Expt No 4 DCMP Lab (Interfacing Logic Controller Card with 8085 Microprocessor Kit)

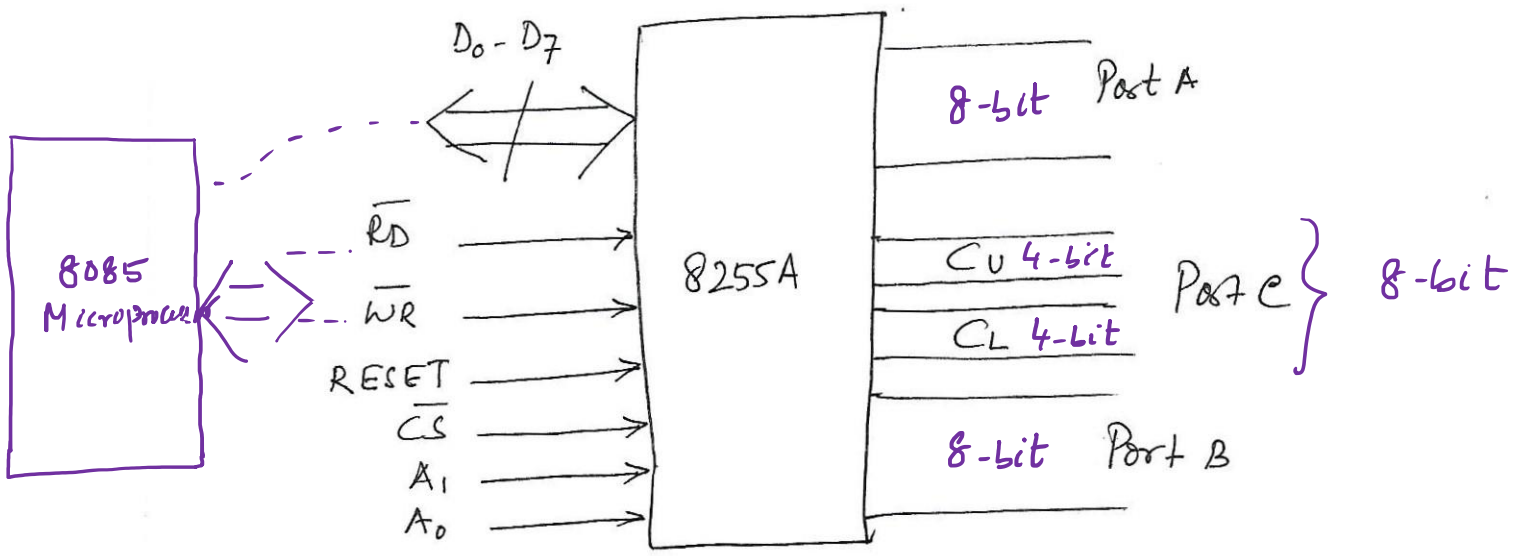
8085 kit has an 8255 IC for digital I/O

8255A Programmable Peripheral Interface

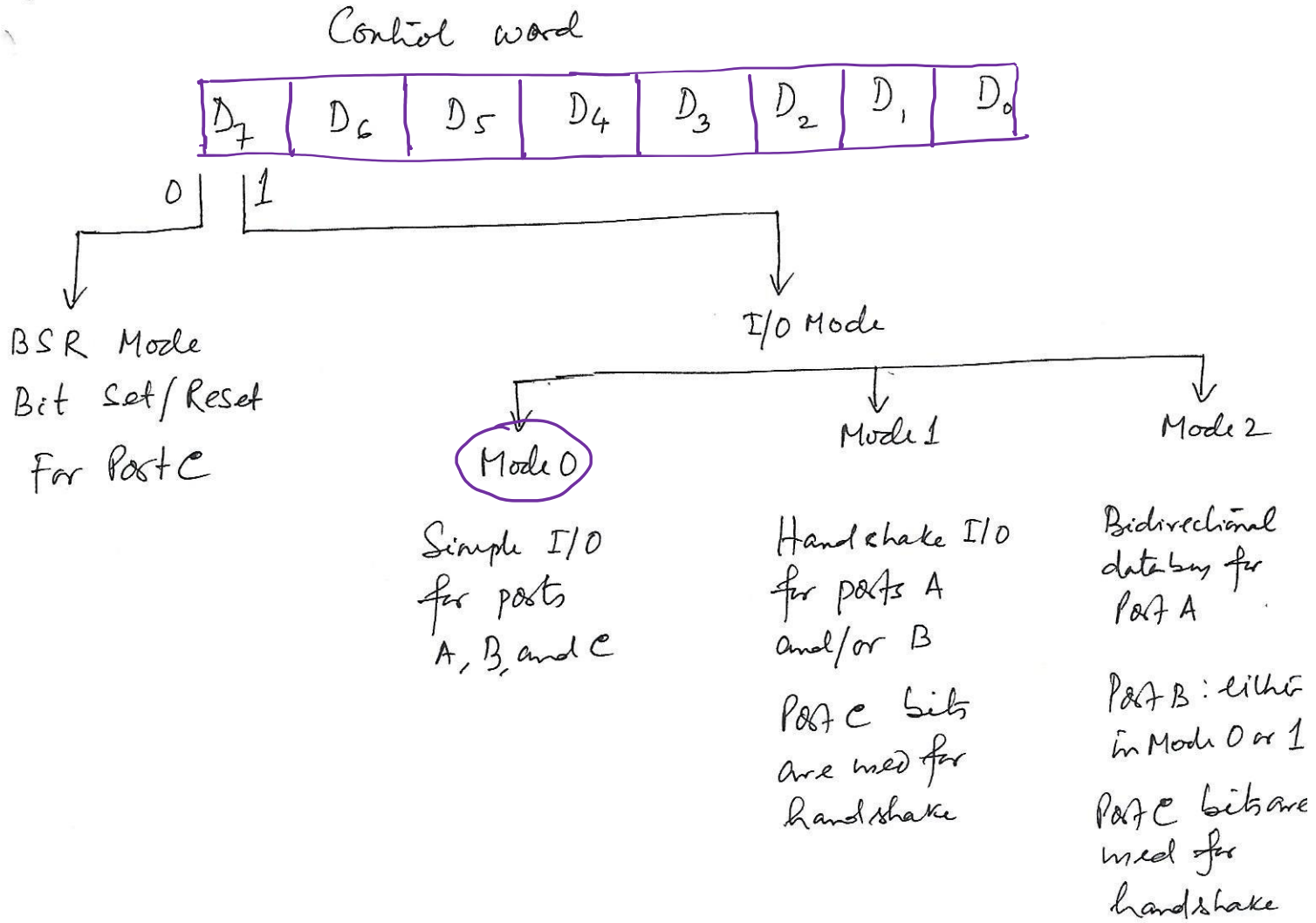
- a widely used programmable parallel I/O device
- Simple I/O
- interrupt I/O
- Can be used with any microprocessor.

24 I/O pins. 24-bits (Input / Output)

- Two 8-bit parallel ports A and B.
- 8 bits of Port C can be used as individual bits or be grouped in two 4-bit ports: C_{UPPER} (C_U) and C_{LOWER} (C_L)
  (4-bit, 4-bit)
- Functions of these ports are defined by writing a control word in the control register.



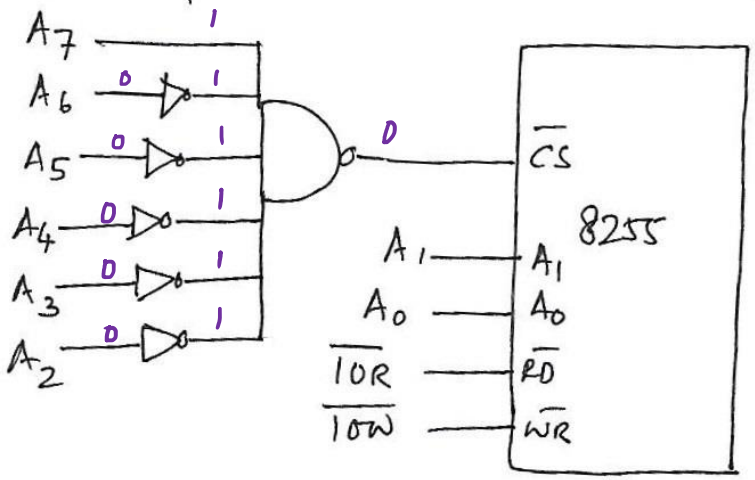
$\overline{CS}$	A_1	A_0	Selected
0	0	0	Port A ✓ — 8-bit
0	0	1	Port B ✓ — 8-bit
0	1	0	Port C ✓ — 8-bit
0	1	1	Control Register ✓ — 8-bit Register → Control Word (8-bit)
1	X	X	8255A is not selected


Control word
D7
D6
D5
D4
D3
D2
D1
D0
0
1
BSR Mode
Bit Set/Reset
For Port C
I/O Mode
Mode 0
Simple I/O
for ports
A, B, and C
Mode 1
Handshake I/O
for ports A
and/or B
Port C bits
are used for
handshake
Mode 2
Bidirectional
data bus for
Port A
Port B: either
in Mode 0 or 1
Port C bits are
used for
handshake

8255A Chip select logic

Example:

From 8085 Address

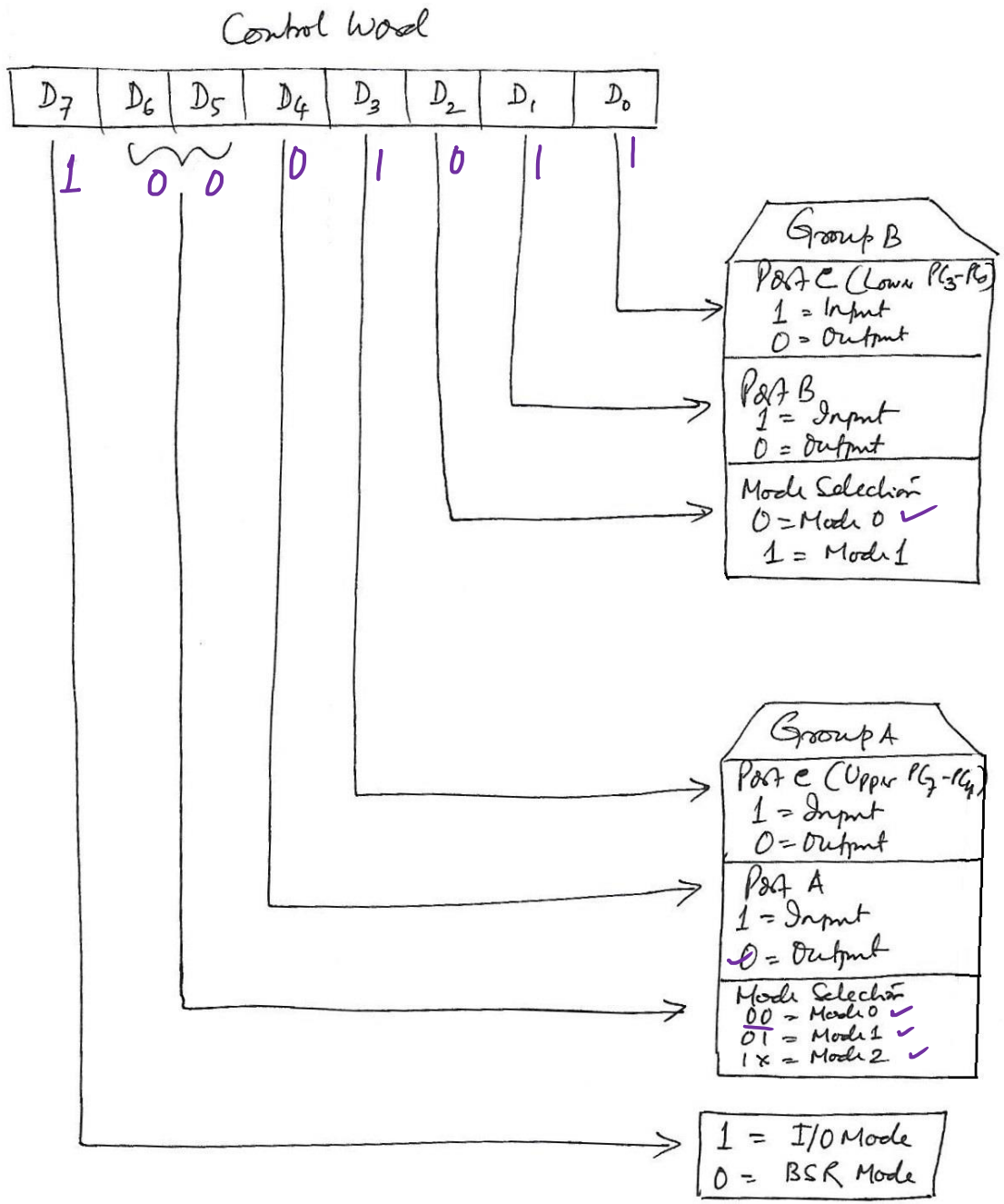


$\overline{CS}$		Hex Add	Port
A_7 A_6 A_5 A_4 A_3 A_2	A_1 A_0		
1 0 0 0 0 0	0 0	80H	A
-do-	0 1	81H	B
-do-	1 0	82H	C
-do-	1 1	83H	Control Reg.

8085 Kit → Logic Controller Card

- 40H → Port A
- 41H → Port B
- 42H → Port C
- 43H → Control Register

Control Word



8BH

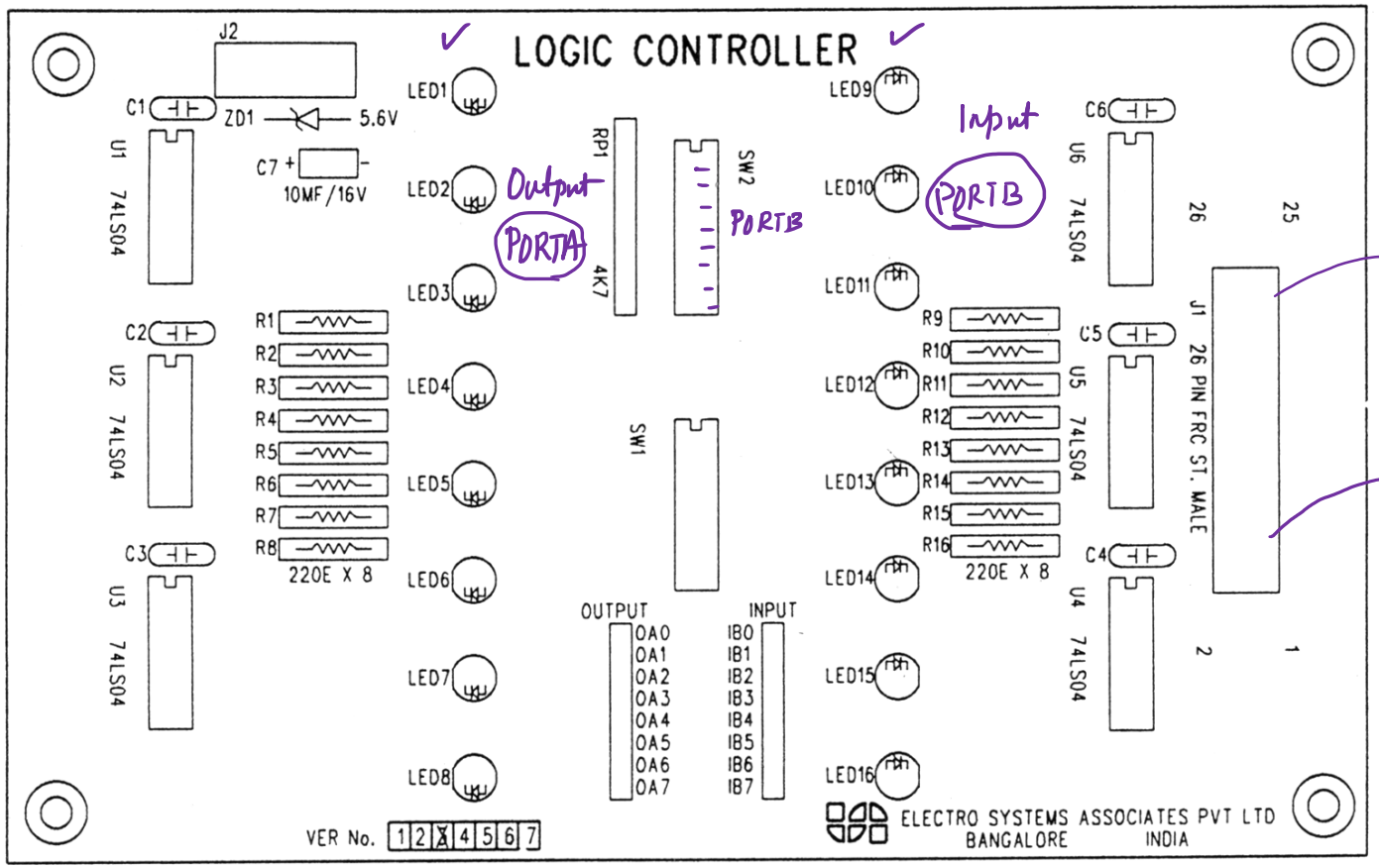
Port B + Port C Lower(4)

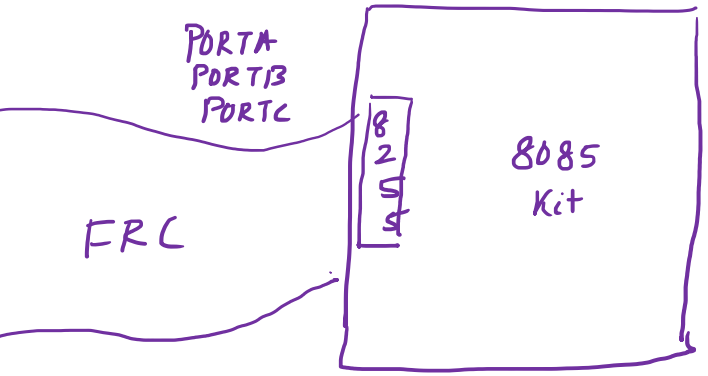
Port A + Port C Upper(4)

EXPERIMENT 4: DIGITAL I/O EXPERIMENTS USING THE LOGIC INTERFACE CARD OF THE 8085 MICROPROCESSOR KIT

PART A: LOGIC INTERFACE CARD

The MPS 85-3 8085 Microprocessor kit has a digital I/O interface connected via the on-board 8255 PPI. 8255 ports are accessible through a 26-pin FRC connector. We will use the Logic Controller Interface Card provided with the kit to do some basic I/O operations under program control. Following are the major features of this Logic Controller board.





✓PORTA → Output Port (8)

✓PORTB → Input Port (8)
SW2

(i) The board has 8 digital input and 8 digital output lines, whose status can be seen using LEDs on the Logic board. Each of the eight digital input lines can be set to either '0' or '1'.

(ii) The Logic board is physically connected to Port A of 8255 (through buffers for protection). It is assumed that Port A is initialized as Mode 0, Output port. Thus at any instant, LED1 to LED8 show the bits PA0 to PA7, respectively. LED1 is located at the Left-top side and LED8 at the Left-bottom side.

(iii) Similarly, Port B of 8255 is physically connected to the eight digital lines of the DIP Switch 2 (SW2) through a buffer. 'ON' position of each switch is taken to be '1' and OFF to be '0'. At any instant the eight LEDs, viz. LED9 to LED16 show the conditions of the eight lines at the Port B input, PB0 to PB7, respectively. Thus, LED9 will show the status of PB7, while LED16 that of PB0.

Port C is NOT used for this expt. Kept in the input mode.

(iv) 8255 Port adresses: Port A : 40H ; Port B : 41H ; Port C : 42H ;
Control Reg : 43H

The above features are good enough to perform a variety of digital I/O experiments using the 8085 kit.

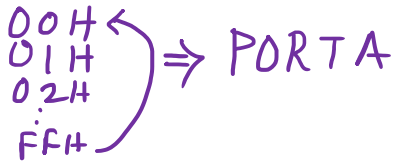
PART B: DIGITAL I/O OPERATIONS USING THE LOGIC INTERFACE CARD

Note: In all your programs initialize Port A as Mode 0 Output port, Port B as Mode 0 Input port, and Port C as input port (even though Port C is not used, initialize it as input port). Use the following code for this purpose: MVI A,h'8B ; OUT h'43.

Program 1: SIMPLE I/O OPERATIONS

Write a simple program (either by hand coding or by assembly etc using the PC) which will read the bits you set using the SW2 DIP switch on the board to Port B and output the same to Port A. Put the program in an infinite loop such that any changes to any of the input bits (these bits are displayed on LED9 to LED16) are immediately sent to the output (LED1 to LED8).

Program 2: GENERATION OF DIFFERENT PATTERNS



(i) Write a DELAY subroutine which generates 0.5 second delay.

(i) Implement an 8-bit binary UP counter in software and display the contents of the counter on Port A. Use the above DELAY subroutine in between the output states.

(i) Modify the program such that the binary UP counter counts up first from 00H to FFH and then it counts down to 00H. Repeat the above sequence using a DELAY of 1 sec.

SIMPLE INPUT OUTPUT OPERATION

```
MVI A, 8BH
OUT 43H
```

} Configure the Control Word ← 43H H'43

```
LOOP: IN 41H
      OUT 40H
      JMP LOOP
```

← Reading PORTB A ← PORTB

← Writing to PORTA A → PORTA

SUBROUTINE TO GENERATE A DELAY OF 0.5 SEC

OF = 4T

OF424H H'F424

```
DELAY:  LXI H, Data16      HL ← Data16 = N
LOOP1:  DCX H              6T      OF-6
        MOV A,L            4T
        ORA H              4T
        JNZ LOOP1          10T/7T
        RET
```

NOP 4T

8085 – 6MHz Crystal

$\frac{6MHz}{2} = 3MHz$ – Clock of 8085

$T = \frac{1}{f} = \frac{1}{3\times10^6}$

$24T \times N = 0.5$

$N = \frac{0.5}{24T} = \frac{0.5 \times 3\times10^6}{24} = 62500$

$\equiv F424H$

Program 3: <u>SOFTWARE IMPLEMENTATION OF 4-BIT SERIAL-PARALLEL MULTIPLIER</u>

Write a program which performs serial-parallel multiplication of two 4-bit numbers. Your program should read the 4-bit numbers set using the SW2 dip switch (Use switches 1 to 4 to specify number X, with switch 1 as the MSB. Use switches 5 to 8 to specify number Y, with switch 5 as the MSB). Display the result on LEDs 1 to 8 (Port A).

RAL	Rotate A Left through CY
RAR	Rotate A Right through CY

2 4-bit Numbers are given through PORT-B

8-bit Result is displayed through PORT A

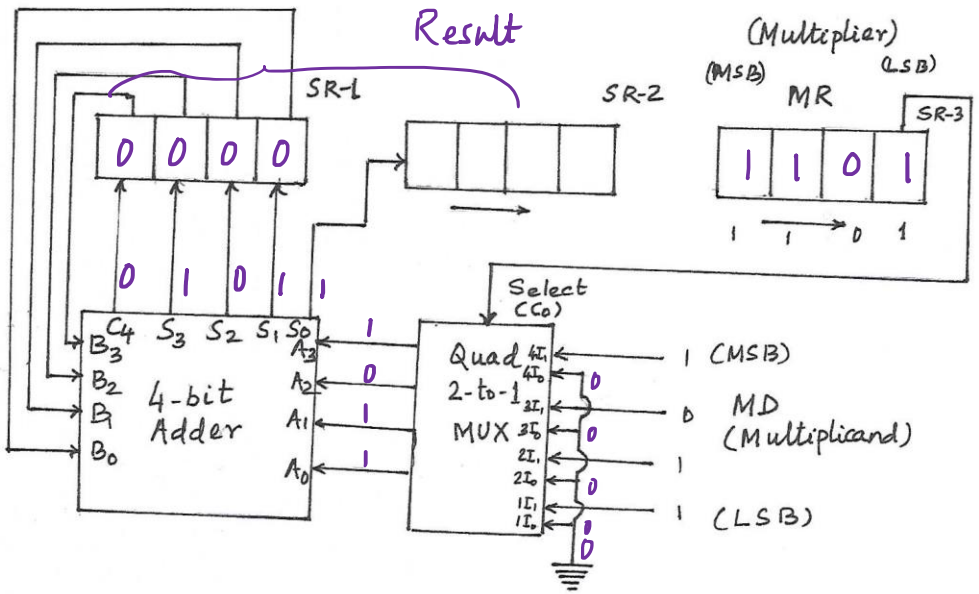
ADD Reg	Add Register to A
ADI Data	Add 8-bit Data to A

$A \leftarrow A + Reg$

$A \leftarrow A + Data$

JNZ Addr16	Jump to Addr16 if Zero flag is 0
JZ Addr16	Jump to Addr16 if Zero flag is 1
JNC Addr16	Jump to Addr16 if CY flag is 0
JC Addr16	Jump to Addr16 if CY flag is 1

Serial - Parallel Multiplier



Decimal: 11 — MD ✓

Decimal: 13 — MR ✓

```
        1 0 1 1      MD
    X   1 1 0 1      MR
  -------------
        0 0 0 0
      1 0 1 1
  -------------
      0 1 0 1 1
      0 0 0 0
  -------------
    0 0 1 0 1 1
    1 0 1 1
  -------------
  0 1 1 0 1 1 1
  1 0 1 1
  -------------
1 0 0 0 1 1 1 1
```

Place MD at MUX input

Load MR to SR-3

SR-1 is in Load Mode

SR-2 is in Right Shift Mode

Decimal = 143 ≡ 8FH