PICDEM-3

DEMO BOARD USER'S GUIDE

PICDEM-3
User's Guide

Table of Contents

Appendix A. PICDEM-3 Schematics

Appendix B. Optional Hardware Demultiplexer

Appendix C. LCD Manufacturers Listing

Appendix D. RS-232 Communication Protocol

Appendix E. On Line Support

Source Code Listing:

List of Figures:

List of Examples:

List of Tables:

Preface

Welcome

Thank you for purchasing the PICDEM-3 product demonstration board for the PIC16C9XX family of microcontrollers from Microchip Technology Incorporated. The PICDEM-3 allows you to quickly and easily become familiar with both the PIC16C9XX products and the PICMASTER® Universal In-Circuit Emulator.

The software provided with the PICDEM-3 for the LCD Software Demultiplexer runs under Microsoft Windows® 3.1 or later.

Documentation Layout

This document describes the PICDEM-3 demonstration board. A detailed description of the demonstration software is provided to give the user an overview of the PIC16C9XX series of Microchip microcontrollers. Detailed usage of the microcontrollers, MPLAB™, or PICMASTER are deferred to the individual product data sheets and User's Manuals, respectively.

Chapter 1: Introduction - This chapter introduces the PICDEM-3 and provides a brief description of the hardware.

Chapter 2: Using the LCD Software Demultiplexer - This chapter discusses how to run the LCD Software Demultiplexer. Also provided is the optional hardware demultiplexer information.

Chapter 3: LCD Fundamentals - This chapter provides a detailed description of how LCD panels operate and microcontrollers drive those panels. This chapter is taken from Application Note *AN658: LCD Fundamentals Using PIC16C92X Microcontrollers*.

Chapter 4: `analog.asm` Description - This chapter provides a detailed description of the demonstration program for the PIC16C924 that reads the potentiometer and thermistor and displays the results on the LCD panel.

Chapter 5: `keypad.asm` Description - This chapter provides a detailed description of the demonstration program for the PIC16C92X. This program reads the keypad and displays the key value on the LCD panel.

Chapter 6: `uart.asm` Description - This chapter provides a detailed description of the demonstration program for the PIC16C92X that transmits and receives data via an RS-232 interface.

Chapter 7: `test.c` Description - This chapter provides a detailed description of a test program for the PIC16C92X. It is used to test the interface between the PICDEM-3 board and the LCD Software Demultiplexer.

Chapter 8: Hardware Description - This chapter describes in detail the hardware of the PICDEM-3.

Appendix A: PICDEM-3 Schematics - This appendix provides the PICDEM-3 parts layout diagram and the board schematic.

Appendix B: Optional Hardware Demultiplexer - This appendix provides the schematic for the optional hardware demultiplexer.

Appendix C: LCD Manufacturers Listing - This appendix provides a listing of companies that manufacture or resell liquid crystal displays.

Appendix D: RS-232 Communication Protocol - This appendix provides the communication protocol for the PIC16C73 to Host PC interface.

Appendix E: On-line Support - This appendix provides information on Microchip's electronic support services.

Worldwide Sales & Service: This reference gives the address, telephone and fax number for Microchip Technology Inc. sales and service locations throughout the world.

Chapter 1. About PICDEM-3

Introduction

This chapter describes the features of the PICDEM-3 demonstration board.

Highlights

This chapter covers the following topics:

- **Processor Sockets**
- **Liquid Crystal Display (LCD)**
- **Power Supply**
- **RS-232 Serial Port**
- **Pushbutton Switches**
- **Oscillator Options**
- **Analog Inputs**
- **Keypad Header**
- **External LCD Panel Connector**
- **LCD Software Demultiplexer**
- **Pre-programmed Sample**

Note: All following part references can be found in Figure A.1 in *Appendix A: PICDEM-3 Schematics*. For example, the 68-pin PLCC socket for PIC16C92X microcontrollers is located at **U1** on the Parts Layout on page 89.

Processor Sockets

The PICDEM-3 supports the following devices:

- 68-pin PLCC socket for PIC16C92X microcontrollers (U1)
- 44-pin PLCC footprint for future products (U9)

Liquid Crystal Display

An LCD Panel is provided on PICDEM-3. It is a four-common by twelve-segment LCD panel capable of displaying time, AM/PM, day of the week, and temperature. Chapter 8 provides a detailed description of this panel.

Power Supply

There are three ways to supply power to PICDEM-3:

- A 9V battery can be plugged into BT1.

- A +9V, 150 mA unregulated AC or DC supply can be plugged into J1.
- A +5V, 150 mA regulated DC supply can be connected to the hooks provided in the prototyping area.

PICMASTER users should note that a regulated +5V power supply is available in the logic probe connector, and can easily be connected to the hooks on PICDEM-3 (Red probe to +5V, and Black probe to GND). Note the following power-up sequence:

1. Power-up the PC.
2. Start MS Windows 3.1 or greater.
3. Power-up the PICMASTER.
4. If an external wall mounted power supply is being used, connect it now and turn on the power.
5. Start MPLAB.

RS-232 Serial Port

An RS-232 level-shifting IC has been provided with all the necessary hardware to support connection of an RS-232 host through the DB9 connector P2. This port is labeled PIC16C9XX PORT on the PICDEM-3. The port is configured as DCE, and can be connected to a PC using a straight through cable. Note that the PIC16C92X devices do not currently have a hardware serial port. If you wish to use a serial port, refer to the sample program in Chapter 6 for a software implementation.

Pushbutton Switches

Five pushbutton switches have been provided on the PICDEM-3 with the following functions:

- $\overline{\text{MCLR}}$ to hard reset the PIC16C9XX device (S1)
- $\overline{\text{MCLR}}$ to hard reset the LCD software demultiplexer (S6)
- CCP1 for capture of timer value (S2)
- Active low switch connected to RA2 (S3)
- Active low switch connected to RA5 (S4)

Oscillator Options

The following oscillator options can be found on the PICDEM-3:

- 685 kHz (approximately) RC oscillator supplied. This oscillator may be disabled by removing jumper JP6. Note that the capacitor C27 (33 pF) is still installed.
- Pads provided for user supplied crystal and capacitors. These pads may also be used with ceramic resonators, including ones with integral capacitors. (Y1)
- Socket provided for clock oscillator. (O1)

Analog Inputs

A 5k potentiometer (R2) is connected through a series 470 resistor (to protect the part should the pin be configured as an output) to RA0/AN0. The potentiometer can be adjusted from V_{DD} to GND to provide an analog input to the PIC16C9XX devices with an A/D convertor. A 4.6k thermistor (RT1) is connected to pin RA1/AN1 to provide temperature measurement.

Keypad Header

A separate header (JP1) is provided to connect a keypad (such as a C&K 4B01H322PCFQ available from Newark Electronics) to the PIC16C9XX through PORTB. The example program in Chapter 5 shows how to use this 4x4 keypad.

External LCD Panel Connector

The connector, JP2, is provided to connect an external LCD panel to the PIC16C9XX device. This panel contains all the LCD driver signals from the PIC16C9XX, including commons and segments. *Appendix A: PICDEM-3 Schematics* shows the pinout for this connector.

LCD Software Demultiplexer

The LCD signals generated by the PIC16C9XX devices can be monitored via the DB9 connector, P1. The PIC16C73, in conjunction with the LM358s and CD5051s, reads the LCD signals and demultiplexes them into individual pixels. Chapter 2 provides a detailed explanation of this interface.

Pre-programmed Sample

A pre-programmed PIC16C924 sample is included with the PICDEM-3. It has the example program shown in Chapter 7. This device has been programmed with the XT oscillator and requires a 4 MHz crystal, resonator, or clock oscillator. Since it also has the production test program you might see some strange messages displayed on the LCD panel. If this occurs just hit the $\overline{\text{MCLR}}$ pushbutton switch, S1.

Chapter 2. Using the LCD Software Demultiplexer

Introduction

This chapter discusses how to use the LCD Software Demultiplexer included with the PICDEM-3.

Highlights

This chapter includes:

- **Installing the LCD Software Demultiplexer**
- **Using the LCD Software Demultiplexer**
- **Optional Hardware Demultiplexer**

Installing the LCD Software Demultiplexer

The PICDEM-3 setup routine installs the LCD software demultiplexer and the sample programs in the MPLAB directory. Microsoft Windows must be running to execute the setup routine. Setup will copy all files to run the LCD software demultiplexer in the specified directory. It will also create two subdirectories; examples and demux. The examples subdirectory will have the four example programs. The demux directory will have the source code files for the PIC16C73. The following procedure should be followed:

1. Insert the PICDEM-3 installation disk in drive A.
2. From the Program Manager Run option, type **A:Setup**. The PICDEM-3 Setup program displays a Welcome! message box with options to continue or exit. Click **OK** to continue.
3. Setup next displays a dialog box to select the MPLAB directory. Enter the directory name to install the PICDEM-3 files to and click **OK**. It is recommended that these files be installed in the same directory as MPLAB.
4. After copying the PICDEM-3 files, Setup displays a message box asking if you want to add the PICDEM-3 icon to the Program Manager. Click **Yes**.
5. Setup displays a dialog that allows you to choose the Program Manager Group you want to install the PICDEM-3 icons into. Setup displays Microchip MPLAB by default. Choose the group to add the icons to and click **OK**.
6. When complete, Setup displays a message box with the caption "The PICDEM-3 Installation has completed." Click O**K**.

Using the LCD Software Demultiplexer

The LCD Software Demultiplexer is a debugging tool that displays the status of each pixel. The pixel status is determined by the PIC16C73 on the PICDEM-3. The LCD signals are MUXed onto the PIC16C73 A/D convertor using LM358s and CD4051s. Using the LCD common signals, the PIC16C73 is able to demultiplex the LCD signals.

The following steps should be followed to start the LCD Software Demultiplexer:

1. Connect a straight through DB9 cable to port P1 on the PICDEM-3 board. This port is labelled LCD DRIVER DECODE.
2. Apply power to the PICDEM-3 board.
3. Using S5, select the correct MUX that the PIC16C9XX is using.
4. Press S6 (labelled MCLR_U6) to reset the PIC16C73.
5. Start the LCD Software Demultiplexer.
6. If the Error Box appears, click on **OK** and then select *File > Comm Port Select*. Choose the COM port that is connected to the PICDEM-3 board.

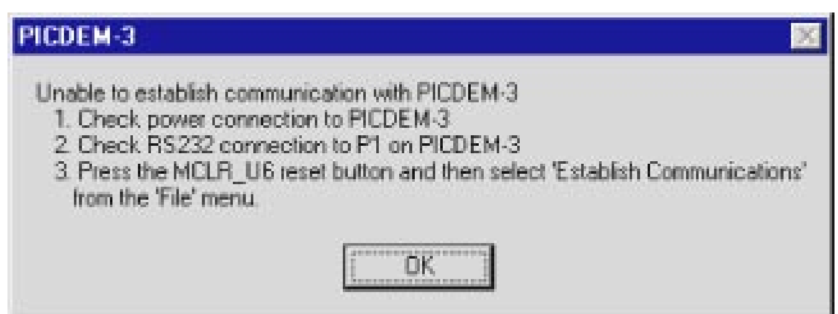


Figure 2.1 Error Box

If the error box continues to appear, check all RS-232 connections between the Host PC and the PICDEM-3 board, and restart the LCD Software Demultiplexer.

Chapter 2. Using the LCD Software Demultiplexer

In the following figures, each box in the window represents a pixel on the LCD panel. A shaded box indicates that the pixel is visible. If the LCD Software Demultiplexer is not functioning properly, refer to the test program in Chapter 7.

The following figures show the LCD Software Demultiplexer in all the four MUX modes.

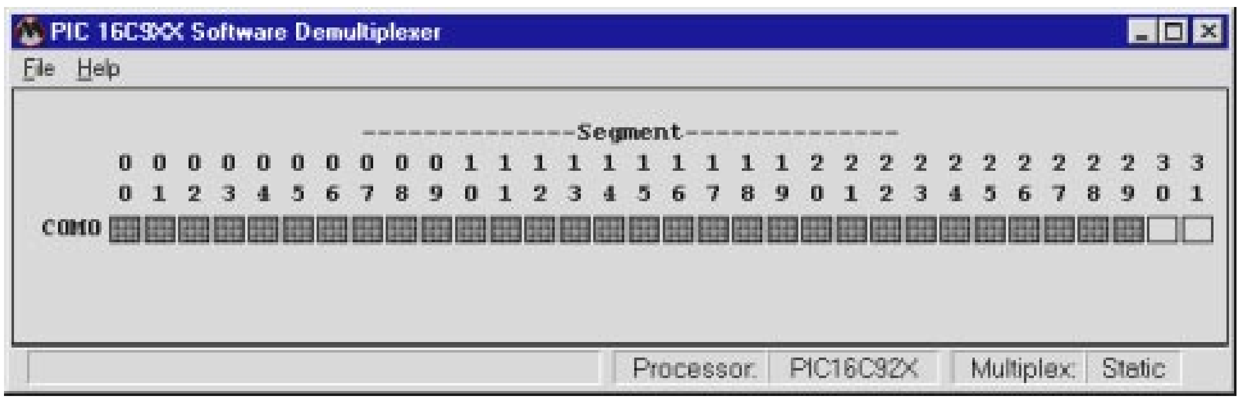


Figure 2.2 Static MUX

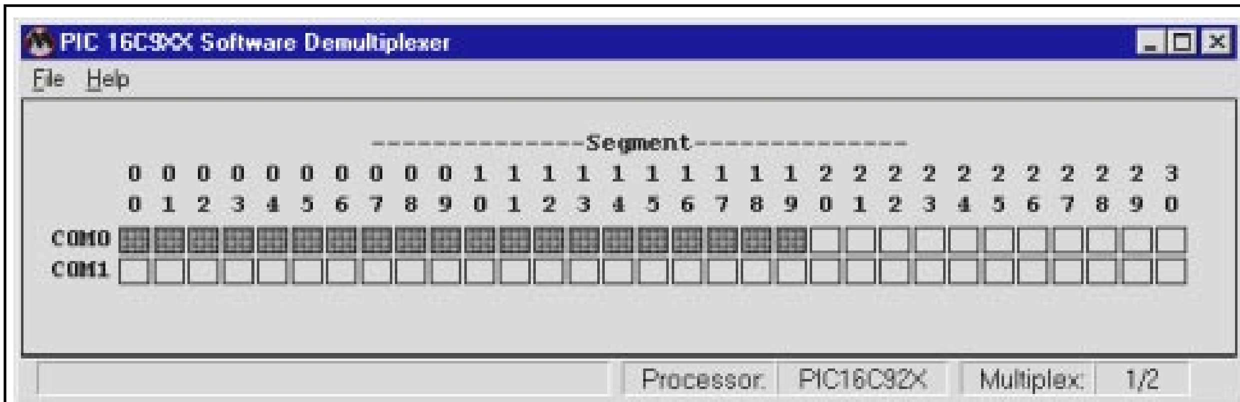


Figure 2.3 1/2 MUX

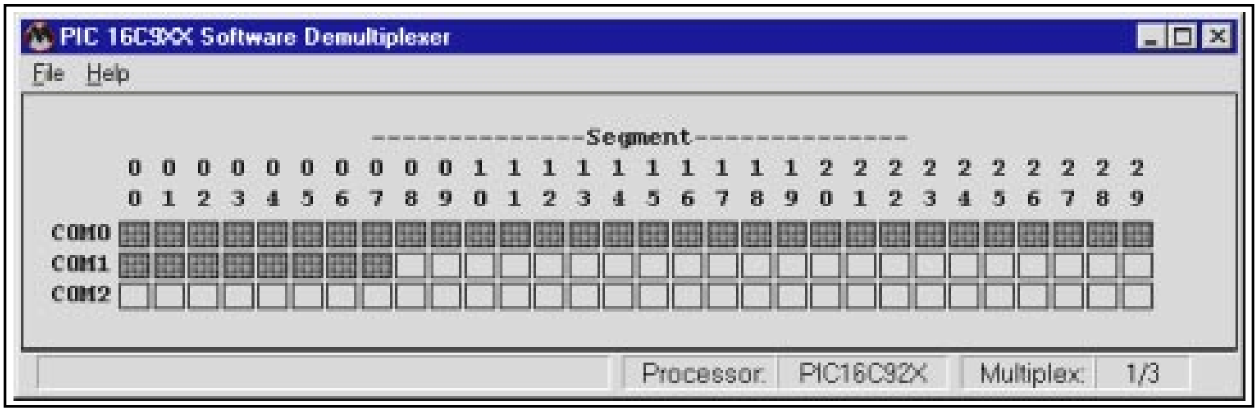


Figure 2.4 1/3 MUX

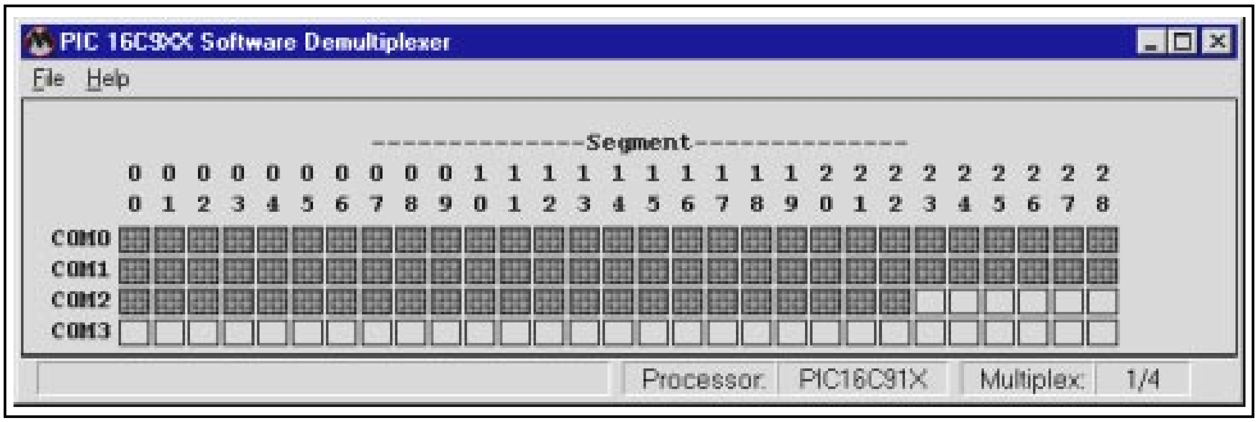


Figure 2.5 1/4 MUX

Optional Hardware Demultiplexer

PICDEM-3 has the capability to connect a hardware demultiplexer display board via connector JP4. JP4 is connected to the SPI (Serial Peripheral Interface) port on the PIC16C73. The data bytes are transmitted out from this interface to a hardware display board. *Appendix Appendix B.: Optional Hardware Demultiplexer* shows the schematic for this board. Instead of using the software demultipexer, you can arrange LEDs as if they were pixels on your LCD panel. These LEDs are then connected to the display board to provide a simulation of what the actual panel would look like.

Chapter 3. LCD Fundamentals

Introduction

This chapter provides a tutorial on LCD fundamentals.

Highlights

- **What are the basic components in an LCD panel?**
- **How does an LCD work?**
- **LCD Images**
- **Driver Voltages**
- **Response Time**
- **Temperature Effects**
- **Capacitance**
- **Backlighting**
- **Connection Methods**
- **What are the different types of LCD panels?**
- **How are LCD panels driven?**
- **Discrimination Ratio**
- **LCD Voltage Generation**
- **Contrast**

What are the basic components in an LCD panel?

An LCD panel, or more commonly known as a piece of “glass,” is constructed of many layers. Figure 3.1 shows all the layers that are typically present in LCD panels. The first layer is called the **front polarizer**.

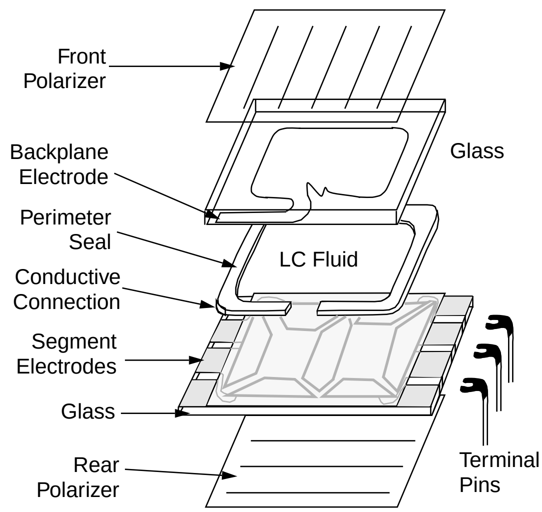


Figure 3.1 Basic LCD Components

Polarization is a process or state in which rays of light exhibit different properties in different directions, especially the state in which all the vibration takes place in one plane. Essentially, a polarizer passes light only in one plane. As shown in Figure 3.2, if light is polarized in one plane, by passing through a polarizer, it cannot pass through a second polarizer if its plane is 90° out of phase to the first.

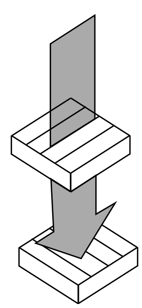


Figure 3.2 Polarizers out of Phase

The front polarizer is applied to the outside surface of the top piece of **glass**. The top piece of glass also provides structural support for the LCD panel.

On the bottom of the top glass, a transparent coating of Indium-Tin Oxide (ITO) is applied to the glass. ITO is conductive and forms the **backplane** or common electrodes of the LCD panel. The patterns of the backplane and segment ITO forms the numbers, letters, symbols, icons, etc.

After the ITO has been applied to the glass, a thin polyimide coating is applied to the ITO. The polyimide is "rubbed" in a single direction that matches the polarization plane of the front polarizer. The action of "rubbing" the polyimide causes the Liquid Crystal (LC) molecules in the outermost plane to align themselves in the same direction.

The next layer is a reservoir of LC. The **LC fluid** has many planes of molecules.

The next layer is the polyimide coating on the bottom glass followed by the ITO **segment electrodes**. The bottom glass also supplies structural integrity for the LCD panel as well as mounting surface for the electrode connections. Applied to the external surface of the bottom glass is the rear polarizer. Depending on the type of viewing mode employed by the LCD panel, the axis of polarization is the same or 90° apart from the front polarizer.

LC molecules are long and cylindrical. On any plane within the LC fluid, the molecules align themselves such that the major axis of each molecule is parallel to all others, as shown in Figure 3.3. The outermost planes of LC molecules will align themselves on the same axis that the polyimide is “rubbed”. The direction of “rubbing” of the polyimide on the bottom glass is 90° apart from that of the polyimide on the top glass. This orientation creates the twist in the LC fluid.

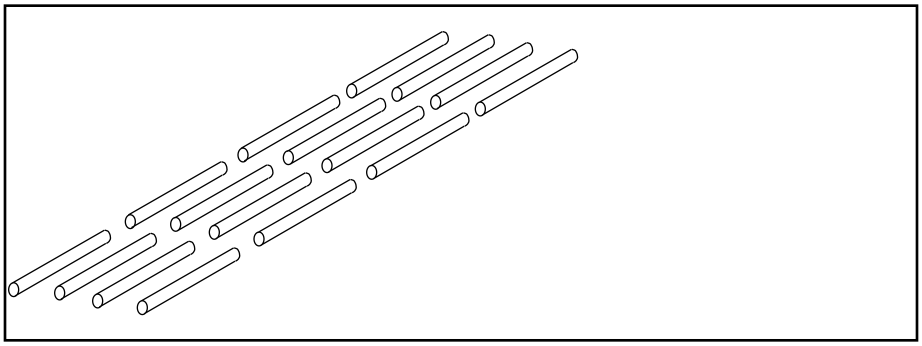


Figure 3.3 LC Molecules in Alignment

A consequence of this alignment is that each intermediate plane of LC molecules will have a slightly different orientation from the plane above or below as seen in Figure 3.4.

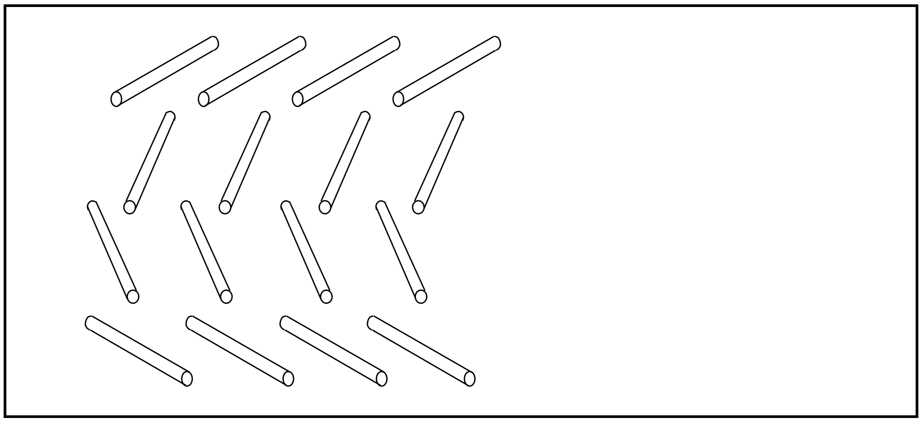


Figure 3.4 LC Molecules Plane Orientation

The twisting of the planes causes the polarization of the light to twist as it passes through the LC fluid. The twisting of the LC planes is critical to the operation of the LCD panel as will be shown in the next section.

Now that the mystery of what the LCD panel is made of has been uncovered, how does an LCD work?

How does an LCD work?

As explained before, the twist created in the LC fluid is the basis of how the panel operates. Figure 3.5 shows how an LCD panel creates a pixel that is OFF. For this example the LC fluid is not energized, i.e. there is 0 V_{RMS} potential between the backplane and segment electrodes. The following is a step-by-step description of the path light takes through the LCD panel.

1. Light enters the panel through the rear polarizer. At this point the light becomes polarized to the vertical plane.
2. The polarized light passes unobstructed through the transparent backplane electrode.
3. As the polarized light passes through the LC fluid it gets twisted into the horizontal plane.
4. The polarized light passes unobstructed through the transparent segment electrode.
5. Since the light is now polarized in the horizontal plane, it passes unobstructed through the front polarizer which has a horizontal polarization.
6. The observer does not detect that the pixel is on because the light has not been obstructed.

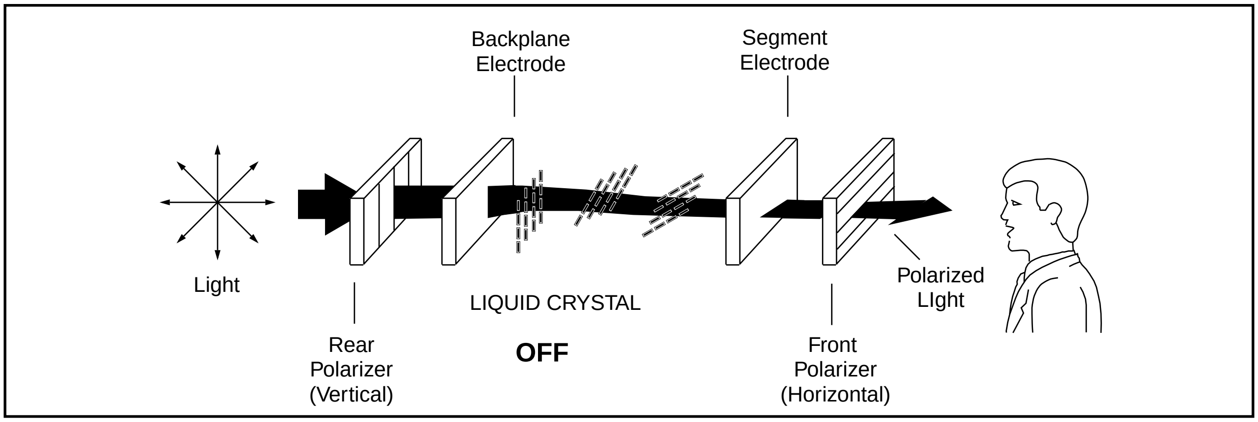


Figure 3.5 Path of Light for Off Pixel

If a potential is applied across the backplane and segment electrodes, the LC fluid becomes energized. The LC molecule planes will now align themselves such that they are parallel to the electrical field generated by the potential difference. This removes the twisting effect of the LC fluid. Figure 3.6 shows a pixel that is ON or more specifically energized. The following is a step-by-step description of the path that the light takes through this LCD panel.

1. Light enters the panel through the rear polarizer. At this point the light becomes polarized to the vertical plane.
2. The polarized light passes unobstructed through the transparent backplane electrode.
3. As the polarized light passes through the LC fluid it does not twist and remains in the vertical plane.
4. The polarized light passes unobstructed through the transparent segment electrode.
5. Since the light is still polarized in the vertical plane, it is obstructed by the front polarizer which has a horizontal polarization.
6. The observer detects that the pixel is on because the light has been obstructed and creates a dark image on the panel.

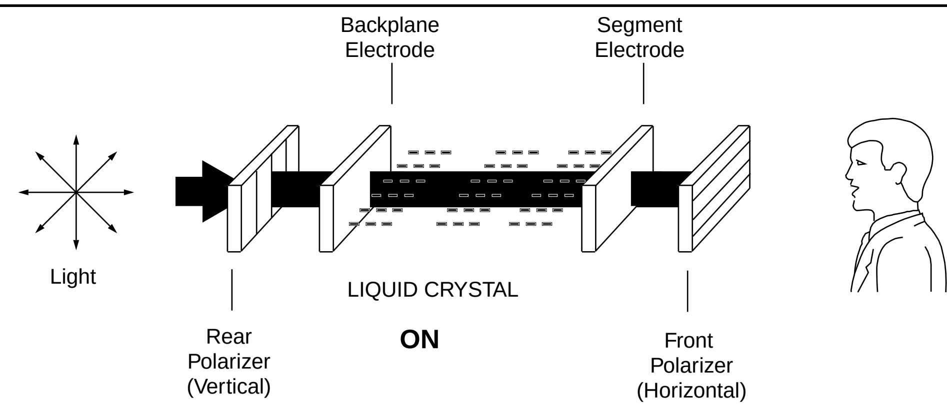


Figure 3.6 Path of Light for On Pixel

LCD Images

LCDs have the capability to produce both positive and negative images. A **positive image** is defined to be a dark image on a light background. In a positive image display, the front and rear polarizers are perpendicular to each other. Unenergized pixels and the background area transmit the light and energized pixels obstruct the light creating dark images on the light background. A **negative image** is a light image on a dark background. In this type of display, the front and rear polarizers are aligned to each other. Unenergized pixels and the background inhibit light from passing through the display. Energized pixels allow the light to pass creating a light image on a dark background.

There are essentially three types of viewing modes for a LCD: reflective, transmissive, and transflective. **Reflective displays** use only positive images. The front and rear polarizers are perpendicular to each other. The LCD panel will have an additional layer added to the bottom of the display, a reflector. Figure 3.7 shows the diagrams for pixels that are ON and OFF for reflective displays. Here again, the path that light takes is described in a step-by-step fashion for a pixel that is OFF in a positive image display.

1. Light enters the panel through the front polarizer. At this point the light becomes polarized to the vertical plane.
2. The polarized light passes unobstructed through the transparent backplane electrode.
3. As the polarized light passes through the LC fluid it gets twisted into the horizontal plane.
4. The polarized light passes unobstructed through the transparent segment electrode.
5. Since the light is now polarized in the horizontal plane, it passes unobstructed through the rear polarizer which has a horizontal polarization.
6. The reflector behind the rear polarizer reflects the incoming light back on the same path.
7. The observer does not detect that the pixel is ON because the light was reflected back.

A pixel that is ON follows the same basic steps except that the light never reaches the reflector and therefore does not return to the observer. Reflective displays lend themselves to battery powered applications because the images are created using ambient light sources. These displays are very bright, with excellent contrast, and have a wide viewing angle.

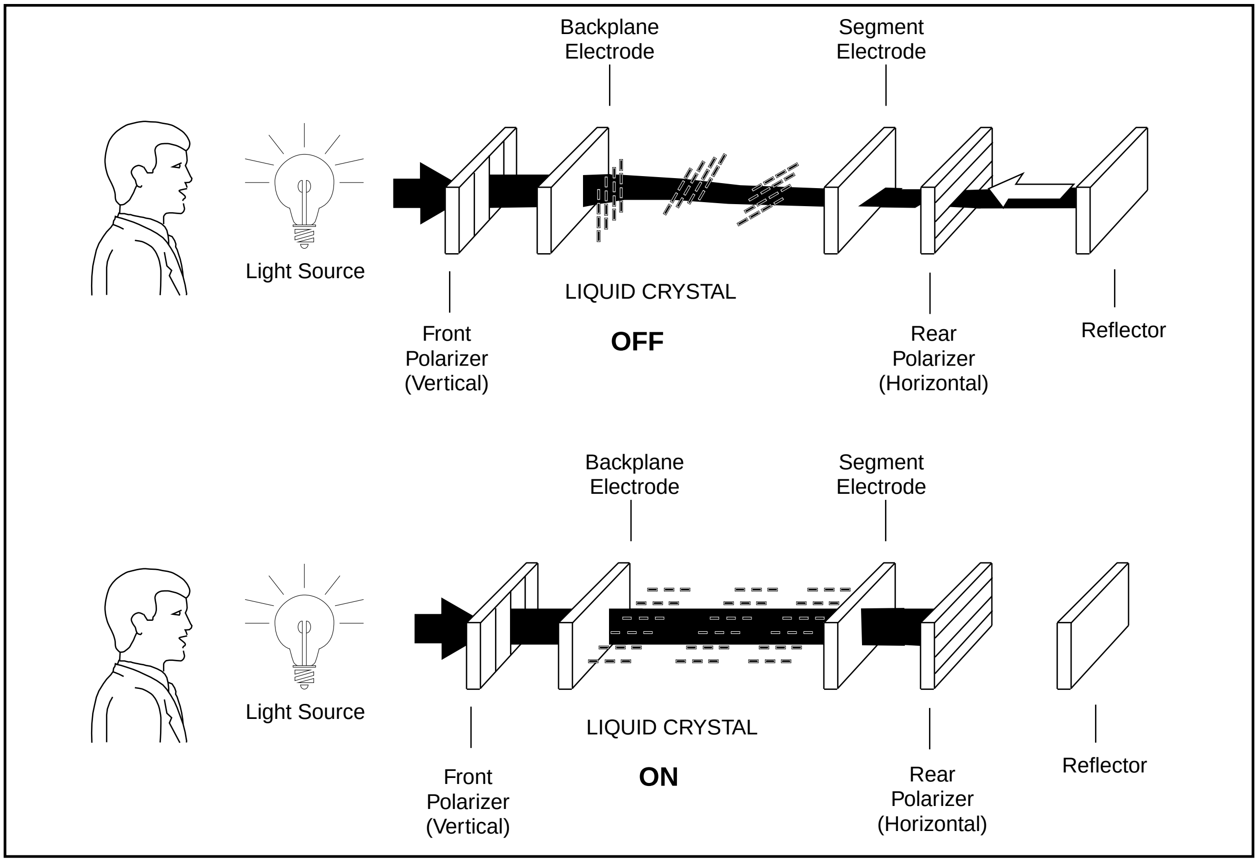


Figure 3.7 Reflective LCD Path of Light

Transmissive displays do not reflect light back to the observer. Instead, they rely upon a light source behind the panel to create images. A transmissive display has front and rear polarizers that are in phase to each other. Figure 3.8 shows the OFF and ON diagrams for a transmissive display. The path of light is described below for the ON state only in a positive image display.

1. Light enters the panel through the rear polarizer. At this point the light becomes polarized to the vertical plane.
2. The polarized light passes unobstructed through the transparent segment electrode.
3. As the polarized light passes through the LC fluid it gets twisted into the horizontal plane.
4. The polarized light passes unobstructed through the transparent backplane electrode.
5. Since the light is now polarized in the horizontal plane, it is obstructed by the front polarizer which has a vertical polarization. Very little light passes through the front polarizer

6. The observer does not detect that the pixel is ON because the light was obstructed.

An OFF pixel would allow the light to pass through the display unobstructed because the polarization does not get twisted by the LC fluid. These displays are very good for very low light level conditions. They are very poor when used in direct sunlight because the sunlight swamps out the backlighting.

The third type of display is called **transflective**. As you can probably tell from the name, it is a combination of reflective and transmissive. A white or silver translucent material is applied to the rear of the display. It reflects some of the ambient light back to the observer while also allowing backlighting. Transflective displays are very good for applications which have varying light conditions such as gas pumps. They must operate during the day in bright sunlight, but must also operate at night. Transflective displays have lower contrast ratios than reflective displays because some of the light passes through the reflector.

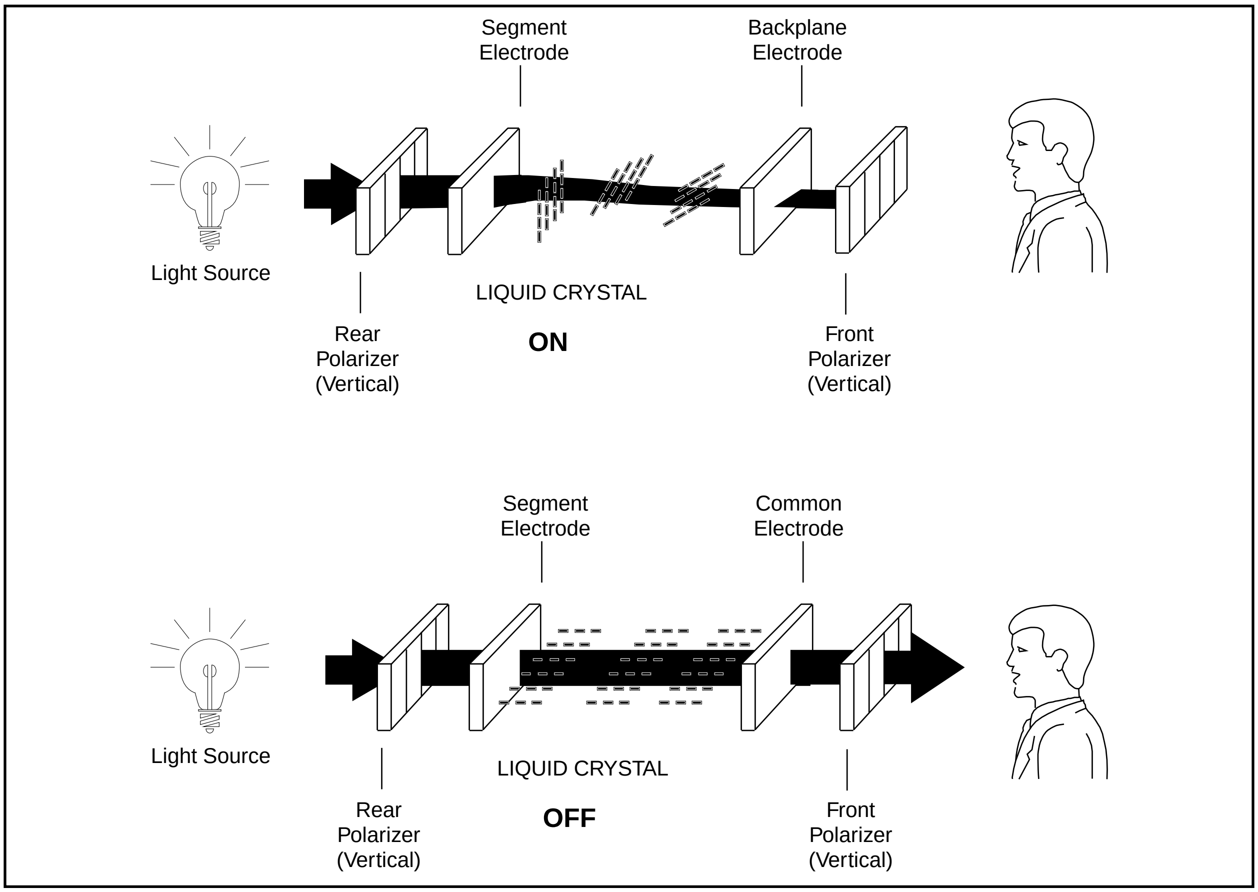


Figure 3.8 Transmissive LCD Path of LIght

The type of LCD that an application requires is largely dependent on the ambient light available. Table 3.1 gives some guidelines for selecting a display according to the lighting conditions.

Table 3.1 Lighting Condition Reference

Viewing Mode	Display Description	Application Comments	Direct Sunlight	Office Light	Very Low Light
Reflective (Positive)	Dark images on light background	No backlighting. Gives best contrast and environmental stability	Excellent	Very Good	Unusable
Transflective (Positive)	Dark images on gray background	Can be viewed with both ambient light and backlighting	Excellent (no backlight)	Good (no backlight)	Very Good (backlight)
Transflective (Negative)	Light gray images on dark background	Requires high ambient light or backlighting.	Good (no backlight)	Fair (no backlight)	Very Good (backlight)
Transmissive (Negative)	Backlit images on dark background	Cannot be viewed by reflection	Poor (backlight)	Good (backlight)	Excellent (backlight)
Transmissive (Positive)	Dark images on a backlit background	Good for very low light conditions, but readable in bright ambient light	Good (no backlight)	Good (backlight)	Excellent (backlight)

Driver Voltages

The number one cause of LCD damage is having a DC voltage applied to it. A DC voltage will deteriorate the LC fluid such that it cannot be energized. The LCD driver waveforms are designed to create a 0 VDC potential across all pixels. The specifications for a LCD panel will include some RMS voltages such as **VOFF** and **VON**. A third voltage is **VTH** which is the RMS voltage across an LCD pixel when contrast reaches a 10% level. Often this voltage is used as VOFF. VON is defined as the RMS voltage applied by the LCD driver to the segment electrode that creates a ON pixel which is typically at the 90% contrast level. It is desirable that VON be much greater than VOFF.

Figure 3.9 graphically represents the voltage potential versus the contrast across a pixel. The final specification for an LCD panel is the discrimination ratio which is VON divided by VOFF (VON/VOFF). The **discrimination ratio** specifies what type of contrast levels the LCD panel will be able to achieve. Examples of discrimination ratio calculations will be given in the section “How are LCD Panels Driven?”.

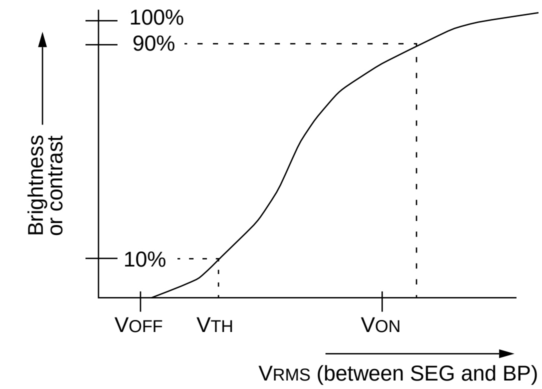


Figure 3.9 Contrast vs. RMs Voltage

Response Time

An LCD panel will have a typical ON and OFF response time. The **ON time** parameter refers to the time for an OFF pixel to become visible after the appropriate voltages have been applied. The **OFF time** parameter specifies the time for an ON segment to disappear. Sometimes these parameters are called rise and decay, respectively. Temperature plays a key role in the response time of an LCD panel. Figure 3.10 shows the response times versus temperature for commercial type LC fluid. For this reason, there are not any LCD panels in gas pumps in Alaska without heaters. Displays with heaters can help to maintain 0°C response time even a temperatures as low as -55°C. The drawback of an LCD heater is that every square inch of surface on the back of the display requires 2 to 3 watts.

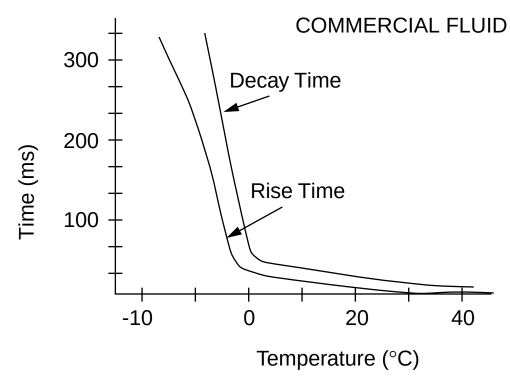


Figure 3.10 Response vs. Temperature

Temperature Effects

As previously shown, temperature has a large impact on the performance of the LCD panel. Not only is the LC fluid affected, but the internal coatings begin to deteriorate. All LC fluids have well defined operating temperature limits. If an LCD is operated above its fluid limits, the LC molecules begin to assume random orientations. The pixels on a positive image display will become completely dark, while pixels on a negative image display will become completely transparent. An LCD can recover from these conditions if the exposure is kept short, however, temperatures above 110°C will cause the ITO and polyimide coatings to deteriorate.

On the low end of the temperature spectrum, response times increase because the viscosity of the LC fluid increases. At very low temperatures, typically -60°C, the LC fluid transitions into a crystalline state. Usually, the LC fluid can recover from the effects of low temperature. Many different types of LC fluid are available, which allows the LCD panel to be tailored to the expected operating conditions. As mentioned in the previous section, heaters can combat the effects of low temperature.

Capacitance

The LCD panel can be modeled as a lossy, non-linear capacitor. The area of the pixel, and therefore the size of the LCD panel, has a direct impact on the value of the capacitance that a common or segment driver must be able to drive. Typical values of capacitance are in the range of 1000 - 1500 pF/cm^2. Figure 3.11 shows an example of a 1/3 MUX panel. As you can see the backplane driver must be capable of driving significantly higher capacitances than the segment driver.

Care must be taken when designing a system such that your LCD driver is capable of driving the capacitance on the segment and common. Otherwise the LCD panel may be damaged due to a DC offset voltage generated by overloaded segment and common drivers.

PIC16C92X microcontrollers are capable of driving backplanes up to 5000 pF and segments up to 500 pF.

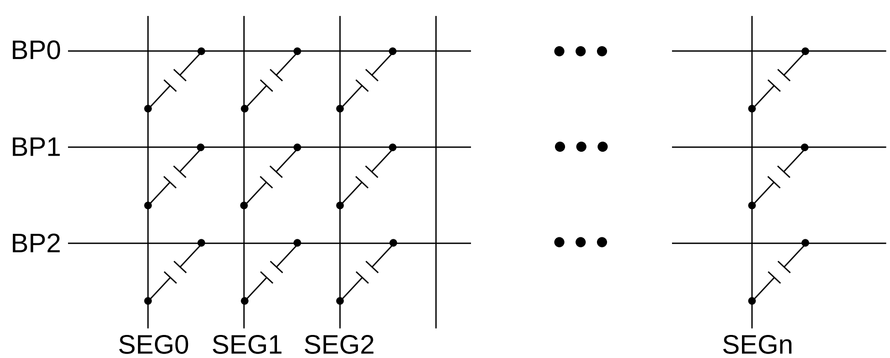


Figure 3.11 1/3 MUX LCD Equivalent Circuit

Backlighting

A variety of methods exist for backlighting LCD panels, such as, incandescent lamps, LEDs, and electroluminescent lamps. Incandescent lamps require some type of reflector to provide uniform lighting to all areas of the panel. LEDs require some type of lightguide or lightpipe to evenly distribute light. Electroluminescent lamps typically come in some type of a panel arrangement. Table 3.2 provides a comparison of these methods of backlighting.

Table 3.2 Backlighting Features Comparison

Feature	LED	Incandescent	Electroluminescent
Brightness	Medium	High	Low-Medium
Color	Limited	White	White
Size	Small	Small-Medium	Thin
Voltage	5V	1.5V - 28V	45V - 100V
Current @5V/sq. in	10-30 mA	20 mA	1 mA - 10 mA
Temperature	Warm	Hot	Cool
Cost/sq. in	$0.10 - $1.00	$0.10 - $0.80	$0.50 - $2.00
Shock Tolerance	Excellent	Fragile	Excellent
Life (hours)	100,000	150 - 10,000	500 - 15,000

Connection Methods

The first method of connecting the LCD panel to the world was the dual-in-line pin shown in Figure 3.12.

These pins provide excellent protection for harsh environments, vibration or shock. The LCD panel is either soldered directly to the printed circuit board (PCB) or inserted into headers.

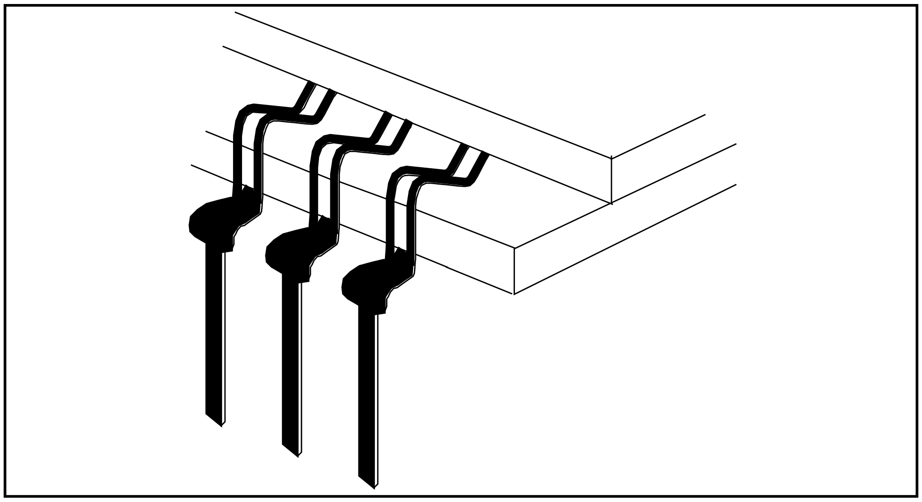


Figure 3.12 Dual In-Line Pins

The second method is elastomeric connectors. This method allows fast assembly/disassembly without having to solder the LCD panel. Elastomeric connectors are used on small applications where space is a concern. These connectors are relatively resistant to shock and vibration, but special consideration must be used when the panel will be exposed to harsh environments. Figure 3.13 shows an assembly drawing of an elastomeric connector.

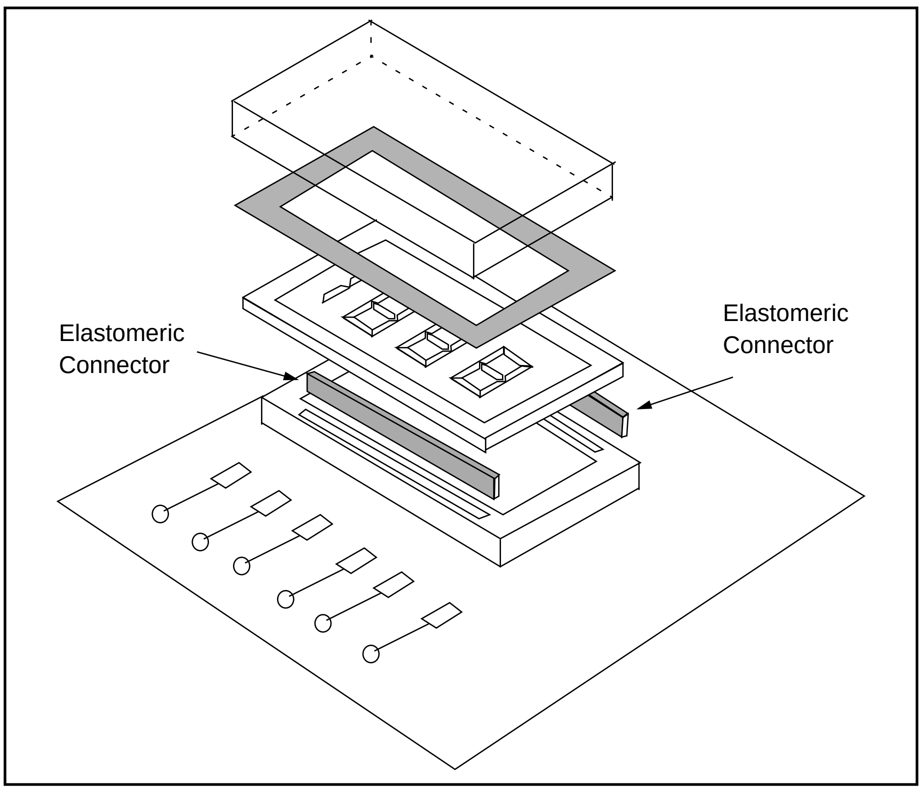


Figure 3.13 Elastomeric Connectors

One of the newer methods is the flex connector. A PCB and the LCD panel are connected by a flexible cable using a heat seal process. The flexible cable is typically a anisotropic connective film that is applied to the PCB and LCD panel using heat and pressure. These connectors were designed for harsh environments where the connector must be flexible enough to prevent breakage during stress. These connectors are becoming more popular with large or remotely mounted LCD panels. Figure 3.14 shows a typical application.

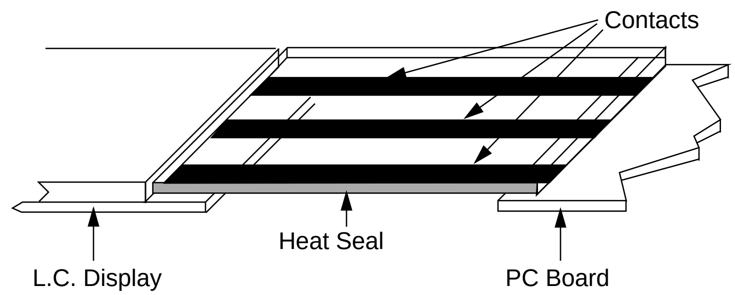


Figure 3.14 Flex Connectors

What are the different types of LCD panels?

LCD panels come in many flavors depending on the application and the operating environment. LCDs can be classified in two ways. First of all, LCDs come in direct drive or multiplex drive variations. Direct drive, otherwise known as static, means that each pixel of the LCD panel has an independent driver. The LCD panel also has only one backplane. A static drive panel also has static bias. Bias is defined as the number of voltage levels the LCD driver uses to create images on the screen. The number of voltage levels is equivalent to the 1 + 1/bias. Static bias refers to two voltage levels which create a square wave, ground and V_{DD}. Static drive panels also have the best contrast ratios over the widest temperature range.

Multiplex drive panels reduce the overall amount of interconnections between the LCD and the driver. Put simply, multiplex panels have more than one backplane. A multiplex LCD driver produces an amplitude-varying, time synchronized waveform for both the segment and backplanes. These waveforms allow access to one pixel on each of the backplanes. This significantly increases the complexity of the driver. The number of backplanes a panel has is referred to the multiplexing ratio or “MUX” of the panel. MUX also refers to duty cycle. For instance, a 1/3 MUX panel has three backplanes. The bias for multiplex panels is at least 1/2 - 1/5 for segment type drivers and from 1/8 - 1/33 for dot matrix. Table 3.3 illustrates the advantage of multiplex panels.

Table 3.3 Static vs. Multiplex Pin Count

LCD panel	Back planes	Segments	Total
3 - 1/2 digit	1	23	24
	2	12	14
8 digits	1	64	65
	4	16	20
2 x 16 character dot matrix, 5 x 7 characters	1	1280	1281
	8	160	168
	16	80	96
128 x 240 graphic display	1	30,720	30721
	64	480	544
	128	240	368

The last time Microchip investigated high pin count packages, 30,000+ was not an option.

PIC16C92X microcontrollers have the following drive capabilities:

Table 3.4 PIC16C92X Drive Capability

MUX	Bias	Backplanes	Segments
Static	Static	1	32
1/2	1/3	2	31
1/3	1/3	3	30
1/4	1/3	4	29

The other method of classifying LCD panels is the type of display notation used, i.e. segment, dot matrix, or functional. Segment displays are usually the 7-segment, 14-segment, or 16-segment ("British Flag") types used to create numbers and letters. These type of displays are static drive which provides the best contrast and readability in sunlight. Figure 3.15 shows all three segment displays mentioned.

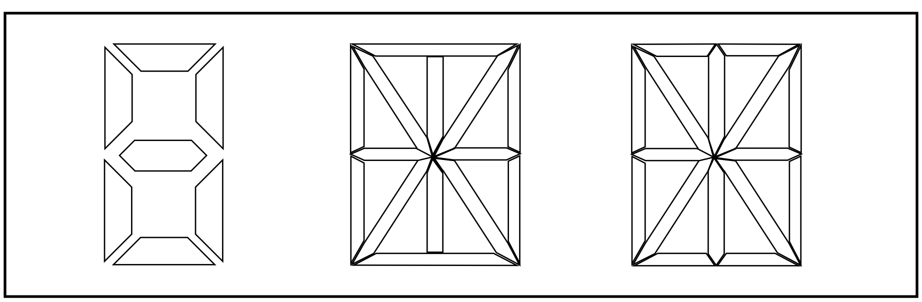


Figure 3.15 Segment Type Display

Dot matrix displays are always multiplex type displays due to the large number of pixels required and pin limitations on the driver. Dot matrix displays can create more natural letters and numbers as well as custom graphic symbols. Figure 3.16 shows a typical 5x7 dot matrix character set.

The third type of display is most commonly used in conjunction with the previous types. A function indicator or icon provides status information about the system. They are only capable of being turned on or off. One example would be a digital multimeter. The meter has 3 - 1/2 digits which are 7-segment type and also some icons for volts, amps, ohms and the ranges for m, μ, K, and M. Another example would be a cellular telephone. The LCD panel will have eight or more 5x7 dot matrix characters with icons for events such as in use, roam, no service, battery status, and signal strength. Figure 3.17 shows what a typical cellular phone panel might resemble.

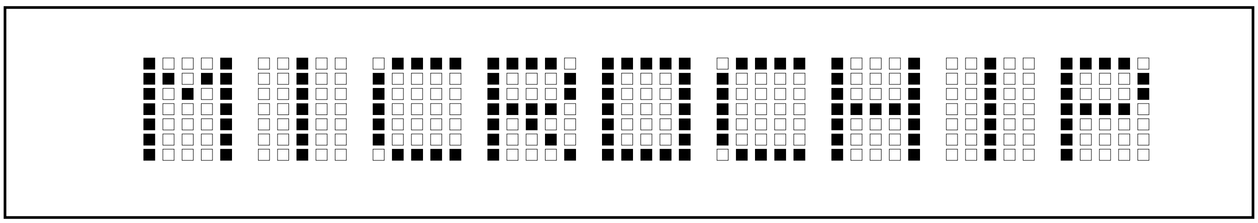


Figure 3.16 5x7 Dot Matrix Display

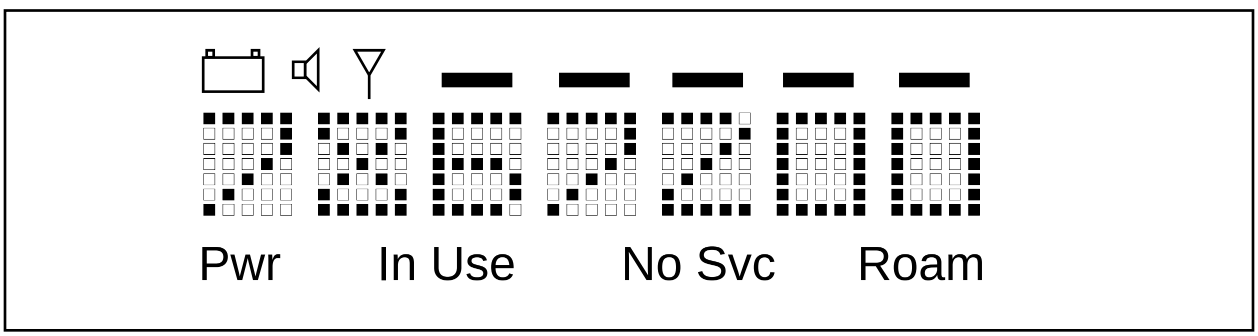


Figure 3.17 Typical Cellular Phone Panel

How are LCD panels driven?

So far, the mysteries of how an LCD is made, how it works, and what the different type of panels are have been uncovered. This section will demystify the LCD waveforms. An LCD can be characterized by the MUX ratio and bias, but one piece of information is still missing - Drive Waveforms. LCDs can be driven by two types of waveforms: Type A and Type B. Before the definitions of the two types are given, the term frame frequency must be defined. The LCD period is the rate at which the backplane and segment outputs change. The frame frequency is then calculated to be the LCD period / 2 • number of backplanes. The range of frame frequencies is from 25 to 250 Hz with the most common being between 50 and 150 Hz. Higher frequencies result in higher power consumption while lower frequencies cause flicker in the images on the LCD panel. An earlier section mentioned that a LCD driver must maintain a 0 V_{DC} potential across each pixel. Type A waveforms maintain 0 V_{DC} over a single frame whereas Type B takes two frames. Figure 3.18 shows both types of waveforms with 1/3 MUX and 1/3 Bias. PIC16C92X microcontrollers support only Type A waveforms.

The voltage applied across a particular pixel is the voltage on the COM pin minus the voltage on the SEG pin. If the resulting voltage is at or above the V_{ON} threshold then the pixel is visible. Otherwise the voltage will be at or below the V_{OFF} threshold and the pixel will not be visible. This formula is used for all drive/bias methods. The following figures show each of the modes that are currently supported by the PIC16C92X devices. Since the PIC16C92X devices only support Type A waveforms, only Type A waveforms for each of the modes are shown. Each figure has the LCD period and the frame locations marked.

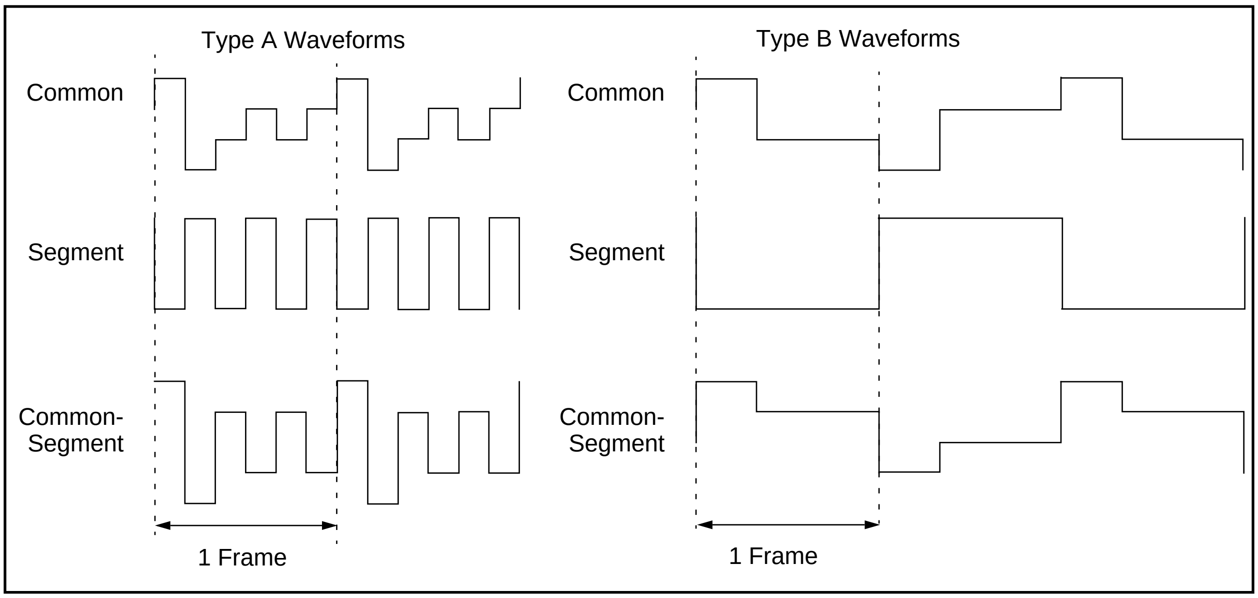


Figure 3.18 Type A vs. Type B Waveforms

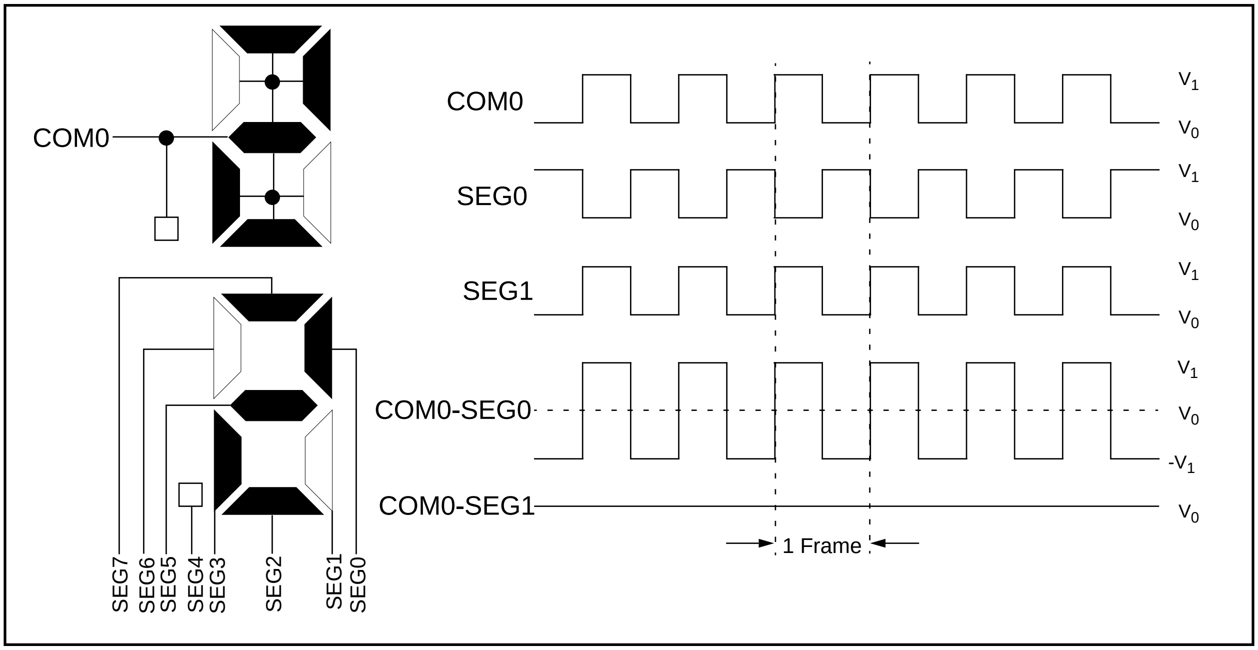


Figure 3.19 STATIC Waveforms

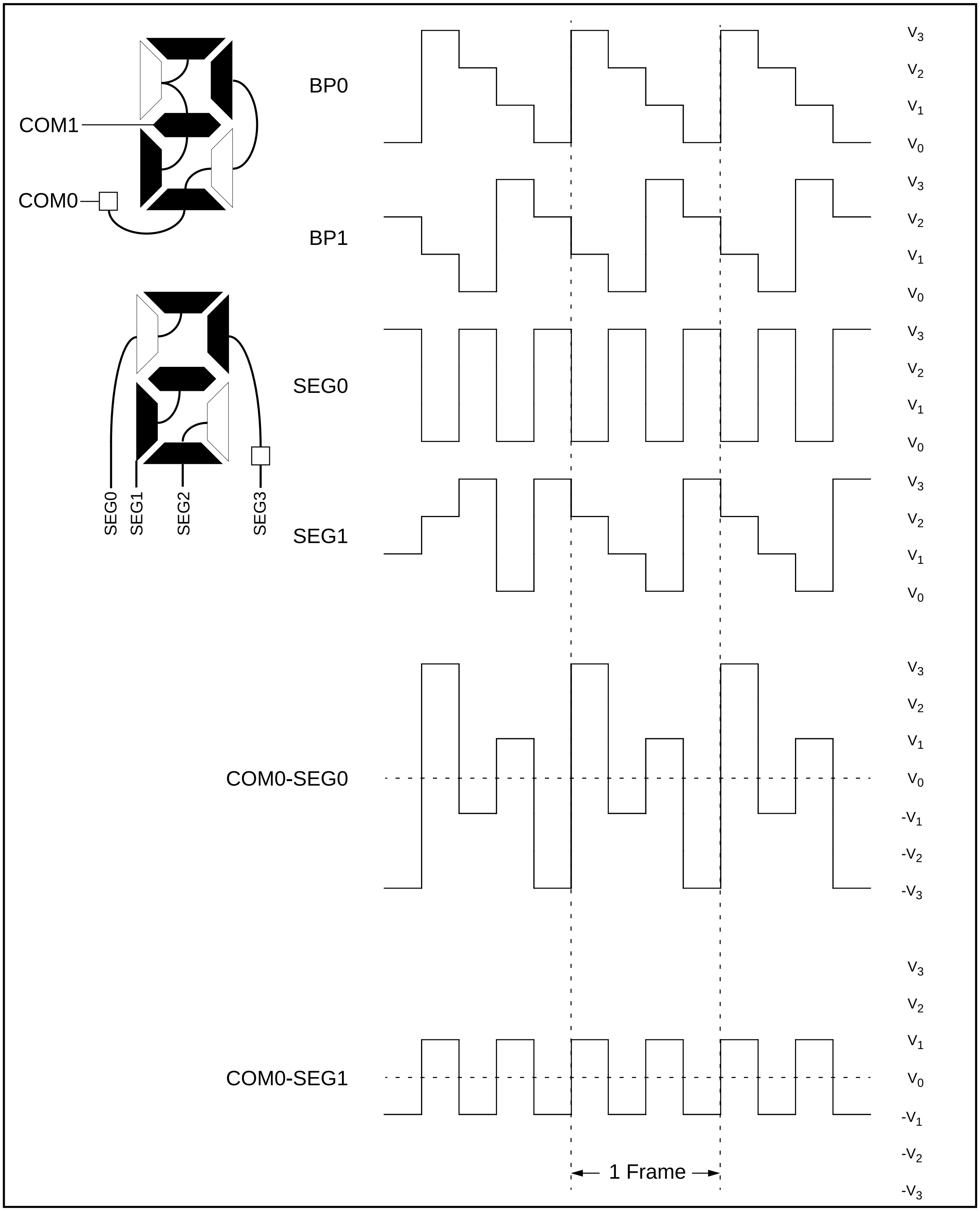


Figure 3.20 1/2 MUX, 1/3 BIAS Waveform

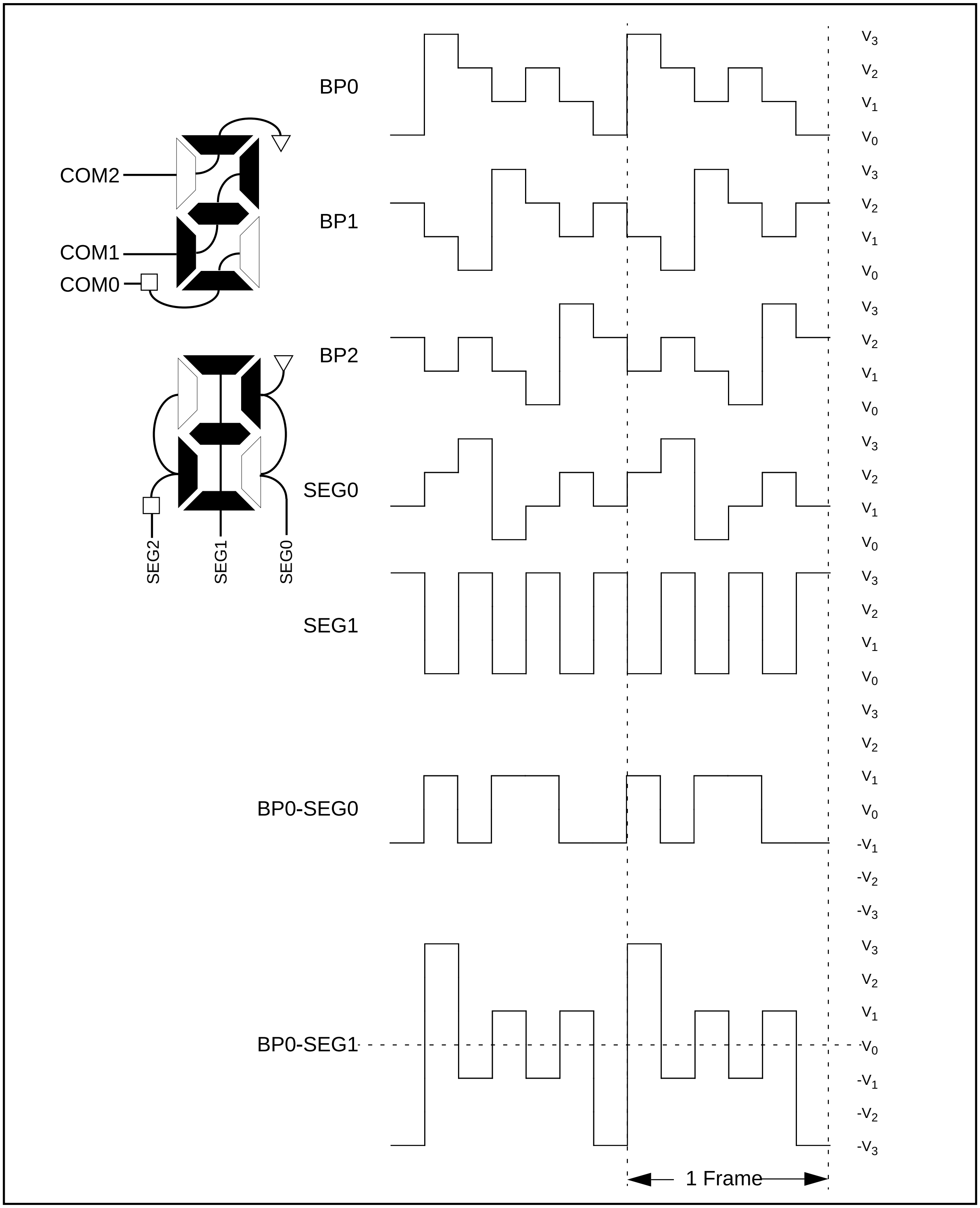


Figure 3.21 1/3 MUX, 1/3 BIAS Waveform

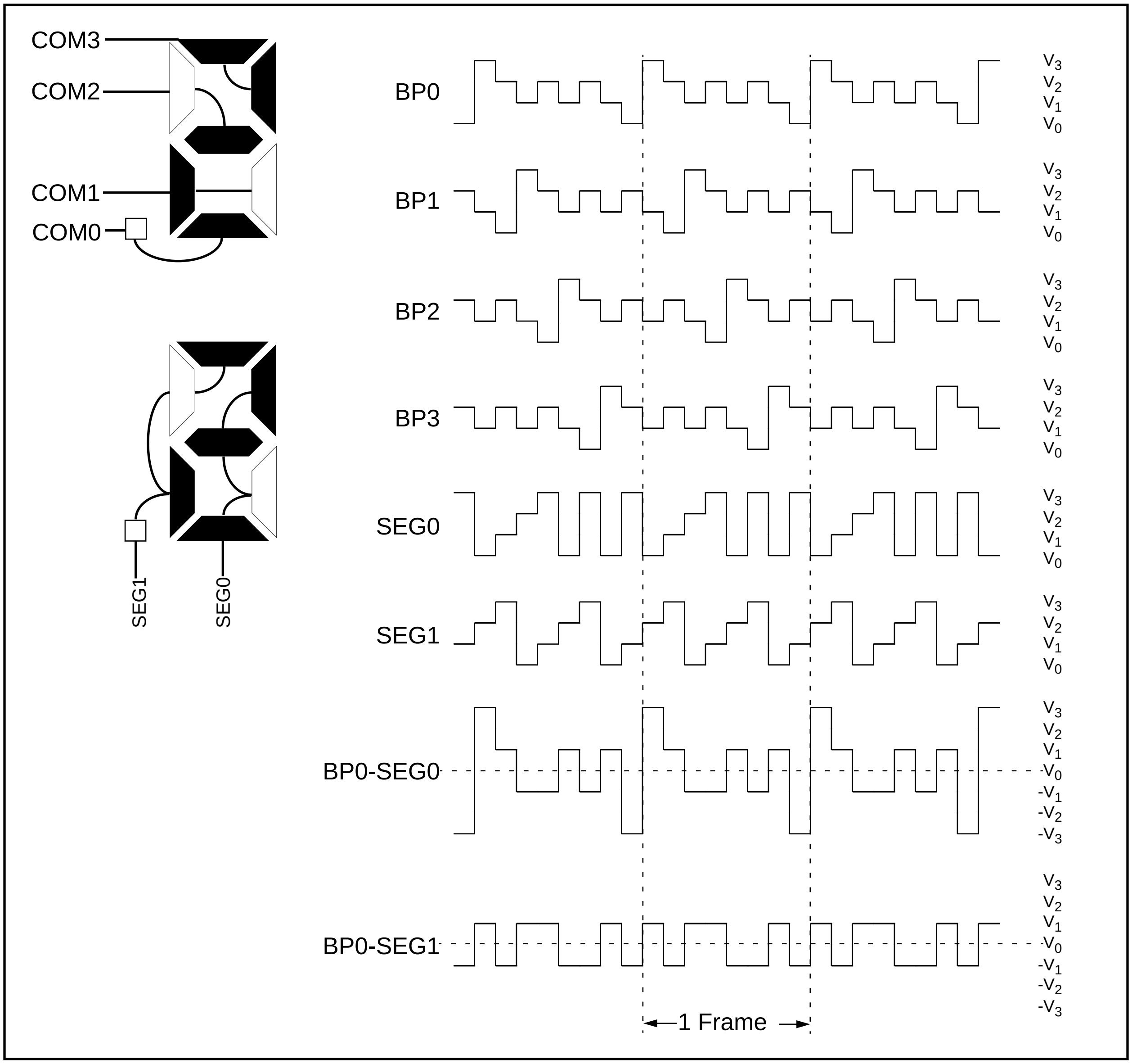


Figure 3.22 1/4 MUX, 1/3 BIAS Waveform

Discrimination Ratio

Now that the LCD waveforms have been presented, let's calculate the discrimination ratio for some of them. The first example is a static waveform from Figure 3.19. The voltages V_1 and V_0 will be assigned values of 1 and 0. The next step is to construct a matrix for one frame to help visualize the DC and RMS voltages present on an individual pixel that is ON and OFF. The rest of the following shows the calculation of the DC, RMS, and Discrimination Ratio.

BPx	0	1	
SEGx	1	0	ON
	0	1	OFF

BPx - SEGx [ON] = -1 + 1, $V_{DC} = 0$

BPx - SEGx [OFF] = 0 + 0, $V_{DC} = 0$

$$V_{RMS}\,[ON] = \Delta V \sqrt{\frac{(-1)^2 + (1)^2}{2}} = 1\Delta V$$

$$V_{RMS}\,[OFF] = \Delta V \sqrt{\frac{(0)^2 + (0)^2}{2}} = 0\Delta V$$

$$D = \frac{V_{RMS}\,[ON]}{V_{RMS}\,[OFF]} = \frac{1\Delta V}{0\Delta V} = \infty$$

Example 3.1 Discrimination Ratio Calculation for Static MUX

The next example is for Figure 3.22 which is a 1/4 MUX, 1/3 BIAS waveform. For this example, the values 3, 2, 1 and 0 will be assigned to V_3, V_2, V_1, and V_0 respectively. The frame matrix, DC voltage, RMS voltage and discrimination ratio calculations are shown in Example 3.2:

BP0	0	3	2	1	2	1	2	1	
BP1	2	1	0	3	2	1	2	1	
BP2	2	1	2	1	0	3	2	1	
BP3	2	1	2	1	2	1	0	3	
SEGx	3	0	3	0	3	0	3	0	ON
	1	2	1	2	1	2	1	2	OFF

BP0 - SEGx [ON] = - 3 + 3 - 1 + 1 - 1 + 1 - 1 + 1 $V_{DC} = 0$

BP0 - SEGx [OFF] = - 1 + 1 - 1 + 1 - 1 + 1 - 1 + 1 $V_{DC} = 0$

$$V_{RMS}\,[ON] = \Delta V \sqrt{\frac{(-3)^2 + (3)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2}{8}} = \sqrt{3}\,\Delta V$$

$$V_{RMS}\,[OFF] = \Delta V \sqrt{\frac{(-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2}{8}} = \Delta V$$

$$D = \frac{V_{RMS}\,[ON]}{V_{RMS}\,[OFF]} = \frac{\sqrt{3}\,\Delta V}{1\ \Delta V} = 1.732$$

Example 3.2 Discrimination Ratio Calculation 1/4 MUX

As shown in these examples, static displays have excellent contrast. The higher the multiplex ratio of the LCD, the lower the discrimination ratio, and therefore, the lower the contrast of the display.

The following table shows the V_{OFF}, V_{ON} and discrimination ratios of the various combinations of MUX and BIAS.

Table 3.5 Discrimination Ration vs. MUX and Bias

	1/3 BIAS		
	Voff	**Von**	**D**
STATIC	0	1	∞
1/2 MUX	0.333	0.745	2.236
1/3 MUX	0.333	0.638	1.915
1/4 MUX	0.333	0.577	1.732

Table 3.5 shows that as the multiplex of the LCD panel increases, the discrimination ratio decreases. The contrast of the panel will also decrease, so to provide better contrast the LCD voltages must be increased to provide greater separation between each level.

LCD Voltage Generation

Among the many ways to generate LCD voltage, two methods stand out above the crowd: resistor ladder and charge pump.

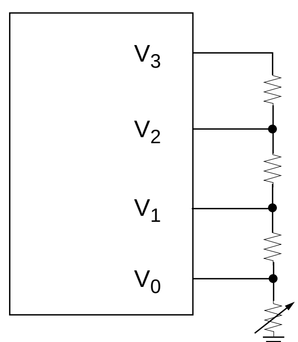


Figure 3.23 Resistor Ladder

The resistor ladder methods, shown in Figure 3.23 is most commonly used for higher V_{CC} voltages. This method uses inexpensive resistors to create the multilevel LCD voltages. Regardless of the number of pixels that are energized the current remains constant. The voltage at point V3 is typically tied to V_{CC}, either internally or externally.

The resistance values are determined by two factors: display quality and power consumption. Display quality is a function of the LCD drive waveforms. Since the LCD panel is a capacitive load, the waveform is distorted due to the charging and discharging currents. This distortion can be reduced by decreasing the value of resistance. However this change increases the power consumption due to the increased current now flowing through the resistors. As the LCD panel increases in size, the resistance value must be decreased to maintain the image quality of the display.

Sometimes the addition of parallel capacitors to the resistance can reduce the distortion caused by charging/discharging currents. This effect is limited since at some point a large resistor and large capacitor cause a voltage level shift which negatively impacts the display quality. In general, R is 1 kΩ to 50 kΩ and the potentiometer is 5 kΩ to 200 kΩ.

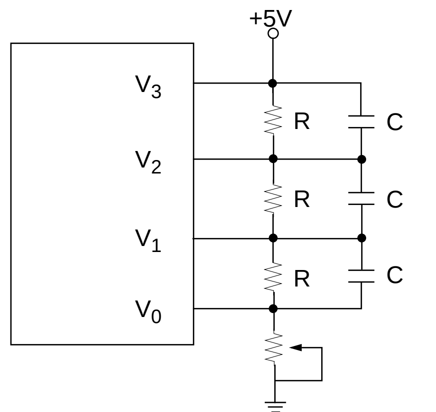


Figure 3.24 R-Ladder with Capacitors

A charge pump is ideal for low voltage battery operation because the VDD voltage can be boosted up to drive the LCD panel. The charge pump requires a charging capacitor and filter capacitor for each of the LCD voltages as seen in Figure 3.25. These capacitors are typically polyester, polypropylene, or polystyrene material. Another feature that makes the charge pump ideal for battery applications is that the current consumption is proportional to the number of pixels that are energized.

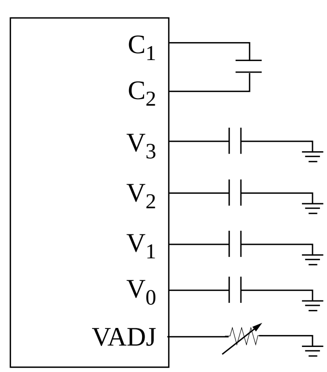


Figure 3.25 Charge Pump

Contrast

Although contrast is heavily dependent on the light source available and the multiplex mode, it also varies with the LCD voltage levels. As previously seen, a potentiometer is used to control the contrast of the LCD panel. The potentiometer sets the separation between each of the LCD voltages. The larger the separation, the better the contrast achievable.

Chapter 4. `analog.asm` Description

Introduction

This chapter describes the demonstration program for the PIC16C924, `analog.asm`. This program is a simple implementation of the PIC16C924's analog-to-digital converter.

Highlights

This chapter covers the following topics:

- **Block Diagrams**
- **`analog.asm` Source Code**

Block Diagrams

The program reads A/D channel 0 (potentiometer R2) and A/D channel 1 (thermistor RT1) and displays the results on the LCD panel. The minutes digits on the LCD panel reflect the A/D conversion result from the potentiometer R2. If the potentiometer is turned all the way clockwise then the display should read 00. If the potentiometer is turned all the way counter-clockwise, the display should read FF. The temperature digits on the LCD panel reflect the measured temperature in °C. The A/D conversion result from channel 1 is converted to °C using `thermtable` in the source code.

Description of Main Routine:

The main routine of `analog.asm` initializes the PIC16C924 and A/D converter and then enters an infinite loop that first converts A/D channel 0, then converts A/D channel 1, and finally updates the LCD panel with the new data. Then the A/D converter is enabled, followed by a 100 ms delay to allow the A/D converter to acquire the signal on channel 0. The GO bit in ADCON0 is set to start the A/D conversion. Then the program enters the infinite loop.

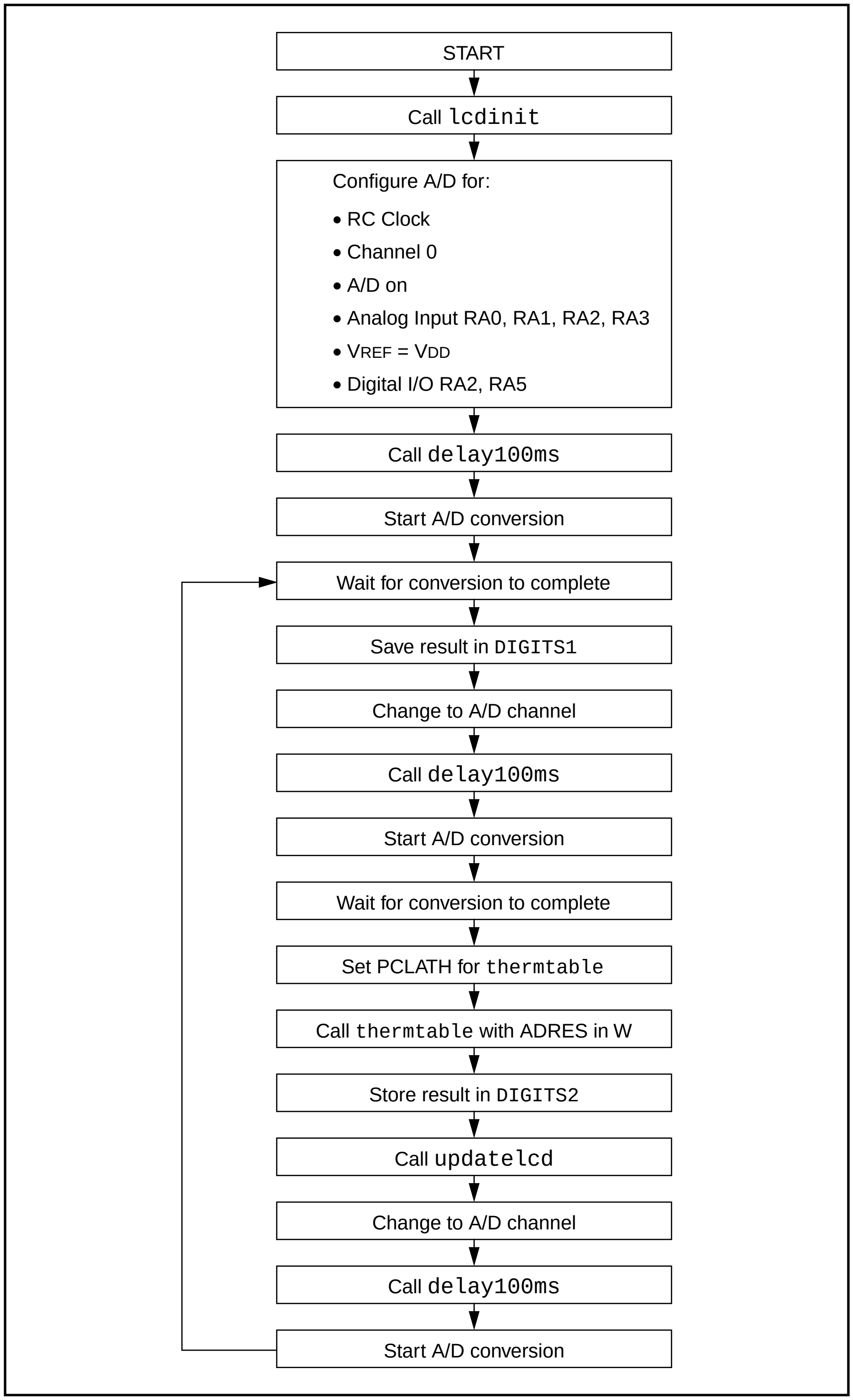


Figure 4.1 Main Routine

Description of lcdinit routine:

The routine `lcdinit` initializes the LCD Module for the following:

- External resistor ladder
- Timer1 clock source
- 1/4 MUX
- Clears only the relevant LCD Data RAM locations

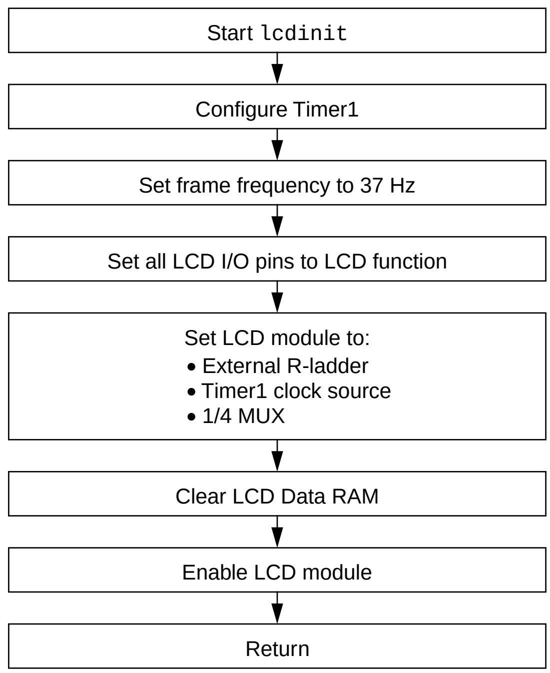


Figure 4.2 `lcdinit` Block Diagram

Description of `updatelcd` routine:

The routine `updatelcd` clears all the relevant LCD Data RAM locations and then, depending on the data, sets the appropriate bits to turn on pixels. The data values are used as an index into `sevensegtable` which provides the 7-segment decode value. This value is then used to turn on the specific pixels in the 7-segment number.

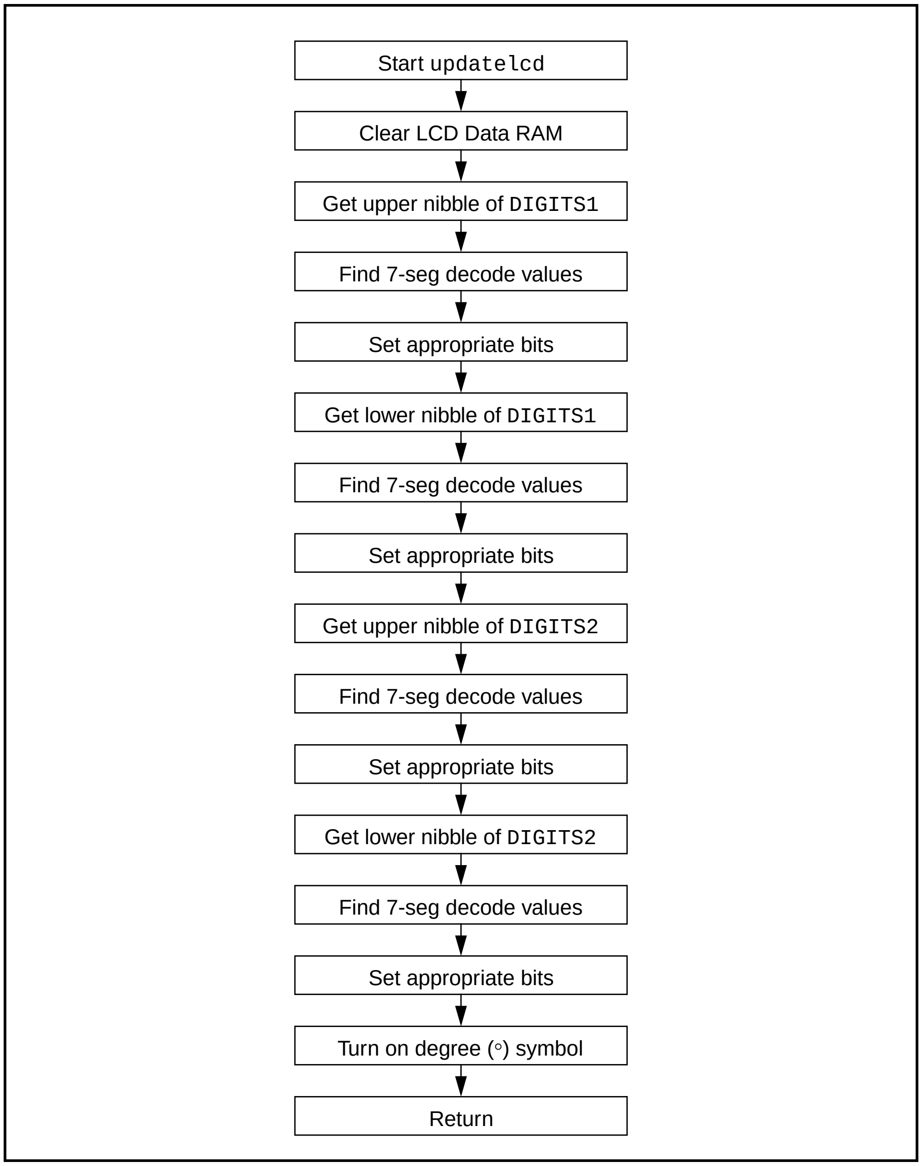


Figure 4.3 `updatelcd` Block Diagram

Description of `delay100ms` routine:

This routine provides a delay of approximately 100 ms. This delay is used to allow the A/D converter to acquire the signal at it's input.

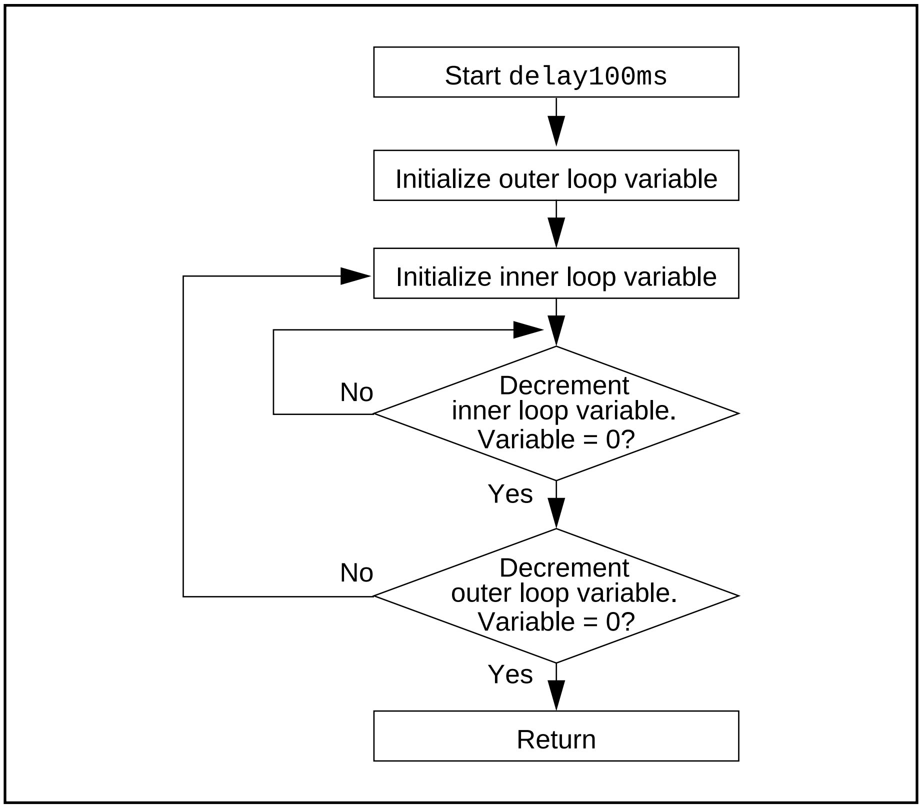


Figure 4.4 `delay100ms` Block Diagram

analog.asm Source Code

```
;*****************************************************************************
;*            Filename:    ANALOG.ASM
;*****************************************************************************
;*            Author:      Rodger Richey
;*            Company:     Microchip Technology Incorporated
;*            Revision:    1.0
;*            Date:        21 November 1996
;*            Assembled using MPASM version 1.40
;*****************************************************************************
;*            Include files:
;*                         p16c924.inc           Version 1.00
;*****************************************************************************
;*            This is a demonstration program for the PICDEM-3 board. It controls the
;*            the A/D converter.  Channel 0 is a 5K potentiometer and channel 1 is a
;*            ~5K thermistor. The A/D result for the potentiometer is displayed in the
;*            minutes digits on the LCD panel. The A/D result for the thermistor is
;*            used as a index into thermtable.  thermtable converts the A/D value into
;*            degrees C.  This value is displayed on the degrees digits on the LCD
;*            panel.
;*****************************************************************************
              list p=16C924
              include "p16c924.inc"

; Define variables, declared at 0x70-0x75 so that they are accessible across all banks
DIGITS1       equ          0x70                ; Data register for Minutes
DIGITS2       equ          0x71                ; Data register for Temperature
SEGMENT       equ          0x72                ; Holds 7-seg table results
INDEX         equ          0x73                ; Holds 7-seg table index
COUNT         equ          0x74                ; Count register for delay
COUNT1        equ          0x75                ; Count register for delay

; Define constants that are not defined in the include file
LCDEN         equ          7
W             equ          0

              org          0x0000
              goto         main

              org          0x0005
main
              bcf          STATUS,RP0
              bcf          STATUS,RP1
              call         lcdinit        ;Initialization routine for LCD Module
              movlw        0xc1           ;Configure the A/D converter for RC clock,
              movwf        ADCON0         ;channel 0, A/D on
              bsf          STATUS,RP0
              movlw        0x04           ;Configure A/D converter for RA0, RA1, and
              movwf        ADCON1         ;RA3 analog inputs, RA2,RA5 as digital I/O
              bcf          STATUS,RP0

              call         delay100ms     ;Delay 100mS for A/D to acquire signal
              bsf          ADCON0,GO      ;Start conversion on channel 0
```



```
mainloop
                btfsc       ADCON0,GO       ;Wait for A/D to complete conversion
                goto        mainloop

                movf        ADRES,W         ;Move the A/D result into DIGITS1 variable
                movwf       DIGITS1

                bsf         ADCON0,CHS0     ;Change the A/D channel to 1
                call        delay100ms      ;Delay 100mS for A/D to acquire signal
                bsf         ADCON0,GO       ;Start A/D conversion on channel 1
waitch1
                btfsc       ADCON0,GO       ;Wait for A/D conversion to complete
                goto        waitch1
                movlw       0x01
                movwf       PCLATH
                movf        ADRES,W         ;Move the A/D result into DIGITS2 variable
                call        thermtable
                movwf       DIGITS2
                clrf        PCLATH
                call        updatelcd       ;Update LCD display

                bcf         ADCON0,CHS0     ;Change the A/D channel to 0
                call        delay100ms      ;Delay 100mS for A/D to acquire signal
                bsf         ADCON0,GO       ;Start A/D conversion on channel 0
                goto        mainloop        ;Do it again!

;*************************************************************************************
;* Routine to initialize the LCD Module
;*************************************************************************************
lcdinit
                movlw       0x0f            ;Enable Timer1 to be used as clock source
                movwf       T1CON           ;for the LCD Module
                bsf         STATUS,RP1      ;Go to Bank 2
                movlw       0x06            ;Set frame freq to 37Hz
                movwf       LCDPS
                movlw       0xff            ;Enable all LCD pins as LCD drivers
                movwf       LCDSE
                movlw       0x07            ;Use ext R-ladder to generate LCD voltages,
                movwf       LCDCON          ;Timer1 clock source, 1/4 MUX
                clrf        LCDD00          ;Clear only relevant LCD data RAM locations
                clrf        LCDD01
                clrf        LCDD04
                clrf        LCDD05
                clrf        LCDD08
                clrf        LCDD09
                clrf        LCDD12
                clrf        LCDD13
                bsf         LCDCON,LCDEN    ;Enable the LCD Module
                bcf         STATUS,RP1      ;Go to Bank 0
                return
```

```
;*******************************************************************************
;* This table is used to provide the 7-SEGMENT decode values for the LCD
;* panel.
;*******************************************************************************
sevensegtable
               addwf        PCL,F          ;Add W to program counter for table offset
               ;               gfedcba
               retlw        b'00111111'    ;zero
               retlw        b'00000110'    ;one
               retlw        b'01011011'    ;two
               retlw        b'01001111'    ;three
               retlw        b'01100110'    ;four
               retlw        b'01101101'    ;five
               retlw        b'01111101'    ;six
               retlw        b'00000111'    ;seven
               retlw        b'01111111'    ;eight
               retlw        b'01101111'    ;nine
               retlw        b'01110111'    ;ten
               retlw        b'01111100'    ;eleven
               retlw        b'01011000'    ;twelve
               retlw        b'01011110'    ;thirteen
               retlw        b'01111001'    ;fourteen
               retlw        b'01110001'    ;fifteen

;*******************************************************************************
;* Routine to take the A/D conversion results and display them to the LCD.
;*******************************************************************************
updatelcd
               bsf          STATUS,RP1     ;Go to Bank 2
               clrf         LCDD00         ;Clear only the relevant LCD RAM locations
               clrf         LCDD01
               clrf         LCDD04
               clrf         LCDD05
               clrf         LCDD08
               clrf         LCDD09
               clrf         LCDD12
               clrf         LCDD13

               movlw        0xf0           ;Use only upper 4-bits of DIGITS1 to find
               andwf        DIGITS1,W      ;the 7-SEGMENT decode
               movwf        INDEX
               rrf          INDEX
               rrf          INDEX
               rrf          INDEX
               rrf          INDEX
               movlw        0x0f
               andwf        INDEX,W
               call         sevensegtable
               movwf        SEGMENT
               btfsc        SEGMENT,0      ;Take the 7-SEGMENT decode value and set the
               bsf          LCDD13,2       ;appropriate bits in the LCD data RAM
               btfsc        SEGMENT,1
               bsf          LCDD08,2
               btfsc        SEGMENT,2
               bsf          LCDD04,2
```

```
        btfsc       SEGMENT,3
        bsf         LCDD01,2
        btfsc       SEGMENT,4
        bsf         LCDD04,1
        btfsc       SEGMENT,5
        bsf         LCDD09,2
        btfsc       SEGMENT,6
        bsf         LCDD05,2

        movlw       0x0f            ;Use only lower 4-bits of DIGITS1 to find
        andwf       DIGITS1,W       ;the 7-SEGMENT decode
        call        sevensegtable
        movwf       SEGMENT
        btfsc       SEGMENT,0       ;Take the 7-SEGMENT decode value and set the
        bsf         LCDD13,1        ;appropriate bits in the LCD data RAM
        btfsc       SEGMENT,1
        bsf         LCDD08,3
        btfsc       SEGMENT,2
        bsf         LCDD04,3
        btfsc       SEGMENT,3
        bsf         LCDD01,1
        btfsc       SEGMENT,4
        bsf         LCDD05,1
        btfsc       SEGMENT,5
        bsf         LCDD12,2
        btfsc       SEGMENT,6
        bsf         LCDD09,1

        movlw       0xf0            ;Use only upper 4-bits of DIGITS1 to find
        andwf       DIGITS2,W       ;the 7-SEGMENT decode
        movwf       INDEX
        rrf         INDEX
        rrf         INDEX
        rrf         INDEX
        rrf         INDEX
        movlw       0x0f
        andwf       INDEX,W
        call        sevensegtable
        movwf       SEGMENT
        btfsc       SEGMENT,0       ;Take the 7-SEGMENT decode value and set the
        bsf         LCDD12,7        ;appropriate bits in the LCD data RAM
        btfsc       SEGMENT,1
        bsf         LCDD12,4
        btfsc       SEGMENT,2
        bsf         LCDD04,4
        btfsc       SEGMENT,3
        bsf         LCDD00,7
        btfsc       SEGMENT,4
        bsf         LCDD04,7
        btfsc       SEGMENT,5
        bsf         LCDD13,0
        btfsc       SEGMENT,6
        bsf         LCDD08,7

        movlw       0x0f            ;Use only lower 4-bits of DIGITS1 to find
```

```
            andwf       DIGITS2,W      ;the 7-SEGMENT decode
            call        sevensegtable
            movwf       SEGMENT
            btfsc       SEGMENT,0      ;Take the 7-SEGMENT decode value and set the
            bsf         LCDD12,6       ;appropriate bits in the LCD data RAM
            btfsc       SEGMENT,1
            bsf         LCDD08,5
            btfsc       SEGMENT,2
            bsf         LCDD04,5
            btfsc       SEGMENT,3
            bsf         LCDD00,6
            btfsc       SEGMENT,4
            bsf         LCDD04,6
            btfsc       SEGMENT,5
            bsf         LCDD08,4
            btfsc       SEGMENT,6
            bsf         LCDD08,6
            bsf         LCDD12,5       ;Turn on the degrees symbol
            bcf         STATUS,RP1     ;Go to Bank 0
            return

;*************************************************************************************
;* Routine to delay approximately 100mS with a 4MHz oscillator.
;*************************************************************************************
delay100ms
            movlw       0x64           ;Move 100 into the COUNT variable
            movwf       COUNT
dlms1
            movlw       0xf9           ;Move 249 into the COUNT1 variable
            movwf       COUNT1
dlms2
            nop
            decfsz      COUNT1,F
            goto        dlms2
            decfsz      COUNT,F
            goto        dlms1
            return

            org         0x0100
;*************************************************************************************
;* This table converts the A/D value of the thermister and converts it
;* to temperature in degrees C.
;*************************************************************************************
thermtable
            addwf       PCL,F
            DT
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
            DT
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
            DT
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x02,0x03,0x04
            DT
0x05,0x06,0x06,0x07,0x08,0x09,0x09,0x10,0x11,0x11,0x12,0x12,0x13,0x13,0x14,0x14
            DT
0x15,0x15,0x16,0x16,0x17,0x17,0x18,0x18,0x19,0x19,0x20,0x20,0x21,0x21,0x22,0x22
```



```
                DT
0x23,0x23,0x24,0x24,0x25,0x25,0x25,0x26,0x26,0x27,0x27,0x28,0x28,0x28,0x29,0x29
                DT
0x30,0x30,0x30,0x31,0x31,0x32,0x32,0x32,0x33,0x33,0x33,0x34,0x34,0x35,0x35,0x35
                DT
0x36,0x36,0x37,0x37,0x38,0x38,0x38,0x39,0x39,0x40,0x40,0x40,0x41,0x41,0x42,0x42
                DT
0x43,0x43,0x43,0x44,0x44,0x45,0x45,0x45,0x46,0x46,0x47,0x47,0x48,0x48,0x48,0x49
                DT
0x49,0x50,0x50,0x50,0x51,0x51,0x52,0x52,0x53,0x53,0x54,0x54,0x55,0x55,0x56,0x56
                DT
0x57,0x57,0x58,0x58,0x59,0x59,0x60,0x60,0x61,0x61,0x62,0x62,0x63,0x63,0x64,0x64
                DT
0x65,0x65,0x66,0x66,0x67,0x67,0x68,0x68,0x69,0x69,0x70,0x71,0x71,0x72,0x73,0x73
                DT
0x74,0x74,0x75,0x76,0x76,0x77,0x78,0x78,0x79,0x79,0x80,0x81,0x81,0x82,0x83,0x84
                DT
0x84,0x85,0x86,0x87,0x88,0x89,0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99
                DT
0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99
                DT 0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99,0x99


                END
```

Chapter 5. keypad.asm Description

Introduction

This chapter describes the demonstration program for the PIC16C924, `keypad.asm`. This program is used to read a 4x4 keypad connected to JP1. The value of each key pressed is displayed on the LCD panel.

Highlights

This chapter covers the following topics:

- **Block Diagrams**
- **`keypad.asm` Source Code**

Block Diagrams

This program uses the interrupt on change feature of PORTB to detect when a key has been pressed and to decode the value of that key. The value is then displayed on the LCD panel.

Description of Main Routine:

The main routine uses the `lcdinit` and `keypadinit` routines to configure the LCD Module and PORTB for use with a keypad. The keypad is connected to JP1. After initialization the routine enters an infinite loop that waits for a key to be pressed, then displays the value of that key on the LCD panel.

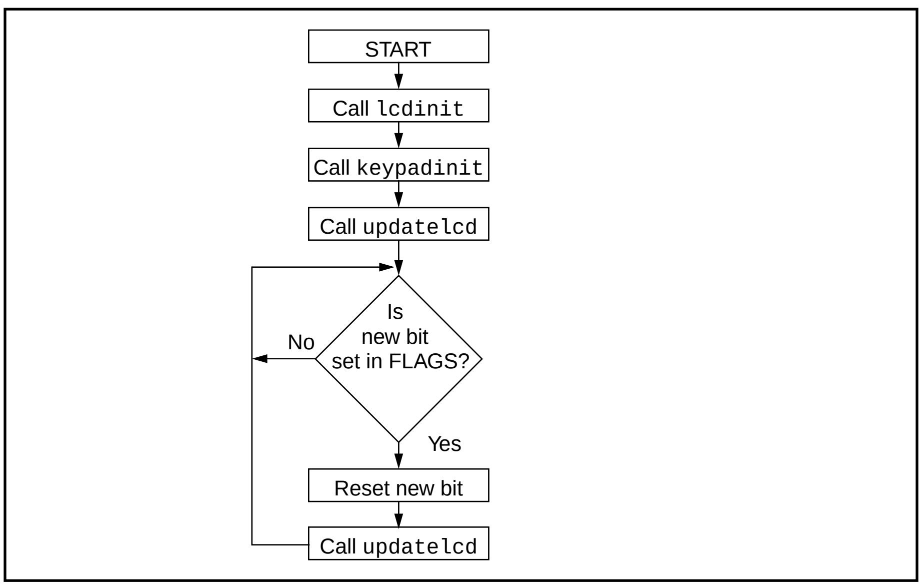


Figure 5.1 Main Routine Block Diagram

Description of `lcdinit` routine:

The routine `lcdinit` initializes the LCD Module for the following:

- External resistor ladder
- Timer1 clock source
- 1/4 MUX
- Clears only the relevant LCD Data RAM locations

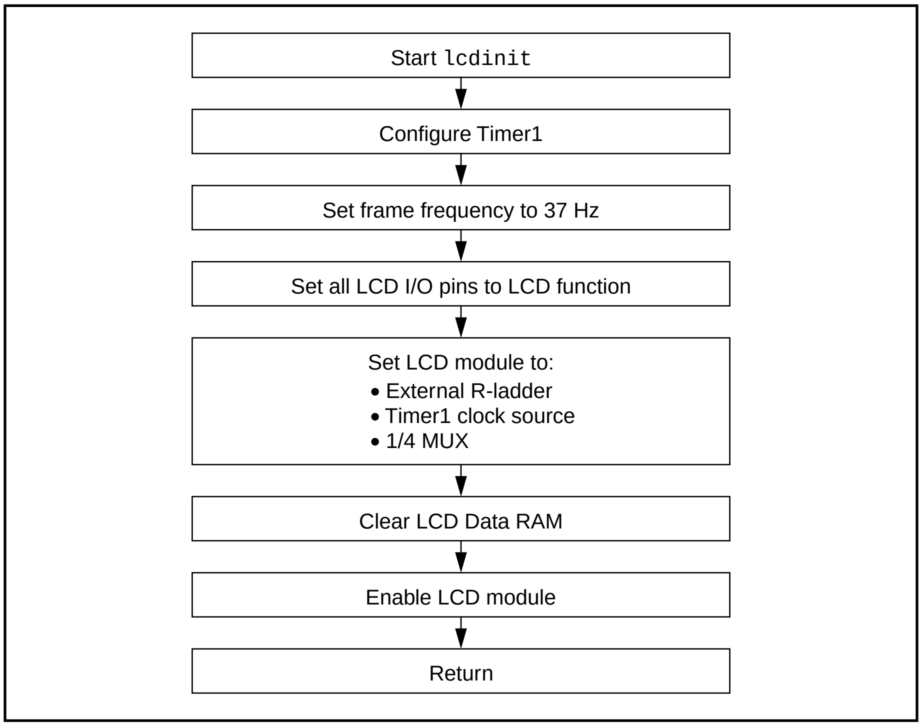


Figure 5.2 `lcdinit` Routine Block Diagram

Description of updatelcd routine:

The routine `updatelcd` clears all the relevant LCD Data RAM locations and then, depending on the data, sets the appropriate bits to turn on pixels. The data values are used as an index into `sevensegtable` which provides the 7-segment decode value. This value is then used to turn on the specific pixels in the 7-segment number.

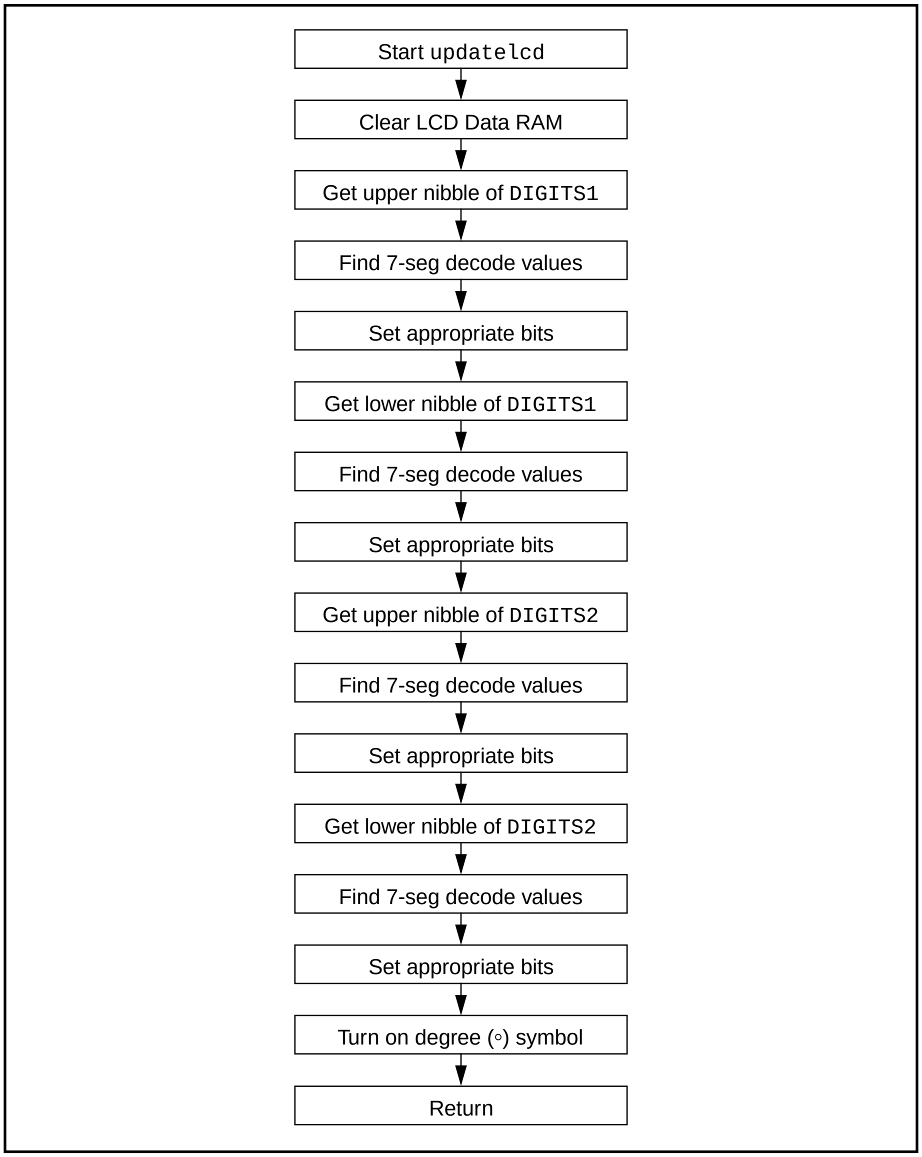


Figure 5.3 updatelcd Routine Block Diagram

Description of `keypadinit` routine:

This routine configures PORTB to connect to the keypad and enables the pull-up resistors. It also clears the mismatch condition on PORTB, clears the RBIF flag, and enables interrupts (RBIE and GIE).

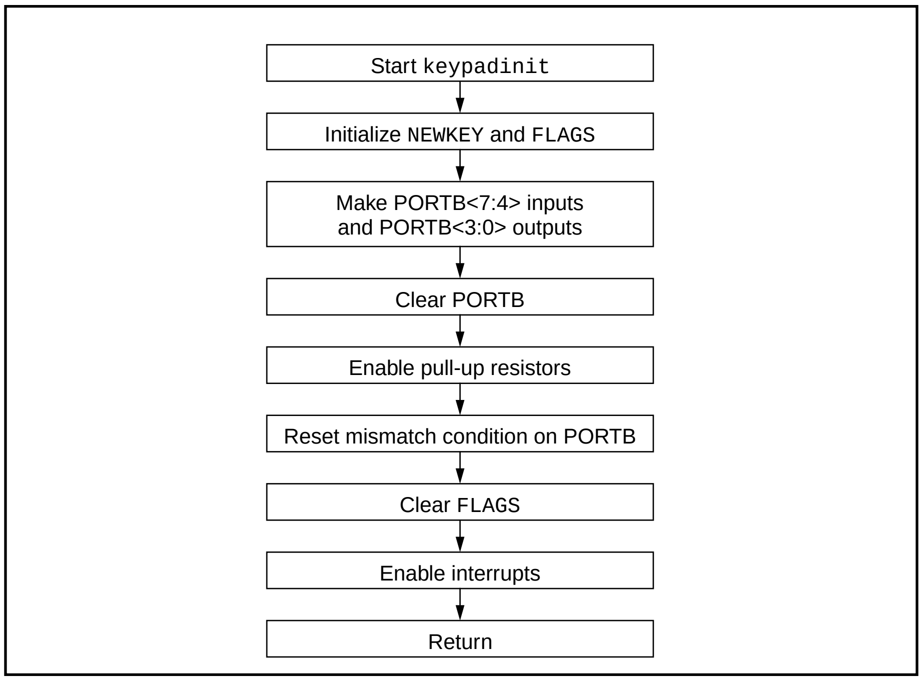


Figure 5.4 `keypadinit` Block Diagram

Description of `servicekeypad` routine:

This routine determines which of the keys in a 4x4 matrix has been pressed. The internal pull-up resistors set the upper 4-bits of PORTB. When a key is pressed, the corresponding pin is pulled low. Figure 8.2 in Chapter 8 shows the configuration of the keypad when connected to JP1 on the PICDEM-3 board. The routine will set all pins connected to the columns except one. It then searches the value on PORTB to see which row is cleared. If there are not any bits cleared, the routine will continue on to the next column. Once the routine has found the key that is pressed, it waits for that key to be released. Finally, it sets the `NEWKEY` flag, clears the mismatch condition on PORTB, and enables the interrupt.

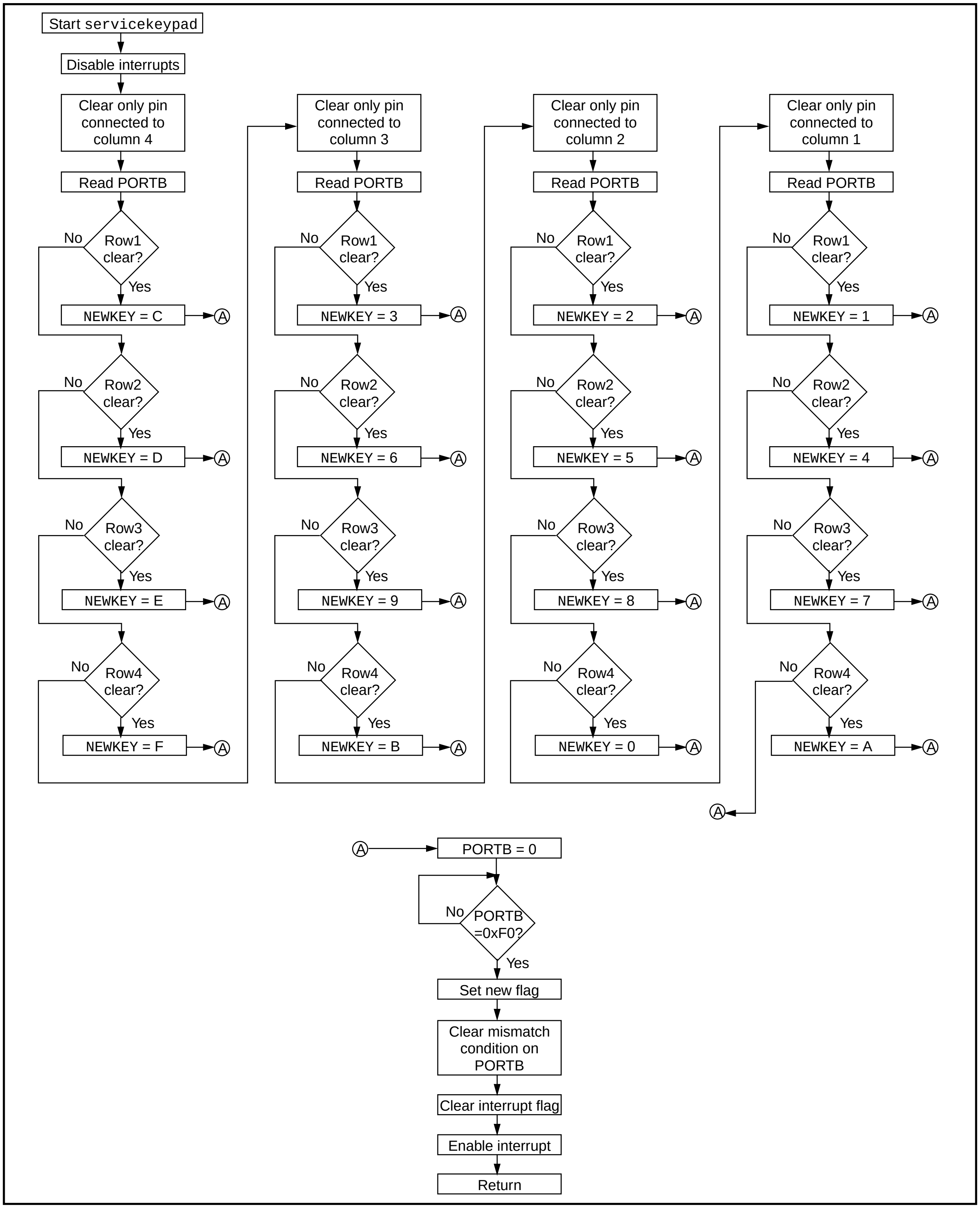


Figure 5.5 servicekeypad Block Diagram

keypad.asm Source Code

```
;****************************************************************
;*            Filename:     KEYPAD.ASM
;****************************************************************
;*            Author:       Rodger Richey
;*            Company:      Microchip Technology Incorporated
;*            Revision:     1.0
;*            Date          26 November 1996
;*            Assembled using MPASM version 1.40
;****************************************************************
;*            Include files:
;*                          p16c924.inc Version 1.00
;****************************************************************
;*            This is a demonstration program for the PICDEM-3
;*            board.  It reads a 4x4 keypad connected to PORTB
;*            using JP1.  The value of the key pressed is dis-
;*            played on the LCD in the temperature digits. The
;*            program uses the wakeup-on-change interrupt of
;*            PORTB.
;****************************************************************
              list p=16c924
              include "p16c924.inc"
; Variable declaration, located at 0x70-0x74 to be accessible across all banks
FLAGS         equu          0x70          ; Flags register
NEWKEY        equ           0x71          ; Holds value of keypress
INDEX         equ           0x72          ; Index into 7-seg table
SEGMENT       equ           0x73          ; Result of 7-seg table access
INCODE        equ           0x74          ; Temp variable

NEW           equ           0             ; Flag indicating new keypress
LCDEN         equ           7             ; LCD enable bit in LCDCON reg
W             equ           0             ; W flag for assembler instr.

              org           0x0000
              goto          main          ; Reset vector, go to main

              org           0x0004
              goto          servicekeypad ; Interrupt vector

              org           0x0010
main
              bcf           STATUS,RP0    ; Bank 0
              bcf           STATUS,RP1
              call          lcdinit       ; Initialize LCD Module
              call          keypadinit    ; Initialize for keypad

              call          updatelcd     ; Update contents of LCD

mainloop
              btfss         FLAGS,NEW     ; Wait for key press
              goto          mainloop

              bcf           FLAGS,NEW     ; Clear flag
```

```
            call        updatelcd    ; Update contents of LCD
            goto        mainloop

;*******************************************************************************
;* Routine to initialize the LCD Module
;*******************************************************************************
lcdinit
            movlw       0x0f         ;Enable Timer1
            movwf       T1CON
            bsf         STATUS,RP1   ;Go to Bank 2
            movlw       0x06         ;Set frame freq to 37Hz
            movwf       LCDPS
            movlw       0xff         ;Enable all LCD pins as LCD drivers
            movwf       LCDSE
            movlw       0x07         ;Use ext R-ladder to generate LCD voltages,
            movwf       LCDCON       ;Timer1 clock source, 1/4 MUX
            clrf        LCDD00       ;Clear only relevant LCD data RAM locations
            clrf        LCDD01
            clrf        LCDD04
            clrf        LCDD05
            clrf        LCDD08
            clrf        LCDD09
            clrf        LCDD12
            clrf        LCDD13
            bsf         LCDCON,LCDEN ;Enable the LCD Module
            bcf         STATUS,RP1   ;Go to Bank 0
            return

;********************************************************************************
;* This table is used to provide the 7-segment decode values for the LCD
;* panel.
;********************************************************************************
sevensegtable
            addwf       PCL,F        ;Add W to the program counter for table offset
            ;              gfedcba
            retlw       b'00111111'  ;zero
            retlw       b'00000110'  ;one
            retlw       b'01011011'  ;two
            retlw       b'01001111'  ;three
            retlw       b'01100110'  ;four
            retlw       b'01101101'  ;five
            retlw       b'01111101'  ;six
            retlw       b'00000111'  ;seven
            retlw       b'01111111'  ;eight
            retlw       b'01101111'  ;nine
            retlw       b'01110111'  ;ten
            retlw       b'01111100'  ;eleven
            retlw       b'01011000'  ;twelve
            retlw       b'01011110'  ;thirteen
            retlw       b'01111001'  ;fourteen
            retlw       b'01110001'  ;fifteen

;*******************************************************************
;* Routine to display results to the LCD.
;*******************************************************************
```

```
updatelcd
            bsf         STATUS,RP1    ;Go to Bank 2
            clrf        LCDD00        ;Clear only the relevant LCD RAM locations
            clrf        LCDD01
            clrf        LCDD04
            clrf        LCDD05
            clrf        LCDD08
            clrf        LCDD09
            clrf        LCDD12
            clrf        LCDD13

            movlw       0xf0          ;Use only the upper 4-bits of digits1 to find
            andwf       NEWKEY,W      ;the 7-segment decode
            movwf       INDEX
            rrf         INDEX,F
            rrf         INDEX,F
            rrf         INDEX,F
            rrf         INDEX,F
            movlw       0x0f
            andwf       INDEX,W
            call        sevensegtable
            movwf       SEGMENT
            btfsc       SEGMENT,0    ;Take the 7-segment decode value and set the
            bsf         LCDD12,7     ;appropriate bits in the LCD data RAM
            btfsc       SEGMENT,1
            bsf         LCDD12,4
            btfsc       SEGMENT,2
            bsf         LCDD04,4
            btfsc       SEGMENT,3
            bsf         LCDD00,7
            btfsc       SEGMENT,4
            bsf         LCDD04,7
            btfsc       SEGMENT,5
            bsf         LCDD13,0
            btfsc       SEGMENT,6
            bsf         LCDD08,7

            movlw       0x0f         ;Use only lower 4-bits of digits1 to find
            andwf       NEWKEY,W     ;the 7-segment decode
            call        sevensegtable
            movwf       SEGMENT
            btfsc       SEGMENT,0    ;Take the 7-segment decode value and set the
            bsf         LCDD12,6     ;appropriate bits in the LCD data RAM
            btfsc       SEGMENT,1
            bsf         LCDD08,5
            btfsc       SEGMENT,2
            bsf         LCDD04,5
            btfsc       SEGMENT,3
            bsf         LCDD00,6
            btfsc       SEGMENT,4
            bsf         LCDD04,6
            btfsc       SEGMENT,5
            bsf         LCDD08,4
            btfsc       SEGMENT,6
            bsf         LCDD08,6
```

```
            bcf         STATUS,RP1          ;Go to Bank 0
            return

;*********************************************************************
;* Routine to initialize the 924 for using a keypad.
;*********************************************************************
keypadinit
            clrf        NEWKEY              ; Clear registers
            clrf        FLAGS
            bsf         STATUS,RP0
            movlw       0xf0                ; RB4:RB7 are interrupt on
            movwf       TRISB               ; change pins
            bcf         OPTION_REG,NOT_RBPU ; Enable internal pullups
            bcf         STATUS,RP0
            clrf        PORTB               ; Set PORTB outputs low
            movf        PORTB,W             ; Clear mismatch on PORTB
            bcf         INTCON,RBIF         ; Clear flag and enable interrupts
            bsf         INTCON,RBIE
            bsf         INTCON,GIE
            return

;*********************************************************************
;* Routine to decode what key is pressed.
;*********************************************************************
servicekeypad
            bcf         STATUS,RP1
            bcf         INTCON,RBIE  ; Disable further interrupts

col4                                 ; Start with column 4 C,D,E,F
            movlw       0x0e         ; Leave only pin connected to
            movwf       PORTB        ; column 4 low
            movlw       0xf0         ; Read PORTB and mask off
            andwf       PORTB,W      ; lower 4-bits
            movwf       INCODE

            movlw       0x70         ; Row 1 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r2c4
            movlw       0x0c         ; If low, C is pressed
            movwf       NEWKEY
            goto        debounce     ; Done, wait for key to be released
r2c4
            movlw       0xb0         ; Row 2 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r3c4
            movlw       0x0d         ; If low, D is pressed
            movwf       NEWKEY
            goto        debounce     ; Done, wait for key to be released
r3c4
            movlw       0xd0         ; Row 3 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r4c4
```

```
            movlw       0x0e        ; If low, E is pressed
            movwf       NEWKEY
            goto        debounce    ; Done, wait for key to be released
r4c4
            movlw       0xe0        ; Row 4 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        col3
            movlw       0x0f        ; If low F is pressed
            movwf       NEWKEY
            goto        debounce    ; Done, wait for key to be released

col3                                ; Column 3
            movlw       0x0d        ; Leave only pin connected to
            movwf       PORTB       ; column 3 low
            movlw       0xf0        ; Read PORTB and mask off
            andwf       PORTB,W     ; lower 4-bits
            movwf       INCODE

            movlw       0x70        ; Row 1 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r2c3
            movlw       0x03        ; If low, 3 was pressed
            movwf       NEWKEY
            goto        debounce    ; Done, wait for key to be released
r2c3
            movlw       0xb0        ; Row 2 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r3c3
            movlw       0x06        ; If low, 6 was pressed
            movwf       NEWKEY
            goto        debounce    ; Done, wait for key to be released
r3c3
            movlw       0xd0        ; Row 3 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r4c3
            movlw       0x09        ; If low, 9 was pressed
            movwf       NEWKEY
            goto        debounce    ; Done, wait for key to be released
r4c3
            movlw       0xe0        ; Row 4 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        col2
            movlw       0x0b        ; If low, B was pressed
            movwf       NEWKEY
            goto        debounce    ; Done, wait for key to be released

col2                                ; Column 2
            movlw       0x0b        ; Leave only pin connected to
            movwf       PORTB       ; column 2 low
            movlw       0xf0        ; Read PORTB and mask off
```



```
            andwf       PORTB,W       ; lower 4-bits
            movwf       INCODE

            movlw       0x70          ; Row 1 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r2c2
            movlw       0x02          ; If low, 2 was pressed
            movwf       NEWKEY
            goto        debounce      ; Done, wait for key to be released
r2c2
            movlw       0xb0          ; Row 2 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r3c2
            movlw       0x05          ; If low, 5 was pressed
            movwf       NEWKEY
            goto        debounce      ; Done, wait for key to be released
r3c2
            movlw       0xd0          ; Row 3 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r4c2
            movlw       0x08          ; If low, 8 was pressed
            movwf       NEWKEY
            goto        debounce      ; Done, wait for key to be released
r4c2
            movlw       0xe0          ; Row 4 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        col1
            clrf        NEWKEY        ; If low 0 was pressed
            goto        debounce      ; Done, wait for key to be released

col1                                  ; Column 1
            movlw       0x07          ; Leave only pin connected to
            movwf       PORTB         ; column 1 low
            movlw       0xf0          ; Read PORTB and mask off
            andwf       PORTB,W       ; lower 4-bits
            movwf       INCODE

            movlw       0x70          ; Row 1 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r2c1
            movlw       0x01          ; If low, 1 was pressed
            movwf       NEWKEY
            goto        debounce      ; Done, wait for key to be released
r2c1
            movlw       0xb0          ; Row 2 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r3c1
            movlw       0x04          ; If low, 4 was pressed
            movwf       NEWKEY
```

```
            goto        debounce      ; Done, wait for key to be released
r3c1
            movlw       0xd0          ; Row 3 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        r4c1
            movlw       0x07          ; If low, 7 was pressed
            movwf       NEWKEY
            goto        debounce      ; Done, wait for key to be released
r4c1
            movlw       0xe0          ; Row 4 low?
            subwf       INCODE,W
            btfss       STATUS,Z
            goto        debounce
            movlw       0x0a          ; If low, A was pressed
            movwf       NEWKEY

debounce                              ; Wait for key to be released
            clrf        PORTB         ; Clear PORTB
release
            movf        PORTB,W       ; Check to see if key released
            sublw       0xf0          ; When key released, PORTB
            btfss       STATUS,Z      ; reads 0xf0
            goto        release

            bsf         FLAGS,NEW     ; Set new key flag
            movf        PORTB,W       ; Reset mismatch on PORTB
            bcf         INTCON,RBIF   ; Clear flag and enable
            bsf         INTCON,RBIE   ; interrupt

            retfie                    ; return from interrupt

            END
```

Chapter 6. `uart.asm` Description

Introduction

This chapter describes the demonstration program for the PIC16C924, `uart.asm`. This program implements a software serial port capable of transmitting and receiving RS-232 data. A 4 MHz crystal is required to run this example program to keep consistent timing of bits.

Highlights

This chapter covers the following topics:

- **Block Diagrams**
- **`uart.asm` Source Code**

Block Diagrams

When the program starts, it sends the message "Greetings from PICDEM-3!". Then it waits for a response from the Host PC. Every character that the Host PC sends to the PICDEM-3 board is bounced back to the Host PC. The hex value of the character is also displayed on the LCD panel. Any terminal program can be used to communicate with the PICDEM-3 board.

Description of Main Routine:

The main routine of `uart.asm` initializes the LCD Module and I/O port and then sends the greeting to the Host PC followed by a carriage return and linefeed. It now enters an infinite loop that waits for the Host PC to send a character. Whenever a character is received, the program will echo the character to the Host PC followed by a carriage return and linefeed. It will also update the LCD panel with the hex value of the character. The routine `printcrlf` simply sends a carriage return (decimal 13) and a line feed (decimal 10) to the Host PC using the `uarttx` routine. The `printstring` routine reads data out of `stringtable` and sends it to the Host PC using `uarttx`. When a '`\0`' is encountered the end of the table has been reached and the routine returns.

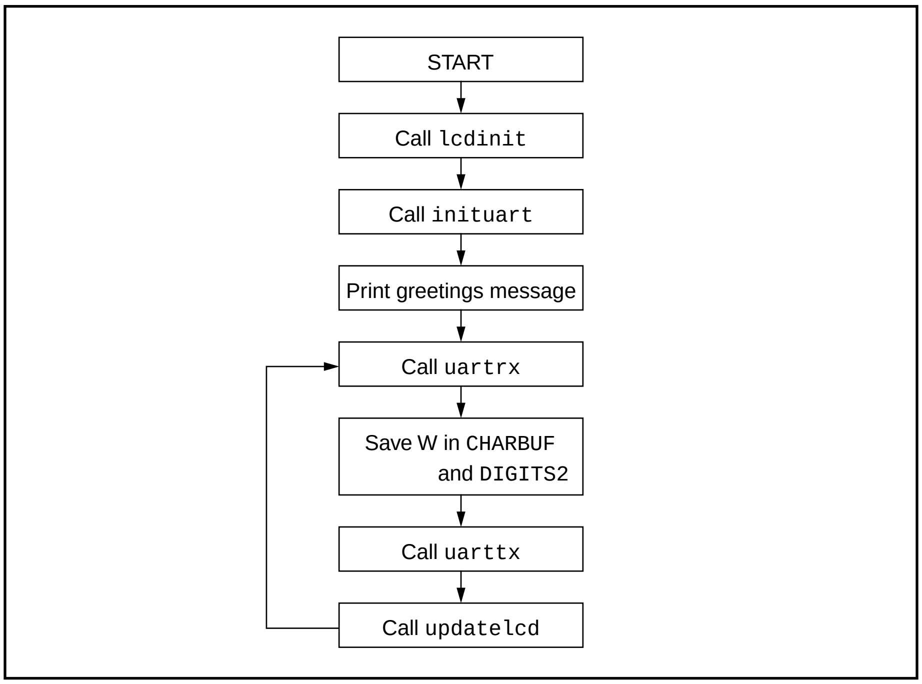


Figure 6.1 Main Routine Block Diagram

Description of `uarttx` routine:

The `uarttx` routine takes the value in the W register and transmits it out serially on the RA3 I/O pin. It first sends a start bit (0) to the Host PC. RA3 is cleared and the routine `delaybit` is used to delay one bit time. Then each bit of the data starting with the LSb is shifted into CARRY. Depending on the state of the CARRY bit, RA3 is set or cleared followed by a delay of one bit time. When all eight bits have been shifted out, RA3 is set to indicate a stop bit again followed by a delay of one bit time.

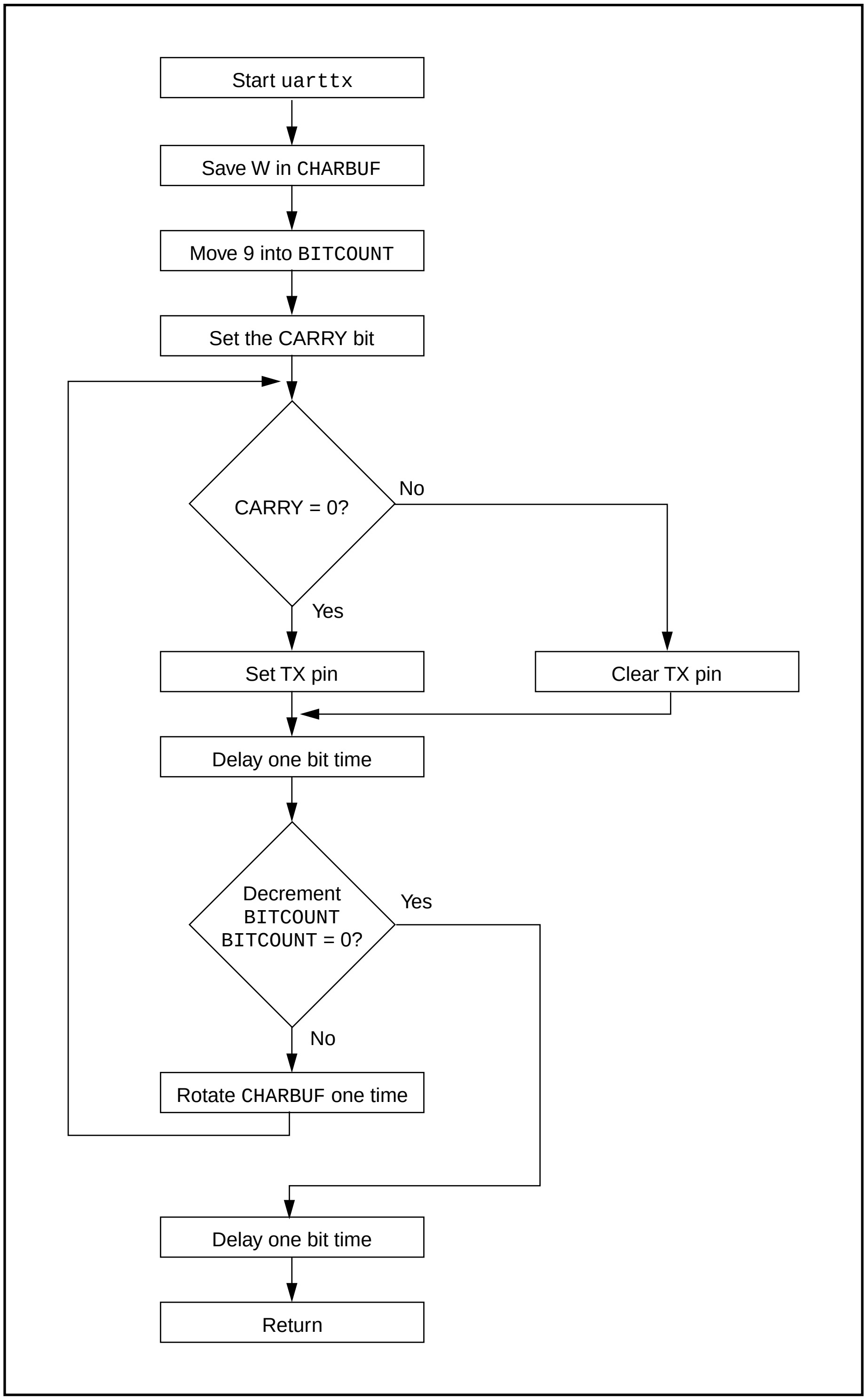


Figure 6.2 `uarttx` Routine Block Diagram

Description of `uartrx` routine:

The `uartrx` routine waits for a transistion from high to low to occur on the RA4 pin. This indicates a start bit from the Host PC. A delay of half a bit time is executed when the start bit has been detected. This ensures that the RA4 pin is sampled in the middle of each bit time. The routine now enters a loop for eight bit times. After each bit time delay, RA4 is sampled and the value is shifted into the receive buffer. Finally, after eight bits have been received, an additional bit time delay is used to account for the stop bit.

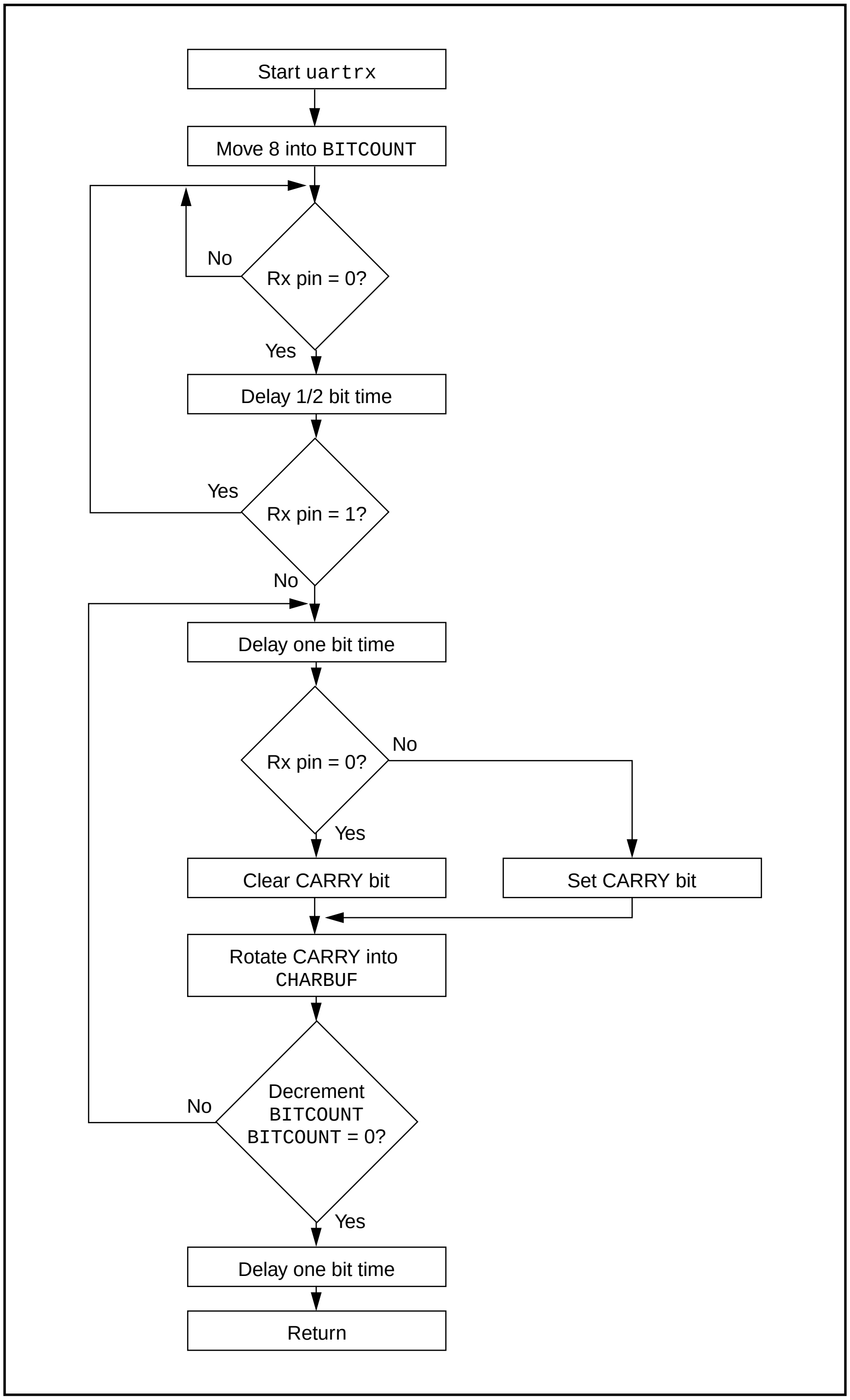


Figure 6.3 `uartrx` Routine Block Diagram

Description of `inituart` routine:

This routine simply sets up the I/O port for the transmit pin (RA3) and the receive pin (RA4).

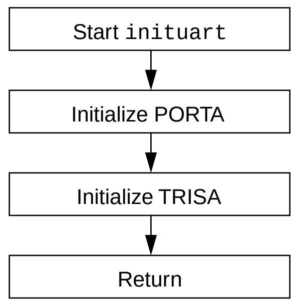


Figure 6.4 `inituart` Routine Block Diagram

Description of `delaybit` routine:

This routine provides a delay that is specified in the W register. The program puts the value into W then calls `delaybit`. This generic routine can execute both full bit time delays as well as half bit time delays.

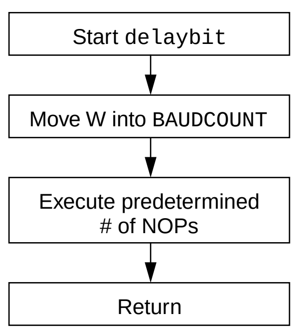


Figure 6.5 `delaybit` Routine Block Diagram

Description of `lcdinit` routine:

The function `lcdinit` initializes the LCD Module for the following:

- External resistor ladder
- Timer1 clock source
- 1/4 MUX
- Clears only the relevant LCD Data RAM locations

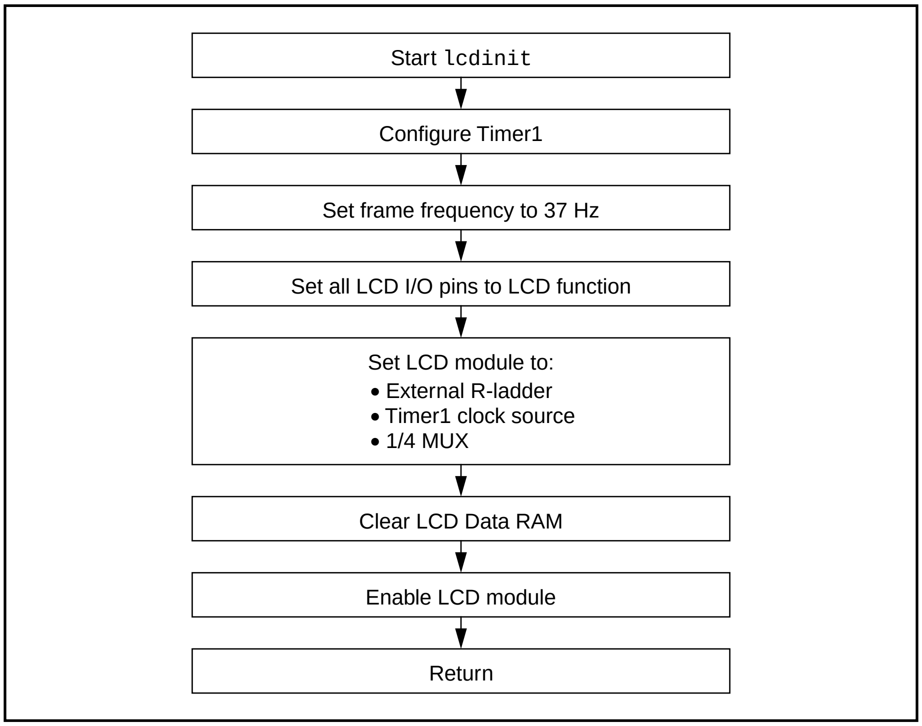


Figure 6.6 `lcdinit` Routine Block Diagram

Description of `updatelcd` routine:

The function `updatelcd` clears all the relevant LCD Data RAM locations and then, depending on the data, sets the appropriate bits to turn on pixels. The data values are used as an index into `sevensegtable` which provides the 7-segment decode value. This value is then used to turn on the specific pixels in the 7-segment number.

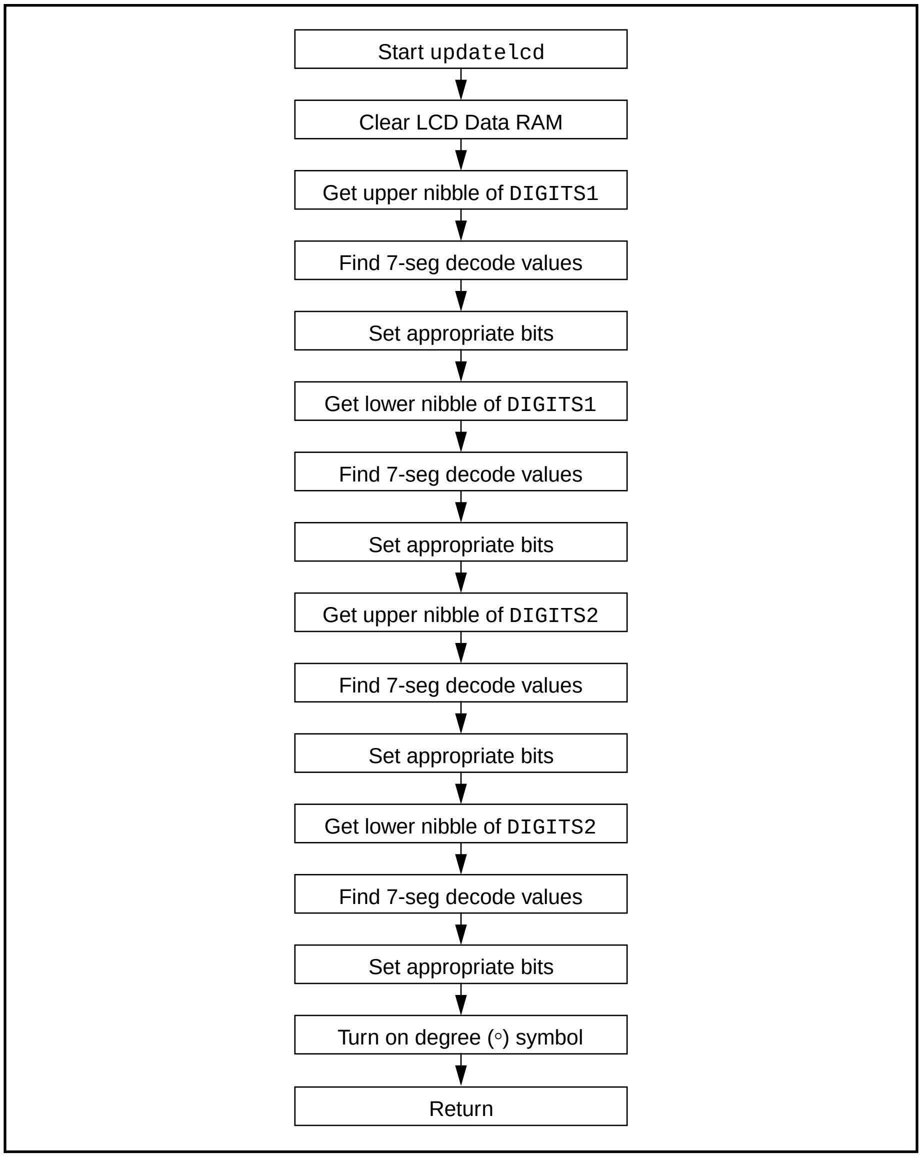


Figure 6.7 `updatelcd` Routine Block Diagram

uart.asm Source Code

```
; Filename: UART.ASM
; *********************************************************************
; * Authors:   John Day
; *            Rodger Richey
; * Revision:  1.1
; * Date       November 21, 1996
; * Part:      PIC16C923 or PIC16C924
; * Fuses:     OSC:  XT (4.0 Mhz xtal)
; *            WDT:  OFF
; * Compiled using MPASM V1.40
; *********************************************************************
; * Include files:
; *           p16c924.inc     Version 1.0
; ***********************************************************************
; * This program shows a very simple half duplex UART code example
; * uarttx - UART Transmitter code      23 Instructions, 3 Bytes RAM
; * uartrx - UART Receiver code         27 Instructions, 3 Bytes RAM
; * uartinit - UART port initialization  5 Instructions, 0 Bytes RAM
; * printstring - Print string table    12 Instructions, 1 Byte  RAM
; * printcrlf - Print carrage return     5 Instructions, 0 Byte  RAM
; *************************************************************************
             list p=16C924
             include "p16c924.inc"
             __CONFIG _CP_OFF&_WDT_OFF&_XT_OSC

; ************************************
; * Port A (RA0-RA4) bit definitions *
; ************************************
TX              EQU     3      ; RA3 is the software UART Transmit pin
RX              EQU     4      ; RA4 is the software UART Recieve pin

; ************************************
; * Misc bit defines                 *
; ************************************
LCDEN        EQU          7            ; LCDEN bit of LCDCON
W            EQU          0            ; W destination bit

; ************************************
; * Port B (RB0-RB7) bit definitions *
; ************************************
; Port B is unused in this example

; *****************************************
; * Define clock speed and baud rate here *
; *****************************************

OSCCLOCK           EQU     .4000000   ; Define external crystal frequency here
BAUDRATE           EQU     .9600      ; Define desired baud rate here

; * The following assembler calculations determine delays:
; BIT_TIME (uS) = (8 + 2 + 4 + NUM_NOPs + 3 * DELAY_VALUE) * PROC_INSTRUCT_CYCLE
;
```

```
; BIT_TIME is the time for each bit or half bit = 1/(baud rate)
; PROC_INSTRUCT_CYCLE is the processor instruction cyle = (crystal freq)/4
; DELAY_VALUE is the integer delay value that is loaded into delay counter
; NUM_NOPs are the number of NOPs that have to be added to equal bit time

OVERHEAD        EQU     .14             ; Number of instructions for routine
INSTPERBIT      EQU     ((2*OSCCLOCK/(4*BAUDRATE))+1)/2 ;Instruction/bit time
INSTPERHALFBIT  EQU     ((2*OSCCLOCK/(8*BAUDRATE))+1)/2 ;Instruction/bit time
DELAYBAUD       EQU  (INSTPERBIT-OVERHEAD)/3    ;
NUMBAUDNOPS     EQU  INSTPERBIT-(3*DELAYBAUD)-OVERHEAD
DELAYHALFBAUD   EQU  (INSTPERHALFBIT-OVERHEAD)/3        ;
NUMHALFBAUDNOPS EQU  INSTPERHALFBIT-(3*DELAYHALFBAUD)-OVERHEAD

CHARBUF         EQU     20h     ; Receive and transmit shift register
BITCOUNT        EQU     21h     ; Bit counter
BAUDCOUNT       EQU     22h     ; Bit delay loop counter
TEMP            EQU     23h     ; Temp register for pointer count
RECCHAR         EQU     24h     ; Recieved character register
; Located at 0x70-0x72 to be accessible across all banks
DIGITS2         EQU     70h     ; LCD display data
INDEX           EQU     71h     ; 7-seg table index
SEGMENT         EQU     72h     ; 7-seg table result

             ORG     0
             goto    testuart
; *******************************************************************
; * Subroutine  for delaying one bit time interval.                 *
; * This is used by both the uartsend and uartreceive routines.     *
; *******************************************************************
delaybit
             movlw   DELAYBAUD       ; Place baud delay value into W
             movwf   BAUDCOUNT       ; Move baud delay value into BAUDCOUNT register
             variable nopcount
nopcount = NUMBAUDNOPS
             WHILE nopcount > 0      ; Add correct number of NOPs
             NOP                     ; Delay one additional cycle
nopcount--
             ENDW
dlylabels
             decfsz  BAUDCOUNT,F     ; Decrement baud delay counter, skip when zero
             goto    dlylabels       ; Jump back and delay for another count cycle
             retlw   0               ; Done with delay, so return

; *******************************************************************
; * uarttx RS-232 character output routine.                         *
; * USAGE:                                                          *
; *       Place ASCII character value into W and call uarttx        *
; *       Transmits 8 bits no parity and 1 stop bit                 *
; *******************************************************************
uarttx
             movwf   CHARBUF     ; Place output char into CHARBUF reg
             movlw   09h         ; total number of bits to send
             movwf   BITCOUNT      ; move this to BITCOUNT reg
             bcf     STATUS,C      ; start by transmit the start bit
             goto    sendstart     ; Jump to send the start bit
```

```
sendbit
            rrf     CHARBUF,F     ; place next bit to transmit into carry bit
sendstart
            btfss   STATUS,C      ; Skip if next bit is zero
            bcf     PORTA,TX      ; Transmit a 1
            btfsc   STATUS,C      ; Skip if next bit is one
            bsf     PORTA,TX      ; Transmit a 0
            movlw   DELAYBAUD     ; Place baud delay one bit into W
            call    delaybit      ; delay for one bit time
            decfsz  BITCOUNT,F    ; Decrement bit counter and skip when done
            goto    sendbit       ; Jump back to put_bit to tranmit next bit
            bsf     PORTA,TX      ; send out the stop bit
            movlw   DELAYBAUD     ; Place baud delay one bit into W
            call    delaybit      ; delay for one bit time
            retlw   0             ; Done - Return to the main program

; *******************************************************************
; * uartrx RS-232 character input routine.                          *
; * USAGE:                                                          *
; *        CALL  uartrx                                             *
; *        receive byte is placed into temp                         *
; *        9600 Baud, 8 bits no parity 1 stop bit                   *
; *******************************************************************
uartrx
            movlw   08h           ; set input bit counter
            movwf   BITCOUNT      ; place bit counter into BITCOUNT
getwait
            btfsc   PORTA,RX      ; Skip when we recieve a start bit
            goto    getwait       ; go back and wait for a start bit
            movlw   DELAYHALFBAUD ; Place half bit delay time into W
            call    delaybit      ; delay for one bit time
            btfsc   PORTA,RX      ; Skip if still the start bit
            goto    getwait       ; Must be noise - go back and wait for start
getloop
            movlw   DELAYBAUD     ; Place baud delay one bit into W
            call    delaybit      ; delay for one bit time
            btfss   PORTA,RX      ; Skip if the next bit is a one
            bcf     STATUS,C      ; Clear carry bit to shift in zero
            btfsc   PORTA,RX      ; Skip if the next bit is a zero
            bsf     STATUS,C      ; Set the carry bit
            rrf     CHARBUF,F     ; Shift next received bit into temp
            decfsz  BITCOUNT,F    ; dec bit count and skip when finished
            goto    getloop       ; Go back if we still have more bits
            movlw   DELAYBAUD     ; Place baud delay one bit into W
            call    delaybit      ; delay for one bit time
            retlw   0

; *******************************************************************
; * inituart RS-232 port initialization routine                     *
; * USAGE:                                                          *
; *        CALL  inituart                                           *
; *******************************************************************
inituart
            movlw   b'00011000' ; Place 11 into RA3,2
            movwf   PORTA       ; Init PORTA output latches
```

```
            movlw   b'11110111' ; Set RA3 as an output
            bsf     STATUS,RP0
            movwf   TRISA       ; Init PORTA tris register
            bcf     STATUS,RP0
            retlw   0           ; Done, so return!

; *******************************************************************
; * testuart routine - send string of chars using uarttx            *
; *******************************************************************
testuart
            call    lcdinit     ; Initialize the LCD Module
            call    inituart    ; Set up UART ports and TRIS bits
            movlw   histring-stringbase ; Move offset addr of str in W
            call    printstring ; Print this string to the UART
            call    printcrlf   ; Send carrage return and line feed
getnextchar
            call    uartrx      ; Get a char from the terminal
            movf    CHARBUF,W   ; Place received char into W
            movwf   DIGITS2     ; Place rx char in display register
            call    uarttx      ; Send it to the terminal
            call    updatelcd   ; Update the LCD display
            goto    getnextchar ; Jump back and do it again!

; *******************************************************************
; * printcrlf routine - send carrage return and line feed           *
; *******************************************************************
printcrlf
            movlw   .13         ; Place carrage return value in W
            call    uarttx      ; Send it to the RS-232 port
            movlw   .10         ; Place Line Feed value in W
            call    uarttx      ; Send it to the RS-232 port
            retlw   0           ; Done, so return!

; *******************************************************************
; * printstring - print out a string of chars                       *
; *******************************************************************
printstring
            movwf   TEMP        ; Place string offset into temp
loopprint
            movf    TEMP,W      ; Place next char to be sent into W
            call    stringtable ; Look up the next char to send
            movwf   CHARBUF     ; Place in CHARBUF for temp storage
            xorlw   '\0'        ; Place end of string char into W
            btfsc   STATUS,Z    ; Skip if not at end of string
            retlw   0           ; End of string - done so go back!
            incf    TEMP,W      ; Point to next character
            movwf   TEMP        ; Update TEMP character pointer
            movf    CHARBUF,W   ; Place print char into W
            call    uarttx      ; Send char to the screen
            goto    loopprint   ; Loop back for the next char

stringtable
          addwf   PCL,F       ; Add W to PC for look-up table
stringbase
histring
```

```
            DT      "Greetings from PICDEM-3!\0"

;*****************************************************************************
;* Routine to initialize the LCD Module
;*****************************************************************************
lcdinit
            clrf    DIGITS2     ;Clear the LCD display register
            movlw   0x0f        ;Enable Timer1
            movwf   T1CON
            bsf     STATUS,RP1  ;Go to Bank 2
            movlw   0x06        ;Set frame freq to 37Hz
            movwf   LCDPS
            movlw   0xff        ;Enable all LCD pins as LCD drivers
            movwf   LCDSE
            movlw   0x07        ;Use ext R-ladder to for LCD voltages,
            movwf   LCDCON      ;Timer1 clock source, 1/4 MUX
            clrf    LCDD00      ;Clear the relevant LCD data RAM
            clrf    LCDD01
            clrf    LCDD04
            clrf    LCDD05
            clrf    LCDD08
            clrf    LCDD09
            clrf    LCDD12
            clrf    LCDD13
            bsf     LCDCON,LCDEN;Enable the LCD Module
            bcf     STATUS,RP1  ;Go to Bank 0
            call    updatelcd
            return

;*****************************************************************************
;* This table is used to provide the 7-segment decode values for the LCD
;* panel.
;*****************************************************************************
sevensegtable
            addwf   PCL,F       ;Add W to PC for table offset
            ;          gfedcba
            retlw   b'00111111'  ;zero
            retlw   b'00000110'  ;one
            retlw   b'01011011'  ;two
            retlw   b'01001111'  ;three
            retlw   b'01100110'  ;four
            retlw   b'01101101'  ;five
            retlw   b'01111101'  ;six
            retlw   b'00000111'  ;seven
            retlw   b'01111111'  ;eight
            retlw   b'01101111'  ;nine
            retlw   b'01110111'  ;ten
            retlw   b'01111100'  ;eleven
            retlw   b'01011000'  ;twelve
            retlw   b'01011110'  ;thirteen
            retlw   b'01111001'  ;fourteen
            retlw   b'01110001'  ;fifteen

  ;*****************************************************************************
  ;*            Routine to display results to the LCD.
```

```
;*************************************************************************
updatelcd
            bsf         STATUS,RP1    ;Go to Bank 2
            clrf        LCDD00        ;Clear only the relevant LCD RAM locations
            clrf        LCDD01
            clrf        LCDD04
            clrf        LCDD05
            clrf        LCDD08
            clrf        LCDD09
            clrf        LCDD12
            clrf        LCDD13

            movlw       0xf0          ;Use only the upper 4-bits of digits1 to find
            andwf       DIGITS2,W     ;the 7-segment decode
            movwf       INDEX
            rrf         INDEX,F
            rrf         INDEX,F
            rrf         INDEX,F
            rrf         INDEX,F
            movlw       0x0f
            andwf       INDEX,W
            call        sevensegtable
            movwf       SEGMENT
            btfsc       SEGMENT,0     ;Take the 7-segment decode value and set the
            bsf         LCDD12,7      ;appropriate bits in the LCD data RAM
            btfsc       SEGMENT,1
            bsf         LCDD12,4
            btfsc       SEGMENT,2
            bsf         LCDD04,4
            btfsc       SEGMENT,3
            bsf         LCDD00,7
            btfsc       SEGMENT,4
            bsf         LCDD04,7
            btfsc       SEGMENT,5
            bsf         LCDD13,0
            btfsc       SEGMENT,6
            bsf         LCDD08,7

            movlw       0x0f          ;Use only lower 4-bits of digits1 to find
            andwf       DIGITS2,W     ;the 7-segment decode
            call        sevensegtable
            movwf       SEGMENT
            btfsc       SEGMENT,0     ;Take the 7-segment decode value and set the
            bsf         LCDD12,6      ;appropriate bits in the LCD data RAM
            btfsc       SEGMENT,1
            bsf         LCDD08,5
            btfsc       SEGMENT,2
            bsf         LCDD04,5
            btfsc       SEGMENT,3
            bsf         LCDD00,6
            btfsc       SEGMENT,4
            bsf         LCDD04,6
            btfsc       SEGMENT,5
            bsf         LCDD08,4
            btfsc       SEGMENT,6
```

```
bsf         LCDD08,6
bcf         STATUS,RP1          ;Go to Bank 0
return

END
```

Chapter 7. test.c Description

Introduction

This chapter provides a detailed description of a test program for the PIC16C92X. It can be used for testing the PICDEM-3 board to LCD Software Demultiplexer interface. This program can be compiled under MPLAB-C or MPLAB-C demo (from Microchip Technology) or MPC (from ByteCraft Ltd.).

Highlights

This chapter covers the following topics:

- **Block Diagrams**
- **`test.c` Source Code**

Block Diagrams

This program simply fills up the LCD Data RAM with ’1’s. It is provided for testing the RS-232 link between the PICDEM-3 board and the Host PC. The program can be easily modified to have either the resistor ladder or charge pump and any of the MUX modes.

Description of Main Routine:

The main routine of `test.c` first initializes the PIC16C924. Then, using indirect addressing, it sets all bits in the LCD Data RAM one bit at a time. First the FSR is set to the address of the register LCDD00. After initialization, the program enters the `i` loop. The variable `i` is used to increment through all LCD Data RAM registers. The program now enters the `j` loop. The variable `j` is used to increment fill up the individual registers with ’1’s. Once the `j` loop finishes, the FSR is incremented to the next register, followed by a delay. Then the first bit in that register is set. After the `i` loop completes, the LCD Data RAM is cleared and the programs starts over.

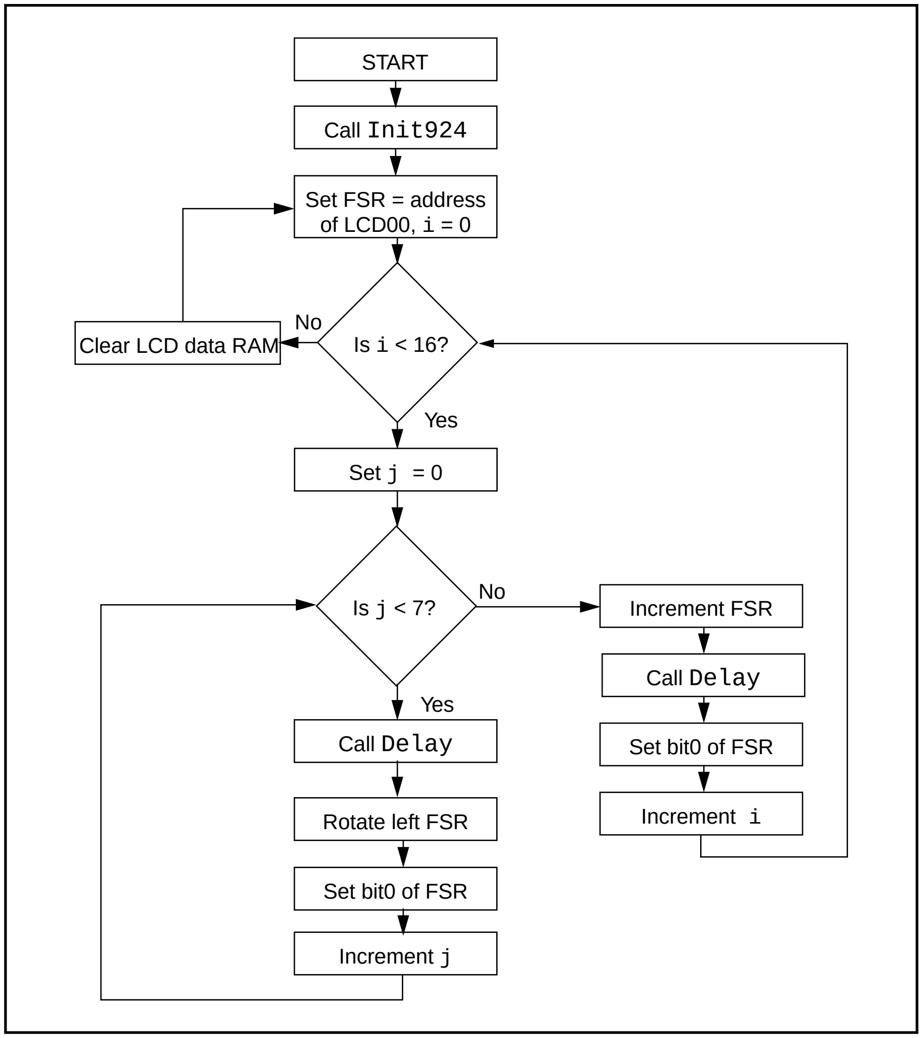


Figure 7.1 Main Routine Block Diagram

Description of `Init924` routine:

The routine `Init924` configures the LCD Module. All other peripherals on the chip are left disabled.

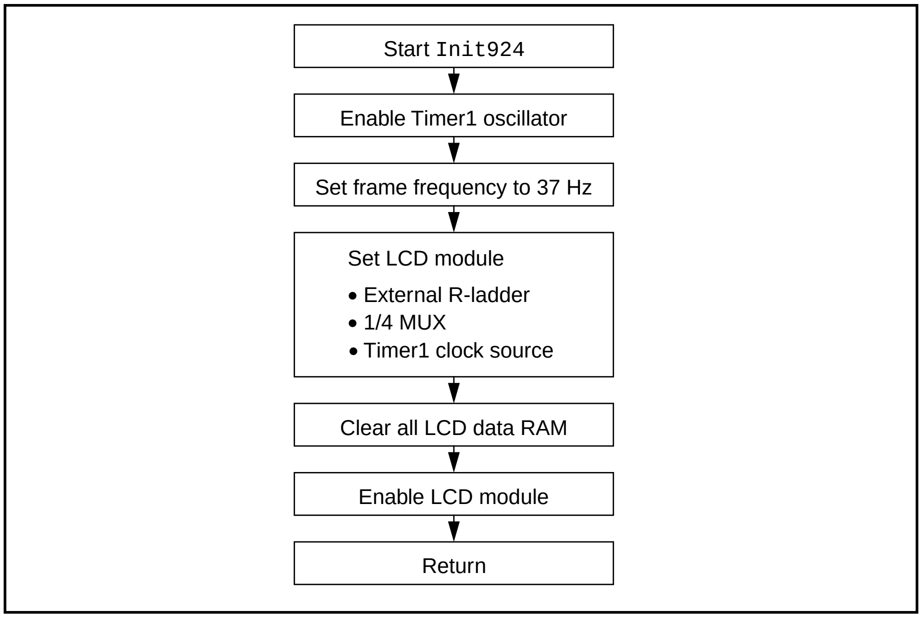


Figure 7.2 `Init924` Block Diagram

Description of Delay routine:

The routine Delay provides a 750 ms delay. The FSR register contains the address of the LCD Data RAM. Since the routine `Delay_Ms_4MHz()` uses the FSR, the FSR register is saved.

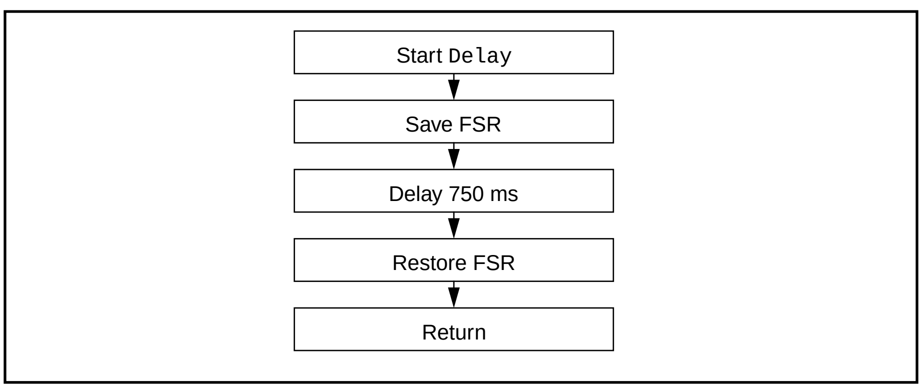


Figure 7.3 Delay Routine Block Diagram

test.c Source Code

```
/***************************************************************
*    Filename:           TEST.C
****************************************************************
*    Author:             Rodger Richey
*    Company:            Microchip Technology Incorporated
*    Revision:           1.0
*    Date:               22 November 1996
*    Compiled using MPLAB-C Version 1.10
****************************************************************
*    Include Files:
*              16C924.h Version 1.00
*              delay14.h Version 1.00
****************************************************************
*    This program first clears the LCD Data RAM
*    and then slowly fills it up with 1's.  This
*    program can be used to debug the LCD Software
*    Demultiplexer interface.
***************************************************************/
#include "16c924.h"
#include "delay14.h"

// Define temporary variables in common area of RAM
int i @ 0x70;
int j @ 0x71;
int TempFSR @ 0x72;

/***************************************************************
*    This routine initializes the LCD Module
***************************************************************/
void Init924(void)
{
     STATUS.RP0 = 0;                // Go to Bank 0
     STATUS.RP1 = 0;

     T1CON = 0b00001111;            // Enable Timer1 oscillator

     STATUS.RP1 = 1;                // Go to Bank 2
     LCDPS = 6;                     // Set the frame freq to 37Hz
     LCDSE = 0xff;                  // All LCD pins have LCD function
     LCDCON = 0b00000111;           // R-ladder,Timer1 clk, 1/4 MUX
     LCDD00 = 1;                    // Set 1st bit in LCD Data RAM
     LCDD01 = 0;                    // Clear all others
     LCDD02 = 0;
     LCDD03 = 0;
     LCDD04 = 0;
     LCDD05 = 0;
     LCDD06 = 0;
     LCDD07 = 0;
     LCDD08 = 0;
     LCDD09 = 0;
     LCDD10 = 0;
     LCDD11 = 0;
```

```
     LCDD12 = 0;
     LCDD13 = 0;
     LCDD14 = 0;
     LCDD15 = 0;
     LCDCON.LCDEN = 1;               // Enable LCD Module
     STATUS.RP1 = 0;                 // Return to Bank 0
     return;
}

/******************************************************************
*    Delay for 750mS
******************************************************************/
void Delay(void)
{
     TempFSR = FSR;                  // Save FSR
     Delay_Ms_4MHz(250);             // This routine uses FSR
     Delay_Ms_4MHz(250);
     Delay_Ms_4MHz(250);
     FSR = TempFSR;                  // Restore FSR
     return;
}

void main(void)
{
     Init924();                      // Initialize the 924

     while(1)
     {
          FSR = &LCDD00;             // Set addr to 1st byte in LCD Data RAM
          i=0;                       // Clear index
          while(i<0x10)              // Fill up all 16 LCD Data RAM registers
          {
              j=0;                   // Clear index
              while(j<0x07)          // Fill all 8-bits in register
              {
               Delay();              // Delay before each rotate
                    STATUS.IRP = 1;  // Setup indirect addressing
                    INDF<<=1;        // Rotate LCD Data register
                    INDF.0 = 1;      // Set the LSB
                    STATUS.IRP = 0;  // Restore the IRP bit
                    j++;             // Increment index
              }
              FSR++;                 // Increment the address
              Delay();               // Delay before setting LSB in next register
              STATUS.IRP = 1;        // Setup indirect addressing
              INDF.0 = 1;            // Set the LSB
              STATUS.IRP = 0;        // Restore the IRP bit
              i++;                   // increment index
          }
          Delay();                   // Delay before clearing LCD Data RAM
          STATUS.RP1 = 1;            // Go to Bank 2
          LCDD00 = 1;                // Set only the 1st bit in LCD Data RAM
          LCDD01 = 0;                // Clear all others
          LCDD02 = 0;
          LCDD03 = 0;
```

```
            LCDD04 = 0;
            LCDD05 = 0;
            LCDD06 = 0;
            LCDD07 = 0;
            LCDD08 = 0;
            LCDD09 = 0;
            LCDD10 = 0;
            LCDD11 = 0;
            LCDD12 = 0;
            LCDD13 = 0;
            LCDD14 = 0;
            LCDD15 = 0;
            STATUS.RP1 = 0;            // Return to Bank 0
        }
    }
```

Chapter 8. Hardware Description

Introduction

The hardware on the PICDEM-3 is simple and is intended to illustrate the ease of use and capabilities of the PIC16C9XX family of devices.

Highlights

This chapter covers the following topics:

- **Port Connections**
- **Switch Circuitry**
- **RS-232 Interface**
- **Timer1 Oscillator**
- **Charge Pump/Resistor Ladder**
- **Jumpers**
- **Headers**
- **LCD Panel**
- **Oscillator Configuration**

Port Connections

The following bullets list the I/O features and port connections for each processor type. Although the potentiometer R2 is connected to RA0 and thermistor RT1 is connected to RA1, the devices without A/D converters will read a digital high or low depending on the voltage level present on the I/O pins.

- RS-232 TX on RA3, RX on RA4
- Pushbutton Switch S1 on $\overline{\text{MCLR}}$ for PIC16C9XX
- Pushbutton Switch S2 on RC2
- Pushbutton Switch S3 on RA2
- Pushbutton Switch S4 on RA5
- Pushbutton Switch S6 on $\overline{\text{MCLR}}$ for PIC16C73
- Potentiometer R2 on RA0
- Thermistor RT1 on RA1
- Keypad connector on PORTB
- Timer1 Oscillator on RC0 and RC1
- LCD Panel U8 on COM0-COM3 and SEG00-SEG11
- External LCD Panel connector on COM0-COM3 and SEG00 - SEG28
- Resistor Ladder on VLCD3, VLCD2, VLCD1, and C1 or C2
- Charge Pump on VLCD3, VLCD2, VLCD1, VLCDADJ, C1 and C2

Switch Circuitry

Switches S1, S2, S3, S4, and S6 all have debounce capacitors. They also have series resistors to prevent damage to the I/O pins when changing from input to output or vice-versa.

RS-232 Interface

The PICDEM-3 board provides a RS-232 interface (U7) device to convert between RS-232 voltage levels and CMOS/TTL voltage levels. One channel is dedicated to the LCD Software Demultiplexer (See Appendix D for protocol). The other channel is connected to pins RA3 (TX) and RA4 (RX) of the PIC16C9XX devices. The DB9 connector, P1 is for use with the LCD Software Demultiplexer and P2 is for the PIC16C9XX devices.

Timer1 Oscillator

Since the LCD Module can operate from a external crystal connected to Timer1, the PICDEM-3 board provides a 32.768 kHz crystal with two 33 pF capacitors. This crystal may also be used as a real-time clock.

Charge Pump/Resistor Ladder

The PICDEM-3 board is shipped with the external resistor ladder components installed. This includes the 10k resistors R28, R29, and R30, the 5k potentiometer R31, and the jumper JP5. Also provided on the PICDEM-3 board are empty pads for the charge pump circuitry. This includes C21, C22, C23, C24, R23, and R27. If the charge pump is used, the resistor ladder components should be removed so as not to affect the performance of the charge pump. Refer to *PIC16C9XX Data Sheet*, DS30444, for detailed information. Figure 8.1 shows a suggested connections for both the resistor ladder and charge pump.

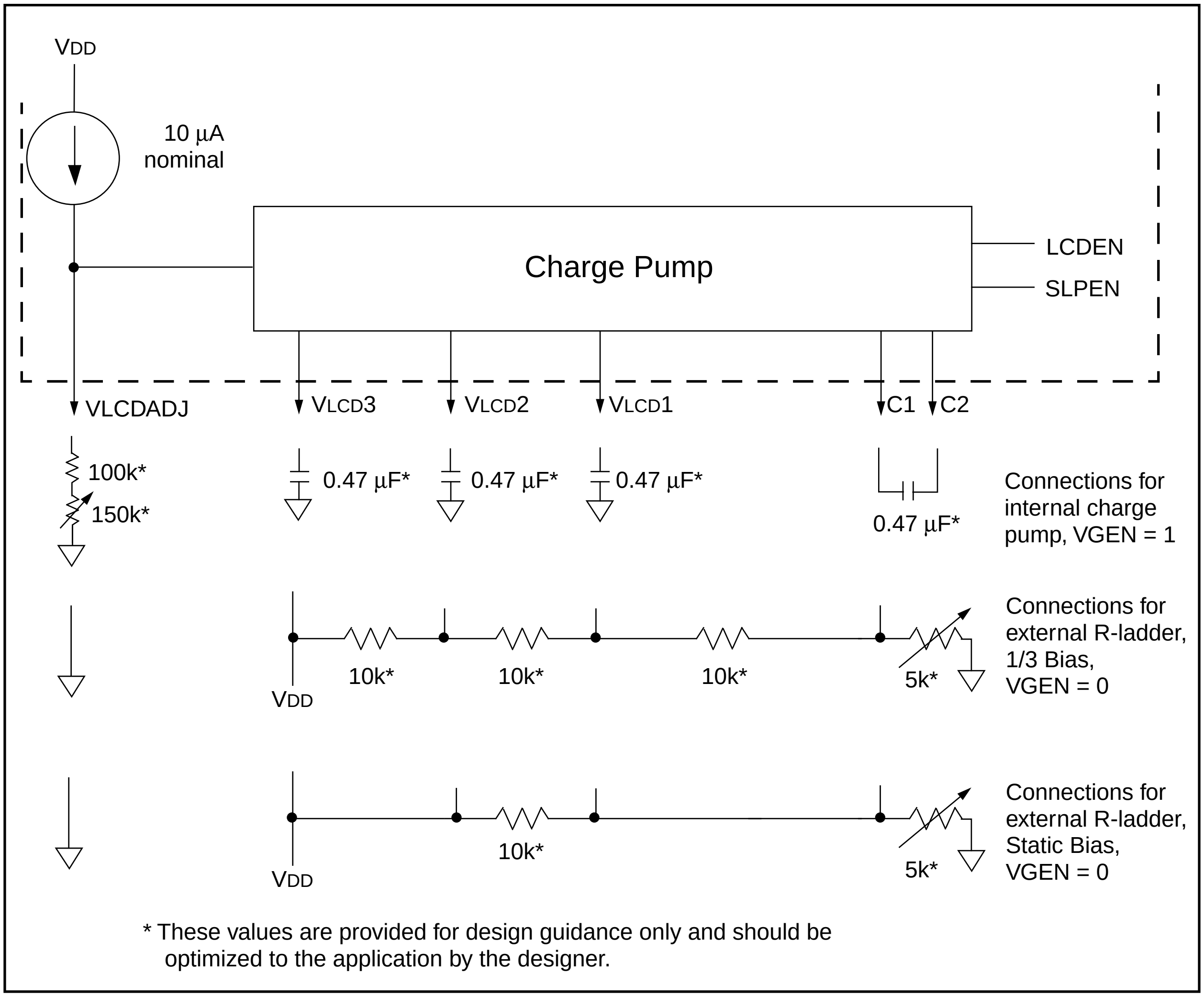


Figure 8.1 Charge Pump and Resistor Ladder Connectors

Jumpers

Two jumpers exist on the PICDEM-3 board. JP5 is used to tie the $V_{LCD}3$ pin to V_{DD}. This jumper should only be used when the PIC16C9XX is using an external resistor ladder. JP6 is used to enable the on-board RC oscillator. A 4.7k resistor (R8) and 33 pF (C27) capacitor produce ~685 kHz.

Headers

JP1 is a nine pin header which may be used as a keypad interface. Any 3x4 or 4x4 keypad may be used, such as C&K part number 4B01H322PCFQ available from Newark Electronics. Figure 8.2 shows typical connections for the keypad.

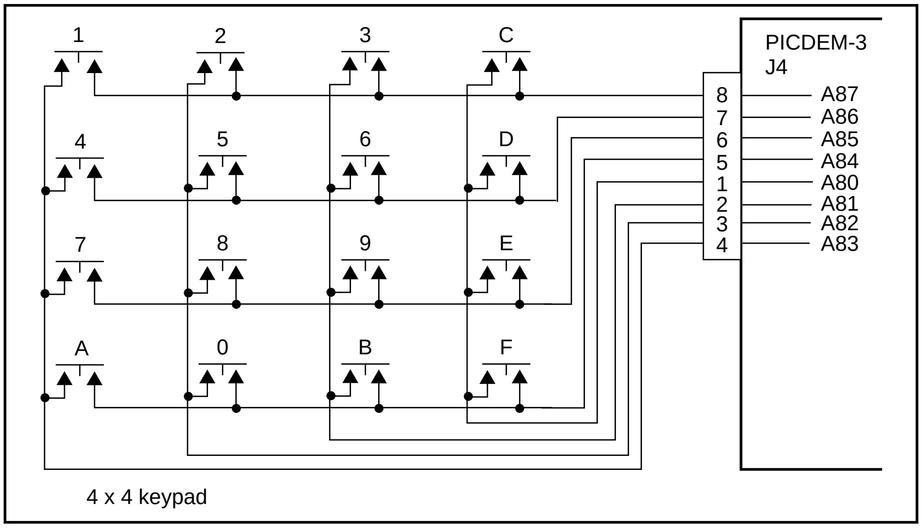


Figure 8.2 4x4 Keypad Arrangement

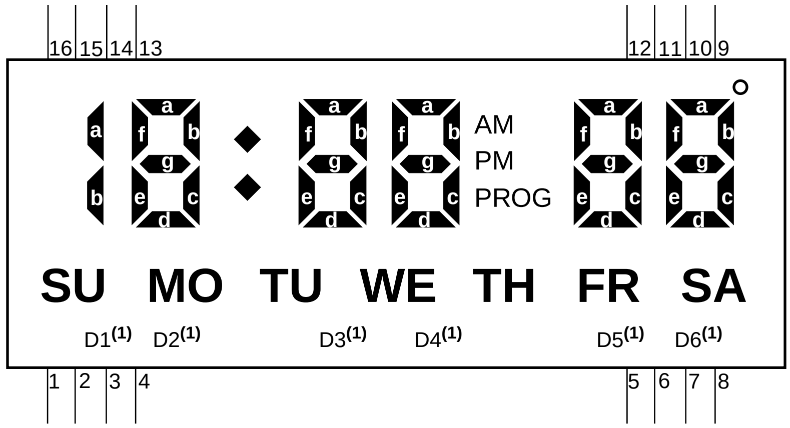
JP2 is a 2x17 pin header which can be used to connect an external LCD panel. All common and segment pins from the PIC16C9XX microcontroller are connected to this header. The PICDEM-3 schematic in Appendix A: PICDEM-3 Schematics shows the pinout for this header.

JP3 is an unpopulated header near the prototyping area. This header is provide so that any circuitry in the prototyping area can easily be connected to PORTA, PORTB, or PORTC.

JP4 is a four pin header which is used with the optional hardware demultiplexer. This header is connected to the SPI port on the PIC16C73. The header provides clock, data out, load and ground signals to the hardware demultiplexer. The pinout for JP4 is shown in the PICDEM-3 schematic in *Appendix A: PICDEM-3 Schematics* and the schematic for the hardware demultiplexer is shown in *Appendix A: Optional Hardware Demultiplexer*.

LCD Panel

The LCD panel is a custom built 1/4 MUX panel that has 12 segments. It has 7-segment numbers for time (hours and minutes) and temperature. Also provided are icons for AM, PM, PROG, day of the week, and the degrees symbol. Figure 8.3 shows the layout of the pixels on the panel and decodes the pinout into commons, segments and which signals control which pixels. The LCD panel used on the PICDEM-3 is not currently available from any LCD manufacturer.



	COM0, pin 1	COM1, pin 8	COM2, pin 9	COM3, pin 16
SEG0, pin 2	SU	D2.e	D2.f	D1.a,b
SEG1, pin 3	MO	D3.e	:	D2.b
SEG2, pin 4	TU	D3.c	D3.b	D4.f
SEG3, pin 5	WE	D4.c	D4.b	AM
SEG4, pin 6	FR	D5.c	D6.f	D5.b
SEG5, pin 7	SA	D6.c	D6.b	°
SEG6, pin 10	D6.d	D6.e	D6.g	D6.a
SEG7, pin 11	D5.d	D5.e	D5.g	D5.a
SEG8, pin 12	TH	PROG	PM	D5.f
SEG9, pin 13	D4.d	D4.e	D4.g	D4.a
SEG10, pin 14	D3.d	D3.g	D3.f	D3.a
SEG11, pin 15	D2.d	D2.c	D2.g	D2.a

(1) These are for reference only and are not part of the LCD Panel.

Figure 8.3 Pixel Layout

Oscillator Configuration

The on-board RC oscillator is enabled by installing the jumper on JP6. A 4.7 kΩ resistor (R8) and a 33 pF capacitor (C27) produce ~685 kHz. A clock oscillator may be installed into O1. Provisions have also been made for a variety of crystals and ceramic resonators. A crystal with capacitors may be installed into Y1, C27, and C28. Ceramic resonators with or without capacitors may be used in Y1. The middle ground pin for ceramic resonators with capacitors has been provided in Y1.

Appendix A. PICDEM-3 Schematics

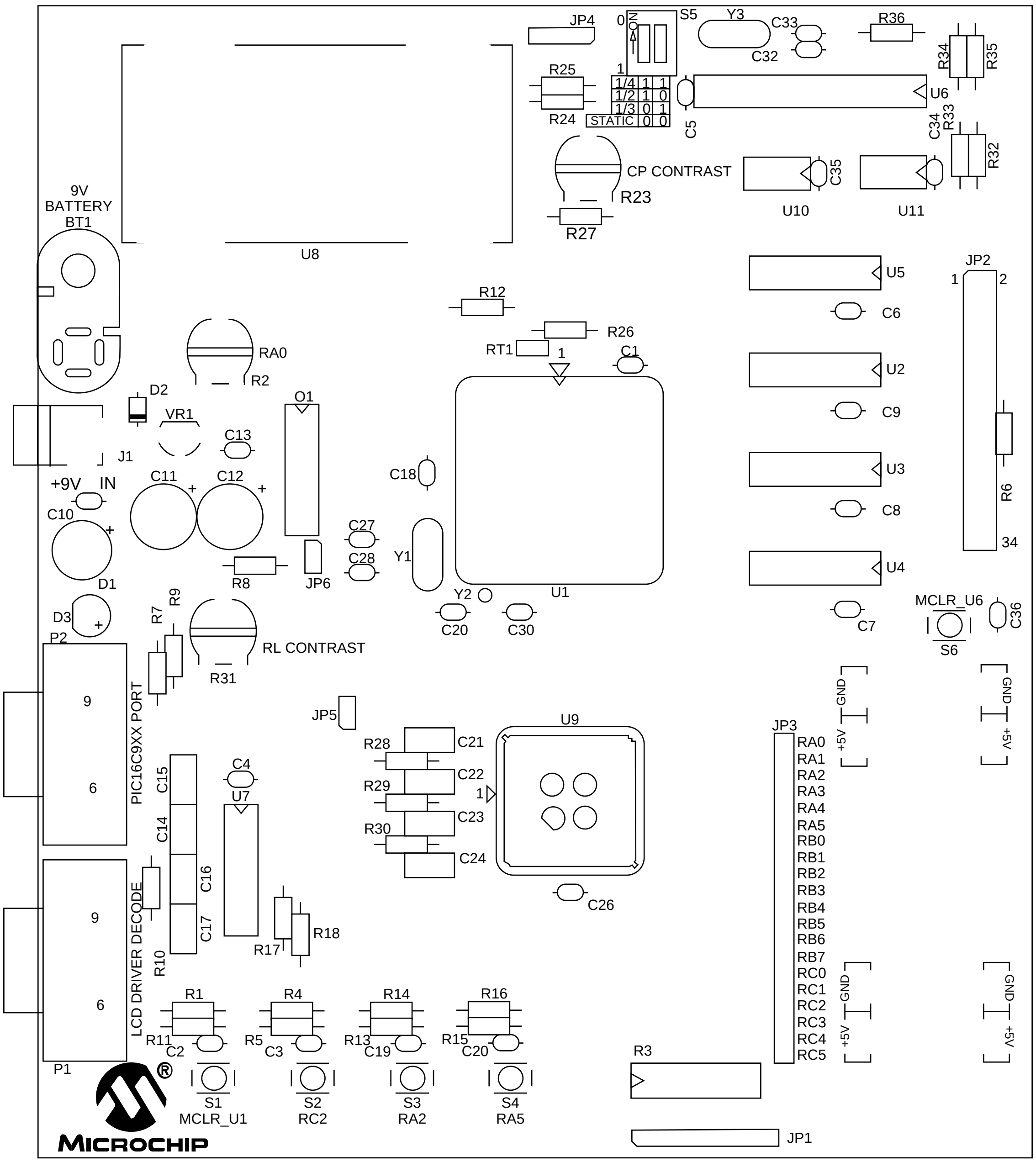


Figure A.1 PICDEM-3 Parts Layout

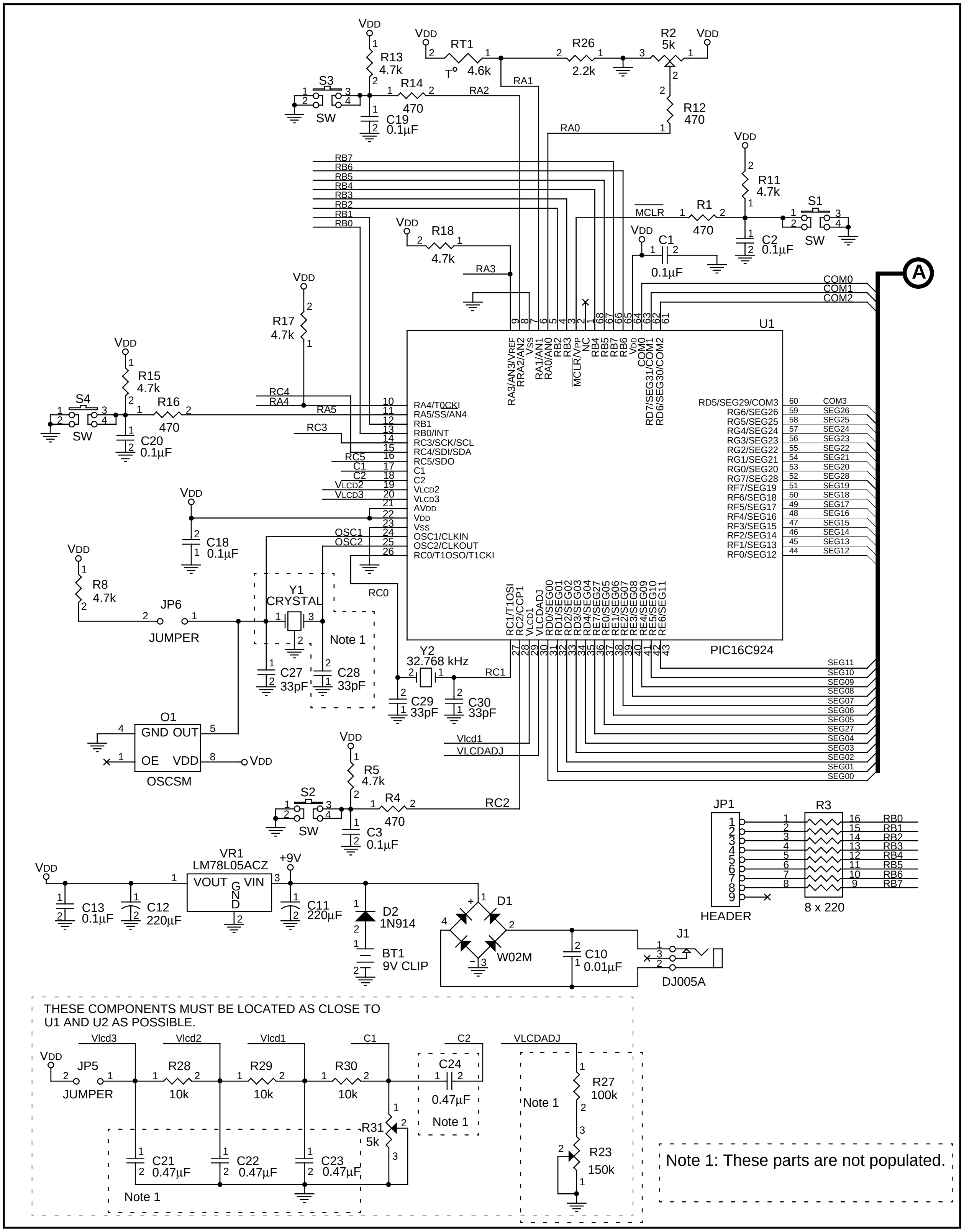


Figure A.2 PICDEM-3 Schematic

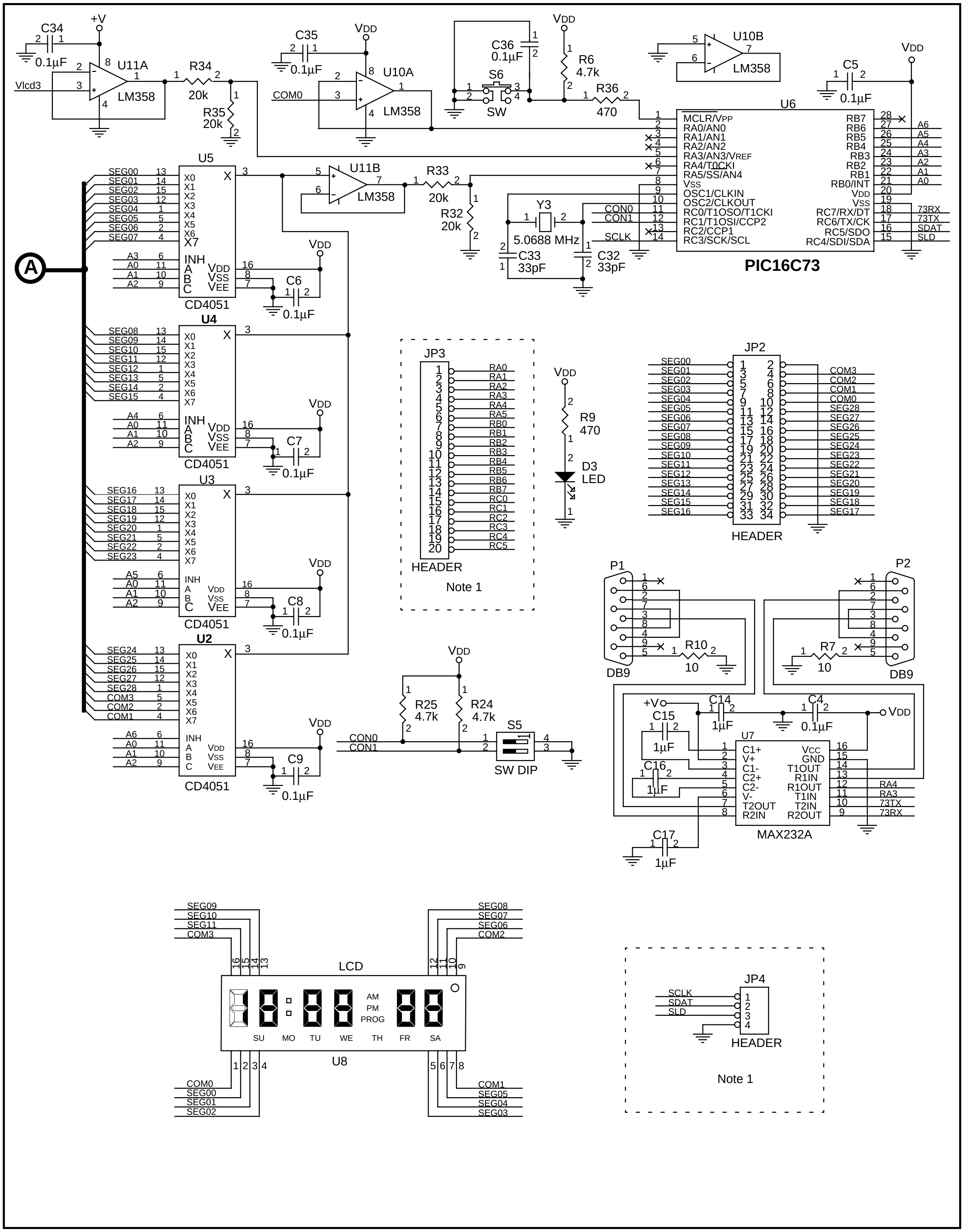


Figure A.2 PICDEM-3 Schematic (continued)

Appendix B. Optional Hardware Demultiplexer

Hardware Demultiplexer Requirements

The following schematic has been provided to help you design a hardware demultiplexer. This board can connect to JP4 on the PICDEM-3 and displays the status of individual pixels on a set of LEDs. The schematic shows only the hardware required to display one common signal. This set of components must be duplicated for each common signal that is to be displayed. The following block diagrams show which common is displayed on which set of components for each of the MUX modes.

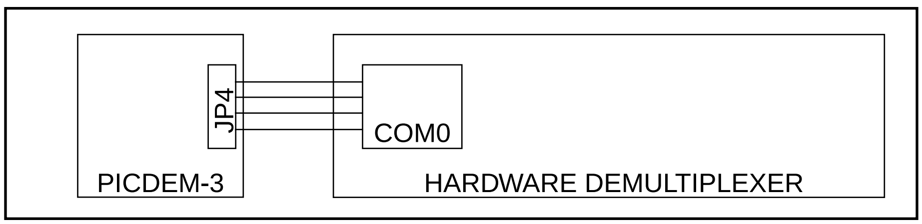


Figure B.1 Static MUX

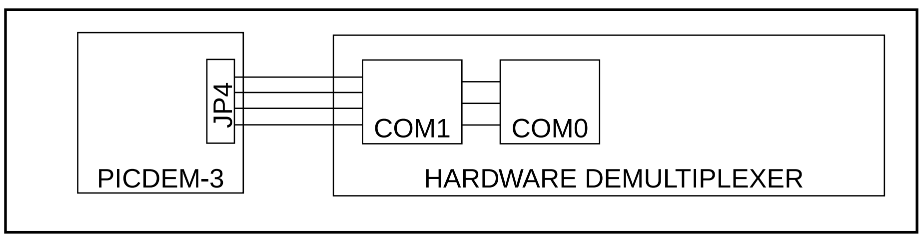


Figure B.2 1/2 MUX

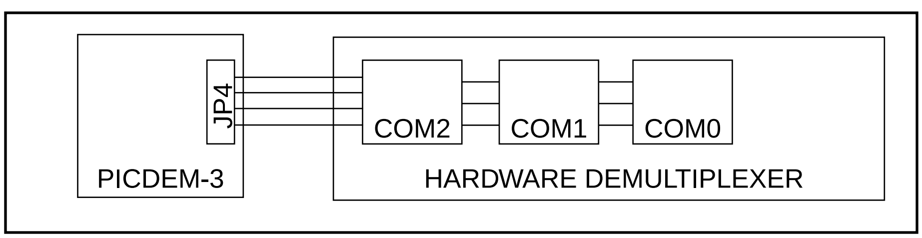


Figure B.3 1/3 MUX

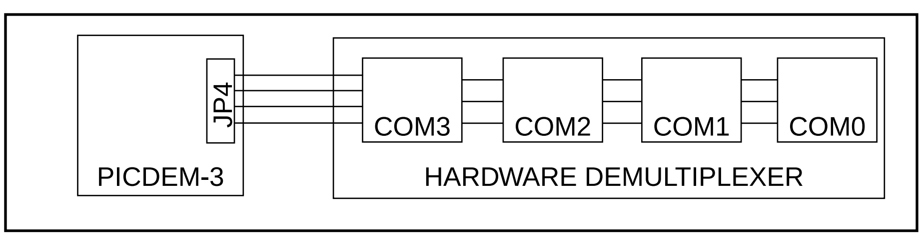


Figure B.4 1/4 MUX

The hardware demultiplexer requires a separate +5V power supply. The following currents represent the typical value of current required in each of the MUX modes. These values may vary depending on the components selected.

- Static - 190 mA
- 1/2 - 380 mA
- 1/3 - 570 mA
- 1/4 - 760 mA

This maximum is reached only when all the LEDs for that common are on.

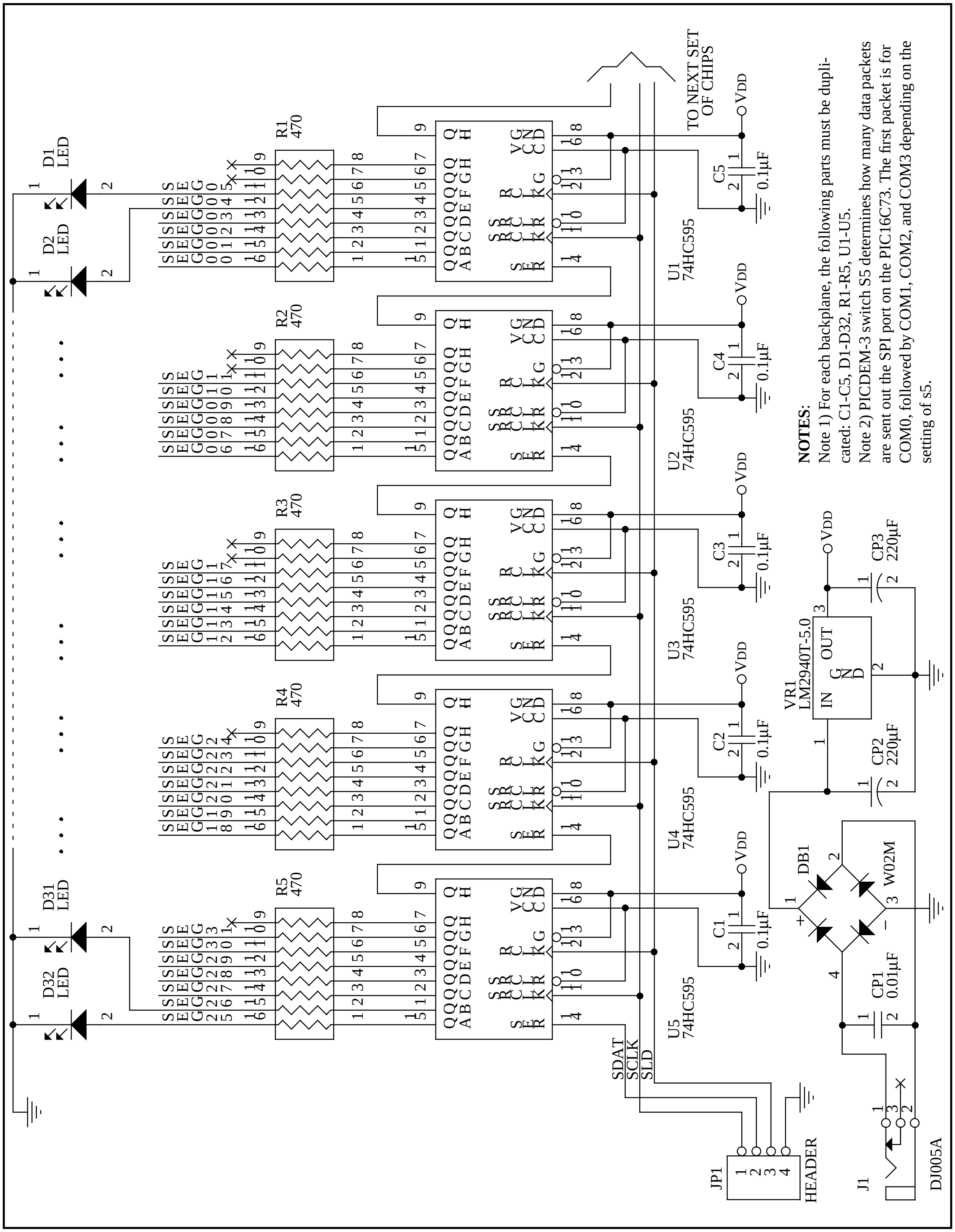


Figure B.5 Hardware Demultiplexer Schematic

Appendix C. LCD Manufacturers Listing

Manufacturers

AEG-MIS
3340 Peachtree Rd. NE Suite 500
Atlanta, GA 30326
TEL: 404-239-0277
FAX: 404-239-0383

All Shore INDS Inc.
1 Edgewater Plaza
Staten Island, NY 10305
TEL: 718-720-0018
FAX: 718-720-0225

Crystaloid
5282 Hudson Drive
Hudson, OH 44236-3769
TEL: 216-655-2429
FAX: 216-655-2176

DCI Inc.
14812 W. 117th St.
Olathe, KS 66062-9304
TEL: 913-782-5672
FAX: 913-782-5766

Excel Technology International Corporation
Unit 5, Bldg. 4, Stryker Lane
Belle Mead, NJ 08502
TEL: 908-874-4747
FAX: 908-874-3278

F-P Electronics/Mark IV Industries
6030 Ambler Drive
Mississauga, ON Canada L4W 2PI
TEL: 905-624-3020
FAX: 905-238-3141

Hunter Components
24800 Chagrin Blvd, Suite 101
Cleveland, OH 44122
TEL: 216-831-1464
FAX: 216-831-1463

Interstate Electronics Corp.
1001 E. Bull Rd.
Anaheim, CA 92805
TEL: 800-854-6979
FAX: 714-758-4111

Kent Display Systems
343 Portage Blvd.
Kent, OH 44240
TEL: 330-673-8784

LCD Planar Optics Corporation
2100-2 Artic Ave.
Bohemia, NY 11716
TEL: 516-567-4100
FAX: 516-567-8516

LXD Inc.
7650 First Place
Oakwood Village, OH 44146
TEL: 216-786-8700
FAX: 216-786-8711

Nippon Sheet Glass
Tomen America Inc.
1285 Avenue of the Americas
New York, NY 10019
TEL: 212-397-4600
FAX: 212-397-3351

OPTREX America
44160 Plymouth Oaks Blvd.
Plymouth, MI 48170
TEL: 313-416-8500
FAX: 313-416-8520

Phillips Components
LCD Business Unit
1273 Lyons Road, Bldg G
Dayton, OH 45459
TEL: 573-436-9500
FAX: 573-436-2230

Satori Electric
23717 Hawthorne Blvd. 3rd Floor
Torrance, CA 90505
TEL: 310-214-1791
FAX: 310-214-1721

Seiko Instruments USA Inc.
Electronic Components Division
2990 West Lomita Blvd.
Torrance, CA 90505
TEL: 213-517-7770
213-517-8113
FAX: 213-517-7792

Standish International
European Technical Center
Am Baümstuck II
65520 Bad Camberg/Erbach
Germany
TEL: 011 49 6434 3324
FAX: 011 49 6434 377238

Standish LCD
W7514 Highway V
Lake Mills, WI 53551
TEL: 414-648-1000
FAX: 414-648-1001

Truly Semiconductors Ltd. (USA)
2620 Concord Ave.
Suite 106
Alhambra, CA 91803
TEL: 818-284-3033
FAX: 818-284-6026

Truly Semiconductor Ltd.
2/F, Chung Shun Knitting Center
1-3 Wing Yip Street,
Kwai Chung, N.T., Hong Kong
TEL: 852 2487 9803
FAX: 852 2480 0126

Varitronix Limited Inc.
3250 Wilshire Blvd. Suite 1901
Los Angeles, CA 90010
TEL: 213-738-8700
FAX: 213-738-5340

Varitronix Limited Inc.
4/F, Liven House
61-63 King Yip Street
Kwun Tong, Kowloon
Hong Kong
TEL: 852 2389 4317
FAX: 852 2343 9555

Varitronix (France) S.A.R.L.
13/15 Chemin De Chilly
91160 Champlan
France
TEL: (33) 1 69 09 7070
FAX: (33) 1 69 09 0535

Varitronix Italia, S.R.L.
Via Bruno Buozzi 90
20099 Sesto San Giovanni
Milano, Italy
TEL: (39) 2 2622 2744
FAX: (39) 2 2622 2745

Varitronix (UK) Limited
Display House, 3 Milbanke Court
Milbanke Way, Bracknell
Berkshire RG12 1BR
United Kingdom
TEL: (44) 1344 30377
FAX (44) 1344 300099

Varitronix (Canada) Limited
18 Crown Steel Drive, Suite 101
Markham, Ontario
Canada L3R 9X8
TEL: (905) 415-0023
FAX: (905) 415-0094

Vikay America Inc.
195 W. Main St.
Avon, CT 06001-3685
TEL: 860-678-7600
FAX: 860-678-7625

DIstributors

Allied Electronics Inc.
7410 Pebble Drive
Fort Worth, TX 76118
TEL: 800-433-5700
http://www.allied.avnet.com

Digikey Corporation
701 Brooks Ave. South
Thief River Falls, MN 56701-0677
TEL: 800-344-4539
http://www.digikey.com

Newark Electronics
Administrative Office
4801 N. Ravenswood Ave.
Chicago, IL 60640-4496
TEL: 312-784-5700
FAX: 312-907-5217

Appendix D. RS-232 Communication Protocol

RS-232 Communication Protocol

The serial protocol consists of a message format for control and data bytes, and data field definitions for each command. The protocol uses a constant data stream protocol, meaning that the PIC16C73 will continually send display pixel data to the Host PC as fast as it is generated. The Host PC does not send any commands to the PIC16C73. All communciations over the RS-232 link are done at a 9600 baud rate, eight data bits, one stop bit, and no parity using only binary data (vs. ASCII-based text data). No hardware or software handshaking or acknowledgement is used.

The message format is defined as follows:

Command Byte	[Data Bytes]	. . .	Checksum Byte

Messages are made up of control and data bytes. Command bytes are signified with a '1' MSb; data bytes are signified with a '0' MSb, with the other 7 bits representing display pixels. Both the Command and Checksum bytes are both control bytes. Two control bytes are sent by themselves, Error Byte and Test Byte.

Data Header Command Byte

This control byte contains information concerning the data which follows the command byte. This byte has the following field definitions.

7	6	5	4	3	2	1	0
1	ID2	ID1	M2	M1	R2	R1	D

Bit 7: Control Byte = 1

Bit 6,5: ID2:ID1 = 00, command identification field for Data Header

Bit 4,3: M2:M1, current display MUX selection

M2	M1	Command
0	0	Static MUX
0	1	1/2 MUX
1	0	1/3 MUX
1	1	1/4 MUX

Bit 2,1: R2:R1, row (COM) select

R2	R1	Command
0	0	COM0
0	1	COM1
1	0	COM2
1	1	COM3

Bit 0: D, Device Bit

D	Device Selected
0	Future 44-pin product
1	PIC16C92X

Data Bytes

Data bytes contain a leading 0 followed by 7 bits of binary data. Each data bit represents a specific segment and common. Each common consists of a possible 32 segments, or pixels, so a total of five data bytes are used to transmit each common of data. These bytes have the following field definitions, shown in the order in which they are transmitted (Data Byte 0 is first, Data Byte 4 is last).

Note: For all data bytes, bit 7 is a 0 to indicate a Data Byte, Rsv bits are reserved (unused).

	7	6	5	4	3	2	1	0
Data Byte 0:	0	Rsv	S5	S4	S3	S2	S1	S0
Data Byte 1:	0	Rsv	S11	S10	S9	S8	S7	S6
Data Byte 2:	0	Rsv	S17	S16	S15	S14	S13	S12
Data Byte 3:	0	S24	S23	S22	S21	S20	S19	S18
Data Byte 4:	0	S31	S30	S29	S28	S27	S26	S25

Error Byte

This control byte contains error information. This byte is not sent with any data or checksum bytes (i.e. it is sent by itself). It is automatically sent by the PIC16C73 whenever an error occurs, possibly even in the middle of a message transfer. This control byte has the following field definitions:

7	6	5	4	3	2	1	0
1	ID2	ID1	rsv	rsv	ste	rxe	txe

Bit 7: Control Byte = 1

Bit 6,5: ID2:ID1 = 01, command identification field for Error Byte

Bits 4,3: rsv, reserved (unused)

Bit 2: ste, Self Test error = 1 if an error occurred in self test, = 0 if no error occurred.

Bit 1: rxe, Receive error = 1 if an error occurred in a serial data reception, = 0 if no error occurred.

Bit 0: txe, Transmit error = 1 if an error (overflow) occurred in a serial data transmission, = 0 if no error occurred.

Test Byte

This control byte is used for production test. This byte is not sent with any data or checksum bytes (i.e. it is sent by itself). It is automatically sent by the PIC16C73 whenever it receives a test command, possibly even in the middle of a message transfer. This control byte has the following field definitions:

7	6	5	4	3	2	1	0
1	ID2	ID1	rsv	rsv	rsv	rsv	rsv

Bit 7: Control Byte = 1

Bit 6,5: ID2:ID1 = 10, command identification field for Test Byte

Bits 4-0: rsv, reserved (unused)

Checksum Byte

This control byte contains a five-bit checksum. The checksum byte follows the five data bytes. This byte has the following field definitions:

7	6	5	4	3	2	1	0
1	ID2	ID1	C5	C4	C3	C2	C1

Bit 7: Control Byte = 1

Bit 6,5: ID2:ID1 = 11, command identification field for Checksum Byte

Bit 4-0: C5:C1, five-bit checksum field

C5:C1 represents the lower five bits of the inverted (one's complement) sum of all bytes of the current message, including both control and data bytes.

Appendix E. On Line Support

Introduction

Microchip provides two methods of on-line support. These are the Microchip BBS and the Microchip World Wide Web (WWW) site.

Use Microchip's Bulletin Board Service (BBS) to get current information and help about Microchip products. Microchip provides the BBS communication channel for you to use in extending your technical staff with microcontroller and memory experts.

To provide you with the most responsive service possible, the Microchip systems team monitors the BBS, posts the latest component data and software tool updates, provides technical help and embedded systems insights, and discusses how Microchip products provide project solutions.

The web site, like the BBS, is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape or Microsoft Explorer. Files are also available for FTP download from our FTP site.

Connecting to the Microchip Internet Web Site

The Microchip web site is available by using your favorite Internet browser to attach to:

www.microchip.com

The file transfer site is available by using an FTP service to connect to:

ftp://ftp.futureone.com/pub/microchip

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Datasheets, Application Notes, User's Guides, Articles and Sample Programs.

A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products

Connecting to the Microchip BBS

Connect worldwide to the Microchip BBS using either the Internet or the CompuServe® communications network.

Internet: You can telnet or ftp to the Microchip BBS at the address
mchipbbs.microchip.com

CompuServe Communications Network: In most cases, a local call is your only expense. The Microchip BBS connection does not use CompuServe membership services, therefore

You do not need CompuServe membership to join Microchip's BBS.

There is **no charge** for connecting to the BBS, except for a toll charge to the CompuServe access number, where applicable. You do not need to be a CompuServe member to take advantage of this connection (you never actually log in to CompuServe).

The procedure to connect will vary slightly from country to country. Please check with your local CompuServe agent for details if you have a problem. CompuServe service allow multiple users at baud rates up to 14400 bps.

The following connect procedure applies in most locations.

1. Set your modem to 8-bit, No parity, and One stop (8N1). This is not the normal CompuServe setting which is 7E1.
2. Dial your local CompuServe access number.
3. Depress **<Enter↵>** and a garbage string will appear because CompuServe is expecting a 7E1 setting.
4. Type +, depress **<Enter↵>** and Host Name: will appear.
5. Type **MCHIPBBS,** depress **<Enter↵>** and you will be connected to the Microchip BBS.

In the United States, to find the CompuServe phone number closest to you, set your modem to 7E1 and dial (800) 848-4480 for 300-2400 baud or (800) 331-7166 for 9600-14400 baud connection. After the system responds with `Host Name:`, type **NETWORK,** depress **<Enter↵>** and follow CompuServe's directions.

For voice information (or calling from overseas), you may call (614) 723-1550 for your local CompuServe number.

Using the Bulletin Board

The bulletin board is a multifaceted tool. It can provide you with information on a number of different topics.

- Special Interest Groups
- Files
- Mail
- Bug Lists

Special Interest Groups

Special Interest Groups, or SIGs as they are commonly referred to, provide you with the opportunity to discuss issues and topics of interest with others that share your interest or questions. SIGs may provide you with information not available by any other method because of the broad background of the PIC16/17 user community.

There are SIGs for most Microchip systems and device families. These groups are monitored by the Microchip staff.

Files

Microchip regularly uses the Microchip BBS to distribute technical information, application notes, source code, errata sheets, bug reports, and interim patches for Microchip systems software products. Users can contribute files for distribution on the BBS. For each SIG, a moderator monitors, scans, and approves or disapproves files submitted to the SIG. No executable files are accepted from the user community in general to limit the spread of computer viruses.

Mail

The BBS can be used to distribute mail to other users of the service. This is one way to get answers to your questions and problems from the Microchip staff, as well as keeping in touch with fellow Microchip users worldwide.

Consider mailing the moderator of your SIG, or the SYSOP, if you have ideas or questions about Microchip products, or the operation of the BBS.

Software Releases

Software products released by Microchip are referred to by version numbers. Version numbers use the form:

`xx.yy.zz`

Where `xx` is the major release number, `yy` is the minor number, and `zz` is the intermediate number.

Intermediate Release

Intermediate released software represents changes to a released software system and is designated as such by adding an intermediate number to the version number. Intermediate changes are represented by:

- Bug Fixes
- Special Releases
- Feature Experiments

Intermediate released software does not represent our most tested and stable software. Typically, it will not have been subject to a thorough and rigorous test suite, unlike production released versions. Therefore, users should use these versions with care, and only in cases where the features provided by an intermediate release are required.

Intermediate releases are primarily available through the BBS.

Production Release

Production released software is software shipped with tool products. Example products are PRO MATE, PICSTART, and PICMASTER. The Major number is advanced when significant feature enhancements are made to the product. The minor version number is advanced for maintenance fixes and minor enhancements. Production released software represents Microchip's most stable and thoroughly tested software.

There will always be a period of time when the Production Released software is not reflected by products being shipped until stocks are rotated. You should always check the BBS or the WWW for the current production release.

Systems Information and Upgrade Hot Line

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive any currently available upgrade kits. The Hot Line Numbers are: 1-800-755-2345 for U.S. and most of Canada, and 1-602-786-7302 for the rest of the world.

These phone numbers are also listed on the "Important Information" sheet that is shipped with all development systems. The hot line message is updated whenever a new software version is added to the Microchip BBS, or when a new upgrade kit becomes available.

Notes:

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

Atlanta
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Consulting (Shanghai) Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu
Microchip Technology Consulting (Shanghai) Co., Ltd., Chengdu Liaison Office
Rm. 2401, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-6766200 Fax: 86-28-6766599

China - Fuzhou
Microchip Technology Consulting (Shanghai) Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

China - Shanghai
Microchip Technology Consulting (Shanghai) Co., Ltd.
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

China - Shenzhen
Microchip Technology Consulting (Shanghai) Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-2350361 Fax: 86-755-2366086

Hong Kong
Microchip Technology Hongkong Ltd.
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Microchip Technology Japan K.K.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

Taiwan
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark
Microchip Technology Nordic ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Microchip Technology GmbH
Gustav-Heinemann Ring 125
D-81739 Munich, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy
Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

03/01/02