SUMMER REPORT EMBEDDED C

8051(MICRO-CONTROLLER) (Trainer: Dr. Jeevan K M)

Rajesh Boya

BU21EECE0100482

- **Learning Objective:**
  - Understanding how to create patterns of LED blinking using the 8051 microcontroller.
- **Inputs and Outputs:**
  - **Inputs:** None
  - **Outputs:** LED states (ON/OFF)
- **Logic:**

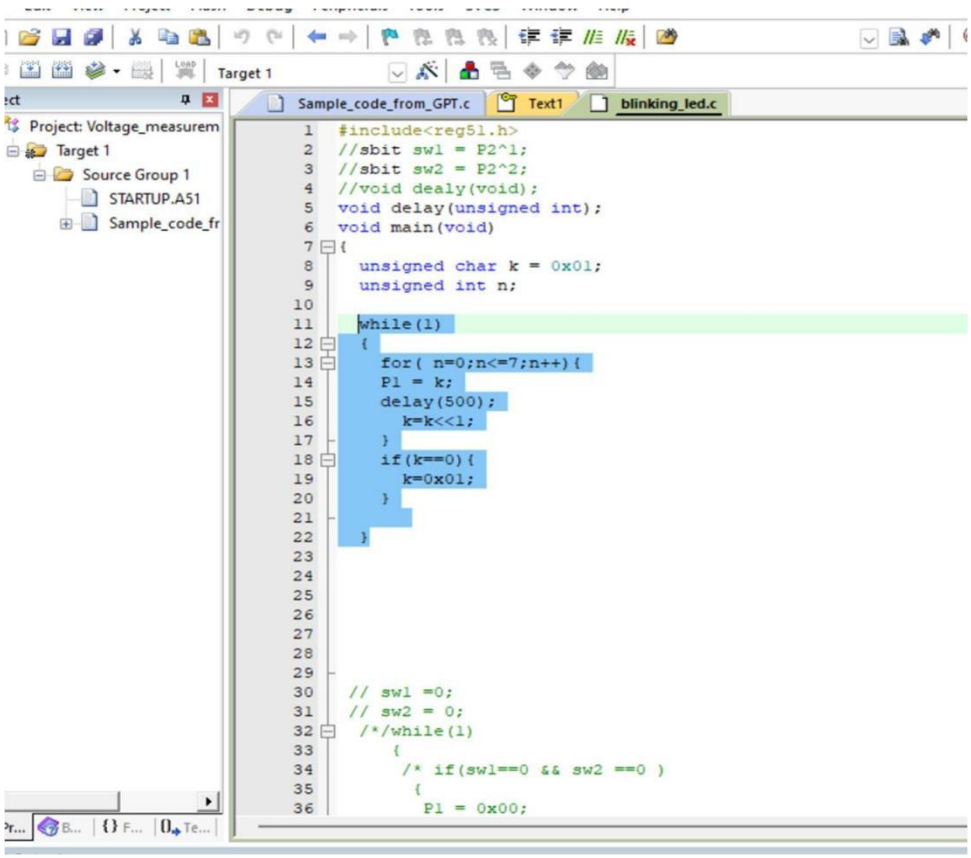
1. Connect LEDs to the Microcontroller:
   - Connect the LEDs to the microcontroller's output pins. Each LED should be connected to a separate pin.
   - The other terminal of each LED should be connected to the ground.

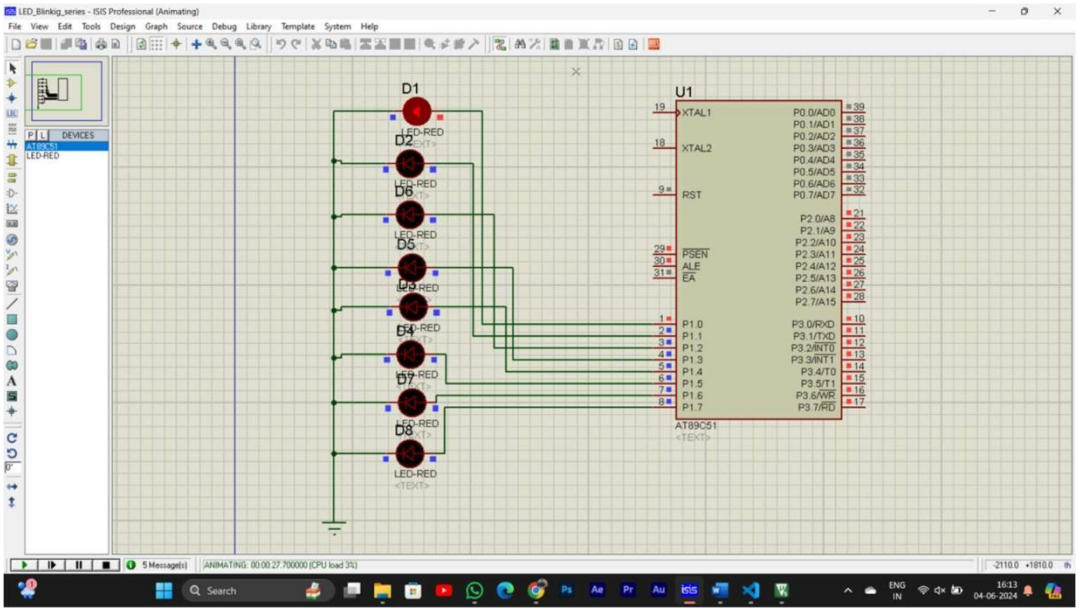
2. Assign Hexadecimal Values to the Port:
   - Assign a hexadecimal value to the microcontroller's port. This value will be converted to binary.
   - Each bit in the binary representation controls an individual LED:
   - A bit value of '1' turns the LED on.
   - A bit value of '0' turns the LED off.

3. Implementing Blinking Patterns:
   - Direct Assignment:
   - Assign different hexadecimal values to the port to create different blinking patterns.
   - Bitwise Left Shift Operations:
   - Use bitwise left shift operations within a loop to create sequential blinking patterns.

- **Common Mistakes:**

  - **Syntax Errors:** Common when writing embedded C code.
  - **Indentation Errors:** Proper code formatting is crucial to avoid logical errors.
  - **Port Mismatch:** Ensuring the correct port is being used for LED control is essential to achieve the desired output.





- **ADC Using 8051**

Learning Objective:

- Understanding the process of converting an analog signal to a digital signal using the ADC 0808.
- Learning how to interface the ADC 0808 with the 8051 microcontroller.

Inputs and Outputs:

- **Input:** A potentiometer (1k or 10k ohms).
- **Outputs:** LEDs.

Logic:

Steps for Interfacing ADC0808 with 8051 Microcontroller:-

1. Connect the Oscillator:
   - Connect the oscillator circuit to pins 19 and 20 of the microcontroller.
   - This includes a crystal oscillator and two capacitors of 22uF each.

2. Connect the ADC0808:
   - Connect the output pins of the ADC0808 to Port 1 of the microcontroller.
   - Connect the selection lines (A, B, C) of the ADC to Port 2 of the microcontroller.
   - Connect the additional control lines (EOC, Start, ALE, and Clock Pulse) to Port 2 of the microcontroller.

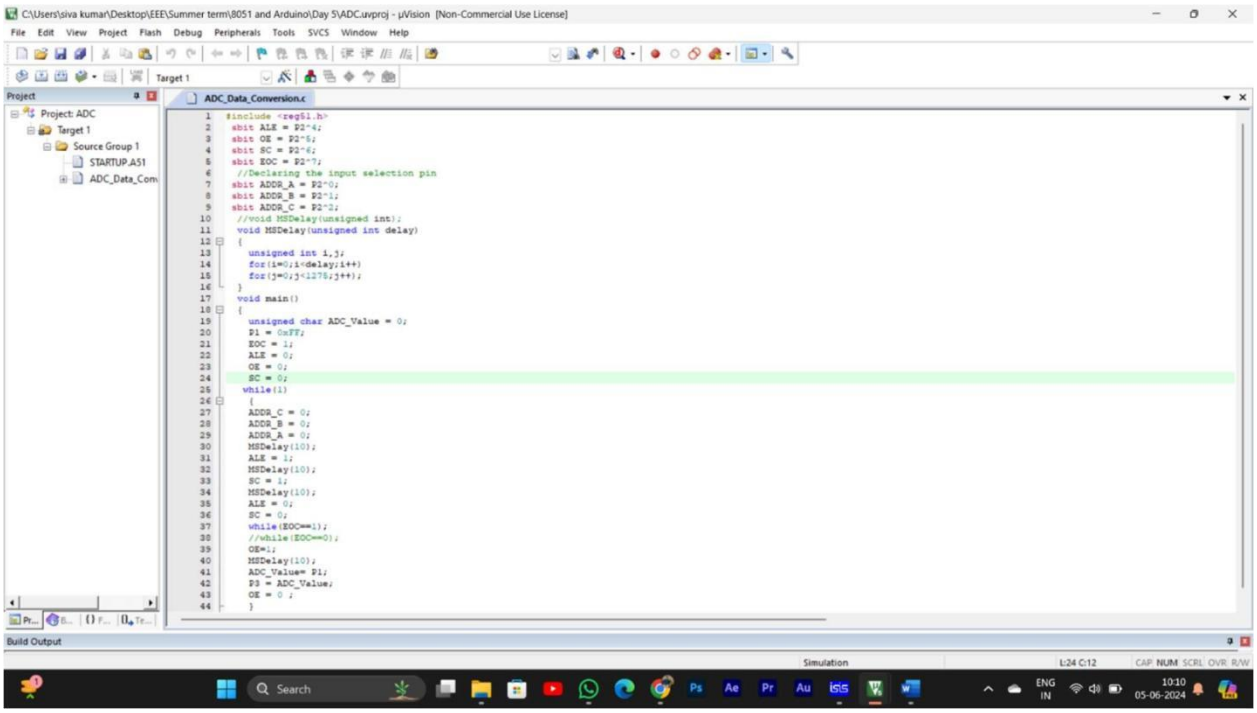
3. Connect the Potentiometer:
   - Connect the analog output from the potentiometer to one of the input channels of the ADC0808.

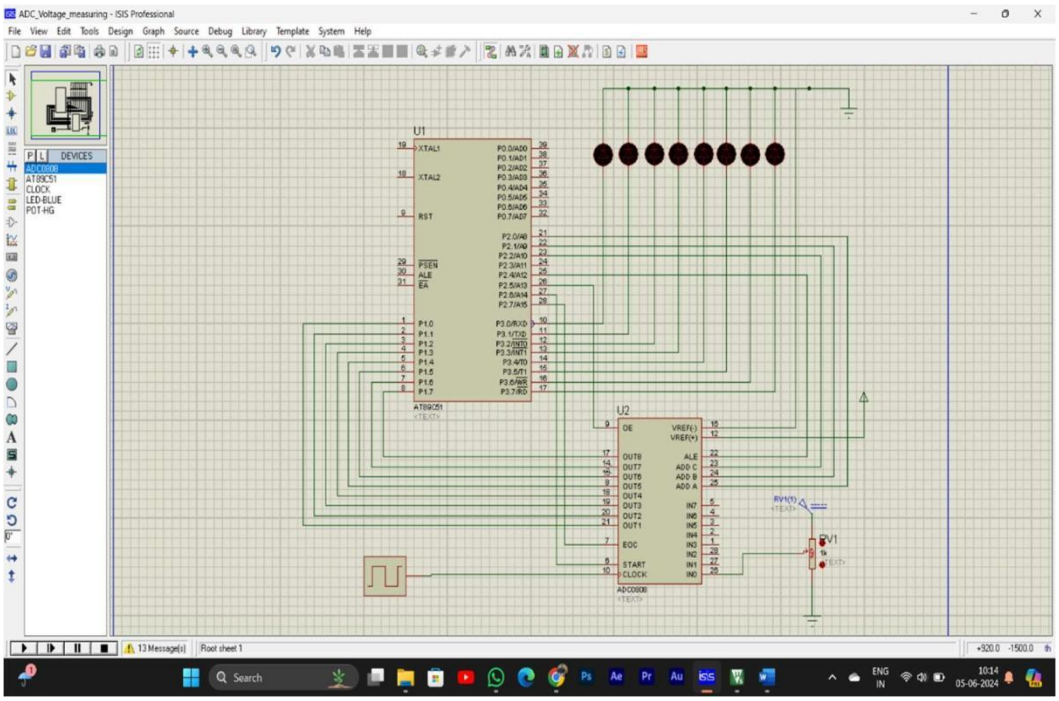
4. Connect the LEDs:
   - Connect the LEDs to Port 3 of the microcontroller.
   - These LEDs will display the digital value of the potentiometer reading.

Common Mistakes:

1. Selection Line Configuration
   - Connect the address lines A, B, and C of the ADC 0808 to port pins of the 8051 to select the desired input channel.
   - For example, connect A to P2.0, B to P2.1, and C to P2.2 on the 8051.
   - To select a specific channel, apply the corresponding 3-bit binary code on A, B, and C.
   - For example, to select channel INT2, set A=0, B=1, and C=0

2. EOC Pin Connection
   - Connect the End of Conversion (EOC) pin of the ADC 0808 to an input pin of the 8051, such as P3.2.
   - The 8051 should monitor the EOC pin to detect when a conversion is complete.
   - After initiating a conversion, the 8051 should wait for EOC to go low, indicating the conversion is finished and the digital output is ready


C:\Users\siva kumar\Desktop\EEE\Summer term\8051 and Arduino\Day 5\ADC.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: ADC
Target 1
Source Group 1
STARTUP.A51
ADC_Data_Con
ADC_Data_Conversion.c
#include <reg51.h>
sbit ALE = P2^4;
sbit OE = P2^5;
sbit SC = P2^6;
sbit EOC = P2^7;
//Declaring the input selection pin
sbit ADDR_A = P2^0;
sbit ADDR_B = P2^1;
sbit ADDR_C = P2^2;
//void MSDelay(unsigned int);
void MSDelay(unsigned int delay)
{
unsigned int i,j;
for(i=0;i<delay;i++)
for(j=0;j<1275;j++);
}
void main()
{
unsigned char ADC_Value = 0;
P1 = 0xFF;
EOC = 1;
ALE = 0;
OE = 0;
SC = 0;
while(1)
{
ADDR_C = 0;
ADDR_B = 0;
ADDR_A = 0;
MSDelay(10);
ALE = 1;
MSDelay(10);
SC = 1;
MSDelay(10);
ALE = 0;
SC = 0;
while(EOC==1);
//while(EOC==0);
OE=1;
MSDelay(10);
ADC_Value= P1;
P3 = ADC_Value;
OE = 0 ;
}
Build Output
Simulation
L:24 C:12
CAP NUM SCRL OVR R/W
Search
ENG
IN
10:10
05-06-2024


ADC_Voltage_measuring - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
DEVICES
ADC0808
AT89C51
CLOCK
LED-BLUE
POT-HG
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
AT89C51
U2
OE
VREF(-)
VREF(+)
OUT8
OUT7
OUT6
OUT5
OUT4
OUT3
OUT2
OUT1
EOC
START
CLOCK
ADC0808
ADD C
ADD B
ADD A
RV1
13 Message(s)
Root sheet 1
Search
ENG
IN
10:14
05-06-2024

- **8051**

Learning Objective:

- Learning how to control LED blinking using the 8051 microcontroller.

Inputs and Outputs:

- **Inputs:** Switches
- **Outputs:** LEDs

- **Logic:**

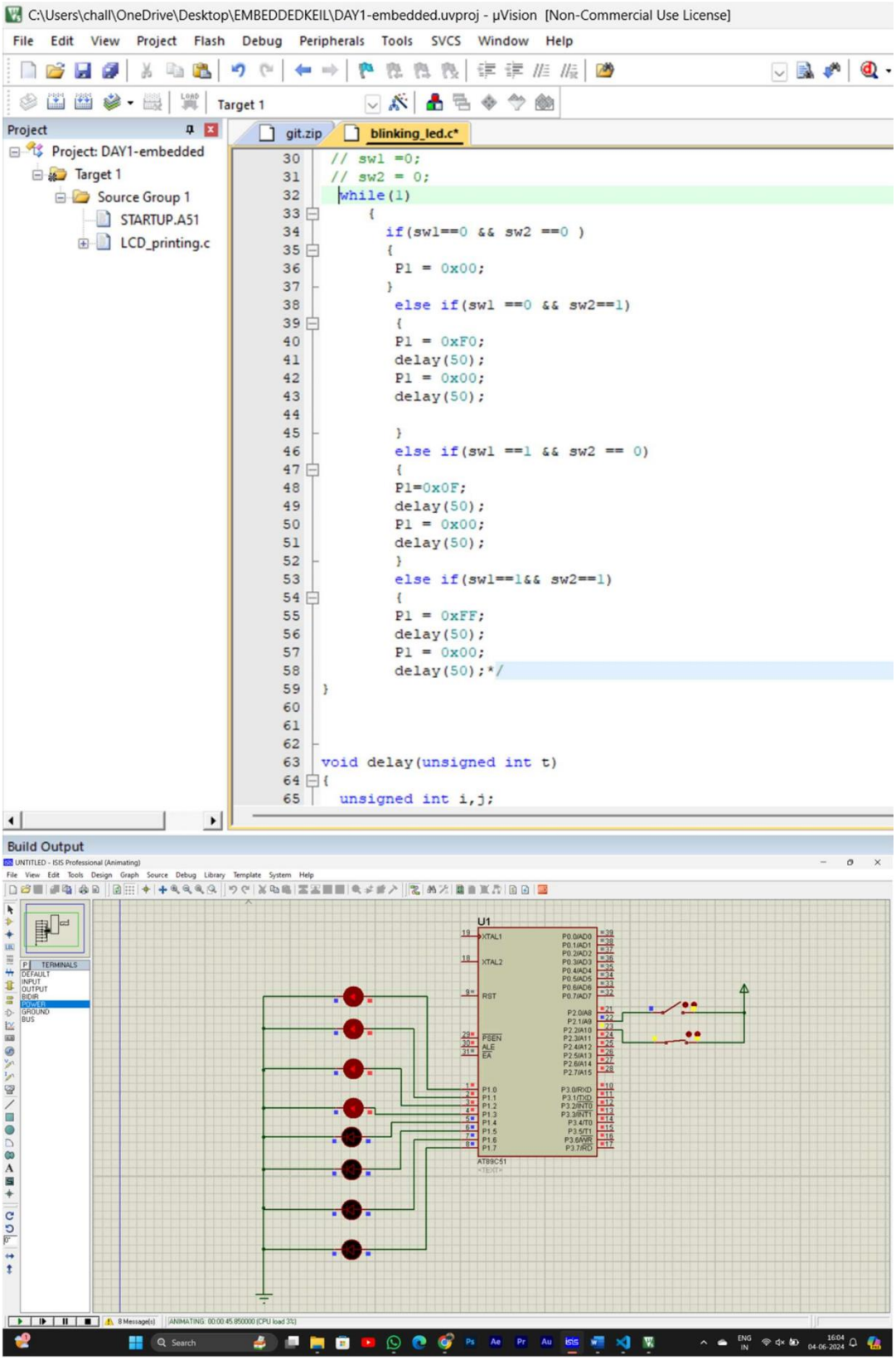
- The code uses the 8051 microcontroller to control LEDs connected to Port 1. It sends hexadecimal values to Port 1 to determine the state of each LED. The blinking effect is achieved by using a delay function to toggle the LEDs on and off.

Common Mistakes:

- Project creation errors:
  Sometimes the project is not properly generated or configured, leading to missing components Or incorrect settings. This can prevent the project from building successfully in Keil

- Syntax and indentation errors:
  Mistakes in the C code syntax or improper indentation can cause compilation errors when trying To build the project.

- Incorrect microcontroller selection:
  - If the wrong microcontroller is selected for the target board, it will not match the actual hardware and cause issues.
  -
- LED connection errors:
  - Connecting the LEDs to the wrong ports on the microcontroller can prevent the code from controlling them as expected


C:\Users\chall\OneDrive\Desktop\EMBEDDEDKEIL\DAY1-embedded.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: DAY1-embedded
Target 1
Source Group 1
STARTUP.A51
LCD_printing.c
git.zip
blinking_led.c*
// sw1 =0;
// sw2 = 0;
while(1)
{
if(sw1==0 && sw2 ==0)
{
P1 = 0x00;
}
else if(sw1 ==0 && sw2==1)
{
P1 = 0xF0;
delay(50);
P1 = 0x00;
delay(50);
}
else if(sw1 ==1 && sw2 == 0)
{
P1=0x0F;
delay(50);
P1 = 0x00;
delay(50);
}
else if(sw1==1&& sw2==1)
{
P1 = 0xFF;
delay(50);
P1 = 0x00;
delay(50);*/
}
void delay(unsigned int t)
{
unsigned int i,j;
Build Output
UNTITLED - ISIS Professional (Animating)
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
AT89C51

- **Voltage Measurement and LCD displaying(8051)**

Learning Objective:

- Understanding how to measure voltage variations using an ADC.
- Displaying the voltage variations on LEDs.
- Displaying the voltage measurement status on an LCD display.

Inputs and Outputs:

- **Input:** Potentiometer.
- **Outputs:** LEDs and LCD display.

- **Logic:**

1.Connections:

- Connect the potentiometer output to the ADC 0808 input:
  Connect the output of the potentiometer to one of the input channels of the ADC 0808. This will allow the ADC to read the analog signal from the potentiometer.

- Provide a 5V power supply to the ADC 0808: Connect a 5V power supply to the ADC 0808 to power it.
  Connect ADC 0808 output to 8051 microcontroller port:
  Connect the digital output pins (D0-D7) of the ADC 0808 to a port on the 8051 microcontroller. This will allow the microcontroller to read the digital output from the ADC.

- Connect LCD display and LEDs to 8051 microcontroller:
  Connect the LCD display and LEDs to the 8051 microcontroller. This will allow the microcontroller to control the LCD display and LEDs.

2.Operation:

- Read the digital signal from the ADC:

The 8051 microcontroller reads the digital signal from the ADC, which represents the analog signal from the potentiometer.

- Convert the digital value to a corresponding voltage value:

The microcontroller converts the digital value read from the ADC to a corresponding voltage value. This can be done by using a lookup table or a simple calculation based on the ADC resolution and the full-scale voltage range.

- Control the LEDs based on the voltage level:

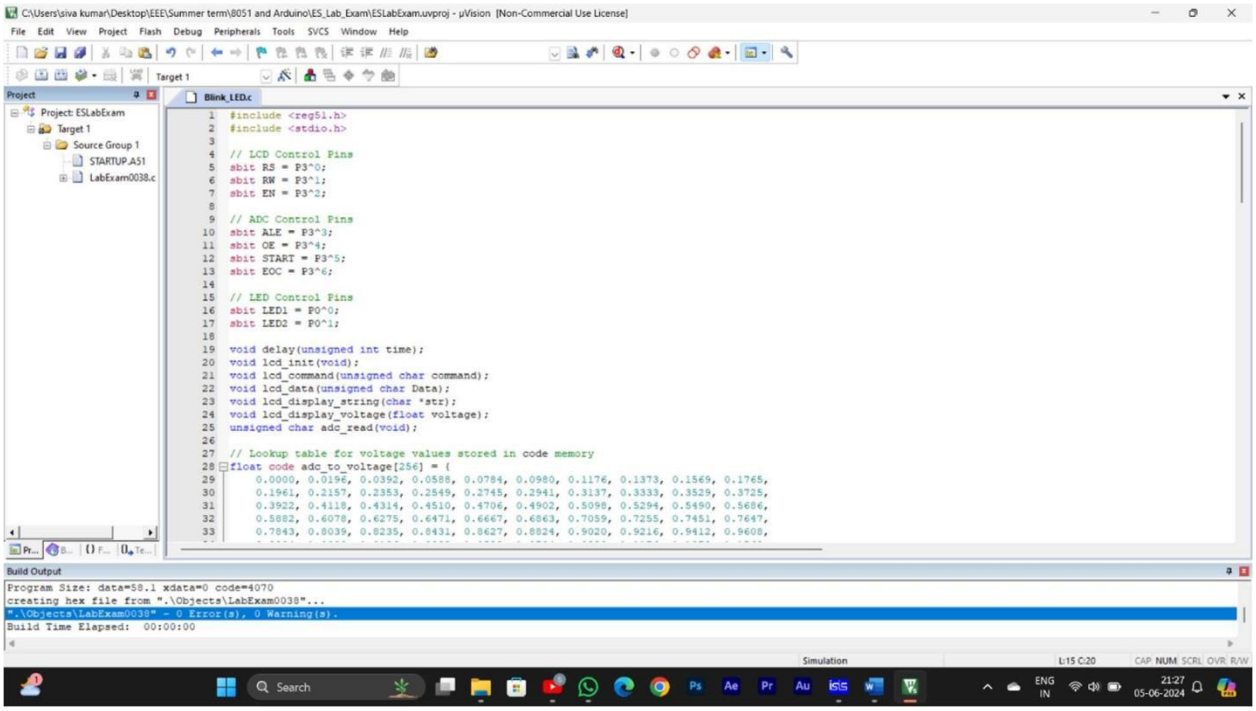
The microcontroller controls the LEDs to indicate the voltage levels based on the converted voltage value. For example, different LEDs can be turned on or off to represent different voltage ranges.

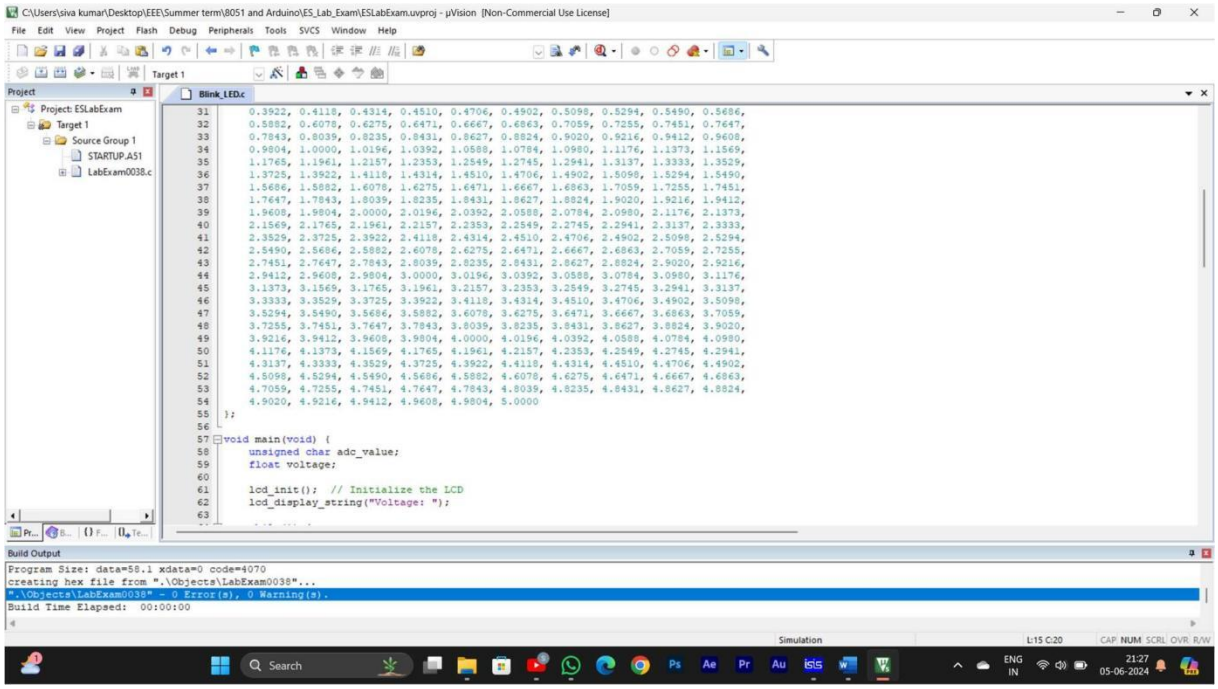
- Display the voltage value on the LCD:

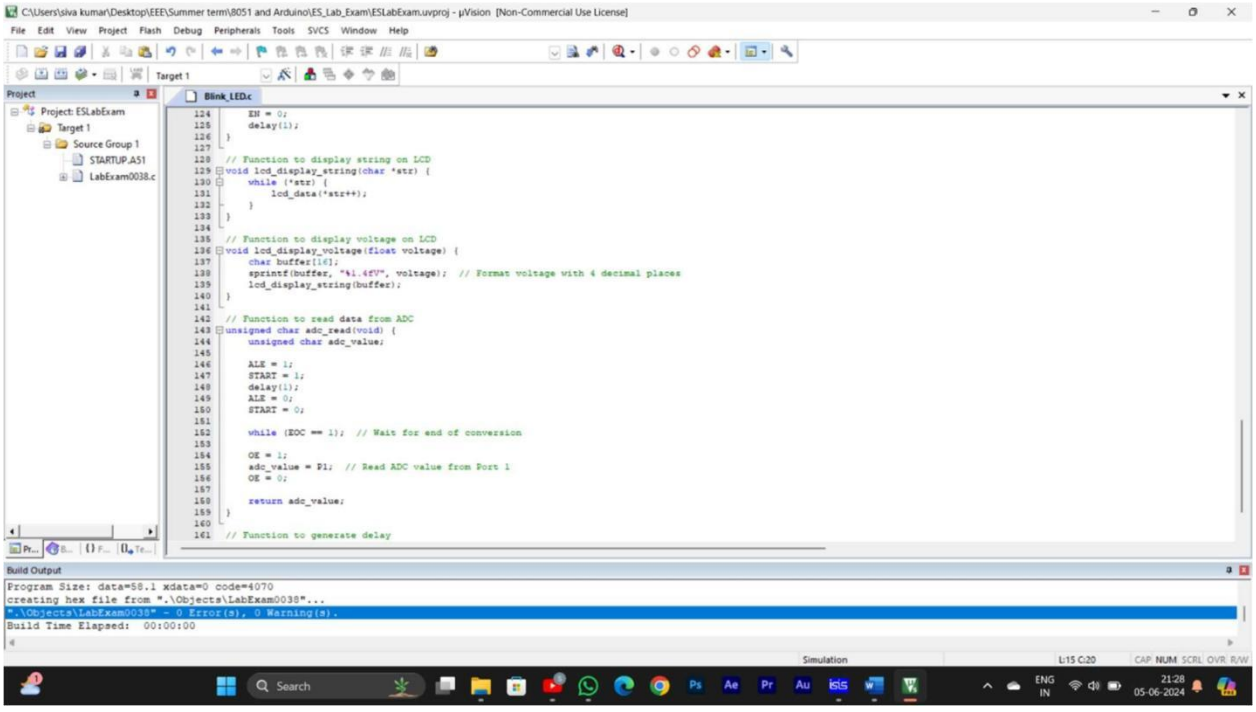
The microcontroller displays the converted voltage value on the LCD display. This can be done by sending the value to the LCD and formatting it according to the LCD's display capabilities.

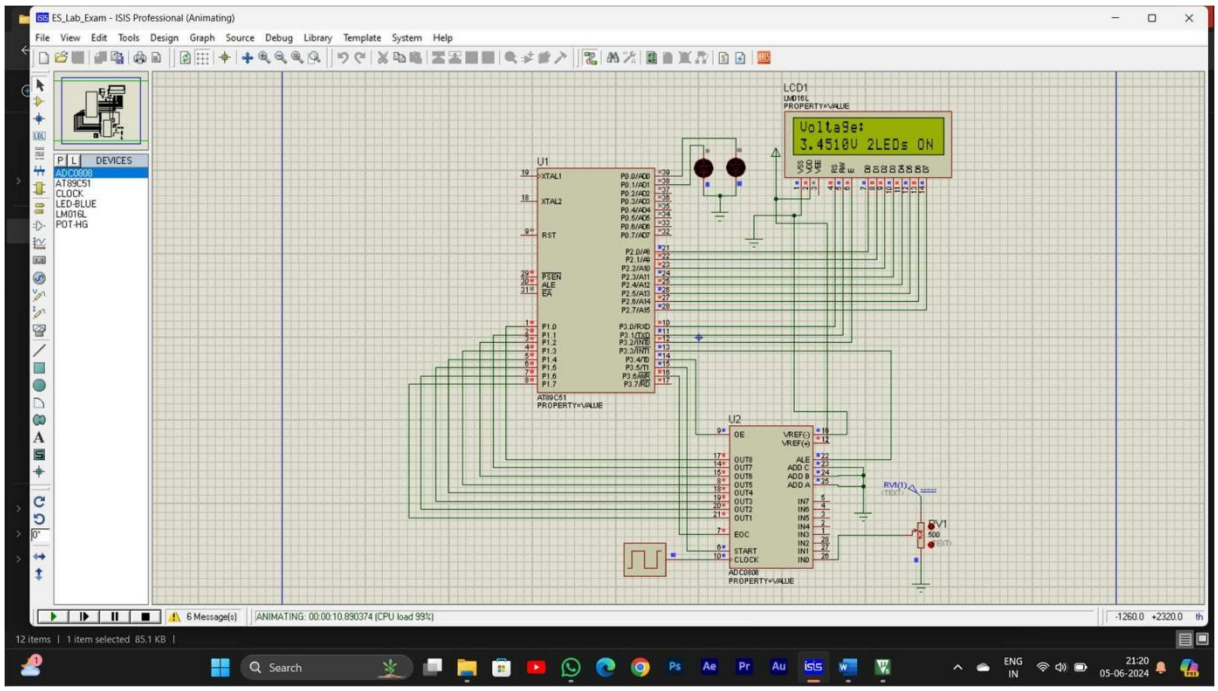
- **Common Mistakes:**

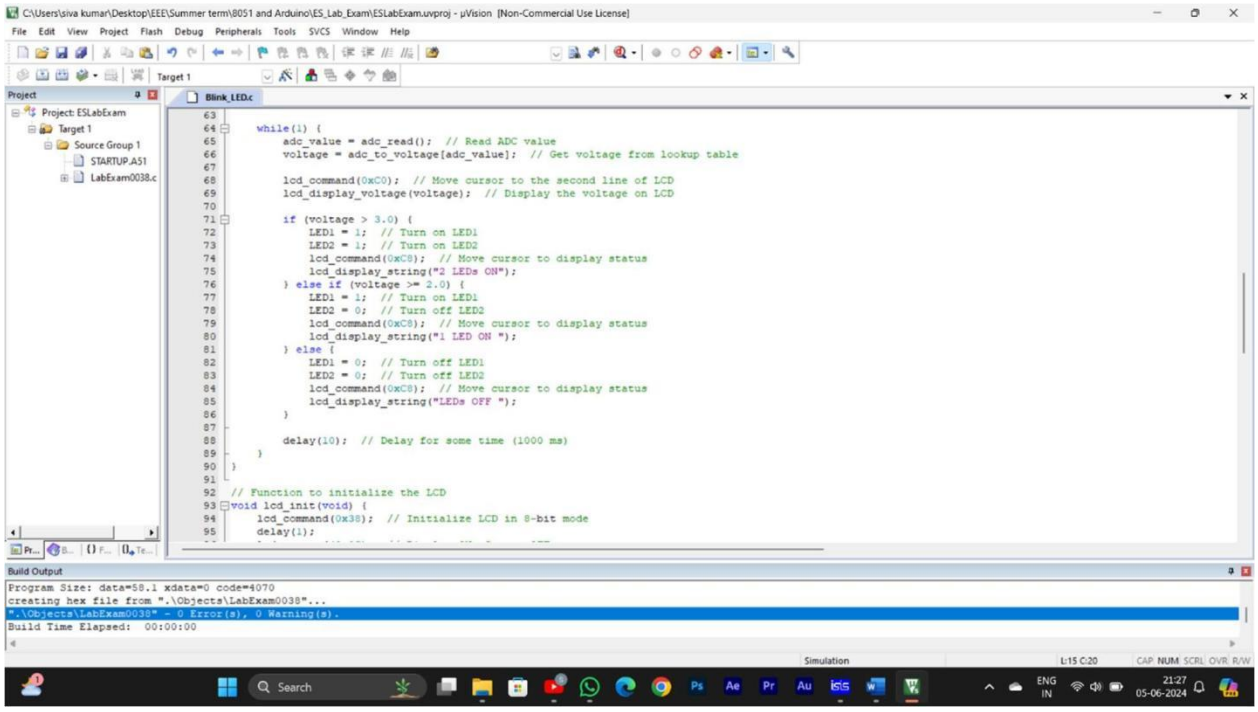
    - **Port Mismatch:** Ensure that the port names used in the code match the actual circuit connections.
    - **Proper Connections:** Verify that all connections are secure. A single loose connection can lead to data loss and no output.
    - **Input Levels:** Ensure that input levels to the microcontroller and ADC are correct (either high or low as required).

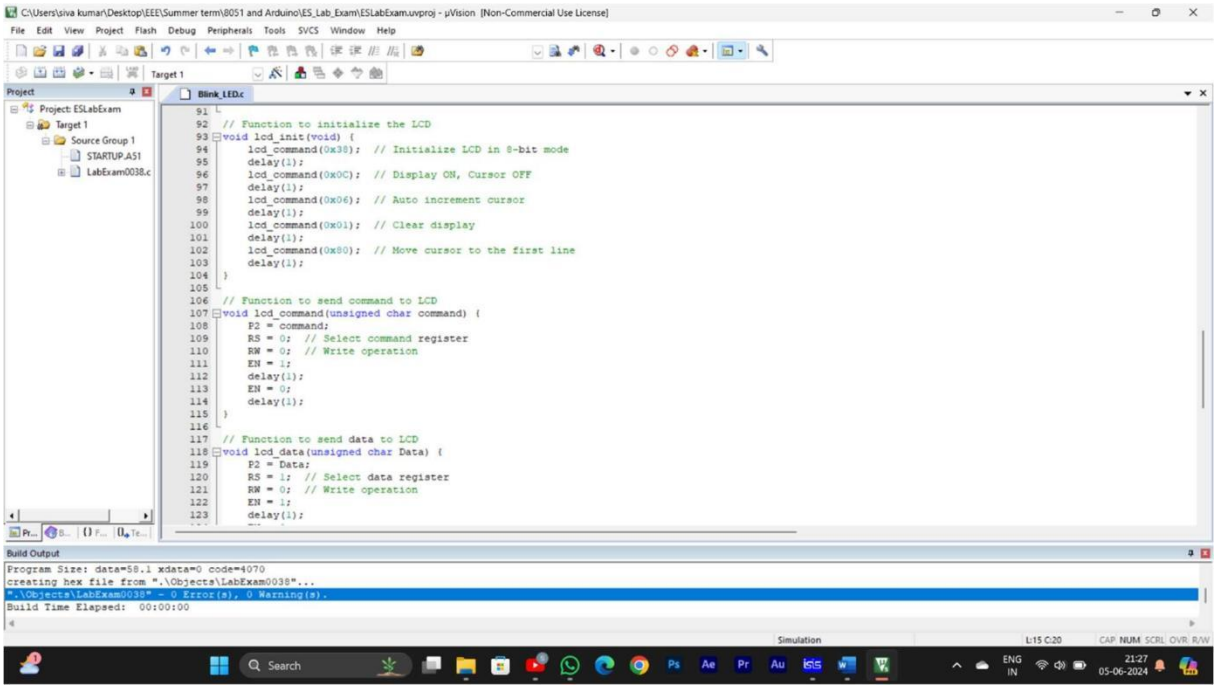



C:\Users\siva kumar\Desktop\EEE\Summer term\8051 and Arduino\ES_Lab_Exam\ESLabExam.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: ESLabExam
Target 1
Source Group 1
STARTUP.A51
LabExam0038.c
Blink_LED.c
0.3922, 0.4118, 0.4314, 0.4510, 0.4706, 0.4902, 0.5098, 0.5294, 0.5490, 0.5686,
0.5882, 0.6078, 0.6275, 0.6471, 0.6667, 0.6863, 0.7059, 0.7255, 0.7451, 0.7647,
0.7843, 0.8039, 0.8235, 0.8431, 0.8627, 0.8824, 0.9020, 0.9216, 0.9412, 0.9608,
0.9804, 1.0000, 1.0196, 1.0392, 1.0588, 1.0784, 1.0980, 1.1176, 1.1373, 1.1569,
1.1765, 1.1961, 1.2157, 1.2353, 1.2549, 1.2745, 1.2941, 1.3137, 1.3333, 1.3529,
1.3725, 1.3922, 1.4118, 1.4314, 1.4510, 1.4706, 1.4902, 1.5098, 1.5294, 1.5490,
1.5686, 1.5882, 1.6078, 1.6275, 1.6471, 1.6667, 1.6863, 1.7059, 1.7255, 1.7451,
1.7647, 1.7843, 1.8039, 1.8235, 1.8431, 1.8627, 1.8824, 1.9020, 1.9216, 1.9412,
1.9608, 1.9804, 2.0000, 2.0196, 2.0392, 2.0588, 2.0784, 2.0980, 2.1176, 2.1373,
2.1569, 2.1765, 2.1961, 2.2157, 2.2353, 2.2549, 2.2745, 2.2941, 2.3137, 2.3333,
2.3529, 2.3725, 2.3922, 2.4118, 2.4314, 2.4510, 2.4706, 2.4902, 2.5098, 2.5294,
2.5490, 2.5686, 2.5882, 2.6078, 2.6275, 2.6471, 2.6667, 2.6863, 2.7059, 2.7255,
2.7451, 2.7647, 2.7843, 2.8039, 2.8235, 2.8431, 2.8627, 2.8824, 2.9020, 2.9216,
2.9412, 2.9608, 2.9804, 3.0000, 3.0196, 3.0392, 3.0588, 3.0784, 3.0980, 3.1176,
3.1373, 3.1569, 3.1765, 3.1961, 3.2157, 3.2353, 3.2549, 3.2745, 3.2941, 3.3137,
3.3333, 3.3529, 3.3725, 3.3922, 3.4118, 3.4314, 3.4510, 3.4706, 3.4902, 3.5098,
3.5294, 3.5490, 3.5686, 3.5882, 3.6078, 3.6275, 3.6471, 3.6667, 3.6863, 3.7059,
3.7255, 3.7451, 3.7647, 3.7843, 3.8039, 3.8235, 3.8431, 3.8627, 3.8824, 3.9020,
3.9216, 3.9412, 3.9608, 3.9804, 4.0000, 4.0196, 4.0392, 4.0588, 4.0784, 4.0980,
4.1176, 4.1373, 4.1569, 4.1765, 4.1961, 4.2157, 4.2353, 4.2549, 4.2745, 4.2941,
4.3137, 4.3333, 4.3529, 4.3725, 4.3922, 4.4118, 4.4314, 4.4510, 4.4706, 4.4902,
4.5098, 4.5294, 4.5490, 4.5686, 4.5882, 4.6078, 4.6275, 4.6471, 4.6667, 4.6863,
4.7059, 4.7255, 4.7451, 4.7647, 4.7843, 4.8039, 4.8235, 4.8431, 4.8627, 4.8824,
4.9020, 4.9216, 4.9412, 4.9608, 4.9804, 5.0000
};
void main(void) {
unsigned char adc_value;
float voltage;
lcd_init(); // Initialize the LCD
lcd_display_string("Voltage: ");
Build Output
Program Size: data=58.1 xdata=0 code=4070
creating hex file from ".\Objects\LabExam0038"...
".\Objects\LabExam0038" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00
Simulation
L:15 C:20
CAP NUM SCRL OVR R/W
Search
ENG
IN
21:27
05-06-2024


C:\Users\siva kumar\Desktop\EEE\Summer term\8051 and Arduino\ES_Lab_Exam\ESLabExam.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: ESLabExam
Target 1
Source Group 1
STARTUP.A51
LabExam0038.c
Blink_LED.c
EN = 0;
delay(1);
}
// Function to display string on LCD
void lcd_display_string(char *str) {
while (*str) {
lcd_data(*str++);
}
}
// Function to display voltage on LCD
void lcd_display_voltage(float voltage) {
char buffer[16];
sprintf(buffer, "%1.4fV", voltage); // Format voltage with 4 decimal places
lcd_display_string(buffer);
}
// Function to read data from ADC
unsigned char adc_read(void) {
unsigned char adc_value;
ALE = 1;
START = 1;
delay(1);
ALE = 0;
START = 0;
while (EOC == 1); // Wait for end of conversion
OE = 1;
adc_value = P1; // Read ADC value from Port 1
OE = 0;
return adc_value;
}
// Function to generate delay
Build Output
Program Size: data=58.1 xdata=0 code=4070
creating hex file from ".\Objects\LabExam0038"...
".\Objects\LabExam0038" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00
Simulation
L:15 C:20
CAP NUM SCRL OVR R/W
Search
ENG
IN
21:28
05-06-2024


ES_Lab_Exam - ISIS Professional (Animating)
File View Edit Tools Design Graph Source Debug Library Template System Help
LCD1
Voltage:
3.4510V 2LEDs ON
U1
AT89C51
U2
ADC0808
6 Message(s)
ANIMATING: 00:00:10.890374 (CPU load 99%)
12 items | 1 item selected 85.1 KB
Search
ENG
IN
21:20
05-06-2024


C:\Users\siva kumar\Desktop\EEE\Summer term\8051 and Arduino\ES_Lab_Exam\ESLabExam.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: ESLabExam
Target 1
Source Group 1
STARTUP.A51
LabExam0038.c
Blink_LED.c
while(1) {
adc_value = adc_read(); // Read ADC value
voltage = adc_to_voltage[adc_value]; // Get voltage from lookup table
lcd_command(0xC0); // Move cursor to the second line of LCD
lcd_display_voltage(voltage); // Display the voltage on LCD
if (voltage > 3.0) {
LED1 = 1; // Turn on LED1
LED2 = 1; // Turn on LED2
lcd_command(0xC8); // Move cursor to display status
lcd_display_string("2 LEDs ON");
} else if (voltage >= 2.0) {
LED1 = 1; // Turn on LED1
LED2 = 0; // Turn off LED2
lcd_command(0xC8); // Move cursor to display status
lcd_display_string("1 LED ON ");
} else {
LED1 = 0; // Turn off LED1
LED2 = 0; // Turn off LED2
lcd_command(0xC8); // Move cursor to display status
lcd_display_string("LEDs OFF ");
}
delay(10); // Delay for some time (1000 ms)
}
}
// Function to initialize the LCD
void lcd_init(void) {
lcd_command(0x38); // Initialize LCD in 8-bit mode
delay(1);
Build Output
Program Size: data=58.1 xdata=0 code=4070
creating hex file from ".\Objects\LabExam0038"...
".\Objects\LabExam0038" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00
Simulation
L:15 C:20
Search
ENG
IN
21:27
05-06-2024



8051(micro-controller) using LCD Display

➢ Learning Objective:

- Learning how to use the 8051 microcontroller to display text on an LCD.

➢ Inputs and Outputs:

- **Inputs:** Commands for the LCD display.
- **Output:** Displayed text on the LCD.

- **Logic:**

➢ Implement functions to send commands and data to the LCD:

Write functions in the microcontroller code to send commands and data to the LCD. This includes sending commands to clear the display, move the cursor, and shift lines.

➢ Use LCD commands: Use various LCD commands such as:

Clear display: Send a command to clear the display.
Cursor positioning: Send a command to move the cursor to a specific position.
Line shifting: Send a command to shift lines up or down.

➢ Convert numerical values to ASCII: Convert numerical values to ASCII characters before displaying them on the LCD. This ensures that the LCD can correctly display the numbers.

➢ Make basic connections: Make basic connections to:

Power supply: Connect the LCD to the power supply.
Ground: Connect the LCD to ground.
Enable pins: Connect the LCD enable pins to the microcontroller pins.

- **Common Mistakes:**

➢ Command Mismatch:

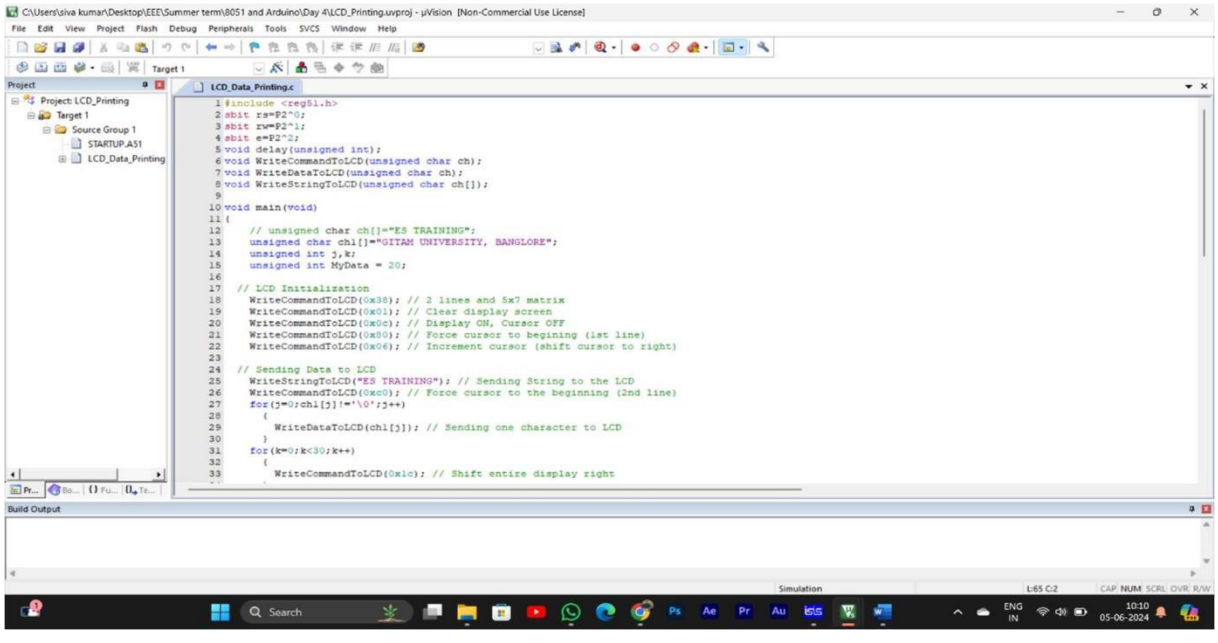
Ensure that the correct commands are sent to the LCD. This includes commands for clearing the display, moving the cursor, and displaying data.

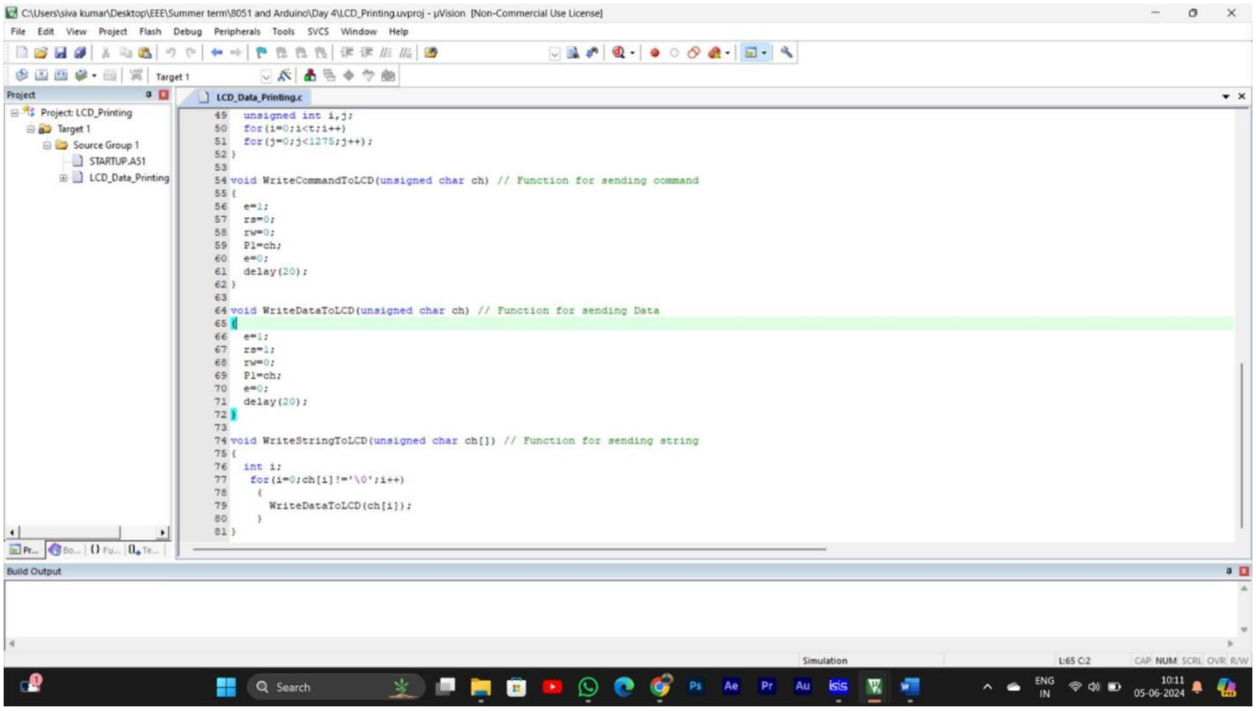
➢ Command Positioning:

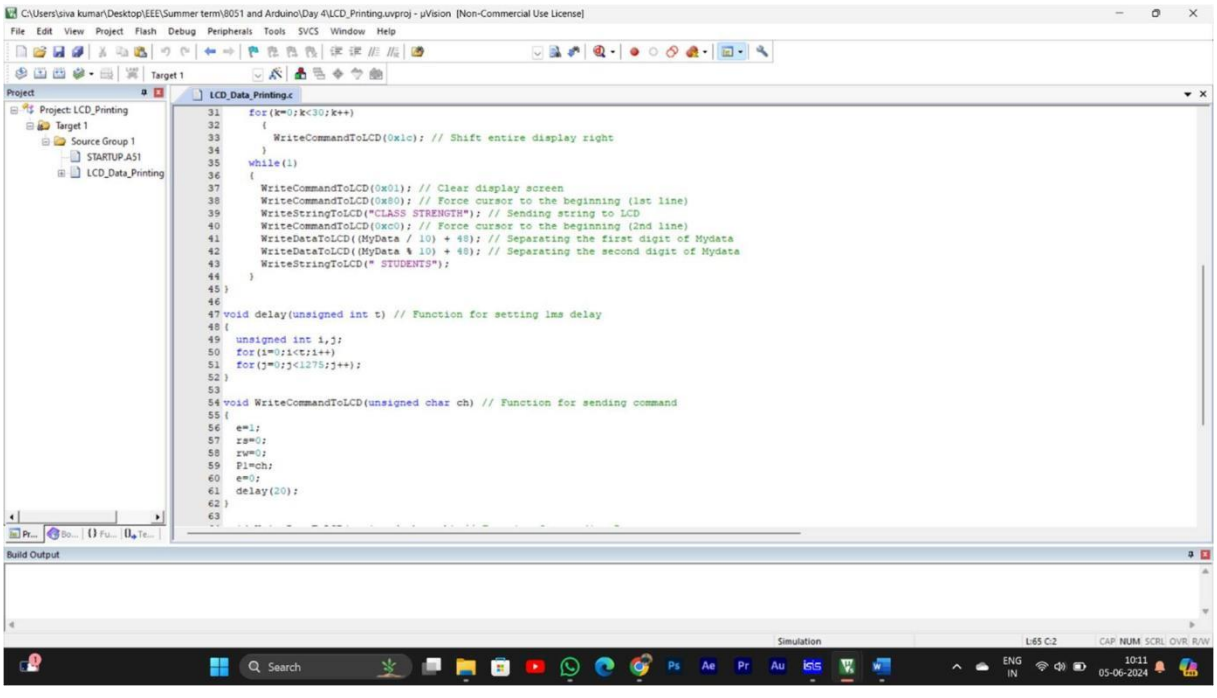
Correct positioning of commands is crucial for proper display. This includes ensuring that commands are sent in the correct order and that the LCD is properly initialized.

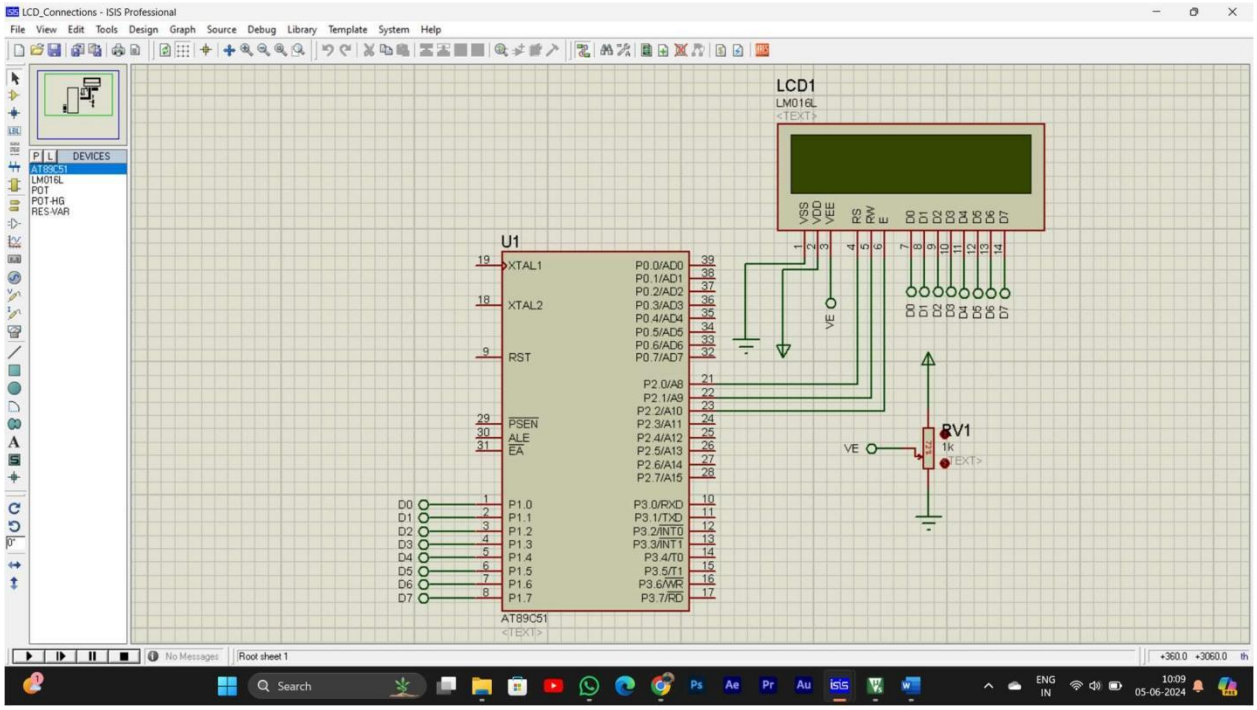
➢ Clear Display and New Line Commands:

Use appropriate commands to clear the display or move to a new line when needed. This includes commands like clear display and new line to manage the display effectively.




C:\Users\siva kumar\Desktop\EEE\Summer term\8051 and Arduino\Day 4\LCD_Printing.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: LCD_Printing
Target 1
Source Group 1
STARTUP.A51
LCD_Data_Printing
LCD_Data_Printing.c
49 unsigned int i,j;
50 for(i=0;i<t;i++)
51 for(j=0;j<1275;j++);
52 }
53
54 void WriteCommandToLCD(unsigned char ch) // Function for sending command
55 {
56 e=1;
57 rs=0;
58 rw=0;
59 P1=ch;
60 e=0;
61 delay(20);
62 }
63
64 void WriteDataToLCD(unsigned char ch) // Function for sending Data
65 {
66 e=1;
67 rs=1;
68 rw=0;
69 P1=ch;
70 e=0;
71 delay(20);
72 }
73
74 void WriteStringToLCD(unsigned char ch[]) // Function for sending string
75 {
76 int i;
77 for(i=0;ch[i]!='\0';i++)
78 {
79 WriteDataToLCD(ch[i]);
80 }
81 }
Build Output
Simulation
L:65 C:2
CAP NUM SCRL OVR R/W
Search
ENG
IN
10:11
05-06-2024


C:\Users\siva kumar\Desktop\EEE\Summer term\8051 and Arduino\Day 4\LCD_Printing.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: LCD_Printing
Target 1
Source Group 1
STARTUP.A51
LCD_Data_Printing
LCD_Data_Printing.c
31 for(k=0;k<30;k++)
32 {
33 WriteCommandToLCD(0x1c); // Shift entire display right
34 }
35 while(1)
36 {
37 WriteCommandToLCD(0x01); // Clear display screen
38 WriteCommandToLCD(0x80); // Force cursor to the beginning (1st line)
39 WriteStringToLCD("CLASS STRENGTH"); // Sending string to LCD
40 WriteCommandToLCD(0xc0); // Force cursor to the beginning (2nd line)
41 WriteDataToLCD((MyData / 10) + 48); // Separating the first digit of Mydata
42 WriteDataToLCD((MyData % 10) + 48); // Separating the second digit of Mydata
43 WriteStringToLCD(" STUDENTS");
44 }
45 }
46
47 void delay(unsigned int t) // Function for setting 1ms delay
48 {
49 unsigned int i,j;
50 for(i=0;i<t;i++)
51 for(j=0;j<1275;j++);
52 }
53
54 void WriteCommandToLCD(unsigned char ch) // Function for sending command
55 {
56 e=1;
57 rs=0;
58 rw=0;
59 P1=ch;
60 e=0;
61 delay(20);
62 }
63
Build Output
Simulation
L:65 C:2
CAP NUM SCRL OVR R/W
Search
ENG
IN
10:11
05-06-2024


LCD_Connections - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
DEVICES
AT89C51
LM016L
POT
POT-HG
RES-VAR
LCD1
LM016L
<TEXT>
VSS VDD VEE RS RW E D0 D1 D2 D3 D4 D5 D6 D7
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
<TEXT>
VE
RV1
1k
<TEXT>
No Messages
Root sheet 1
+360.0 +3060.0 th
Search
ENG IN
10:09
05-06-2024