**<Project Name>: Team 17**

*Place picture of project with team members here.*

*Complete the caption, below.*

*Delete this text box before pasting in your photo!*

Team members, left to right are: <name 1>, <name 2>, <name 3>, and <name 4>.

|  |  |
| --- | --- |
| *Name* | *Class No.* |
| Daniel Li | 6014-L |
| Zihao Lu | 8445-L |
| Jinglun Huang | 0389-H |
| Jiyuan Zhao | 7815-Z |

|  |  |
| --- | --- |
| GRADING CRITERIA | MAX POINTS |
| Originality, creativity, level of project difficulty | 20 |
| Technical content, succinctness of report | 10 |
| Writing style, professionalism, references/citations | 10 |
| Demonstration of functionality | 20 |
| Overall quality/integration of finished product | 10 |
| Effective utilization of microcontroller resources | 10 |
| Significance of individual contributions\* | 20 |
| Poster bonus | 10 |
| Video bonus | 10 |
| Royal Showcase participation bonus (poster and video required) | 10 |

##### \**scores assigned to individual team members may vary*

##### Scoring Multiplier:

|  |  |
| --- | --- |
| 1.0 | *Excellent – among the very best projects/reports completed this semester* |
| 0.8 - 0.9 | *Good – all requirements were amply satisfied* |
| 0.6 - 0.7 | *Average – some areas for improvement, but all basic requirements were satisfied* |
| 0.4 - 0.5 | *Below average – some basic requirements were not satisfied* |
| 0.1 - 0.3 | *Poor – very few of the project requirements were satisfied* |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| 1.0 Introduction | 1 |
| 2.0 Interface Design | 2 |
| 3.0 Peripheral Utilization | 3 |
| 4.0 Software Narrative | 4 |
| 5.0 Packaging Design | 5 |
| 6.0 Summary and Conclusions | 6 |
| 7.0 References | 7 |
| Appendix A: Individual Contributions and Activity Logs | 8 |
| Appendix B: Interface Schematic | 17 |
| Appendix C: Software Flowcharts | 19 |
| Appendix D: Packaging Design | 21 |

**1.0 Introduction**

**2.0 Interface Design**

The 9S12C microcontroller is connected to several components, including push buttons, the 74HC164 Shift Register, the HD44780U LCD, and a 273-092 8-Ohm Mini Speaker. The power for all of these devices is provided by the WDU9-300. Schematics for the entire design can be seen in Appendix B. The datasheets for the components can be seen in Section 7.0.

The HD44780U LCD is connected to the microcontroller with the 74HC164 Shift Register and Port T. The shift register is used along with SCI to quickly send 8 bits of data using only two pins. The MOSI pin of the microcontroller sends the data signal to the 74HC164, and the CLK pin sends the clocking signal. Port T pins are used as control signals to the LCD. PTT2 is connected to the RS pin, which controls whether the data is used as a character or a command. PTT3 is connected to the R/W pin, which controls if the LCD will read or write data. PTT4 is connected to the LCDCLK pin, which clocks the registers in the LCD. Each bit in the shift register is connected to the corresponding bit in the LCD data in. The datasheets for the shift register and LCD show the numbering of these pins.

Four push buttons are connected to Port AD for the user interface. They are used as active-high and connected to ground with a pull-down 10k resistor. Another push button is used to the reset the microcontroller. It is active-low and connected to the RES pin (Pin 30) of the microcontroller.

The red LED and the speaker are connected to Port T of the microcontroller. Both of these are directly connected between the microcontroller and ground. The LED is used as a standard output of the microcontroller, but the speaker connected to PWM Channel 0 at PTT0.

Since the LCD needed to be on the front panel of the device, it was placed on a miniature breadboard along with the shift register. The miniature breadboard was placed on its side and taped to the back of the front panel. This design allowed us to solder a minimal amount of wires together.**3.0 Microcontroller Resource Utilization**

The SPI peripheral is used by the microcontroller to interface with the 74HC164 Shift Register. Based on the shift register datasheet, the maximum frequency is always higher than 6 MHz at room temperature. Therefore, the SPI is initialized for 6 Mbps, and most significant bit first. Since the data is only being written to the shift register and not read, the SPI is initialized for master mode, single direction, and no interrupts. When data is written into the SPI data register, the SPI automatically shifts out the data to the shift register.

The timer peripheral is used for precise delays and counting how long a push button is asserted. Timer channel 7 is used for interrupts only, at a rate of 1 ms. First, channel 7 is enabled and set to output compare mode with interrupts enabled. The timer is initialized for a prescale of 16, and TC7 = 1500, with reset when output compare channel 7 is asserted. Morse code is heavily dependent on the timing of signals, so the timer is necessary to ensure that time periods are accurate. With an interrupt rate of 1 ms, the program can easily and accurately wait certain amounts of time.

The PWM peripheral is used to create a waveform for the speaker. The PWM will output on Channel 0, so PWM Channel 0 is enabled, using Clock A and MODRR is used to route it to PTT0. The frequency of the input square wave should be between 1 kHz and 2 kHz to be an audible sound. Therefore the PWM is initialized for active-high, no concatenate, left-aligned, with a prescale of 64 and a period of 255. This results in a frequency of about 24000000/255/64 = 1470 Hz. The PWM and speaker can be easily turned off by setting the duty cycle to 0 and turned on by setting it to a nonzero value such as 0x7F.

The RTI system is used to sample push buttons. The interrupt rate is set to 2.048 ms to be fast as well as accurate. The previous state of the push buttons is stored in a variable and compared to the current state. If the buttons changed from inactive to active, then that button’s flag is set.

**4.0** **Software Narrative**

The software in the Morse Code Converter is best described as a state machine (flag-driven) approach. The program initializes all flags, tables, and peripherals before entering the main loop. The main loop of the program checks for push button flags set by the RTI and performs the appropriate action. A more detailed flowchart can be found in Appendix C.

If the mode1 flag is set and the program is currently in mode 2, the input string is reset and mode 1 is entered. If the flag is set while it is in mode 1, the currently selected character is entered into the input string. If the leftpb flag is set, the cursor for the currently selected character moves left. Similarly, the rghtpb flag causes the cursor to move right. After each cursor move, if the cursor is outside the bounds of the screen, the page is changed. If BS (backspace) is selected, then the last entered character is deleted. If EN (enter) is selected, then the program begins conversion from the input string to morse code. During the conversion, the LED on the front panel will light up for 0.25 seconds for a dot and 0.75 seconds for a dash. While the LED is lit, a speaker connected to the PWM is activated. The amount of time in between actions is controlled by timer interrupts. The program waits for 250 interrupts for 0.25 seconds, and 750 interrupts for 0.75 seconds. In mode 1, the LCD displays a character menu on the top row and the input string on the bottom row.

If the mode2 flag is set, the program enters mode 2 and clears the input string. The program takes in Morse input from the mode2 button until either 16 characters have been entered or user breaks the loop by changing to mode1. Timer interrupts constantly sample the mode2 push button, which allows the program to detect how long the button has been asserted or deserted. These values are converted into a Morse sequence of dots and dashes. Also, the timer interrupts light the LED and activates the speaker when the mode2 button is asserted. After a Morse sequence is entered, it will be converted into a letter and entered into the input string. If an unrecognized sequence of Morse code is entered, such as ‘-...’, then the LCD displays an error and discards that sequence. While in mode 2, if the leftpb flag is set, the last entered character will be deleted. In mode 2, the LCD displays the current Morse sequence on the top row and the input string on the bottom row.

Lookup tables are used to convert from Morse code to characters and from characters to morse code. Morse code is converted into a base 4 number with the most significant digit being the first input. A dot in Morse code is converted to a 1, a dash is converted to a 2. If the entry does not contain a total of four dots and dashes, the rest of the values are filled with 0s. For example, ‘--..’ would convert to 2211 (165 in decimal) and ‘-.-’ would convert to 2120 (152 in decimal). For the table that converts characters into Morse code, morsetable[0] corresponds to A, morsetable[1] corresponds to B, etc. Since A is ‘.-’ in Morse code, morsetable[0] is 1200 base 4 (94 in decimal). B is ‘-...’, so morsetable[1] is 2111 base 4 (149 in decimal). The inverse table is morsetable2, which converts morse code into characters. morsetable2[94] is equal to ‘A’ and morsetable2[149] is equal to ‘B’.

**5.0 Packaging Design**

It was decided at the beginning of the project on the group meeting that a carbon box was used to house all electrical components and wires. The pushbuttons, LED, LCD screen and power supply could be easily mounted. The breadboard was used for all the internal wiring and the microcontroller and all the wires were simply plugged on the board. So a carbon box with length of 21 inches, and width and height of 15 inches was chosen.

The speaker was wired and hidden inside the box since user was no need to see it. One small and four larger holes were drilled on a display panel for fitting the red LED showing the signal and four pushbuttons, two for choosing left and right, two for Mode 1 and 2. On the right side of the panel, a rectangular shaped area was cut to show the LCD screen. All parts were glued tightly on the panel to both save some space on the breadboard and prevent falling. After that, the display panel would be tapped on the front of the box. On the left side of the box, a hole was drilled to pass the reset pushbutton. The breadboard was tapped inside the box preventing moving. The 5 volts power supply was connected to the microcontroller directly and came out at the back of the box. The box was closed on the back and only left a small hole to let the cable on

When all components were fixed in the box, the box was sealed and a sheet of hard black cardboard was used to cover the whole box with pushbuttons and LCD screen shown outside. A white paper printed the Morse code was pasted on top of the box for user to find the corresponding English letters.

**6.0 Summary and Conclusions**

**7.0 References**

74HC164. Retrieved from <http://www.nxp.com/documents/data_sheet/74HC_HCT164.pdf>

HD44780U (LCD-II). Retrieved from <http://www.serialwombat.com/parts/hd44780.pdf>

WDU9-300. Retrieved from <http://www.mouser.com/ds/2/410/WDU9-300-224465.pdf>

**Appendix A:**

**Individual Contributions**

**and**

**Activity Logs**

**Activity Log for:** Daniel Li **Role:** Software Leader

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
| **Team meeting to plan project** | 4/5/2013 | 8:00 | 9:00 | 1h |
| Wrote the main program | 4/6/2013 | 7:00 | 9:00 | 2h |
| Went to Radio shack |  |  |  |  |
| In lab discussion for particular parts needed | 4/9 | 7:00 | 9:00 | 2h |
| Team meeting to discuss the final version of project | 4/14 | 7:00 | 8:00 | 1h |
| Started wiring | 4/15 | 7:00 | 12:00 | 5h |
| Finish wiring | 4/16 | 12:00 | 5:00 | 5h |
| Start packaging | 4/17 | 4:00 | 7:00 | 3h |
| Finish packaging | 4/18 | 1:00 | 4:00 | 3h |
| Demon | 4/19 | 10:30 | 10:45 | 15min |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Written Summary of Technical Contributions:** <name-1>

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*Length should be about one page.***Activity Log for:** Jinglun Huang **Role:** Peripheral Leader

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Written Summary of Technical Contributions:** <name-2>

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*Length should be about one page.***Activity Log for:** Jiyuan Zhao **Role:** Interface Leader

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Written Summary of Technical Contributions:** <name-3>

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*Length should be about one page.***Activity Log for:** Zihao Lu **Role:** TDP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

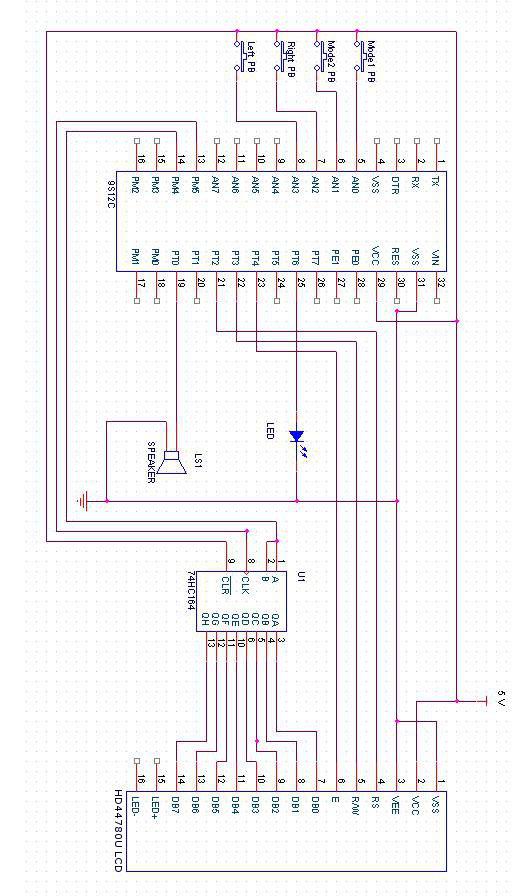
**Written Summary of Technical Contributions:** <name-4>

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*Length should be about one page.*

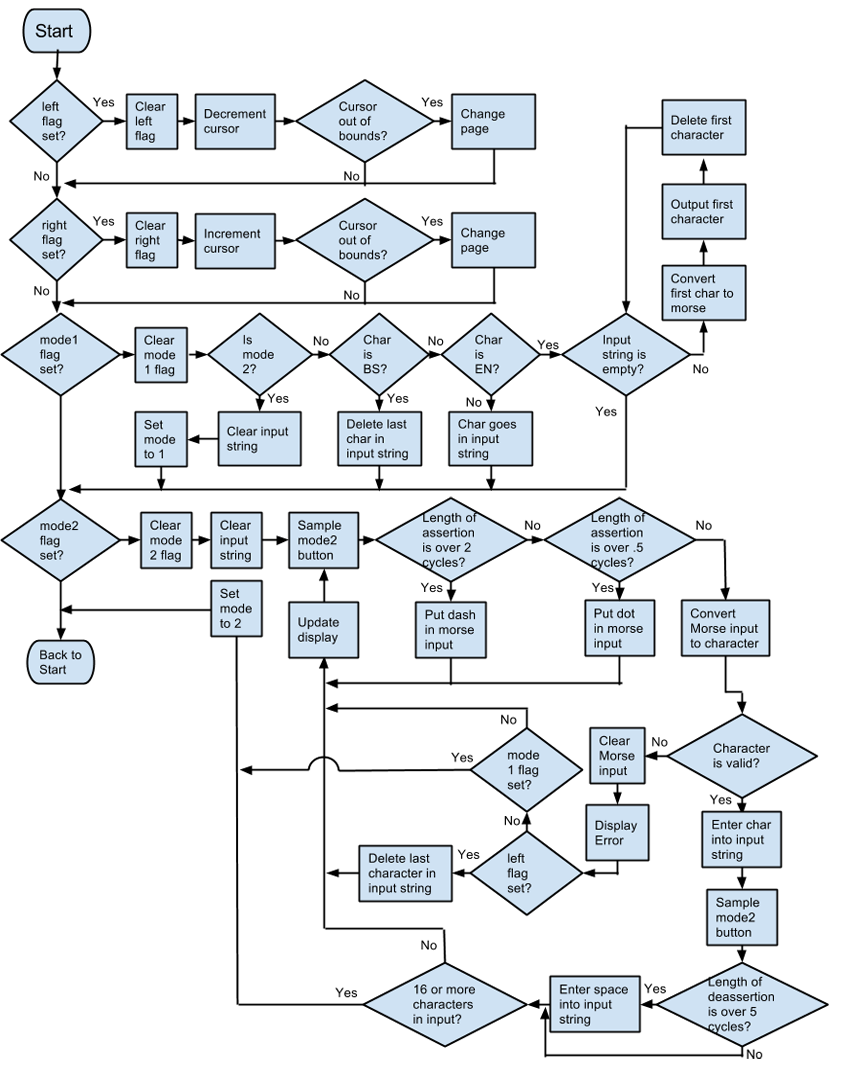
**Appendix B:**

**Interface Schematic**



**Appendix C:**

**Software Flowcharts**

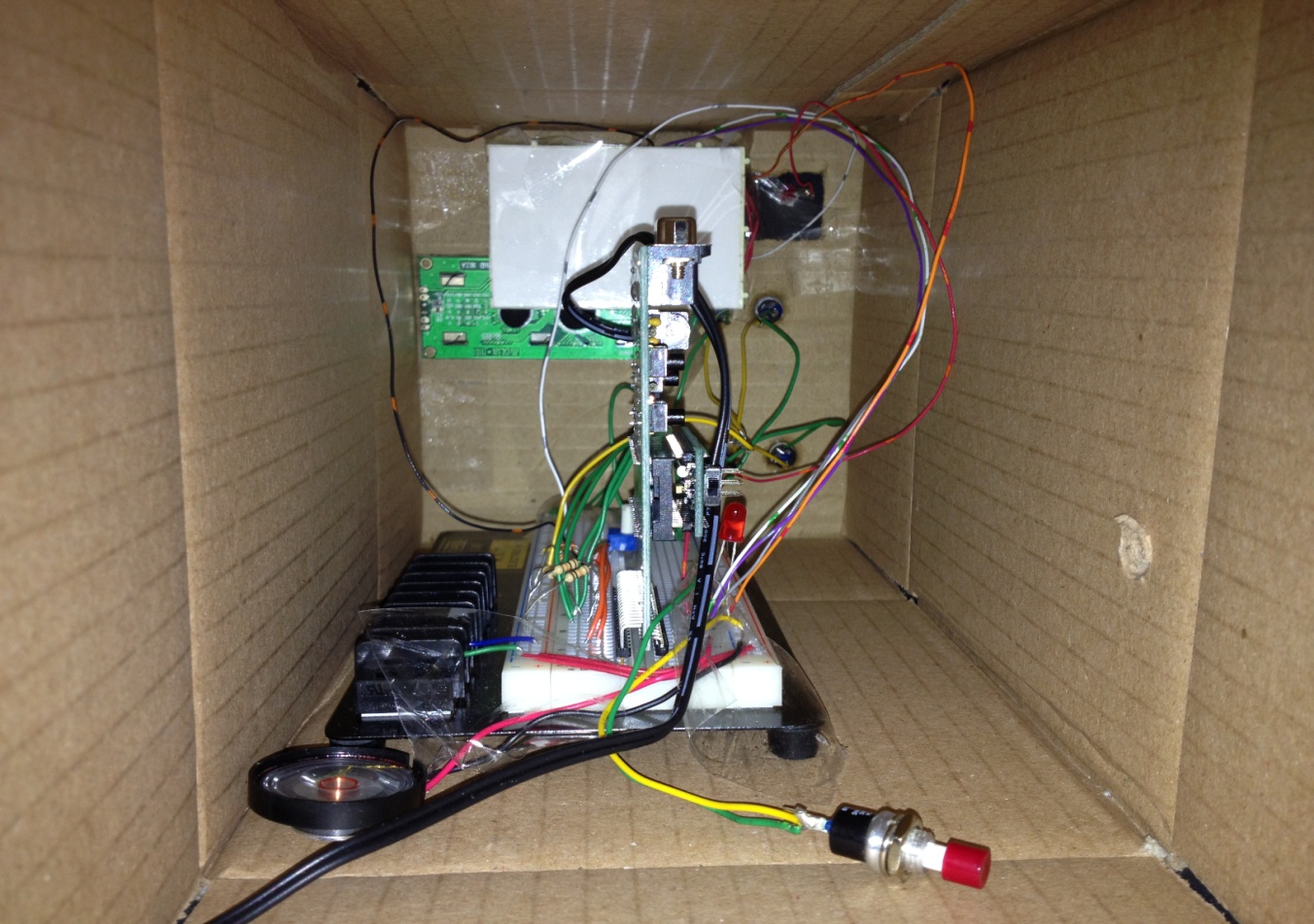


**Appendix D:**

**Packaging Design**



**Figure 5.1 – Outside of the box**



**Figure 5.2 – Inside wiring on the breadboard and microcontroller**