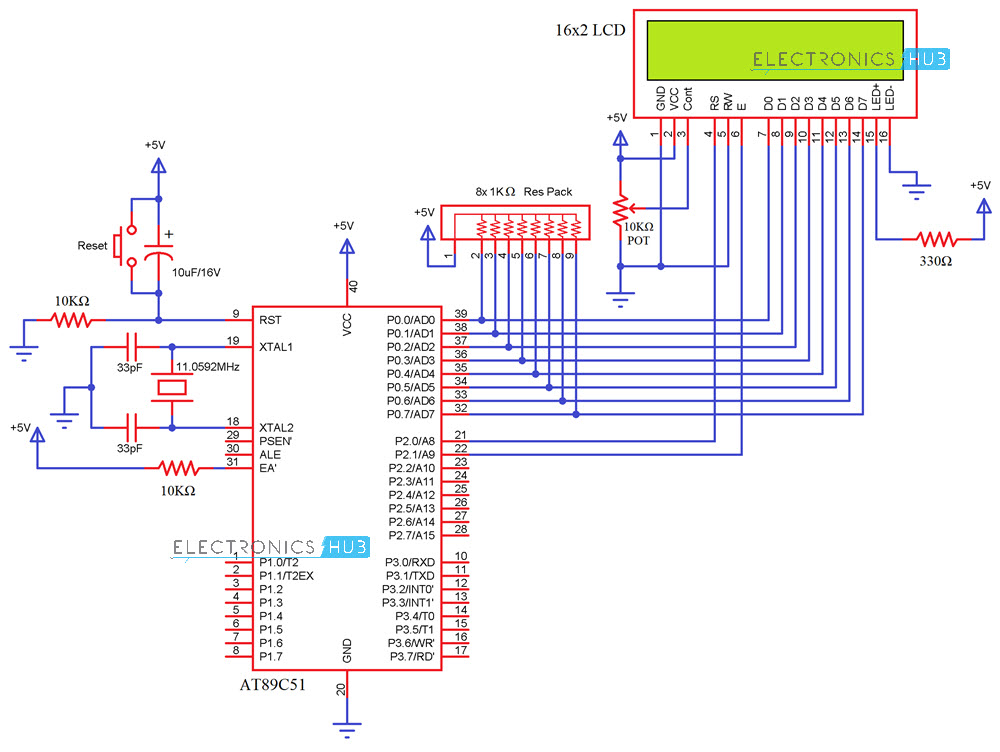
**Interfacing 16×2 LCD with 8051**



In this project, we will have brief discussion on how to interface 16×2 LCD module to AT89C51, which is an 8051 family microcontroller. We use LCD display for the displaying messages in a more interactive way to operate the system or displaying error messages etc. Interfacing 16×2 LCD with 8051 microcontroller is very easy if you understanding the working of LCD.

Hence, in this project, I will not only give the information of LCD and also provide the code in C language which is working fine without any errors.

Table of Contents

* [A Brief Note on 16×2 LCD](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#A_Brief_Note_on_162_LCD)
* [Interfacing 16×2 LCD with 8051 Circuit Diagram](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Interfacing_162_LCD_with_8051_Circuit_Diagram)
* [Components Required](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Components_Required)
* [Circuit Explanation](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Circuit_Explanation)
* [Programming LCD to 8051](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Programming_LCD_to_8051)
  + [Initializing LCD](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Initializing_LCD)
  + [Sending Commands to the LCD](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Sending_Commands_to_the_LCD)
  + [Writing the Data to the LCD](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Writing_the_Data_to_the_LCD)
* [Code](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Code)
* [Additional Codes](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Additional_Codes)
  + [Code 1](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Code_1)
  + [Code 2](https://www.electronicshub.org/interfacing-16x2-lcd-8051/#Code_2)

**A Brief Note on 16×2 LCD**

16×2 Liquid Crystal Display which will display the 32 characters at a time in two rows (16 characters in one row). Each character in the display is of size 5×7 pixel matrix. This matrix differs for different 16×2 LCD modules, if you take JHD162A, this matrix goes to 5×8. There are 16 pins in the LCD module, the pin configuration us given below

| **PIN NO** | **NAME** | **FUNCTION** |
| --- | --- | --- |
| 1 | VSS | Ground pin |
| 2 | VCC | Power supply pin of 5V |
| 3 | VEE | Used for adjusting the contrast commonly attached to the potentiometer. |
| 4 | RS | RS is the register select pin used to write display data to the LCD (characters), this pin has to be high when writing the data to the LCD. During the initializing sequence and other commands this pin should low. |
| 5 | R/W | Reading and writing data to the LCD for reading the data R/W pin should be high (R/W=1) to write the data to LCD R/W pin should be low (R/W=0) |
| 6 | E | Enable pin is for starting or enabling the module. A high to low pulse of about 450ns pulse is given to this pin. |
| 7 | DB0 |  |
| 8 | DB1 |  |
| 9 | DB2 |  |
| 10 | DB3 |  |
| 11 | DB4 | DB0-DB7 Data pins for giving data(normal data like numbers characters or command data) which is meant to be displayed |
| 12 | DB5 |  |
| 13 | DB6 |  |
| 14 | DB7 |  |
| 15 | LED+ | Back light of the LCD which should be connected to Vcc |
| 16 | LED- | Back light of LCD which should be connected to ground. |

So by reading the above table you can get a brief idea how to display a character. For displaying a character you should enable the enable pin (pin 6) by giving a pulse of 450ns, after enabling the pin6 you should select the register select pin (pin4) in write mode. To select the register select pin in write mode you have to make this pin high (RS=1), after selecting the register select you have to configure the R/W to write mode that is R/W should be low (R/W=0).

Follow these simple steps for displaying a character or data

* **E=1;** enable pin should be high
* **RS=1;** Register select should be high
* **R/W=0;** Read/Write pin should be low.

To send a command to the LCD just follows these steps:

* **E=1;** enable pin should be high
* **RS=0;** Register select should be low
* **R/W=0;** Read/Write pin should be low.

**Commands:** There are some preset commands which will do a specific task in the LCD. These commands are very important for displaying data in LCD. The list of commands given below:

| **Command** | **Function** |
| --- | --- |
| 0F | For switching on LCD, blinking the cursor. |
| 1 | Clearing the screen |
| 2 | Return home. |
| 4 | Decrement cursor |
| 6 | Increment cursor |
| E | Display on and also cursor on |
| 80 | Force cursor to beginning of the first line |
| C0 | Force cursor to beginning of second line |
| 38 | Use two lines and 5x7 matrix |
| 83 | Cursor line 1 position 3 |
| 3C | Activate second line |
| 0C3 | Jump to second line position 3 |
| 0C1 | Jump to second line position1 |

To get the detailed information, [Click Here and Download the Datasheet](https://www.electronicshub.org/wp-content/uploads/2013/09/HD44780_16x2_Character_LCD_Display.pdf)

**Interfacing 16×2 LCD with 8051 Circuit Diagram**

**Components Required**

* AT89C51 (8051 Microcontroller)
* 16X2 LCD Display
* 11.0592MHz Crystal
* 2 X 33pF Capacitors
* 2 X 10 KΩ Resistors
* 1 KΩ X 8 Resistor Pack
* 10 KΩ Potentiometer
* 330Ω Resistor
* Push Button
* 10μF/16V Capacitor
* 8051 Programmer
* 5V Power Supply
* Connecting Wires

**Circuit Explanation**

The crystal oscillator, along with two 33pF Capacitors, are connected to XTAL1 and XTAL2, which will provide the system clock to the microcontroller.

RST Pin is pulled-LOW with the help of a 10KΩ Resistor. With the help of a 10μF Capacitor and a Push Button, you can reset the 8051 Microcontroller. EA is pulled-HIGH with the help of a 10KΩ resistor.

The data pins of the LCD are connected to PORT0 (first, the PORT0 pins must be pulled-HIGH with the help of a 1KΩ Resistor Pack). RS and E are connected to PORT2 pins P2.0 and P2.1.

A 10KΩ Potentiometer is used to adjust the contrast of the LCD.

**Programming LCD to 8051**

Coming to the programming you should follow these steps:

* **STEP1:** Initialization of LCD.
* **STEP2:** Sending commands to LCD.
* **STEP3:** Writing the data to LCD.

**[Also Read:**[**How To Make an Adjustable Timer**](https://www.electronicshub.org/adjustable-timer/)**]**

**Initializing LCD**

To initialize LCD to the 8051 the following instruction and commands are to be embed in to the functions

* 0x38 is used for 8-bit data initialization.
* 0xoC for making LCD display on and cursor off.
* 0X01 for clearing the display of the LCD.
* 0x80 for positioning the cursor at first line .

**Sending Commands to the LCD**

* E=1; enable pin should be high
* RS=0; Register select should be low for sending commands
* Placing the data on the data registers
* R/W=0; Read/Write pin should be low for writing the data.

**Writing the Data to the LCD**

* E=1; enable pin should be high
* RS=1; Register select should be high for writing data
* Placing the data on the data registers
* R/W=0; Read/Write pin should be low for writing the data.

**Important Post –** [**Interfacing 7 Segment Display with 8051 Microcontroller**](https://www.electronicshub.org/interfacing-7-segment-display-8051/)

**Code**

|  |
| --- |
|  |
| #include<reg51.h> | |
|  | |

|  |
| --- |
| #define lcd P0 |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| sbit rs=P2^0; |
|  |

|  |
| --- |
| sbit e=P2^1; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void delay (int); |
|  |

|  |
| --- |
| void cmd (char); |
|  |

|  |
| --- |
| void display (char); |
|  |

|  |
| --- |
| void custom (void); |
|  |

|  |
| --- |
| void string (char \*); |
|  |

|  |
| --- |
| void init (void); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| unsigned char custom\_char[]= {0x00,0x04,0x0E,0x04,0x15,0x0E,0x04,0x00, |
|  |

|  |
| --- |
| 0x00,0x04,0x0E,0x0E,0x0E,0x1F,0x04,0x00, |
|  |

|  |
| --- |
| 0x00,0x0A,0x15,0x11,0x11,0x0A,0x04,0x00, |
|  |

|  |
| --- |
| 0x0E,0x11,0x11,0x1F,0x1B,0x1B,0x1B,0x1F, |
|  |

|  |
| --- |
| 0x0E,0x11,0x10,0x1F,0x1B,0x1B,0x1B,0x1F, |
|  |

|  |
| --- |
| 0x00,0x0E,0x15,0x1B,0x0E,0x0E,0x00,0x00, |
|  |

|  |
| --- |
| }; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void delay (int d) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| unsigned char i; |
|  |

|  |
| --- |
| for(;d>0;d--) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| for(i=250;i>0;i--); |
|  |

|  |
| --- |
| for(i=248;i>0;i--); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void cmd (char c) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| lcd=c; |
|  |

|  |
| --- |
| rs=0; |
|  |

|  |
| --- |
| e=1; |
|  |

|  |
| --- |
| delay(5); |
|  |

|  |
| --- |
| e=0; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void display (char c) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| lcd=c; |
|  |

|  |
| --- |
| rs=1; |
|  |

|  |
| --- |
| e=1; |
|  |

|  |
| --- |
| delay(5); |
|  |

|  |
| --- |
| e=0; |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void custom (void) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| int k; |
|  |

|  |
| --- |
| cmd(0x40); |
|  |

|  |
| --- |
| for(k=0;k<48;k++) |
|  |

|  |
| --- |
| display(custom\_char[k]); |
|  |

|  |
| --- |
| cmd(0x80); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void string (char \*p) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| while(\*p) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| display(\*p++); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void init (void) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| cmd(0x38); |
|  |

|  |
| --- |
| cmd(0x0c); |
|  |

|  |
| --- |
| cmd(0x01); |
|  |

|  |
| --- |
| cmd(0x80); |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| void main() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| init(); |
|  |

|  |
| --- |
| custom(); |
|  |

|  |
| --- |
| string("Electronics Hub "); |
|  |

|  |
| --- |
| cmd(0xc0); |
|  |

|  |
| --- |
| string(" Presents "); |
|  |

|  |
| --- |
| delay(2000); |
|  |

|  |
| --- |
| cmd(0x01); |
|  |

|  |
| --- |
| cmd(0x80); |
|  |

|  |
| --- |
| string(" Custom Chars "); |
|  |

|  |
| --- |
| cmd(0xc0); |
|  |

|  |
| --- |
| string(" on 16x2 LCD "); |
|  |

|  |
| --- |
| delay(2000); |
|  |

|  |
| --- |
| cmd(0x01); |
|  |

|  |
| --- |
| cmd(0x80); |
|  |

|  |
| --- |
| string("ABCDEFGHIJKLMNOP"); |
|  |

|  |
| --- |
| cmd(0xc0); |
|  |

|  |
| --- |
| string("1234567890"); |
|  |

|  |
| --- |
| cmd(0xca); |
|  |

|  |
| --- |
| display(0); |
|  |

|  |
| --- |
| display(1); |
|  |

|  |
| --- |
| display(2); |
|  |

|  |
| --- |
| display(3); |
|  |

|  |
| --- |
| display(4); |
|  |

|  |
| --- |
| display(5); |
|  |

|  |
| --- |
| while(1); |
|  |

|  |
| --- |
| } |

[view raw](https://gist.github.com/elktros/24f453b1c5ac5bb0a3f96ad33b1edf4a/raw/81e81ff04cff7a7e87313cb4622ab69baeb25663/8051_16x2_LCD_Interface.c) [8051\_16x2\_LCD\_Interface.c](https://gist.github.com/elktros/24f453b1c5ac5bb0a3f96ad33b1edf4a#file-8051_16x2_lcd_interface-c) hosted with by [GitHub](https://github.com)

**Additional Codes**

The programs given below will use above functions and display the complete string which is given by the programmer to display the data. We have provided two demo codes working properly and easy to understand.

**Code 1**

|  |
| --- |
|  |
| #include <REG51.H> | |
|  | |

|  |
| --- |
| #include <string.h> |
|  |

|  |
| --- |
| #include <stdio.h> |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| sfr  LCD=0x80; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| sbit EN=P2^1; |
|  |

|  |
| --- |
| sbit RS=P2^0; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void nop(void); |
|  |

|  |
| --- |
| void delay\_1s(unsigned char t); |
|  |

|  |
| --- |
| void initial\_lcd(void); |
|  |

|  |
| --- |
| void delay(void); |
|  |

|  |
| --- |
| void string\_to\_lcd(unsigned char \*s); |
|  |

|  |
| --- |
| void write\_lcd(unsigned char dat,unsigned int com); |
|  |

|  |
| --- |
| void delay\_50ms(unsigned char x); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void main() |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| P1=0xff; |
|  |

|  |
| --- |
| P2=0xff; |
|  |

|  |
| --- |
| P3=0xff; |
|  |

|  |
| --- |
| delay\_50ms(4); |
|  |

|  |
| --- |
| initial\_lcd(); |
|  |

|  |
| --- |
| write\_lcd(0x80,0); |
|  |

|  |
| --- |
| string\_to\_lcd("   Welcome to   "); |
|  |

|  |
| --- |
| write\_lcd(0xc0,0); |
|  |

|  |
| --- |
| string\_to\_lcd("Electronics Hub "); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void nop(void) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| unsigned char n; |
|  |

|  |
| --- |
| for(n=0;n<20;n++); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| //..................delay routine.................// |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void delay\_1s(unsigned char t) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| unsigned char i,j; |
|  |

|  |
| --- |
| for(i=0;i<t;i++) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| for(j=0;j<20;j++) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| TMOD=0x01; |
|  |

|  |
| --- |
| TH0=0x3c;              //for 12MHz   (12/12MHZ)=1u>per cycle operation |
|  |

|  |
| --- |
| TL0=0xb0;            //50ms delay get (50m/1u)=50000; |
|  |

|  |
| --- |
| TR0=1;                 //Load value is =65536-50000=15536(=3cb0H) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| while(TF0!=1);  //wait for overflow flag |
|  |

|  |
| --- |
| TF0=0; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void initial\_lcd(void) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| write\_lcd(0x38,0); |
|  |

|  |
| --- |
| write\_lcd(0x0c,0); |
|  |

|  |
| --- |
| write\_lcd(0x01,0); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void write\_lcd(unsigned char dat,unsigned int com) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| RS=com; |
|  |

|  |
| --- |
| LCD=dat;nop(); |
|  |

|  |
| --- |
| EN=1;nop(); |
|  |

|  |
| --- |
| EN=0; |
|  |

|  |
| --- |
| nop(); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void string\_to\_lcd(unsigned char \*s) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| unsigned char i,l; |
|  |

|  |
| --- |
| l=strlen(s); |
|  |

|  |
| --- |
| for(i=0;i<l;i++) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| write\_lcd(\*s,1);delay\_50ms(1); |
|  |

|  |
| --- |
| s++; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void delay\_50ms(unsigned char x) |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| unsigned char i; |
|  |

|  |
| --- |
| for(i=0;i<x;i++) |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| TMOD=0x01; |
|  |

|  |
| --- |
| TH0=0x3c; |
|  |

|  |
| --- |
| TL0=0xb0; |
|  |

|  |
| --- |
| TR0=1; |
|  |

|  |
| --- |
| while(!TF0); |
|  |

|  |
| --- |
| TF0=0; |
|  |

|  |
| --- |
| TR0=0; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |

[view raw](https://gist.github.com/elktros/d8fb96c15cbd05faa9bd911964020dbd/raw/8749e56eaff72e6221384ba06b28065581e0e82b/8051_16x2_LCD_Interface_Code1.c) [8051\_16x2\_LCD\_Interface\_Code1.c](https://gist.github.com/elktros/d8fb96c15cbd05faa9bd911964020dbd#file-8051_16x2_lcd_interface_code1-c) hosted with by [GitHub](https://github.com)

**Code 2**

|  |
| --- |
|  |
| #include<reg51.h> | |
|  | |

|  |
| --- |
| #define cmdport P2 |
|  |

|  |
| --- |
| #define dataport P0 |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| sbit rs = cmdport^0;        //register select pin |
|  |

|  |
| --- |
| sbit e = cmdport^1;        //enable pin |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void delay(unsigned int msec)    //Function to provide time delay in msec. |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| int i,j ; |
|  |

|  |
| --- |
| for(i=0;i<msec;i++) |
|  |

|  |
| --- |
| for(j=0;j<1275;j++); |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void lcdcmd(unsigned char item)    //Function to send command to LCD |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| dataport = item; |
|  |

|  |
| --- |
| rs= 0; |
|  |

|  |
| --- |
| e=1; |
|  |

|  |
| --- |
| delay(1); |
|  |

|  |
| --- |
| e=0; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void lcddata(unsigned char item)    //Function to send data to LCD |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| dataport = item; |
|  |

|  |
| --- |
| rs= 1; |
|  |

|  |
| --- |
| e=1; |
|  |

|  |
| --- |
| delay(1); |
|  |

|  |
| --- |
| e=0; |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| void main() |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| lcdcmd(0x38);    //for using 8-bit 2 row mode of LCD |
|  |

|  |
| --- |
| delay(100); |
|  |

|  |
| --- |
| lcdcmd(0x0C);    //turn display ON and cursor OFF |
|  |

|  |
| --- |
| delay(100); |
|  |

|  |
| --- |
| lcdcmd(0x01);    //clear screen |
|  |

|  |
| --- |
| delay(100); |
|  |

|  |
| --- |
|  |
| delay(100); |
|  |

|  |
| --- |
| lcddata('A'); |
|  |

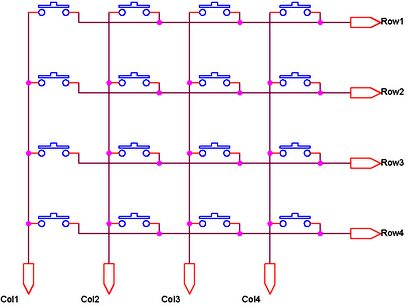
|  |
| --- |
|  |
|  |

|  |
| --- |
| } |

[view raw](https://gist.github.com/elktros/94204740f11bb237682b01dba4a42c7f/raw/99b04328c0fe31f3b807ae11bdf6435963e35c68/8051_16x2_LCD_Interface_Code2.c) [8051\_16x2\_LCD\_Interface\_Code2.c](https://gist.github.com/elktros/94204740f11bb237682b01dba4a42c7f#file-8051_16x2_lcd_interface_code2-c) hosted with by [GitHub](https://github.com)

# Matrix Keypad

Matrix Keypads are commonly used in calculators, telephones etc where a number of input switches are required. We know that matrix keypad is made by arranging push button switches in row and columns. In the straight forward way to connect a 4×4 keypad (16 switches) to a microcontroller we need 16 inputs pins. But by connecting switches in the following way we can read the status of each switch using 8 pins of the microcontroller.

4×4-Matrix-Keypad

The status of each keys can be determined by a process called Scanning. For the sake of explanation lets assume that all the column pins (Col1 – Col4) are connected to the inputs pins and all the row pins are connected to the output pins of the microcontroller. In the normal case all the column pins are pulled up (HIGH state) by internal or external pull up resistors. Now we can read the status of each switch through scanning.

1. A logic LOW is given to Row1 and others (Row2 – Row-4) HIGH
2. Now each Column is scanned. If any switch belongs to 1st row is pressed corresponding column will pulled down (logic LOW) and we can detect the pressed key.
3. This process is repeated for all rows.

If you need to save more pins of your microcontroller then you can interface keypad using the ADC module of your microcontroller.

# Interfacing with 8051 Microcontroller

## Circuit diagram

Interfacing Keypad with 8051 Microcontroller using Keil C

10KΩ resistor and 10μF will provide the required Power On Reset (POR) signal to the 8051 microcontroller. 12MHz crystal is used to provide required clock for the microcontroller and 22pF capacitors will stabilize the oscillations of the crystal. AT89C51 can works upto 24MHz. We can choose the required frequency by changing the crystal and clock frequency in the project settings of Keil C. Keypad is connected to the Port P1 and column inputs pins are pulled up internally. 16×2 [LCD](https://electrosome.com/lcd-display-fundamentals/) is connected to Port P2 and P0. P0.0 and P0.1 pins are pulled up externally using 10KΩ resistors since Port P0 has no internal pull up.

## Keil C Code

#include<reg52.h> //including sfr registers for ports of the controller

#include<lcd.h>

//LCD Module Connections

sbit RS = P0^0;

sbit EN = P0^1;

sbit D0 = P2^0;

sbit D1 = P2^1;

sbit D2 = P2^2;

sbit D3 = P2^3;

sbit D4 = P2^4;

sbit D5 = P2^5;

sbit D6 = P2^6;

sbit D7 = P2^7;

//End LCD Module Connections

//Keypad Connections

sbit R1 = P1^0;

sbit R2 = P1^1;

sbit R3 = P1^2;

sbit R4 = P1^3;

sbit C1 = P1^4;

sbit C2 = P1^5;

sbit C3 = P1^6;

sbit C4 = P1^7;

//End Keypad Connections

void Delay(int a)

{

int j;

int i;

for(i=0;i<a;i++)

{

for(j=0;j<100;j++)

{

}

}

}

char Read\_Keypad()

{

C1=1;

C2=1;

C3=1;

C4=1;

R1=0;

R2=1;

R3=1;

R4=1;

if(C1==0){Delay(100);while(C1==0);return '7';}

if(C2==0){Delay(100);while(C2==0);return '8';}

if(C3==0){Delay(100);while(C3==0);return '9';}

if(C4==0){Delay(100);while(C4==0);return '/';}

R1=1;

R2=0;

R3=1;

R4=1;

if(C1==0){Delay(100);while(C1==0);return '4';}

if(C2==0){Delay(100);while(C2==0);return '5';}

if(C3==0){Delay(100);while(C3==0);return '6';}

if(C4==0){Delay(100);while(C4==0);return 'X';}

R1=1;

R2=1;

R3=0;

R4=1;

if(C1==0){Delay(100);while(C1==0);return '1';}

if(C2==0){Delay(100);while(C2==0);return '2';}

if(C3==0){Delay(100);while(C3==0);return '3';}

if(C4==0){Delay(100);while(C4==0);return '-';}

R1=1;

R2=1;

R3=1;

R4=0;

if(C1==0){Delay(100);while(C1==0);return 'C';}

if(C2==0){Delay(100);while(C2==0);return '0';}

if(C3==0){Delay(100);while(C3==0);return '=';}

if(C4==0){Delay(100);while(C4==0);return '+';}

return 0;

}

void main()

{

int i=0;

char c,p;

Lcd8\_Init();

while(1)

{

Lcd8\_Set\_Cursor(1,1);

Lcd8\_Write\_String("Keys Pressed:");

Lcd8\_Set\_Cursor(2,1);

Lcd8\_Write\_String("Times:");

while(!(c = Read\_Keypad()));

p=c;

while(p==c)

{

i++;

Lcd8\_Set\_Cursor(1,14);

Lcd8\_Write\_Char(c);

Lcd8\_Set\_Cursor(2,7);

Lcd8\_Write\_Char(i+48);

Delay(100);

while(!(c = Read\_Keypad()));

}

i=0;

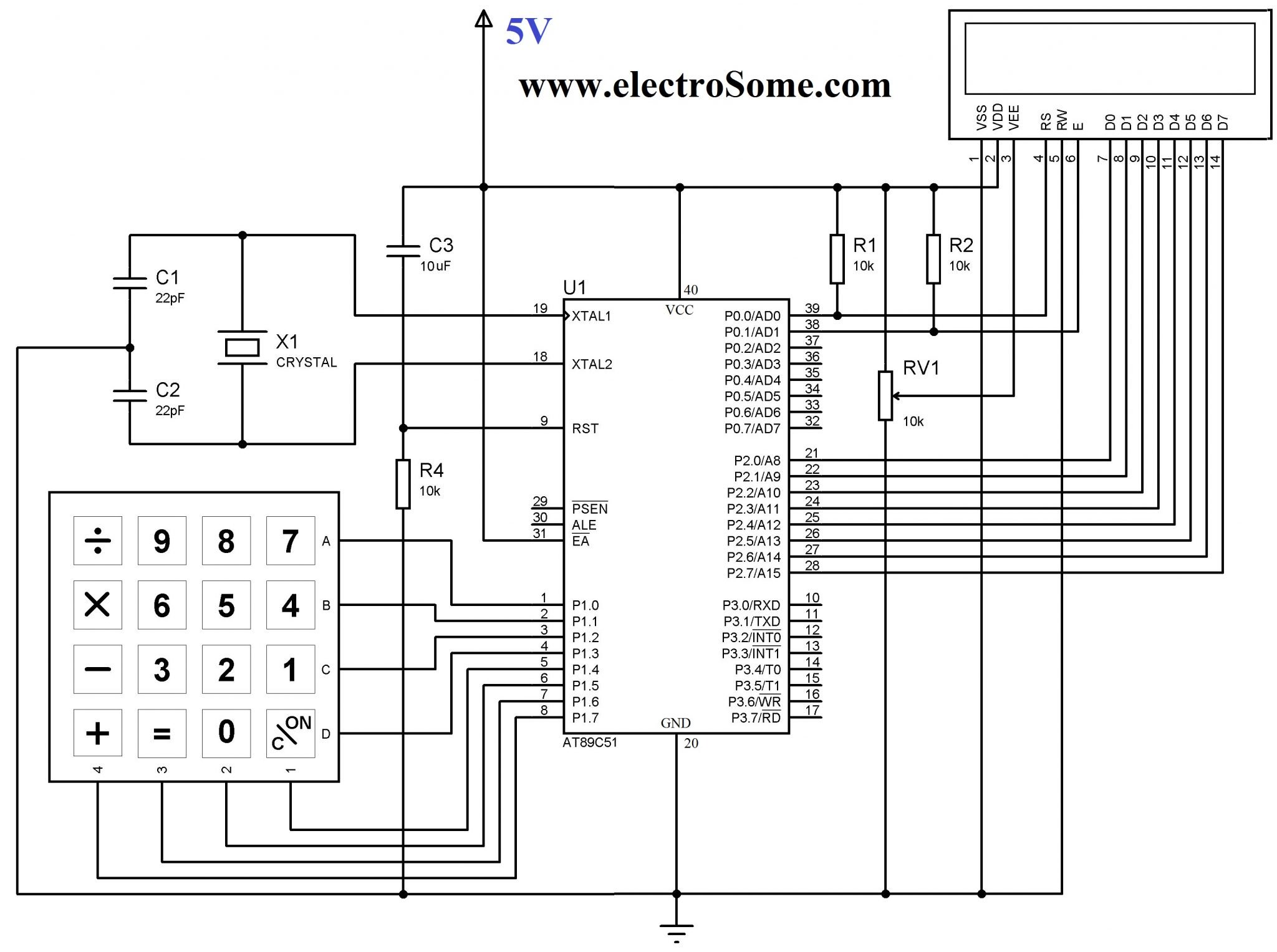
Lcd8\_Clear();

}

}

The code consists of two user defined functions. The **Delay()** is used to make delay in the program execution. The **Read\_Keypad()** reads the keypad. If any key is pressed it waits until the key is released and returns the corresponding character. If no key is being pressed it returns zero. As told before one of the pins of the row is kept at a logic 0 and the columns are checked for a logic 0. If a logic 0 is found the function returns a value according to the key pressed. This process is repeated for each row until a pressed key is found. If not found it will return zero.

This example project will show the character of key pressed and the number of times that key is pressed on the LCD screen.



**while(!(c = Read\_Keypad()))**is used in the program to read keypad. It is a locking call as it will wait until a key is pressed and released.

**Note :** The lcd is interfaced using the ‘lcd.h’ header file which is to be included in the project folder. For more details about LCD interfacing please read our article [Interfacing LCD with 8051 Microcontroller](https://electrosome.com/interfacing-lcd-with-8051-using-keil-c-at89c51/).

You can download Keil C files and Proteus files here…  
[Interfacing Keypad with 8051 using Keil C](https://electrosome.com/wp-content/uploads/2013/06/Interfacing-Keypad-with-8051-using-Keil-C.zip)