HARDWARE DESIGN DOCUMENT

1. INTRODUCTION

This document explains in detail the hardware components used in the design of Sparrow's eVote as well as describe the firmware code used to make the system functional. The mention of eVote throughout this document refers to the Sparrow design.

2. DESIGN CRITERIA AND SPECIFICATIONS

This section describes the hardware components used in detail as well as their power consumption within the system.

2.1. HARDWARE SYSTEM SPECIFICATIONS

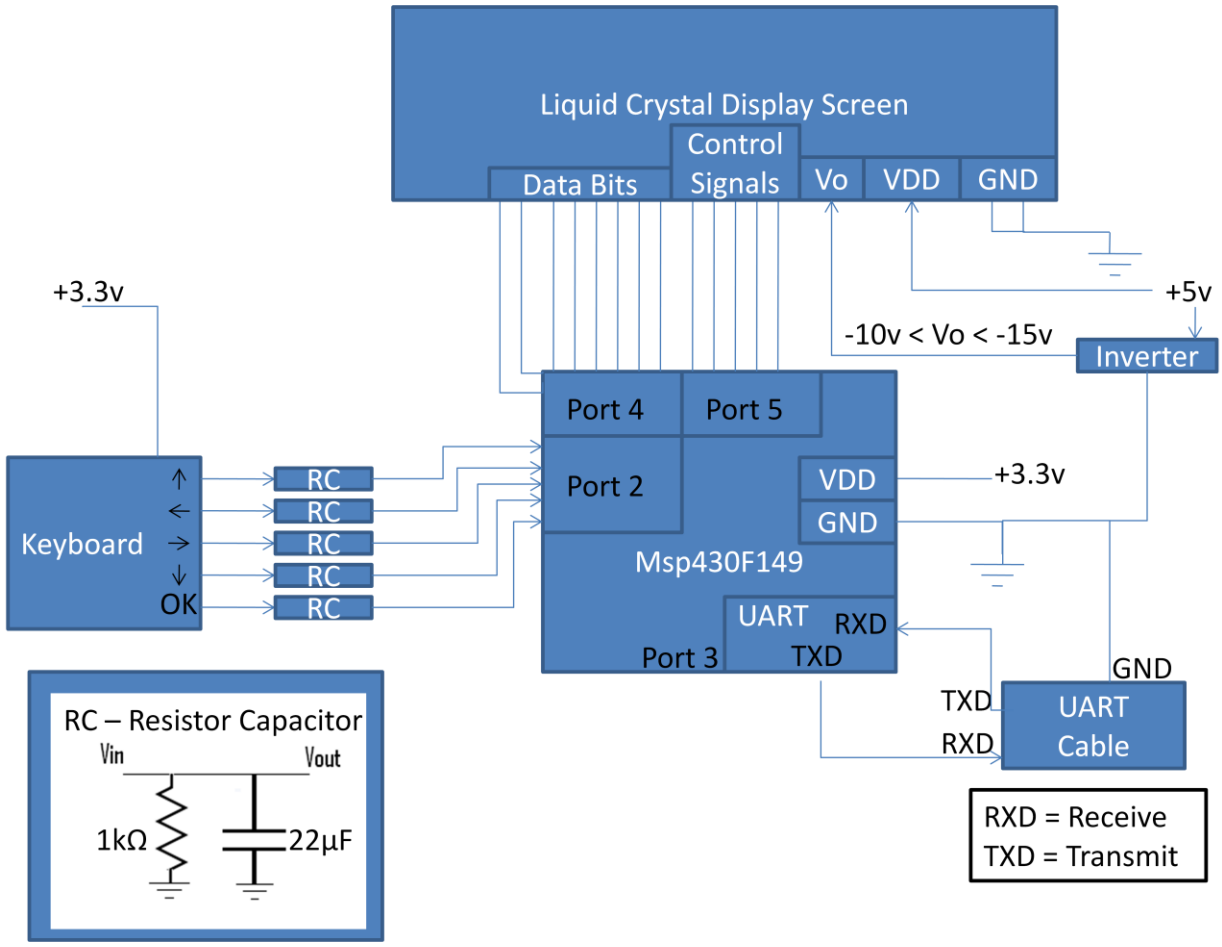
This section describes the components used for the eVote system and their corresponding implementation in eVote as well as their power dissipation.

2.1.1. MSP430F149 MICROPROCESSOR

The Sparrow was originally designed to work with an MSP430FG4619 microprocessor because of the one hundred pins in which only eighty were I/O pins (input, output respectively). The MSPFG4619 also possessed the functionality to transfer information through a UART cable (Universal Asynchronous Receiver/Transmitter) to a software application. During the Third phase the microprocessor was changed to an MSP430F149 due to the nonfunctional UART pins. It is believed that the internal crystal which is used to create a frequency in which information is sent to the computer software was damaged due to static electricity or simply came shipped damaged. The team realized that a new microprocessor would be needed once an external crystal was used to bypass the internal crystal. This verification resulted in the microprocessor not changing its behavior when sending characters to the computer since garbage would be presented in the hyper-terminal. The decision to switch to an MSP430F149 was made since the microprocessor successfully sent characters to the hyper-terminal using the same crystal used in the previous microprocessor. The budget has not been affected by changing the microprocessor since nothing new was purchased to accommodate the design changes. Please refer to the budget in the Final Report document for more details. [3]

The MSP430F149 possesses two UART ports which allows for an additional expansion if necessary. The microprocessor allows for firmware up to 60 kilobytes in size, but due to use of the free use of Code Composer (firmware development software) eVote is only allowed 16 kilobytes of firmware. Altering the hardware design to use this microprocessor only resulted in

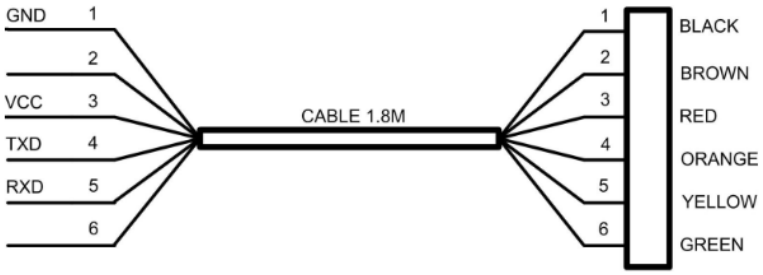
changes in the code and no additional hardware elements were needed for the transition. Figure 1 shows the overall hardware design of eVote. For the power consumption of the design please refer to section 2.2.



eVote Hardware Design Diagram - Figure 1

2.1.2. UART

The UART model used is called "TTL-232R-3V3" and is connected to pins four and five of port three, for transmitting and receiving data respectively. The ground found in the UART cable is connected to the same ground used by all the hardware components. The ground used by the liquid crystal display is also on the same node as the microprocessor and the UART. Figure 2 shows the UART cable connections in which only three are used, TXD (transmit), RXD (receive) and GND (ground). The VCC was not used since the system is powered externally with a nine volt dc adapter. The nine volts are regulated through-out the system to provide specific level of voltages to each hardware component as needed. [1]



UART Connections - Figure 2

2.1.3. KEYPAD

The keypad used does not require decoding the output to verify which button was pressed like other keypads which have a matrix of columns and rows. Depending on the row and column of the button pressed a specific value would have to be decoded to determine the button pressed. The keypad implemented simple outputs only one active pin for each button resulting in 12 possible outputs. The thirteenth pin on the keypad is for the input voltage of 3.3 volts.

The keypad is connected to port two of the microprocessor with RC (resistor-capacitor circuits) on each output. This circuit was implemented to eliminate the bouncing effect that was made apparent during the write-in process of eVote. The keypad would repeat the selected key many times repeating a selected character, with the RC circuit, this was eliminated. Refer to figure 1 to see the RC circuit, the values for the resistors and capacitors were 1kilo-ohm and 22µF respectively. The power dissipated is negligible unless the user keeps a key pressed for a prolonged amount of time. In the case of eVote, the voter is not expected to keep any keys pressed for more than a few moments (during testing the average time keeping a button pressed

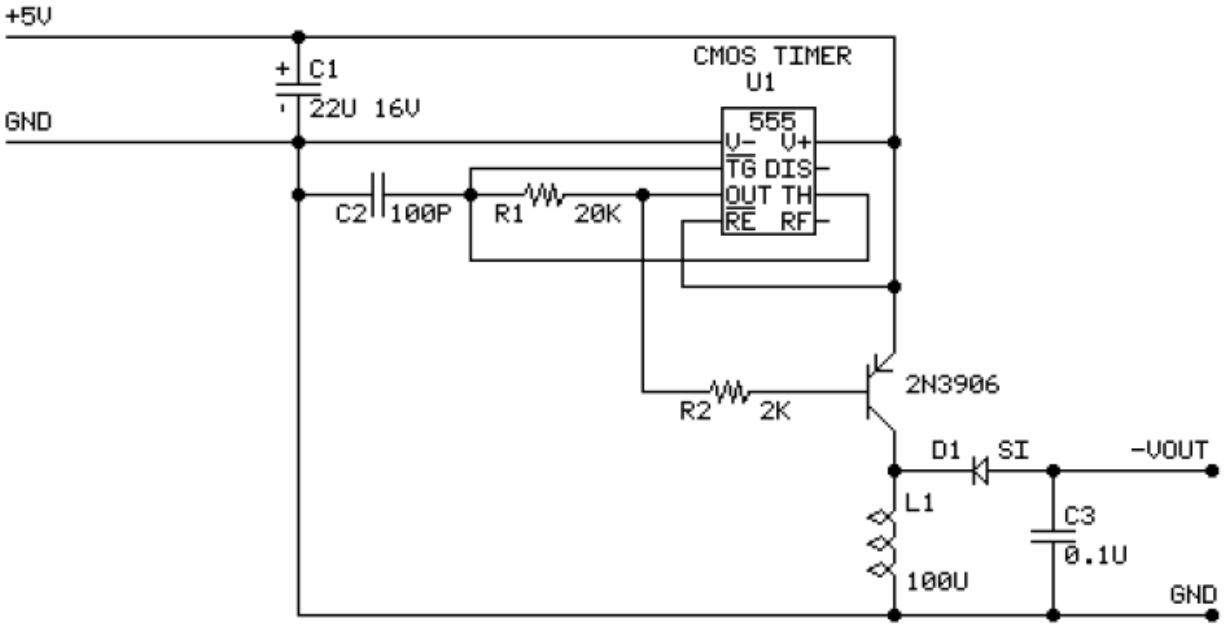
never surpassed a second or two) and since only the keys that are pressed become active, they do not dissipate power otherwise.

Although the keypad originally had thirteen pins, only six are used in the implementation of eVote. The functionality desired only required buttons to allow for navigating up, down, left, right and to press "ok" once the voter makes a selection and the last pin for the input voltage.

2.1.4. LIQUID CRYSTAL DISPLAY

The LCD (liquid crystal display) is a Hyundai HG25504 5.8" x 4.58" graphic display module, which has its own microcontroller to manage its internal functions such as text and graphics capabilities. The LCD needs to be initialized and preconfigured via firmware before anything is properly displayed. This particular LCD allows adjustment for the size in text up to 16 pixels in height, JSAL has decided to use 8 pixels in height considering the amount of text that is to be displayed in each screen of the voting ballots. The LCD has five main areas of interest: control signals, data bits, Vo, VDD and GND. As seen in Figure 1 the control signals have their own port as well as the data bits. This was done to simplify the firmware implementation and also because each port only allows eight pins to be connected. The voltage needed for the LCD is 5 volts and this voltage is acquired by regulating the 9 volts input into the system. [2]

The LCD also has an inverter which is used to control the contrast (darkness levels) of the LCD screen. The contrast is adjustable by modifying the potentiometer found at the load of the inverter circuit. The potentiometer used in the circuit is of 5kΩ which is what adjusts the contrast. Turning the potentiometer clockwise darkens the contrast, while counterclockwise with lighten it. The inverter circuit can be seen in Figure 3.



Inverter Design - Figure 3

The inverter design was obtained with permission and the copyright can be found in the appendix of this document.

2.1.5. VOLTAGE REGULATORS

The regulators used for the eVote hardware are of the LP2950-XXLPRE3 variety, where for a 3.3 and 5.0 volt, the "XX" values are 3.3 and 5.0 respectively. In the implementation of the hardware one 5 volt and 3.3 volt regulator was used to control the input voltage for the microprocessor, keypad, LCD and inverter. Please refer to figure 1 for the input voltage values which are regulated before being input into each hardware component. [4]

2.2. POWER CONSUMPTION

After analyzing the needed voltage and current for each hardware component it was possible to calculate the total power dissipation of the eVote system. The power consumption is as follows:

LCD : (5 volts)* (15mA) = **75mW**

Microprocessor : (3.3 volts)*(280µA) = **0.924mW**

Inverter : (15 volts)*(4mA) = **60mW**

UART Cable : (5 volts)*(75mA) = **375mW**

Total Power Consumption : **510.924mW**

The inverter provides a negative value of -15 volts for controlling the backplane voltage of the LCD which adjusts the contrast. Although the microprocessor is low power, it seems negligible compared to the power dissipation of the other components.

2.3. FIRMWARE CODE DESIGN

Initially the firmware code design exceeded the maximum space allowed while using IAR Workbench for free so the project was migrated to Code Composer. After completion of all the code, the firmware was dangerously close to filling up the 16 kilobytes allowed. Frequent lines of code that was frequent through out the firmware was reduced by creating specific functions to save code space. The project now only uses 11.39 kilobytes of code and 1.68 kilobytes of data from the allotted 16 kilobytes.

2.3.1. LCD FIRMWARE FUNCTIONS

- **wrCom** – Subroutine that writes a command to the LCD after the data port is set up with the corresponding byte of information.
- **wrData** – Subroutine that sends data information to be used for a command, such as typing a character to the screen.
- **iLCD** – Subroutine to initialize the LCD.
- **clrP4** – Subroutine to clear the data port (port 4)
- **clearLCD** – Clears the LCD from graphics drawn and text on screen. Used when initializing the LCD to clean onscreen gibberish.
- **Locked** – Subroutine that displays the locked screen.
- **bal1** – Subroutine that displays the governor choices.
- **bal2** – Subroutine that displays the resident commissioner choices.
- **Confirm** – Screen that displays a confirmation message
- **sumDisp** – Displays the voting summary
- **party** – Displays the political party choices.
- **Enabled** – Verifies if the voting official activated the kiosk.
- **Type** – Types a string on screen
- **Lan** – Displays the language selection screen.
- **setPos** – Sets the position on screen to place the cursor. Used also to indicate where text should be typed.

- **clrT** – Clears all onscreen text.
- **voteSel** – Displays screen for voting integral or mixed and candidature.
- **wrTest** – subroutine with the navigation logic for the write-in screen.
- **eVote** – The main subroutine for eVote which possesses the navigation logic for the entire voting experience with the eVote device.
- **wrDisp** – Displays the write-in screen.
- **curOn** – Turns on the cursor.
- **curOff** – Turns off the cursor.
- **Wait** – waits until the user lets go of a keypad key before continuing with the eVote firmware. This is to prevent repeated button presses when only one is desired.
- **iUart** – Initializes the UART with frequency parameters and interrupts needed for receiving data.
- **transmit** – Transmits a character through the UART to the computer software.
- **connectPC** – Initializes a connection test between the computer and the eVote device.
- **connectPC2** – periodically tests the connection between eVote and the computer software. An error is displayed if the computer software fails to reply.
- **uartDela**y – A delay to allow for the UART communication to complete.
- **prints** – Displays the printing status screen. If printing fails, an error message is displayed.

2.4. DEVELOPMENT TOOLS

2.4.1. IAR WORKBENCH

This software allowed the team to test in real time the execution of the firmware. On the occasion an error would arise, the step by step debugging procedure was made easy with the help of IAR Workbench. The software would also compile the firmware and upload the code into the MSP430F149. Another benefit of using this program is the ability to see real time what the value of each port is. This allows visual corroboration of the bytes of information being sent or if a port is receiving the correct signals. Ultimately the team changed development tools due to the four kilobyte limitation for firmware.

2.4.2. CODE COMPOSER

The team migrated to Code Composer because the ability to use up to sixteen kilobytes of firmware code allowed us to complete the implementation of the eVote firmware. Code composer also has the capabilities for debugging and view the values of corresponding variables throughout the firmware. Code Composer was used to debug the firmware as well as upload it to the eVote device for external debugging (interacting with the eVote device) alongside the firmware.

2.4.3. MICROSOFT OFFICE 2007

The team drew up many design diagrams using power point which has many options for drawing flowcharts. Also, all the design documents typed for eVote were made using Microsoft Office 2007.

2.4.4. HARDWARE TOOLS

The team used wire-wrapping tools but due to the a dysfunctional circuit it was reverted to a breadboard. The team also used welding tools to create pins used for connections between the different hardware components and the microprocessor. Another tool used is the sanding

tool, which had a peripheral that was used to create the openings of the project enclosure. The most useful of the hardware tools, was the multimeter which allowed for circuit parameter verifications as well as check if the connections of a circuit are correctly setup.

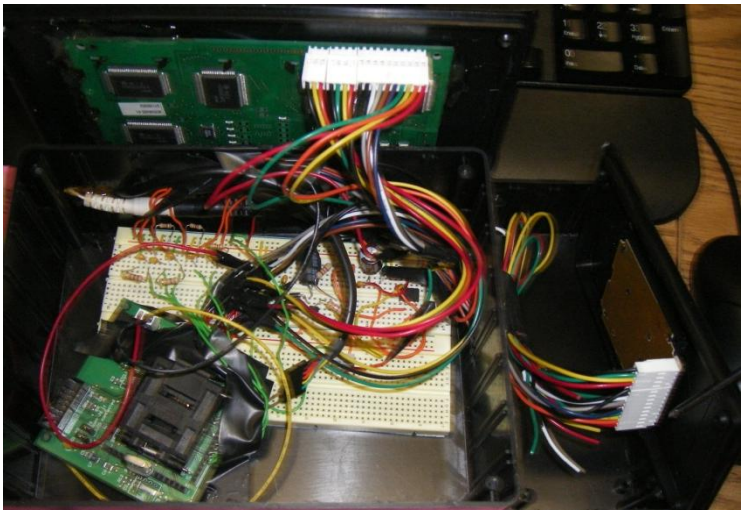
2.5. PROTOTYPE DESIGN

This section describes the design of the eVote prototype and the corresponding prototype design decisions.

2.5.1. EVOTE DEVICE

Two different enclosures were purchased for the eVote device. One smaller enclosure for the keypad since the biggest enclosure would only provide space for the LCD screen. The biggest enclosure contains the entire circuit and was joined together with the smaller enclosure which contains the keypad as seen in figure 4 and 5.

Completed Prototype - Figure 4



Inside Completed Prototype - Figure 5

Using a hardware tool, the team was able to make an opening to pass the keypad cables into the main enclosure where it was then connected to the corresponding port. Electric tape was used to hold the cables into place to allow for more organization. The organization as seen in figure 5 allowed for easy fixes in the case a node of the circuit was not conducting electricity properly. Since the microprocessor was not made permanent to the enclosure, it made the migration process to the MSP430F149 much easier.

2.5.2. DESIGN CONSIDERATIONS

Due to the fact that wire-wrapping and soldering the circuit from the breadboard resulted in a a drop in voltage from the inverter output, the circuit was reverted to a breadboard for an analysis. The source of the problem was found to be a resistor that was not allowing the circuit to function properly. After changing the resistor the project return to its functional state. It was decided to keep the project on the breadboard since too much time was lost on trying to finalize the project circuit.

The content of each screen displayed was modified to fit the screen as well as allow for a more intuitive interface while making selections. The firmware now implements a flashing cursor which highlights the selections made by the user.

The microprocessor MSP430FG4619 was changed to a MSP430F149 in order to complete the functionality of the UART cable. The MSP430FG4619 UART capabilities were not functioning properly, it was supposed to use specific parameters to set a frequency of sending

or receiving data. After calculating these values and implementing them in the firmware, the desired result was never acquired. Changing to the MSP430F149 with the same values calculated previously made the UART function as desired. This result implied that the internal crystal (used for setting the frequency of the UART) may have been damaged due to static electricity.

3. REFERENCES

[1] TTL to USB Serial Converter Range of Cables Datasheet . [Online]. Available: http://ftdichip.com/Documents/DataSheets/Modules/DS_TTL-232R_CABLES_V201.pdf.

[2] SPEC SHEET FOR CAT # LCD -101[Online]. Available: http://www.allelectronics.com/mas_assets//spec/LCD-101.pdf.

[3] "MSP430F149." Texas Instruments. Texas Instruments. [Online]. Available: http://focus.ti.com/docs/prod/folders/print/msp430f149.html.

[4] ADJUSTABLE MICROPOWER VOLTAGE REGULATORS WITH SHUTDOWN [Online]. Available: http://focus.ti.com/lit/ds/symlink/lp2950-33.pdf.

[5] MSP430xG461x
MIXED SIGNAL MICROCONTROLLER [Online]. Available: http://focus.ti.com/lit/ds/symlink/msp430fg4619.pdf.

4. APPENDIX

1. INVERTER COPYRIGHT EVIDENCE

From: "Duane Becker" <snowleop@sover.net>
To: "Javier Torres" <legendary85@hotmail.com>
Subject: Re: GDISP3 - Diode Design
Date: Monday, September 29, 2008 8:56 PM

Just include in your documentation that "The voltage inverter design is
copyrighted in 2006 by Snowleopard Labs
and is used in this project by permission."
There is no licensing fee required, just attribution.
Duane

----- Original Message -----
From: "Javier Torres" <legendary85@hotmail.com>
To: <snowleop@sover.net>
Sent: Monday, September 29, 2008 3:14 AM
Subject: Re: GDISP3 - Diode Design

> Thanks again!
>
> One last question, since I am doing this for a university project and we
> needed to make the financial aspects of the project realistic, the use of
> your design of the inverter is free to use or there are some requirements
> needed before being able to use the design such as copyright or things
> like
> that?
>
> Thanks,
>
> Javier

>
> ----------
> From: <snowleop@sover.net>
> Sent: Monday, September 29, 2008 9:12 AM
> To: "Javier Torres" <legendary85@hotmail.com>
> Subject: Re: GDISP3 - Diode Design
>
>> The marking of SI is meant to mean silicon. Any silicon diode will
>> work here: 1N914, 1n4004, anything.
>>
>> db