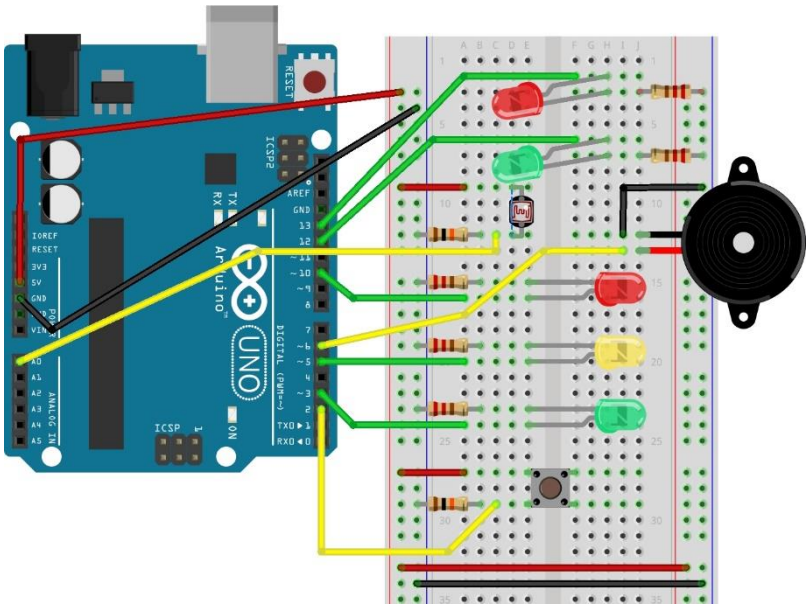
Arduino – Programmierpraxis mit Schall

Aufgabe 1) Dem angegebenen Lösungscode ist die „Ampelschaltung mit Fußgängerampel und akustischem Signalgeber“ gemäß nebenstehender Abbildung zugrunde gelegt – die Codesequenzen zur Ansteuerung des Piezo-Lautsprechers können aber auch verwendet werden, wenn die zwei Leuchtdioden für die „Fußgängerampel“ nicht in die Schaltung integriert sind.



Der Programmcode ist eine Variation des Lösungsvorschlags zu Aufgabe 2 aus der Datei ST_ LO_08Arduino_Verzweigungen:

```
int redLedPin = 10;
int yellowLedPin = 5;
int greenLedPin = 3;
int redPLedPin = 13;
int greenPLedPin = 12;
int buttonPin = 2;
int piezoPin = 6;
int switchstate = 0;
 void ledBlink(int pin, int times){
   int counter = 1;
   int piezoCounter;
    while(counter <= times){
      digitalWrite(pin,LOW);
      piezoCounter = 0;
      while(piezoCounter < 5){
        digitalWrite(piezoPin,HIGH);
        delay(125);
        digitalWrite(piezoPin,LOW);
        delay(75);
        piezoCounter = piezoCounter + 1;
      }
      //delay(1000); ersetzt durch die piezoPin-Schleife
      digitalWrite(pin,HIGH);
      piezoCounter = 0;
      while(piezoCounter < 5){
        digitalWrite(piezoPin,HIGH);
        delay(125);
        digitalWrite(piezoPin,LOW);
        delay(75);
        piezoCounter = piezoCounter + 1;
      }
      //delay(1000); ersetzt durch die piezoPin-Schleife
      counter = counter + 1;
    }
 }
```

```
void setup() {
  pinMode(redLedPin,OUTPUT);
  pinMode(yellowLedPin,OUTPUT);
  pinMode(greenLedPin,OUTPUT);
  pinMode(redPLedPin,OUTPUT);
  pinMode(greenPLedPin,OUTPUT);
  pinMode(buttonPin,INPUT); //!
  pinMode(piezoPin,OUTPUT);
   //set inital LED-states
   digitalWrite(redLedPin,LOW);
   digitalWrite(yellowLedPin,LOW);
   digitalWrite(greenLedPin,HIGH);
   digitalWrite(redPLedPin,HIGH);
   digitalWrite(greenPLedPin,LOW);
   //set initial piezo-state
   digitalWrite(piezoPin,LOW);
}
```

```
void loop() {
  int counter;
  int piezoCounter;

  switchstate = digitalRead(2);
  if(switchstate == HIGH){
    ledBlink(greenLedPin,3);
    digitalWrite(yellowLedPin,HIGH);
    digitalWrite(greenLedPin,LOW);
    piezoCounter = 0;
    while(piezoCounter < 10){
      digitalWrite(piezoPin,HIGH);
      delay(125);
      digitalWrite(piezoPin,LOW);
      delay(75);
      piezoCounter = piezoCounter + 1;
    }
    //delay(2000); ersetzt durch die piezoPin-Schleife
    digitalWrite(redLedPin,HIGH);
    digitalWrite(yellowLedPin,LOW);
    digitalWrite(greenPLedPin,HIGH);
    digitalWrite(redPLedPin,LOW);
    piezoCounter = 0;
    while(piezoCounter < 40){
      digitalWrite(piezoPin,HIGH);
      delay(75);
      digitalWrite(piezoPin,LOW);
      delay(50);
      piezoCounter = piezoCounter + 1;
    }
    // delay(5000); ersetzt durch die piezoPin-Schleife
    digitalWrite(redLedPin,LOW);
    digitalWrite(yellowLedPin,HIGH);
    ledBlink(greenPLedPin,2);
    digitalWrite(redPLedPin,HIGH);
    digitalWrite(greenPLedPin,LOW);
    digitalWrite(yellowLedPin,LOW);
    digitalWrite(greenLedPin,HIGH);
  }
  else{
    digitalWrite(piezoPin,HIGH);
    delay(200);
    digitalWrite(piezoPin,LOW);
    delay(100);
  }
}
```

Aufgabe 2) Bei Modifikation der Lösungen von Aufgabe 5) bzw. Aufgabe 6) gemäß der Datei **ST_LO_06Arduino_Modularisierung** genügen folgende (beispielhafte) Anpassungen:

…bei Verwendung des Befehls `digitalWrite`:

```
void shortSignal(int pin){
  digitalWrite(pin,HIGH);
  delay(200);
  digitalWrite(pin,LOW);
  delay(500);
}
```

```
void longSignal(int pin){
  int counter;

  counter = 0;
  while(counter < 5){
    digitalWrite(pin,HIGH);
    delay(200);
    digitalWrite(pin,LOW);
    delay(10);
    counter=counter+1;
  }
  delay(500);
}
```

…bei Verwendung der Befehle **tone**/**noTone**:

```
void shortSignal(int pin){
  tone(piezoPin,440);
  delay(200);
  noTone(piezoPin);
  delay(500);
}
```

```
void longSignal(int pin){
  tone(piezoPin,440);
  delay(800);
  noTone(piezoPin);
  delay(500);
}
```

Aufgabe 3)

```
int piezoPin;

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  // 1. Takt
  tone(piezoPin, 297);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 330);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 352);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 396);
  delay(200);
  noTone(piezoPin);
  delay(200);

  // 2. Takt
  tone(piezoPin, 440);
  delay(400);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 440);
  delay(400);
  noTone(piezoPin);
  delay(200);

  // 3. Takt
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);

  // 4. Takt
  tone(piezoPin, 440);
  delay(400);
  noTone(piezoPin);
  delay(200);
  // Viertelpause
  delay(400);
  delay(200);

  // Wiederholung 3. Takt
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 495);
  delay(200);
  noTone(piezoPin);
  delay(200);

  // Wiederholung 4. Takt
  tone(piezoPin, 440);
  delay(400);
  noTone(piezoPin);
  delay(200);
  // Viertelpause
  delay(400);
  delay(200);

  // 5. Takt
  tone(piezoPin, 396);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 396);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 396);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 396);
  delay(200);
  noTone(piezoPin);
  delay(200);

  // 6. Takt
  tone(piezoPin, 352);
  delay(400);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 352);
  delay(400);
  noTone(piezoPin);
  delay(200);

  // 7. Takt
  tone(piezoPin, 440);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 440);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 440);
  delay(200);
  noTone(piezoPin);
  delay(200);
  tone(piezoPin, 440);
  delay(200);
  noTone(piezoPin);
  delay(200);

  // 8. Takt
  tone(piezoPin, 297);
  delay(800);
  noTone(piezoPin);
  delay(200);
}
```

Aufgabe 4)

a)

```
int piezoPin;
int frequenzen[] = {297,330,352,396,440,440,495,495,495,495,440,0,495,495,495,
                    495,440,0,396,396,396,396,352,352,440,440,440,440,297};
int durations[] = {200,200,200,200,400,400,200,200,200,200,400,400,200,200,200,
                   200,400,400,200,200,200,200,400,400,200,200,200,200,800};
int arrayLen = sizeof(frequenzen)/sizeof(int);

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = 0;
  while(index < arrayLen){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index + 1;
  }
}
```

b) …Variablendeklaration wie bei Lösung zu Aufgabenteil a)

```
...

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = arrayLen-1;
  while(index >= 0){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index - 1;
  }
}
```

c) Die Aufgabenstellung „… so, dass nur jede zweite Note gespielt wird.“, kann mehrdeutig verstanden werden: Entweder jede Note, auf einem ungeraden Index, oder jede Note auf einem geraden Index der Feldvariablen. Zudem können die Noten auch wahlweise „von vorne nach hinten“ oder „von hinten nach vorne“ gespielt werden. Beispielcodierungen für alle vier dieser Möglichkeiten finden sich umseitig (Variablendeklaration jeweils wie bei Lösung zu Aufgabenteil a)):

```
// alle Noten auf ungeradem Index von vorne nach hinten

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = 1;
  while(index < arrayLen){
    if(frequenzen[index] > 0){ //zur Beri
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index + 2;
  }
}
```

```
// alle Noten auf geradem Index von vorne nach hinten

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = 0;
  while(index < arrayLen){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index + 2;
  }
}
```

```
// alle Noten auf ungeradem Index von hinten nach vorne (höchster Index ist 28)

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = 27;
  while(index >=0){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index - 2;
  }
}
```

```
// alle Noten auf geradem Index von hinten nach vorne (höchster Index ist 28)

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = 28;
  while(index >=0){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index - 2;
  }
}
```

Hinweis: Wenn die Indizes „von hinten nach vorne“ durchlaufen werden, lässt sich der höchste ungerade bzw. der höchste gerade Index auch mit Hilfe der Restberechnung bei Division durch 2 (der sogenannten Modulo-Operation) bestimmen. Entsprechende Codefragmente sehen dann folgendermaßen aus (Variablendeklaration jeweils wie bei Lösung zu Aufgabenteil a)):

```
// alle Noten auf ungeradem Index von hinten nach vorne

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  if((arrayLen%2)==0){
    index = arrayLen - 1;
  }
  else{
    index = arrayLen - 2;
  }
  while(index >=0){
   ...
```

```
// alle Noten auf geradem Index von hinten nach vorne

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  if((arrayLen%2)==0){
    index = arrayLen - 2;
  }
  else{
    index = arrayLen - 1;
  }
  while(index >=0){
   ...
```

…zur Erklärung:

`arrayLen % 2` liefert den Rest bei Division des Wertes der Variablen namens **`arrayLen`** (= Anzahl der gespeicherten Werte in der Feldvariablen namens **`frequenzen`**) durch 2. Ist dieser Rest gleich 0, so ist in der Feldvariablen eine gerade Anzahl von Werten gespeichert, ist dieser Rest gleich 1, eine ungerade Anzahl von Werten.

Da die Feldindizierung mit 0 beginnt, ist im ersten Fall der größte ungerade Feldindex **`arrayLen - 1`** (und der größte gerade Feldindex **`arrayLen - 2`**), im zweiten Fall der größte ungerade Feldindex **`arrayLen - 2`** (und der größte gerade Feldindex **`arrayLen - 1`**).

Diese „automatische“ Berechnung des jeweils größten Feldindizes kann besonders schnellen Programmier(anfäng)ern als Zusatzübung erklärt werden.

d) Auch bei der Aufgabenstellung d) gibt es zwei Interpretationsmöglichkeiten: Die Noten können in der angegebenen Reihenfolge entweder „von vorne nach hinten“ oder „von hinten nach vorne“ gespielt werden. Wieder werden mögliche Codierungen für beide Varianten angegeben (Variablendeklaration jeweils wie bei Lösung zu Aufgabenteil a)):

```
// alle Noten auf ungeradem Index und dann alle Noten auf geradem Index von vorne nach hinten

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;
```

```
  index = 1;
  while(index < arrayLen){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index + 2;
  }
  index = 0;
  while(index < arrayLen){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index + 2;
  }
}

// alle Noten auf ungeradem Index und dann alle Noten auf geradem Index von hinten nach vorne

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}

void loop() {
  int index;

  index = 27;
  while(index >= 0){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index - 2;
  }
  index = 28;
  while(index >= 0){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index - 2;
  }
}
```

e) …Variablendeklaration wie bei Lösung zu Aufgabenteil a):

```
// alle Noten auf ungeradem Index von vorne nach hinten
// und dann alle Noten auf geradem Index von hinten nach vorne

void setup() {
  piezoPin = 6;
  pinMode(piezoPin, OUTPUT);
}
```

```
void loop() {
  int index;

  index = 1;
  while(index < arrayLen){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index + 2;
  }
  index = 28;
  while(index >= 0){
    if(frequenzen[index] > 0){ //zur Berücksichtigung der Pause!
      tone(piezoPin, frequenzen[index]);
    }
    delay(durations[index]);
    noTone(piezoPin);
    delay(200);
    index = index - 2;
  }
}
```