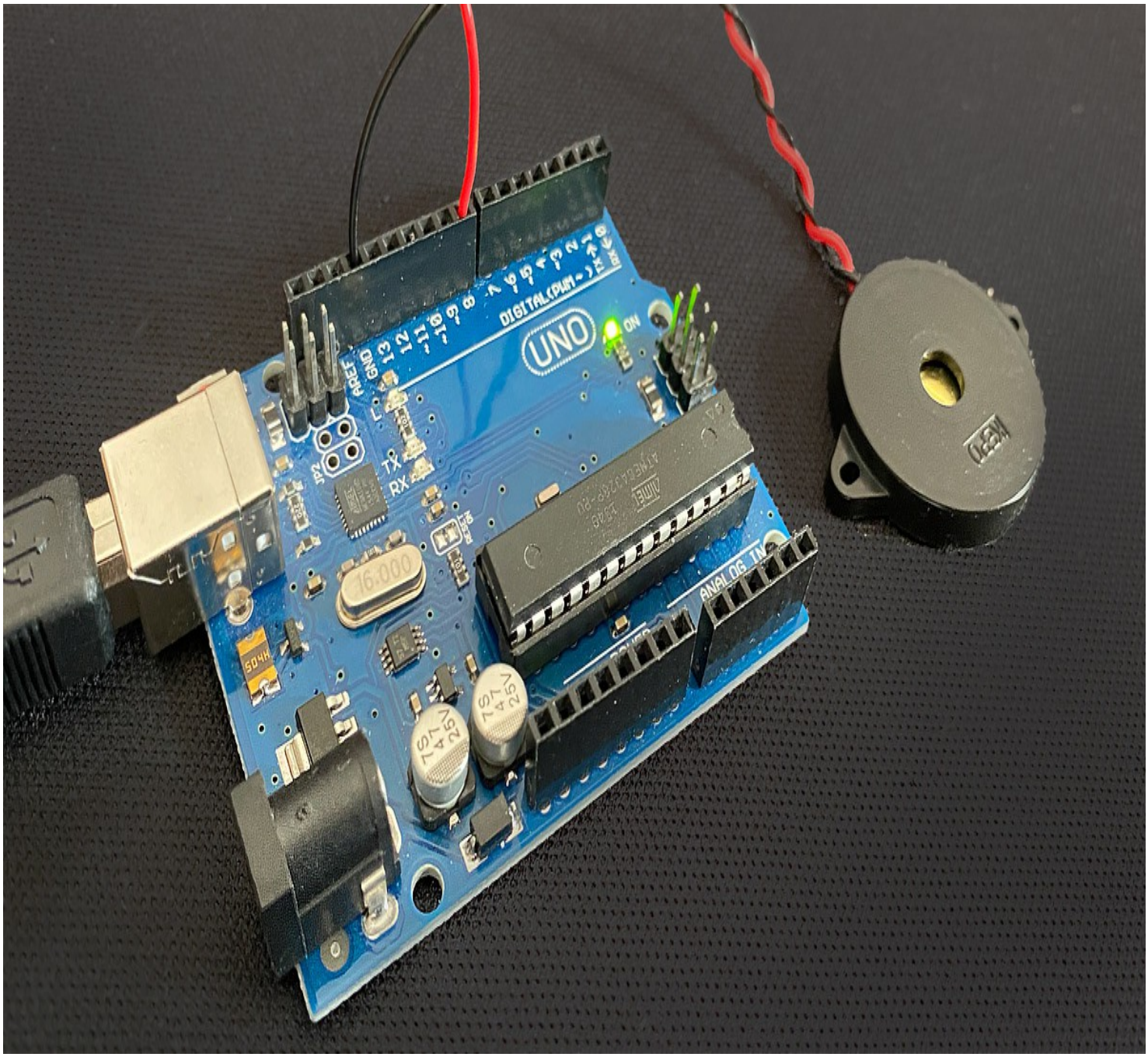
Morsen mit Arduino



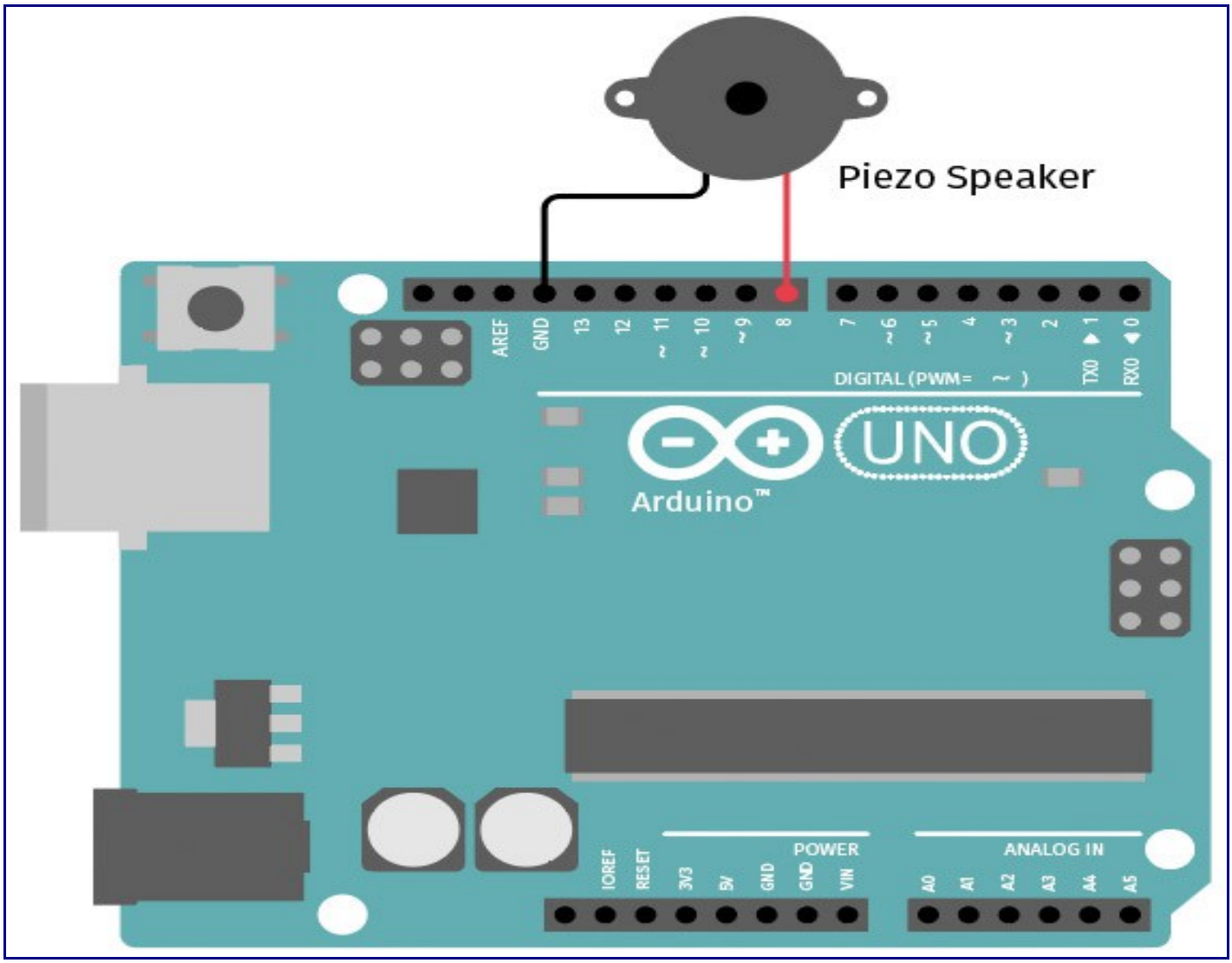
Wenn man in die Welt der Mikrocontroller-Programmierung eintaucht ist eines der ersten Übungen das Blinken eines SOS in Morse-Code. Aber wie geht es dann weiter? Morsen mit Arduino: In diesem Projekt erfährst du, wie du über den Seriellen Monitor Text eingeben und per Arduino in Blinken und Piepsen umwandeln kannst.

Bauteile

- Arduino Set, Breadboard, Kabel, etc.*
- Piezo-Speaker* (Tipp: Die kann man auch gut aus alten Melodie-Grußkarten rausoperieren.)

https://starthardware.org/morsen-mit-arduino/

Schaltung



Die Schaltung besteht aus einem Arduino UNO und einem Piezo-Speaker. Dieser wird einfach an den digitalen Pin 8 und den GND des Arduinos verbunden.

Code: Morsen mit Arduino

```
String inputString = "";          // speichert eingehende Daten
bool stringComplete = false;      // ist true, wenn der Empfangene String vollständig ist
int ledPin = 13;                  // Ausgabe Pin LED
int speakerPin = 8;               // Pin des Lautsprechers

static String morseLetters[] = {".-", "-...", "-.-.", "-..", ".", "..-.", "--.", "....", "..",
".---", "-.-", ".-..", "--", "-.", "---", ".--.", "--.-",
                                ".-.", "...", "-", "..-", "...-", ".--", "-..-", "-.--",
"--..", "E"
                               };
static String morseNumbers[] = {"-----", ".----", "..---", "...--", "....-", "....." ,
"-....", "--...", "---..", "----."};


void setup() {
```

https://starthardware.org/morsen-mit-arduino/

```
  Serial.begin(115200);           // serielle Kommunikation starten
  inputString.reserve(200);       // erstellt Speicherpuffer (Buffer) von 200 Bytes für
eingehende Daten
  pinMode(ledPin, OUTPUT);        // deklariert ledPin als Output
}

void loop() {
  if (stringComplete) {           // wenn er vollständig empfangen wurde
    Serial.println(inputString);  // gib ihn im Seriellen Monitor aus
    morseOutput(inputString);     // rufe morseOutput-Methode auf und übergib den String
    inputString = "";             // String zurücksetzen
    stringComplete = false;       // stringComplete auf false setzen, bis wieder ein String
vollständig empfangen wurde
  }
}

void serialEvent() {              // wird nach jedem Durchgang von loop() automatisch
ausgeführt
  while (Serial.available()) {    // solange serielle Daten verfügbar sind
    char inChar = (char)Serial.read();  // lies das aktuelle Byte aus
    inputString += inChar;        // füge das Byte als Char (Zeichen) dem String inputString
hinzu

    if (inChar == '\n') {         // falls das Zeichen ein Zeilenumbruch ist, ist der Empfang
der Daten abgeschlossen
      stringComplete = true;      // setze dann stringComplete auf true, damit die loop-
Methode die Daten ausgeben kann
    }
  }
}

void morseOutput(String myString) {
  // aktueller Buchstabe wird in Integer umgewandelt (Ascii-Code), das ergibt z.B. für das
kleine a 97
  String currentLetter = "";                                    // speichert den aktuellen
Morse-Code für einen Buchstaben zwischen
  for (int i = 0; i < myString.length(); i++) {                 // für 0 bis Länge des
Strings
    if ((int(myString[i]) >= 97) && (int(myString[i]) <= 122)) {  // falls Buchstabe von a bis
z
      currentLetter = morseLetters[int(myString[i]) - 97];      // currentLetter = Stelle im
Morse-Code Array ab 97
    } else if ((int(myString[i]) >= 65) && (int(myString[i]) <= 90)) { // falls Buchstabe von
A bis Z
      currentLetter = morseLetters[int(myString[i]) - 65];      // currentLetter = Stelle im
Morse-Code Array ab 65
    } else if ((int(myString[i]) >= 48) && (int(myString[i]) <= 57)) { // falls Zahl von 0 bis
9
      currentLetter = morseNumbers[int(myString[i]) - 48];      // currentLetter = Stelle im
Morse-Code Zahlen-Array ab 48
    } else {
      currentLetter = " ";                                      // bei allen anderen
Zeichen, füge ein Leerzeichen ein
    }

    Serial.print(currentLetter);                                // gib den Morse-Code im
Seriellen Monitor aus
    Serial.print(" ");                                          // Leerzeichen im Seriellen
Monitor

    for (int j = 0; j < currentLetter.length(); j++) {          // für 0 bis Länge von
```

```
current Letter
      if (int(currentLetter[j]) == 46) {                          // prüfe, ob aktuelles
Zeichen ein Punkt ist (Ascii-Code 46)
        digitalWrite(ledPin, HIGH);                               // schalte LED ein
        tone(speakerPin, 200);                                    // schalte Lautsprecher ein,
Tonhöhe 200 Hz
        delay(50);                                                // kurze Pause
        digitalWrite(ledPin, LOW);                                // schalte LED aus
        noTone(speakerPin);                                       // schalte Lautsprecher aus
        delay(50);                                                // kurze Pause
      } else if (int(currentLetter[j]) == 45) {                   // prüfe, ob aktuelles
Zeichen ein Strich ist (Ascii-Code 45)
        digitalWrite(ledPin, HIGH);                               // schalte LED ein
        tone(speakerPin, 200);                                    // schalte Lautsprecher ein,
Tonhöhe 200 Hz
        delay(150);                                               // längere Pause
        digitalWrite(ledPin, LOW);                                // schalte LED aus
        noTone(speakerPin);                                       // schalte Lautsprecher aus
        delay(150);                                               // längere Pause
      } else {
        delay(150);                                               // längere Pause, Zeichen
unbekannt
      }
    }
    delay(200);                                                   // Pause zwischen Zeichen
  }
}
```