Microcontroller System Laboratory

Experiment 4 – Part 4, 5, 6

Utkarsh Patel 18EC35034

Part 4 – Sinusoidal Signal Generation

Objective

Write programs to generate a sinusoidal waveform through the DAC interface. Display the waveform on the DAC.

Circuit Diagram

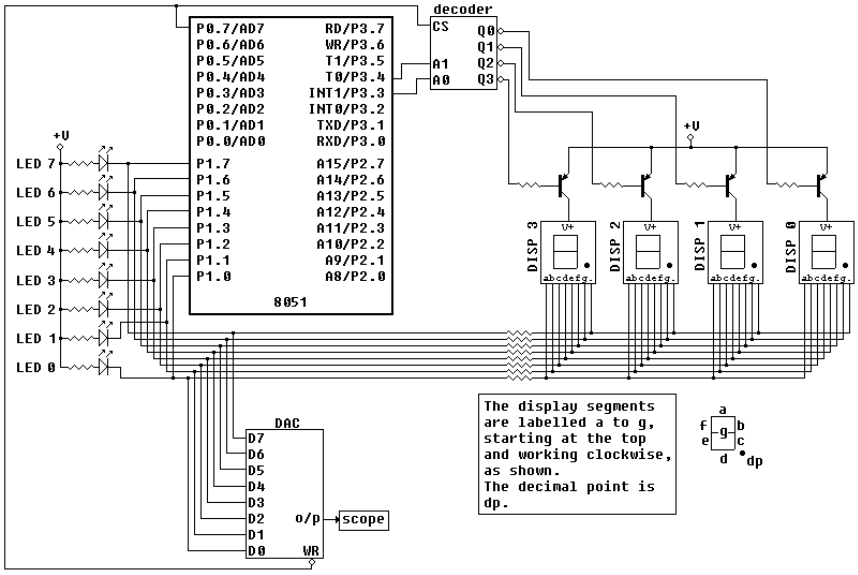

Fig. 1. Circuit diagram

Code

```
; Run this code with update frequency 10

ORG 0000              ; reset vectors
CLR P0.7              ; enabling DAC WR line

start:                ; starting a new cycle
  MOV DPTR, #0070H    ; setting DPTR to point to memory at 70H
  MOV R0, #12         ; for loop

loop:                 ; reading value of signal and displaying it on the scope
  CLR A               ; clearing accumulator
  MOVC A, @A+DPTR     ; reading value using DPTR
  MOV P1, A           ; move content of accumulator to P1 for display
  INC DPTR            ; increment DPTR to read new value
  DJNZ R0, loop       ; looping 12 times
```

```
SJMP start            ; jump back to start

ORG 0070H             ; storing value of sinusoidal signal using step size of 30 degrees
DB 128, 192, 238, 255, 238, 192, 128, 64, 17, 0, 17, 64, 128;

END
```

Waveform

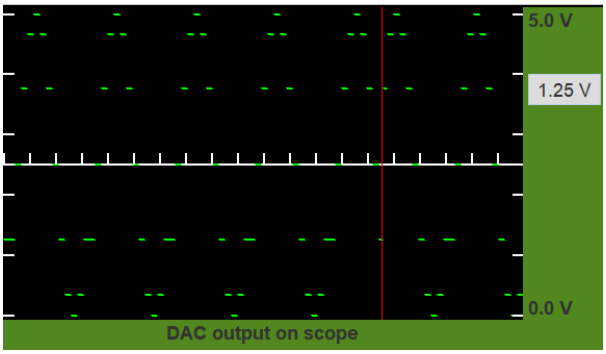

Fig. 2. Sinusoidal waveform

Discussion

- In this part, a sinusoidal signal was simulated using DAC module in Edsim simulator.
- The Digital to Analog converter (DAC) is a device, that is widely used for converting digital pulses to analog signals.
- **MC1408 (DAC0808)**
  - In this chip the digital inputs are converted to current.
  - The output current is known as I_{out} by connecting a resistor to the output to convert into voltage.
  - The total current provided by the I_{out} pin is basically a function of the binary numbers at the input pins $D_0 - D_7$ (D_0 is the LSB and D_7 is the MSB) of DAC0808 and the reference current I_{ref}.
  - The following formula is showing the function of I_{out}:

$$I_{Out} = I_{ref}\left(\frac{D7}{2} + \frac{D6}{4} + \frac{D5}{8} + \frac{D4}{16} + \frac{D3}{32} + \frac{D2}{64} + \frac{D1}{128} + \frac{D0}{256}\right)$$

- **Sinusoidal Signal Generation**
  - The vectors are reset and the values of the function $f(t) = 5 + 5\sin(t)$ is stored in DDRAM for step size of 30 degrees.
  - P0.7 is set to logic low, to enable DAC WR line (see circuit diagram).
  - The start module constructs a new cycle of the sinusoidal signal.
  - The loop module reads the value pointed by the DPTR and DPTR is incremented. The loop is executed 12 times and then the control passes to start module to initiate a new cycle.
  - The frequency of waveforms can be made arbitrarily high, however they might not be distinctly representing that function anymore because of utilization of limited points. Hence the values stated in the following points will state the frequency where the candidate himself feels that it still represents that function. It is a very subjective value and can change with perspective.
  - Amplitude of all the waveforms were 5V peak to peak (almost) (it can be easily done by taking a very low(say 4,6) and highest value(255) to the port of ADC.
  - The resolution of the DAC is 0.02V and the peak value of DAC is 5V, although total distinct possible values are 0 to 255 for a full scale of 5V, the resolution is approximated to 0.02V.

Part 5 – Displaying ADC Output on LCD panel

Objective
Using the ADC, convert a voltage (analog) into digital form. Display the output of the ADC on the LCD panel.

Circuit Diagram

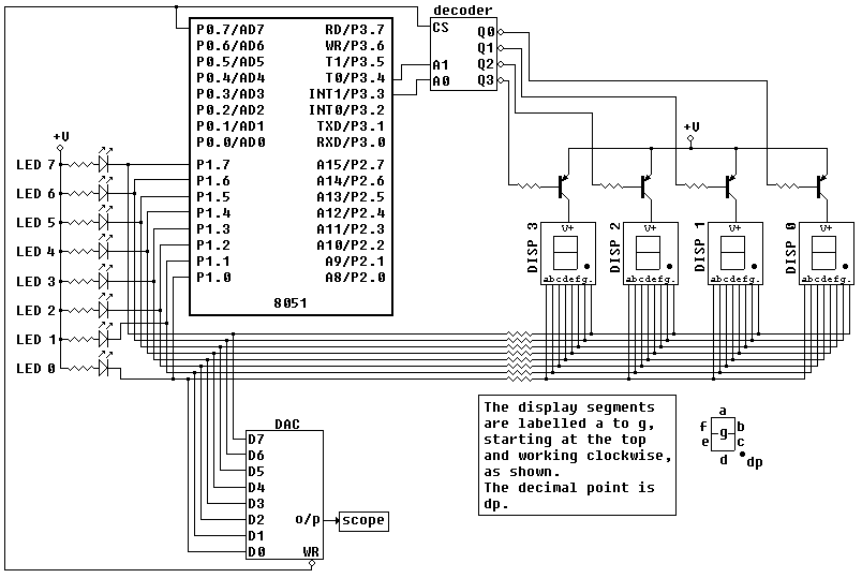

Fig. 3. Circuit diagram

Code
```
; Run this code with update frequency 1000

org 0000H        ; reset vectors

MOV 30H, #'0'    ; storing ASCII values for digits
MOV 31H, #'1'
MOV 32H, #'2'
MOV 33H, #'3'
MOV 34H, #'4'
MOV 35H, #'5'
MOV 36H, #'6'
MOV 37H, #'7'
MOV 38H, #'8'
MOV 39H, #'9'

CLR P1.3         ; lcd instruction mode on

; configuring lcd 4-bit mode operation

CLR P1.7         ; |
CLR P1.6         ; |
SETB P1.5        ; |
CLR P1.4         ; | higher nibble value
CALL pass        ; negative edge on enable
CALL pass
SETB P1.7        ; lower nibble value
CALL pass        ; negative edge on enable
```

```
; entry mode set

CLR P1.7         ; |
CLR P1.6         ; |
CLR P1.5         ; |
CLR P1.4         ; | higher nibble value
CALL pass        ; negative edge on enable
SETB P1.6        ; |
SETB P1.5        ; | lower nibble value
CALL pass        ; negative edge on enable

; lcd display ON - cursor ON - blinking ON

CLR P1.6         ; |
CLR P1.5         ; | higher nibble value
CALL pass        ; negative edge on enable
SETB P1.7        ; |
SETB P1.6        ; |
SETB P1.5        ; |
SETB P1.4        ; | lower nibble value
CALL pass        ; negative edge on enable

SETB P1.3        ; lcd data mode on

CLR P3.7         ; turning ADC output ON
loop:
  CLR P3.6       ; |
  SETB P3.6      ; | postive edge on ADC WR to start conversion
  CALL delay     ; waiting for conversion to start and INTR to go high
  JB P3.2, $     ; waiting for the INTR to go low i.e completion of the conversion
  MOV A, P2      ; take the data from the ADC on P2 and send it to the accumulator

  ; converting the obtained value to BCD
  MOV B, #10
  DIV AB
  MOV R3, B
  MOV B, #10
  DIV AB
  MOV R2, B
  MOV R1, A

  ; R1:R2:R3 contains the BCD value, e.g. 5V -> 255 -> R1 = 2, R2 = 5, R3 = 5
  CALL sendcharacter    ; displaying R1
  MOV A, R2
  CALL sendcharacter    ; displaying R2
  MOV A, R3
  CALL sendcharacter    ; displaying R3
  CALL delay

  ; reset display to make room for next value
  CLR P1.3        ; lcd instruction mode on

  CLR P1.7        ; |
  CLR P1.6        ; |
  CLR P1.5        ; |
  CLR P1.4        ; | higher nibble value
  CALL pass       ; negative edge on enable
  SETB P1.4       ; | lower nibble value
  CALL pass       ; negative edge on enable
  CALL delay;
```

```
  SETB P1.3       ; lcd data mode on
  JMP loop

pass:             ; negative edge on enable
  SETB P1.2
  CLR P1.2
  MOV R7, #50     ; small delay for lcd buffer
  DJNZ R7, $
  RET

sendcharacter:    ; send data in accumlator in LCD to display it
  ADD A, #30H
  MOV R1, A
  MOV A, @R1
  MOV C, ACC.7    ; |
  MOV P1.7, C     ; |
  MOV C, ACC.6    ; |
  MOV P1.6, C     ; |
  MOV C, ACC.5    ; |
  MOV P1.5, C     ; |
  MOV C, ACC.4    ; |
  MOV P1.4, C     ; | higher nibble value
  CALL pass       ; negative edge on enable
  MOV C, ACC.3    ; |
  MOV P1.7, C     ; |
  MOV C, ACC.2    ; |
  MOV P1.6, C     ; |
  MOV C, ACC.1    ; |
  MOV P1.5, C     ; |
  MOV C, ACC.0    ; |
  MOV P1.4, C     ; | lower nibble value
  CALL pass       ; negative edge on enable
  RET

delay:
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  RET
```

Discussion

- The LCD module consists of 16 rows and 2 columns of 5x8 dot matrices.
- Name of the pins and their corresponding functions are as follows:

Pin Name	Function
VSS	Must be grounded
VCC	5V DC power supply
RS	Register Selection
R/W	Read/write
E	Enable
DB[7:0]	Data

- From the circuit diagram, it can be observed that P1.3 is the RS of the LCD display, and P1.2 is the E of the LCD display.
- To execute any set of command, a negative edge has to be generated by E, i.e., set P1.2 to logic high and then to logic low and add some delay
- There are two types of register modes:
    - **Command mode**: Indicates flow of instruction to the LCD module, RS (P1.3) must be set to logic low to select this register mode
    - **Data mode**: Indicates flow of data to the LCD module, RS (P1.3) must be set to logic high to select this register mode
- Every operation linked with LCD display has a unique hexadecimal code. Some of them are:

Code (in hexadecimal)	Operation
0F	LCD ON, cursor ON, blinking ON
01	Clear screen
02	Return home
04	Decrement cursor
06	Increment cursor
0E	Display ON, cursor OFF
80	Force cursor to the beginning of 1st line
C0	Force cursor to the beginning of 2nd line
38	Use 2 lines and 5x7 matrix
83	Cursor line 1 position 3
3C	Activate second line
08	Display OFF, cursor OFF
C1	Jump to second line, position 1
C2	Jump to second line, position 2
0C	Display ON, cursor OFF

- To perform any operation with code, say 0x75 = 01110101B, we divide the instruction into high nibble set (0111 in this case) and low nibble set (0101 in this case). Then we do:

```
CLR P1.7     ;  0|
SETB P1.6    ;  1|
SETB P1.5    ;  1|
SETB P1.4    ;  1| high nibble set
SETB P1.2    ;   |
CLR P1.2     ;   | negative edge on E
CLR P1.7     ;  0|
SETB P1.6    ;  1|
CLR P1.5     ;  0|
SETB P1.4    ;  1| low nibble set
SETB P1.2    ;   |
CLR P1.2     ;   | negative edge on E
CALL delay
```

- sendcharacter module is used for displaying the current character on the LCD display
- **Reading from ADC**
    - P3.7 is set to logic low to turn ADC output ON
    - To start the conversion, a positive edge is applied on ADC WR line P3.6.
    - When the completion is completed (P3.2 is the conversion completion flag), the digital value is stored in P2.
    - This value can be copied to another register for the specific operation.

Part 6 – ADC Output on LCD Panel and DAC Scope

Objective

Take Samples from the ADC for three different input voltage values: 1V, 2V and 3V and, display these on the Scope via the DAC and the respective digital values on the LCD simultaneously

Circuit Diagram

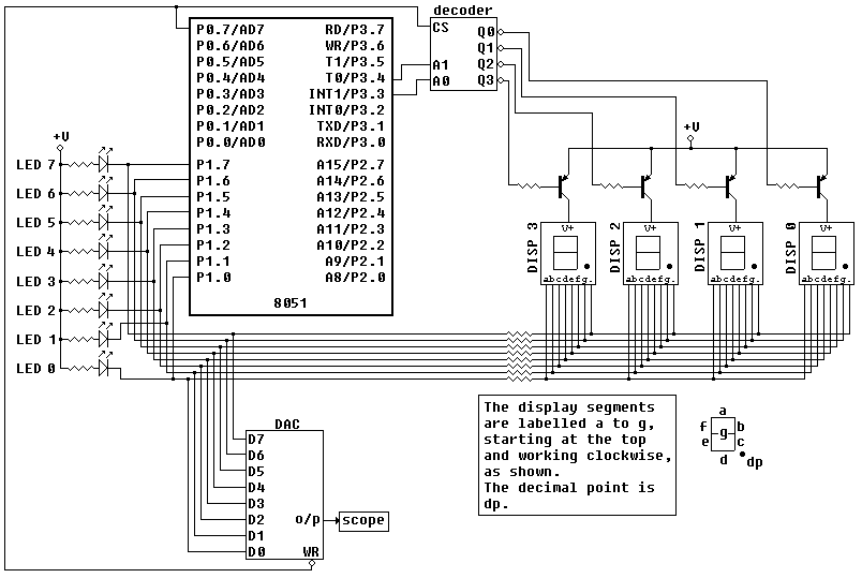

Fig. 4. Circuit diagram

Code

```
; Run this code with update frequency 1000

org 0000H        ; reset vectors

MOV 30H, #'0'    ; storing ASCII values for digits
MOV 31H, #'1'
MOV 32H, #'2'
MOV 33H, #'3'
MOV 34H, #'4'
MOV 35H, #'5'
MOV 36H, #'6'
MOV 37H, #'7'
MOV 38H, #'8'
MOV 39H, #'9'

CLR P1.3         ; lcd instruction mode on

; configuring lcd 4-bit mode operation

CLR P1.7         ; |
CLR P1.6         ; |
SETB P1.5        ; |
CLR P1.4         ; | higher nibble value
CALL pass        ; negative edge on enable
CALL pass
SETB P1.7        ; lower nibble value
CALL pass        ; negative edge on enable

; entry mode set
```

```
CLR P1.7         ; |
CLR P1.6         ; |
CLR P1.5         ; |
CLR P1.4         ; | higher nibble value
CALL pass        ; negative edge on enable
SETB P1.6        ; |
SETB P1.5        ; | lower nibble value
CALL pass        ; negative edge on enable

; lcd display ON - cursor ON - blinking ON

CLR P1.6         ; |
CLR P1.5         ; | higher nibble value
CALL pass        ; negative edge on enable
SETB P1.7        ; |
SETB P1.6        ; |
SETB P1.5        ; |
SETB P1.4        ; | lower nibble value
CALL pass        ; negative edge on enable

SETB P1.3        ; lcd data mode on

CLR P3.7         ; turning ADC output ON
loop:
  CLR P3.6       ; |
  SETB P3.6      ; | postive edge on ADC WR to start conversion
  CALL delay     ; waiting for conversion to start and INTR to go high
  JB P3.2, $     ; waiting for the INTR to go low i.e completion of the conversion
  MOV A, P2      ; take the data from the ADC on P2 and send it to the accumulator

  ; converting the obtained value to BCD
  MOV B, #10
  DIV AB
  MOV R3, B
  MOV B, #10
  DIV AB
  MOV R2, B
  MOV R1, A

  ; R1:R2:R3 contains the BCD value, e.g. 5V -> 255 (digital value) -> R1 = 2, R2 = 5, R3 =
5
  CALL sendcharacter    ; displaying R1
  MOV A, R2
  CALL sendcharacter    ; displaying R2
  MOV A, R3
  CALL sendcharacter    ; displaying R3

  MOV P1, P2            ; displaying the ADC Output on DAC scope
  CLR P0.7              ; enabling DAC WR line
  SETB P0.7             ; disabling DAC WR line
  CALL delay

  ; reset display to make room for next value
  CLR P1.3        ; lcd instruction mode on

  CLR P1.7        ; |
  CLR P1.6        ; |
  CLR P1.5        ; |
  CLR P1.4        ; | higher nibble value
  CALL pass       ; negative edge on enable
```

```
  SETB P1.4       ; | lower nibble value
  CALL pass       ; negative edge on enable
  CALL delay;

  SETB P1.3       ; lcd data mode on
  JMP loop

pass:             ; negative edge on enable
  SETB P1.2
  CLR P1.2
  MOV R7, #50     ; small delay for lcd buffer
  DJNZ R7, $
  RET

sendcharacter:    ; send data in accumlator to current address of DDRAM in LCD to display
  ADD A, #30H
  MOV R1, A
  MOV A, @R1
  MOV C, ACC.7    ; |
  MOV P1.7, C     ; |
  MOV C, ACC.6    ; |
  MOV P1.6, C     ; |
  MOV C, ACC.5    ; |
  MOV P1.5, C     ; |
  MOV C, ACC.4    ; |
  MOV P1.4, C     ; | higher nibble value
  CALL pass       ; negative edge on enable
  MOV C, ACC.3    ; |
  MOV P1.7, C     ; |
  MOV C, ACC.2    ; |
  MOV P1.6, C     ; |
  MOV C, ACC.1    ; |
  MOV P1.5, C     ; |
  MOV C, ACC.0    ; |
  MOV P1.4, C     ; | lower nibble value
  CALL pass       ; negative edge on enable
  RET

delay:
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  MOV R7, #255
  DJNZ R7, $
  RET
```

Discussion

- The LCD module consists of 16 rows and 2 columns of 5x8 dot matrices.
- Name of the pins and their corresponding functions are as follows:

Pin Name	Function
VSS	Must be grounded
VCC	5V DC power supply
RS	Register Selection
R/W	Read/write
E	Enable
DB[7:0]	Data

- From the circuit diagram, it can be observed that P1.3 is the RS of the LCD display, and P1.2 is the E of the LCD display.
- To execute any set of command, a negative edge has to be generated by E, i.e., set P1.2 to logic high and then to logic low and add some delay
- There are two types of register modes:
    - **Command mode**: Indicates flow of instruction to the LCD module, RS (P1.3) must be set to logic low to select this register mode
    - **Data mode**: Indicates flow of data to the LCD module, RS (P1.3) must be set to logic high to select this register mode
- Every operation linked with LCD display has a unique hexadecimal code. Some of them are:

Code (in hexadecimal)	Operation
0F	LCD ON, cursor ON, blinking ON
01	Clear screen
02	Return home
04	Decrement cursor
06	Increment cursor
0E	Display ON, cursor OFF
80	Force cursor to the beginning of 1st line
C0	Force cursor to the beginning of 2nd line
38	Use 2 lines and 5x7 matrix
83	Cursor line 1 position 3
3C	Activate second line
08	Display OFF, cursor OFF
C1	Jump to second line, position 1
C2	Jump to second line, position 2
0C	Display ON, cursor OFF

- To perform any operation with code, say 0x75 = 01110101B, we divide the instruction into high nibble set (0111 in this case) and low nibble set (0101 in this case). Then we do:

```
CLR P1.7     ;  0|
SETB P1.6    ;  1|
SETB P1.5    ;  1|
SETB P1.4    ;  1| high nibble set
SETB P1.2    ;   |
CLR P1.2     ;   | negative edge on E
CLR P1.7     ;  0|
SETB P1.6    ;  1|
CLR P1.5     ;  0|
SETB P1.4    ;  1| low nibble set
SETB P1.2    ;   |
CLR P1.2     ;   | negative edge on E
CALL delay
```

- sendcharacter module is used for displaying the current character on the LCD display

- **Reading from ADC**
  - P3.7 is set to logic low to turn ADC output ON
  - To start the conversion, a positive edge is applied on ADC WR line P3.6.
  - When the completion is completed (P3.2 is the conversion completion flag), the digital value is stored in P2.
  - This value can be copied to another register for the specific operation.
- **MC1408 (DAC0808)**
  - In this chip the digital inputs are converted to current.
  - The output current is known as I_{out} by connecting a resistor to the output to convert into voltage.
  - The total current provided by the I_{out} pin is basically a function of the binary numbers at the input pins $D_0 - D_7$ (D_0 is the LSB and D_7 is the MSB) of DAC0808 and the reference current I_{ref}.
  - The following formula is showing the function of I_{out}:

$$I_{Out} = I_{ref}\left(\frac{D7}{2} + \frac{D6}{4} + \frac{D5}{8} + \frac{D4}{16} + \frac{D3}{32} + \frac{D2}{64} + \frac{D1}{128} + \frac{D0}{256}\right)$$

  - To display the digital value on DAC scope, the data from ADC output (P2) is copied into DAC input (P1) and DAC WR line (P0.7) is set to logic low (enabled), and then set to logic high (disabled).