Interfacing LCD with Atmega32 Microcontroller using Atmel Studio

Interfacing LCD with Atmega32 Microcontroller using Atmel Studio

20 May By Ligo George ATMEL AVR, Tutorials Atmega32, Atmel, Atmel Studio, AVR, Microcontroller, Proteus, Tutorials
75 Comments

As we all know LCD (Liquid Crystal Display) is an electronic display which is commonly used nowadays in applications such as calculators, laptops, tablets, mobile phones etc. 16×2 character LCD module is a very basic module which is commonly used by electronic hobbyists and is used in many electronic devices and project. It can display 2 lines of 16 character and each character is displayed using 5×7 or 5×10 pixel matrix.

Contents

Interfacing 16×2 LCD with Atmega32 Atmel AVR Microcontroller using Atmel Studio is bit complex as there is no built in libraries. To solve this difficulty we developed a LCD library which includes the commonly used features. Just include our header file and enjoy. You can download the header file from the bottom of this article.

16×2 Character LCD

16×2 LCD can be interfaced with a microcontroller in 8 Bit or 4 Bit mode. These differs in how data and commands are send to LCD. In 8 Bit mode character data (as 8 bit ASCII) and LCD command are sent through the data lines D0 to D7. That is 8 bit data is send at a time and data strobe is given through E of the LCD.

But 4 Bit mode uses only 4 data lines D4 to D7. In this 8 bit data is divided into two parts and are sent sequentially through the data lines. The idea of 4 bit communication is introduced to save pins of microcontroller. 4 bit communication is bit slower than 8 bit but this speed difference has no significance as LCDs are slow speed devices. Thus 4 bit mode data transfer is most commonly used.

Function in lcd.h

Lcd8_Init() & Lcd4_Init() : These functions will initialize the 16×2 LCD module connected to the microcontroller pins defined by the following constants.

For 8 Bit Mode :

```
#define D0 eS_PORTD0
#define D1 eS_PORTD1
#define D2 eS_PORTD2
#define D3 eS_PORTD3
#define D4 eS_PORTD4
#define D5 eS_PORTD5
#define D6 eS_PORTD6
#define D7 eS_PORTD7
#define RS eS_PORTC6
#define EN eS_PORTC7
```

For 4 Bit Mode :

```
#define D4 eS_PORTD4
#define D5 eS_PORTD5
#define D6 eS_PORTD6
#define D7 eS_PORTD7
#define RS eS_PORTC6
#define EN eS_PORTC7
```

These connections must be defined for the proper working of the LCD library. **Don't forget to define these pins as Output.**

Lcd8_Clear() & Lcd4_Clear() : Calling these functions will clear the 16×2 LCD display screen when interfaced with 8 bit and 4 bit mode respectively.

Lcd8_Set_Cursor() & Lcd4_Set_Cursor() : These function will set the cursor position on the LCD screen by specifying its row and column. By using these functions we can change the position of character and string displayed by the following functions.

Lcd8_Write_Char() & Lcd4_Write_Char() : These functions will write a single character to the LCD screen and the cursor position will be incremented by one.

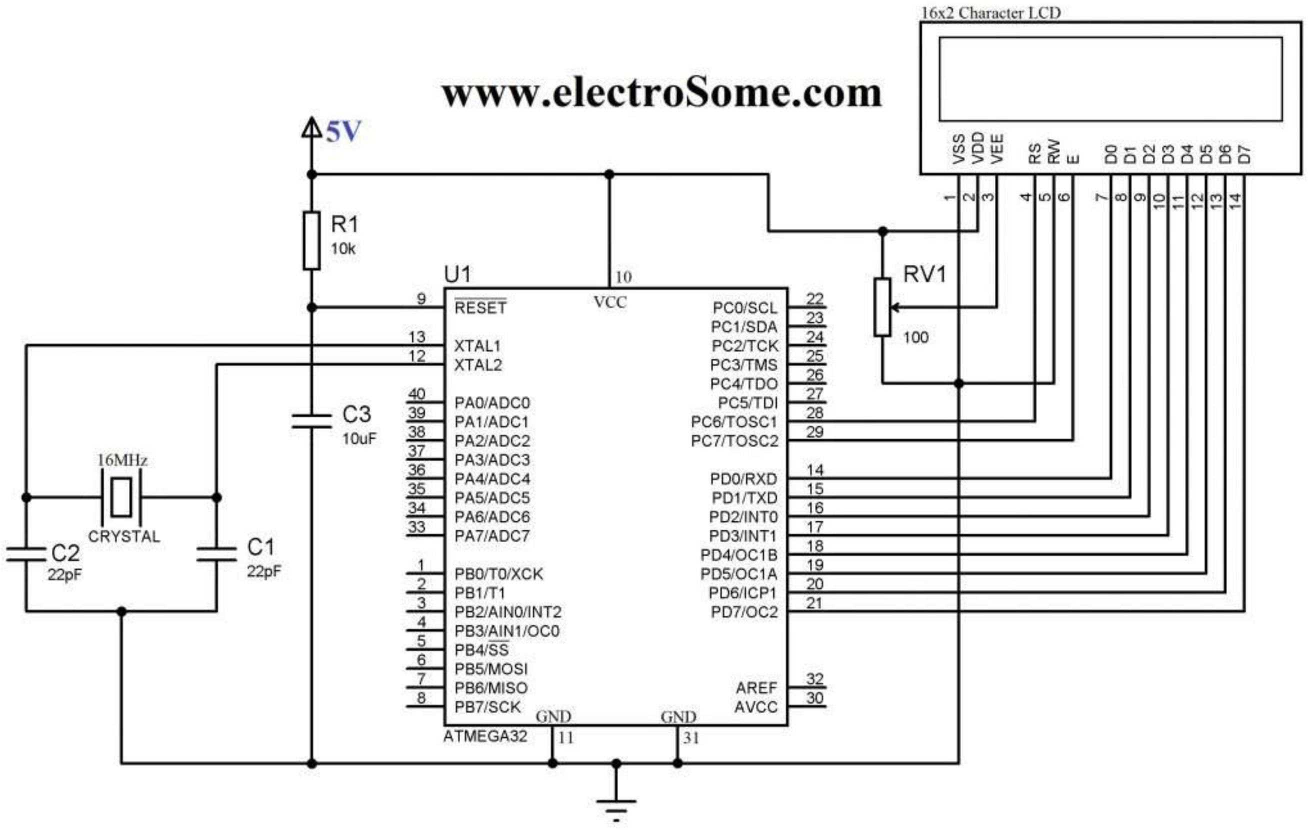
Lcd8_Write_String() & Lcd8_Write_String() : These function will write string or text to the LCD screen and the cursor positon will be incremented by length of the string plus one.

Lcd8_Shift_Left() & Lcd4_Shift_Left() : This function will shift data in the LCD display without changing data in the display RAM.

Lcd8_Shift_Right() & Lcd8_Shift_Right() : This function will shift data in the LCD display without changing data in the display RAM.

8 Bit Mode Interfacing

Circuit Diagram



Interfacing LCD with Atmega32 Microcontroller 8 Bit Mode

Atmel Studio C Code

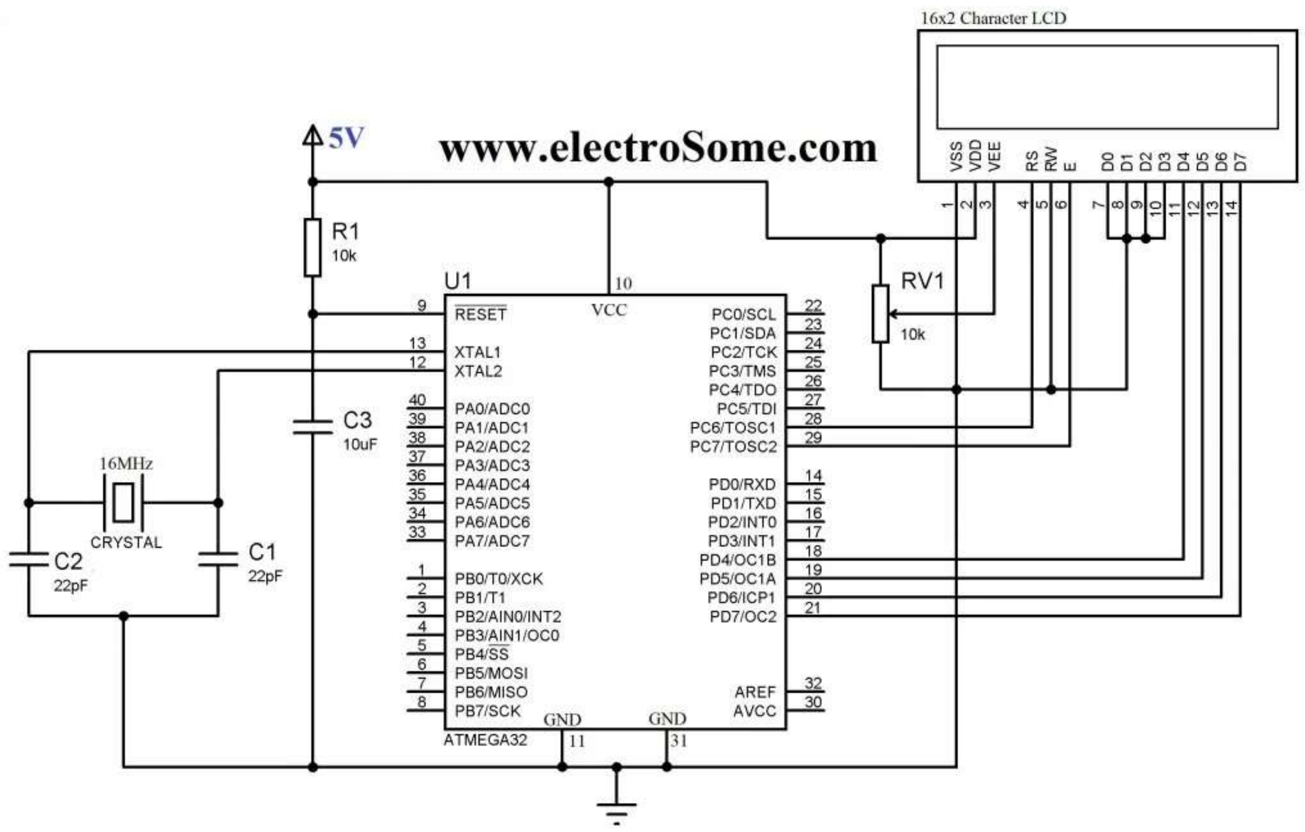
```
#ifndef F_CPU
#define F_CPU 16000000UL // 16 MHz clock speed
#endif

#define D0 eS_PORTD0
#define D1 eS_PORTD1
#define D2 eS_PORTD2
#define D3 eS_PORTD3
#define D4 eS_PORTD4
#define D5 eS_PORTD5
#define D6 eS_PORTD6
#define D7 eS_PORTD7
#define RS eS_PORTC6
#define EN eS_PORTC7

#include <avr/io.h>
#include <util/delay.h>
#include "lcd.h" //Can be download from the bottom of this article
int main(void)
{
  DDRD = 0xFF;
  DDRC = 0xFF;
  int i;
  Lcd8_Init();
  while(1)
  {
    Lcd8_Set_Cursor(1,1);
    Lcd8_Write_String("electroSome LCD Hello World");
    for(i=0;i<15;i++)
    {
      _delay_ms(500);
      Lcd8_Shift_Left();
    }
    for(i=0;i<15;i++)
    {
      _delay_ms(500);
      Lcd8_Shift_Right();
    }
    Lcd8_Clear();
    Lcd8_Write_Char('e');
    Lcd8_Write_Char('S');
    _delay_ms(2000);
  }
}
```

4 Bit Mode Interfacing

Circuit Diagram



Interfacing LCD with Atmega32 Microcontroller 4 Bit Mode

Atmel Studio C Code

```
#ifndef F_CPU
#define F_CPU 16000000UL // 16 MHz clock speed
#endif
#define D4 eS_PORTD4
#define D5 eS_PORTD5
#define D6 eS_PORTD6
#define D7 eS_PORTD7
#define RS eS_PORTC6
#define EN eS_PORTC7

#include <avr/io.h>
#include <util/delay.h>
#include "lcd.h" //Can be download from the bottom of this article

int main(void)
{
  DDRD = 0xFF;
  DDRC = 0xFF;
  int i;
  Lcd4_Init();
  while(1)
  {
    Lcd4_Set_Cursor(1,1);
    Lcd4_Write_String("electroSome LCD Hello World");
    for(i=0;i<15;i++)
    {
      _delay_ms(500);
      Lcd4_Shift_Left();
    }
    for(i=0;i<15;i++)
    {
      _delay_ms(500);
      Lcd4_Shift_Right();
    }
    Lcd4_Clear();
    Lcd4_Set_Cursor(2,1);
    Lcd4_Write_Char('e');
    Lcd4_Write_Char('S');
    _delay_ms(2000);
  }
}
```

How to use the Header File?

You can download the lcd.h header file at the bottom of this article. Add the lcd.h to your project source group.

1. Right Click on your project folder on the solution explorer on the right side.
2. Add >> Existing Item
3. Then Browse lcd.h
4. Click Add

Optimize Code for More Efficiency

You already seen that by using our header file lcd.h, you can connect your 16×2 LCD to any of the output pins of the microcontroller. More coding is required for this feature which reduces the code efficiency and increases the size of hex file. You can solve this problem by making simple changes in the header file according to your LCD connections. For example consider above sample programs. I have used PORTD for sending data, 6th bit of PORTC as RS and 7th bit of PORTC as EN.

- Change Lcd4_Port(data) and Lcd8_Port(data) to PORTD = data
- Change pinChange(EN,1) to PORTC |= (1<<PC7)
- Change pinChange(EN,0) to PORTC &= ~(1<<PC7)
- Change pinChange(RS,1) to PORTB |= (1<<PB6)
- Change pinChange(RS,0) to PORTC &= ~(1<<PC6)

You can download header file, atmel studio files and proteus files here…

[Interfacing LCD with Atmega32 Microcontroller with Atmel Studio]

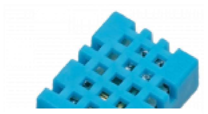
Like 4.8K people like this. Be the first of your friends.

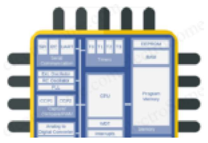
Related Posts:

Interfacing LCD with CloudX Board

What is a Microcontroller ?

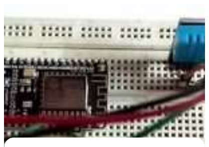
Bluetooth Home Automation With CloudX


Interfacing DHT11 Temperature and Humidity Sensor...


Microcontroller vs Microprocessor

Digital Alarm Clock using PIC Microcontroller and DS3234 RTC

Sending E-mail from ESP8266 - IoT Project


Interfacing DHT11 Sensor with ESP8266

Share this post

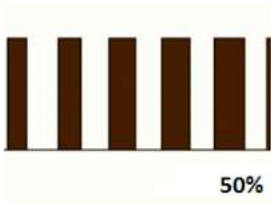
Author

Ligo George

[Follow Me on LinkedIn]

RELATED **POSTS**

[Custom Characters on LCD using PIC – MikroC]



[Generating PWM with PIC Microcontroller using Hi-Tech C]

[Configuration Bits in Mid-Range PIC Microcontrollers]

[Getting Sta with PIC 18 Microcontr MikroC]

January 5, 2015 | Ligo George | MikroC, PIC Microcontroller, Tutorials | 16x2, LCD, Microcontroller, MikroC, MikroC Pro, PIC, Proteus | 6 Comments

I hope that you already go through our tutorial, Interfacing Character LCD with PIC Microcontroller - MicroC Pro. These...

Read More →

October 23, 2013 | Ligo George | Hi-Tech C, PIC Microcontroller, Tutorials | CCP, Hi-Tech C, PIC, Proteus, PWM, Tutorials | 35 Comments

PWM (Pulse Width Modulation) is a powerful technique used to generate analog voltage using digital signals. It has a...

Read More →

February 13, 2014 | Ligo George | PIC Microcontroller, Tutorials | Embedded, Microcontroller, PIC, Tutorials | 1 Comment

Device Configuration Bits allows the programmer to adjust certain condition that determines the operation modes of the microcontroller. That...

Read More →

June 23, 2014 | Lig
MikroC, PIC Microc
Tutorials | 18F, Mic
MikroC, PIC, PIC 18
Tutorials | 3 Comm

In this tutorial we w
to program PIC 18F
Microcontrollers us
Pro compiler. I hop

Read More →

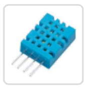
RECENT POSTS

Interfacing Relay with Arduino Uno
May 18, 2021

Remote Debugging of Embedded Systems
October 28, 2020

Getting Started with STM32 ARM Cortex-M Microcontroller using Keil IDE
September 4, 2020

Interfacing HC-05 Bluetooth Module with Arduino Uno
June 27, 2020


Interfacing DHT11 Temperature and Humidity Sensor with Arduino Uno
May 28, 2020

Interfacing L298N Motor Driver with Arduino Uno
May 21, 2020

FOLLOW US

CATEGORIES

- Electronics
  - 555 Circuits
  - Basic Electronics
    - Opamp
  - Electronics News
  - IC
  - Power Supply
  - Sensors
  - What is Inside?
- Featured
- Mobiles
- News
- Processors
- Projects
  - Embedded
    - Arduino
    - AVR Microcontroller
    - ESP8266
    - PIC Microcontroller
  - Hobby Circuits
  - IoT
- Promotional
- Softwares
- Tablets
- Technology Transfer
- Tutorials
  - 8051 Microcontroller
  - 8085
  - Arduino
  - ARM
    - ARM Cortex-M
      - STM32
    - ARM7
  - ATMEL AVR
  - CloudX
  - ESP8266

- MATLAB
- OpenWrt
- PCB Designing
  - DipTrace
  - OrCad
- PIC Microcontroller
  - CCS C Compiler
  - Hi-Tech C
  - MikroC
  - MPLAB XC8
- Python
- Raspberry Pi
  - Python Rpi
- SCILAB

TAGS

3D 16F877A 555 8051 MICROCONTROLLER ANDROID ARDUINO ARDUINO UNO ARM ATMEGA32 ATMEL AVR DC MOTOR DHT22 ELECTRONICS EMBEDDED ESP8266 GOOGLE HI-TECH C IOT L293D LCD LED MATLAB MICROCONTROLLER MIKROC MOBILE MOTOR MPLAB MPLAB XC8 PCB PIC PROTEUS PYTHON RASPBERRY PI SAMSUNG SENSOR SERVO MOTOR SMARTPHONE TABLET TEMPERATURE TRANSISTOR TRANSISTORS TUTORIALS ULTRASONIC XC8

RECENT COMMENTS

- Ligo George on Interfacing EM-18 RFID Module with PIC Microcontroller
- REMICHI Zohra on Interfacing EM-18 RFID Module with PIC Microcontroller
- Ligo George on USB PIC Programmer : PICKit2
- Steve on What is a Microprocessor ?
- W.u.s. sampath on USB PIC Programmer : PICKit2

FOLLOW US