Receptor de Código Morse com Display de 7 Segmentos e Arduino

Projeto implementa a recepção de **código morse** através do uso de botões e usa o **display de 7 segmentos** para mostrar o caracter equivalente ou a mensagem completa decodificada.

Cauã dos Santos , Leonardo Galvão , Mateus Hashimoto , Miguel de Paulo , Pedro Nassif , Thiago Ferreira

Código Morse

O Código Morse, desenvolvido em **1835** por **Samuel Morse**, é um **sistema** de comunicação **codificado** que utiliza uma combinação de **sinais curtos (pontos) e longos (traços) e espaço** para representar **letras, números e caracteres especiais**. Pode ser utilizado para transmitir mensagens por meio de sinais **sonoros** , **visuais**(luzes) e **elétricos** (radio), especialmente em situações onde a comunicação verbal **não é possível**. Ele foi muito utilizado nas guerras.

O código normalmente é **gerado** da seguinte forma

Tempo de Ponto

1 dit = 1 unidade de tempo

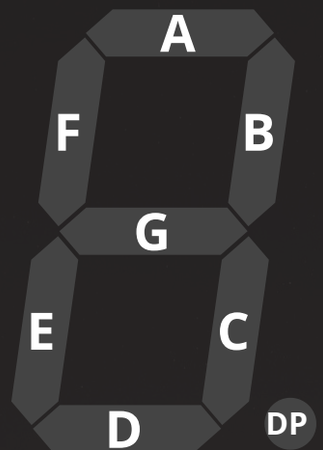
Tempo de Traço

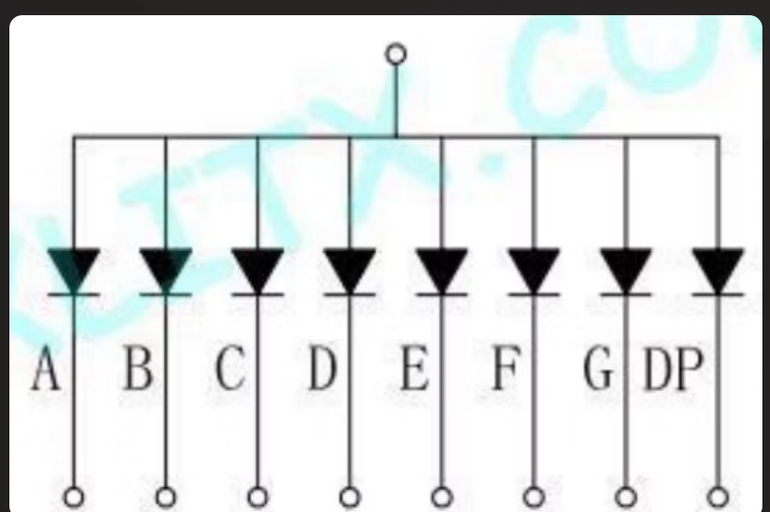
1 dah = 3 dits

Intervalos

- Entre sinais: 1 dit
- Entre letras: 3 dits
- Entre palavras: 7 dits

Display 7 Segmentos





MODEL: 5161BS

MODE: Common-Anode(CA)

Significa que os pins de segmentos(A-G, DP) estão conectados ao **catodo** e o pino do digito(common ou COM) esta no **anodo**. Para esse modo os pinos de segmento devem receber um valor de voltagem **LOW** para serem ativados.

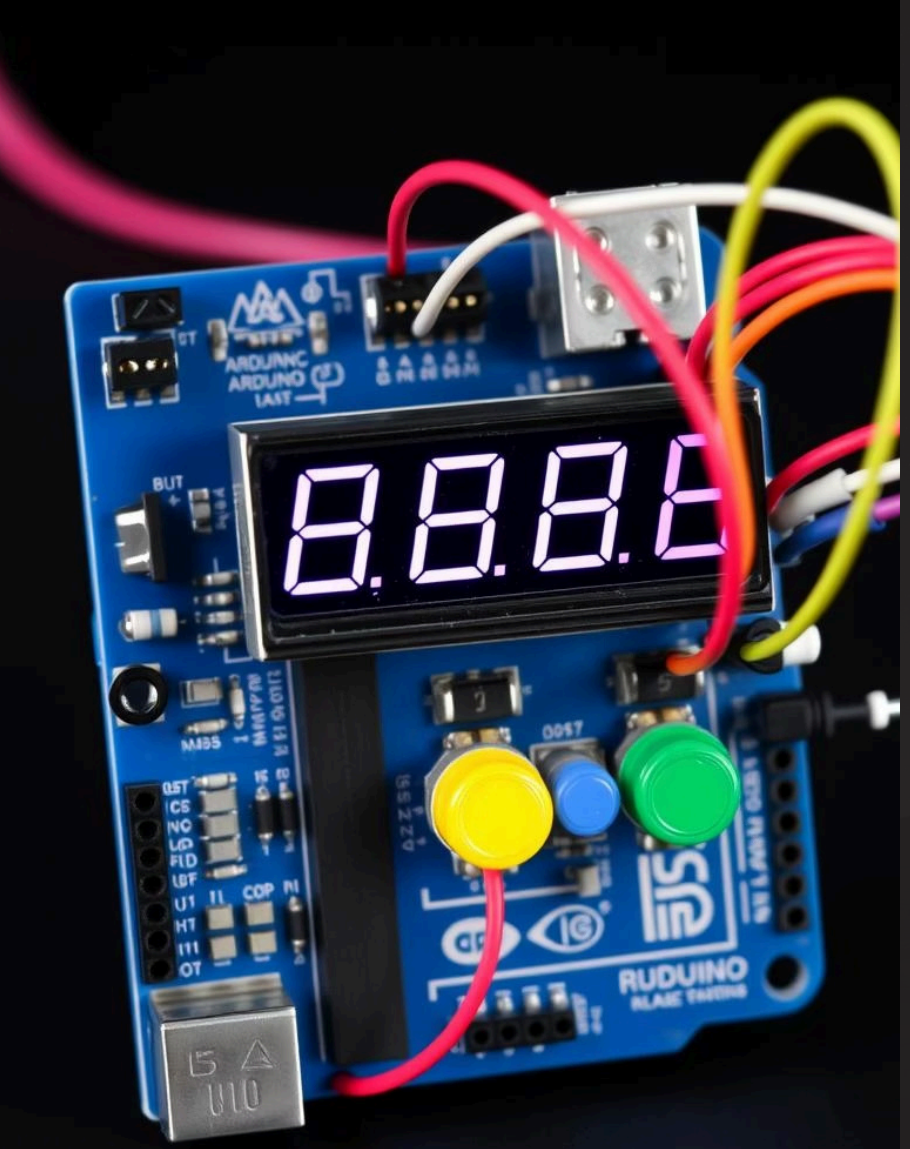
CONTINUOS FOWARD CURRENT PER DICE: 30mA

Indica a **corrente máxima** suportada por segmento, como a fonte de tensão usada no arduino é de **5V**, a resistência mínima se da por:

R = V/I = 5V/30mA = 166.67Ω

Por isso usa-se o **resistor** de 220Ω

fonte: datasheet 5161BS



Materiais Usados

Arduino Uno

1x

Display de 7 Segmentos

1x

Botão

3x

Resistor

3x10kΩ - botão

1x220Ω - display

Jumper

18x

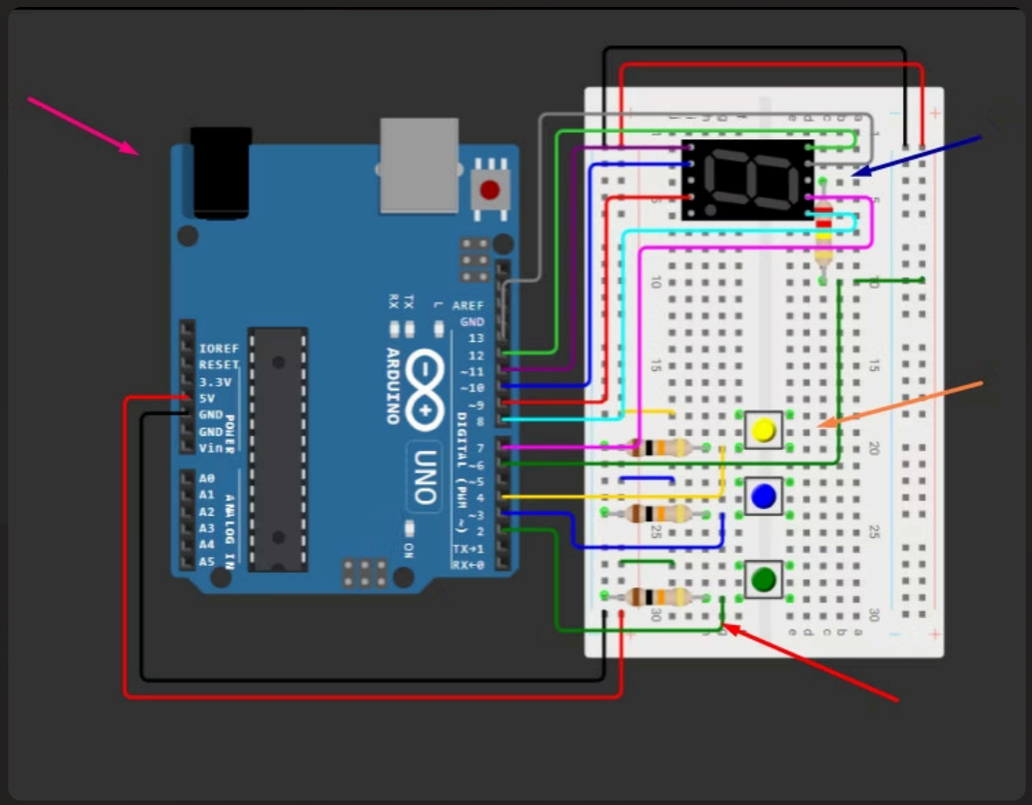
Montagem do Circuito

Arduino uno

Entradas Display
13,12,11,10,9,8,7

Conexão Placa
5v , GND

Entradas Botões
4,3,2



Display 7 seguimentos
Resistor 220Ω

3 Botões

3 Resistores 10kΩ

Função dos Botões

① Receptor de código Morse (MC)

Detecta a duração do clique (curto ou longo) e traduz como · (DIT) ou _ (DAH), **formando** o código Morse de uma letra.

Pode ser usado para **resetar** o buffer do código com um pressionamento longo

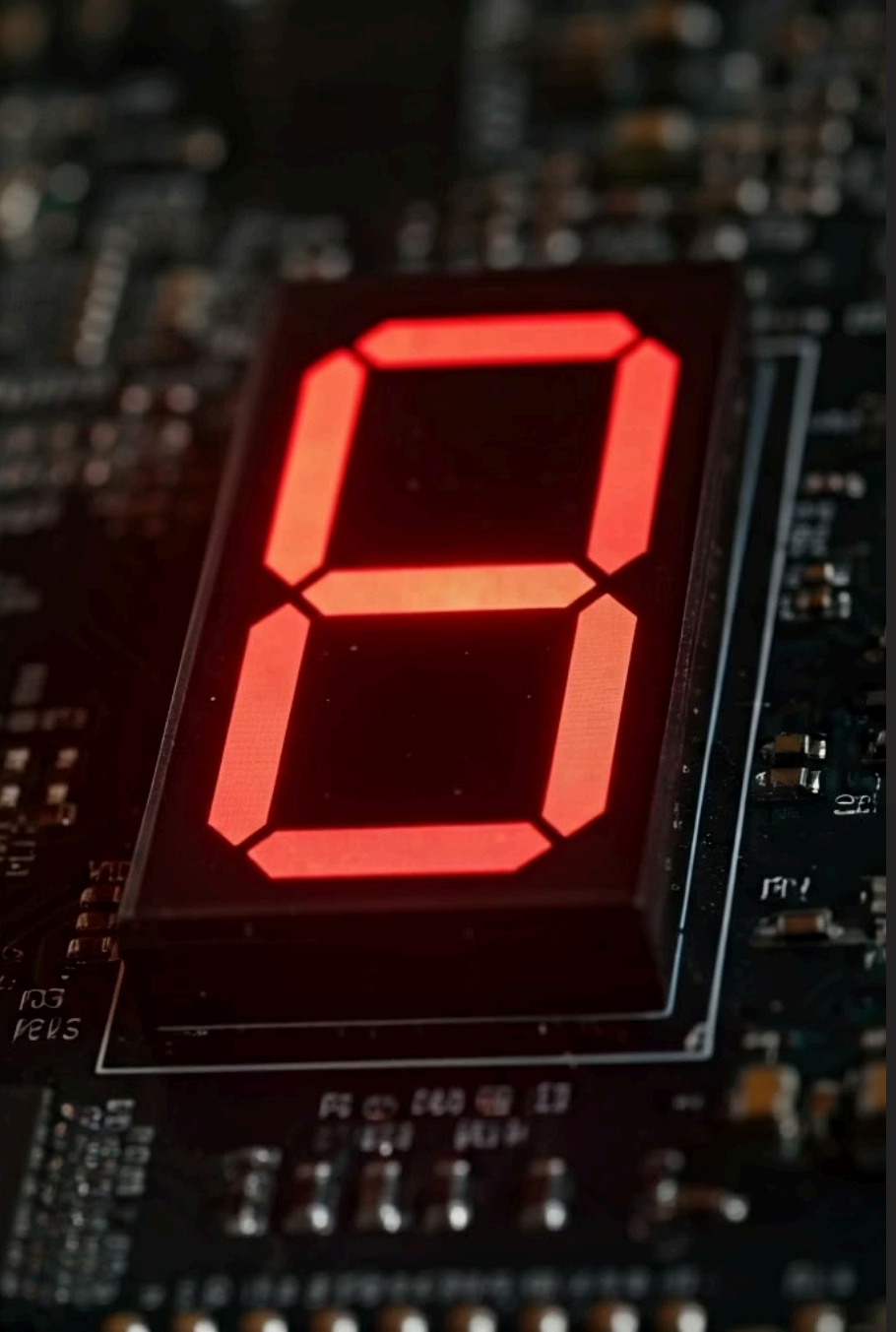
② Fim de Letra (EOL)

Indica o fim da adição de caracteres morse para a letra atual. O **código Morse** atual é então convertido para ASCII, se for reconhecido, e adicionado a mensagem

③ Fim de Palavra (EOW)

Indica o fim de uma palavra, adicionando um (1) espaço em branco na mensagem e a **imprimindo** no display.

Usado para imprimir a mensagem no display ou para **reseta-la** com um pressionamento longo



Função Main loop()

O sistema usa o **display de 7 segmentos** para exibir a letra reconhecida:

- Após o botão **EOW** ser pressionado e a letra ser decodificada, ela é enviada ao display.
  - A letra é **convertida** em sinais elétricos que acendem os segmentos apropriados, **formando** o caractere.
    - Isso **permite** **feedback visual imediato.**

Ideal para ajudar iniciantes a visualizar o resultado da entrada e aprender a escrever cógigo morse.

Função Main loop

```
void loop(){
  /*  Morse Code receiver Control */
  // Read a morse code letter
  if (digitalRead(button_MC) == HIGH && EOL == false){
    currentElement = dit_Or_Dah(countTimeButtonPress(button_MC)); // convert the current button press to an elment
    if (currentElement != '\0'){ // only add the element to the letter if it is a valid one
      morseCode += currentElement;
      Serial.print("current buffer: ");
      Serial.println(morseCode); // only print when the morse code is updated
    }
  }

  /*  EOL Control */
  // The EOL button is pressed, show the letter or the error message, then reset the morse code
  if (digitalRead(button_EOL) == HIGH && EOL == false){
    EOL = true;
    if (morseCode.length() > 0){
      currentElement = morseToASCII(morseCode);
      // If the morses code is valid, add its letter to the message
      if (currentElement != '\0'){
        message += currentElement;
        Serial.println("morse code [" + morseCode + "]: " + currentElement);
      }
      else{
        Serial.println("The code: " + morseCode + " has no match.");
      }
    }
  }
  // The button is released
  else if (digitalRead(button_EOL) == LOW && EOL == true){
    EOL = false; // reset EOL flag to accept new letters
    morseCode = ""; // reset the morse code
  }

  /*  EOW Control */
  if (digitalRead(button_EOW) == HIGH && EOW == false){
    EOW = true;
    message += ' '; // add a space to the word
    Serial.println(message);
  }
  else if (digitalRead(button_EOW) == LOW && EOW == true){
    EOW = false; // reset EOW flag
  }
}
```

Realiza o controle dos botões.

EOL: Flag de **controle** de bouncing do botão 2 e reset do código morse.

EOW: Flag de **controle** de bouncing do botão 3

\0: Abstração de um caracter "vazio", **indica** ao programa que nenhum caracter conhecido foi lido.

Melhorias Futuras

Caracteres Especiais e numéricos

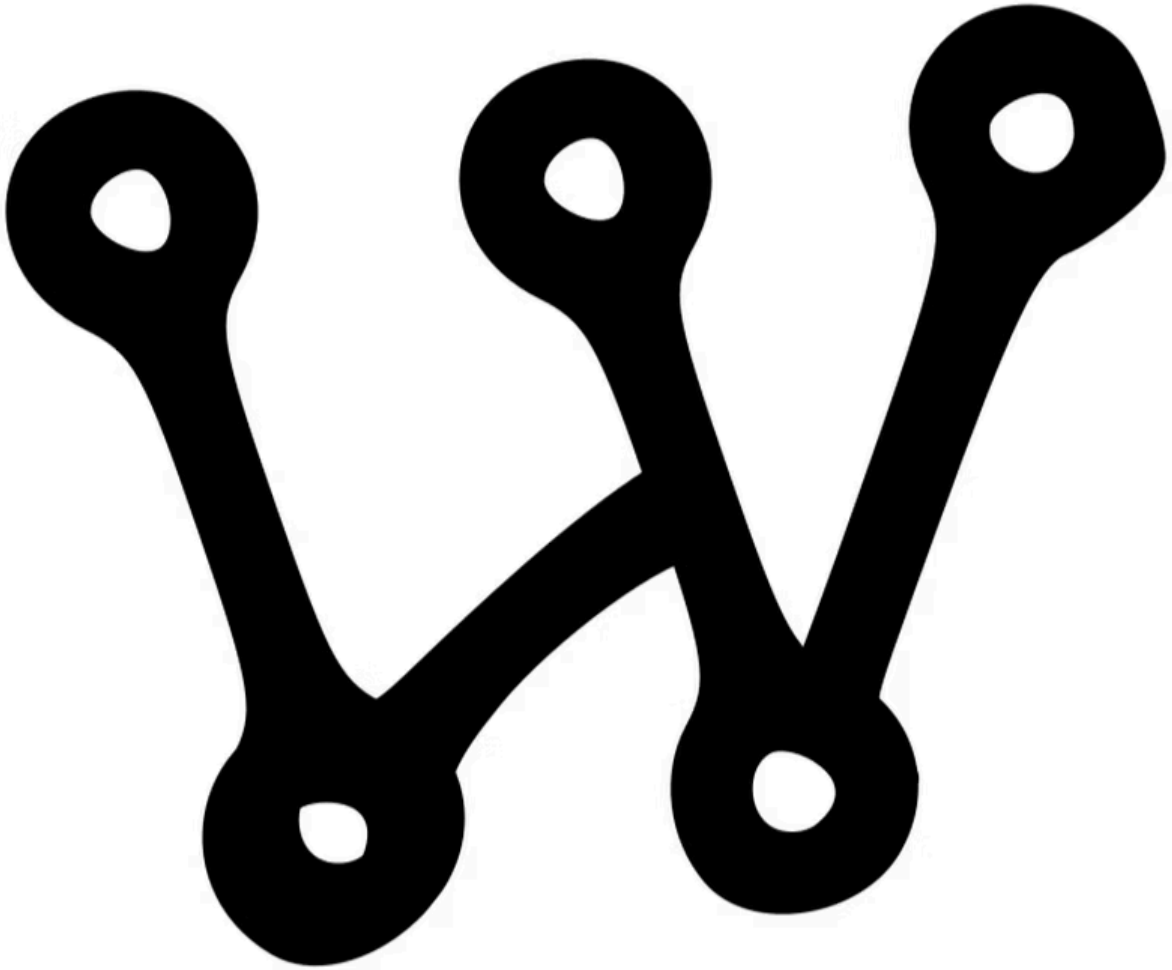
Incluir sinais para símbolos, pontuação e números.

Display LCD

Permitir mensagens longas com melhor leitura.

Buzzer

Para fazer a **tradução** sonora.



Demostração

Através do site wokiwi.com.