после него h
3. Складываем или вычитаем: одиночные регистры с одиночными, двойные - с двойными .

В примерах пропущены команды сохранения регистра флагов до и после выполнения команды. Необходимо выбрать значения операндов с изменением признаков результата (нуля, переноса, четности и т.д. В отчете выполнить анализ и соответствие результатов техническому описанию

Зафиксировать все результаты работы, показав для проверки преподавателю, после его отметки работа считается выполненной.

Оформить отчет и защитить лабораторную работу.

Список использованных источников

1. Новиков Ю.В., Скоробогатов П.К. Основы Микропроцессорной техники. Курс лекций. Интернет-Университет Информационных Технологий. Москва, 2004.

2. МикроЭВМ: В 8 кн.: Практическое пособие/Под ред. Л.Н.Преснухина. Кн. 3. Семейство ЭВМ «Электроника К1»/А.В. Кобылинский, А.В. Горячев, Н.Г. Сабадаш, В.В. Проценко - М.:В.школа, 1988.

3. Технические описания:

3.1. UM0050.PDF – Z8018x Family MPU/ User Manual -326 стр.

3.2. UM0055.PDF – eZ80C - Compiler Version 1.03/ User Manual -146 стр.

Тактаев Владимир Васильевич

Дмитриева Ольга Венедиктовна

ПОДГОТОВКА, ТРАНСЛЯЦИЯ И ПРОГРАММНОЕ ТЕСТИРОВАНИЕ СИСТЕМЫ КОМАНД

Методические указания
к выполнению лабораторной работы по дисциплине
«Микроконтроллеры и микропроцессоры в системах управления»
для студентов очной и заочной форм обучения направления 220700.62
«Автоматизация технологических процессов и производств»

Авторская редакция

Подписано в печать 17.03.15	Формат 60x84 1/16	Бумага 65 г/м^2
Печать цифровая	Усл. печ.л. 1,5	Уч.-изд.л. 1,5
Заказ 57	Тираж 25	Не для продажи

РИЦ Курганского государственного университета.
640000, г. Курган, ул. Советская, 63/4.
Курганский государственный университет.