Serial LCD quickstart

by MikeGrusin | January 17, 2011 | 21 comments Skill Level: ★ Beginner

Welcome!

Thank you for purchasing our serial-enabled LCD! This LCD will allow your project to display all kinds of text and numbers. LCDs can be difficult to use, so we've added an embedded processor to this LCD that does the hard work for you. This LCD is easy to connect to any 5V microprocessor that has a serial port, such as an Arduino, AVR, PIC, etc.

Connecting the hardware

There are only three connections you need to make to the LCD:

signal name	signal spec	wire color (JST cable)
RX (receive)	Serial receive (input to the display). 5V TTL level, 9600 baud (default rate, can be changed), 8 bits, 1 stop, no parity.	Yellow
GND (ground)	Ground for the power supply.	Black
VDD (power)	Power supply, this should be +5V (or 3.3V if you're using the LCD-9068) at up to 60mA if the backlight is fully on.	Red

The board has two headers with the above three signals. They are electrically identical, so you can use either one. One header is bare, the other has a 3-pin JST connector preattached to it that matches a JST cable. The cable only connects one way; press it in until it clicks. JST connectors are designed to be very snug; don't pull on the wires to disconnect it, see our tutorial on the proper way to disconnect JST cables.

(Tip: if you want to connect the display to a breadboard, tin the ends of the wires to make them easier to insert into the breadboard holes. To tin wire, strip about 1/4", and put some solder on the bare wire to make it stiffer).

Note that the RX input should be a 5V TTL-level signal directly from a 5V microcontroller or other 5V system. You should NOT connect the board to RS232-level voltages, which are +/-10V and will damage the board (see our explanation here). If you do wish to connect this display to RS232 signals, you can use a level-shifting board such as our PRT-00449 to translate the RS232 signals to TTL-level signals.

Using the display

When you power up the board, you'll briefly see a SparkFun splash screen, and then the display will go blank. To send text to the board, wait 1/2 second (500ms) after powerup for the splash screen to clear, then send text to the display through your serial port. The display understands all of the standard ASCII characters (upper and lowercase text, numbers, and punctuation), plus a number of graphic symbols and Japanese characters. See the HD44780 datasheet for the full list of supported characters.

If you send data that goes past the end of the first line, it will skip to the start of the second line. If you go past the end of the second line, the display will jump back up to the beginning of the first line. (Tip: you can simulate a scrolling window in software by copying the second line to the first line, and clearing the second line.)

Note that the Arduino and other systems with bootloaders may send "garbage" characters to the display while the system is starting up or being reprogrammed. To avoid this, you can use a software serial library to create a separate serial port from the USB port, as in the following examples.

NOTE that these examples were written for **Arduino 1.0 and later**. If you are using an older version of Arduino, you can download the older examples here: serial_lcd_quickstart_Arduino02.zip.

You can copy and paste these sketches into your Arduino 1.0 (or later) editing window, or download them here: serial_lcd_quickstart_Arduino10.zip.

```
// SparkFun Serial LCD example 1
// Clear the display and say "Hello World!"

// This sketch is for Arduino versions 1.0 and later
// If you're using an Arduino version older than 1.0, use
// the other example code available on the tutorial page.

// Use the softwareserial library to create a new "soft" ser
ial port
// for the display. This prevents display corruption when up
loading code.
#include <SoftwareSerial.h>

// Attach the serial display's RX line to digital pin 2
SoftwareSerial mySerial(3,2); // pin 2 = TX, pin 3 = RX (unu
sed)
```

Moving the cursor

A common LCD technique is to repeatedly display changing numbers such as RPM or temperature in the same place on the display. You can easily do this by moving the cursor before sending your data.

To move the cursor, send the special character 254 decimal (0xFE hex), followed by the cursor position you'd like to set. Each cursor position is represented by a number, see the table below to determine the number to send:

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
line 1	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
line 2	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207

For example, if you want to move to the beginning of the second line, send the bytes 254 192 (decimal).

Here's a slightly more complex example showing how to display data at fixed points on the display, plus the use of sprintf to convert numbers to strings (this right-justifies the numbers with leading spaces, which keeps them from "jumping around" if the number of digits changes):

```
// SparkFun Serial LCD example 2
// Format and display fake RPM and temperature data

// This sketch is for Arduino versions 1.0 and later
// If you're using an Arduino version older than 1.0, use
// the other example code available on the tutorial page.

// Use the softwareserial library to create a new "soft" ser
ial port
// for the display. This prevents display corruption when up
loading code.
#include <SoftwareSerial.h>

// Attach the serial display's RX line to digital pin 2
SoftwareSerial mySerial(3,2); // pin 2 = TX, pin 3 = RX (unu
sed)
```

More information

Other commands are available to change the backlight level, turn the splash screen on and off (and customize it to your own text), change the baud rate, etc. See the LCD datasheet for information on all the available commands.

For a more extensive example sketch that shows oyu how to create a scrolling marquee, create a timer, display sensor data and control the backlight, downlaod the following examples:

- SerLCD Arduino example - Arduino 0023 and earlier
- SerLCD Arduino example - Arduino 1.0.2 and later

Alternatively, you can use the SerLCD libary found on the Arduino website. If you are using Linux, you may want to try this library instead.

Tips and troubleshooting

If the display is powered up without the RX line connected to anything, the display may fill with strange characters. This is because the display is receiving random noise on the disconnected line. If you connect the RX line to a true TX port, this will not happen.

If the display is unreadable or washed out, the contrast may need to be adjusted. Send some text to the display (see the first example sketch above), then use a miniature Phillips screwdriver to gently turn the contrast trimpot labeled VR1 on the back of the display until the text is as clear as possible (please be gentle with the trimpot). This display also has a backlight that can be adjusted for best readability, see the LCD datasheet for information.

This display has a feature where if the display receives a CTRL-R character during its half-second splash screen display, it will temporarily revert to 9600 baud until power is cycled.

This is to allow you to regain control of the display if you set it to an unknown baud rate. Some systems like Arduino send bootloader information out the serial port when the system starts up, which can fool the LCD into this recovery mode. If this is a problem, there are a few solutions: you can use a different pin and the NewSoftSerial library to create a TX port that doesn't get used during startup (as shown in the example sketches above), or leave the display at the default 9600 baud rate, and clear the display when your program starts.

Questions?

Enjoy your new display! If you have any problems, feel free to contact SparkFun Technical Support at techsupport@sparkfun.com.

Comments 21 comments

Log in or register to post comments.

bboyho / about 3 years ago * / ★ 5 / 📌

– – – – – – – Tech Support Tips/Troubleshooting/Common Issues – – – – – – –

If you are seeing two rows of ASCII blocks ▉ similar to this image here => https://drive.google.com/open?id=0B0jwgLkjMWzDYXdUZWNaQWRvTmM, it's possible that you might have bricked the serial enabled LCD by putting it into an unknown state. This is a common problem if you are uploading code to the Arduino while another device is connected to the same hardware UART line (i.e. pin 0 and 1). This will cause the LCD to output random characters usually on the screen or not even show anything on the screen.

Changing Baud Rate

By reading the datasheet on section 3.4 https://www.sparkfun.com/datasheets/LCD/SerLCD_V2_5.PDF, it indicates that you need to send a command character **124** in decimal form or **0x7C** in hexadecimal. If you look at the ASCII table for CTRL-K through CTRL-P, there are hex values for the different baud rates http://academic.evergreen.edu/projects/biophysics/technotes/program/ascii_ctrl.htm that you would need to send to the LCD:

Therefore:

```
2400 baud, "<control>k" => 0x0B
4800 baud, "<control>l" => 0x0C
9600 baud, "<control>m" => 0x0D
14400 baud, "<control>n" => 0x0E
19200 baud, "<control>o" => 0x0F
38400 baud, "<control>p" => 0x10
reset to default baud while LCD is the splash screen is still active, "<control>r" => 0x12
```

There should be a tutorial on how to use the special command characters in this tutorial and example code that was posted on the product page under the documents section => https://www.sparkfun.com/tutorials/246. The section that says ***More information*** has a more extensive code labeled *"SerLCD Arduino example - Arduino 1.0.2 and later"* and will help you see how the other functions are written. All you have to do is create a new void function and call it when you are running the Arduino sketch file like this:

```
void changeBaud(){
  LCD.write(0x7C);// special command byte => 0d124 or 0x7C
  LCD.write(0x0B); //change current baud to 2400 baud
}
```

Once you change the baud rate, you need to recompile the code in Arduino and set the software serial baud to 2400 and comment out that function in order to use that baud rate. To do this for a baud rate of 9600, you would just send the special command byte and the associated hex value to the LCD.

Recovering from Unknown State

Sending Characters to the LCD

If the Serial LCD gets into an unknown state and you are not able to communicate with it anymore, just write **0x12** in the loop so that it is constantly sending the hex value to the screen when the LCD's splash screen is active (or when the LCD is powered) to reset the unit to 9600 baud similar to the methods explained above for changing the baud rate.

Here's some code with Arduino to try and unbrick the PIC16F88 on the serial enabled LCDs => https://github.com/bboyho/SerLCD . This sometimes works and there is a higher probability of recovering your LCD if you still are able to see the splash screen.

Pickit 3 Programmer

– -

A.) MPLAB IDE v8.66

The last resort is to use the Pickit 3 programmer and reupload the firmware on the LCD using *MPLAB v8.66* http://www.microchip.com/pagehandler/en-us/devtools/dev-tools-parts.html. There is code for the PIC16F88 that is used on the 20x4 Serial Enabled LCD Screen's product page where it says "Firmware" => https://www.sparkfun.com/products/9568. This code is specific to the 20x4 LCD screen. By clicking on *File>Import* in the menu of the MPLAB IDE, I was able to import the .hex file. To upload the "serlcd-v2_7_4line_10MHz.hex", you connect the PICkit 3 programmer https://www.sparkfun.com/products/9973 to the 1x6 header pin with the arrow pointing to the VPP pin. Then you select *Select Programmer> PICkit3* to set the programmer that you are using. Finally, by clicking on the *Programmer>Program,* this programs the PIC chip that is on the LCD.

– -

B.) MPLAB X IDE v2.30

I also used MPLAB X V2.30 www.microchip.com/pagehandler/en-us/family/mplabx/ and it was pretty straight forward in uploading the hex file to recover a bricked LCD:

1.) Select **File>Import>Hex/ELF… (Prebuilt) File** .

2.) You must type in the directory where the hex file is stored on your computer for the **Prebuilt Filename** which is *…serlcd-v2_7_4line_10MHz.hex* .

3.) Then type in the **Device** field: PIC16F88.

4.) Click **Next**

5.) Click **Finish** to save the project.

6.) Click on the icon that looks like the software is downloading to a PIC chip (i.e. the button next to the Run Project button) and select *Make and Program Device Programmer to Go PICkit3 (Project serlcd-v2_7_4line_10MHz)*.

7.) A message will pop up if you have not used the programmer with the software before:

```
ICD 3 not found. The last tool used for this project is unavailable. Please select the tool that you want to use to debug or program from the
following list of available tools.

 Select PICkit 3 in the tree.
```

8.) Click OK.

9.) The software will pop up with this message:

```
CAUTION: Check that the device selected in MPLAB IDE (PIC16F88) is the same one that is physically attached to the debug tool. Selecting a 5V
device when a 3.3V device is connected can result in damage to the device when the debugger checks the device ID. Do You wish to continue?
```

Click OK.

10.) From there, the hex file should have flashed on the chip and the program will run immediately after displaying this output:

```
**************************************************

Connecting to MPLAB PICkit 3...
Firmware Suite Version.....01.28.90 *
Firmware type..............Midrange

Target voltage detected
Target device PIC16F88 found.
Device ID Revision = 8

The following memory area(s) will be programmed:
program memory: start address = 0x0, end address = 0x4ab
configuration memory

Device Erased...

Programming...
Programming/Verify complete
```

Setting LCD Cursor Position

To set the LCD's cursor position, you would send the command character/flag. Then send additional number related to the cursor's position on the LCD screen. Just add *128* to the cursor's position as stated on page 3 of the datasheet [https://www.sparkfun.com/datasheets/LCD/SerLCD_V2_5.PDF] to place the cursor at the correct coordinates.

For example, if you are trying to place the cursor at (line 3, position 0), you would send the command character *0xFE* and the associated coordinates. Looking at the datasheet, for the 20x4 serial enabled LCD screen, it looks like 128+20 = *148* . Therefore:

```
selectLineThree(){
    LCD.write(0xFE); //command flag
    LCD.write(148); //line 3 , position 0
}
```

As another example, if you are trying to place the cursor at (line 4, position 0), you would send the command character *0xFE* and the associated coordinates. Looking at the datasheet, for the 20x4 serial enabled LCD screen, it looks like 128+84= *212* . Therefore:

```
selectLineFour(){
    LCD.write(0xFE); //command flag
    LCD.write(212); //line 4 , position 0
}
```

Using the Serial Enabled LCD on an Atmega32U4's Hardware UART

If you are using the serial enabled LCD with an Atmega32U4 (like a Pro Micro, Arduino Leonardo, Arduino LilyPad USB etc), you might need to add a small delay in the setup before you can get it working with the hardware UART. Here's an example:

```
///test example using ATmega32U4's hardware UART and delay
void setup() {
  delay(2000);//add delay so the ATmega32U4 can have a second before sending serial data to the LCD
  Serial1.begin(9600);//set up the hardware UART baud
}

void loop() {
  Serial1.print("print something");//send something to the serial enabled LCD
  delay(50);
}
```

 bboyho / about 3 years ago * / ★ 4 /

– – – – – – – Additional Tech Support Tips/Troubleshooting/Common Issues – – – – – – –

Extended Command 0x0C

By testing the extended LCD command 0x0C, the command used all three of the commands:

```
Turn visual display on
Underline cursor off
Blinking box cursor off
```

So if I had the cursor/blinking box on and turned the visual display off, the cursor/blinking box would not remain on after after issuing the 0xFE command and 0x0C value to turn the screen back on.

Changing Width and Character Lines

There is some code listed in this example [https://github.com/bboyho/SerLCD/blob/master/Arduino/SerLCD_Arduino_Example_v3/SerLCD_Arduino_Example_v3.ino#L66] based on the datasheet to configure the serial enabled LCD.

Creating Custom Characters/ User Defined Graphics

If you would like to create a custom character, you would need to send a command byte before controlling the individual pixels in the character square.

Try looking at the forums and datasheet for the LCD for more information:

Resources

HD44780 https://www.sparkfun.com/datasheets/LCD/HD44780.pdf

HD44780 LCD User-Defined Graphics: http://www.quinapalus.com/hd44780udg.html

SparkFun Forums: Custom Characters on the Serial Enabled LCD Backpack https://forum.sparkfun.com/viewtopic.php?f=14&t=29564

Arduino Forums: http://forum.arduino.cc/index.php?topic=19091.msg139423#msg139423

Arduino Due and Software Serial with Serial Enabled LCD

Unfortunately, you are not able to use the serial enabled LCDs with an Arduino Due due the differences in how change interrupts are used for the ARM processor. The software serial library is not included in the Arduino Due's tree => http://forum.arduino.cc/index.php?topic=142902.0 .

Old v2.5 Datasheet

The old datasheet for the serial enabled backpacks can be found here => https://www.sparkfun.com/datasheets/LCD/SerLCD_V2_5.PDF .

> Member #820509 / about 2 years ago / ★ 1
> Hello some answers to your due's trouble :
>
> I've been working for two years with the serial lcd module. You will find the arduino librairie suited to due, uno and mega on www.altituduino.xyz on download zone
>
> regards

Member #459779 / about 5 years ago / ★ 3

After sending the special command 124 and then 128-157 to adjust the brightness, you 'll have to wait half a second to let the display settle. It isn't mentioned in the LCD datasheet. Maybe other special commands 124 need this pause too.

>  Alfredomova / about 3 years ago / ★ 1
> yes! you're right! without delay(1000); the displays goes all dark and no text is shown! thank you!

Member #523181 / about 4 years ago / ★ 2

How do I set the cursor position on a 20x4 LCD? Anyone who know the correct decimals to send please help me.

> RobotRacer / about 4 years ago / ★ 1
> I have the exact same question

N00b_Programmer / about 7 years ago / ★ 2

I have ported LiquidCrystal library for use with the serial LCD you can look at my code here. Still working on finishing all the documentation. But putting up for now hopefully someone will find it usefull.
http://arduino.cc/playground/Code/SerLCD
-Thanks

> bboyho / about 3 weeks ago / ★ 1
>
> Thanks! I'm not sure if anyone responded to you but it has been added to this tutorial for a while. I am currently about to move this over to our learn.sparkfun.com site. It will be linked there with updated information. ;D

RocketGuy / about 3 years ago / ★ 1

Does anybody know if one can bodge this together with lcdproc without herculean effort? I expect to be using a usb/serial adapter, motherboards being sans serial these days.

> Nate / about 3 years ago / ★ 1
>
> Hmm. It should really only require an FTDI breakout, some jumper wires, and a screw driver.

Member #508423 / about 4 years ago / ★ 1

I hooked up the LCD display to an Arduino 2560. Every time I reprogrammed the 2560 through the USB the LCD would print garbage. Finally, after a couple of time I reprogrammed the 2560, the LCD would dim. To get it back to the right light level, I had to write the command to bring the light level up. Otherwise, it would stay at that level even after recycling power.

After reprogramming the 2560 a few more times, My character map changed. It looked like it went to a 20 character display or something. I tried resetting it back to the 16x2 by sending the 124 command then sending a 4. But that didn't work. Does anyone out there know how to reset this device back to its normal parameters?

Member #562224 / about 4 years ago / ★ 1

If you have a MEGA 2560, be advised that the SoftwareSerial.h library indicates that pin 2 (as used in the example code above) is not supported for RX. You must use one of the viable pins. I used Digital Pin 10 and was successful.

http://arduino.cc/en/Reference/SoftwareSerial

Not all pins on the Mega and Mega 2560 support change interrupts, so only the following can be used for RX: 10, 11, 12, 13, 14, 15, 50, 51, 52, 53, A8 (62), A9 (63), A10 (64), A11 (65), A12 (66), A13 (67), A14 (68), A15 (69).

Not all pins on the Leonardo support change interrupts, so only the following can be used for RX: 8, 9, 10, 11, 14 (MISO), 15 (SCK), 16 (MOSI).

> M-Short / about 4 years ago / ★ 1
>
> Also, keep in mind that you shouldn't need software serial on either of those boards. The code uses software serial because the only serial pins on the ATMega328 based boards are the ones connected to the USB port. The Mega2560 has 4 serial ports and the Leonardo has 1 serial port which is separate from USB. In both cases you should be able to use Serial1 instead of software serial.

ramezc / about 5 years ago * / ★ 1

Hi, have this 16x2 LCD. Works fine once I have disconnected and reconnected ground. pressing reset button on arduino or unpowering arduino freezes the lcd on the next run….. only re-work once i have disconnected and reconnected lcd ground (thus each time I m loosing the first commands of my program, until i have done this ground unplug/plug) . FYI, I m waiting half a second after my setup and any special command sent to the lcd… anyone knows what is going wrong tks for help

Member #456098 / about 5 years ago / ★ 1

This may be a stupid question, probably is. I just got an Mp3- Player shield, which I'm using with an Arduino Uno R3. Where exactly do I connect the LCD's RX pin to?

Any help would be very appreciated!

> gremlin88 / about 4 years ago / ★ 1
>
> Any pin not being used by the shield. Check the code to find these.

Member #184127 / about 6 years ago / ★ 1

Is there a command to make the text displayed scroll or must it be done manually?

Member #215251 / about 7 years ago / ★ 1

You use the newsoftserial library, but you don't elaborate. For those that want the library to make the examples work it is at:
http://arduiniana.org/libraries/newsoftserial/
You have to download and extract it to your arduino/libraries directory.
I was able to get it working relatively easily after soldering 22 wire to the RX,GND, and VDD. You include a cable but that was not very good as it had to be tined so not good with a breadboard. If you are going to include a cable make it one that works with a breadboard then the experts can just cut it but the newbs can still play.
I had to figure out the pin setup by looking at the sketches though.
I think you should mention the lib you are using, how to download and install it, where to find it, etc if you plan on using non-standard install libs.
Otherwise great product! It works fabulously.

NICK.KEEPER / about 5 years ago / ★ 2

as of 1.0 this is a part of the core, no need to download a library so long as you are using 1.0 or above.

MikeGrusin / about 6 years ago * / ★ 1

Thanks for the comments! We've updated the example sketches with more information about retrieving the required library (sorry about that, we use it so often we forgot that not everyone does), and created new examples for Arduino 1.0 (which has adopted NewSoftSerial as the default SoftwareSerial library).