

Morse encoder/decoder

Author: Andrei Pintilie - Student at the Computer Science Departament,

Technical University of Cluj-Napoca

E-mail: [✉ Pintilie.Lu.Andrei@student.utcluj.ro](mailto:Pintilie.Lu.Andrei@student.utcluj.ro)

TABLE OF CONTENTS

- Project Overview
-
- Project Features
 - Mode Selection
 - Morse Encoder
 - Morse Decoder
-
- Additional Resources
-

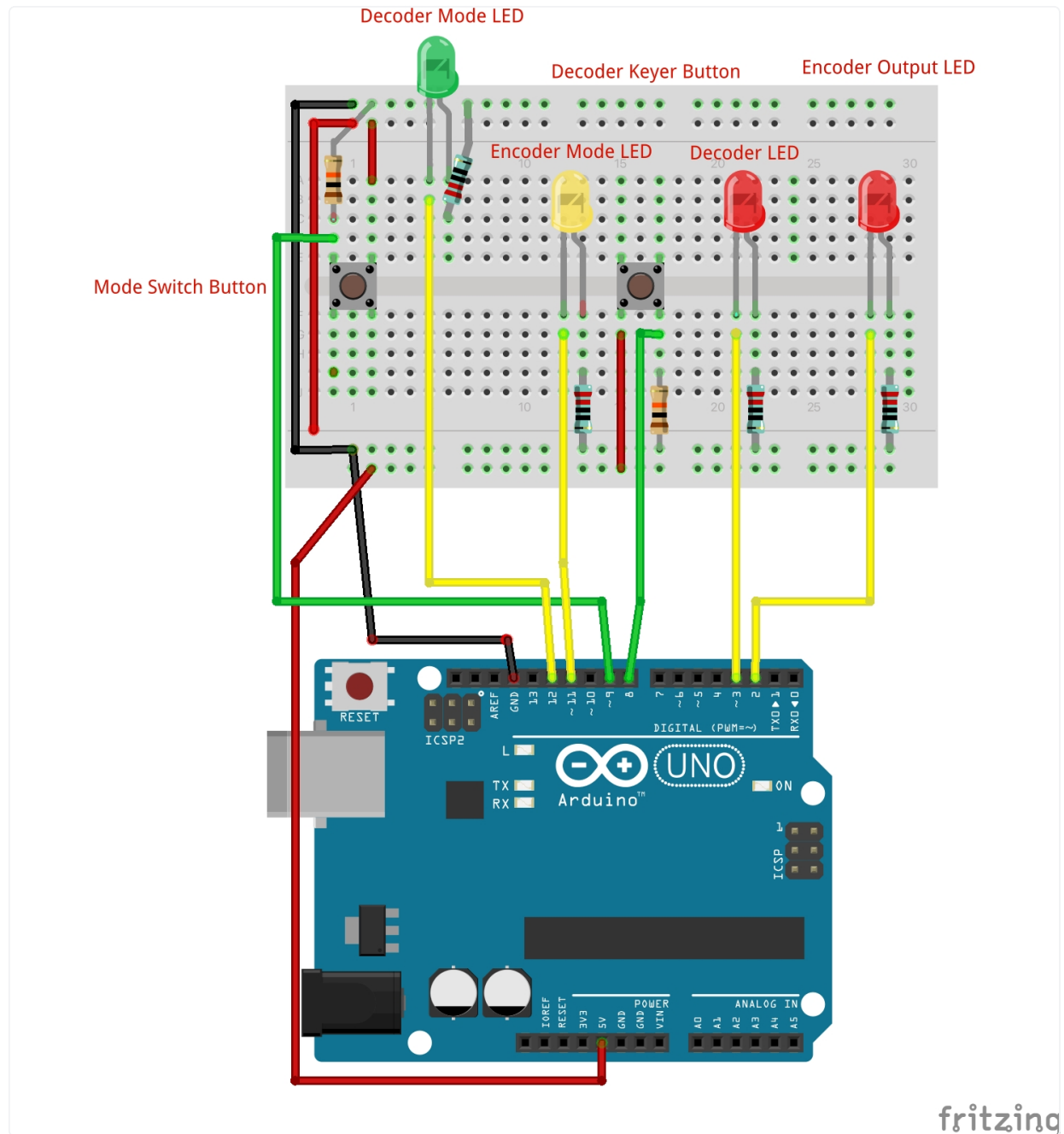
Project Overview

Simple Morse encoder/decoder implementation using an Arduino UNO R3 microcontroller and several off-the-shelf electronic components.

Materials used:

- Arduino UNO R3 microcontroller
- 2 x red LED
- 1 x green LED
- 1 x yellow LED
- 4 x 220Ω resistors
- 2 x push-buttons
- 2 x 10kΩ resistors
- red, black, green and yellow wires

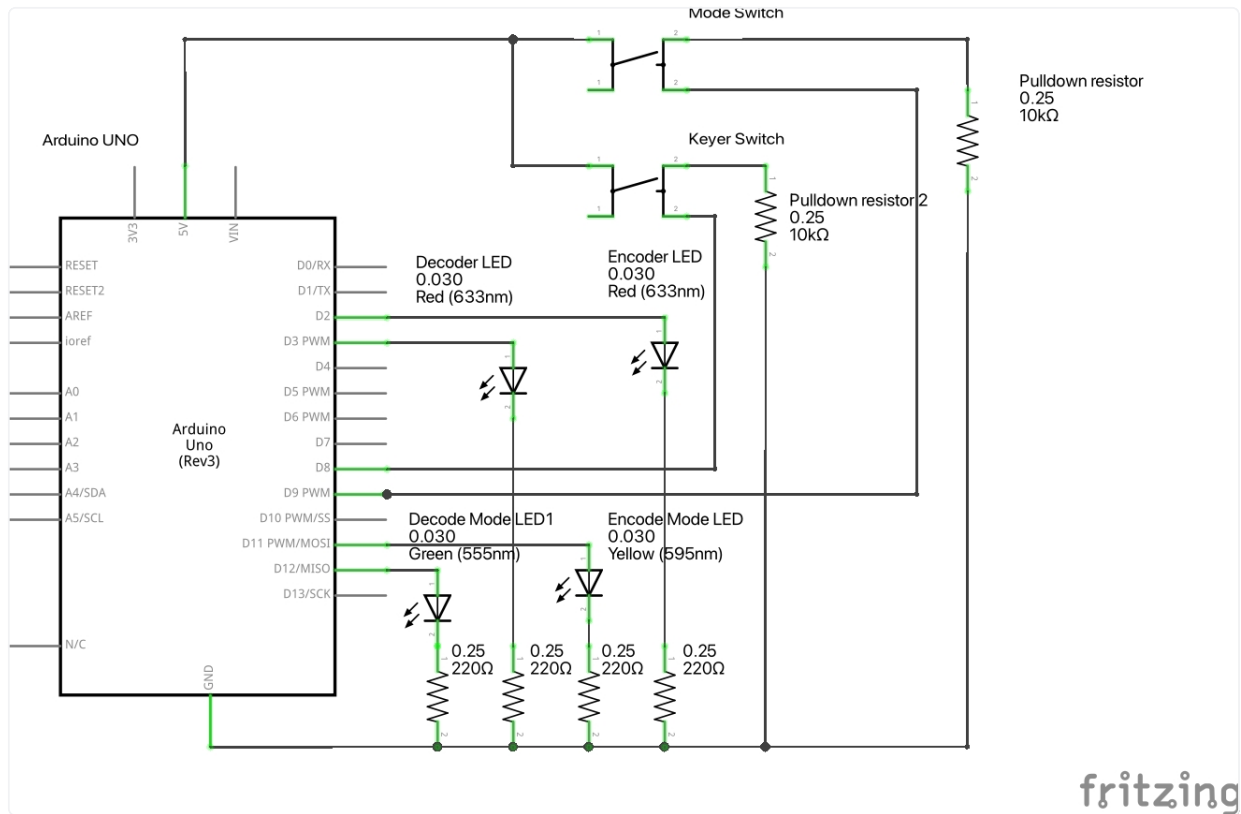
Project requires connection between Arduino microcontroller and a serial monitor to function



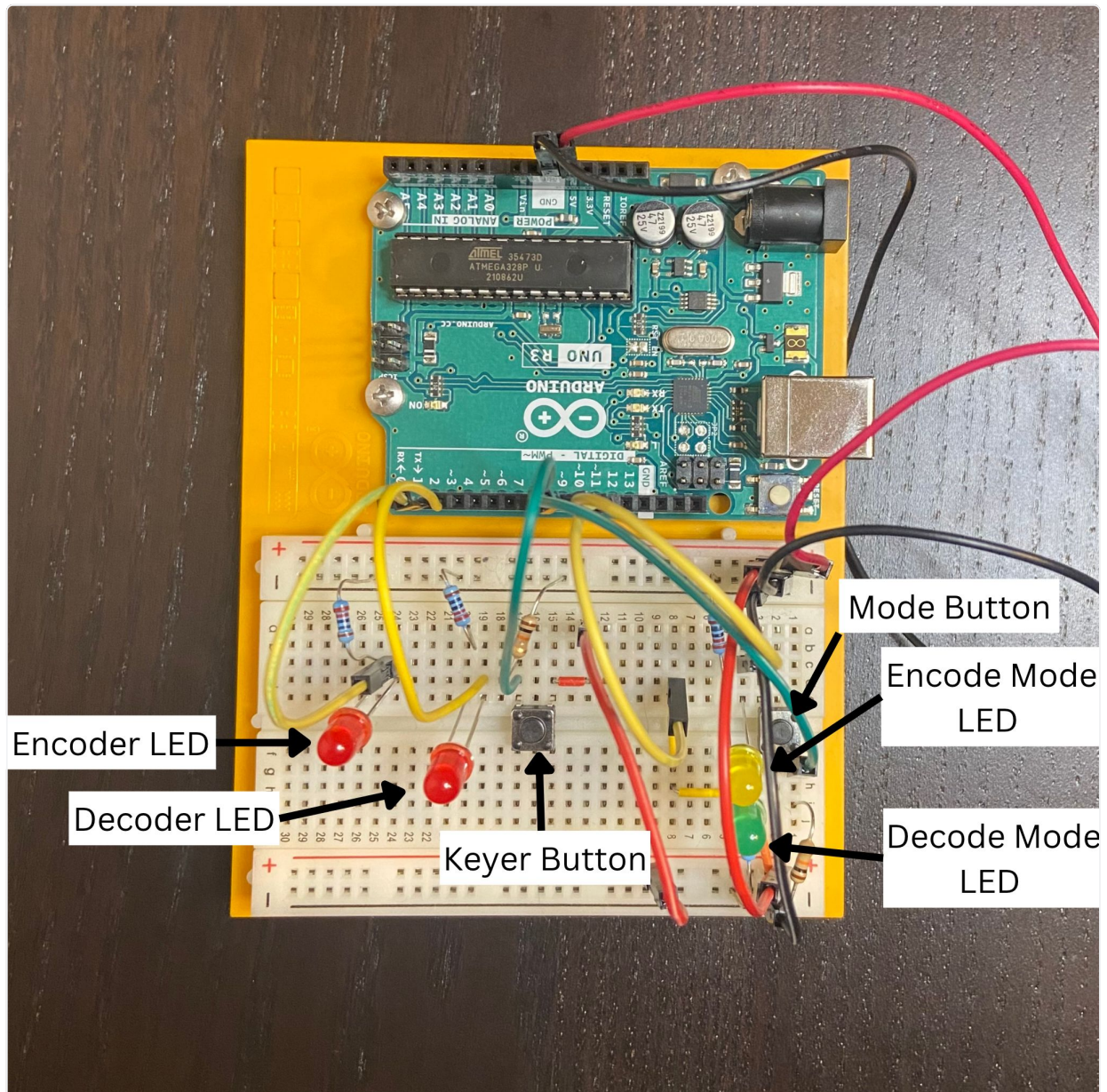
Breadboard layout

Legend:

- green wire - input
- yellow wire - output
- red wire - 5V
- black wire - GND
- decoder LED - used for visual verification of push-button input



Circuit schematic



Real-life implementation

Project Features

Mode Selection

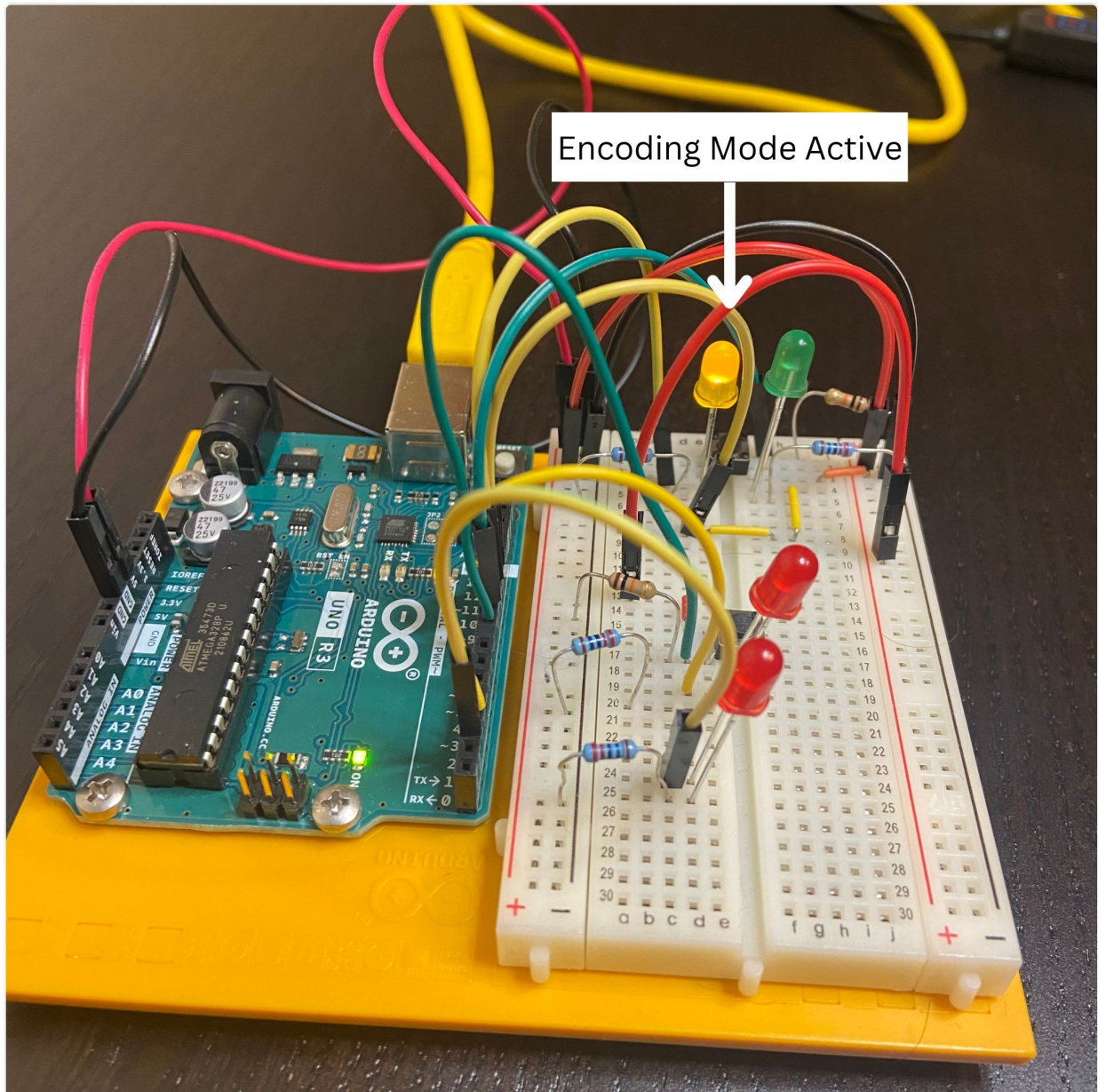
The project features a push-button used to select the functioning mode (either encoding or decoding) along with 2 LEDs used to indicate the current selected mode (yellow for encoding, green for decoding).

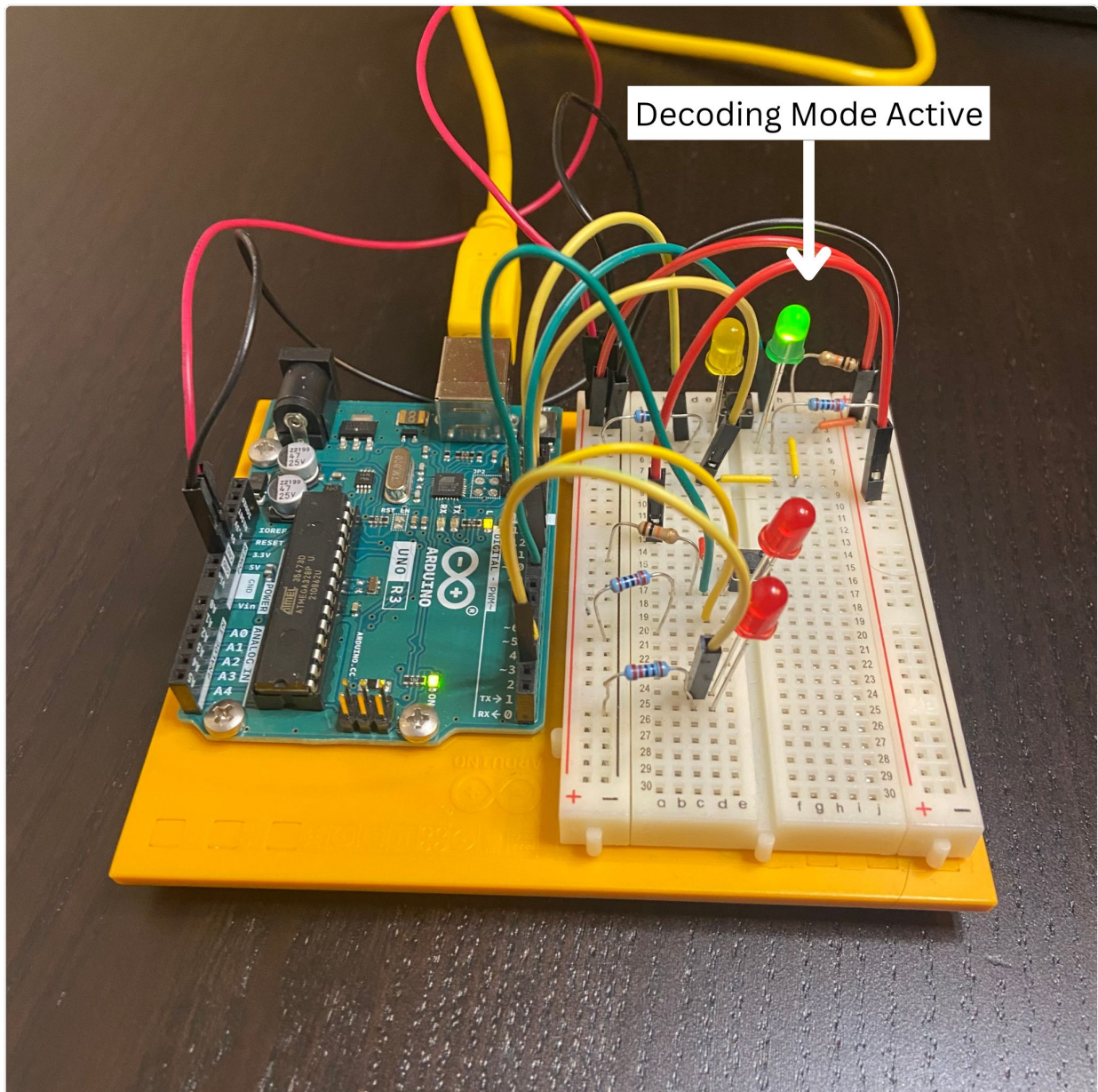
To compensate for any physical defects in the button, the input is debounced through software by ignoring any input after the rising edge for a short period of time (50ms)

```
1 modeButtonValue = digitalRead(modeButtonPin);
2 if ((millis() - lastDebounceTime) > debounceDelay) {
```



```
3   ///  
4   lastDebounceTime = millis();  
5 }
```





Morse Encoder

The encoder works by taking a message through the serial port (in this case aided by the computer connected to the board), converting it to a series of dashes and dots and finally flashing the message on the red output LED, producing a visible Morse sequence.

The following code snippet presents the function used to display a character (dot or dash) on the output LED.

```
1 void flashDotOrDash(char dotOrDash) {  
2     digitalWrite(encoderOutputLedPin, HIGH);  
3     if (dotOrDash == '.') {  
4         delay(dotDelay);  
5     } else // must be a -
```



```

6    {
7        delay(dashDelay); //dashDelay = 3 * dotDelay
8    }
9    digitalWrite(encoderOutputLedPin, LOW);
10   delay(dotDelay);
11   }

```

Morse Decoder

The decoder works by taking input from a push-button, measuring the time the button stays pressed and converting that signal to dots and dashes. These are translated to letters which are then printed to the serial monitor.

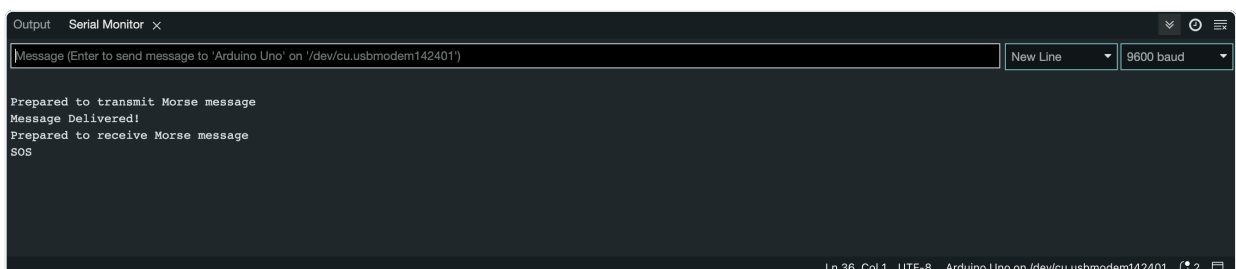
The following code snippet presents the function used for translating button presses to dashes or dots.

```

1  char convertSignal() {
2      if (noiseThreshold < signalLength && signalLength < cutoffTime)
3          return '.';
4      else if (signalLength >= cutoffTime)
5          return '-';
6  }

```

The serial monitor output of the demonstration video in the next section is the following:



The screenshot shows the Arduino Serial Monitor window. The input field contains the text "Message (Enter to send message to 'Arduino Uno' on '/dev/cu.usbmodem142401')". The output shows the following sequence of messages: "Prepared to transmit Morse message", "Message Delivered!", "Prepared to receive Morse message", and "SOS". The status bar at the bottom indicates "Ln 36, Col 1 UTF-8 Arduino Uno on /dev/cu.usbmodem142401".

Additional Resources

Complete project code available at:



GitHub - apintilie12/arduino-morse-encoder-decoder: Arduino Morse encode...

<https://github.com/apintilie12/arduino-morse-encoder-decoder>

Short demonstration video available at:

<https://youtu.be/PiValhgYrI4>



<https://youtu.be/PiValhgYrI4>

This document also available online at:

<https://apintilie12.slite.page/p/7gFyEIJ-RMV6Oz/Morse-encoder-decoder>