Morse Code Arduino Machine

Kenshiro Iwaki, Osmar Hernandez

Kiwaki2, oherna25

kiwaki2@uic.edu, oherna25@uic.edu

University of Illinois at Chicago

CS 362 Computer Design

April 2, 2021

Abstract

Morse code is a way of representing alphabet, numerals, and punctuation marks with the

dots (dit), dashes (dah), and spaces, according to the encyclopedia Britannica.just like how operators sent out messages using a telegraph machine to send messages we can do the same with . due to the nature of the telegraph and its operation it costs at least 30k to build one and hundreds to send one. Today we can accomplish the same thing using 40 dollars worth of equipment. A simple Arduino, without any expensive equipment, can reproduce a telegraph machine. Our machine can not only display the message but can be equipped to send it to almost anywhere the user wants.

Project Ideas

We are creating a morse code machine that translates morse code into plain English. The

Arduino will understand the user’s input as morse code and will show the equivalent message in

English on the display. We will also send the message from Arduino via Bluetooth and send it to

a smartphone. The Arduino will receive the morse code back and will display the message in

English on the display. When the Arduino is not connected to a smartphone, it can still be used

as a single player morse code translator.

Two buttons are used for the input. One button is for entering the morse code and another

one is for resetting the message. The 16x2 display will be the output to show the translation. A

Bluetooth adapter will be another output device, sending the morse code to a computer. For the

two-way communication, the adapter will also be used as the input to receive a morse code. The

user’s message will be displayed on the top screen while the receiving message will be displayed

on the bottom screen.

A smartphone app will be used to connect the two devices via Bluetooth. There are

multiple apps available for sending and receiving messages to the smartphone. The smartphone

user will see the morse code and respond back in morse code. The morse code will be in text

format with dot and dashes, so the user can translate it themselves. Since the Arduino can

translate the message, the Arduino will see the translated English text instead of the morse code.

This is still two devices communicating in morse code since the message was initially in morse

code but was automatically translated for the user’s convenience.

The two-way communication may allow the device to be an Arduino or not. It just needs

accept the morse code from an Arduino and be able to send back a morse code. This means that

the device sending the message doesn’t care if it the receiving device was able to translate it,

because it only communicates in morse code. This is original because the morse code translator

ideas on the Internet is smaller scale in comparison without this two-way communication. The

morse code class is also original since using a tree is a simple way to get the letter. Another way

could be using a map/hash map, but a tree is easier to handle both the dot and dash codes because

it doesn’t count them.

We used a tree diagram as a reference to implement the translation, since it used a tree

data structure for the logic. We found a better diagram than last time on a Medium article by

Berk Ozer. The way it moves down the tree depends on whether the message was a dot or a dash.

The current node in the tree is the current character. We have both the circuit and the code

working for the single player mode. It translates the button presses to the correct letter.

Section 5

The LCD and the Bluetooth adapter are optional as the code for the project is reusable and easily modified to allow for other forms of connectivity to a external source. Basically for the circuit, connect the LCD and the button to the breadboard. Run the code which is basically 2 buttons and a linked list class that is used to translate the morse code to alphanumerical English

Section 6

Depending on what hardware you are using or have your experience may vary. There are 2 buttons, button is the main input button. This is where you will taking the morse code you input and translating it to English and numeric values. The second button is a reset button. It resets the entire input string at the current point and resets to blank. This link will help you if don’t already know the morse code language.

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

The development for the one player mode was done during the spring break. The circuit

for lab 8 was used since they both use two buttons and a 16x2 display. The Morse Code class was

created, which translates the current letter. The class use the Node struct for each node, which

contains a letter and the left and right pointers to other nodes. The class keeps track of the root

node and the current node, so that the current node can be reset when moving to the next letter.

The implementation of getting the dot and dash codes from the button presses was done

next. Counter variables were used to determine the time elapsed since the last button press every

loop. That means how long the user pressed the button changed whether it was a dot code, dash

code, or complete command. Same logic applies to the reset button, although that only does reset

with instead of multiple options. The improvement was made at the start of the program to wait

for the user to press button one for the first time, since not pressing the button also counts as a

code to add a space to the message. This way, time is only counted starting from the button

press. The code was tested on the circuit with success: the morse code machine was showing the

translated message in real time! We bought the Bluetooth adapter during the break.

Time of milestone 5 and 6

|  |  |  |
| --- | --- | --- |
| Start of project | 2/12/2021 |  |
|  |  | Project brainstorm |
| Project idea submitted | 2/26/2021 |  |
|  |  | Code brainstorming |
| Circuit brainstorming |  |  |
|  |  | Hardware needed was finalized |
| Initial build of circuit completed | 3/26/2021 |  |
|  |  |  |
| Video submitted | 4/2/2021 | Updated project doc submitted |
|  |  |  |
| Presentation at expo | 4/23/2021 |  |
|  |  |  |
|  | 4/30/2021 | Final design doc due |

List of materials

Diagram

Hardware you will need:

* Arduino (any will do)
* Bluetooth adapter (if you want PC connectivity or smartphone connectivity since you cant use the USB port)
* 2 buttons
* LCD
  + A potentiometer (for the lcd)
  + 220-ohm resistor for the lcd
* 2 10k ohm resistor (for the buttons)
* Jumper cables (at least 15 maybe more if you want Bluetooth connectivity)
* breadboard

The following is a piece of code of the Node struct and the MorseCode class definition.

This is one of the class methods. It moves on to the next letter, update the length of the message,

and resets the current node.

void MorseCode::dot() {

if (curr->left != nullptr) {

curr = curr->left;

}

message[len - 1] = curr->c; // update the last character

}

void MorseCode::dash() {

if (curr->right != nullptr) {

curr = curr->right;

}

message[len - 1] = curr->c; // update the last character

}

References

Morse code. (n.d.). Retrieved February 27, 2021, from https://www.britannica.com/topic/MorseCode

Ozer, B. (2020, April 06). How tree data structures help us Understand morse code. Retrieved

April 02, 2021, from <https://medium.com/swlh/how-tree-data-structures-help-usunderstand-morse-code-a95f6f7f2219>