Embedded System Lab

(ELC3930)

Experiment No.: 05

Object:

Write an 8085 program to generate square, triangular, and sawtooth waveform using DAC card.

G. No: GL3136

S. No: A3EL-02

F. No: 19ELB056

Name: Maha Zakir Khan

Date of performing experiment: 14|02|2022

Date of report submission: 27|02|2022

Simulator Used:

8085 Simulator by Jubin Mitra. It helps in getting started easily with example codes, and to learn the architecture playfully. This tool is an integrated software environment for teaching microprocessor concepts. The software is shared under opensource GNU license.

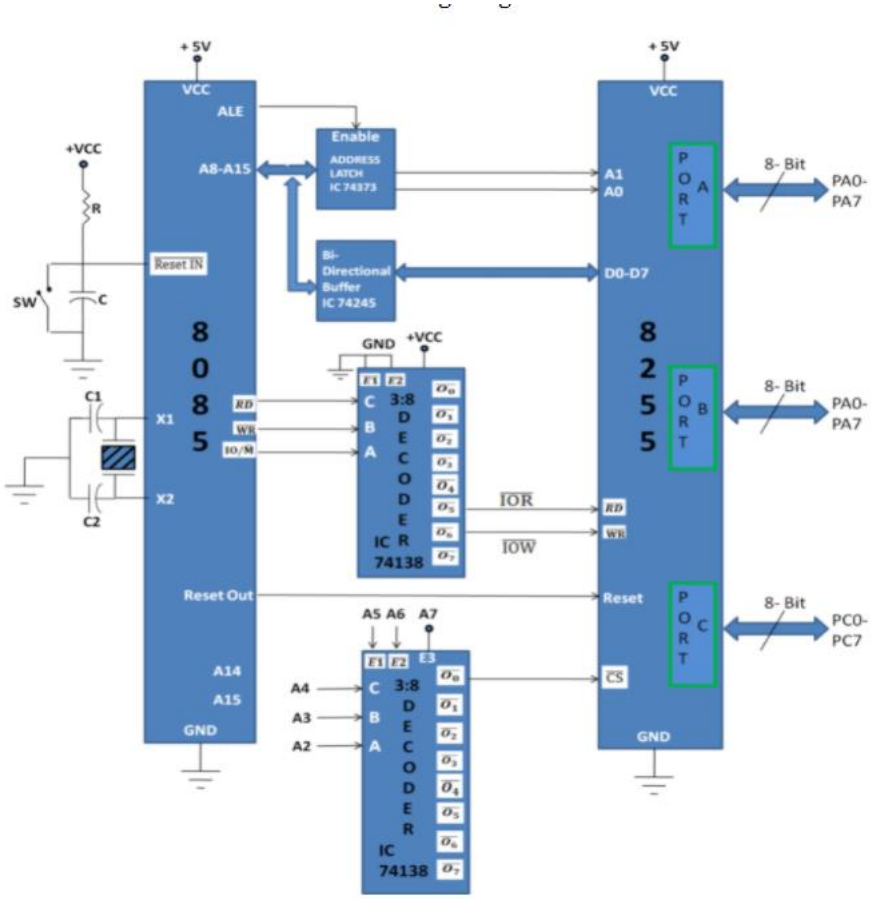
Link: 8085 Jubin Simulator

Algorithm:

To generate waveform using DAC card, we first need to interface 8085 with 8255 programmable I/O device to have access to the DAC. Given below are the steps to interface 8255 with 8085 microprocessor

Interfacing 8255 with 8085:

Interfacing Diagram of 8255



Step 1:

Lower order of 8-bit address A0-A7 is separated from AD0-AD7 using address latch/buffer (Ex: IC 74373) and ALE signal.

The separated address lines A0-A7 are connected to A0-A7 input pins of 8255 and the separated data bus D0-D7 are connected to D0-D7 pins of 8255.

Reset out of 8085 is connected to reset pin of 8255.

Step 2:

8255 does not have internal (separate) control logic generator, hence the IO/M(bar), RD(bar) and WR(bar) control signals are not connected directly to 8255. These pins are 1st given to decoder and decoded using 3:8 decoder (Ex: IC 74138).

The generated control signals IOR(bar) and IOW(bar) are connected to RD(bar) and WR(bar) input of 8155.

Step 3:

An active low signal of chip select logic is obtained decoding remaining address lines of lower order addresses A2- A7.

Chip select logic and IO port address for this interfacing circuit are as:

Chip select address lines	Address lines to select port	HEX address	Selected I/O						
A7	A6	A5	A4	A3	A2	A1	A0		
1	0	0	0	0	0	0	0	80H	PORT A
1	0	0	0	0	0	0	1	81H	PORT B
1	0	0	0	0	0	1	0	82H	PORT C
0	0	0	0	0		1	1	03H	Chip select register

Interfacing DAC (IC 0808) with 8255

Follow the initial 3 steps of interfacing of 8255 with 8085 that are given before.

The DAC 0808 is 8-bit digital to analog convertor IC. It converts digital data into equivalent analog current.

Therefore, I to V converter is used to convert analog output current of DAC to equivalent analog voltage.

PA0-PA7 pins of Port A are connected to D0-D7 pins of DAC.

In DAC given below dual power supply of +/- 10V is applied with reference voltage 10V as shown in diagram.

According to theory of DAC, Equivalent analog output is given as:

V0=Vref
Ex:
1. If data =00H [00000000], Vref= 10V
V0= 0 Volts.

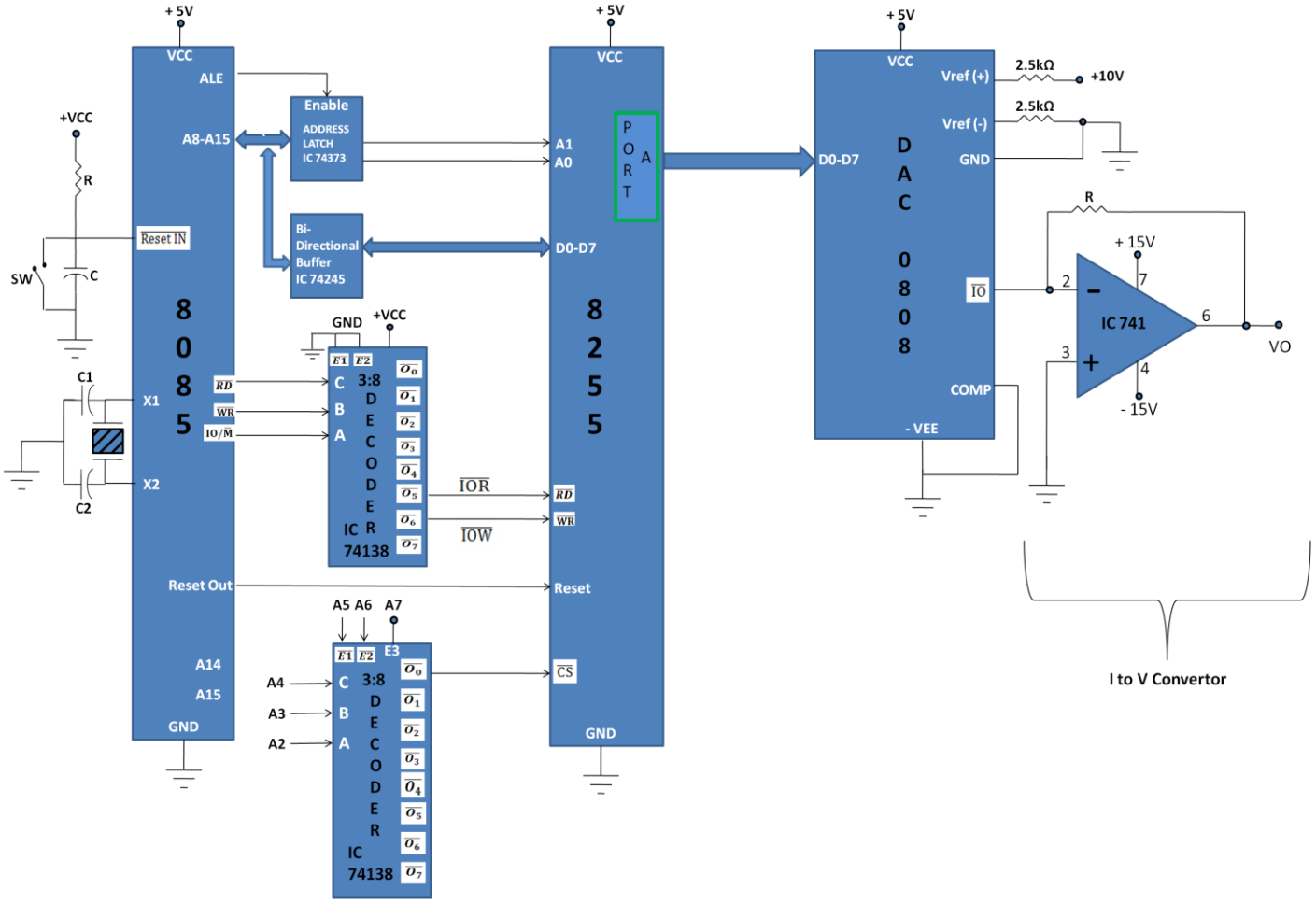
2. If data is 80H [10000000], Vref= 10V
V0= 5 Volts.

The control word format of 8255 for above interfacing is given as:

D7	D6	D5	D4	D3	D2	D1	D0
IO/BSR	MA	PA	PCU	MB	PB	PCL	
1	0	0	0	0	0	0	0

=80H

Interfacing Diagram of DAC



For Square Waveform:

Step 1: Load Accumulator with 00H

Step 2: Send Content of Accumulator to port A

Step 3: Call Delay

Step 4: Load Accumulator with FFH

Step 5: Call Delay

Step 6: Goto Step 1

For Saw-tooth Waveform:

Step 1: Load Accumulator with 00H

Step 2: Send Content of Accumulator to port A

Step 3: Increment Accumulator by 1 bit

Step 4: If Contents of Accumulator = FFH Goto Step 1, otherwise Goto Step 2

For Triangular Waveform:

Step 1: Load Accumulator with 00H

Step 2: Send Content of Accumulator to port A

Step 3: Increment Accumulator by 1 bit

Step 4: If Contents of Accumulator = FFH Goto Step 5, otherwise Goto Step 2

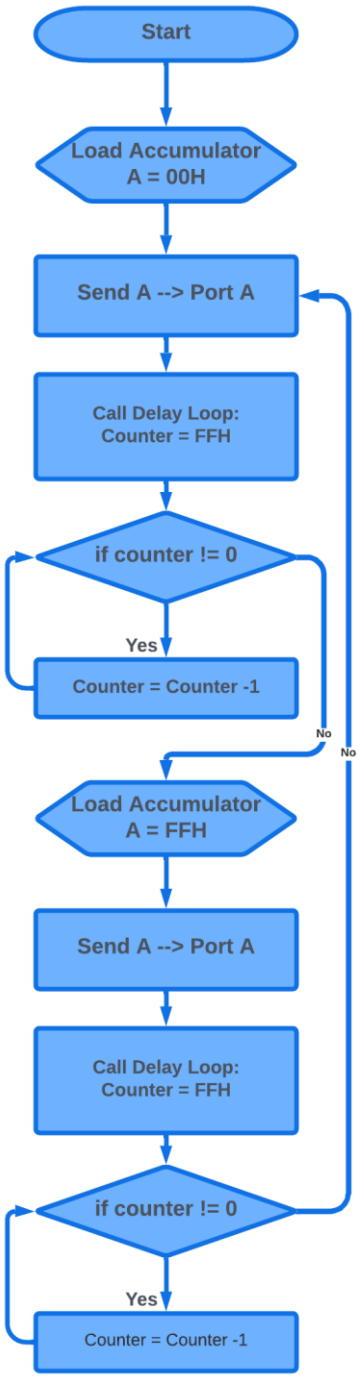
Step 5: Send Content of Accumulator to port A

Step 6: Decrement Accumulator by 1 bit

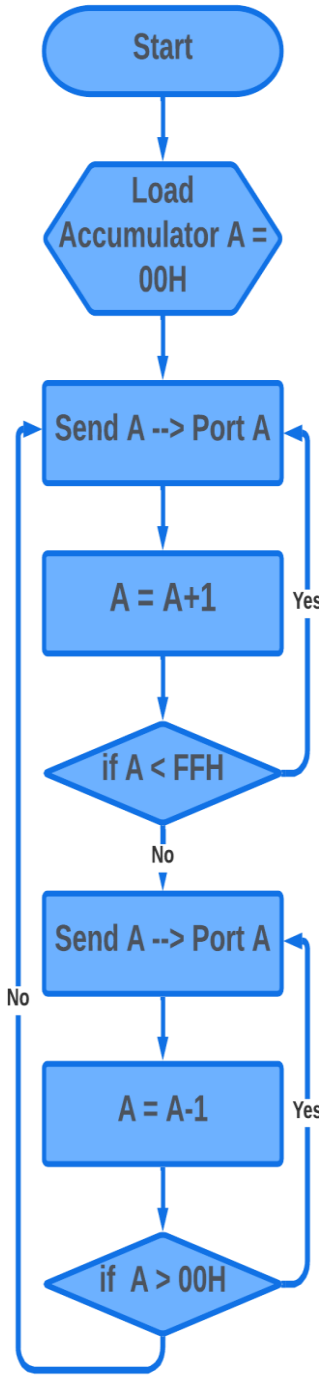
Step 7: If Contents of Accumulator = 00H Goto Step 2, otherwise Goto Step 5

Flow Chart:

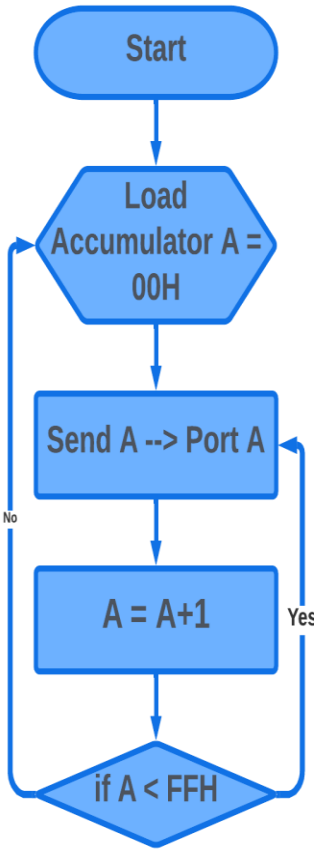
For Square Waveform:



For Triangular Waveform:



For Sawtooth Waveform:

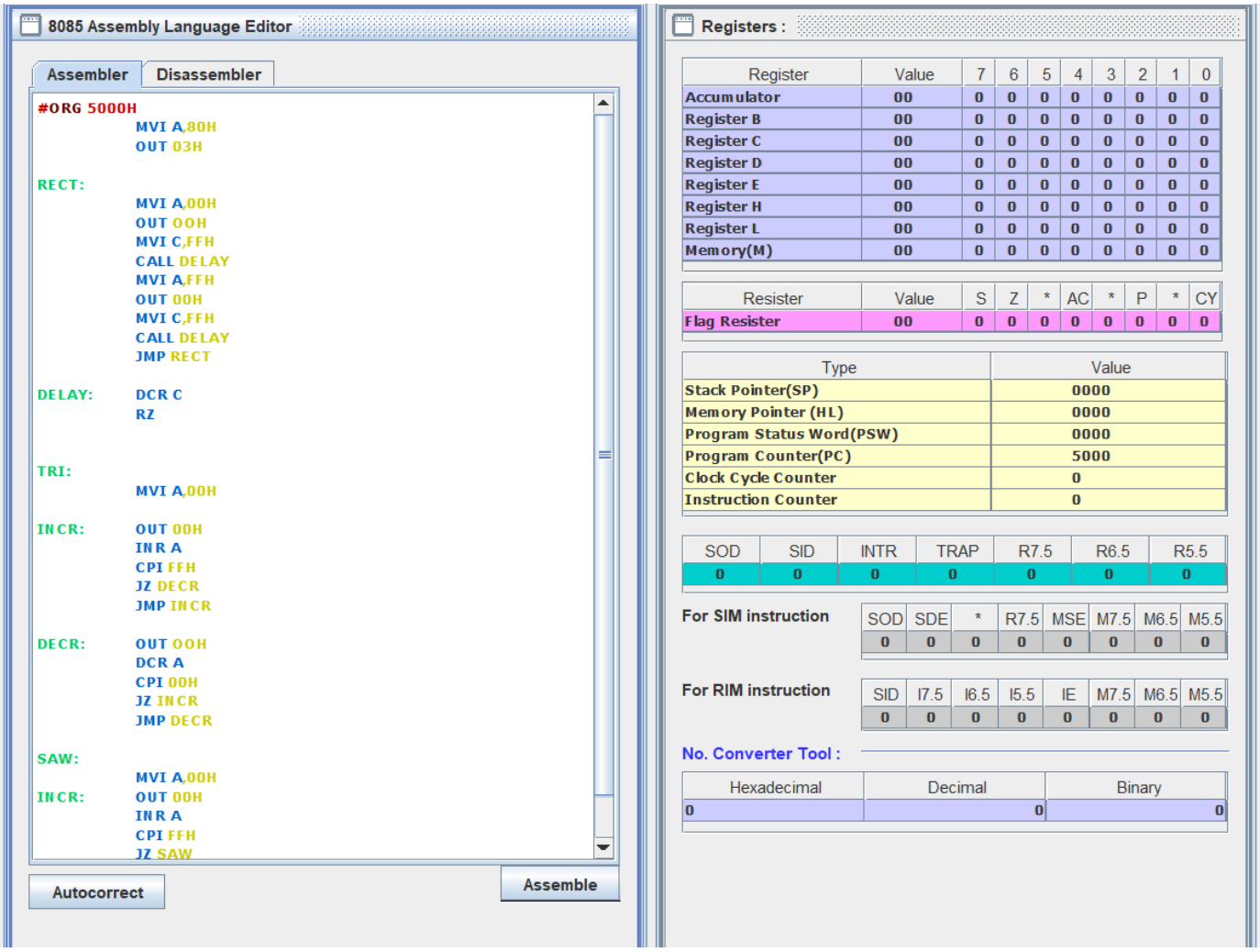


Program:

```
1.  # ORG 5000H
2.    MVI A,80          // Load accumulator with CW to set port A as output port
3.    OUT 03            // Control word send to DAC
4.
5.  // Block generates Square Waveform
6.  RECT:   MVI A,00
7.          OUT 00      // Send Low signal to Port A
8.          MVI C,FF
9.          CALL DELAY
10.         MVI A,FF    // Amplitude of the Wave
11.         OUT 00      // Send High signal to Port A
12.         MVI C,FF
13.         CALL DELAY
14.         JMP RECT
15. DELAY:  DCR C       // Delay block decrements counter C until it is 0
16.          RZ         // Return to RECT block when C = 0
17.          JMP DELAY
18.
19. //TRI block generates Triangular Waveform
20. TRI:    MVI A,00
21. INCR:   OUT 00      // INCR block increments Accumulator till it reaches FFH
22.         INR A
23.         CPI FF
24.         JZ DECR
25.         JMP INCR
26.
27. // DECR block decrements content of Accumulator till it reaches 00H
28. DECR:   OUT 00
29.         DCR A
30.         CPI 00
31.         JZ INCR
32.         JMP DECR
33.
34. // SAW block generates Sawtooth Waveform
35. SAW:    MVI A,00
36.
```

```
37. // SAWINCR increments content of Accumulator till FFH, then it resets the accumulator to
    00H
38. SAWINCR:  OUT 00
39.           INR A
40.           CPI FF
41.           JZ SAW
42.           JMP SAWINCR
```

Screen-grab of Simulator:



Discussion:

In this Experiment, I used three different code blocks RECT, TRI and SAW to represent different waveform. We can generate any single one of the waveforms by commenting out the other two blocks. Also, we can change the amplitude of the generated square waveform by simply modifying the content of the accumulator before sending to port A in *line-10*. For triangular or sawtooth just modify FFH in *line-23* or *line-40* from CPI FFH to desired value of amplitude respectively.

Embedded System Lab

(ELC3930)

Experiment No.: 06

Object:

Write a program to interface an 8-bit ADC 0808 with 8085 microprocessor.

G. No: GL3136

S. No: A3EL-02

F. No: 19ELB056

Name: Maha Zakir Khan

Date of performing experiment: 21 | 02 | 2022

Date of report submission: 27 | 02 | 2022

Simulator Used:

8085 Simulator by Jubin Mitra. It helps in getting started easily with example codes, and to learn the architecture playfully. This tool is an integrated software environment for teaching microprocessor concepts. The software is shared under opensource GNU license.

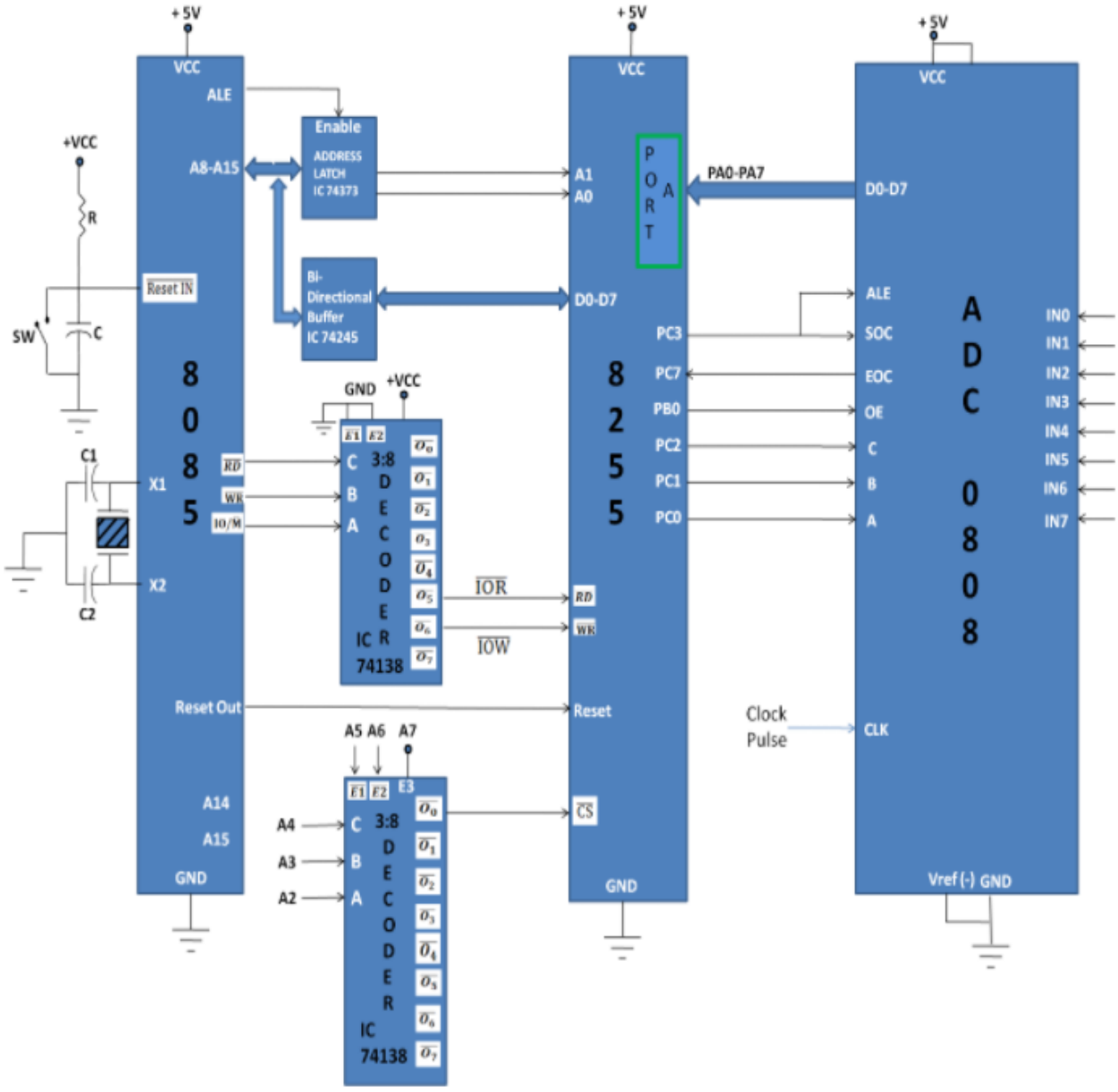
Link: 8085 Jubin Simulator

Algorithm:

Interfacing ADC 0808 (Analog to Digital Converter) with 8085:

Follow the initial 3 steps of interfacing of 8255 with 8085 that are given before in *Experiment-05*. The ADC 0808 is 8-channel 8-bit ADC chip.

Interfacing Diagram of ADC



It has 8 analog inputs i.e. IN0-IN7. One of these channels is selected by sending address to an address line of ADC. The logic level and selected channel is as shown:

A	B	C	Channel
0	0	0	IN0
0	0	1	IN1
0	1	0	IN2
0	1	1	IN3
1	0	0	IN4
1	0	1	IN5
1	1	0	IN6
1	1	1	IN7

The analog signal is connected to channel 1

- The digital equivalent data D0-D7 is connected to PA0-PA7 of Port A.
- The PB0, PB1 and PB2 lines of Port B are connected to channel select address lines of 8255.
- PC0 is connected to SOC (Start of conversion), PC1 is connected to ALE and PC2 of Port C is connected to OE (Output Enable) of ADC.
- EOC (End of conversion) for reading the status of ADC is connected to PC4 of Port C.

The control word format for above interface is given as:

D7	D6	D5	D4	D3	D2	D1	D0
IO/BSR	MA	MA	PA	PCU	MB	PB	PCU
1	0	0	1	1	0	0	0

=98H

- Port A and Port C-upper (PC7-PC4) are configured as Input Port
- Port B and Port C-lower (PC3-PC0) are configured as Output Port

PC7	PC 6	PC 5	PC 4	PC 3	PC 2	PC 1	PC 0
0	0	0	0	0	0	1	0

= 02H (ALE = HIGH)

- ALE is required to load the selected address lines into the ADC

PC7	PC 6	PC 5	PC 4	PC 3	PC 2	PC 1	PC 0
0	0	0	0	0	0	0	1

= 01H (SOC=HIGH)

- A high to low SOC pulse is applied for obtaining data from ADC.

PC7	PC 6	PC 5	PC 4	PC 3	PC 2	PC 1	PC 0
0	0	0	1	0	0	0	0

=10H (EOC=HIGH)

- ADC sets EOC signal to high at the end of analog to digital conversion

PC7	PC 6	PC 5	PC 4	PC 3	PC 2	PC 1	PC 0
0	0	0	0	0	1	0	0

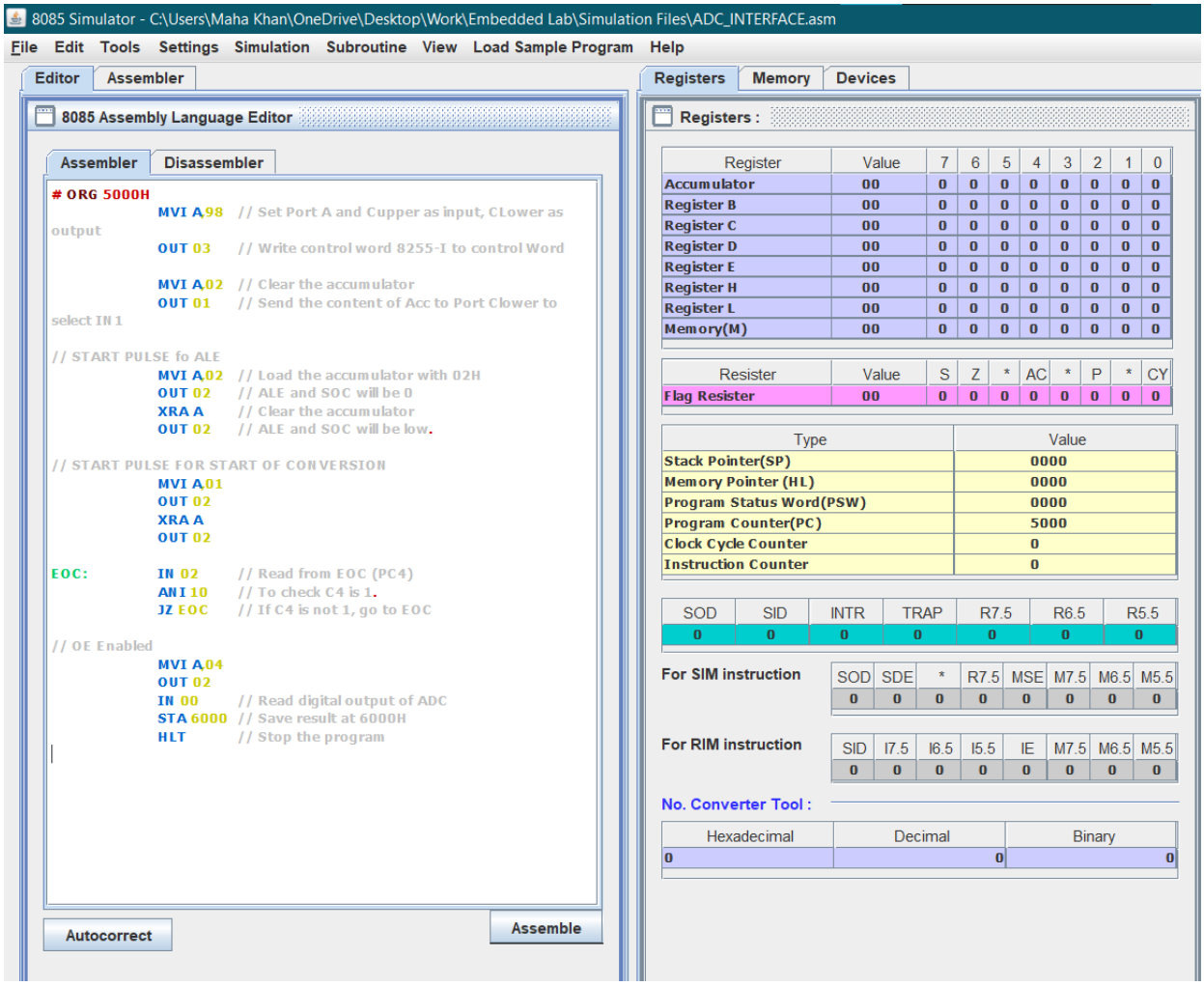
=01H (OE=HIGH)

- Output Enable set to high to read from the digital output pins D0-D7 of ADC

Program:

```
1.  # ORG 5000H
2.     MVI A,98    // Set Port A and C-upper as input, C-Lower as output
3.     OUT 03      // Send control word to 8255
4.
5.     MVI A,01    // Address of channel 1 of Analog input of ADC
6.     OUT 01      // Send the content of Acc to Port Clower to select IN1
7.
8.  // Generating pulse for ALE
9.     MVI A,03    // Load the accumulator with 03H
10.    OUT 02      // Send to port C-lower, ALE will be high i.e. PC1 = 1
11.    XRA A       // Clear the accumulator
12.    OUT 02      // Send to port C-lower, ALE will be low i.e. PC1 = 0
13.
14. // Generating pulse for Start Of Conversion (SOC)
15.    MVI A,03    // Load the accumulator with 03H
16.    OUT 02      // Send to port C-lower, SOC will be high i.e. PC0 = 1
17.    XRA A       // Clear the accumulator
18.    OUT 02      // Send to port C-lower, SOC will be low i.e.  PC0 = 0
19.
20. EOC:   IN 02        // Read from EOC (PC4)
21.        ANI 10       // To check if PC4 is 1
22.        JZ EOC       // If PC4 is not 1, go to EOC
23.
24. // OE Enabled
25.    MVI A,04
26.    OUT 02      //Send OE(PC2) signal to Port C
27.
28.    IN 00       // Read digital output of ADC at Port A
29.    STA 6000    // Save result at 6000H
30.    HLT         // Stop the program
```

Screen-grab of Simulator:



Discussion:

In this Experiment, we performed interfacing of ADC with 8085 microprocessor to read Analog data. It should be noted that The ALE should be pulsed for at least 100ns in order for the addresses to get loaded properly. As with all control signals it is required to have an input value of Vcc - 1.5 up to 15V for a high and 1.5V down to -0.3V for a low. The SOC signal can be tied to the ALE signal when the clock frequency is below 500kHz. At clock speeds greater than that we must make certain that enough time has passed since the ALE signal was pulsed so that the correct address is loaded into the multiplexer before a conversion begins, it can take up to 2.5 microseconds for this to occur.