

ECC 512 Assignment 2

Microprocessor And Microcontroller



August 4, 2023

CHANDNI JHA

Registration -690 | Roll-CSE/21030

# Github Link: <https://github.com/ChandniJha630/CSC512-MIcroprocessor-And-Microcontroller-Lab/tree/main/Assignment%202>

# Q1. 16 Bytes of data are stored in memory location from 8850H To 885FH. Transfer the entire block of data bytes to new memory location starting from 8870H

# Code

;<Program title>

jmp start

;data

;code

start: nop

LXI H,8850H

LXI B,8870H

MVI D, 10H

LOOP: MOV A,M

STAX B

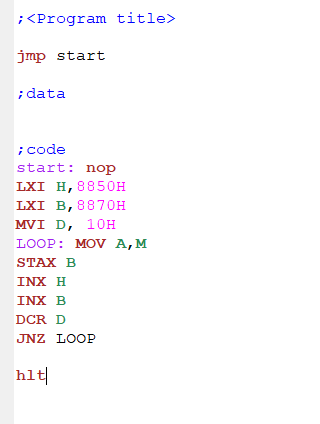
INX H

INX B

DCR D

JNZ LOOP

hlt



# ALGORITHM

1. `LXI H,8850H`: This is a "Load Immediate Extended" instruction that loads the 16-bit register pair `H` with the value `8850H`. The `H` register will have `88H` and the `L` register will have `50H`.

2. `LXI B,8870H`: This instruction loads the 16-bit register pair `B` with the value `8870H`. The `B` register will have `88H` and the `C` register will have `70H`.

3. `MVI D, 10H`: The "Move Immediate" instruction loads the `D` register with the immediate value `10H` (which is 16 in decimal).

4. `LOOP: MOV A, M`: The label `LOOP` is defined here, and the instruction `MOV A, M` moves the contents of the memory location pointed to by the `HL` register pair (i.e., the value stored at address `HL`) into the accumulator (`A`).

5. `STAX B`: The "Store Accumulator" instruction stores the value of the accumulator (`A`) into the memory location pointed to by the `BC` register pair (i.e., the value of `B` and `C` combined).

6. `INX H`: The "Increment Register Pair" instruction increments the value of the `HL` register pair, effectively pointing to the next memory location.

7.`INX B`: The "Increment Register Pair" instruction increments the value of the `BC` register pair, effectively pointing to the next memory location.

8. `DCR D`: The "Decrement Register" instruction decrements the `D` register by one.

9. `JNZ LOOP`: The "Jump if Not Zero" instruction checks the `Z` flag (Zero flag) of the status register. If the zero flag is not set (i.e., the result of the previous operation was not zero), the program jumps back to the `LOOP` label, and the process from step 6 to 10 is repeated. If the `Z` flag is set (i.e., `D` becomes zero), the program continues to the next instruction.

INPUT OUTPUT

# Screenshot 2023-08-04 010704.png Screenshot 2023-08-04 010742.png

# Q2. 16 Bytes of data are stored in memory location from 8850H To 885FH. Transfer the entire block of data bytes to new memory location starting from 8870H in Reverse Order

## Code

# 

;<Program title>

jmp start

;data

;code

start: nop

LXI H,885FH

LXI D,8870H

MVI B, 10H

LOOP: MOV A,M

STAX D

DCX H

INX D

DCR B

JNZ LOOP

hlt

# ALGORITHM

1. `LXI H, 885FH`: This is a "Load Immediate Extended" instruction that loads the 16-bit register pair `H` with the value `885FH`. The `H` register will have `88H`, and the `L` register will have `5FH`.

2. `LXI D, 8870H`: This instruction loads the 16-bit register pair `D` with the value `8870H`. The `D` register will have `88H`, and the `E` register will have `70H`.

3. `MVI B, 10H`: The "Move Immediate" instruction loads the `B` register with the immediate value `10H` (which is 16 in decimal).

4. `LOOP: MOV A, M`: The label `LOOP` is defined here, and the instruction `MOV A, M` moves the contents of the memory location pointed to by the `HL` register pair (i.e., the value stored at address `HL`) into the accumulator (`A`).

5. `STAX D`: The "Store Accumulator" instruction stores the value of the accumulator (`A`) into the memory location pointed to by the `DE` register pair (i.e., the value of `D` and `E` combined).

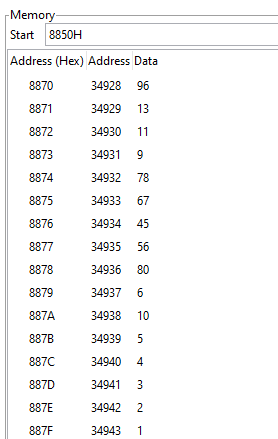
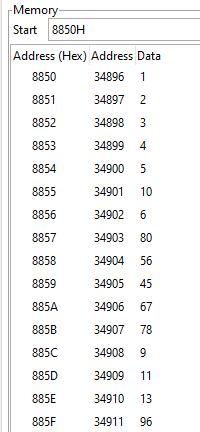
6. `DCX H`: The "Decrement Register Pair" instruction decrements the value of the `HL` register pair, effectively pointing to the previous memory location.

7. `INX D`: The "Increment Register Pair" instruction increments the value of the `DE` register pair, effectively pointing to the next memory location.

8. `DCR B`: The "Decrement Register" instruction decrements the `B` register by one.

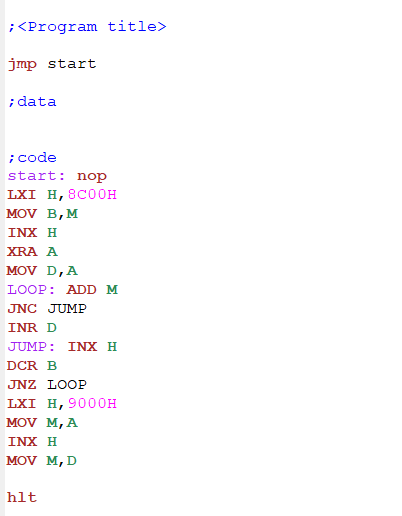
9. `JNZ LOOP`: The "Jump if Not Zero" instruction checks the `Z` flag (Zero flag) of the status register. If the zero flag is not set (i.e., the result of the previous operation was not zero), the program jumps back to the `LOOP` label, and the process from step 4 to 8 is repeated. If the `Z` flag is set (i.e., `B` becomes zero), the program continues to the next instruction.

# INPUT OUTPUT



# Q3. Write ALP that will add N numbers of Bytes starting at 8C01.Value Of N is stored in 8C00H. Store the result in 9000H and if carry is present store it at 9001H.

## Code



;<Program title>

jmp start

;data

;code

start: nop

LXI H,8C00H

MOV B,M

INX H

XRA A

MOV D,A

LOOP: ADD M

JNC JUMP

INR D

JUMP: INX H

DCR B

JNZ LOOP

LXI H,9000H

MOV M,A

INX H

MOV M,D

hlt

# ALGORITHM

1. `LXI H, 8C00H`: This is a "Load Immediate Extended" instruction that loads the 16-bit register pair `H` with the value `8C00H`. The `H` register will have `8CH`, and the `L` register will have `00H`.

2. `MOV B, M`: The "Move" instruction moves the value of the memory location pointed to by the `HL` register pair (i.e., the value at address `8C00H`) into the `B` register.

3. `INX H`: The "Increment Register Pair" instruction increments the value of the `HL` register pair, effectively pointing to the next memory location (`8C01H`).

4. `XRA A`: The "Exclusive OR Accumulator" instruction performs an XOR operation between the accumulator (`A`) and itself, effectively setting the accumulator to zero.

5. `MOV D, A`: The "Move" instruction moves the value of the accumulator (`A`) into the `D` register.

6. `LOOP: ADD M`: The label `LOOP` is defined here, and the "Add" instruction adds the value of the memory location pointed to by the `HL` register pair (i.e., the value at address `8C01H`) to the accumulator (`A`).

7. `JNC JUMP`: The "Jump if No Carry" instruction checks the carry flag (C flag) of the status register. If the carry flag is not set (i.e., there was no carry during the addition operation), the program jumps to the `JUMP` label. If there was a carry, the program continues to the next instruction.

8. `INR D`: The "Increment Register" instruction increments the value of the `D` register by one.

9. `JUMP: INX H`: The label `JUMP` is defined here, and the "Increment Register Pair" instruction increments the value of the `HL` register pair, effectively pointing to the next memory location.

10. `DCR B`: The "Decrement Register" instruction decrements the `B` register by one.

11. `JNZ LOOP`: The "Jump if Not Zero" instruction checks the zero flag (Z flag) of the status register. If the zero flag is not set (i.e., `B` is not zero), the program jumps back to the `LOOP` label, and the process from step 6 to 10 is repeated. If the zero flag is set (i.e., `B` becomes zero), the program continues to the next instruction.

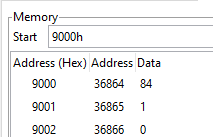
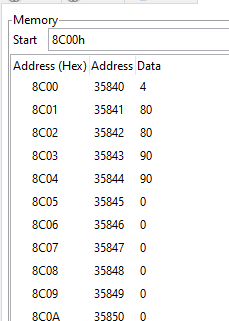
12. `LXI H, 9000H`: This is a "Load Immediate Extended" instruction that loads the 16-bit register pair `H` with the value `9000H`. The `H` register will have `90H`, and the `L` register will have `00H`.

13. `MOV M, A`: The "Move" instruction moves the value of the accumulator (`A`) into the memory location pointed to by the `HL` register pair (i.e., the value at address `9000H`).

14. `INX H`: The "Increment Register Pair" instruction increments the value of the `HL` register pair, effectively pointing to the next memory location (`9001H`).

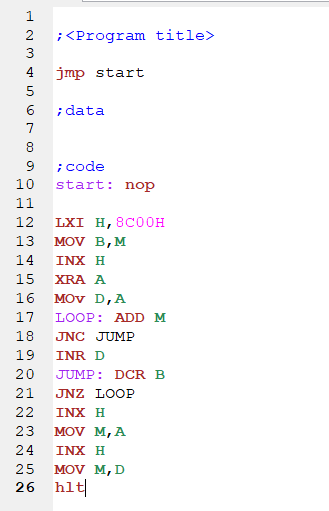
15. `MOV M, D`: The "Move" instruction moves the value of the `D` register into the memory location pointed to by the `HL` register pair (i.e., the value at address `9001H`).

# INPUT OUTPUT



# Q4. Two 8 bit numbers are stored in memory locations 8C00H and 8C01H. Perform Multiplication and store result in 8C02H

## CODE



;<Program title>

jmp start

;data

;code

start: nop

LXI H,8C00H

MOV B,M

INX H

XRA A

MOV D,A

LOOP: ADD M

JNC JUMP

INR D

JUMP: DCR B

JNZ LOOP

INX H

MOV M,A

INX H

MOV M,D

hlt

# ALGORITHM

1. The algorithm starts by initializing the **HL** register pair with **8C00H**, which is the memory location from where the data will be read.
2. The value at memory location **8C00H** is moved to the **B** register.
3. The **HL** register pair is incremented to point to the next memory location (**8C01H**).
4. The accumulator **A** is XORed with itself, effectively setting it to zero.
5. The value in the accumulator **A** (which is zero) is moved to the **D** register.
6. The algorithm enters a loop (**LOOP**) where the value at the current memory location pointed to by **HL** is added to the accumulator **A**.
7. If there is no carry during the addition (JNC JUMP), it increments the **D** register by one.
8. The algorithm then decrements the **B** register by one (**DCR B**) and checks if **B** is not zero (**JNZ LOOP**). If **B** is not zero, it jumps back to the **LOOP** label and repeats the addition and increment process.
9. Once the **B** register becomes zero, the algorithm increments the **HL** register pair to point to the next memory location (**8C02H**).
10. The value in the accumulator **A** is moved to the memory location pointed to by the **HL** register pair (**8C02H**).
11. The **HL** register pair is incremented again to point to the next memory location (**8C03H**).
12. The value in the **D** register is moved to the memory location pointed to by the **HL** register pair (**8C03H**).

# INPUT - OUTPUT

