## Objectives:

Morse code is a method of transmitting text information as a series of on-off tones, lights, or clicks that can be directly understood by a skilled listener or observer without special equipment. It is named for Samuel F. B. Morse, an inventor of the telegraph.

Morse code is used by some amateur radio operators, although knowledge of and proficiency with it is no longer required for licensing in most countries. Pilots and air traffic controllers usually need only a cursory understanding. Aeronautical navigational aids, such as VORs and NDBs, constantly identify in Morse code. Compared to voice, Morse code is less sensitive to poor signal conditions, yet still comprehensible to humans without a decoding device. Morse is, therefore, a useful alternative to synthesized speech for sending automated data to skilled listeners on voice channels. Many amateur radio repeaters, for example, identify with Morse, even though they are used for voice communications. In an emergency, Morse code can be sent by improvised methods that can be easily "keyed" on and off, making it one of the simplest and most versatile methods of telecommunication. The most common distress signal is SOS or three dots, three dashes, and three dots, internationally recognized by treaty.

## System Design:

There are different types of Morse code for some different countries. We are using International Morse Code for our project. A chart of International Morse Code signals and their representative values is given below:


**Fig-1: International Morse code Chart**

Each Morse code symbol represents either a text character (letter or numeral) or a 'prosign' and is represented by a unique sequence of dots and dashes. The duration of a dash is three times the duration of a dot. Each dot or dash is followed by a short silence, equal to the dot duration. The letters of a word are separated by a space equal to three dots (one dash), and the words are separated by a space equal to seven dots. The dot duration is the basic unit of time measurement in code transmission.

Before going into details of the procedure, it's better to observe the circuit diagram first. Our circuit diagram is given below:

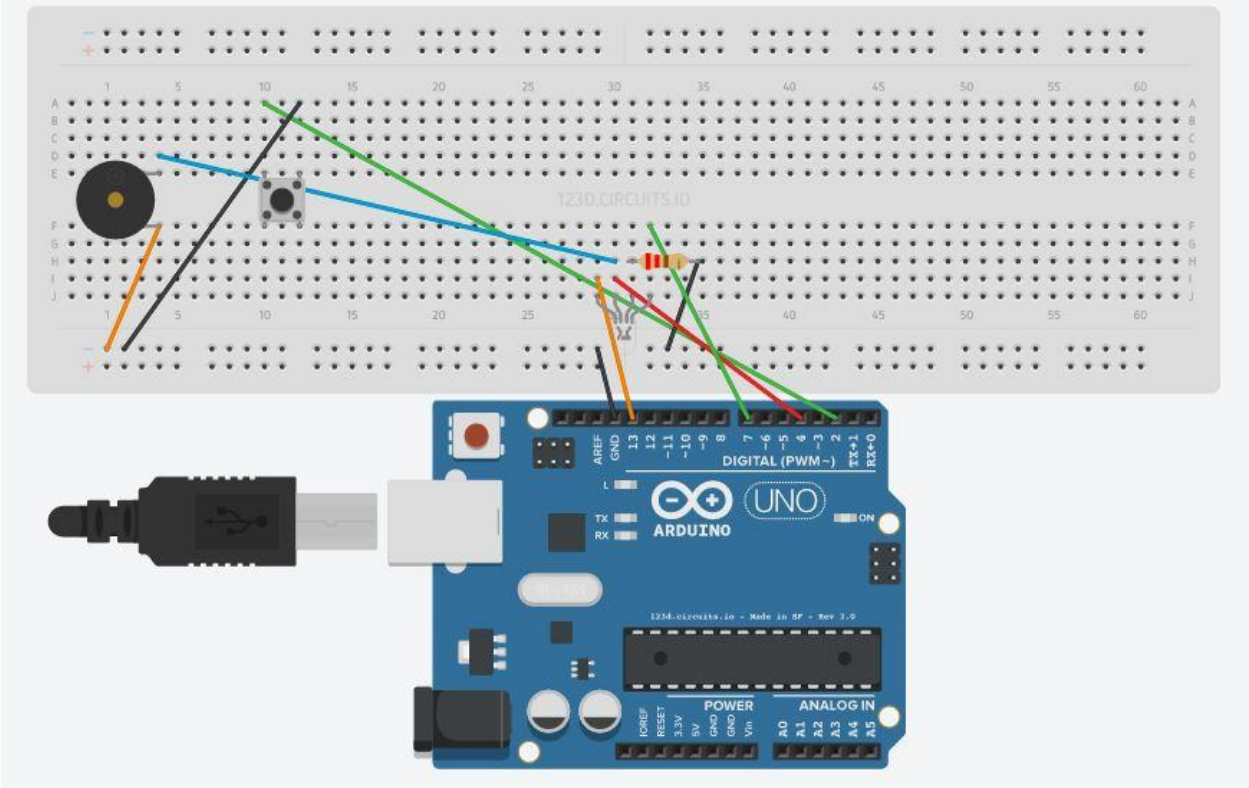


**Fig-2: Circuit diagram for Morse code decoder**

From the circuit diagram it is observable that we're using:

- An Arduino Uno.
- A push switch.
- A LED light(RGB).
- A Buzzer.
- A Resistor.

## Source Code:

```
unsigned long signal_len, t1, t2;
int inputPin = 2;
int ledPin = 4;
int ledPin2 = 7;
int ledPin3 = 13;

String code = "";

void setup()
{
   pinMode(ledPin3,OUTPUT);
   Serial.begin(9600);
   pinMode(inputPin, INPUT_PULLUP);
   pinMode(ledPin, OUTPUT);
}

void loop()
{
   digitalWrite(ledPin2, HIGH);
   digitalWrite(ledPin3,HIGH);
   delay(50);
   digitalWrite(ledPin3,LOW);
NextDotDash:
   while (digitalRead(inputPin) == HIGH) {}
   t1 = millis();
   digitalWrite(ledPin, HIGH);
   while (digitalRead(inputPin) == LOW) {}
   t2 = millis();
   digitalWrite(ledPin, LOW);
   signal_len = t2 - t1;
   if (signal_len > 50)
   {
      code += readio();
   }
   while ((millis() - t2) < 500)
   {
      if (digitalRead(inputPin) == LOW)
      {
         goto NextDotDash;
      }
   }
   convertor();
}

char readio()
{
   if (signal_len < 600 && signal_len > 50)
   {
      return '.';
   }
   else if (signal_len > 600)
   {
      return '-';
   }
}

void convertor()
{
   static String letters[] =
   {
      ".-", "-...", "-.-.", "-..", ".", "..-.", "--.",
      "....", "..", ".---", "-.-", ".-..", "--", "-.",
      "---", ".--.", "--.-",".-.", "...", "-", "..-",
      "...-", ".--", "-..-", "-.--", "--..", "E"
   };
   static String numbers[] =
   {
      "-----", ".----", "..---", "...--", "....-", ".....",
      "-....","--...", "---..", "----.", "E"
   };
   int i = 0;
   if (code == ".-.-.-")
   {
      Serial.print(".");
   }
   else if (code == "-.....")
   {
      Serial.print(" ");
   }
   else if (code == ".....-")
   {
      Serial.println(" ");
   }
   else
   {
      while (letters[i] != "E")
      {
         if (letters[i] == code)
         {
            Serial.print(char('A' + i));
            break;
         }
         if (numbers[i] == code)
         {
            Serial.print(char('0' + i));
            break;
         }

         i++;
      }
      if (letters[i] == "E"||numbers[i] == "E")
      {
         Serial.println("<Wrong input>");
      }
   }
   code = "";
}
```

## Working Procedure:

Dot and Dash time length is defined in Arduino code. We are using pulses for less than or equal 0.67 seconds as Dots and pulses having duration greater than 0.67 seconds as Dashes. From fig-2, we can see that every character has unique pattern of Dots and Dashes. By pressing the push switch for different time intervals, we give a pattern of Dot or Dash input to the circuit. Our input information is passed on Arduino. After Dots and Dashes, it passes the output (Decoded morse code) to Display unit. In our project we're using a Laptop for display device.

## Testing Results:

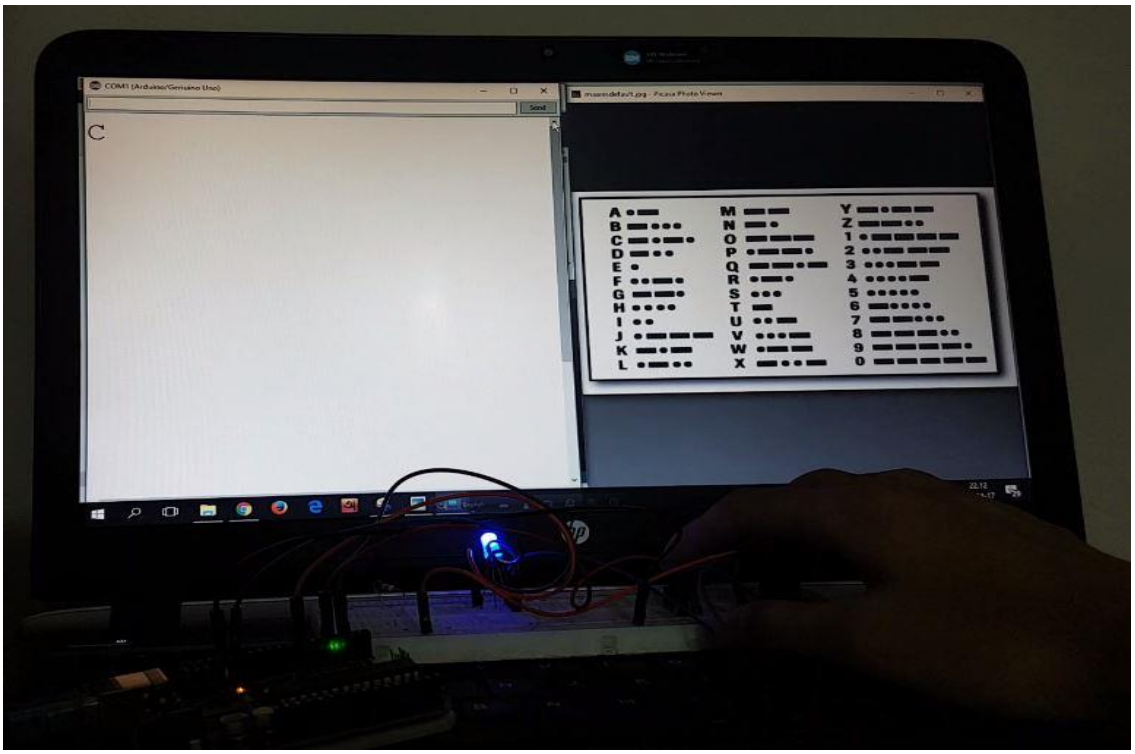


**Fig-3: Testing of the circuit.**

## Limitations:

- Needs months of practice to send accurate morse code.
- Long distance transmission requires constant electricity.
- All other medium are used in short distance.

## Practical use of the project:

- In case of national emergencies like Natural Calamities, Curfew, War etc. when major communication systems are disabled; our project can be used to establish a backup method of communication.
- We can also use this project in foggy weather by using light as medium.
- Morse code is still popular among amateur radio enthusiasts, although proficiency in Morse Code is no longer a requirement to obtain your amateur radio license.

## Conclusion:

Throughout this semester we learned about different types of microcontrollers. By doing this project we have learned how real life problem can be solved with the help of the microcontrollers. We also learned some simple simulations with the help of Arduino which may be helpful to our jobs or research.