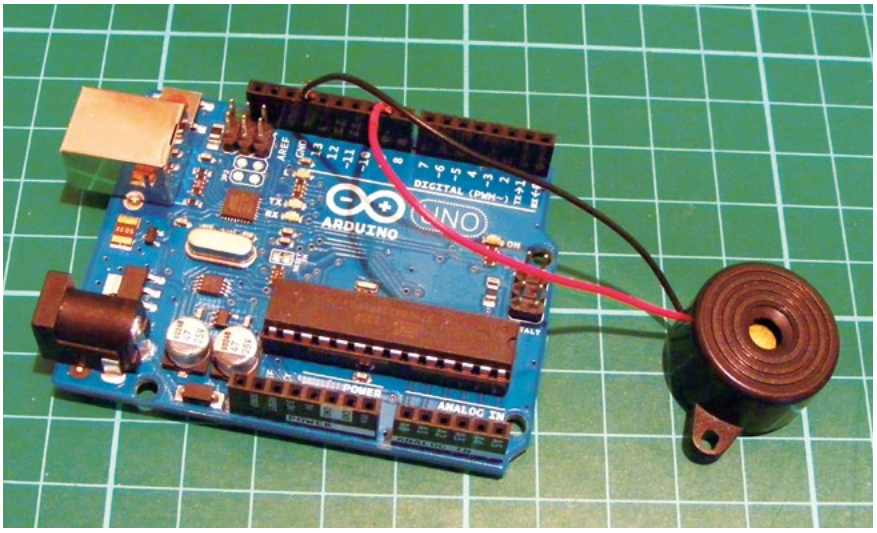
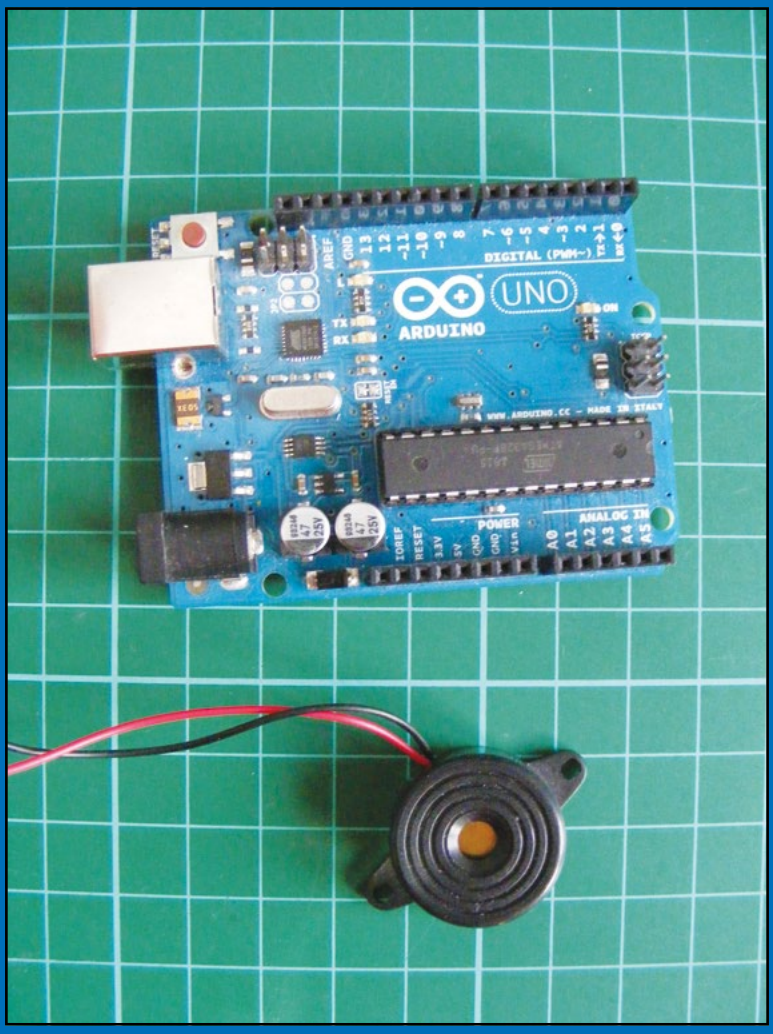
Projekt 7

Arduino-Musik

Bisher waren die Projekte meist optischer Natur. Jetzt wollen wir etwas Musik machen. In diesem Projekt verwenden wir einen Piezo-Summer, um Melodien zu spielen.





Benötigte Bauteile

- Arduino-Platine
- Piezo-Summer

So funktioniert es

Das Musik-Projekt verwendet einen Piezo-Summer, um Töne in unterschiedlichen Höhen auszugeben, die Noten entsprechen. Mit der Arduino-IDE können Sie die Reihenfolge, Geschwindigkeit und Dauer der Noten angeben, um eine bestimmte Melodie zu spielen.

Piezo-Summer sind günstige Summer, die oft in kleinen Spielzeugen zum Einsatz kommen. Ein Piezo-Element ohne Plastikgehäuse sieht aus wie eine goldene Metallscheibe mit zwei Leitungen (meist rot für plus und schwarz für minus). Ein Piezo-Element kann nur ein Klickgeräusch ausgeben, das wir über Anlegen einer Spannung erzeugen. Noten können wir dadurch abspielen, dass wir das Piezo-Element mehrere Hundert Male pro Sekunde bei einer bestimmten Frequenz klicken lassen. Dazu müssen wir wissen, welche Frequenz verschiedene Noten haben. Tabelle 7-1 zeigt die Noten und ihre Frequenzen. Intervall gibt die Zeitdauer in Mikrosekunden an. Diesen Wert halbieren wir, um `timeHigh` zu erhalten, das wir in unserem Code verwenden, um die Note zu erzeugen.

Tabelle 7-1: Noten und ihre Frequenzen

Note	Frequenz	Intervall	timeHigh
C	261 Hz	3830	1915
D	295 Hz	3400	1700
E	329 Hz	3038	1519
F	349 Hz	2864	1432
G	392 Hz	2550	1275
A	440 Hz	2272	1136
H	493 Hz	2028	1014
C	523 Hz	1912	956

Der Code sendet eine Rechteckschwingung in der entsprechenden Frequenz an das Piezo-Element und erzeugt dadurch den passenden Ton (mehr über Schwingungen erfahren Sie in Projekt 2). Die Töne werden anhand der folgenden Gleichung berechnet:

```
timeHigh = period / 2 = 1 / (2 * toneFrequency)
```

Der Aufbau dieses Projekts ist sehr einfach, denn es werden nur zwei Kabel an den Arduino angeschlossen.

Der Aufbau

1. Schließen Sie das schwarze Kabel des Piezo-Elements direkt an GND des Arduino an und das rote Kabel an Pin 9.

Piezo	Arduino
Rotes Kabel	Pin 9
Schwarzes Kabel	GND

2. Überprüfen Sie, ob Ihr Aufbau dem in Abbildung 7-1 entspricht, und laden Sie dann den Sketch auf den Arduino.

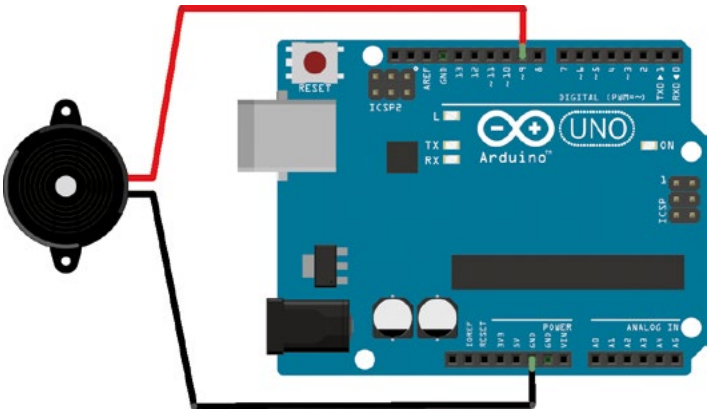


Abbildung 7-1:
Schaltplan für das Musik-Projekt

Der Sketch

Wir beginnen mit einer einfachen Melodie. Bei ❶ teilen wir der IDE mit, dass die Melodie aus 15 Noten besteht. Dann speichern wir die Noten der Melodie als Textstring in einem Array in der Reihenfolge, in der sie abgespielt werden sollen. Die Länge der Noten speichern wir in einem anderen Integer-Array. Wenn Sie die Melodie ändern möchten, ändern Sie die Noten im Array bei ❷ und deren Länge in Array bei ❸. Bei ❹ legen wir das Tempo fest, mit dem die Melodie gespielt wird. Welche Melodie spielen wir?

```
// Melodie (cleft) 2005 D. Cuartielles für K3

int speakerPin = 9;   // Pin für Piezo-Summer
int length = 15; ❶    // Anzahl der Noten
char notes[] = "ccggaagffeeddc "; ❷  // Leerzeichen = Pause
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 4 }; ❸
int tempo = 300; ❹

void playTone(int tone, int duration) {
  for (long i = 0; i < duration * 1000L; i += tone * 2) {
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(tone);
    digitalWrite(speakerPin, LOW);
    delayMicroseconds(tone);
  }
}

// Richtet timeHigh-Wert auf bestimmte Noten ein
void playNote(char note, int duration) {
  char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
  int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };
  for (int i = 0; i < 8; i++) { // Spielt die entsprechenden Töne
    if (names[i] == note) {
      playTone(tones[i], duration);
    }
  }
}

void setup() {
  pinMode(speakerPin, OUTPUT); // Richtet speakerPin als
                               // Ausgang ein
}

// Spielt die Melodie
void loop() {
  for (int i = 0; i < length; i++) {
    if (notes[i] == ' ') {
      delay(beats[i] * tempo); // Pause
    }
    else {
      playNote(notes[i], beats[i] * tempo);
    }
    delay(tempo / 2); // Pause zwischen Tönen
  }
}
```