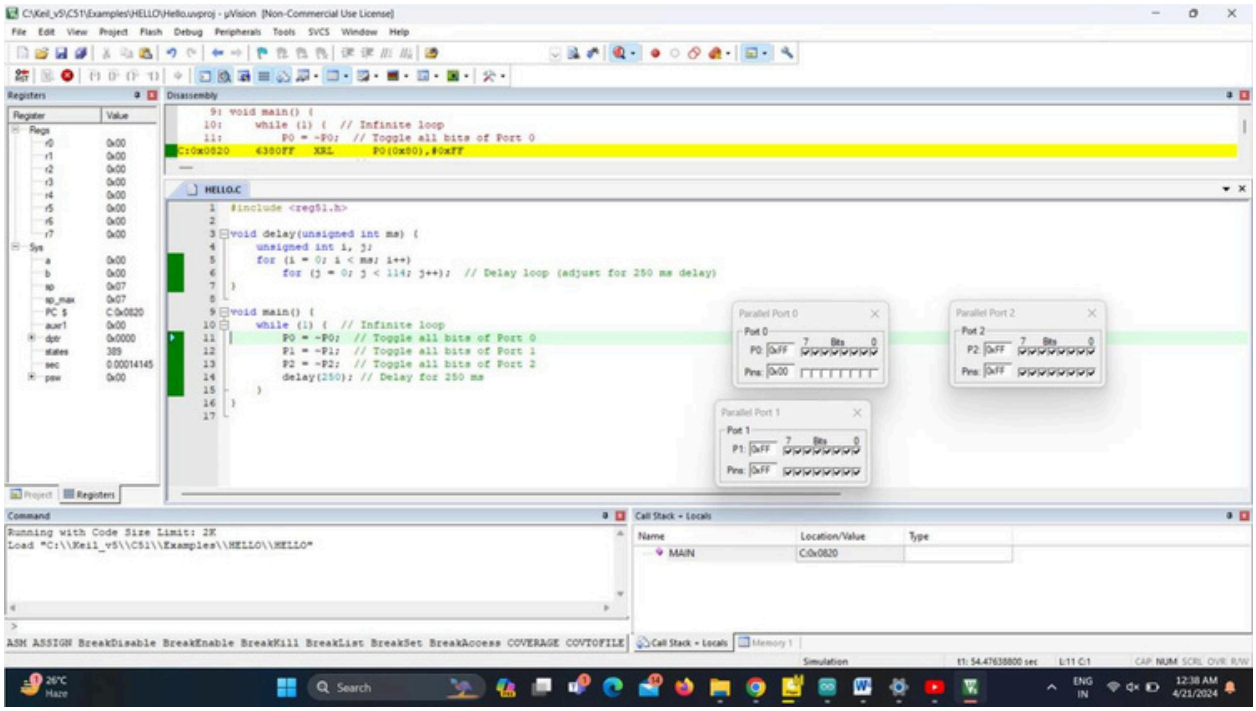
# Embedded System Lab Assigment-7

## Write Program in KEIL Embedded C:

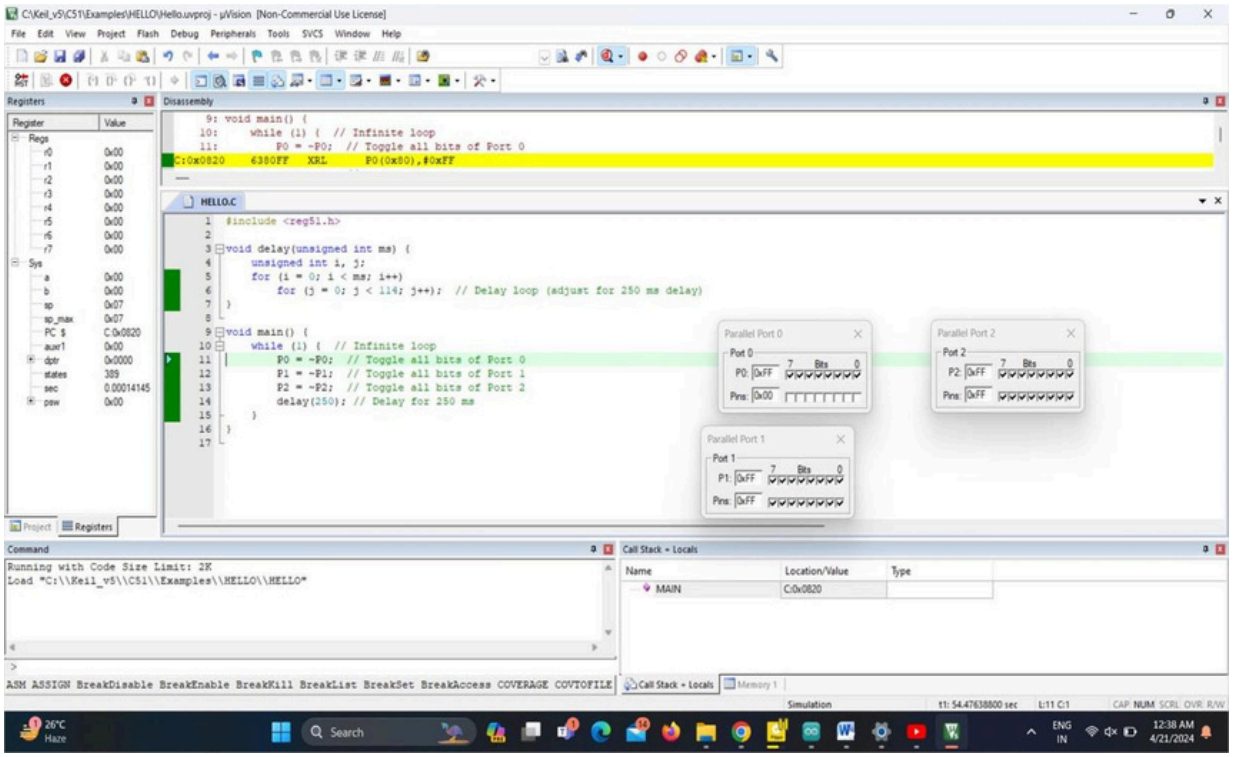
1.Write an 8051 C program to toggle all the bits of P0 ,P1 , and P2 continuously with a 250 ms delay. Use the sfr keyword to declare the port address.

```
#include <reg51.h>

void delay(unsigned int ms)
{ unsigned int i, j;
for (i= 0; i < ms; i++)
for (j = 0; j < 114; j++); // Delay loop (adjust for 250 ms delay)
}

void main() {
while (1) { // Infinite loop
P0 = ~P0; // Toggle all bits of Port 0
P1 = ~P1; // Toggle all bits of Port 1
P2 = ~P2; // Toggle all bits of Port 2
delay(250); // Delay for 250 ms
}
}
```



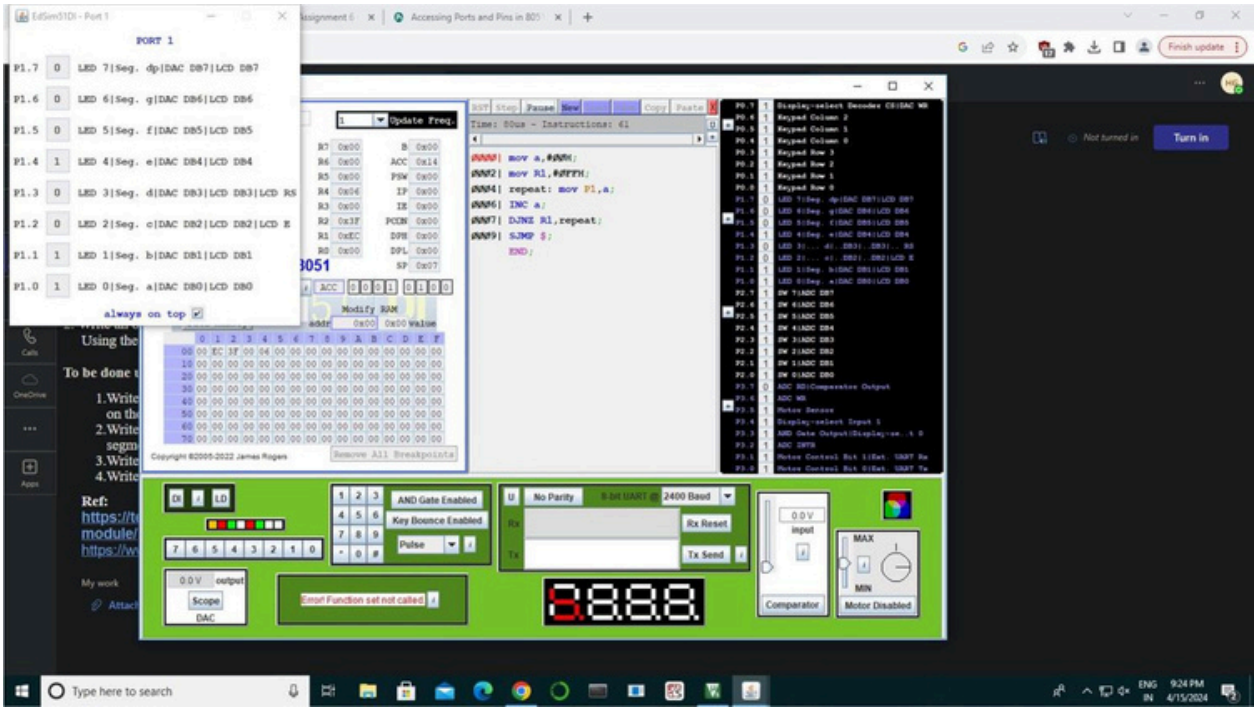

2- Write an 8051 c program to toggle all the bits of P0 and P2 continuously with a 250 ms delay. Using the inverting and Ex-Or operators, respectively. Ans-

```
#include <reg51.h>

void delay(unsigned int ms)
{ unsigned int i, j;
for (i = 0; i < ms; i++)
for (j = 0; j < 114; j++); // Delay loop (adjust for 250 ms delay)
}

void main() {
while (1) { // Infinite loop
P0 = ~P0; // Toggle all bits of Port 0 using the inverting operator
P2 = P2  ^0xFF; // Toggle all bits of Port 2 using XOR with 0xFF
delay(250); // Delay for 250 ms
}
}
```




Q1 Write an assembly program that displays the binary pattern from to 255 (and back to 0) on the LEDs interfaced with port 1.

```
mov a,#00H;
mov R1,#0FFH;
repeat: mov P1,a;
INC a;
DJNZ R1,repeat;
SJMP $;
END;
```

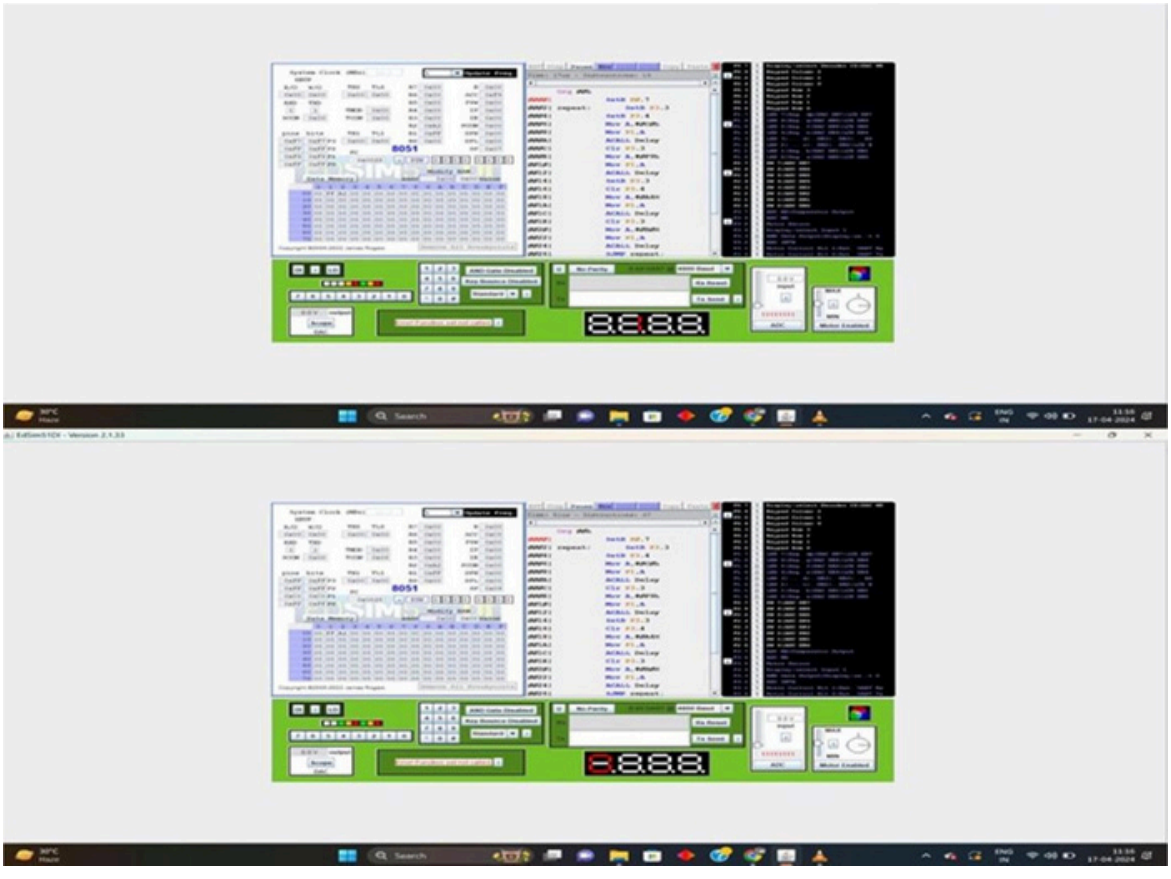


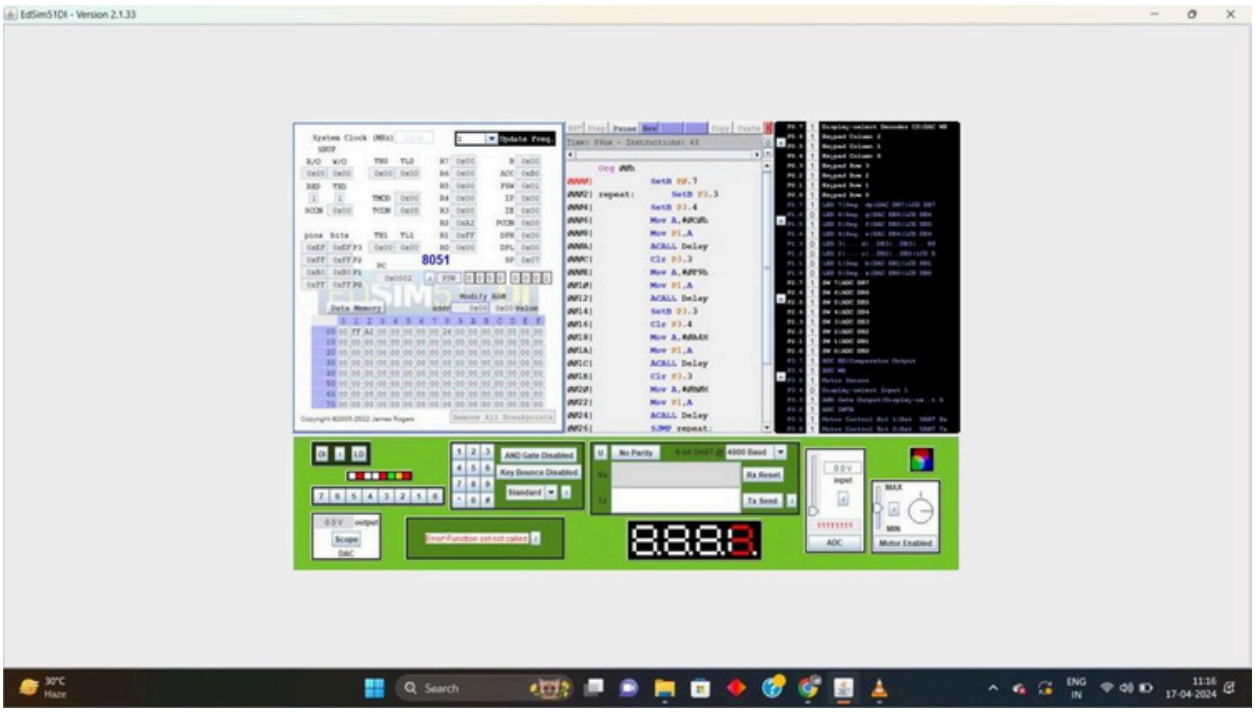
Q2 Write an assembly language program that multiplexes the number 1234 on the four 7-segment displays.

```
Org 00h
SetB P0.7
repeat: SetB P3.3
SetB P3.4
Mov A,#0C0h
Mov P1,A
ACALL Delay
Clr P3.3
Mov A,#0F9h
Mov P1,A
ACALL Delay
SetB P3.3
Clr P3.4
Mov A,#0A4H
```

```
 Mov P1,A
ACALL Delay
Clr P3.3
Mov A,#0B0H
Mov P1,A
ACALL Delay

SJMP repeat;
delay:
MOV R0,#1H
loop: DJNZ R0,loop
RET
End
```

Since EDSim51 tool can’t switch between two LED quickly we will only able to see one digit at a time.



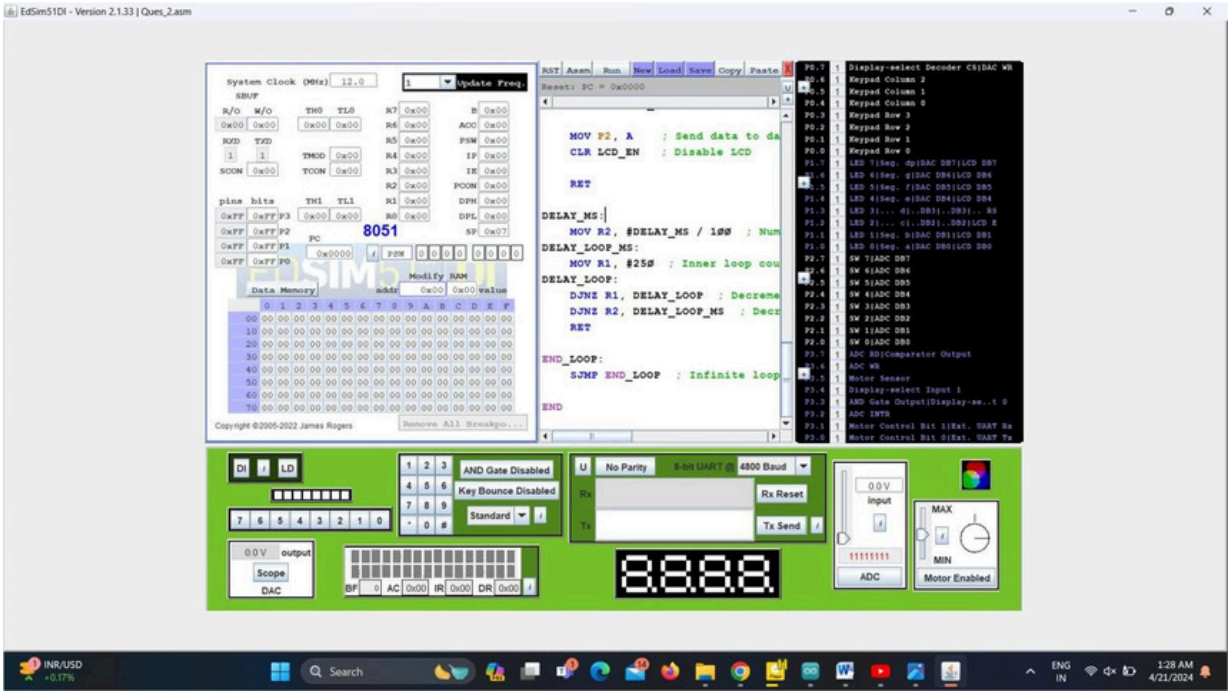



# 3- Write a program to display message on the LCD of 8051 microcontroller. Ans-

```
ORG 0H ; Start of program memory

; Define LCD control signals
LCD_RS EQU P1.0 ; Register select pin of LCD
LCD_EN EQU P1.1 ; Enable pin of LCD

; Define LCD data pins
LCD_DATA EQU P2 ; Data pins of LCD
; Define delay function
DELAY_MS EQU 1000 ; Adjust this value for the desired delay
; Define message
MESSAGE: DB "Hello, World!", 0 ; Null-terminated message string
MAIN:
CALL INIT_LCD ; Initialize LCD

MOV DPTR, #MESSAGE ; Load address of message string

DISPLAY_LOOP:
MOVX A, @DPTR ; Load character from message
INC DPTR ; Move to next character
CJNE A, #0, SEND_DATA ; If not null character, send data to LCD
```

```
 SJMP END_LOOP ; Otherwise, end loop

INIT_LCD:
MOV A, #38H ; Function set: 2 lines, 5x7 font
CALL SEND_COMMAND

MOV A, #0EH ; Display control: Display ON, Cursor ON, Blinking ON
CALL SEND_COMMAND
MOV A, #01H ; Clear display
CALL SEND_COMMAND
MOV A, #06H ; Entry mode set: Increment cursor, No display shift
CALL SEND_COMMAND
RET

SEND_COMMAND:
CLR LCD_RS ; Set RS pin LOW for command mode
ACALL DELAY_MS ; Wait for LCD to accept command

SETB LCD_EN ; Enable LCD
ACALL DELAY_MS ; Wait for LCD to accept command
MOV P2, A ; Send command to data pins
CLR LCD_EN ; Disable LCD

RET

SEND_DATA:
SETB LCD_RS ; Set RS pin HIGH for data mode
ACALL DELAY_MS ; Wait for LCD to accept data

SETB LCD_EN ; Enable LCD
ACALL DELAY_MS ; Wait for LCD to accept data
MOV P2, A ; Send data to data pins
CLR LCD_EN ; Disable LCD
RET

DELAY_MS:
MOV R2, #DELAY_MS / 100 ; Number of milliseconds to delay (adjust for clock frequency)
DELAY_LOOP_MS:
MOV R1, #250 ; Inner loop count (adjust for clock frequency)
DELAY_LOOP:
DJNZ R1, DELAY_LOOP ; Decrement inner loop counter
DJNZ R2, DELAY_LOOP_MS ; Decrement outer loop counter
RET

END_LOOP:
SJMP END_LOOP ; Infinite loop
```

END




# 4- Write a program to display your name on the LCD of 8051 microcontroller. Ans-

ORG 0H ; Start of program memory

; Define LCD control signals
LCD_RS EQU P1.0 ; Register select pin of LCD
LCD_EN EQU P1.1 ; Enable pin of LCD

; Define LCD data pins
LCD_DATA EQU P2 ; Data pins of LCD

; Define delay function
DELAY_MS EQU 1000 ; Adjust this value for the desired delay
; Define message
NAME: DB "Your Name", 0 ; Null-terminated name string
MAIN:
CALL INIT_LCD ; Initialize LCD
MOV DPTR, #NAME ; Load address of name string

DISPLAY_LOOP:
MOVX A, @DPTR ; Load character from name
INC DPTR ; Move to next character
CJNE A, #0, SEND_DATA ; If not null character, send data to LCD
SJMP END_LOOP ; Otherwise, end loop

INIT_LCD:

```
 MOV A, #38H ; Function set: 2 lines, 5x7 font
CALL SEND_COMMAND

MOV A, #0EH ; Display control: Display ON, Cursor ON, Blinking ON
CALL SEND_COMMAND

MOV A, #01H ; Clear display
CALL SEND_COMMAND
MOV A, #06H ; Entry mode set: Increment cursor, No display shift
CALL SEND_COMMAND
RET

SEND_COMMAND:
CLR LCD_RS ; Set RS pin LOW for command mode
ACALL DELAY_MS ; Wait for LCD to accept command

SETB LCD_EN ; Enable LCD
ACALL DELAY_MS ; Wait for LCD to accept command
MOV P2, A ; Send command to data pins
CLR LCD_EN ; Disable LCD

RET

SEND_DATA:
SETB LCD_RS ; Set RS pin HIGH for data mode
ACALL DELAY_MS ; Wait for LCD to accept data

SETB LCD_EN ; Enable LCD
ACALL DELAY_MS ; Wait for LCD to accept data
MOV P2, A ; Send data to data pins
CLR LCD_EN ; Disable LCD
RET

DELAY_MS:
MOV R2, #DELAY_MS / 100 ; Number of milliseconds to delay (adjust for clock frequency)
DELAY_LOOP_MS:
MOV R1, #250 ; Inner loop count (adjust for clock frequency)
DELAY_LOOP:
DJNZ R1, DELAY_LOOP ; Decrement inner loop counter
DJNZ R2, DELAY_LOOP_MS ; Decrement outer loop counter
RET

END_LOOP:
SJMP END_LOOP ; Infinite loop
END
```