Lab 4: Familiarization with Branching Operations

Branching Operations: Branching is the process of transferring the program control to somewhere else instead of executing next instruction. The branching instructions change the content of the program counter to execute instruction somewhere else.

a) Jump instructions (`JMP, Jx, PCHL`)

Jump instructions are used to transfer the control of the program to some other location instead of the next instruction. These instructions are used as follows

`JMP 16-bit`	Unconditional Jump	`JM 16-bit`	Jump on minus
`JNZ 16-bit`	Jump on no zero	`MP 16-bit`	Jump on plus
`JZ 16-bit`	Jump on zero	`JPE 16-bit`	Jump on parity even
`JNC 16-bit`	Jump on no carry	`JPO 16-bit`	Jump on parity odd
`JC 16-bit`	Jump on carry		

`JMP` instruction is the unconditional jump and `Jx` instruction is the conditional jump. Conditional jumps use the flag conditions for the branching.
`PCHL` instruction copies the content of the H reg. pair into PC, i.e., this command branches the control to the location specified by H reg. pair.
Looping is done with the conditional jump instruction. When we have to insert delay we can use the loops for the delay.

Example 1: Load and run the following program

```
8000 MVI A, 80H
8002 OUT 43H
8004 MVI A, 01
8006 OUT 40H
8008 RLC
8009 NOP
800A NOP
800B NOP
800C JMP 8002
800F RST 5
```

Run this program in single step mode and note the output in port A and note the sequence of the execution of the instructions. Will the program terminate?

Now insert a delay loop and run the program in full speed (Hint: Replace NOP to jump to a delay loop).

Example 2: Load the following program

```
8050 MVI A, 80
8052 OUT 43
8054 MVI A, FF
8056 LXI H, 8080
8059 PCHL
805A RST 5

8080 DCR A
8081 OUT 40
8083 JMP 8080
```

Run this program in single step mode and see what happens when `PCHL` and `JMP` instruction is executed.

Assignment

1. Write a program to count the no of bits that are 1 in register A.
2. Write a program to add nos. from one to fifty and display the 16 bit result at output ports.
3. Write a program that will count up from 00 to FF at port A. Be sure to use `PCHL` command.

b) Call and Return instructions

Like JUMP command, CALL command changes the normal sequence of executing instructions. The objective is to have the computer go off and execute a series of program steps, called subroutine. Unlike the JUMP command, the CALL causes the computer to remember where it used CALL, so it can go back to the main program when it finds a RET command during execution. Stack is automatically accessed in call and return instructions.

The call instructions are used as follows.

`CALL 16-bit`	Unconditional Call	`CM 16-bit`	Call on minus
`CNZ 16-bit`	Call on no zero	`CP 16-bit`	Call on plus
`CZ 16-bit`	Call on zero	`CPE 16-bit`	Call on parity even
`CNC 16-bit`	Call on no carry	`CPO 16-bit`	Call on parity odd
`CC 16-bit`	Call on carry		

The return instructions are used as follows.

`RET`	Unconditional Return	`RM`	Return on minus
`RNZ`	Return on no zero	`RP`	Return on plus
`RZ`	Return on zero	`RPE`	Return on parity even
`RNC`	Return on no carry	`RPO`	Return on parity odd
`RC`	Return on carry		

`CALL` instruction is the unconditional call and C`x` instructions are the conditional calls. Similarly `RET` instruction is the unconditional return and `Rx` instruction is the conditional return. Conditional call and return use the flag conditions for the branching and return.

Example 3: Load and verify the following program

```
8000 MVI A, 80          8010 INR A              8020 ADI 03
8002 OUT 43             8011 CALL 8020          8022 RET
8004 MVI A, 01          8014 RET
8006 CALL 8010
8009 OUT 40
800B RST 5
```

Run this program and note down the sequence of the execution of the instructions. What is the output at port `40H`? Note down the SP content before and after the execution of the `CALL` and `RET` instructions also observe the stack content.

Conditional calls are useful if the call is to be occurred when some condition is satisfied. The conditional calls occur depending upon the flag conditions.

Example 4: Load the following program

```
8000 MVI A, 80          8020 MVI A, FF          8030 MVI A, 01
8002 OUT 43             8022 OUT 40             8032 OUT 41
8004 LDA 8050           8024 RET                8034 RET
8007 CPI 01
8009 CZ 8020
800C CNZ 8030
800F RST 5
8050 FF
```

Run this program and examine where the jump occurs (at 8020 or at 8030). Change the data at 8050 to 01 and see where the jump occurs. In the above two cases what output do you see in the port.

Conditional return instructions are used in returning from the subroutines when some condition occurs.

Example 5: Load the following program

```
8000 MVI A, 80          8020 LXI B, FFFF
8002 OUT 43             8023 DCR B
8004 MVI A, 01          8024 JNZ 8023
8006 OUT 40             8027 DCR C
8008 CALL 8020          8028 RZ
800B RLC                8029 JMP 8023
800C JMP 8002
```

Run this program in full speed and explain what is happening.

Assignments

4. Write a program to transfer the data at 8020 to 8030 if the data is greater than 127. You can assume data yourself.
5. Write a program to rotate the data 3C in a port. Call a delay subroutine for the visible output.
6. Write a program that will check whether the bit D_6 of a number stored at 4123 is 0 and its bit D_3 is 1. If the condition satisfies display the number.
7. Write a program that will check whether the number in reg. B is even or not. If the number is even display it in a output port.

c) Monitor Routines Accessible to User

MPS-85 monitor offers several routines that can be called by user programs.

List of some subroutines accessible to user and their functions:

Calling Address	Functions
0440H	**Update Address field** of the display. The contents of the locations 8FEFH & 8FF0H are displayed in the address field. The contents of all the CPU registers and flags are affected. Reg. B=1→ dot at the right edge of the field; B=0→ no dot.
044CH	**Update Data field** of the display. The contents of the location 8FF1H are displayed in the data field. The contents of all CPU registers and flags are affected. If Reg. B=1, dot at the right edge of the field; if B=0, no dot.
0389H	**Output characters** to display. The parameters for this routine are as follows: Reg A=0→ Use address field Reg A=1→ Use data field Reg B=1→ Dot at the right edge of the filed. =0→ No dot. Reg HL=Starting address of character string to be displayed.
02BEH	**Clear the display**. This routine blanks the entire display field. Parameter is: Reg B=1→ dot at the right edge of the address field. =0→ No dot.
03BAH	**Read keyboard**. This routine waits until a character is entered from the system keyboard and upon return, it places the character in the A register. The register A and F/F's are affected.

(**Note**: It is recommended to save the registers of interest before calling the monitor routines and restore them after returning from the monitor routines).

Example 6: Load the following program and run it. Observe the output by pressing a key in keyboard.

```
8000 CALL 03BAH                ;Call subroutine to read keyboard
8003 MVI B,00H                 ;No dot
8005 STA 8FF1H                 ;Store key-code from register A to memory
8008 CALL 044CH                ;Call subroutine to display content from memory
800B JMP 8000H
```

Run the above program by changing second instruction as: `MVI B, 01H`. Compare the result with that of above program.

Example 7: Load the following program and observe the output by running it.

```
8800  MVI A, 00                  ;Use address field
8802  MVI B,00                   ;No dot
8804  LXI H, 8840H               ;Use character string, starting at location 8840 H
8807  CALL 0389H                 ;Call subroutine to display "FIrE"
880A  MVI A,01                   ;Use data field
880C  MVI B,00                   ;No dot
880E  LXI H,8844H                ;Character string starts at 8844 H.
8811  CALL 0389H                 ;Call subroutine to display blanks in data field
8814  CALL 8831H                 ;Introduce a delay
8817  MVI A,00                   ;Use address field
8819  MVI B,00                   ;No dot
881B  LXI H, 8846H               ;display "HELP" in
881E  CALL 0389H                 ;address field
8821  MVI A,01                   ;Use data field
8823  MVI B,00                   ;No dot
8825  LXI H,884AH                ;Message start
8828  CALL 0389H                 ;Display "US" in data field
882B  CALL 8831H                 ;Introduce a delay
882E  JMP 8800H                  ;Repeat the sequence
8831  LXI D, FFFFH               ;Delay subroutine
8834  DCX D
8835  MOV A,D
8836  ORA E
8837  JNZ 8834H
883A  RET

8840  0F 13 14 0E 16 16 10 0E 11 12 15 05        ;Data
```

Run the above program by changing the display content as well as delay. Change the display content as:

```
8840  12 11 0E 0A 05 0E 10 0E 11 12 15 05
```

Also observe by changing the data as:

```
8840  16 16 16 16 16 16 10 0E 11 12 15 05
```