Controlling the LCD

Display text and numerical values on an LCD.

Site: iCODE
Course: Machine Science Guides (Arduino Version)
Book: Controlling the LCD
Printed by: Ivan Rudnicki
Date: Wednesday, July 30, 2014, 03:32 PM

Contents

About LCDs

Adding the LCD

Sending Text to the LCD

Making Text Flash

Making Text Move

Displaying Numerical Values

Displaying ANSI Characters on the LCD

About LCDs

In a liquid crystal display (LCD), a liquid crystal solution is sandwiched between two thin plates of glass. In its normal state, the crystal solution is a clear liquid. However, when electric current passes through the solution, it crystallizes, turning opaque. Since LCD technology uses far less electricity than older cathode ray tube displays, it is quickly becoming the standard for both small-scale electronic devices and larger television sets, like those shown in Figure 1.

Figure 1. LCD screens.

The LCD included in the kit (Figure 2) is designed to display text only, on two 16-character lines.

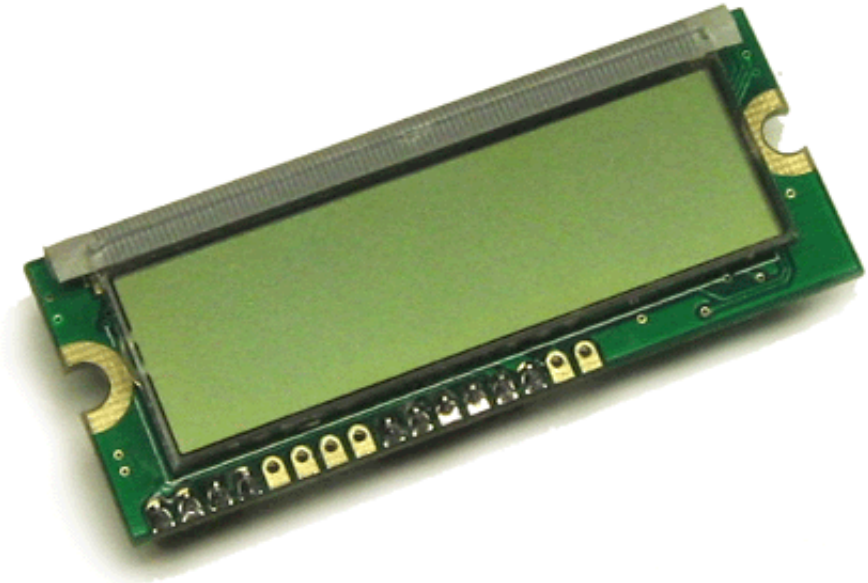

Figure 2. LCD in kit.

Adding the LCD

Following the schematic shown in Figure 3, connect the LCD to your ATmega Board. Note that it is possible to make all of the indicated chip connections by plugging the LCD into the breadboard on the top side of the chip. One additional jump wires to power and two wires to ground are required. Note that pin 1 on the LCD is labeled with a white numeral 1 on the underside of the device. Pins 2 to 6 and 11 to 14 are in order from left to right as you view the underside of the LCD (right to left, with the LCD plugged into the breadboard).

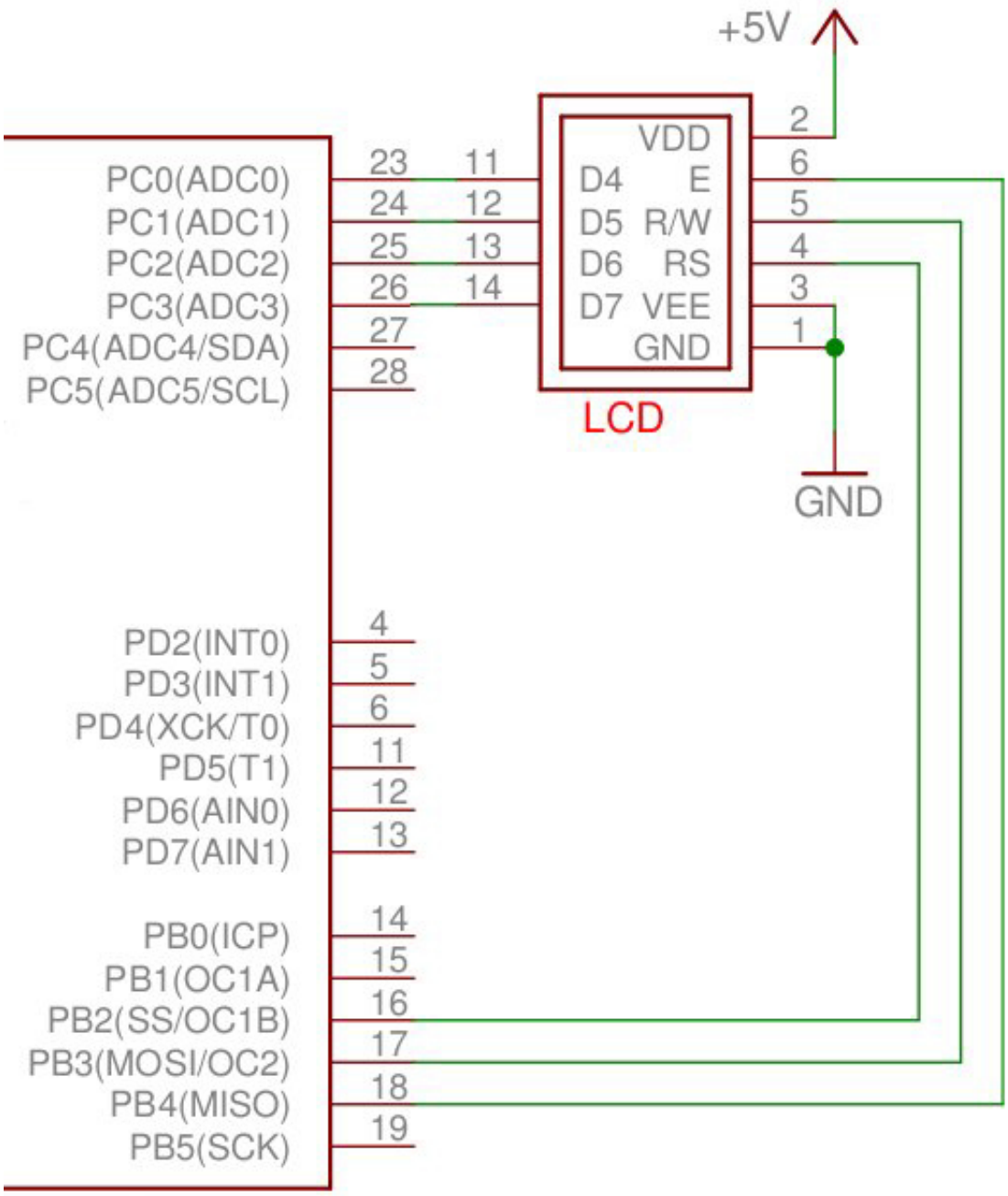

Figure 3. Adding the LCD.

Figures 4 and 5 shows one way to make the required connections for the LCD. Be careful when inserting the LCD into the breadboard. Misaligning the LCD pins can cause a short circuit and damage the device.

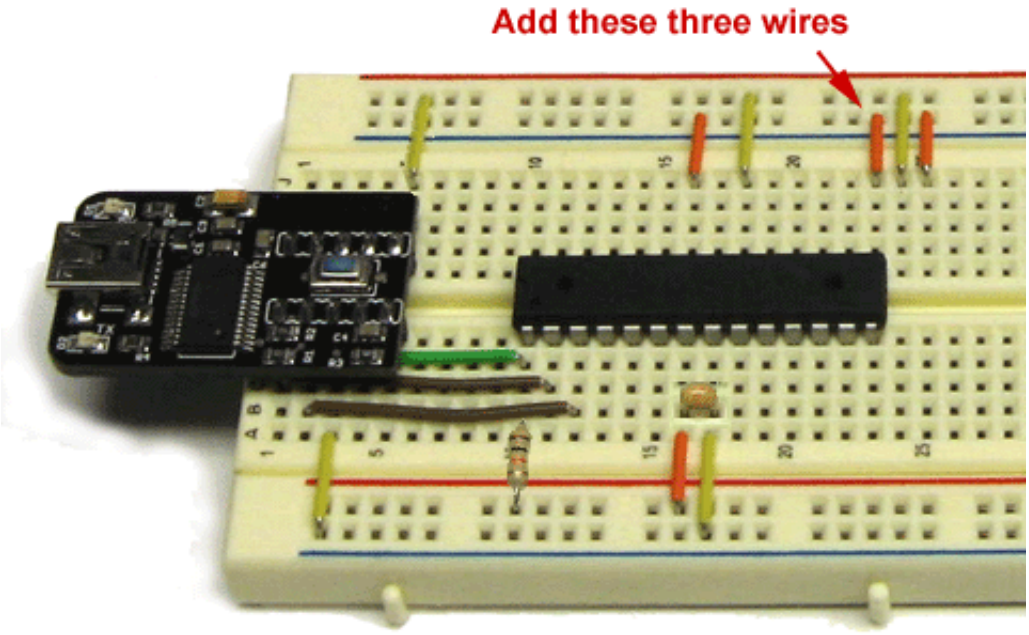

Figure 4. Jump wires for LCD.

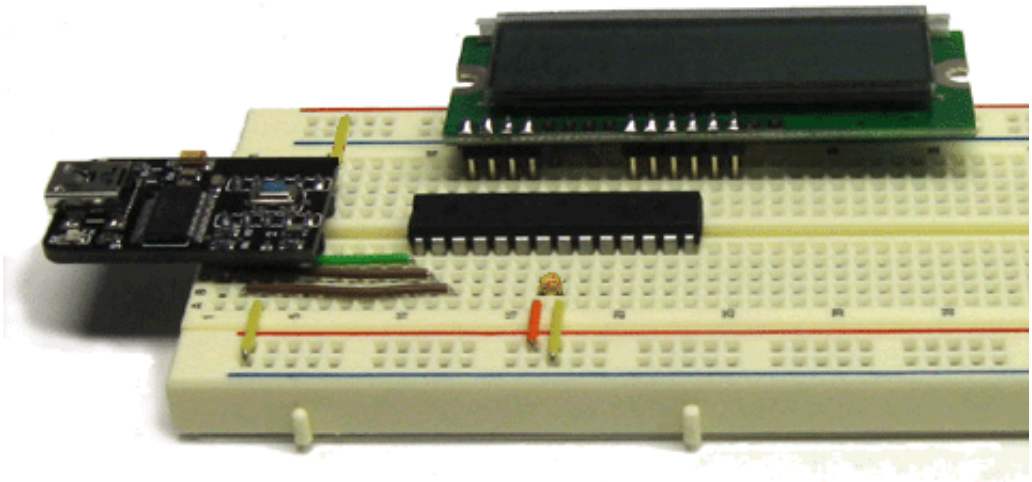

Figure 5. Aligning the LCD.

Sending Text to the LCD

The LCD displays up to 32 characters of text on two lines. To send a string of text to the LCD, you use a function called lcd_text, specifying two values:

- The position of the first character in the string
- The text string itself, enclosed in quotation marks

Figure 6 shows how the text position should be specified.

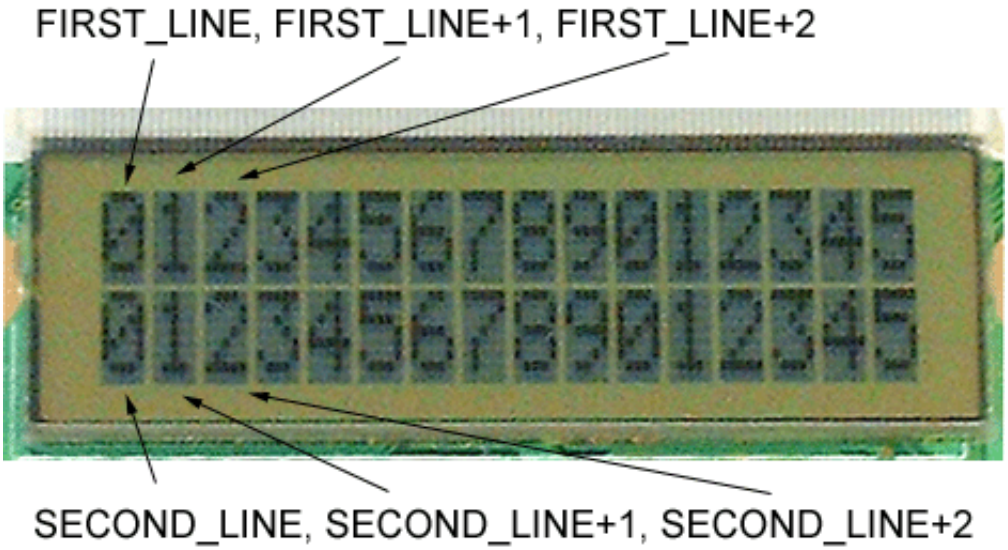

Figure 6. LCD text positions.

In this first program, you will display a simple line of text on the LCD: "Hello World!"

1. Rename your file text.c.
2. Modify your code file, as follows:

```
#include <mxapi.h>
#include <lcd.h>

int main(void)
{
   lcd_init();                                  //Initialize the LCD
   lcd_text(FIRST_LINE, "Hello World!");        //Send text to the LCD
   while(1==1);                                 //End the program
}
```

3. Compile and test your new code.
4. Check out the LCD. It should be displaying "Hello World!"

Programming Challenge

Change your code to get the microcontroller to display different text (e.g., your name). Try a few different lines of text. Next, reposition "Hello World!" in the lower right hand corner of the LCD. Then, place one line of text on Line 1 of the LCD and another line of text on Line 2 of the LCD.

Making Text Flash

By displaying a line of text, clearing the screen, and then displaying the text again, you can make your text flash on and off. With delay statements, you can control the precise rate of flashing.

1. Rename your code file flash.c.
2. Modify your code file, as follows:

```
#include <mxapi.h>
#include <lcd.h>

int main(void)
{
   lcd_init();                                  //Initialize the LCD
   while(1==1)
   {
      lcd_clear();                              //Clear the LCD
      delay_ms(500);                            //Wait 500 milliseconds
      lcd_text(FIRST_LINE, "Hello World!");     //Send text to the LCD
      delay_ms(500);                            //Wait 500 milliseconds
   }
}
```

3. Compile your code and download it to the microcontroller.

Programming Challenge

In your flash.c file, change all the delays to 5,000 milliseconds. Try a 5,000-microsecond delay, then a 200-millisecond delay.

Making Text Move

Text on the LCD can be made to move to the left or the right, using the following two statements: *lcd_instruction(DISPLAY_LEFT)* and *lcd_instruction(DISPLAY_RIGHT)*. Each of these statements takes all of the text on the LCD and moves it one character to the left or right. By pairing these statements with delay statements, you can make your text appear to scroll--like the "tickers" that are often used to display sports scores, stock prices, and news headlines.

1. Rename your code file scroll.c.
2. Modify your code file, as follows:

```
#include <mxapi.h>
#include <lcd.h>

int main(void)
{
   lcd_init();                              //Initialize the LCD
   lcd_text(FIRST_LINE, "Hello World!");    //Send text to the LCD
   delay_ms(200);                           //Wait 200 milliseconds
   lcd_instruction(DISPLAY_RIGHT);          //Move text to the right
   delay_ms(200);
   lcd_instruction(DISPLAY_RIGHT);
   delay_ms(200);
   lcd_instruction(DISPLAY_RIGHT);
   delay_ms(200);
   lcd_instruction(DISPLAY_RIGHT);
   delay_ms(200);
   lcd_instruction(DISPLAY_LEFT);
   delay_ms(200);
   lcd_instruction(DISPLAY_LEFT);
   delay_ms(200);
   lcd_instruction(DISPLAY_LEFT);
   delay_ms(200);
   lcd_instruction(DISPLAY_LEFT);
   while(1==1);
}
```

3. Compile your code and reprogram the microcontroller.
4. Check out the LCD. "Hello World!" should scroll to the right and then to the left. To restart your code, just press the reset button.

Programming Challenge

Change the lcd_instruction statements to move the text in new ways. What happens to text when it moves off the screen? Does it ever come back? Add code to display text on both lines of the LCD. Do both lines move in the same way? Change the delay_ms values to speed up or slow down the rate at which the text scrolls.

Displaying Numerical Values

The ATmega code library provides several different functions for displaying the value of variables on the LCD:

- lcd_decimal - displays the decimal value of the variable
- lcd_signed - displays the decimal value with a + or - sign
- lcd_binary - displays the binary value of the variable
- lcd_hexadecimal - displays the hexadecimal value of the variable

Each function takes three arguments:

- the position on the LCD - FIRST_LINE, SECOND_LINE, etc.
- the name of the variable
- the number of digits to display - e.g., if set to 3, 1 is displayed as 001

The code example below demonstrates how these arguments are used.

1. Rename your code file values.c.
2. Modify your code file, as follows:

```
#include <mxapi.h>
#include <lcd.h>

int main(void)
{
   signed int x=0;                    //Declare a signed integer variable
   lcd_init();                        //Initialize the LCD
   for(x=-128; x<=128; x++)           //Set up a for loop
   {
      lcd_signed(FIRST_LINE, x, 3);           //Display x in signed form
      lcd_hexidecimal(FIRST_LINE+13, x, 2);   //Display x in hex form
      lcd_binary(SECOND_LINE, x, 8);          //Display x in binary form
      delay_ms(200);                          //Wait 200 milliseconds
   }
   while(1==1);
}
```

3. Compile and test your new code.

Displaying ANSI Characters on the LCD

Another LCD control function, called lcd_character, takes a numeric value and displays the corresponding ANSI character. The ANSI character set is a standard set of letters, numbers, and symbols, any of which can be displayed on the LCD. Every number from 32 to 255 has a corresponding ANSI character, as shown in Figure 7 below.

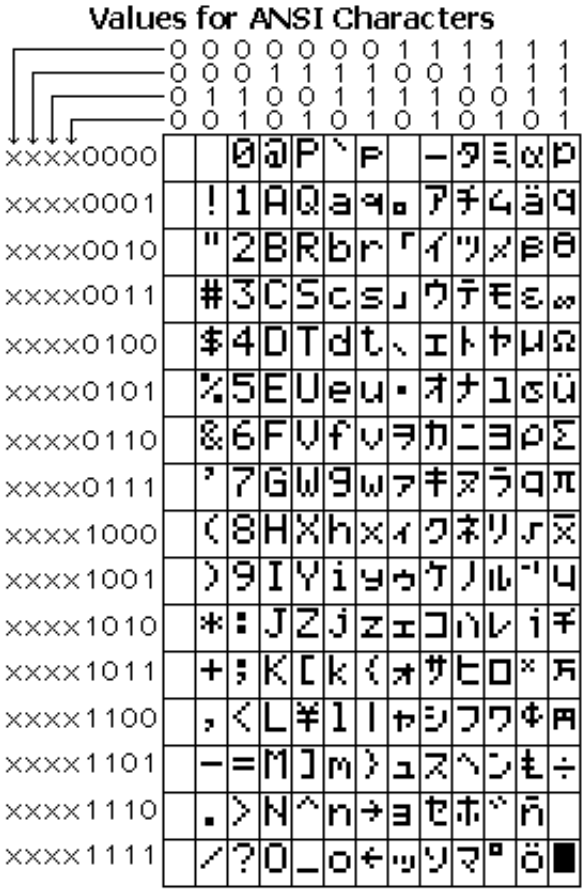

Figure 7. ANSI characters that can be displayed on the LCD.

The following code example cycles through all of the ANSI characters. Note that the conditions in your for loop change somewhat from the previous example, so that the value of x starts at 32 and goes up to 255 before stopping.

1. Rename your code file lcdcharacter.c.
2. Modify your code file, as follows:

```
#include <mxapi.h>
#include <lcd.h>

int main(void)
{
    int x;                                  //Declare a variable called x
    lcd_init();                             //Initialize the LCD
    for(x=32; x<256; x++)                   //Set up a for loop
    {
       lcd_character(FIRST_LINE, x);        //Display ANSI character for x
       lcd_decimal(SECOND_LINE, x, 3);      //Display value of x on LCD
       delay_ms(200);                       //Wait for 200 milliseconds
```

```
    }
    while(1==1);                          // End the program
}
```

3. Compile and test your new code.