Morse Code Coder

Students:

Andrei Cioanca - 266105

Claudiu Rediu - 266129

Stefan Harabagiu – 266116

# Problem statement

The Morse code played an important part throughout history since its development in the 19th century. The most extensive use of the code was seen during World War 2 where standard long-range radio communication was impractical due to size and security. The solution was the Morse code. It was used in order to encode and decode the messages.

The Morse code is a method to encode text into series of dots and dashes which can be sent by light, sound or radio waves.

Our project consists of a way for a user to type a specific letter or digit into the computer and then see it translated in Morse code through the microcontroller.

<https://upload.wikimedia.org/wikipedia/commons/b/b5/International_Morse_Code.svg>

# Building the Morse Code Coder

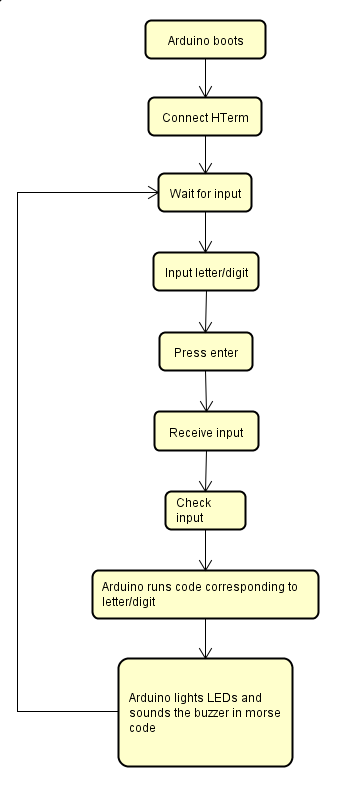
Schematics.png

To create the system, we need an Arduino Mega 2560 development board, a breadboard, a buzzer, at least one LED, a resistor and cables to direct the electricity.

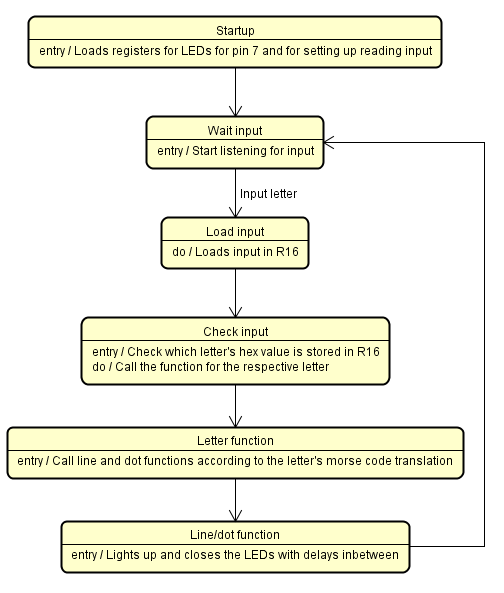
The end of a cable will be set into port PB7 which is marked with 13 on the board and the other end will be set in series with a resistor. The LED will be put so it is in series with the resistor. If there is enough space on the breadboard, you can add the buzzer in series after the LED. Take another cable, set one end into the GND(grounding) and the other at the end of the buzzer. Deploy the code and then you are ready to test it.

If there isn’t enough space to connect the buzzer, there is a solution. Use a cable to add another route to electricity so it can get to the buzzer. Set up a resistor before electricity enters in the buzzer. Take a last cable, connect it so it routes the electricity to the cable that goes in GND.

# Analysis and Design



This is the graphical representation of the behavior of the system. The Arduino is waiting for input as long as the user still wants to input data. After doing the actions specific to the character inputted, it comes back to waiting for input.



The system goes through different phases as it performs the actions specific to each state. The latter goes into more detail with the implementation and results. The result is given by checking the data in a specific registry that was designated for storing the characters from the user. After the check, certain functions (Line and Dot) are called according to the translation into morse code.

# Testing

Testing.png

Connect Arduino to the computer

Start HTerm (Serial Communication)

Set baud to 9600

Click on connect

Select type ASC

Input a letter/digit then press enter

|  |  |  |
| --- | --- | --- |
| No. | Letter/Digit | Expected result |
| 1 | E | Pause Dot |
| 2 | T | Pause Line |
| 3 | A | Pause Dot Pause Line |
| 4 | J | Pause Dot Pause Line Pause Line Pause Line |
| 5 | 5 | Pause Dot Pause Dot Pause Dot Pause Dot Pause Dot |
| 6 | 9 | Pause Line Pause Line Pause Line Pause Line Pause Dot |

Dot = 0.3 sec light up

Line = 0.9 sec light up

Pause (between dots/lines) = 0.3 sec

16 MHz = 16 million cycles

1 delay = 0.3 sec or 300 ms