INF351H: Fall/Winter 2022-23

Design Reflection #1

Mingjia Chen

For the first design project, our group created an Arduino mainframe machine for a special purpose: Morse Code presentation. The design idea started with one of our group members Quang. Quang was interested in how a machine could demonstrate lyrics with rhythms and lights in a way that people with hearing disabilities could enjoy when a song is played. The Morse codes machine seems an applicable challenge; our primary plan for the design is to translate input lyrics into Morse code, and then the RGB LED light will flash accordingly to dots and dashes. The machine is considered an arithmetic counting machine since it counts a list of Morse codes and delivers them by using visual signals. After building foundational codes and functions, approaches for extra functionality would further be done.

Potential Users of the Machine

Since the machine is designed for people with potential hearing disabilities, visual demonstration of the Morse code will be one of the essential functions of the machine. The machine is expected to deliver explicit indications of the Morse code. RGB LED lights and the lengths of flash pauses were chosen as the primary indicators of the Morse codes.

Coding of Morse Code Translation

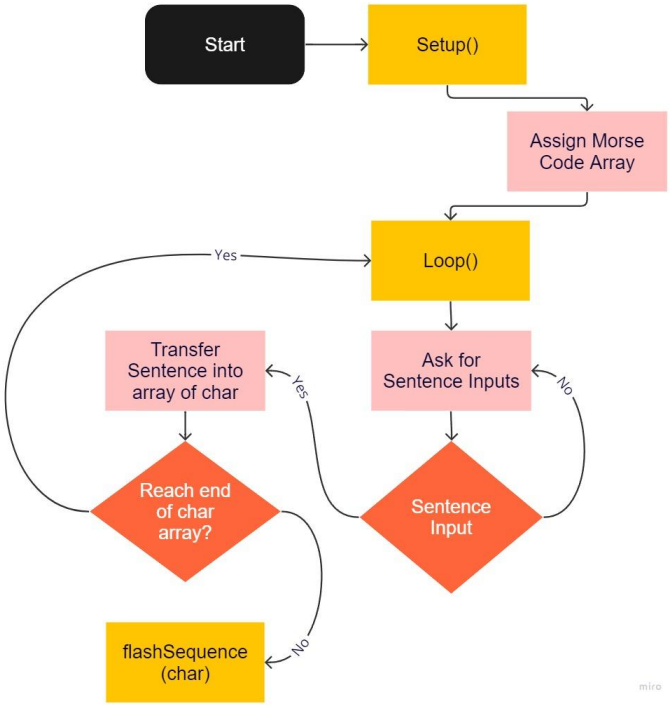
Based on past knowledge of the C programming language, Arduino language should be able to read and go through the text file to create a Morse code dictionary. Our team's first attempt was to create a text file with letters and their corresponding Morse codes, then the program should be able to read through them and come up with a Morse code dictionary for the machine. The first attempt was, unfortunately, time-consuming to fix and not worth the effort.

After brainstorming and Arduino educational projects (Arduino Education, 2023), we came up with a simpler version of translation, which ended up as our foundational logic for programming the machine. A string array is considered critical for Morse code translation. At the setup of the program, an array of Morse strings is assigned. The index numbers are the same as their corresponding letters' order in alphabetic order (starting with 0). The array is shown below:

```
//Morse code recorded for letters
char* letters[] = {
".-", "-...", "-.-.", "-..", ".", "..-.", "--.", "....", "..", // A-I
".---", "-.-", ".-..", "--", "-.", "---", ".--.", "--.-", ".-.", // J-R
"...", "-", "..-", "...-", ".--", "-..-", "-.--", "--.."};// S-Z
```

The machine waits until it gets a string-type object input and turns it into an array of individual characters, so we can go through them one by one and turn them into Morse codes. All the letters of the input will be capitalized since Morse codes are not case-sensitive. Due to the fact that each letter has its numbers in the ASCII system, the program is capable of calculating the index of each character in alphabet order by minusing the first letter “A”. We can then find the according Morse codes by using the index.

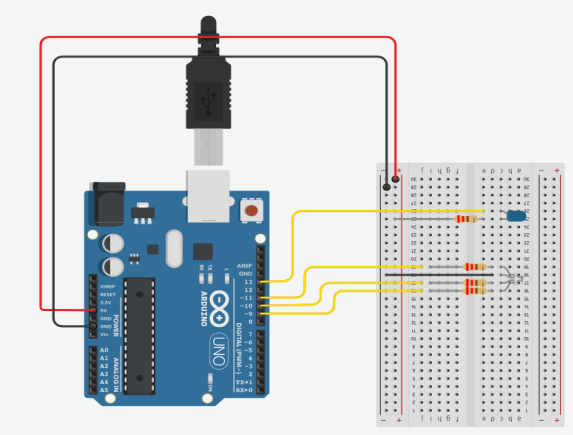
The Morse code will then be sent to a function called flashSequence(), in which LED lights flash accordingly to dots and lines. In addition to that, the machine waits longer for any possible space. After each letter’s morse code is played, trying to be more clear, we decided to add an extra blue LED light to tell our users that the machine reached the end of a letter’s morse code.

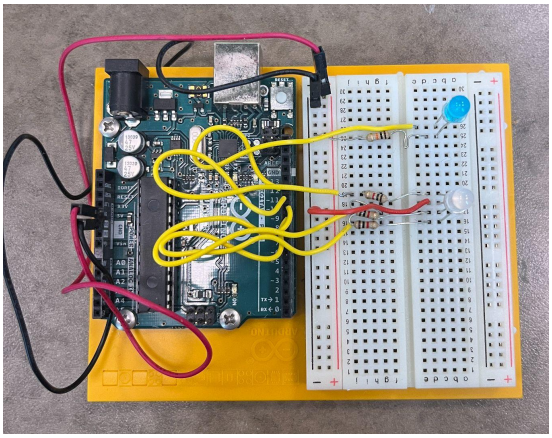


Flowchart Demonstrating the Process of the Program

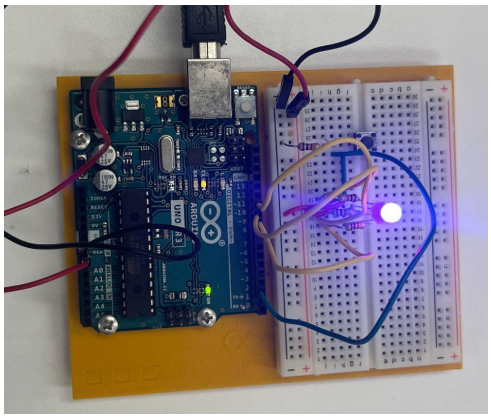
Wiring The Broad: Trials and Errors

Overall, the foundational wiring of the board is relatively easy. The blue LED light is connected to portal 13 and is responsible for all the pauses between letters. PWM pins 9, 10, and 11 are set for outputs and designed to deliver different values of Red, Green, and Blue to the RGB LED light, allowing us to change the colors accordingly. The broad design is straightforward; however, we went through multiple trials and errors for possible improvements until we reached the final design.



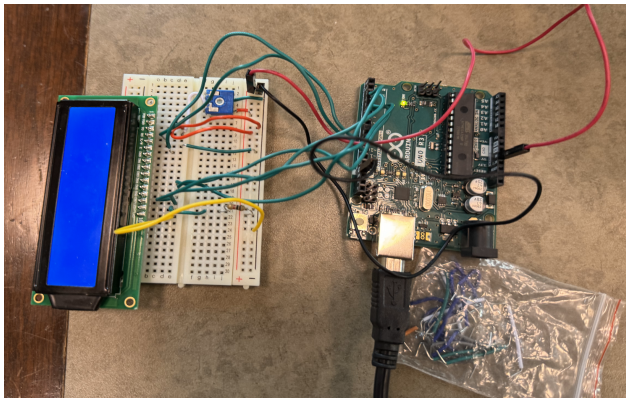


To induce better user experiences and the board's interactive feature, a button was implemented. The purpose of the button was to let users choose whenever they wanted to start the flashing of the Morse Code series. The photo below demonstrates the broad design trials. The button failed to function, due to possible interferences with other wirings and program timings. Storing the Morse codes beforehand could solve the problem; however, the group decided that a button was not worth spending more time on, since its improvement of user experiences overall is mediocre.



After consulting Professor Wang, the group decided that an LCD screen with an external keyboard might be a suitable approach for a better user experience. Potential users will then be able to type their inputs and have an LCD screen for feedback. Connecting a keyboard to the board would require a large amount of wiring and time, thus, we planned to have it connected directly to the computer. On the other hand, additional wirings for LCD were made for this approach (In the picture below). With further improvements in wiring and codes, LCD could become a feature of the machine, contributing to better user experiences. Unfortunately, the LCD screen did not work at the time; due to the limited time we had for machine designing, we chose to give up on further developments and debugging of this approach.

We look forward to using LCD screens for other upcoming machine-designing projects.



Lastly, decorations for improving user experiences and aesthetics were applied. A cardboard box was modified to both create a darker environment for the light and hide the broad. Hiding the broad helps protect from being accidentally knocked over. Tinfoil was used to cover the inside of the box to amplify the light.

Reflections

Much knowledge we obtained during the class helped us to achieve the successful building of this Morse code machine. For instance, the essential foundation of the machine is based on the use of arrays, which is a data type mentioned in the lectures. The array type works as a dictionary, allowing us to transfer sentences into corresponding Morse codes. Additionally, If statements, For loops, and functions were helpful for the machine to loop through each individual character and their Morse codes in a particular sentence. Materialwisely, the RGB light is a typical transistor we used for the assignment, allowing us to amplify and switch electrical signals according to dashes and dots.

As mentioned, the machine serves the purpose of counting. The machine as well serves the purpose of information processing: it uses meaningless information, such as LED's changes of colors, flashes, pauses, and letter's ASCII values, to create semantic information by assigning them the characteristics of the framework of Morse codes. For instance, green flashing lights with long pauses become meaningful since they are assigned to represent slower dashes in Morse codes. The physical medium such as lights, flashes, and pauses of the LED lights of the Arduino machine helps meaningful information, which is the lyric sentence we prompted, to be expressed. The LED signals, therefore, became a physical transition of language information. Hence, the machine is not only a counting machine but also a computing machine. As the blue light flashes, indicating the end of the Morse code for a certain letter, we receive the letter as information. After the whole process, as the blue LED light flashes multiple times, we collect enough information elements. Those information elements combined allow us to gain knowledge of the whole sentence.

Although the machine completed its tasks for prototype demonstration, there still are limitations in terms of its functions. Unfortunately, there are not many features designed for enhancing Human-computer interaction (HCI). As mentioned in the design and build sections, a button, an LCD screen, and an external keyboard were used as attempts to increase HCI; those

plugins could be implemented in the future for HCI improvements. As a result, users are only able to interact with the machine remotely through terminals (in this case, our laptops). When the machine starts, there are not many possible interactions with the running machine other than re-entering a prompt for restarting. Those are the characteristics of mainframe computers. Unlike supercomputers, the machine can only process input sentences by going through its letters one by one manually, rather than being optimized for parallel processing and numerical simulations. The processing speed thus is slower. The machine moreover only serves a special purpose: it is designed for a pre-defined set of tasks specifically for processing sentences and words into Morse codes light shows.

For future directions, other than HCI features, the machine's characteristics serve its supposed purpose: the machine does not need to be a general-purpose computer or supercomputer. Features regarding HCI could be speculated for supplemental user experiences. For now, the machine is not independent: its behaviors are based on any general-purpose computers (laptops, etc.) that are capable of using high-level coding languages. Ideally, future developments in this machine could allow users to play Morse code light shows automatically when they play a song, instead of manually typing in sentences. Due to limitations and expectations of the Arduino tool kit, it will continue to function depending on computers for now; however, in the possible future, the machine could be able to respond independently based purely on vocal information from songs.

Bibliography

Arduino Education. (2023). Morse Code Project. https://www.arduino.cc/education/morse-code-project/