Experiment 5

Testing the 80806 Microcomputer System

5.1 Introduction

By this experiment your system should include the following:

1. an 8086 microprocessor,
2. a clock generator with 15MHz crystal,
3. a fully demultiplexed bus system (74LS373 octal latches),
4. a memory system including two SRAM memory chips and two EPROM memory chips each of size 8Kbytes,
5. decoders, and
6. simple I/O ports (switches, LEDs, tri-state buffer, and octal latch)

To find if the system is working properly, we will write a simple program (see Figure 5.1) that light LEDs one after another one at a time in a sequence from right to left.

```
    MOV  AL,  01h    ;set the LSB of register AL

L1: MOV  CX,  0FFFFh ;load the counter CX with FFFFh

L2: OUT  00h, AL     ;output AL to port 00h (output port)

    LOOP L2          ;repeat the operation until CX becomes 0

    ROL  AL,  1      ;rotate AL one bit position to the left

    JMP  L1          ;go back to L1
```

Figure 5.1: Test Program

5.2 Equipment

- Use of a prototype-board that already includes an 8086 CPU operating in minimum mode with clock generator and a fully demultiplexed data and address buses in addition to two 8 Kbytes SRAM memories (6264) and two 8 Kbytes EPROM memories (2764),
- MS-DOS Debugger,
- EPROM Eraser,
- EPROM Programmer,
- Oscilloscope,
- Logic Probe, and
- Multimeter

5.3 Procedure

1. Use the MS-DOS debugger to find the machine code of the test program as shown in Figure 5.2.
2. Take out the EPROMs from your system and label them as EVEN BANK and ODD BANK.
3. Place the two chips in the EPROM eraser.
4. Load the machine code of the test program (see Figure 5.3) into the EPROMs using the EPROM programmer (Load even bytes of the machine code into the even back, and the odd bytes into the odd bank).
5. Place the programmed EPROMs back on your prototype-board.
6. Make sure that the EVEN BANK chip is connected to the even byte of the bus (D0-D7) and the ODD BANK chip is connected the odd byte of the bus (D8-D15).
7. Connect your system to the power supply and check output displayed on the LEDs.

```
C:\> Debug
-a 0000:0000
0000:0000    MOV  AL,   1
0000:0002    MOV  CX,   FFFF
0000:0005    OUT  0,    AL
0000:0007    LOOP 0005
0000:0009    ROL  AL,   1
0000:000B    JMP  0002
0000:000D
-

-u 0000:0000
0000:0000    B001      MOV  AL,   1
0000:0002    B9FFFF    MOV  CX,   FFFF
0000:0005    E600      OUT  0,    AL
0000:0007    E2FC      LOOP 0005
0000:0009    D0C0      ROL  AL,   1
0000:000B    EBF5      JMP  0002
.            .         .
.            .         .
```

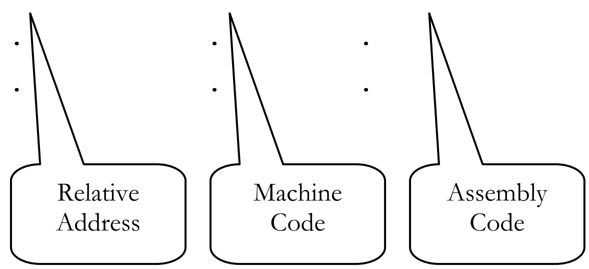

Figure 5.2: Finding the machine code

Byte #	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Machine Code	B0	01	B9	FF	FF	E6	00	E2	FC	D0	C0	EB	F5	FF

Figure 5.3: Machine code to be loaded into the EPROMs

5.4 Debugging the System

In case that your system is not functioning, you can carry out hardware testing of the microcomputer system through general inspection and hardware debugging as explained in the following steps:

Step 1: Visual inspection and testing:

1. Make sure that the VCC and GND you are using are appropriate. Use oscilloscope to measure VCC on your system.

2. Identify the VCC and GND lines on your board and make sure all chips receive them on the right pins. For this you may carry out visual inspection to avoid applying reverse voltage on the chips.

3. Test all VCC and GND on all the chips using the Oscilloscope or a Logic Probe.

4. Check the following signals using the Oscilloscope:

    a. the Reset circuit at the input of 8086 CPU,

    b. the CLK input of 8086 CPU, and

    c. the ALE output of 8086 CPU.

5. Using a multimeter and the map of your design you need to check the following:

    a. all inter-connections between the address bus lines at output of octal latches and the memories,

    b. all inter-connections between the CPU and memories data lines, and

    c. all control connections for Read/Write, chip-select, and **all default connections**.

Step 2: Testing the logic operations of the system.

We expect the test program (Figure 5.1) to generate some pattern of chip select on the EPROM memory because the program is stored there. Also this program is supposed to write data to the I/O port, so we expect I/O write cycles on the control of output port. One may consider that the system is working fine if the chip-select pattern and

I/O write cycles are observed in the right order. For this test we have to follow the steps below:

1. Analyze the program and find out the expected chip-select pattern on the EPROM and I/O write cycles.

2. Turn on your microprocessor system, and use the oscilloscope to check memory and I/O read signals (i.e., chip-select of the EPROMs and latch-enable of the I/O port).

3. If you do not see the expected pattern, then there is still a problem with your system. In this case you need to do further investigation of the system:

    a. Check Reset, CLK, and ALE on the CPU. If an error is found, then correct it and repeat the test.

    b. Check the ALE signal on the control of the octal latches.

    c. You may need to use the Logic Analyzer and set the triggering condition to valid EPROM select, and then follow up the timing step-by-step. Following a reset the CPU must generate the bootstrap address. If it does not, then the starting conditions are not OK. You better carefully check Reset, CLK, and ALE on the CPU. If the above address is generated but control is lost, then your address connections and data connections from CPU to memories are likely to contain some errors. You need to check them again and restart the procedure.

Exercises

5.1. Write an assembly program that continuously reads one byte from the input port, complement it and sent it to the output port. Test this program on your system.

5.2. Write an assembly program to display an 8-bit counter on the LEDs of your system.

5.3. Write an assembly program to add 2 four-bit numbers and display the result on the LEDs of your system. The two numbers are entered through the 8-DIP switch (i.e. each 4 switches represent one number).