

NORTHWESTERN UNIVERSITY TECHNOLOGICAL INSTITUTE

ECE 270

Experiment #9
MICROPROCESSORS
PRELAB

1. Fill in the Hex Code and Comment in the table for Experiment C.
2. Fill in the Hex Code and Comment in the table for Experiment D.
3. Do number 15 under Experiment G.

NORTHWESTERN UNIVERSITY TECHNOLOGICAL INSTITUTE

ECE 270

Experiment #9 **MICROPROCESSORS**

Reference: Smith, R.J., Electronics, Circuits and Devices, Wiley, 1980. Chapters 9.

In the past experiments, we examined various digital devices and understood how these basic elements are used to develop larger and more elaborate digital systems. One of these digital systems is known as the microprocessor. A microprocessor is a single integrated circuit (or sometimes just called a chip) that possesses at least 75% of the computing and data manipulation power of a digital computer. It usually cannot function without the aid of support chips and memory. A microcomputer is a fully operational digital computer that is based on a microprocessor chip.

A computer program can be defined as a series of instructions written in a form acceptable to the computer, the purpose of which is to achieve a certain result. With microcomputers, we are more interested in writing microcomputer programs that control the operation of a device or machine. For example, in a home clothes washer, you may want to control the amount of water used, or the duration of the various washing cycles. All of this can be done with a correctly written and organized computer program.

All instructions used in computer programs are made up of a set of characters that define an operation. Computer instructions may be expressed in a variety of forms. They can be expressed as binary numbers (11010011, 00111110), hex numbers (D3H, 3EH), mnemonic code (OUT 3EH; LD A, 02H), etc.

Our common numbering system is a base 10 system. We can count from 0 to 9 and then we have to add a second digit, etc. This counting system is then said to have a base or radix of 10. A binary numbering system has a base of 2 and a hexadecimal numbering system has a base of 16. A large number represented in the binary numbering system would have many digits and could be difficult to remember. One way to remember 8-bit binary numbers is to use hexadecimal code (we are using 8-bit numbers here because we will use them when we program our microprocessor). It is possible to convert binary numbers into hex numbers so they are easier to remember. Note that when a number has a capital H after it, it refers to a number in the hexadecimal system. For example, 3EH refers to a hex number of 3E and can be converted to a binary number using the table below. In our example, 3EH can be converted into the binary code 00111110. It is a standard contention to put a space between each 4-bit group of binary numbers, so it is easier to read. (0011 1110). Each 8-bit number is converted into a 2 digit hex code.

<u>Decimal Number</u>	<u>Binary Number</u>	<u>Hex Number</u>
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

THE Z80 MICROPROCESSOR

The basic function of a microprocessor is to perform arithmetic, logic, and control operations on data obtained from input devices, perform some operation on the data based on instructions the programmer has placed into the microprocessor, and provide the results of that operation to various output devices, such as a video display terminal.

The microprocessor that we are going to use is the Zilog Z80. This microprocessor was introduced in 1976 and is still one of the most popular devices on the market. It is contained in a 40-pin DIP (Dual In-Line Package) and uses a set of 158 instructions. For more information about microcomputers technology and the Z80, refer to Chapter 9 of Electronics.

The Z80 uses instructions that are entered into the microprocessor using hex code, with mnemonic codes to assist the programmer in formulating his program. For most of the experiments, the instructions will be provided. The complete instruction set can be obtained in the lab.

OPERATION OF THE MICROPROFESSOR

The Micro-Professor contains a QWERTY keyboard, which is used to input program instructions and control the operation of the microprocessor trainer. See pp. 30-47 in the user's manual accompanying the trainer for a description of the commands you'll be using in the experiments.

PROCEDURE

In this experiment, our goal is to become familiar with the keyboard controls and display elements of the Micro-Professor. Make sure to read carefully the explanations of the various commands BEFORE you start the experiment. As an introduction, perform the following example.

LOAD the A (accumulator) register with 27H. Load the B register with 03H. ADD the contents of the B register to the A register and store the result in A. Halt the Z80.

To find the instructions necessary to implement this program, we look into the Z80 instruction set and find:

<u>Mnemonic</u>	<u>Hex Code</u>	<u>Comment</u>
LD A,n	3E n	Load register A with a 2 digit hex number n
LD B,n	06 n	Load register B with a 2 digit hex number n
ADD A,B	80	Add B to A and store the result in A
BREAK	F7	Break the execution

To load this into the trainer, press RS to reset the system. The display should read ****MPF-I-PLUS****. The first program line must start at the memory location F000H. To get to this location, type in M. The display should then show <M>. Now enter the memory address F000. If you then press the carriage return, you will see displayed the present contents of memory location F000 to F003 (try it).

We wish instead to enter numbers into memory, which can be done by typing a colon after the memory address. To try this, type in MF000: You can now begin entering the hex code for the program, separating each two digit hex number with a space. After entering the final instruction (BREAK), press the carriage return. Up to 48 numbers or spaces can be entered at a time. Now that all instructions have been entered, they can be checked by entering MF000 (carriage return). The first four bytes of instructions are shown in the display. To see the next four bytes, press the down arrow key.

When the breakpoint is encountered, program execution stops, and the display show the address of the next instruction (for this example, F006), and the contents of register pairs AF and BC. The contents of the other registers can be observed by pressing the down arrow key. Since this program added two hex numbers and deposited the sum in A, the first two hex digits in the AF register pair display should be the sum of these two numbers.

To run the program, press GF000 (carriage return). "G" means run, and F000 is the program's starting address in memory. The program is halted by the breakpoint, and the contents of register pairs AF and BC is displayed.

1. What is the hex number that is in register A?
2. Does the number in register A correspond to the sum of 27H and 03H?
3. What would be the procedure to ADD 2FH and 45H? Execute the program with these numbers. What do you get for the answer? Is it correct?

PROGRAMMING EXPERIMENTS:

The following experiments will provide you with an opportunity to understand the basics of programming. For some of the experiments, the instructions will be provided. However, you should look up these instruction in the instruction set provided to make sure that they are correct.

Experiment A:

LOAD A with 48H, LOAD C with FFH, ADD A and C and store the result in B, then BREAK.

4. What are the contents of registers A, B and C after running this program?

Experiment B:

<u>Mnemonic</u>	<u>Hex Code</u>	<u>Comment</u>
LD A,11H	3E 11	Load register A with 11H
LD (F007H),A	32 07 F0	Load memory location Foo7 with contents of A
BREAK	F7	

Enter the above program into the trainer and run it. Notice that we are using a new notation for loading a number from a register to memory location. Also notice that the location number is entered with the last two digits of the address (07) coming first. This is a feature of the Z80 instruction set. In general, the instruction is LD (nn),A, where nn is a 4-digit hex number indicating the address to which data is to be deposited.

Experiment C:

Manually load this data: 27H at address F00D, and 03H at F00E. Translate and enter the following program:

<u>Mnemonic</u>	<u>Hex Code</u>	<u>Comment</u>
LD A,(F00D)		
LD B,A		
LD A,(F00E)		
ADD A,B		
LD(F00F),A		
BREAK		

5. What does this program do?
6. What is in memory location F00F?

Experiment D:

Translate and enter the following program:

<u>Mnemonic</u>	<u>Hex Code</u>	<u>Comment</u>
LD B,08H		
INC B		
LD C,B		
INC C		
LD D,C		
INC D		
LD E,D		
DEC E		
LD H,E		
DEC H		
LD L,H		
DEC L		
LD A,L		
BREAK		

Verify that this program was loaded correctly.

7. Analyze the program and predict the value which will be in each register at the end of the program.
8. After running the program, what are the values in register A, B, C, D, E, H, and L?
9. Now change the data byte at location F001H to 01H and repeat 7 and 8.

Experiment E:

Load the following program at F000H.

<u>Mnemonic</u>	<u>Hex Code</u>	<u>Comment</u>
LD A,FFH	3E FF	Load A with FFH
INC A	3C	Increment A
JP F002H	C3 02 F0	Jump to F002H

10. Execute the program at full speed. What happened?
11. Execute the program in the single step mode (S command) and write down the contents of Register A after going through the program ten times.
12. What does the program do?

Experiment F:

13. Come up with a program of your own (one for each lab group - not a program that is listed above) and use the trainer to execute it. First, draw a flow chart of what you want to do. Then, write the mnemonics to realize it, and then translate the mnemonic code into hex code and enter it.
14. After you get your program to work, show it to the T.A., explaining what it does and verifying its operation.

Experiment G:

Transferring data to and from memory and the CPU is not the only way to communicate with a microprocessor. Communication with the "outside world" is done with I/O or input and output instruction. Input and output ports are used to implement these instructions. A port is essentially an 8-bit latch similar to an internal register except that it is accessible in some way from outside the microprocessor. For example the instruction: OUT 02,A sends the 8 bits of data from the accumulator to output port 02. I/O ports are addressed similarly to memory locations in RAM or ROM, except 8 bit addresses are used (rather than 16). Some typical uses for I/O ports are for keyboards (input), LED and CRT displays (output), and for many other types of microprocessor-based control systems.

The MPF-I+ has some I/O ports already defined on it. For example:

Port 80	7 segment display digits 1-8
Port 81	7 segment display digits 9-16
Port 92	LED and speaker (bit 5)

Information on other ports can be found in the user manual on pages 113-114.

In this experiment, you will generate a sound on the speaker and/or blink the LED. The LED will be "on" when a 0 is sent to port 92, bit 5. To generate a tone on the speaker, a square wave must be sent to it. This means sending a succession of 0's and 1's to the correct port.

15. Write a program to generate a tone to the speaker.
16. Does the LED blink? Why or why not?
17. Determine the frequency of the tone emitted by the speaker.
18. How could you raise or lower this tone?
19. Change the frequency (and, therefore, the pitch) of the tone.

Note: To achieve an audible tone, you may have to implement "delays" in your program. These are simply program loops which have no purpose but to use up some of the microprocessor's instruction cycle time. For example:

```
LOOP:    LD D,FFH
          DEC D
          JNZ LOOP
```