

How to Program a Liquid Crystal Display

Temika Cage
April 2, 2010

Executive Summary

Liquid crystal displays are very important devices in embedded systems. They offer high flexibility to users as they can display data. Due to the lack of a proper approach to LCD programming, many individuals fail when trying to complete this task. LCD programming is a complex job but can be simple when a logical approach is taken. This application note is to help enthusiasts who want to program LCDs with thorough understanding.

Introduction

A liquid crystal display (LCD) is a thin and flat electronic visual display. They are very common in small and large electronics due to their many capabilities. LCDs are chosen for their portability, low cost, light weight, and small size. There are two types of LCDs, passive and active. The passive technology is used for devices with small readouts like calculators and wristwatches. They are less costly because they require less hardware than the active technology which allows for easier manufacturing. Active technology LCDs are more common and are seen in most desktop and computer monitors. They are sharper, faster, brighter, and have more contrast. In this application note, an LCD with passive technology is discussed.

Segmented Passive Displays



7-Segment
Digit



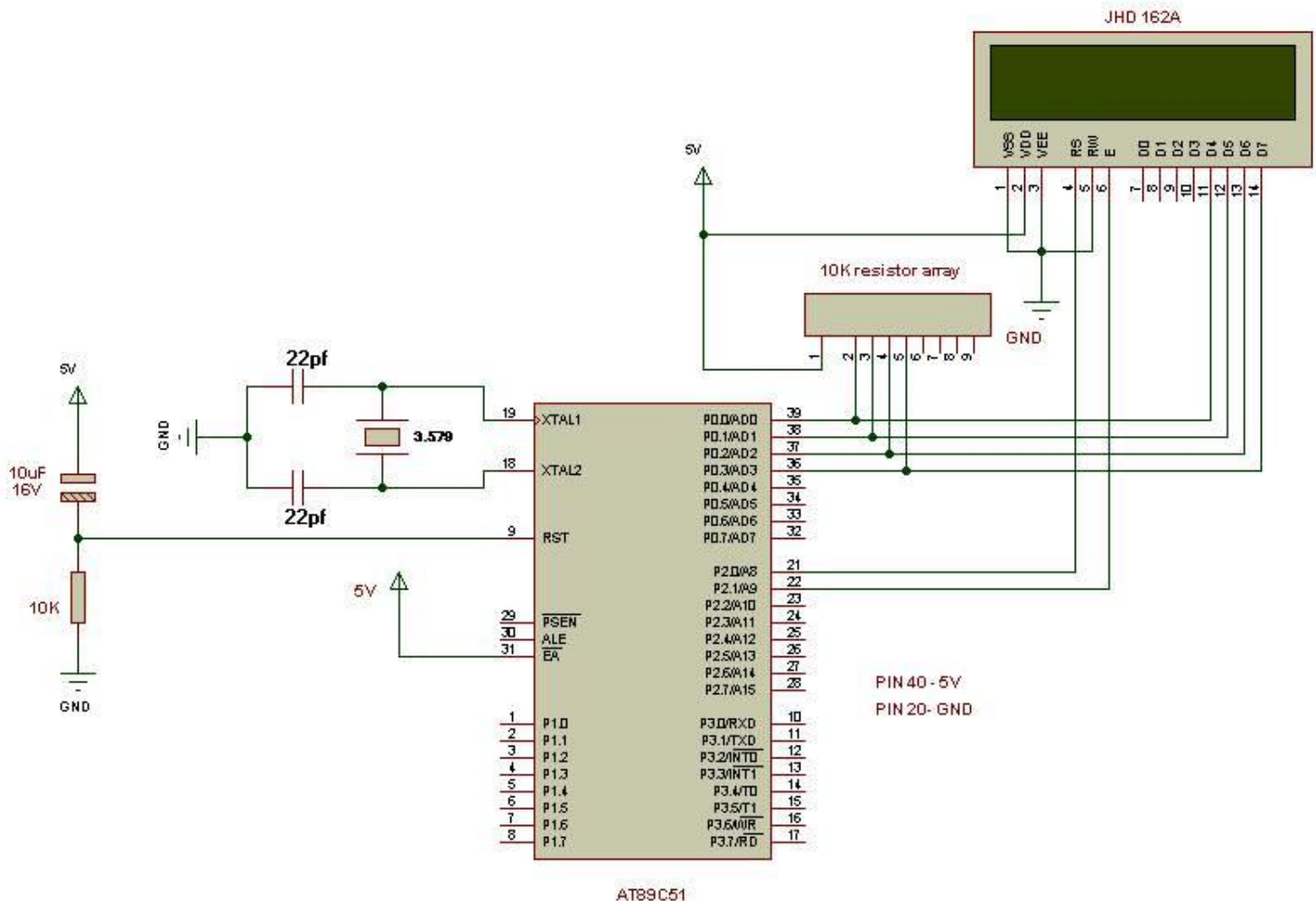
14 or 16-Segment
Character

Steps/Issues

When choosing an LCD, you must first consider many things. LCDs have various specifications like size, screen size, operating temperature, and resolution. The application for which the LCD will be used must be taken into consideration first. Size is important because as size increase, weight increases. Screen size is vital if the potential user is unable to view the display in various conditions (i.e. low light or visually impaired). The typical weather condition for the display has to be determined because LCDs have strict operating temperature ranges. Resolution is important mostly with active technology, not passive.

In order for an LCD to communicate with a microcontroller, there needs to be a driver. The LCD does not automatically know how to communicate with the

microcontroller. You have to program the LCD based on the driver specifications. To understand the algorithms of both components, the respective datasheets must be reviewed.



When analyzing the datasheets, several things should be recognized in order for proper operation. Type of interface, operating voltage, maximum interface speed, display data size, number of pixels, and bits per pixel are important factors. Also, knowing which pins on the LCD are used for communication is vital.

The three control lines are referred to as **EN**, **RS**, and **RW**.

The **EN** line is called "Enable." This control line is used to tell the LCD that you are sending it data. To send data to the LCD, your program should make sure this line is low (0) and then set the other two control lines and/or put data on the data bus. When the other lines are completely ready, bring **EN** high (1) and wait for the minimum amount of time required by the LCD datasheet (this varies from LCD to LCD), and end by bringing it low (0) again.

The **RS** line is the "Register Select" line. When RS is low (0), the data is to be treated as a command or special instruction (such as clear screen, position cursor, etc.). When RS is high (1), the data being sent is text data which could be displayed on the screen. For example, to display the letter "T" on the screen you would set RS high.

The **RW** line is the "Read/Write" control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying (or reading) the LCD. Only one instruction ("Get LCD status") is a read command. All others are write commands--so RW will almost always be low.

Finally, the data bus consists of 4 or 8 lines (depending on the mode of operation selected by the user). In the case of an 8-bit data bus, the lines are referred to as DB0, DB1, DB2, DB3, DB4, DB5, DB6, and DB7.

The first major task in LCD programming is initialization. During initialization, command bytes must be sent to the LCD. The interface mode, display mode, address counter increment direction, contrast of LCD, horizontal or vertical addressing mode, color format can be set. This sequence is given in respective LCD driver datasheets. Studying the function set of an LCD gives the definition of command bytes. It varies from one LCD to another. After proper initialization of the LCD, the process is almost completely done.

Example:

```
CLR RS
MOV DATA,#38h
SETB EN
CLR EN
LCALL WAIT_LCD
```

The next step after initialization is to send data bytes to the required display data RAM memory location. Set the address location using address set command bytes and then send data bytes using the DDRAM write command. To address a specific location in display data RAM, the knowledge of how to increment the address counter is needed. Example:

```
CLR RS
MOV DATA,#0Eh
SETB EN
CLR EN
LCALL WAIT_LCD
```

References

<http://www.yourdictionary.com/computer/lcd-types>
<http://www.lcdinterfacing.info/Interface-LCD-to-8051.php>
<http://www.8052.com/tutlcd>