EE 451

COMMUNICATION SYSTEMS II

FALL 2023

P2

Morse Code Translator (Arduino UNO + Bluetooth Module)

Students:

Boğaçhan Ali Düşgül

Cem Tolga Münyas

Ozan Gülbaş

Department of Electrics & Electronics Engineering

Izmir Institute of Technology

February 25, 2024

Contents

1 Objective **2**

2 Background **3**
- 2.1 Encoding . . . 3
- 2.2 Decoding . . . 3
- 2.3 Wireless Communication using Bluetooth . . . 4

3 System Model & Method **5**
- 3.1 Creating The Block Diagram of The Project . . . 5
- 3.2 Hardware Setup . . . 5
- 3.3 Software Setup . . . 7
  - 3.3.1 Morse Code Encoding Algorithm . . . 7
  - 3.3.2 Morse Code Decoding Algorithm . . . 7

4 Results & Discussion **10**

5 Conclusion **11**

1 Objective

In the beginning of the communication history, Morse code has played a significant role in transmitting messages over long distances [1]. Morse code uses short and long signals, called dots and dashes to represent letters, numbers, and special punctuation characters. Samuel Morse, an American scientist, helped develop it to send messages over long distances using wires, radios, sound, or light[2]. Messages are sent by tapping, buzzing, clicking, or flashing lights. In this project, we aim to design a Morse code translator inspired by these processes.

The primary objective of our project is to design and implement a Morse code translator using Arduino UNO and the Bluetooth device (HC-05). The system aims to provide a user-friendly interface for inputting Morse code through button presses which enables users to communicate with wireless communication using the International Morse code alphabet. The translator which is our computer in this case, detects the dots and dashes inputted by the user and converts them into their corresponding letters, numbers, and special punctuation characters in real time. We use Bluetooth technology to enhance the effectiveness of the system in terms of wireless communication functionality which allows users to send input signals from various distances with a range of 10 to 30 meters.

To summarize the main objective and the scope of this project, our system detects the Morse code data sequence which consists of 6 bits at a maximum of dots and dashes that are generated from the user input through the button clicks. While the long press on the button generates a dash, the short press on the button generates a dot. This generated input data sequence is given directly to the microcontroller and the corresponding decoded message is displayed on the serial port interface. We have also used an LED to give visual feedback to the user every time the button is pressed.

Here is the International Morse Code alphabet. The user inputs are the data sequence, which consists of dots and dashes. The data sequences are then sent through the Bluetooth interface, and we can observe the corresponding letter, number, or special character in the serial port display as output in real time.

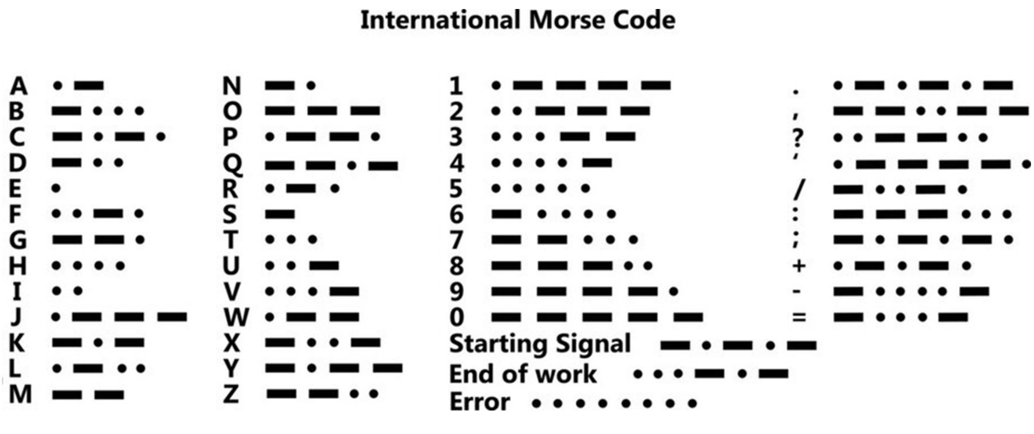


Figure 1: International Mors Code Alphabet

Through this project, we aim to provide a concrete and interesting solution that demonstrates the ability of Arduino-based systems with Bluetooth interface to close the gap between conventional communication techniques and contemporary technology.

2 Background

The theoretical background of Morse Code Translator can be framed within the context of communication theory, particularly focusing on encoding and decoding mechanisms as well as integrating Bluetooth wireless communication into the whole system.

Morse code is a symbolic representation of the alphabet letters, numbers, and special characters using a series of dots and dashes to represent each character in one string. This coding scheme was used in telegraphy in the history before and known as a simple and robust method of communication.

2.1 Encoding

In the encoding phase of our project, the user provides input in the form of dots and dashes through button presses. Each "dot" serves as the basis of time for the code. One dash is equivalent to the length of three dots. After each character, there is a silence that is equivalent to the length of one dot. Each dot or dash corresponds to a Morse code symbol. These Morse code symbols are then encoded with the rule of short button press and long button press. While a short button press represents a dot, a longer button press represents a dash. In other words, the encoding process translates the user's physical input into a sequence of Morse code symbols, which creates the intended message.

One of the essential processes of encoding is symbol mapping. In this phase, the created symbols are mapped to their respective letters, numbers, and special characters. Symbol mapping establishes a bond between the input signals and the corresponding output symbols.

In the final phase of encoding, the Arduino UNO microcontroller processes the user input signals and executes the encoding mechanism. In this way, the Morse code symbols are created through the translation of button presses from the user.

2.2 Decoding

In the decoding process, the Bluetooth device HC-05 receives encoded Morse code symbols wirelessly. The received Morse code symbols are interpreted by the Arduino UNO microcontroller. To identify certain dots and dashes, this interpretation requires understanding the duration and pattern of signals.

The decoding process translates these Morse code symbols back into their corresponding letters, numbers, and special characters based on the predefined symbol mapping which we used strings for that process.

The final phase of decoding is generating the output message through the serial port of our computer and displaying it. To achieve this, we used an external tool called "Processing" [3] which has the same functionality as Arduino IDE so that our decoding code converts the decoded Morse code symbols into a string of characters (human-readable format). Then, this output is displayed on the serial port of our computer on either COM4 or COM5.

2.3 Wireless Communication using Bluetooth

Bluetooth operates as a communication protocol that enables devices to communicate (transmit and receive data) wirelessly.

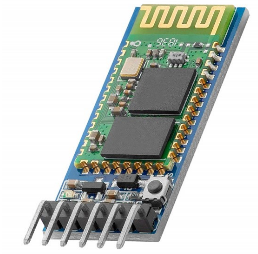


Figure 2: HC-05 Bluetooth Serial Transceiver Module [4]

In our project, we have used an “HC-05 Bluetooth Module Serial Transceiver” for the transmission of generated Morse code signals and the translated text between the Arduino UNO–based Morse code translator and our computer via Bluetooth protocol. Since we use wireless connectivity between transmitter and receiver, the user could be able to input the data sequence from a range of up to 30 meters of distances[5]. As a result, Bluetooth eliminates the need for physical connections between the hardware setup and our computer.

To summarize, user inputs through button presses (combination of dashes or dots) are converted into the Morse code symbols during the encoding phase. Then, those encoded symbols are transmitted through the Bluetooth interface to our computer. Then, our computer receives those encoded symbols via Bluetooth and decodes them in order to reconstruct the original message and displays it in the Processing App’s display screen. In this way, user could be able to input the desired data sequence with our hardware prototype wirelessly via Bluetooth transmission and observe the translated Morse code text on the Processing App’s display screen.

3 System Model & Method

3.1 Creating The Block Diagram of The Project

We constructed a block diagram for the project that helped us determine the components of the project, what is the expected behavior and what are the connection between the project components. The system behavior we designed using the block diagram is as follows: Have the switch state as input, give the switch's state as visual feedback using LED, interpret and encode the input data using Arduino, and send the data using HC05 Bluetooth Module through the serial port, interpret and decode the received data through Bluetooth serial port using Processing App, and display the decoded data using Processing App.

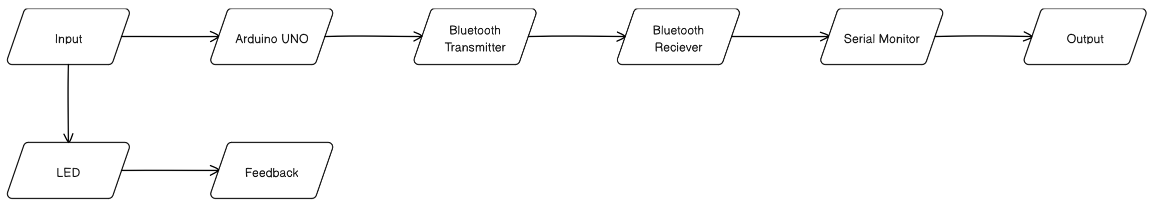


Figure 3: Block Diagram of the Project

3.2 Hardware Setup

After determining the project's outline and necessary electronic components, we drew the circuit schematic and prototyped it on a breadboard with the testing scripts [6] that check if a certain component works as intended that we prepared. The needed components to execute the desired system behavior of the project are:

- Arduino UNO R3
- HC05 Bluetooth Module
- 7.4V LiPo Battery
- Micro Switch
- LED
- 220Ω Resistor

Arduino is used as a main controller of the transmitting side in which the LED, Switch, and Bluetooth Module are controlled. Arduino also has the encoding algorithm software inside that interprets incoming signals from the switch and encodes them to Morse Code messages. The HC05 Bluetooth Module is used to send data wirelessly from the transmitting side to the receiver side which is the Processing App working on a computer. The 7.4V LiPo Battery is used to power Arduino and the transmitting side consequently, for mobile usage of the device. The Micro Switch is used as the input data for the Morse Code encoding algorithm and the LED is used for visual feedback of the device status. The 220Ω resistor is used to configure the Micro Switch input as a pulldown button.

When we were satisfied that the prototyping setup was working as expected, we transferred the circuit onto a copper plate with holes to solder components. We made the necessary connections by soldering wires, connectors, isolators, and pins using soldering iron.

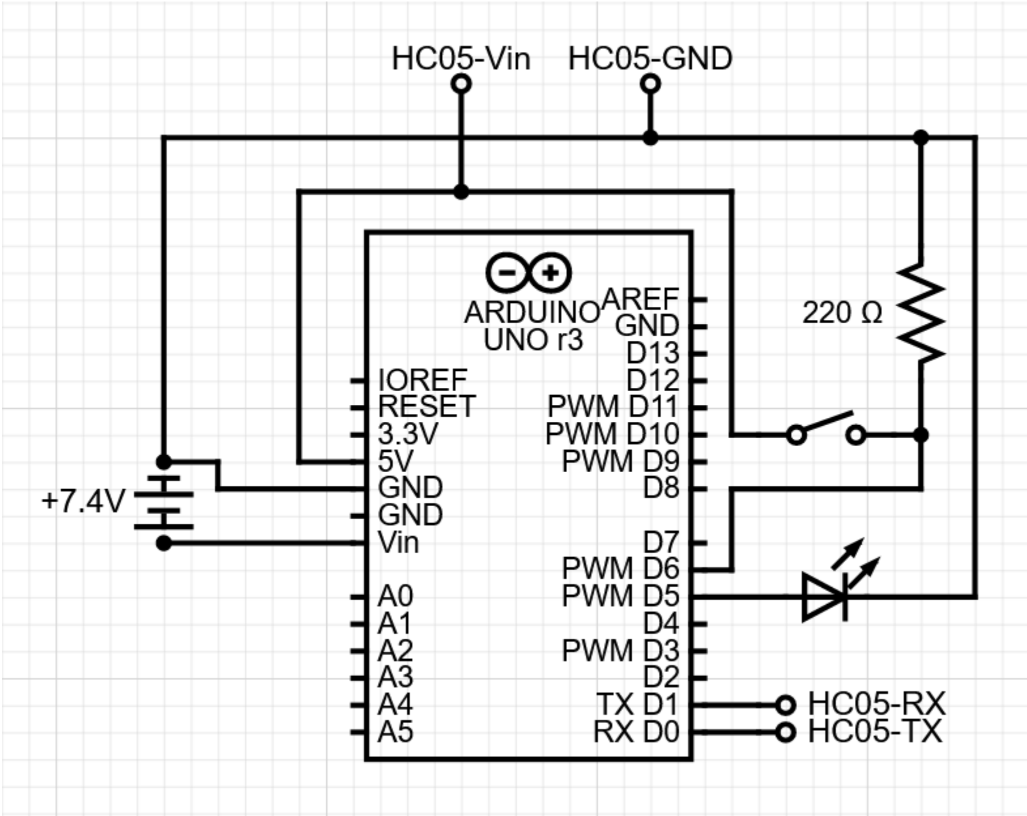


Figure 4: Circuit Schematic of the Project

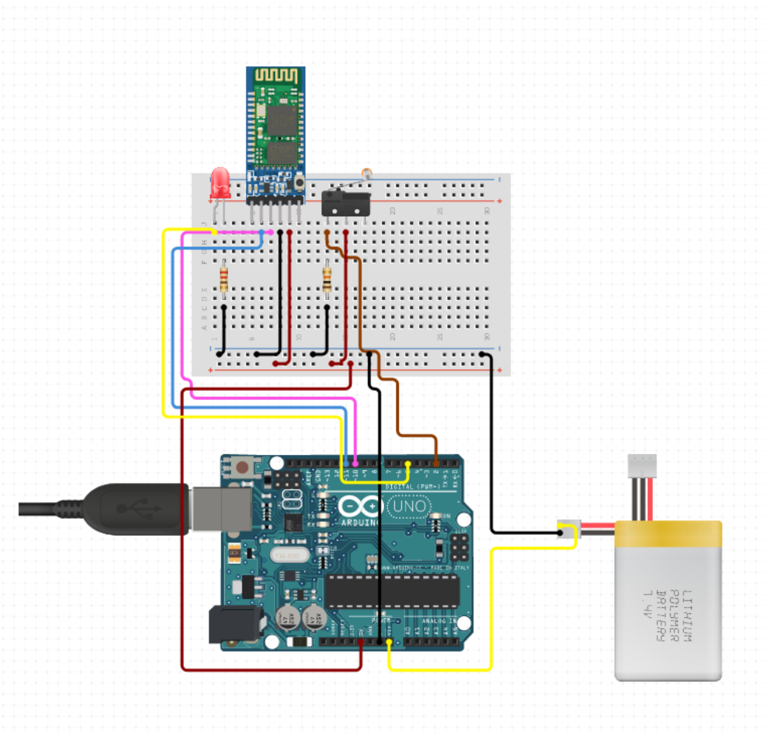
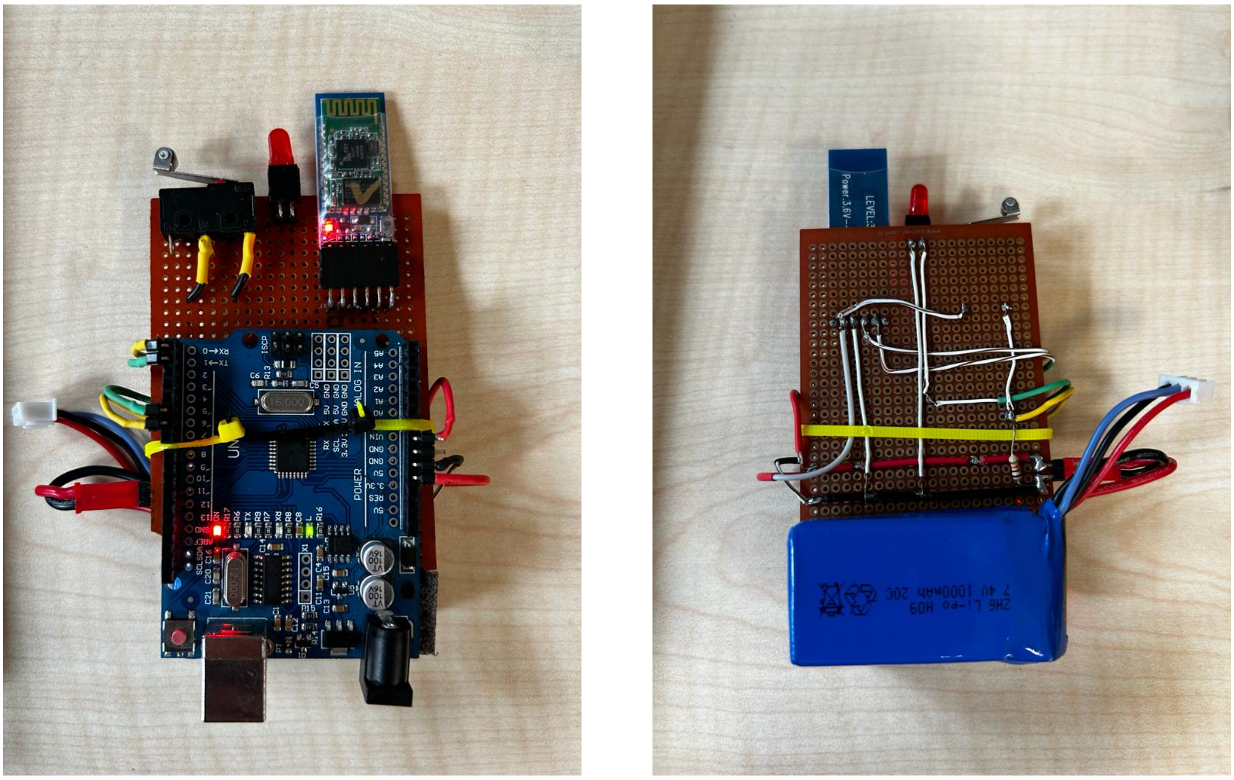


Figure 5: Prototyping Circuit



(a) Front (b) Back

Figure 6: Morse Code Translator Device

3.3 Software Setup

The system contains two different algorithms which runs on different devices. One is the encoding algorithm that runs in the Arduino and handles the encoding of the switch inputs to Morse alphabet and writing the encoded data to Serial port. The other one is the decoding algorithm that runs in the receiving device using Processing App and handles the decoding of the received data from the Serial port and displaying it.

3.3.1 Morse Code Encoding Algorithm

The encoding algorithm's fundamental task is to listen to switch input and save the duration of how long the switch is pressed to one of the 6 variables which are named as registers. As we can see from the flow chart (Figure 7) at the begging of the loop we check whether the switch is pressed or not, if pressed start the counting function to save how long the switch is pressed and write the value to the least significant empty register. After each write, start the reset counting function to determine the message symbol length which means, after a predetermined period of time (approximately one second in the demo) if the switch is not pressed then the symbol length of the message is equal to the amount of used registers. If the reset count function is reached its threshold value, then the data is sent by writing to the Serial port and all of the registers are resetted.

3.3.2 Morse Code Decoding Algorithm

The decoding algorithm is pretty straightforward, it only reads the Serial port for incoming data and by looking at data length it decodes received Morse Coded string to their correspond-

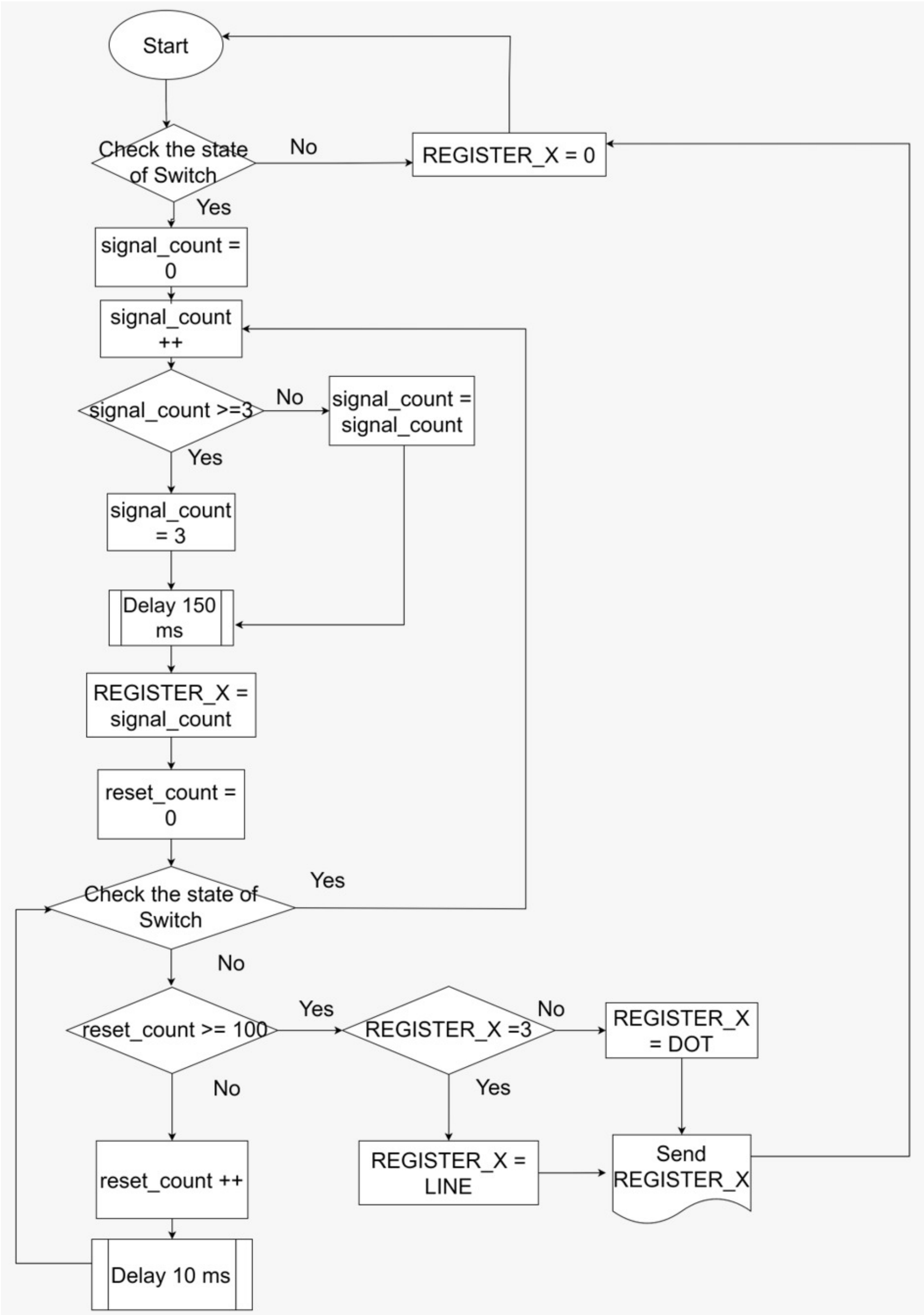


Figure 7: Flow Chart of Morse Code Encoding Algorithm

ing letters, numbers, symbols or commands. After decoding the received data, it also displays the output using the Processing App (Figure 7).

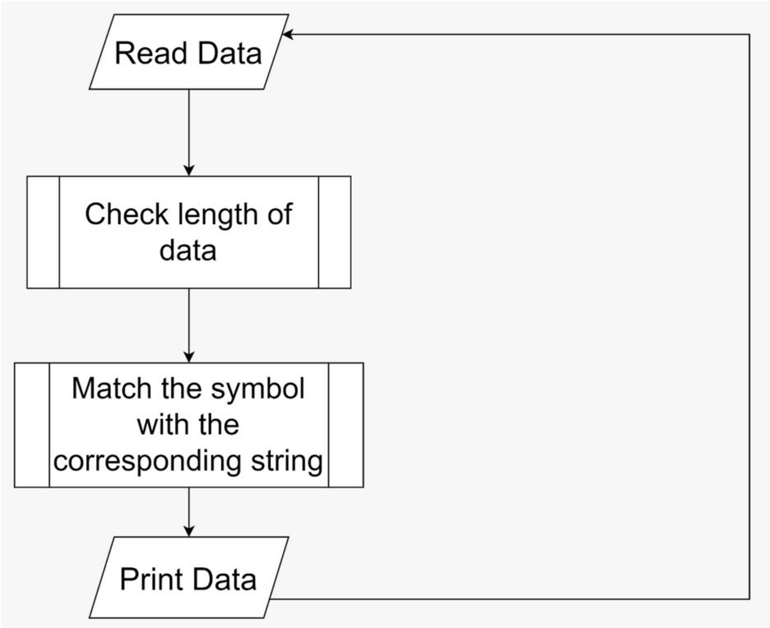


Figure 8: Flow Chart of Morse Code Decoding Algorithm

4 Results & Discussion

0 and 1 are expressed as Dots and Lines in the Morse code algorithm. The duration of holding down the key represented by a Dot is 1/3 of the Line. In this way, we find that the incoming data corresponds to the Dot or Line.

We can send letters, numbers and punctuation marks up to 6 bits in length. In this way, we can send 126 different symbols. Because 6 different lengths of data can be sent each time, when calculating the number of symbols, it must be calculated separately for each length situation. The number of symbols is calculated as $2^1 + 2^2 ... + 2^6 = 126$.

After converting the electrical signals into Dotes and Lines, we send this data to the computer via Bluetooth. We can send data up to 30 meters with the Bluetooth module we use.

The decode algorithm we wrote in the "Processing" program on our computer reads the data coming via Bluetooth, then looks at the length of the data and matches it with characters of the same length. In this way, we can decode our data consisting of Dots and Lines and turn them into meaningful strings.

In this way, we understood the importance of encode and decode operations for communication systems. We learned to encode and send the electrical signals received through the switch and to make the received data meaningful by decoding it.

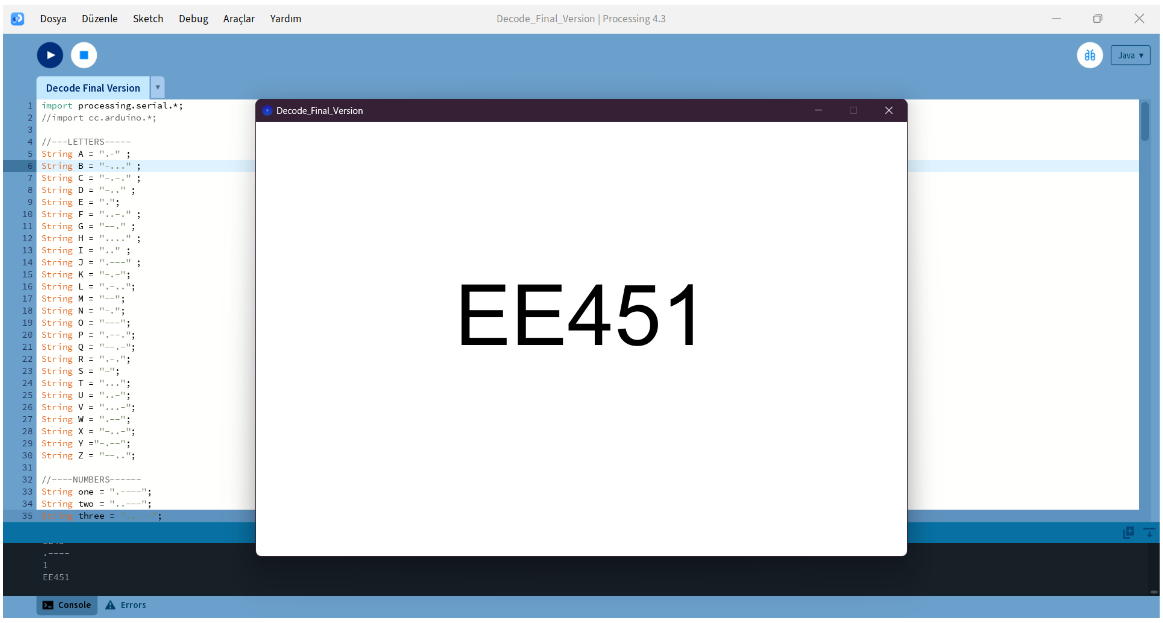


Figure 9: Screenshot from Receiving Device Showing the Decoded Data

5 Conclusion

In conclusion, the "Morse Code Translator using an Arduino UNO and a Bluetooth Module" project has successfully achieved its objectives. Our Arduino-based hardware system which is integrated with the encoding and decoding software algorithms provides a comprehensive exploration of Morse code communication through an Arduino-based and Bluetooth system. The project has also shown the integration of modern communication technologies such as Bluetooth, into traditional communication methods.

When we delve into the project from the communication perspective, we focus on the encoding and decoding processes. In order to summarize the encoding process, it basically translates the user inputs into Morse code symbols utilizing short button presses for dots and longer button presses for dashes.

On the other hand, the decoding process essentially translates these Morse code symbols back into their corresponding messages of different bit lengths which may be letters, numbers, and special characters. We encoded messages with up to 6 bits as a string. Then, according to the length of predefined strings for letters, numbers, and special characters, we characterized each string which was encoded as a configuration of dots and dashes. In this way, we translated the Morse code symbols back into their original corresponding message which are letters, numbers, and special characters in the International Morse Code Alphabet.

One of the collateral duties of this project is to integrate the wireless Bluetooth technology into our hardware so that the user can control and generate input data sequences at a distance. It means that the system does not have to have a wired connection. Bluetooth technology enables seamless communication between the Arduino-based Morse code translator and the external device which is our computer.

In the system model that we designed; Bluetooth is used as a key component since it provides the flexibility for receiving decoded messages without physical constraints when user inputs through the button presses. In other words, Bluetooth interface enhances the operations of encoding and decoding. As we discussed in previous parts, in encode phase the user input is translated into Morse code symbols, which are 6 bits of data transmission at maximum for one symbol. Then, through the Bluetooth module (HC-05), the data is sent to our computer. When our computer receives those Morse code symbols which are the combination of dots and dashes via the Bluetooth interface, it translates them back into the corresponding messages which are letters, numbers, and special characters. From this perspective, our project shows that how Bluetooth technology makes a conventional Morse code translator more flexible and easier to use.

All in all, the integration of Bluetooth interface demonstrates how our project is aligned with contemporary communication paradigms and its significance in communication systems.

References

[1] "Morse code & telegraph: Invention & samuel morse," HISTORY. (Aug. 12, 2022), [Online]. Available: https://www.history.com/topics/inventions/telegraph (visited on 01/02/2024).

[2] T. English. "Origins of the morse code and how it works." (Mar. 17, 2020), [Online]. Available: https://interestingengineering.com/innovation/origins-of-the-morse-code-and-how-it-works (visited on 01/02/2024).

[3] "Reference," Processing. (), [Online]. Available: https://processing.org//reference (visited on 01/02/2024).

[4] A. K. Ravi. "How to use HC-05 bluetooth module with arduino." (), [Online]. Available: https://www.c-sharpcorner.com/uploadfile/167ad2/how-to-use-hc-05-bluetooth-module-with-arduino/ (visited on 01/02/2024).

[5] "HC-05 master / slave bluetooth module." (), [Online]. Available: https://www.sgbotic.com/index.php?dispatch=products.view&product_id=1939#:~:text=The%20HC-05%20is%20a,becomes%20transparent%20to%20the%20user (visited on 01/02/2024).

[6] NoahB5. "How to set up and test arduino bluetooth connection," Instructables. (), [Online]. Available: https://www.instructables.com/How-to-Set-Up-and-Test-Arduino-Bluetooth-Connectio/ (visited on 01/02/2024).