Morse Code Project Report

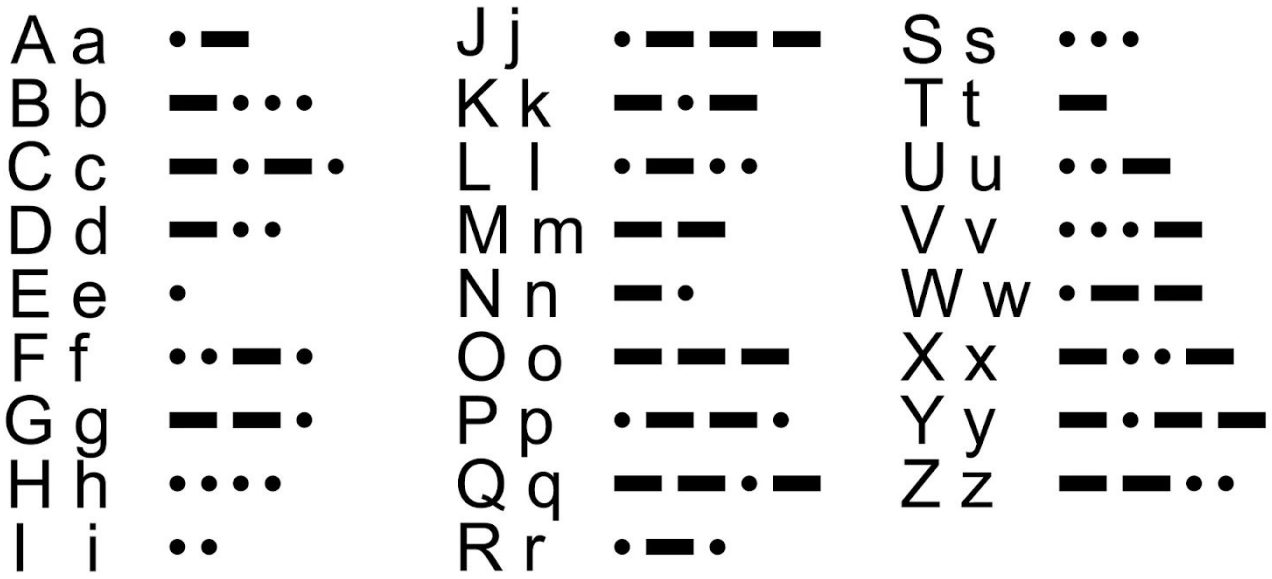
12/11/19

Microcontrollers

By Chikaodiri Nwachukwu, Chris Thompson, Yeray Lopez

Intro

Morse Code is an alphabet or code in which letters are represented by combinations of long and short signals of light or sound. This was used mainly by the military to encrypt messages and make sure only the proper people understood these messages. With our Microcontrollers project, we implemented a Morse code emulator. Our goals in this project was to input human characters, have the program convert the characters to morse code signals, and display the morse code signal through the LEDs and through the speaker via tones.



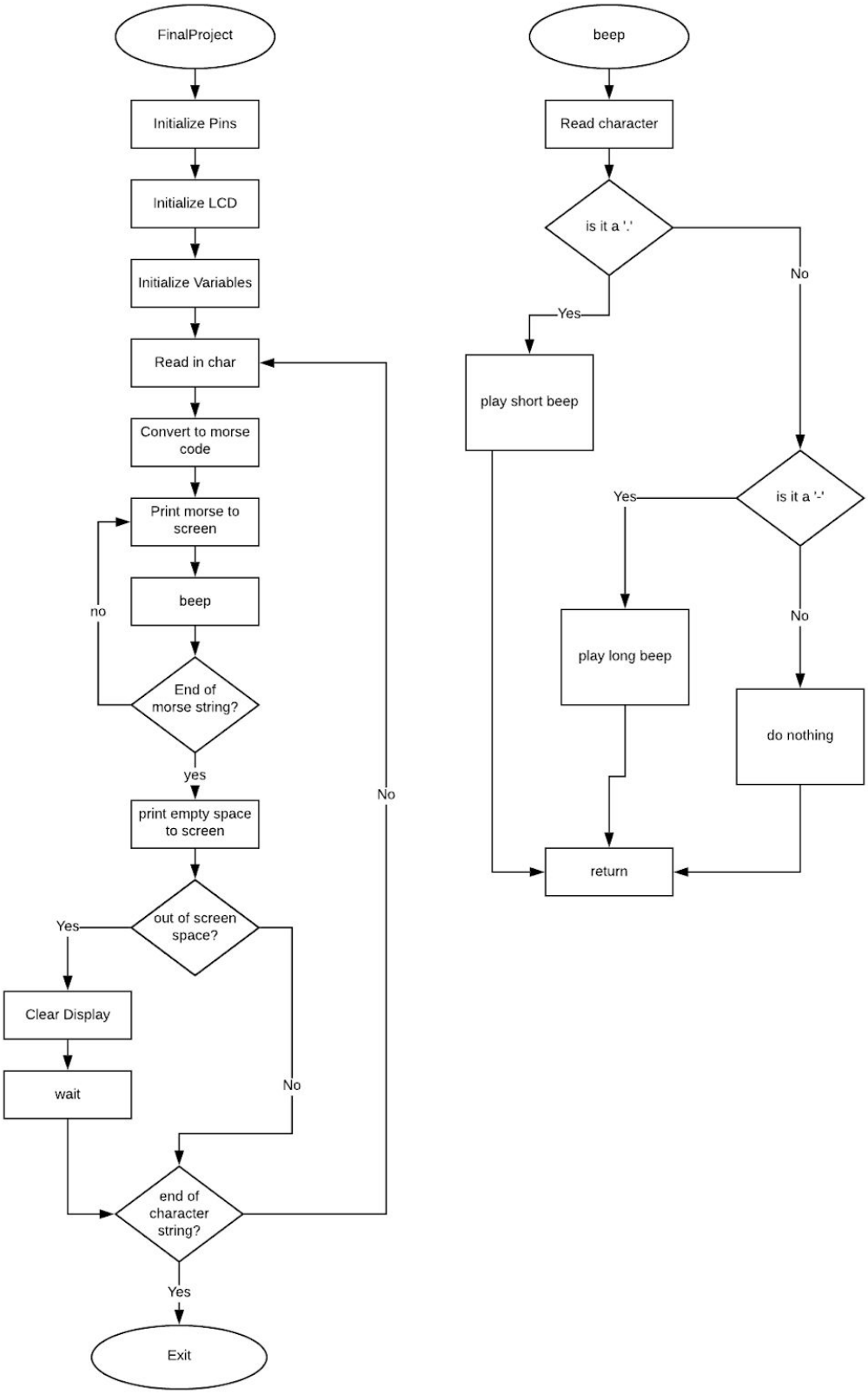
The above figure represents how each character in the alphabet is represented in Morse Code. In this figure, you see two shapes a dash and a dot. The dot represents 1 unit of time while the dash represents 3 units of time and this ratio is what differentiates these two signal lengths and produces all the combinations for all the different characters.

Methodology

The logic used in the character to Morse Code conversion starts with a library called <morse.h>. This library basically had all the combinations that represented all the different characters. For the 1 unit of time signal we used a period (‘.’) and for the 3 units of time signal we used the underscore character(‘_’). So basically after a message is inputed the output would look something like this (‘___… __…’).

The next part of this project was displaying the Morse code signal on the LED display. We basically just repeated the same implementation that we did in the earlier lab to display the Morse Code message. The only difficult part of this project was clearing the display when it ran out of room to display the message. We had to use some conditional to put constraints on when to clear the display. The last segment of our project was the auditory signal that would represent the morse code. We basically setup a function called the beeper function that took in the converted to Morse Code output as an input and played the auditory signal based on the different periods (‘.’) and underscores (‘_’) . We were able to accomplish this by making separate functions that enabled the speaker and produced either a tone of 1 length of time or 3 lengths time units. The Beeper function goes through the Morse Code string and if the char is a period (‘.’) will call the short beep function and if the if the char is an underscore (‘_’) , then it would call the long beep function. This is how the Morse Code is represented through the speaker.

Flowchart


FinalProject
Initialize Pins
Initialize LCD
Initialize Variables
Read in char
Convert to morse code
Print morse to screen
beep
End of morse string?
no
yes
print empty space to screen
out of screen space?
Yes
No
Clear Display
wait
end of character string?
No
Yes
Exit
beep
Read character
is it a '.'
No
Yes
play short beep
is it a '-'
Yes
No
play long beep
do nothing
return

Conclusion

We were able to successfully make a Morse Code emulator. We were able to make a system that takes human characters as input, converts the characters to morse code signals, and displays the morse code signal by the LED display and through the speakers via auditory tones. We ran into one problem which we believe is a hardware issue due to the display lagging. We believe we could solve this problem by using different hardware or taking a different approach to displaying the data. Although we had this small bug our project definitely showed proof of concept and we were able to learn more ways in which we can use the LPC218 board.