Final Project Report
E3390 Electronic Circuit Design Lab

Electronic Notepad

Keith Dronson
Sam Subbarao

Submitted in partial fulfillment of the requirements for the
Bachelor of Science Degree

May 10, 2008

Department of Electrical Engineering
Columbia University

Table of Contents

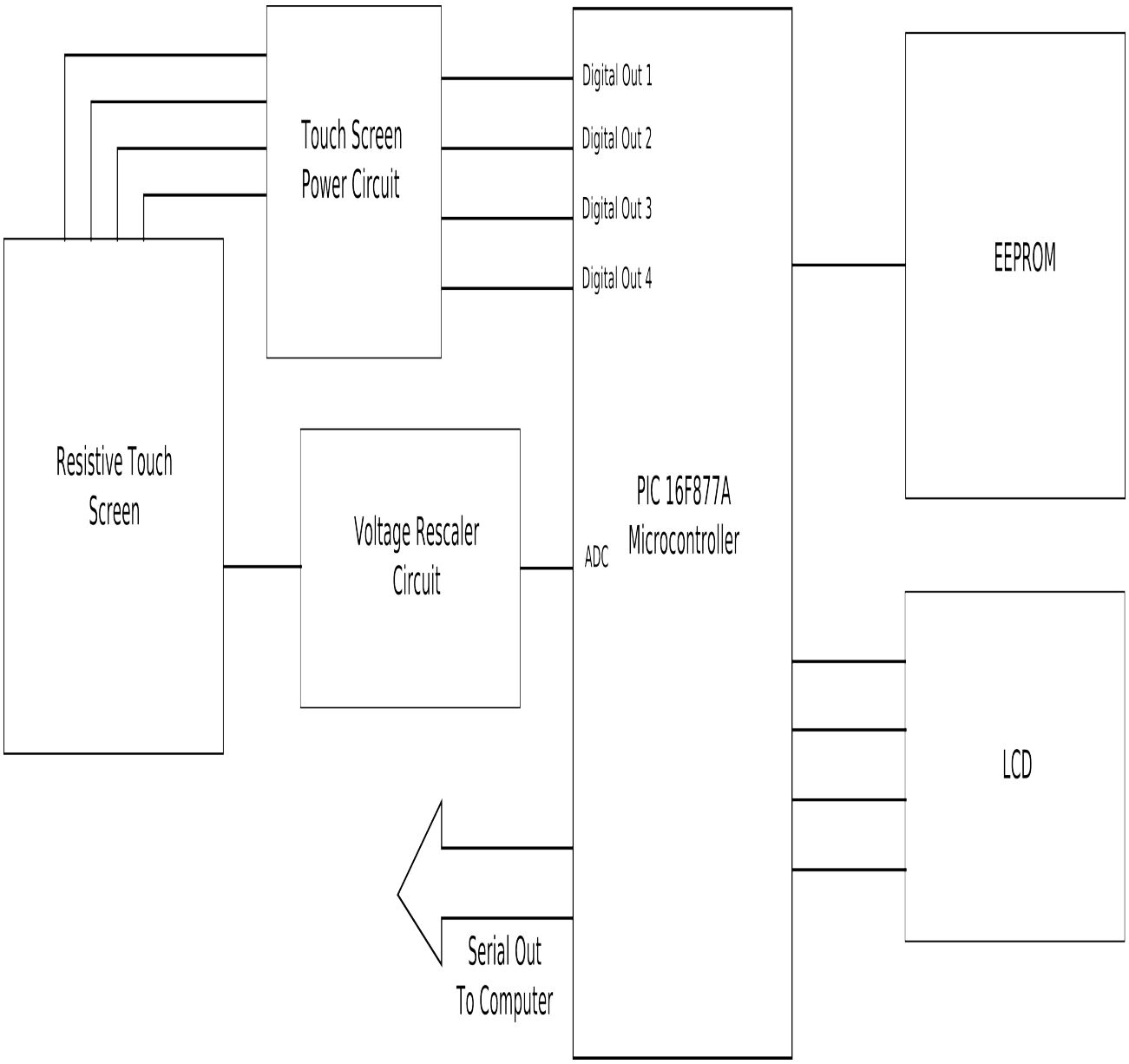
I. Executive Summary

The goal of our project was to design and build a portable electronic notepad. The notepad would have a touchscreen for writing on and memory to store your drawings and notes in. At your convenience you should be able to transfer the data to a computer where a software program can read it and then display it on the screen for you to see. The notepad should have a fast sampling rate and high accuracy so your images will appear as you drew them.

This is achieved by using a micro-controller to read X and Y coordinates off of a resistive touchscreen. These X,Y coordinates correspond to the point that you are poking the screen. These values are then stored locally on a *SD card. When you flip a switch the micro-controller transfers the data from the SD card to the computer via a serial connection. This data is read in on the computer and then displayed using software that is provided with the notepad.

*At this point the SD card interface is only implemented in hardware. The software interface to the SD card is still under development.

II. Block Diagram, Design Targets and Specifications



Specifications and Design Targets

Device implements a PIC16F877/A Microctroller for A/D conversion and interfacing. A SD card is used for external memory. This must have a capacity of at least 1MB to store a single page. The LCD screen is used to display real time information to the user. The LCD uses the standard HD44780 controller.

III. Individual Block Descriptions

(a) PIC16F877A Microcontroller

The MCU provides an interface to the touchscreen, LCD, SD card and serial communication to a computer. The MCU is able to create a conductive path to ground in both the X and Y planes of the touchscreen by driving two of the four pins connected to the resistive touchscreen high and pulling the other two to ground. The voltage levels are then sampled and the X and Y coordinates are determined from these measurements.

(b) Resistive touchscreen

The touchscreen uses a five-wire interface for determining X and Y coordinates. Four of the five wires correspond to the four corners of the screen. The fifth wire provides the voltage level at the point of contact. By setting the two top corner points to Vdd and the bottom two to ground current will flow from top to bottom. Pressing a point on the screen will connect the top conductive plane of the touchscreen with the bottom layer at that point. By measuring the voltage at this bottom layer we can determine the Y coordinate. The X coordinate is determined the same way except the two left most points are set to Vdd and the two right most points to ground. By switching back and forth between X and Y it is possible to recreate images that are traced on the screen.

(c) Touchscreen Power Circuit

The PIC16F877A is not able to supply the current needed for the touchscreen. Therefore power opamps are used to buffer the output of the MCU and supply the power to the touchscreen.

(d) Voltage Rescaler Circuit

The voltage rescaler circuit is required in order to realize the full 10-bit range of the A/D. Without this circuit the voltage measured from the screen is between ~1-4V. This reduces our resolution by almost a bit. In order to fix this the rescaler circuit subtracts off a voltage in order to zero the lower part of the range. Then the resulting voltage is gained up in order to spread the resolution from 0 to 5V thus giving us the full 10-bits of resolution.

(e) LCD

The LCD is used to display data to the user. The LCD has a 2x8 character display and can be programmed to display almost any kind of data to the user. Currently the X and Y coordinates measured are displayed in real time to the user.

(f) EEPROM - SD card

The SD card is used to store pages of data before transferring them to a computer. Currently this is only implemented in hardware. However the algorithm for storing data to the SD card has been determined and an appropriate file system developed. The SD card is block addressed with each block containing 512-bytes of data. Due to memory limitations of the PIC only the first 100-bytes of each block will contain data for a specific page. For a 1 mega-pixel resolution this will require that 1250 blocks be used for each page. However using this method allows us to overlap pages by adding an offset of 100 bytes to each consecutive page. This way up to 5 pages can be stores in each 1250 blocks of data. When the MCU reads a X,Y pair from the touchscreen the PIC will determine which block the point is in. After which the 100-bytes will be read in from the SD card and the pixel corresponding to the X,Y coordinate will be set to a 1.

(g) Serial Communication to Computer

The PIC is setup to transfer data to the computer through the serial port when a switch is flipped. The data is sent at a rate of 9600 baud using the RS-232 protocol.

IV. Individual Bill of Materials

PART	MANUFACTURER	#	COST
PIC16F877A	Microchip Technology	1	8.5
LF411 OPAMP	National Semiconductor	4	1.63
MCP601 OPAMP	Microchip Technology	1	.65
CPDR07086 Dual-power OPAMP	ON Semiconductor	2	1.25
20MHz Oscillator	Abracon	1	1.30
SD Card Slot	3M	1	1.33
SD Card	Transcend	1	12.25
5V Regulator	STMicroelectronics	1	1.35
3.3V Regulator	STMicroelectronics	1	.68
LCD Screen	CrystalFontz	1	15.70
Touchscreen	Bergquist	1	55.00
PCB board	Sunstone Circuits	1	50.00
Total Cost			149.64

V. Health, Safety and Environmental Issues

a. Product Dangers

This product does not pose any danger when used properly.

b. Health Hazards

There are no known health hazards associated with this product.

c. Environmental Hazards

i. This product complies with all FCC standards.
ii. The AC adapter should not be disassembled or tampered with. The operating voltage for this device is 12V which does not pose any risk of electric shock,

VI. Final Gantt Chart

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	
Project Brainstorming	■	■											K/S
Paper Design			■										K/S
Touchscreen Controller				■	■								K
Rescaler Circuit					■	■							S
SD Card Interface							■	■	■	■	■	■	K
Serial Interface								■	■				S
Display Software								■	■	■			K/S
Board Design									■	■			S
Final Packaging										■	■	■	K/S

VII. Course Suggestions

I think the course ran pretty smoothly. I definitely agree that two semesters would be beneficial. One semester isn't enough time to develop a product from the ground up it seems, especially when you don't have a project that you want to work on. If it is only one semester perhaps people would have more luck if they were assigned a project, maybe each team work on a single part of a very large design. This way people would know what they had to do from the start. Other than that I felt that I learned a lot and got to experience what it like to design a product.

Appendix A

-Display Software

```
import java.awt.*;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JTextField;
import javax.swing.JLabel;
import java.io.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;

public class ImageConverter {

		public static String fileName;
		public static JTextField jtf;


	protected static class DrawArea extends Panel {
	/** * Does the actual drawing. */
		private static final int DIM = 2;
		private static final int XRES = 1000;
		private static final int YRES = 1000;
		private static final int PIXELWIDTH = 2;



		public void paint(Graphics g) {
		try
		{
		// You add/change the statements here to draw
		// the picture you want.
		//	g.drawLine(0,0,300,300);
		//Open the file and initialize buffered reader
			//fileName = "art1.TXT";
			FileInputStream fstream = new FileInputStream(fileName);
			DataInputStream in = new DataInputStream(fstream);
```

```
                BufferedReader dataReader = new BufferedReader(new InputStreamReader(in));

                String dataLine;
                String[] coordinate = new String[DIM];
                int x,y;

                g.setColor(new Color(0, 0, 0));

                while((dataLine=dataReader.readLine()) != null)
                {
                        coordinate=dataLine.split(" ",DIM);
                        x = 1000-Integer.parseInt(coordinate[0]);
                        y = Integer.parseInt(coordinate[1]);
                        g.fillRect(2*x/3, 2*y/3, PIXELWIDTH, PIXELWIDTH);
                        System.out.println("X coordinate: " + x +" Y coordinate: " + y);
                }
        }
        catch(Exception e)
        {
                System.err.println("Error: " +e.getMessage());
        }
        }

        /** * Makes sure that
         * the window drawing area ends up being the
         * right size. You don't need to change this.
         */
        public Dimension getPreferredSize() {
                return new Dimension(WIDTH,HEIGHT) ;
        }
        // These set the size of the drawing area.
        // Change the sizes to suit what you need.
        private int WIDTH = 1000 ;
        private int HEIGHT = 1000 ;
    }


public static void addComponentsToPane(Container pane) {


   pane.setLayout(new GridBagLayout());
```

```
    GridBagConstraints gBC = new GridBagConstraints();
    gBC.fill = GridBagConstraints.HORIZONTAL;

        DrawArea drawing = new DrawArea() ;
        gBC.gridwidth = 3;
        gBC.gridx = 0;
        gBC.gridy = 0;
        pane.add(drawing, gBC);

}



 private static void createAndShowGUI() {

    JFrame.setDefaultLookAndFeelDecorated(true);
    JFrame frame = new JFrame("Image Converter");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    //Set up the content pane.
    addComponentsToPane(frame.getContentPane());

    frame.pack();
    frame.setVisible(true);


 }

 public static void main(String[] args) {

        try {
                fileName = args[0];
        }
        catch(ArrayIndexOutOfBoundsException e) {
                System.err.println(e.getMessage());
        }

    javax.swing.SwingUtilities.invokeLater(new Runnable() {
        public void run() {
           createAndShowGUI();
        }
```

```
      });
   }
}
```

-MCU Firmware

```
'*****************************************************************
'*  Name    : Pad.bas                                            *
'*  Author  : Keith                 *
'*  Notice  : Copyright (c) 2008 [select VIEW...EDITOR OPTIONS] *
'*          : All Rights Reserved                                *
'*  Date    : 1/24/2008                                          *
'*  Version : 1.0                                                *
'*  Notes   :                                                    *
'*          :                                                    *
'*****************************************************************
 Include "modedefs.bas"


 DEFINE OSC 20              'Define 20MHZ clock frequency
 DEFINE LCD_DREG PORTD      'Set LCD data port to port D
 DEFINE LCD_DBIT 4          'Set starting data bit on 4
 Define LCD_RSREG PORTE     'Set Register select port to E
 DEFINE LCD_RSBIT 0         'Set starting data bit on 0
 Define LCD_EREG  PORTE     'Set Enable port
 DEFINE LCD_EBIT  1         'Set enable bit on 1
 DEFINE LCD_BITS  4
 DEFINE LCD_LINES 2
 DEFINe ADC_BITS 10         'Set number of bits in A/D conversion
 DEFINE ADC_SAMPLEUS 50     'Set sample time in uS
 TRISA = %11111111          'Set PortA to all input
 TRISC.0 = 0
 TRISC.1 = 1
 CMCON = 7
 ADCON1 = %10000100         'AN3 and AN2 are Vref+ and Vref- respectively
 ADCON0 = %01000001
 TRISA.0 = 1
 TRISD = 0
 TRISB = 0
 TRISE.0 = 0
```

```
TRISE.1 = 0
Result_X Var WORD
Result_Y Var Word
Last_X VAR WORD
Last_Y Var WORD
CNT VAR BYTE
TSTART VAR PORTC.1
Up_Left VAr PORTB.0       'Upper Left of screen
Up_Right Var PORTB.1      'Upper Right
Lo_Right VAR PORTB.2      'Lower Right
Lo_Left  Var PORTB.3      'Lower Left
AD_ON VAR ADCON0.0
AD_START  VAR ADCON0.2
SDO VAR PORTD.0
SCL VAR PORTD.1
mode var word
mode = %0101000000111100
PAUSE 500                 'Wait for LCD to setup
CNT = 0
PORTC.0 = 1
Last_X = 0
Last_Y = 0

main:


  Gosub AD_X
  pauseUS 10
  GOSUB AD_Y



  if cnt = 10 THEN
     LCDOUT $FE, 1
     LCDOUT $FE, $80, "X: ",DEC Result_X
     LCDOUT $FE, $C0, "Y: ",DEC Result_Y
     CNT = 0
  ENDIF

  CNT = CNT + 1
  if TSTART = 1 THEN
     IF Result_X != Last_X || Result_Y != Last_Y Then
```

```
        serout SDO, N9600,[#Result_X," ",#Result_Y,13]
        Last_X = Result_X
        Last_Y = Result_Y
      ENDIF
   endif
Goto main

AD_Y:                   'Measure X
   Up_Left = 1
   Up_Right = 1
   Lo_Right = 0
   Lo_Left = 0
   Result_Y = 0
   pauseus 500     'PAUSE 5
   AD_START = 1
   While AD_START = 1
   WEND
   PAUSEUS 50
   Result_Y.HighByte = ADRESH
   Result_Y.LowByte = ADRESL
   Up_left = 0
   Up_Right = 0
RETURN

AD_X:                   'Measure Y
   Up_Right = 1
   Lo_Right = 1
   Up_Left = 0
   Lo_Left = 0
   Result_X = 0
   pauseus 500
   AD_START = 1
   While AD_START = 1
   WEND
   PAUSEUS 50
   Result_X.HighByte = ADRESH
   Result_X.LowByte = ADRESL
   Up_Right = 0
   Lo_Right = 0
Return
```

```
'TEST_SERIAL:
 '   SHIFTOUT SDO,SCL,1,[Result_x/16,Result_Y/16]
'RETURN

END
```