# Lab: 5
# Familiarization with miscellaneous problems

## Miscellaneous problems

8085 microprocessor do not have instructions for the multiplication and division. So some algorithm has to be devised for the multiplication and division using the available instructions. Similarly using 8085's instruction set it is possible to write programs to solve various types of problems.

a) Multiplying 8-bit numbers

The multiplication of two unsigned 8-bit numbers can be done in two ways a) repetitive addition or use of a register shifting operation.

Repetitive addition is the most straight forward method. For example, if we want to multiply 12 times 8 we have to add 12 to the initially zeroed accumulator 8 times.

Shift operation provides faster multiplication and division. Shifting a byte left one bit is equivalent to multiplying by 2, while shifting a byte right one bit is equivalent to dividing by 2.

The three parts of multiplication problems are:
MULTILICAND
MULTIPLIER
PRODUCT

E.g., 03 (multiplicand)
02 (multiplier)
06 (product)

Multiplying 8-bit number using repetitive addition

Note: Do not forget to initialize output ports if `out` instruction is used.

Enter the following program

```
8000 XRA A
8001 MVI B, 06
8003 ADI 25
8005 DCR B
8006 JNZ 8003
8009 OUT 40
800B RST 5
```

Run this program in full speed and see the output in the port A. What will happen if the value in the reg. B is changed? Modify the program when the result exceeds 8-bit.

Multiplying 8-bit numbers using shifting
Enter the following program

```
8010 MVI B, 00
8012 MVI C, 3C
8014 MVI D, 2A
8016 MVI E, 09
8018 MOV A, C
8019 RAR
801A MOV C, A
801B DCR E
801C JZ 8029
801F MOV A, B
8020 JNC 8024
8023 ADD D
8024 RAR
8025 MOV B, A
8026 JMP 8018
8029 MOV A, B
802A OUT 41
802C MOV A, C
802D OUT 42
802F RST 5
```

Run this program in full speed and record the output in the ports.
Make this program a subroutine and use this subroutine to multiply 54H and 3AH.

Single Precision Multiplication Routine for 8085

```
8000 MVI A, 54
8002 LXI D, 7432
8005 LXI H, 0000
8008 MVI B, O8
800A DAD H
800B RLC
```

```
800C JNC 8010
800F DAD D
8010 DCR B
8O11 JNZ 800A
8014 MOV A, H
8015 OUT 40
8017 MOV A, L
8018 OUT 41
801A RST 5
```

Run this program in full speed and note the output.
Change the program with different multiplier and multiplicand and verify the program.

b) Dividing 8-bit numbers

As multiplication division can also be performed using two methods: repetitive subtraction and shifting methods. However it is more difficult to implement the repetitive subtraction to perform the division. In repetitive subtraction we count the number of subtractions until zero is reached.

Division involves following terminologies.
DIVIDEND – number into which another number is divided
DIVISOR – number divided into another number
QUOTIENT – the number of times the divisor goes into the dividend
REMAINDER – the dividend minus the product of the divisor and the quotient.
Dividing 8-bit numbers by subtraction

```
8000 MVI C, 00                      800D JMP 8006
8002 MVI B, 0A                      8010 MOV E, B
8004 MVI D, 03                      8011 MOV A, C
8006 MOV A, B                       8012 OUT 40
8007 SUB D                          8014 MOV A, E
8008 JC 8010                        8015 OUT 41
800B MOV B, A                       8017 RST 5
800C INR C
```

Run this program and note down the quotient and remainder.
Can you write a program that implements the division using the shifting? Try to implement the division using the other techniques.

c) Miscellaneous problems

In this section, different problems are given so that student will be familiar with different types of the situations in programming 8085. At this point it is assumed that almost all of the instructions of 8085 microprocessor are already familiar.

Delay Loop:
Load the following program

```
8000 MVI A 01
8002 OUT 40
8004 INR A
8005 NOP
8006 NOP
8007 NOP
8008 JMP 8002
```

Run the program and run in full speed. Observe the output at port A.
Modify the above program as following to call the subroutine at 8050

```
8000 MVI A 01                     8050 MVI B 80
8002 OUT 40                       8052 MVI C 80
8004 INR A                        8054 DCR C
8005 CALL 8050                    8055 JNZ 8054
8008 JMP 8002                     8058 DCR B
                                  8059 JNZ 8052
                                  805C RET
```

Again run the program in full speed, and observe the output at port A. Compare the result with previous case.
Here the subroutine is used to make a delay in between two successive displays. The delay calculation is as follows:

| Delay Subroutine | No. of Repetition of Instruction | T-states | Total |
|---|---|---|---|
| 8050 MVI B 40 | 1 | 7 | 7*1 |
| 8052 MVI C 80 | 64 | 7 | 7*64 |
| 8054 DCR C | 128*64 | 4 | 4*128*64 |
| 8055 JNZ 8054 | (127 Jump, 1 No Jump)*64 | 7/10 | (10*127+7*1)*64 |
| 8058 DCR B | 64 | 4 | 4*64 |
| 8059 JNZ 8052 | (63 Jump, 1 No Jump) | 7/10 | 10*63+7*1 |
| 805C RET | 1 | 10 | 10*1 |
| Grand Total | | | 115854 |

Thus total 115854 clocks are required.

If the microprocessor is of 2 MHz, then each pulse has time period of ½ μ sec.

So total time taken for execution of subroutine = 115854/$(2 *10^6)$ sec
= 57.927 m sec

In this way delay loop of any interval can be created. The instruction NOP can also be used for delay adjustment.

**Assignments**

1. Write a program to make a clock. Use port A, B and C to display second, minute and hour respectively. The clock should be 24 hours basis.
2. Write a program that takes a number from memory address 9000H, and stores the multiplication table of the corresponding number in memory address from 9001H to 900AH. (The number at memory address 9000H will not exceed twenty-five.)
3. Write a program that takes a BCD number from memory location 8090H, and displays the multiplication table in a port at interval of two seconds (approximately). (Assume the number at address 8090H will not exceed nine).
   Let 8090H contains 05 then display 05 first and after 2 second display 10 and again after 2 seconds 15 and so on up to 50.
4. Sixteen bit data are stored in two tables starting at 8040H and 8060H having ten data each. Write an 8085 program to store the sum in the corresponding index of the third table starting at 8080H. (Assume the sum will not exceed 16 bit).
5. A table contains ten 8-bit data starting at 9000H. Write an 8085 program to transfer the data from this table to next table starting at 9020H if the number is greater than 40H and less than C0H, else store 00H.
6. A table contains ten 8-bit data starting at 8050H. Write an 8085 program to store the sum of odd numbers at 8060H and store sum of even numbers at 8070H. Also display the sum of even numbers at output ports after 2-3 seconds of displaying the sum of odd numbers.

**Optional Assignments:**

7. A table contains ten 8-bit numbers from memory location 9500H; write a program to arrange in ascending order.
8. Write a program at memory location FE00H, which resets all memory starting from 8000H to FE00H.
   Modify the above program to set 01H to all memory location.

**Write programs for each of the assignments. After completing the execution show the output to the instructor.**