8051 Mini Development Board Kit

An alternative to Arduino Uno, for Studying Embedded Systems hands-on

Reference Manual

AMOTECH LABS

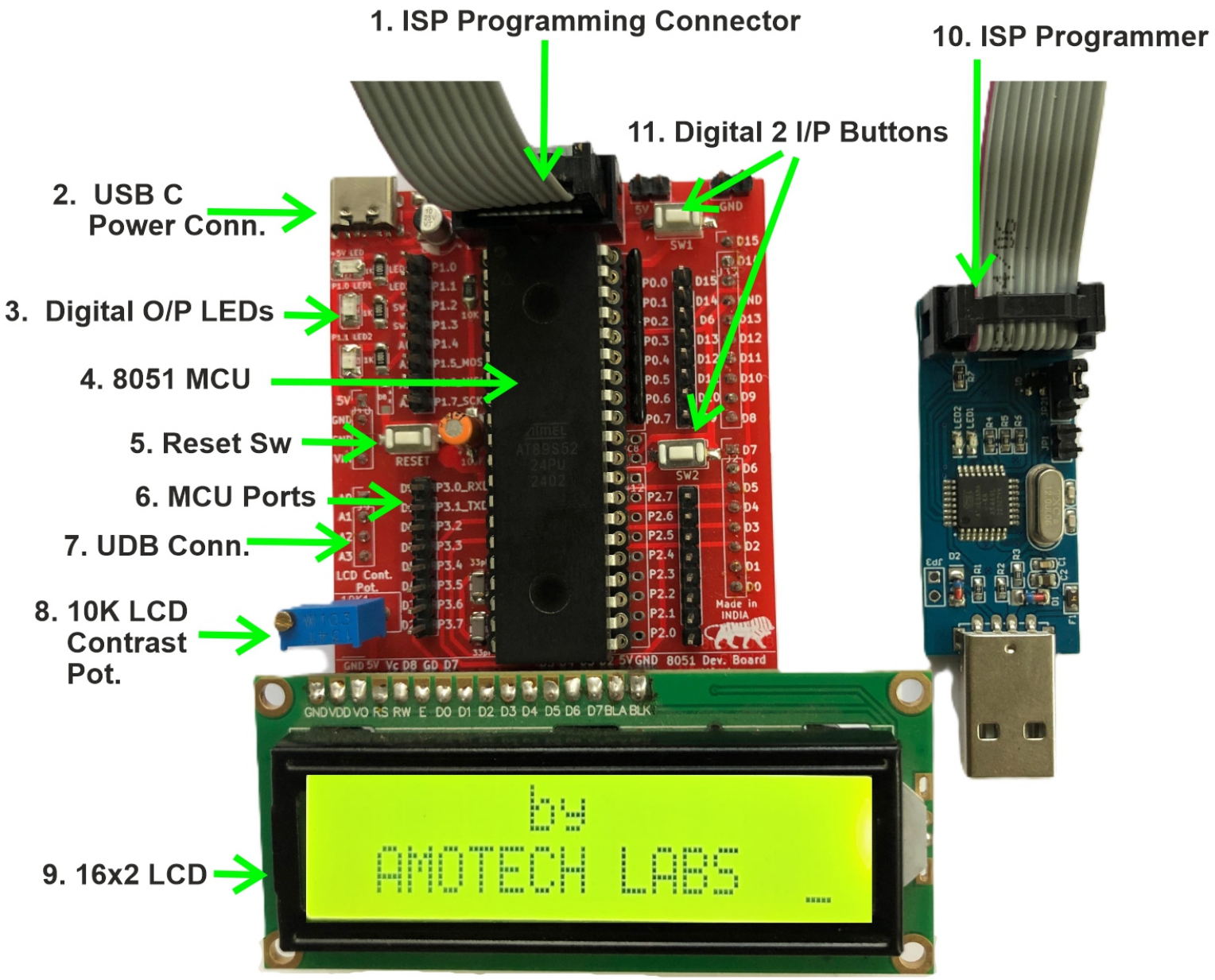
Embedded systems | Industrial Automation | Robotics

Scan to download Soft Copy of Manual & Online Purchase

Scan for Video Tutorials

Contents

Overview of 8051 Mini Development Board



1. ISP Programming Conn : A socket for the ISP Programmer, which allows easy insertion/removal or in-system programmer.

2. Power I/P (Power Input): This is the power supply input to the board, providing necessary +5V voltage from USB C connector to power the board when programmer is not connected.

4. 8051 MCU: Any 8051 based 40 Pin DIP IC can be used with this board. This board is equipped with 89S52.

7. UDB Conn: These connectors are used to attach this kit to 'Universal Development Board(UDB)' by Amotech Labs. Check UDB details on our website.

11. Digital 2 I/P Button Switches: 2 Push buttons serve as digital input switches connected to P1.2 and P1.3. When configured as Inputs, they are pulled up at level High. On pressing, they are connected to ground and making Pins State as Low.

‘8051 Mini Development Board’ can be mounted on below UDB Kit:

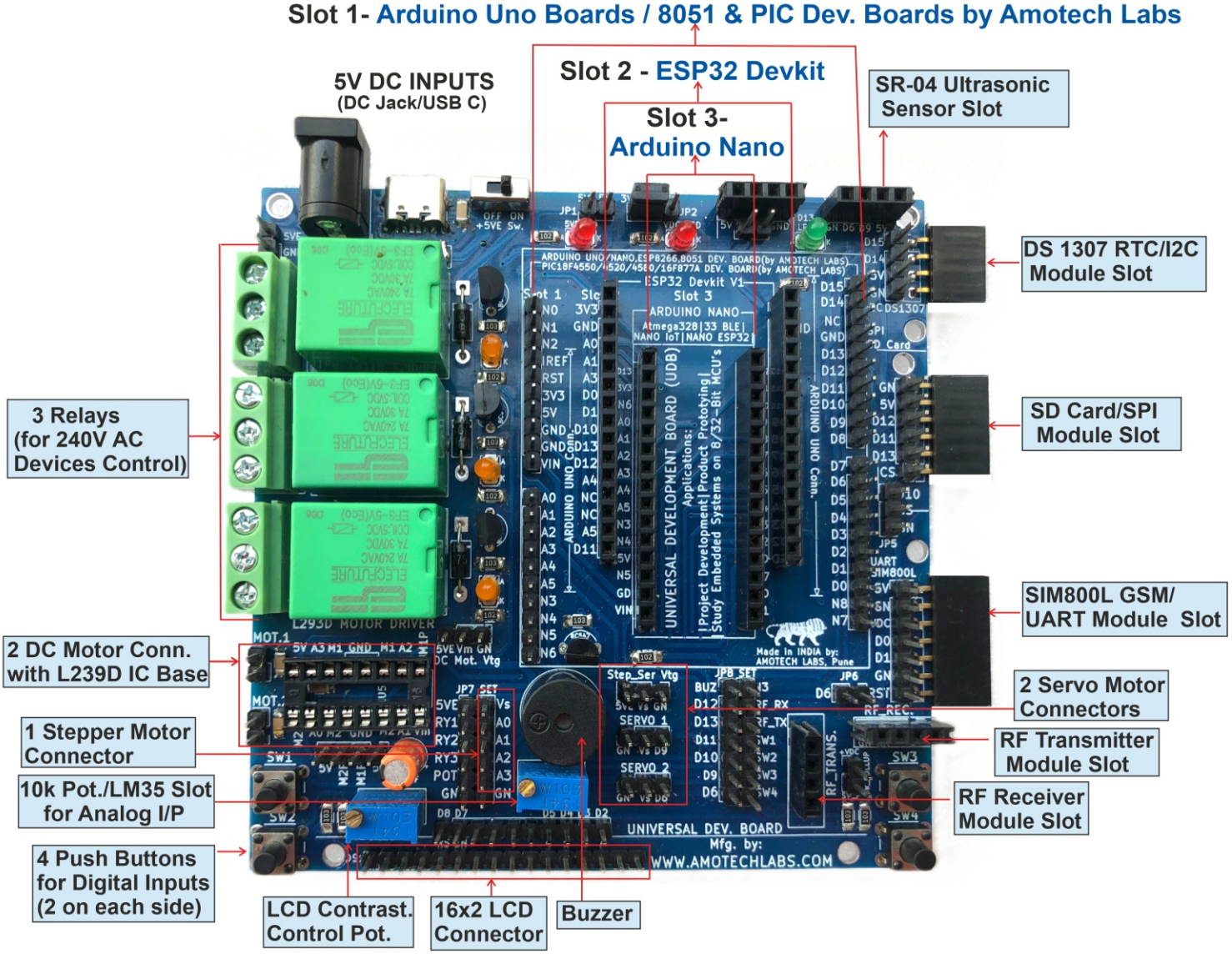
Universal Development Board(UDB) kit

One Development for

Studying Embedded Systems | Project Development | Product Prototyping

on below multiple 8 & 32-Bit Microcontrollers

8051 Shield & PIC18F/16F Dev. Boards | Arduino Uno | ESP32 Devkit V1 | Arduino/STM32 Nano Boards
(Made by Amotech Labs)

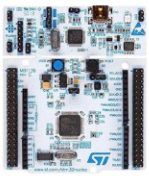


Below Micro-controller Shields/Boards can be inserted into UDB and interfaced with all above Peripherals on it

Slot 1

Arduino Uno

STM32F0/F4 Nucleo Boards



8051 & PIC18F/16 Mini Dev. Boards by Amotech Labs

8051 & PIC18F/16 Dev. Boards by Amotech Labs

Slot 2

ESP32 Devkit V1

Slot 3

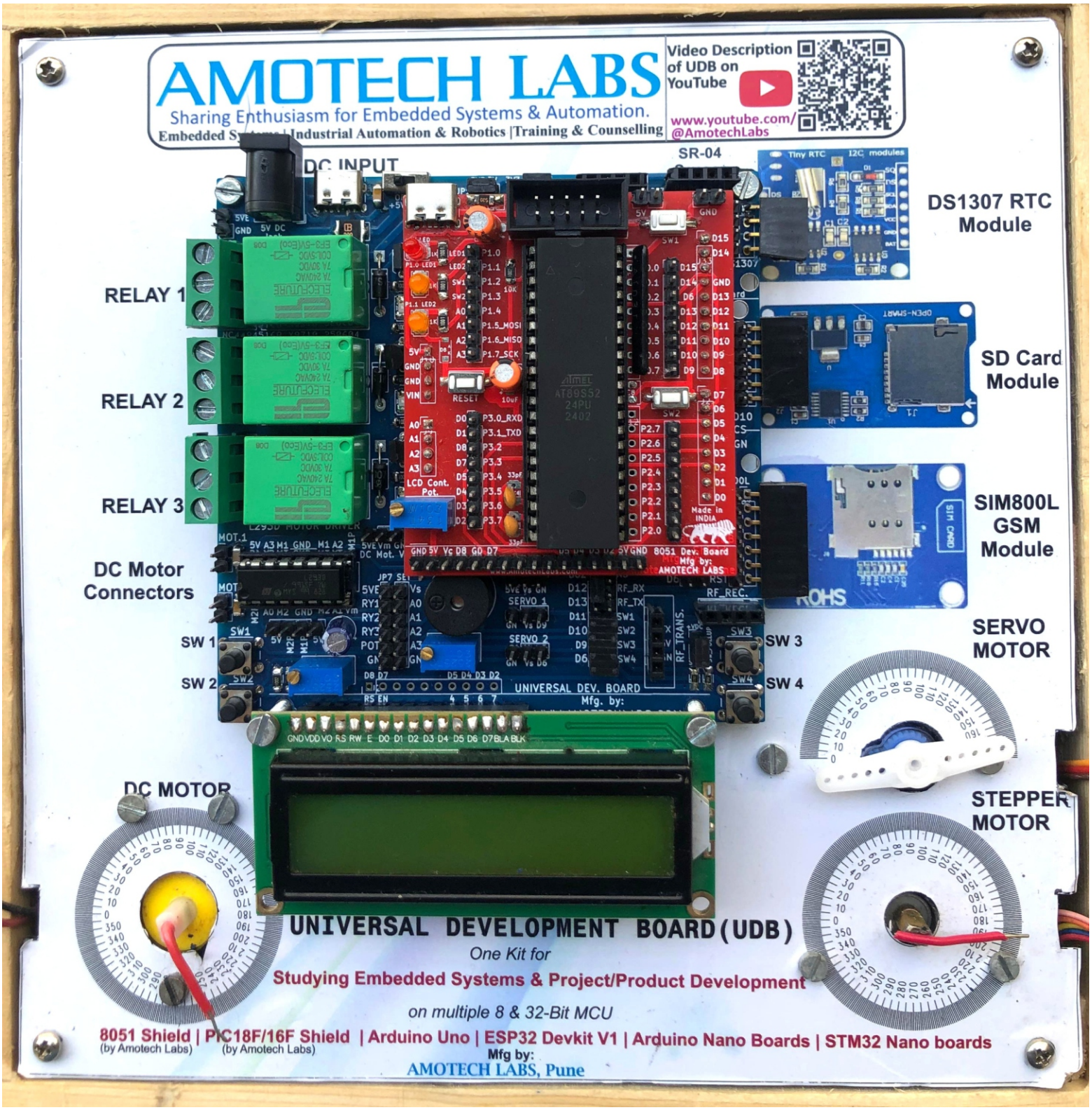
Arduino Nano Boards

STM32 Nucleo Nano Boards

Arduino Nano Rp2040

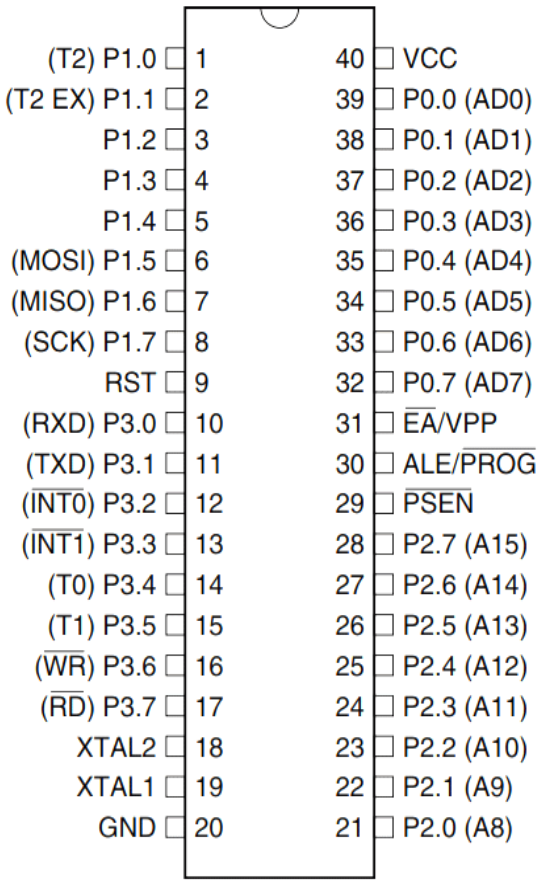
All Development Boards/Shields in Arduino Nano Footprint &

'8051 Mini Development Board' mounted on UDB Kit's Slot 1 can be seen in the below UDB Kit which has DC, Servo and Stepper Motors mounted on-board.



Note: This Kit was assembled for College Lab use in Wooden Box Enclosure. That's why it is fitted on a Acrylic Sheet with Motors mounted on it for studying their interfacing using 8051 MCU & UDB. For Students, only UDB is also available to buy with 'UDB 2 Wheel Robot Acrylic Sheet'. Students can configure the UDB by adding the peripherals as per their application needs.

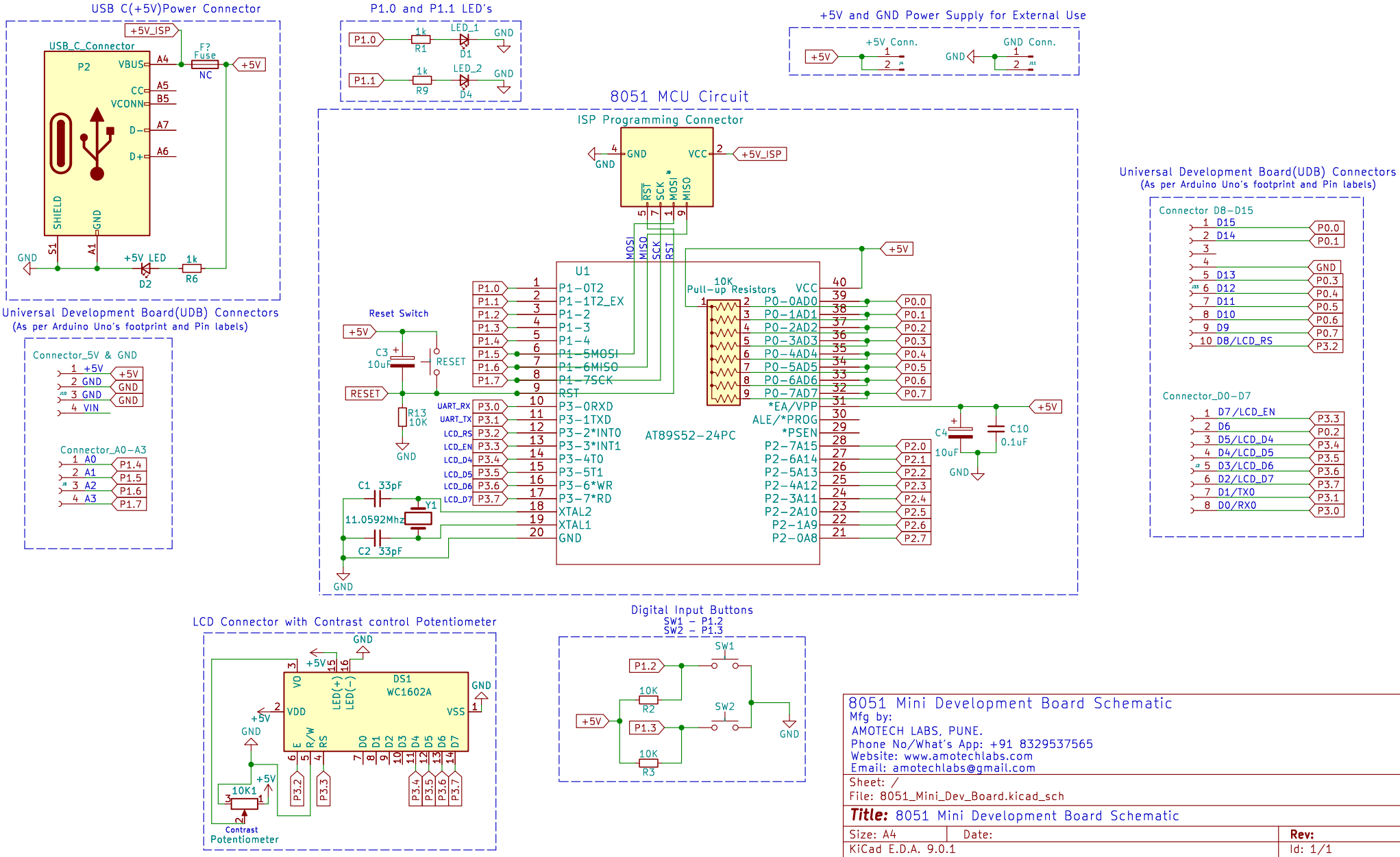
Introduction to AT89S52 Microcontroller



Features of AT89S52:

- ✓ Operating Frequency : DC - 33 Mhz
- ✓ Program Memory (Bytes): 8K
- ✓ Program Memory (Instructions) : 8192
- ✓ Data Memory (Bytes): 256
- ✓ Data EEPROM Memory (Bytes) : Not Availabe
- ✓ Interrupt Sources: 6
- ✓ I/O Ports : Ports 0,1,2,3
- ✓ Timers : 3(16 bit each)
- ✓ Capture/Compare/PWM Modules : No
- ✓ Enhanced Capture/ Compare/PWM Modules : No
- ✓ Serial Communications : MSSP, Addressable UART
- ✓ Universal Serial Bus (USB) Module : No
- ✓ Streaming Parallel Port (SPP) : No
- ✓ 10-Bit Analog-to-Digital Module : No
- ✓ Comparators : No
- ✓ Programmable Low-Voltage Detect : No
- ✓ Instruction Set :255
- ✓ Packages : 40-pin DIP, 44-pin PLCC , 44-pin TQFP

8051 Mini Development Board Schematic

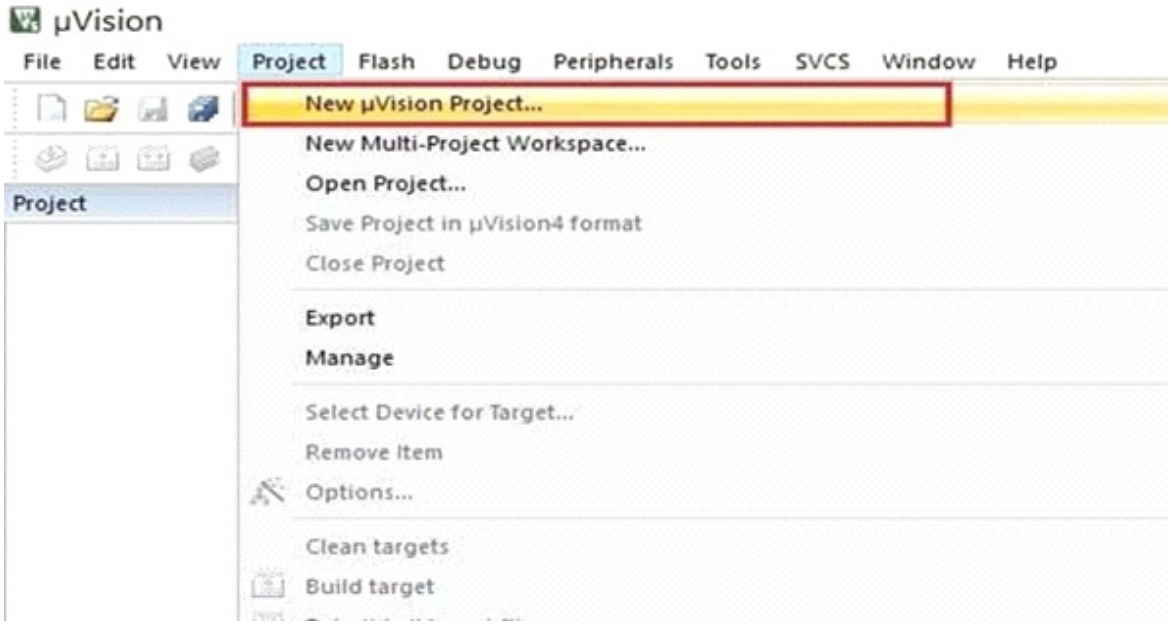


8051 Mini Development Board Schematic	
Mfg by:	
AMOTECH LABS, PUNE.	
Phone No/What's App: +91 8329537565	
Website: www.amotechlabs.com	
Email: amotechlabs@gmail.com	
Sheet: /	
File: 8051_Mini_Dev_Board.kicad_sch	
Title: 8051 Mini Development Board Schematic	
Size: A4	Date:
KiCad E.D.A. 9.0.1	Id: 1/1

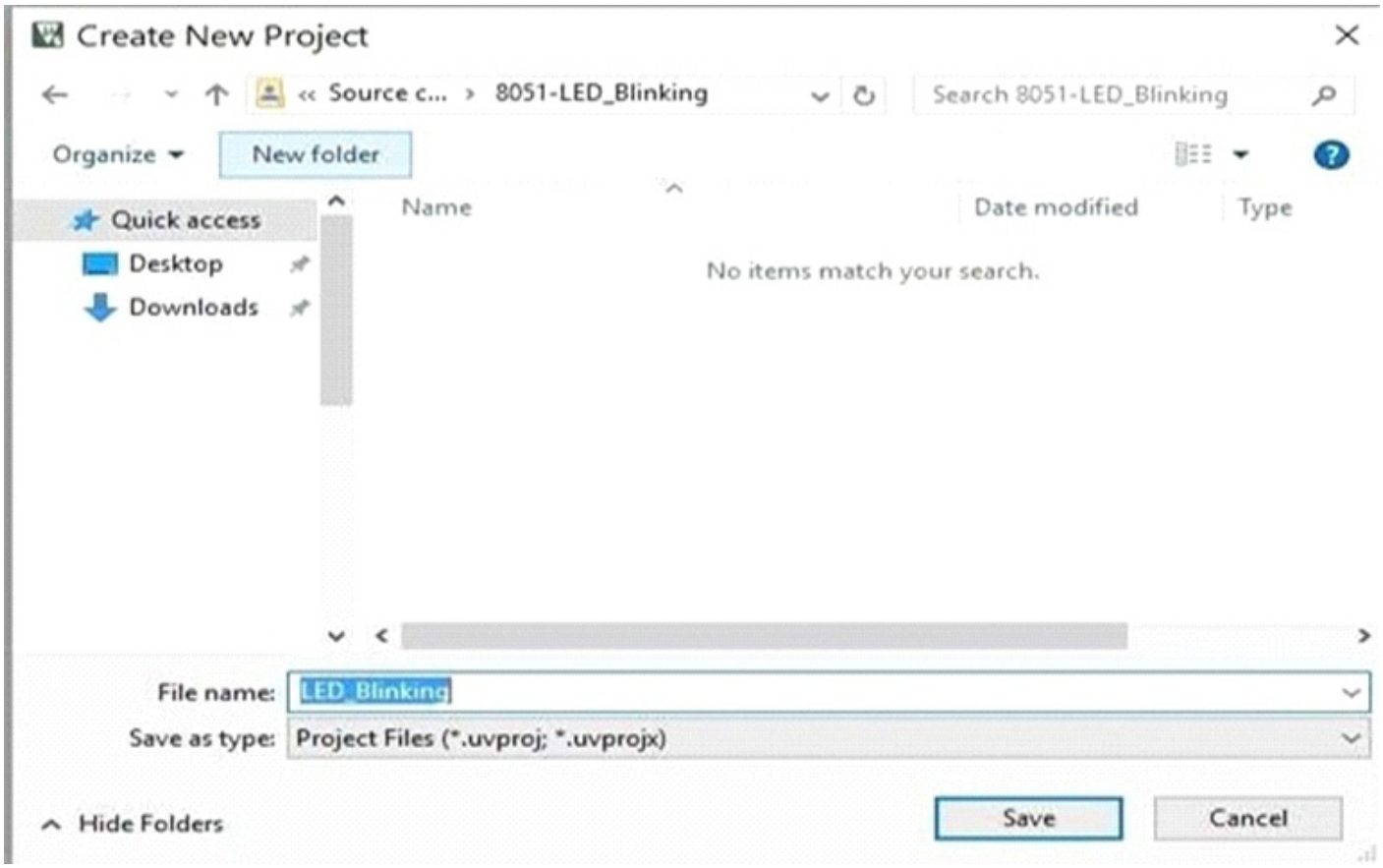
Keil IDE for Programming of 8051 (AT98S52) MC

Below are the steps to program any AT89S52 code.

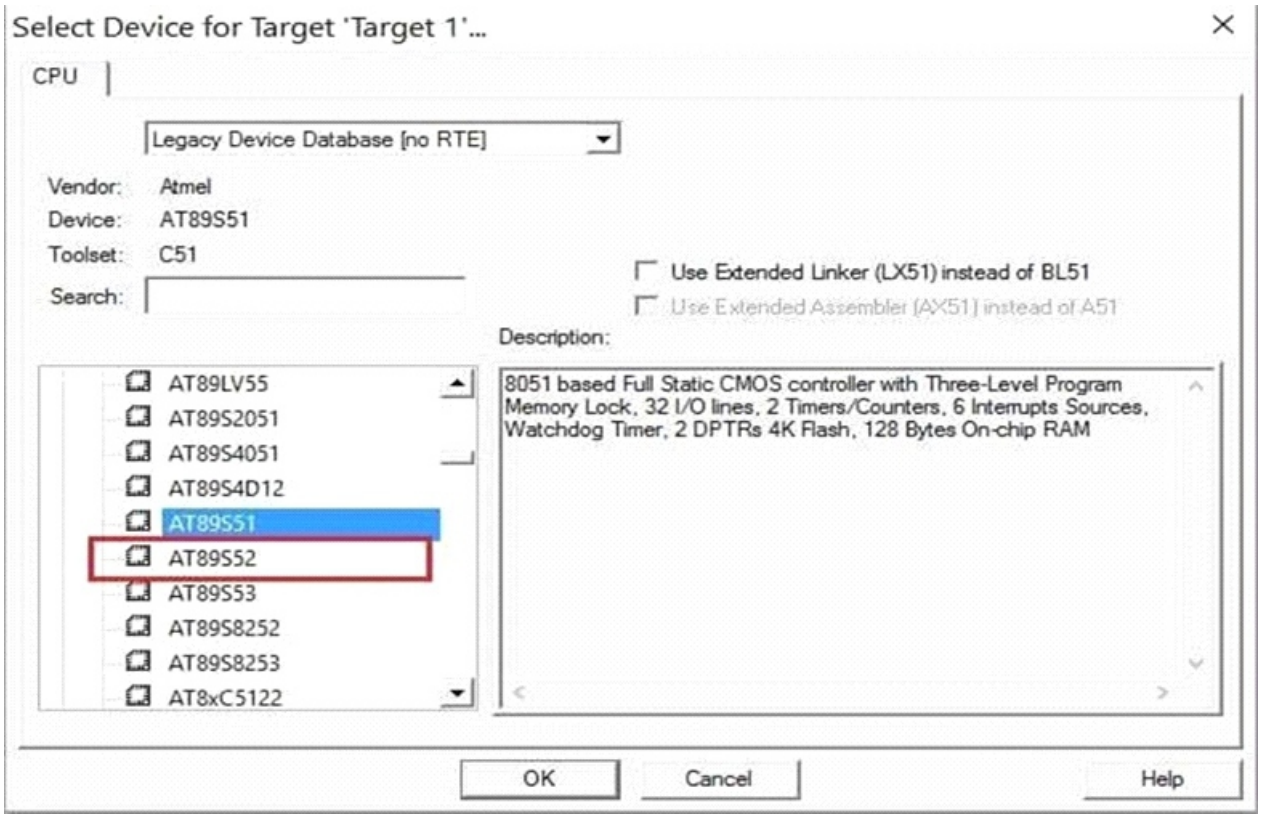
1.Select New µVision Project from the Project Menu.



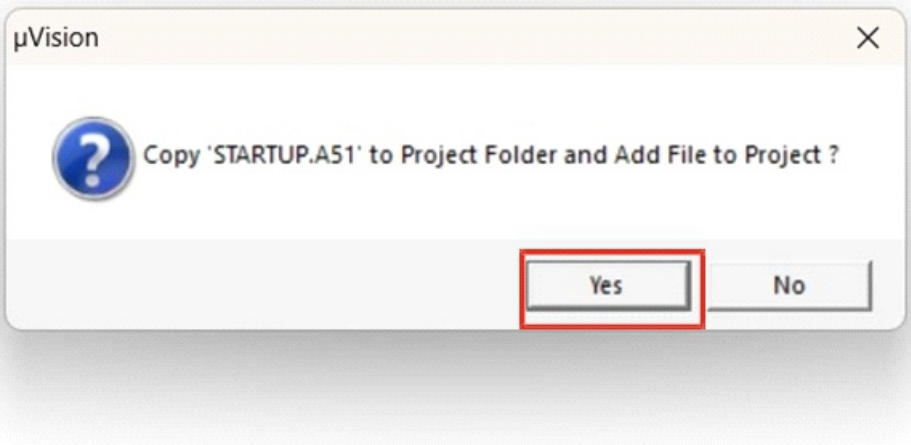
2. 'Create New Project' window will pop up, type a project name and location for the project and save.



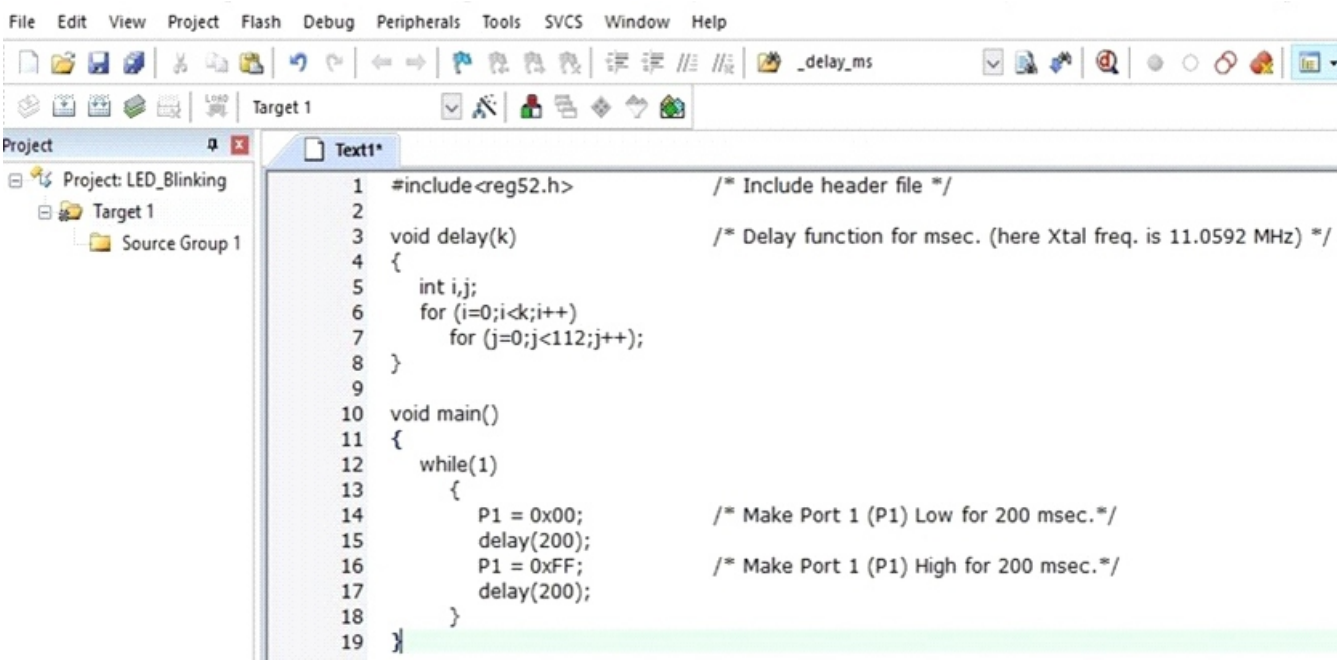
3.“Select Device for Target” window will pop up, select your device (here we selecting AT89S52).



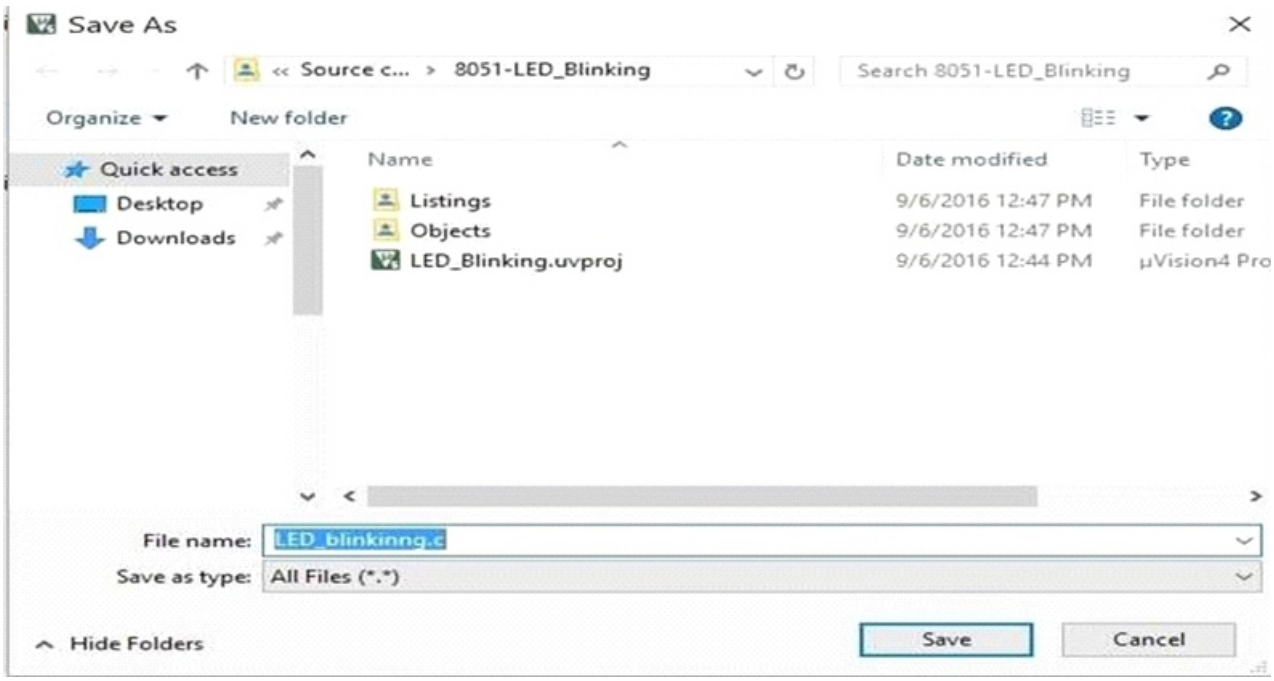
4. . “µVision” window will ask for copy STARTUP.A51 to the project folder and add a file to the project (If necessary you can select “No” but we recommended you to select “Yes” because it will help to run the program smoothly.)



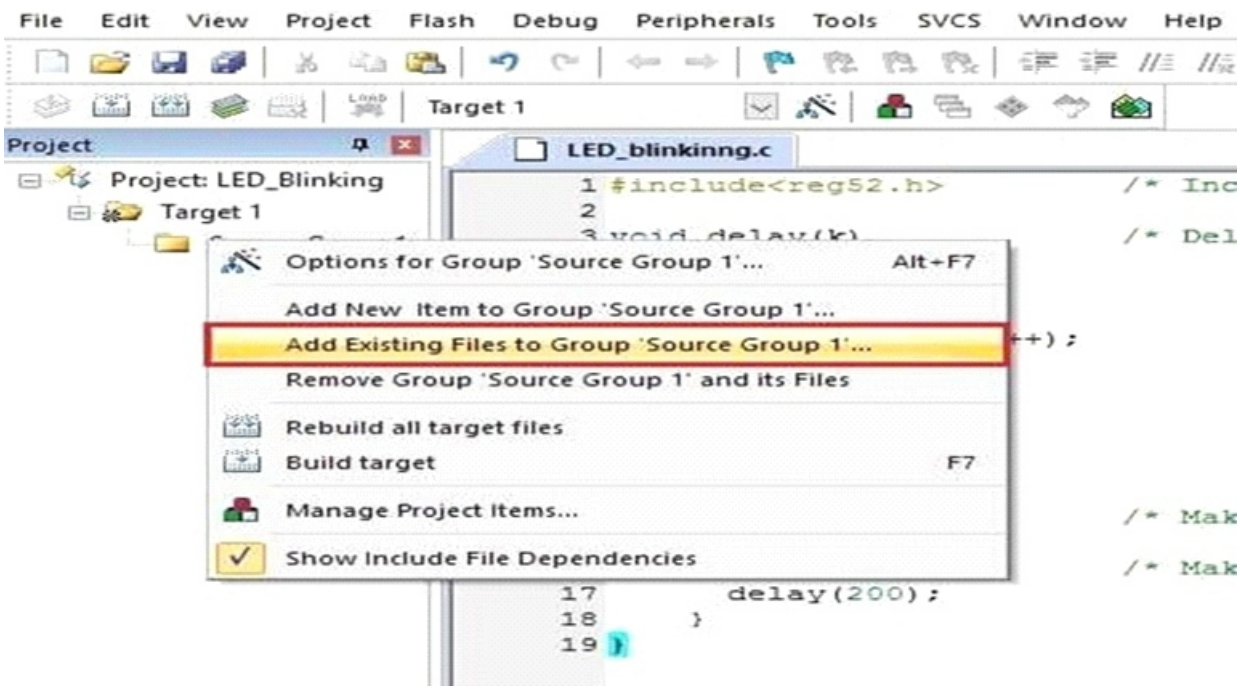
5. . Now select New file from the File menu and type your program co de (here we have typed LED blinking program)



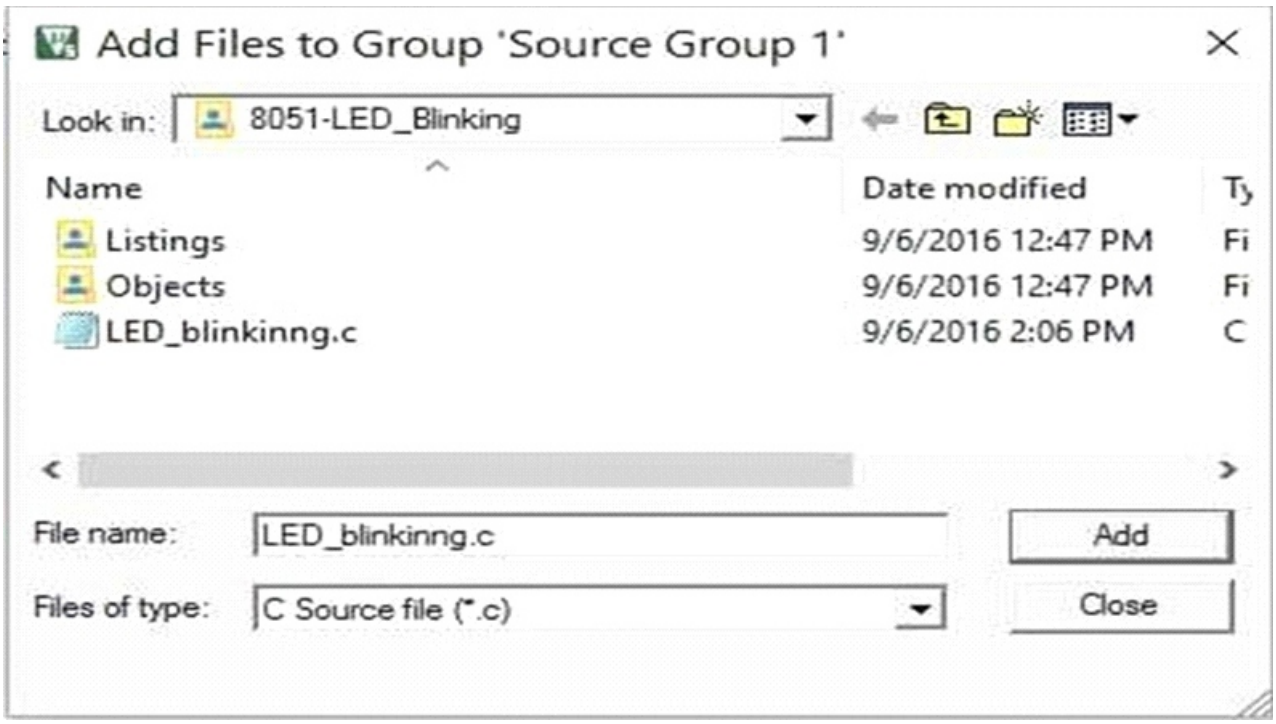
6.Save program code with ".c" extension (In case if you are using assembly language then save



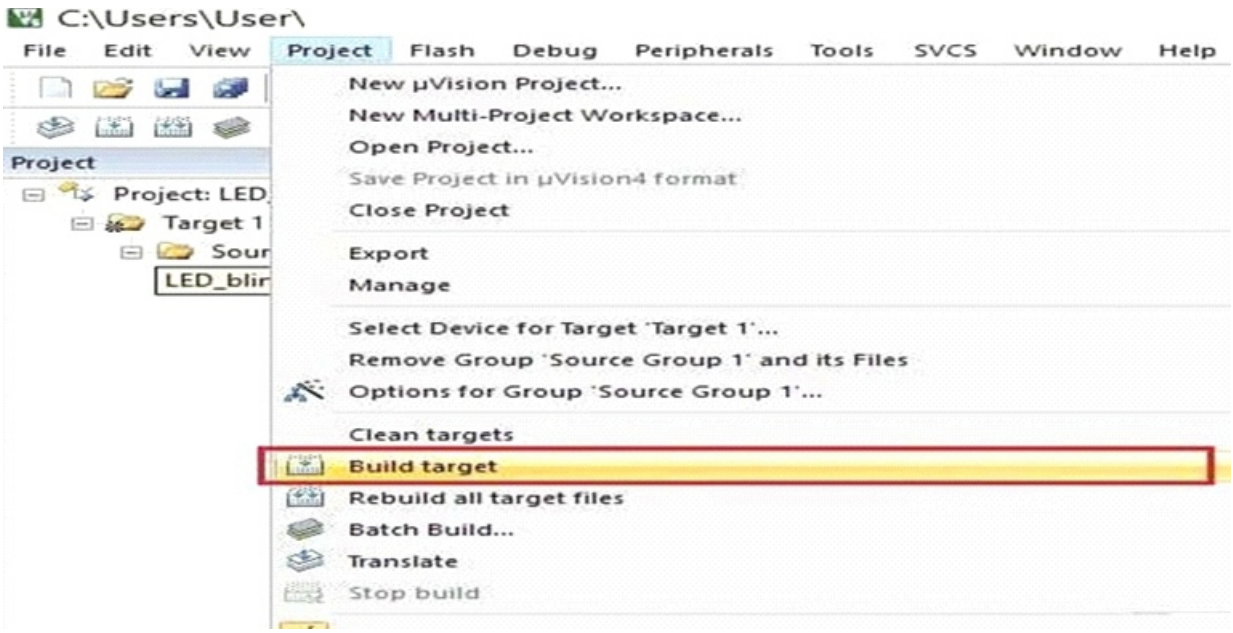
7. Right-click on Source Group 1 folder from Target 1 and select “Add existing files to Group 'Source Group 1'”



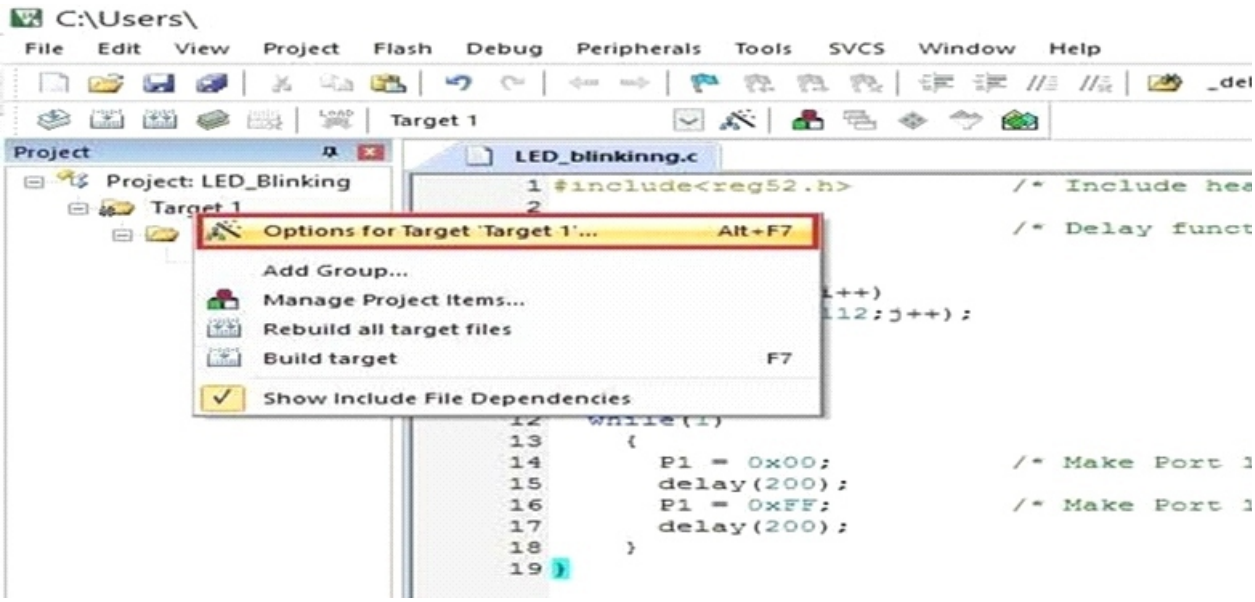
8. Select the program file saved with “.c “or “.asm” (in case of assembly language) and add it. Then close that window. You can see the added file in the “Source Group 1” folder in the left project window.



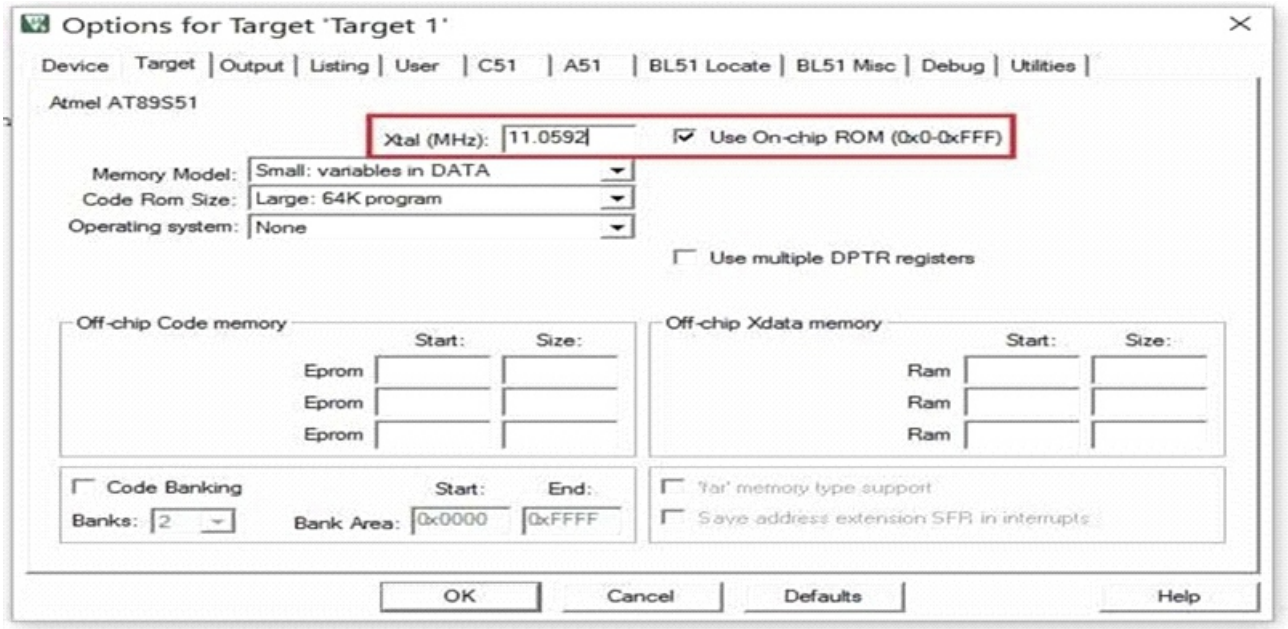
9.Now select the Project menu and click on “Build target”, it will build a project and give status in the Buildoutput window with Error and Warning count if any



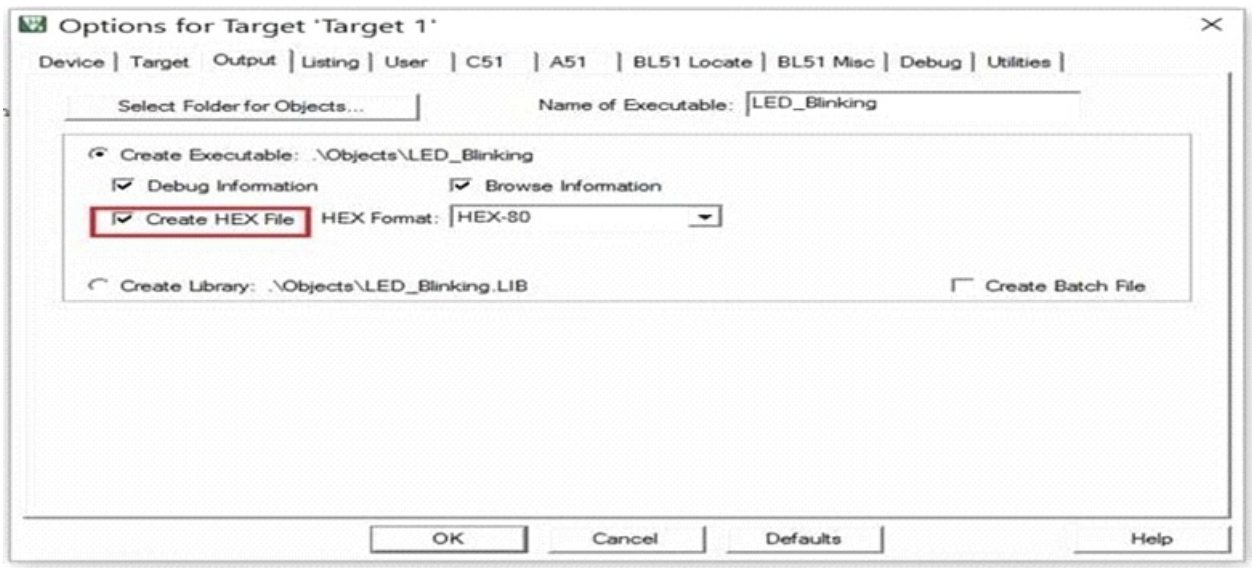
10.To create a Hex file right click on Target 1 and select Option for Target 'Tar



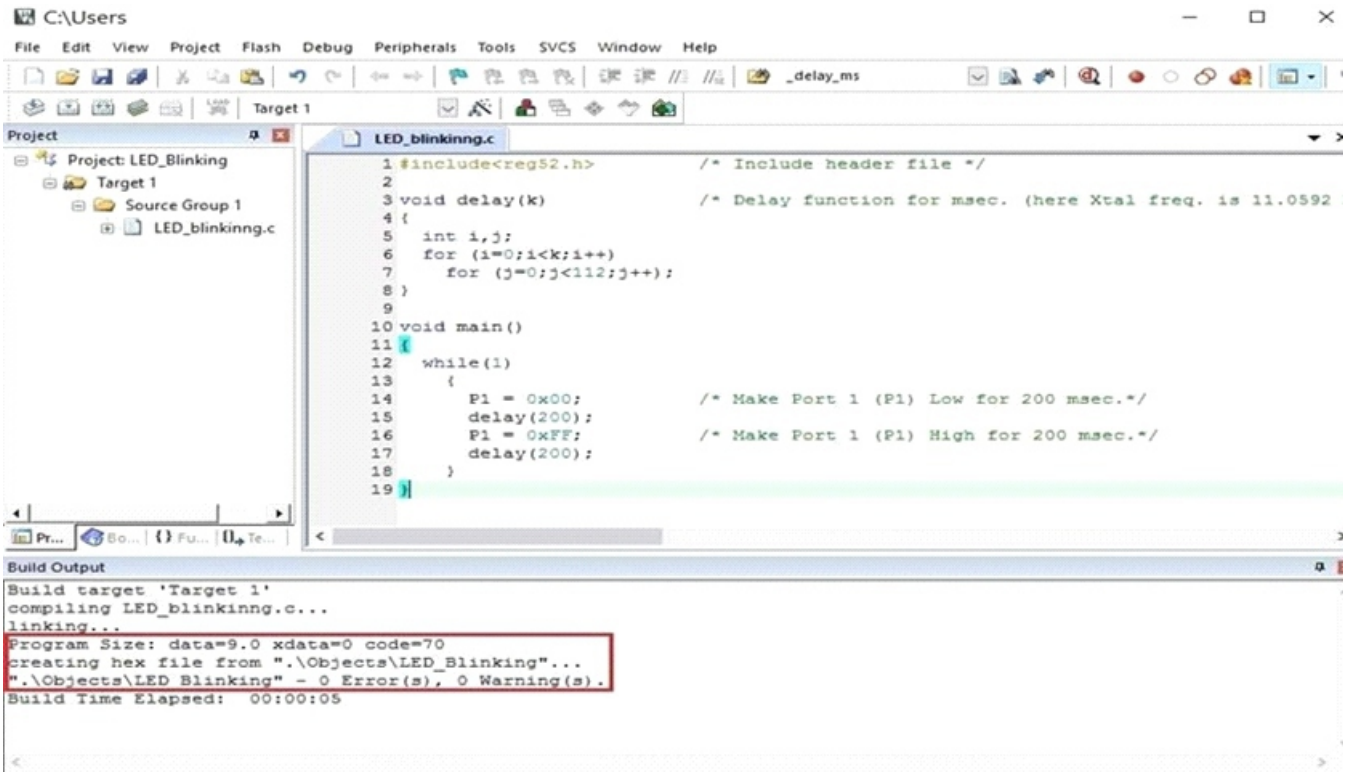
11. The target window will pop up, enter Xtal (MHz) frequency (here we used 11.0592 MHz), and tick mark in front of the “Use On-chip ROM” tag.



12. Within the same window select the “Output” option and tick mark in front of the “Create Hex File” tag and click on OK.



13. Again select build target from the Project menu or simply hit the F7 shortcut key for the same, it will build target and also create a Hex file. You can see creating a Hex file in the Build output window

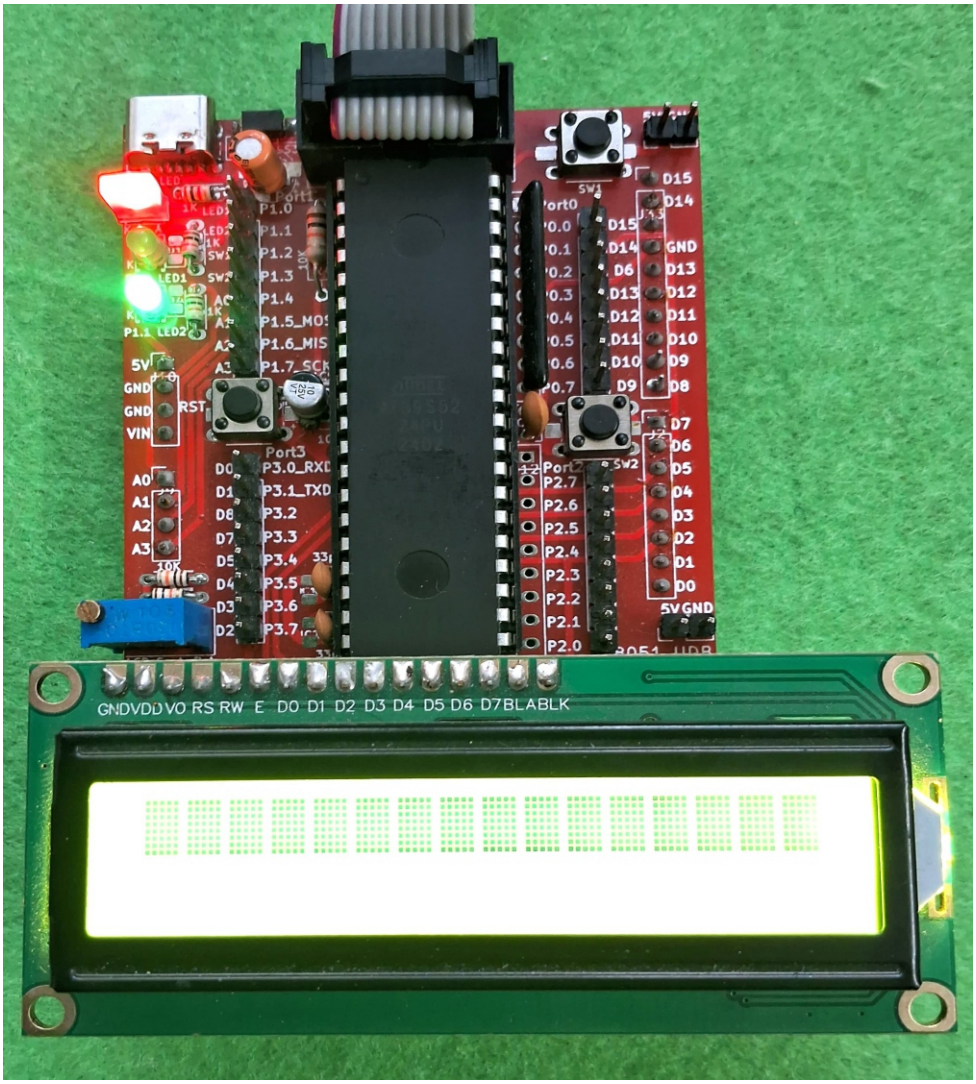


14. Check Video Tutorials on our YouTube Channel, for Instructions of Installing ISP Programmer Drivers, Compiling the program and Downloading the Hex File into our 8051 Development Boards. In Videos for '8051 Development Boards', under their description you will also find a Google Drive Link to download ISP Programmer, Drivers, Programmer Software Files, Sample Programs and Schematic.

YouTube Channel: https://www.youtube.com/@AmotechLabs

Lab 1 - LED Blinking

Aim: To study the LED's Interfacing with AT89S52 mini development board.



Procedure:

1. Read the LED Blink Program on the next page with all the comments explaining the code.
2. Open the Project folder in Keil µVision IDE or create a new project following the Keil manual.
3. Build the project and generate the HEX file.Use Keil Programmer or another suitable tool to download the HEX file to the 8051 microcontroller.
4. Open the Progsip and load the hex file of the program and then run the program.

Program:

```
1. #include<reg52.h>// special function register declarations
2.  sbit LED1 = P1^0; //LED1 pin
3.  sbit LED2 = P1^1; //LED2 pin
4.  void delay(unsigned int count);
5.  void main (void)
6.  {
7.      while(1) // infinite loop
8.      {
1.              LED1 = 0;
2.              LED2 = 1;
3.              delay(1000);
4.              LED1 = 1;
5.              LED2 = 0;
6.              delay(1000);
7.    }
9.  }
10. //delay function generates delay in millisecond.
11. void delay(unsigned int count)
12. {
13.    int i,j;
14.    for(i=0;i<count;i++)
15.    for(j=0;j<112;j++);
16. }
```

Lab 2 - LCD Interfacing

Aim: To study the 16x2 LCD Interfacing with AT89S52 mini development board.

Procedure:

1. Read the LCD Interfacing Program with all comments explaining the code.
2. Open Keil µVision, create/open a project for 8051 (AT89C51), write the program, and build it to generate the Hex file.
3. Use Progsip to download the Hex file to the 8051 microcontroller.
4. Adjust the LCD contrast using the 10k pot. Once programmed and disconnected, the expected content will appear on the LCD.

Program:

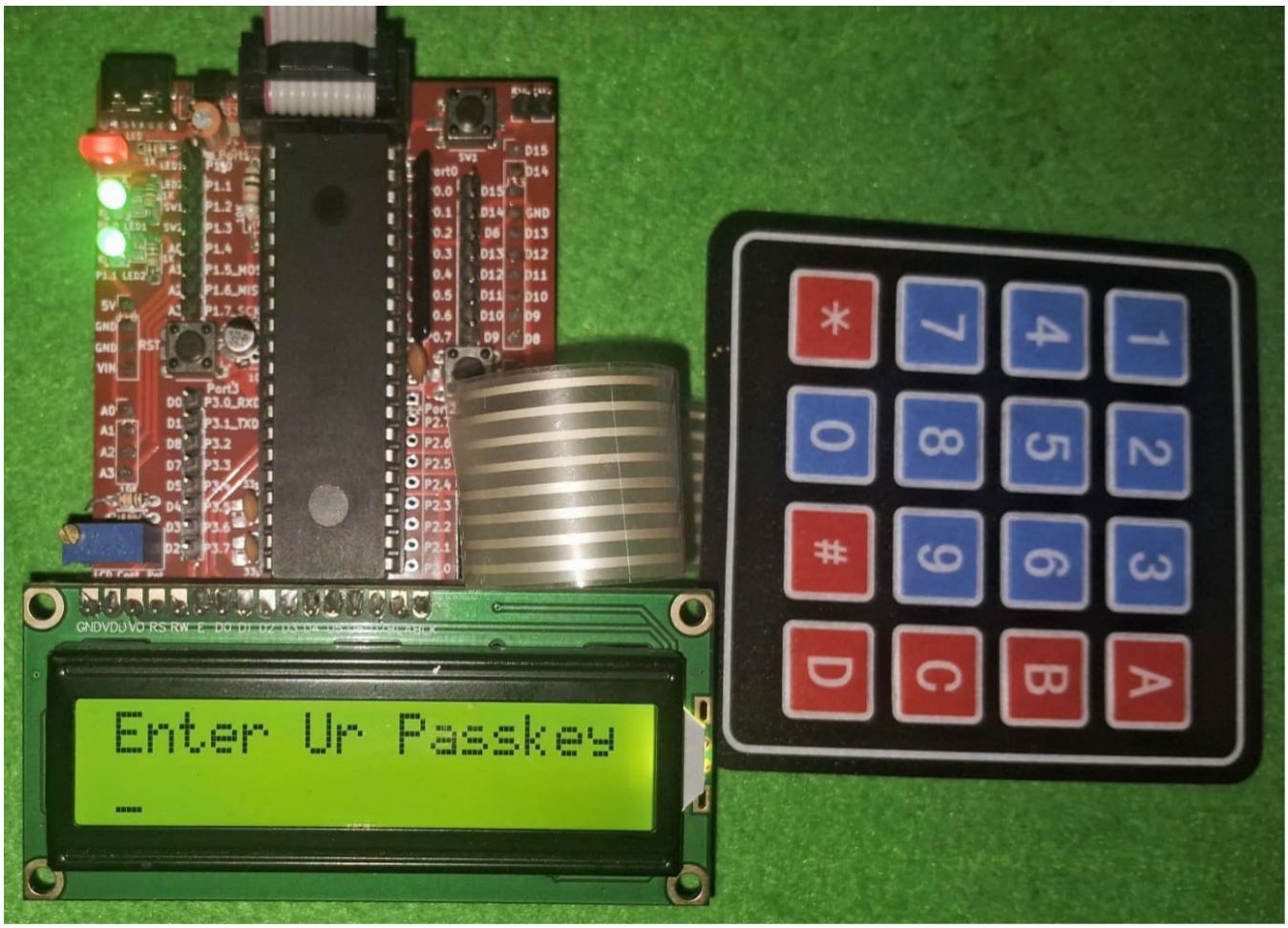
```
#include<reg51.h>
sbit D7=P3^7; //LCD D7
sbit D6=P3^6; //LCD D6
sbit D5=P3^5;//LCD D5
sbit D4=P3^4; //LCD D4
sbit rs=P3^2;  /* RS Register select pin */ //Ard D8
sbit en=P3^3;  /* EN Enable pin */ //Ard D7
int LCD_Port;

 /* Function to provide delay Approx 1ms with 11.0592 Mhz crystal*/
 void delay(unsigned int count)
 {
 int i,j;
 for(i=0;i<count;i++)
 for(j=0;j<112;j++);
 }

 void LCD_Command (char cmnd) /* LCD16x2 command funtion */
 {
 LCD_Port =(cmnd&0xF0)>>4; /* Send upper nibble */
 D7=LCD_Port&0X08;
 D6=LCD_Port&0X04;
 D5=LCD_Port&0X02;
 D4=LCD_Port&0X01;
 rs=0;     /* Command reg. */
 // rw=0;  /* Write operation */
 en=1;
 delay(1);
 en=0;
 delay(2);

 LCD_Port =(cmnd&0x0F); /* Send lower nibble */
 D7=LCD_Port&0X08;
 D6=LCD_Port&0X04;
 D5=LCD_Port&0X02;
 D4=LCD_Port&0X01;
 rs=0;
 en=1;/* Enable pulse */
 delay(1);
 en=0;
 delay(5);
 }

 void LCD_Char (charchar_data) /* LCD data write function */
 {
 LCD_Port =(char_data&0xF0)>>4; /* Send upper nibble */
```

```
47  D7=LCD_Port&0X08;
48  D6=LCD_Port&0X04;
49  D5=LCD_Port&0X02;
50  D4=LCD_Port&0X01;
51  rs=1; /*Data reg.*/
52  // rw=0; /*Write operation*/
53  en=1;
54  delay(1);
55  en=0;
56  delay(2);
57
58  LCD_Port =(char_data&0x0F); /* Send lower nibble */
59  D7=LCD_Port&0X08;
60  D6=LCD_Port&0X04;
61  D5=LCD_Port&0X02;
62  D4=LCD_Port&0X01;
63  en=1; /* Enable pulse */
64  delay(1);
65  en=0;
66  delay(5);
67  }
68
69  void LCD_String (char*str) /* Send string to LCD function */
70  {
71     int I;
72     for(i=0;str[i]!=0;i++)  /* Send each char of string till the NULL */
73    {
74              LCD_Char(str[i]); /* Call LCD data write */
75     }
76  }
77
78  void LCD_String_xy (char row, char pos, char *str) /* Send string to LCD function */
79  {
80     if (row == 0)
81     LCD_Command((pos & 0x0F)|0x80);
82     else if (row == 1)
83     LCD_Command((pos & 0x0F)|0xC0);
84     LCD_String(str);  /* Call LCD string function */
85  }
86  void LCD_Init (void) /* LCD Initialize function */
87  {
88    delay(20);   /* LCD Power ON Initialization time >15ms */
89    LCD_Command (0x02);  /* 4bit mode */
90    LCD_Command (0x28);  /* Initialization of 16X2 LCD in 4bit mode */
91    LCD_Command (0x0C); /* Display ON Cursor OFF */
92    LCD_Command (0x06); /* Auto Increment cursor */
93    LCD_Command (0x01); /* clear display */
94    LCD_Command (0x80); /* cursor at home position */
95     }
```

```
 98  void main()
 99  {
100   LCD_Init(); /* Initialization of LCD*/
101    LCD_String("8051 Mini Kit");/* write string on 1st line of LCD*/
102    LCD_Command(0xc0); /* Go to 2nd line*/
103    LCD_String("by AMOTECH LABS");/*write string on 2nd line*/
104    delay(1000);
105    while(1); /* Infinite loop. */
106  }
```

Lab 4 - 4x4 Keypad Interfacing

Aim: To study the Interfacing of 4x4 Keypad with AT89S52 mini development board.



Procedure:

1. Read the Keypad Interfacing Program on the next page with all the comments explaining the program.
2. Open the Project folder for the program in Keil µVision IDE or create a new project following the procedure explained in the Keil manual. Build the project and generate the HEX file.
3. Use Keil Programmer or another suitable tool to download the HEX file to the 8051 microcontroller.
4. If you entered the correct passkey then you will get acess otherwise you will get the message “Access denied”.

Program:

```
#include<reg51.h>
#include<string.h>
#define lcdport P3

sbit col1=P2^3;
sbit col2=P2^2;
sbit col3=P2^1;
sbit col4=P2^0;
sbit row1=P2^7;
sbit row2=P2^6;
sbit row3=P2^5;
sbit row4=P2^4;
sbitrs =P3^2;
sbit en=P3^3;
sbit buzzer=P1^0;
char pass[4],i=0;

void delay(int itime)
{
   int i,j;
   for(i=0;i<itime;i++)
   for(j=0;j<1275;j++);
}
void daten()
{
   rs=1;
   en=1;
   delay(5);
   en=0;
}
void lcddata(unsigned char ch)
{
   lcdport=ch & 0xf0;
   daten();
   lcdport=(ch<<4) & 0xf0;
   daten();
}
void cmden(void)
{
   rs=0;
   en=1;
   delay(5);
   en=0;
}
.. void lcdcmd(unsigned char ch)
{
   lcdport=ch &
   cmden();
   lcdport=(ch<<4) & 0xf0;
  cmden();
}
void lcdstring(char *str)
{
   while(*str)
   {
  lcddata(*str);
  str++;
   }
}
void lcd_init(void)
{
   lcdcmd(0x02);
  lcdcmd(0x28);
  lcdcmd(0x0e);
   lcdcmd(0x01);
}
 void keypad()
{
   int cursor=192,flag=0;
   lcdcmd(1);
   lcdstring("Enter Ur Passkey");
   lcdcmd(0xc0);
  i=0;
 while(i<4)
   {
   flag=cursor;
    col1=0;
    col2=col3=col4=1;
    if(!row1)
    {
      lcddata('1');
       pass[i++]='1';
       cursor++;
       while(!row1);
     }
     else if(!row2)
    {
       lcddata('4');
      pass[i++]='4';
       cursor++;
       while(!row2);
       }
     else if(!row3)
     {
       lcddata('7');
```

Program:

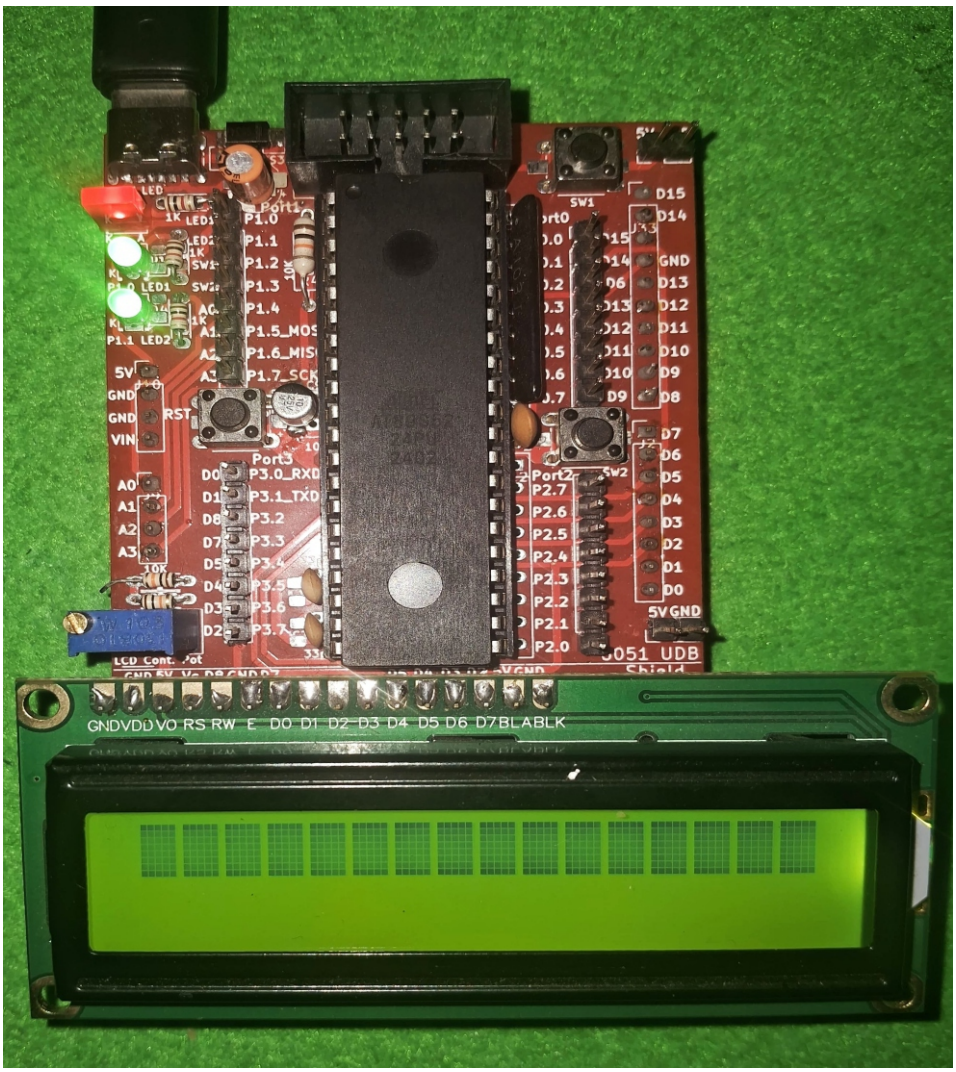
```
97.  pass[i++]='7';
98.        cursor++;
99.        while(!row3);
100.   }
110 else if(!row4)
111.   {
112.       lcddata('*');
113.        pass[i++]='*';
114.        cursor++;
115.       while(!row4);
116.        }
117.     col2=0;
118.     col1=col3=col4=1;
119.     if(!row1)
120.     {
121.        lcddata('2');
122.        pass[i++]='2';
123.        cursor++;
124        while(!row1);
125.        }
126.    else if(!row2)
127.     {
128.        lcddata('5');
129.        pass[i++]='5';
130.        cursor++;
131.        while(!row2);
132.        }
133.      else if(!row3)
134.     {
135.        lcddata('8');
136.        pass[i++]='8';
137.        cursor++;
138.        while(!row3);
139.        }
140.      else if(!row4)
141.     {
142.        lcddata('0');
143        pass[i++]='0';
144.        cursor++;
145.        while(!row4);
146.        }
147.     col3=0;
148.     col1=col2=col4=1;
149. if(!row1)
150.     {
151.        lcddata('3');
151. pass[i++]='3';
152.        cursor++;
153        while(!row1);
154.        }
155.      else if(!row2)
156.     {
157.        lcddata('6');
158.      pass[i++]='6';
159.        cursor++;
160.        while(!row2);
161.        }
162.     else if(!row3)
163.     {
167.        lcddata('9');
168.        pass[i++]='9';
169.       cursor++;
170.      while(!row3);
171.        }
172     else if(!row4)
173.    {
174.       lcddata('#');
175.        pass[i++]='#';
176.        cursor++;
177.        while(!row4);
178.        }
179.    col4=0;
180.     col1=col3=col2=1;
181.     if(!row1)
182.    {
183.        lcddata('A');
184        pass[i++]='A';
185.        cursor++;
186.        while(!row1);
187.        }
188.      else if(!row2)
189.    {
190.        lcddata('B');
191.        pass[i++]='B';
192.        cursor++;
193.        while(!row2);
194.        }
195. else if(!row3)
196.     {
197.        lcddata('C');
198.        pass[i++]='C';
199.        cursor++;
200.        while(!row3);
201        }
```

Program:

```
202. else if(!row4)
203.     {
204.        lcddata('D');
205.        pass[i++]='D';
206.        cursor++;
207.        while(!row4);
208.          }
209.     if(i>0)
210.     {
211.         if(flag!=cursor)
212.         delay(100);
213.         lcdcmd(cursor-1);
214.         lcddata('*');
215.     }
216. }
217.}
218. void accept()
219. {
220.    lcdcmd(1);
221.     lcdstring("Welcome");
222.     lcdcmd(192);
223.   lcdstring("Password Accept");
224.     delay(200);
225. }
225. void wrong()
226. {  buzzer=0;
227.            lcdcmd(1);
228.     lcdstring("Wrong Passkey");
229.    lcdcmd(192);
230.    lcdstring("PLZ Try Again");
231.    delay(200);
232.    buzzer=1;
233. }
234. void main()
235. {
236.    buzzer=1;
237.    lcd_init();
238.    lcdstring("AMOTECH LABS");
239.    lcdcmd(0xc0);
240.   lcdstring("   PUNE     “);
241.   delay(1000);
242.    lcdcmd(1);
243.    lcdstring("Electronic Code");
244.    lcdcmd(0xc0);
245.    lcdstring("  Lock System  ");
246.    delay(400);
247. while(1)
248.   {
249.          i=0;
250.          keypad();
251.          if(strncmp(pass,"4201",4)==0)
252.          {
253.          accept();
254.          lcdcmd(1);
255.          lcdstring("Access Granted ");
256.          delay(300);
257.             }
258.       else
259.       {
260.          lcdcmd(1);
261.          lcdstring("Access Denied");
262.          wrong();
263.          delay(300);
264.       }
265.
266.    }
267. }
```

Lab 3 - Digital Input Button

Aim: To study the Interfacing of Digital Input Button with AT89S52 mini development board.



Procedure:

1. Read the Digital Input Button Program with comments explaining the code.
2. Open the Project folder in Keil µVision IDE or create a new project following the Keil manual.
3. Build the project and generate the HEX file.Use Keil Programmer or another tool to download the HEX file to the 8051 microcontroller
4. Open the Progsip and load the hex file of the program and then run the program.

Program:

```
1.  #include <reg52.h> // Include register definitions
2.  sbit LED1 = P1^0; // LED1 connected to P1.0
3.  sbit LED2 = P1^1; // LED2 connected to P1.1
4.  sbit SW1 = P1^2; // Switch 1 connected to P1.2
5.  sbit SW2 = P1^3; // Switch 2 connected to P1.3
6.  void delay(unsigned int count);
7.  void main(void)
8.  {
9.       // Configure SW1 and SW2 pins as Input pins by writing 1 to them.
10.     //  After configuring them as Inputs pins, they will be pulled-up internally.
11.     SW1 = 1;
12.     SW2 = 1;
13.     // Initialize LEDs to OFF state
14.     LED1 = 0;
15.     LED2 = 0;
16.     while(1)
17.     {
18.             LED1 = Sw1;// After pressing SW1, Pin P1.2 wiil be Low making LED1 Low.
19.             delay(50);
20.             LED2 = Sw2;// After pressing SW2, Pin P1.3 wiil be Low making LED2 Low.
21.              delay(50);
22.     }
23. }
24. // Simple delay function (approximately 1ms per count)
25. void delay(unsigned int count) {
26. {
27.     unsigned int I, j;
28.     for(I = 0; i < count; i++)
29.     for(j = 0; j < 112; j++);
30. }
```