Arduino STEM-Shield: Demonstration

1 Vorbemerkungen

(Inhalt)

1.1 Beschreibung STEM-Shield

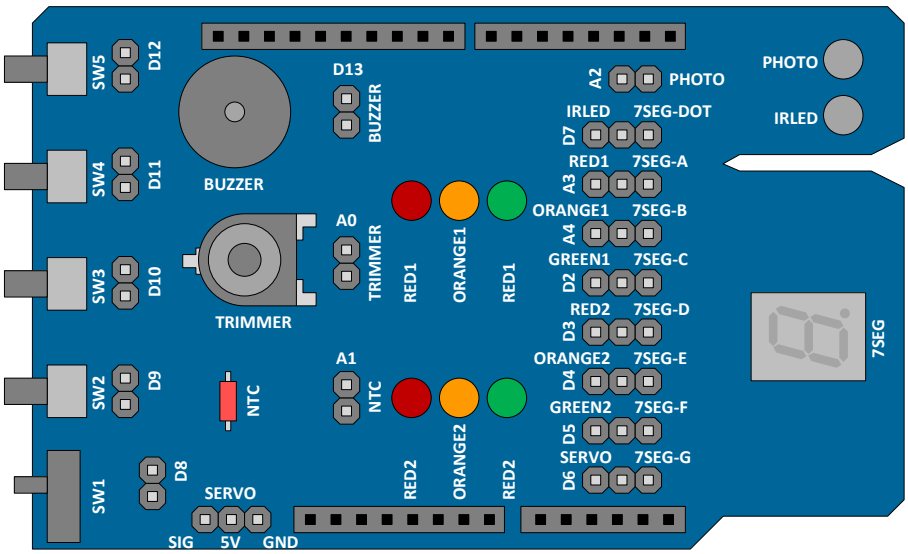
(Thema)

Die Firma Velleman (https://www.velleman.eu/products) bietet ein so genanntes STEM-Shield für den Arduino UNO an. STEM steht hier für die Anwendung von Science, Technology, Engineering und Mathematics im Bildungsbereich. Leider ist das Shield bisher nur als Bausatz verfügbar.

Das STEM-Shield ermöglicht die Arbeit mit folgenden Komponenten:

- 1 Schalter (SW1),
- 4 Taster (SW2…SW5),
- 1 Minilautsprecher (BUZZER),
- 1 Potentiometer (TRIMMER),
- 1 Heißleiter (NTC),
- 1 Anschluss für Servomotor (SERVO),
- 6 farbige LED (RED1, ORANGE1, GREEN1, RED2, ORANGE2, GREEN2),
- 1 Fototransistor (PHOTO),
- 1 Infrarot-LED (IRLED),
- 1 7-Segmentanzeige (7SEG).

Die folgende Abbildung zeigt die Anordnung der Komponenten



Die Auswahl der genannten Komponenten erfolgt durch Kontaktbrücken auf den zugeordneten Stiftleisten. Die Stiftleisten haben jeweils 2 oder 3 Kontakte.

In der folgenden Tabelle sind die benutzten Anschlüsse des Arduino und die entsprechenden Komponentenanschlüsse zusammengestellt.

Anschluss Arduino	Komponente	Alternative Komponente
D2	GREEN1	7SEG-C
D3 (PWM)	RED2	7SEG-D
D4	ORANGE2	7SEG-E
D5 (PWM)	GREEN2	7SEG-F
D6 (PWM)	SERVO	7SEG-G
D7	IRLED	7SEG-DOT
D8	SW1	
D9 (PWM)	SW2	
D10 (PWM)	SW3	
D11 (PWM)	SW4	
D12	SW5	
D13	BUZZER	
A0	TRIMMER	
A1	NTC	
A2	PHOTO	
A3	RED1	7SEG-A
A4	ORANGE1	7SEG-B
A5		

1.2 Übersicht Demonstration

(Thema)

Durch die Realisierung einfacher Programme wird die Zusammenarbeit des Arduino UNO mit den vorgestellten Komponenten demonstriert.

Übersicht:

- LED_BUZZER
  Blinken von LED und Einsatz des Buzzers
- SEGMENT_A
  Anzeige der einzelnen Segmente der 7-Segmantanzeige
- SW
  Abfrage des Umschalters und eines Tasters
- TRIMMER
  Abfrage der eingestellten Spannung am Potentiometer
- NTC
  Abfrage eines Wertes, der durch die Temperatur verändert wird
- PHOTO
  Abfrage eines Wertes, der durch die Lichtstärke verändert wird
- PHOTO_IRLED
  Kombination von Foto-Transistor und Infrarot-LED
- SERVO
  Ansteuern eines Servo-Motors
- SEGMENT_B
  Anzeige von Ziffern mit der 7-Segmentanzeige

Nach dem erfolgreichen Realisieren der Demonstrationsprogramme ist man in der Lage die Mehrzahl der später gestellten Aufgaben zu lösen.

2 Demonstration: LED_BUZZER

(Inhalt)

2.1 Zielstellung

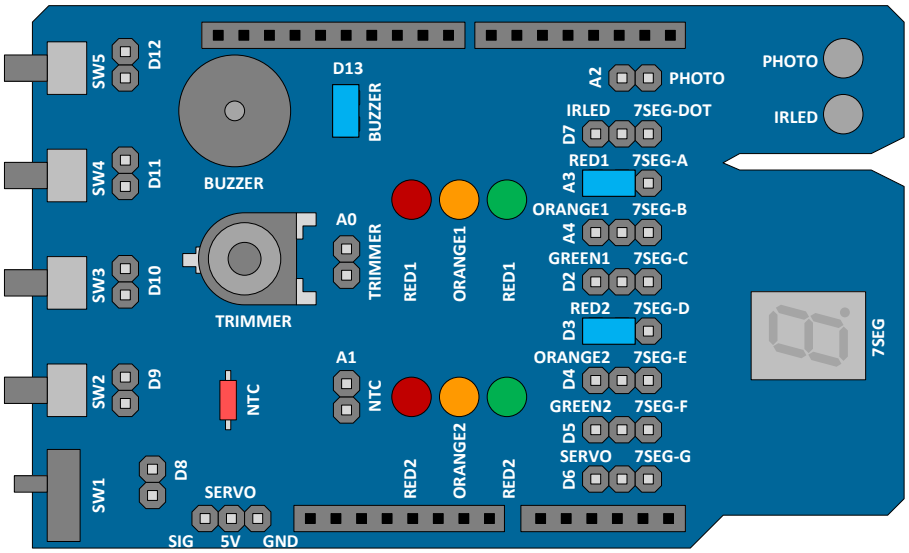
(Thema)

Die roten LEDs blinken fortlaufend (eine halbe Sekunde ein, eine halbe Sekunde aus). 500 Millisekunden entsprechen einer halben Sekunde.
Während der Leuchtphase ist ein Ton mit einer Frequenz von 1000 Hz zu hören.

2.2 Vorbereitung

(Thema)

Verwendete Komponenten: RED1, RED2, BUZZER.
Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



2.3 Lösung

(Thema)

LED_BUZZER

```
1  const int PAUSE = 500;
2  const int RED1 = A3;
3  const int RED2 = 3;
4  const int BUZZER = 13;
5
6  void setup() {
7    pinMode(RED1, OUTPUT);
8    pinMode(RED2, OUTPUT);
9  }
10
11 void loop() {
12   digitalWrite(RED1, HIGH);
13   digitalWrite(RED2, HIGH);
14   tone(BUZZER, 1000);
15   delay(PAUSE);
16   digitalWrite(RED1, LOW);
17   digitalWrite(RED2, LOW);
18   noTone(BUZZER);
19   delay(PAUSE);
20 }
```

Die Arbeit wird durch geeignete Konstanten erleichtert.

So repräsentieren zum Beispiel die Namen der eingesetzten Komponenten die Pins des Arduino.

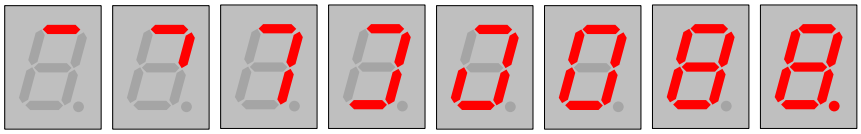
3 Demonstration: SEGMENT_A

(Inhalt)

3.1 Zielstellung

(Thema)

Im Abstand von 500 Millisekunden leuchten nacheinander alle Segmente der 7-Segmentanzeige und der Punkt.

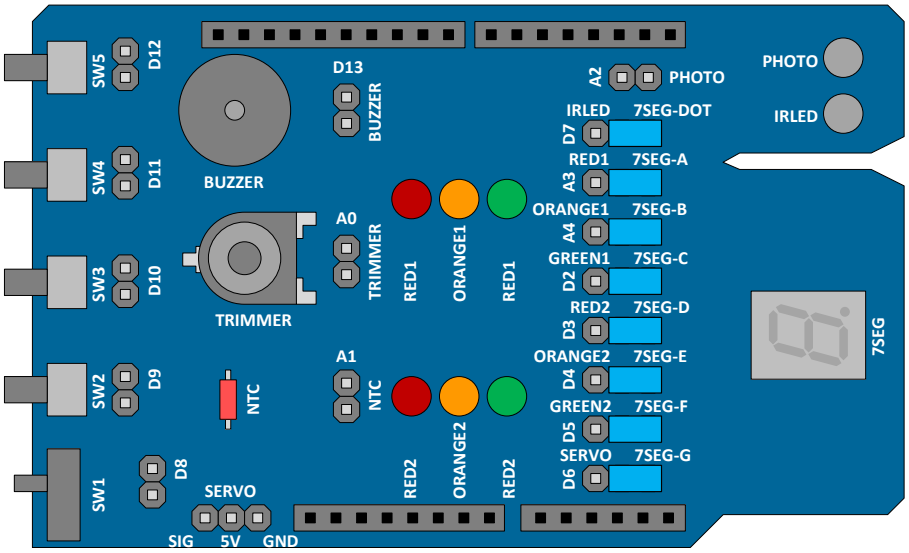


Nach einer Pause von 500 Millisekunden gehen alle Segmente und der Punkt aus.
Der Vorgang beginnt von vorn.

3.2 Vorbereitung

(Thema)

Verwendete Komponenten:
7SEG-A, 7SEG-B, 7SEG-C, 7SEG-D, 7SEG-E, 7SEG-F, 7SEG-G, 7SEG-DOT.
Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



3.3 Lösung

(Thema)

Es kommen wieder sinnvolle Konstanten für die Pins zum Einsatz. Außerdem werden die Konstanten in einem Feld (segment[]) strukturiert. Das vereinfacht in diesem Fall die Umsetzung der Demonstration.
Alle LEDs der eingesetzten 7-Segmentanzeige besitzen eine gemeinsame Anode. Das bedeutet, dass die entsprechenden Pins auf LOW gesetzt werden müssen, um die einzelnen Segmente und den Punkt zum Leuchten zu bringen.

SEGMENT_A

```
const int PAUSE = 500;
const int SEG_A = A3;
const int SEG_B = A4;
const int SEG_C = 2;
const int SEG_D = 3;
const int SEG_E = 4;
const int SEG_F = 5;
const int SEG_G = 6;
const int SEG_DOT = 7;
int segment[] = {SEG_A, SEG_B, SEG_C, SEG_D, SEG_E, SEG_F, SEG_G, SEG_DOT};
int i;

void setup() {
  pinMode(SEG_A, OUTPUT);
  pinMode(SEG_B, OUTPUT);
  pinMode(SEG_C, OUTPUT);
  pinMode(SEG_D, OUTPUT);
  pinMode(SEG_E, OUTPUT);
  pinMode(SEG_F, OUTPUT);
  pinMode(SEG_G, OUTPUT);
  pinMode(SEG_DOT, OUTPUT);
}

void loop() {
  for (i = 0; i < 8; i++) {
    digitalWrite(segment[i], LOW);
    delay(PAUSE);
  }

  for (i = 0; i < 8; i++) {
    digitalWrite(segment[i], HIGH);
  }
  delay(PAUSE);
}
```

4 Demonstration: SW

(Inhalt)

4.1 Zielstellung

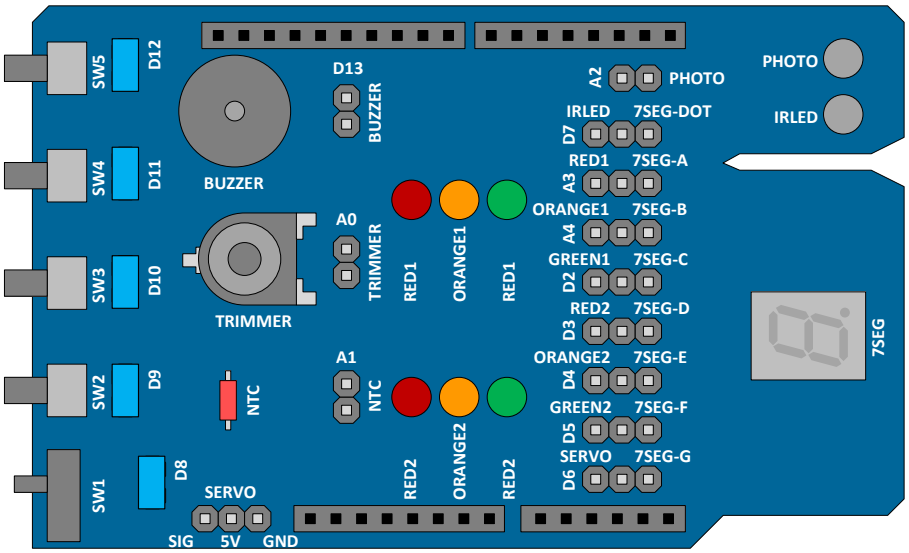
(Thema)

Im Abstand von einer Sekunde werden die Zustände des Schalters und der 4 Taster fortlaufend abgefragt und im seriellen Monitor angezeigt.

4.2 Vorbereitung

(Thema)

Verwendete Komponenten: SW1, SW2, SW3, SW4, SW5.
Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



4.3 Lösung

(Thema)

Auch hier werden die Konstanten der Pins in einem Feld (sw[]) strukturiert. Zusätzlich kommt noch ein Feld für die Zustände von Schalter und Tastern (statusSW[]) zum Einsatz. Alle 5 Einträge werden mit LOW initialisiert.

SW

```
const int PAUSE = 1000;
const int SW1 = 8;
const int SW2 = 9;
const int SW3 = 10;
const int SW4 = 11;
const int SW5 = 12;
int sw[] = {SW1, SW2, SW3, SW4, SW5};
bool statusSW[] = {false, false, false, false, false};
int i;

void setup() {
  Serial.begin(9600);
  pinMode(SW1, INPUT);
  pinMode(SW2, INPUT);
  pinMode(SW3, INPUT);
  pinMode(SW4, INPUT);
  pinMode(SW5, INPUT);
}

void loop() {
  for (i = 0; i < 5; i++) {
    statusSW[i] = digitalRead(sw[i]);
    Serial.print("SW");
    Serial.print(i+1);
    Serial.print("=");
    Serial.println(statusSW[i]);
    delay(PAUSE);
  }
  Serial.println();
}
```

5 Demonstration: TRIMMER

(Inhalt)

5.1 Zielstellung

(Thema)

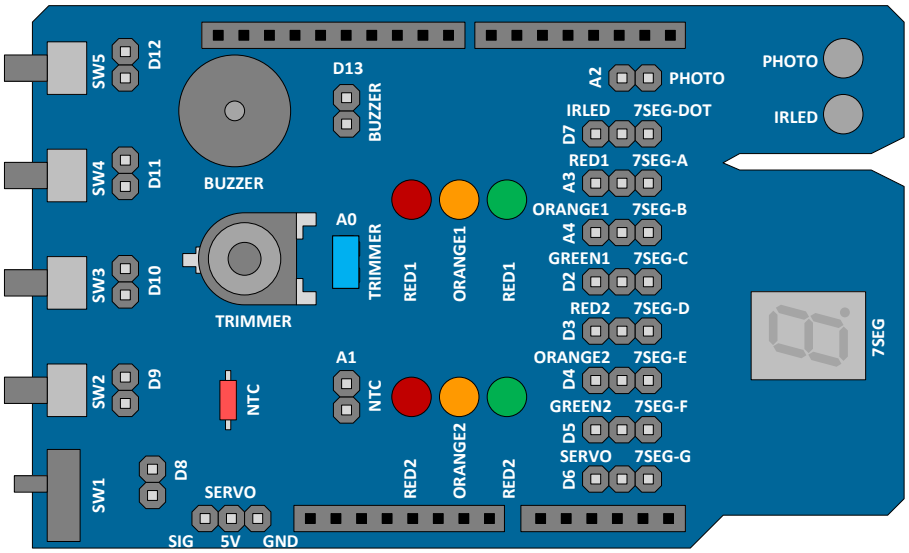
Der analoge Eingang A0 liest fortlaufend im Abstand von einer Sekunde den Wert VALUE (0..1023) aus, der durch das Potentiometer eingestellt wird. Zusätzlich erfolgt eine Umrechnung des Wertes in eine Spannung (0..5 V).

5.2 Vorbereitung

(Thema)

Verwendete Komponenten: TRIMMER.

Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



5.3 Lösung

(Thema)

TRIMMER

```
const int PAUSE = 1000;
const int TRIMMER = A0;
int value;
double voltage = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  value = analogRead(TRIMMER);
  voltage = value / 1023.0 * 5.0;
  Serial.println(value);
  Serial.print("U=");
  Serial.print(voltage);
  Serial.println("V");
  delay(PAUSE);
}
```

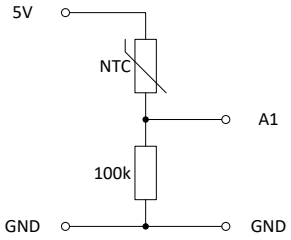
6 Demonstration: NTC

(Inhalt)

6.1 Zielstellung

(Thema)

Der analoge Eingang A1 liest fortlaufend im Abstand von einer Sekunde den Wert VALUE (0..1023) aus, der am Spannungsteiler mit einem konstanten Widerstand und dem Heißleiter (NTC) eingestellt wird.

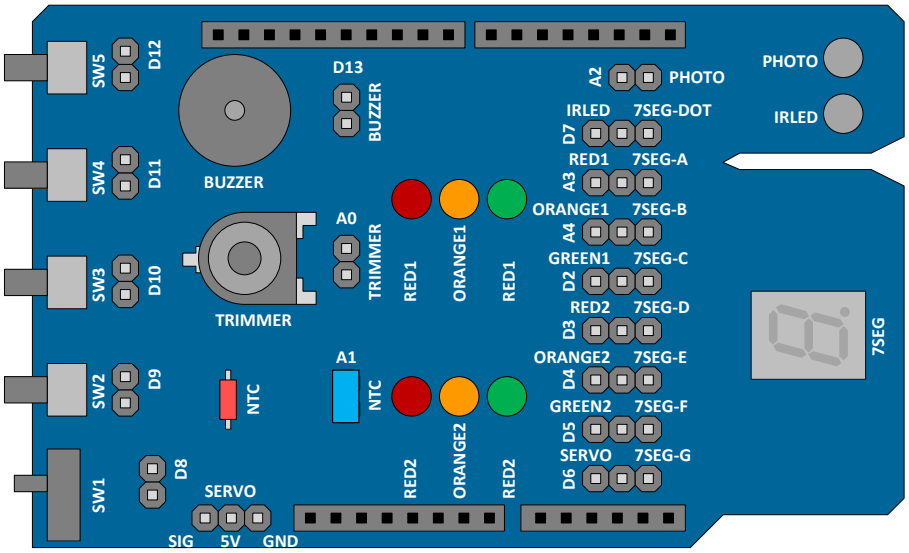


Da sich der Widerstand des Heißleiters in Abhängigkeit von der Temperatur ändert, kann VALUE als ein Maß für die Temperatur verstanden werden. Bei Berührung des Heißleiters mit dem Finger erhöht sich VALUE.

6.2 Vorbereitung

(Thema)

Verwendete Komponenten: NTC.
Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



6.3 Lösung

(Thema)

NTC

```
1  const int PAUSE = 1000;
2  const int NTC = A1;
3
4  void setup() {
5    Serial.begin(9600);
6  }
7
8  void loop() {
9    Serial.println(analogRead(NTC));
10   delay(PAUSE);
11 }
```

7 Demonstration: PHOTO

(Inhalt)

7.1 Zielstellung

(Thema)

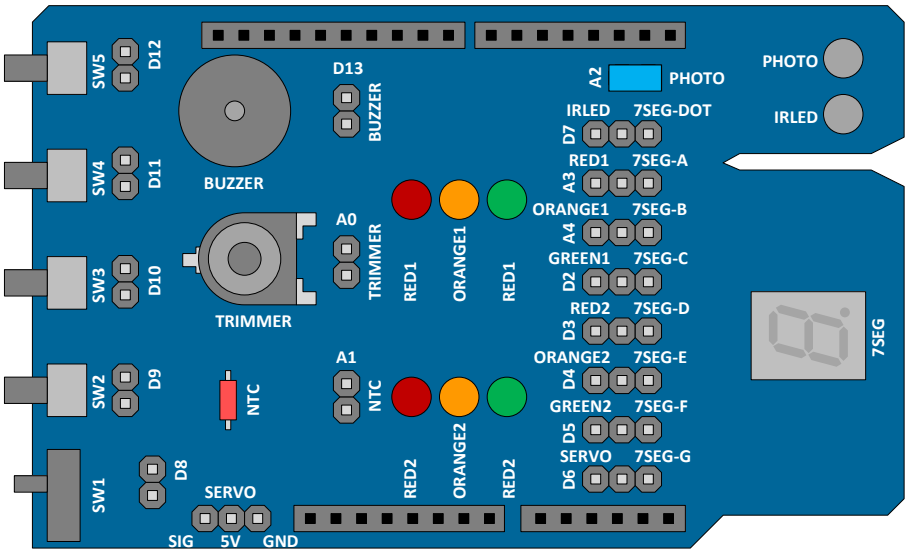
Der analoge Eingang A2 liest fortlaufend im Abstand von einer Sekunde den Wert VALUE (0...1023) aus, der durch Fototransistor geliefert wird. Bei geringerer Lichtstärke (z.B. Abdunkeln) verringert sich die Größe von VALUE.

7.2 Vorbereitung

(Thema)

Verwendete Komponenten: PHOTO.

Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



7.3 Lösung

(Thema)

PHOTO

```
1  const int PAUSE = 1000;
2  const int PHOTO = A2;
3
4  void setup() {
5    Serial.begin(9600);
6  }
7
8  void loop() {
9    Serial.println(analogRead(PHOTO));
10   delay(PAUSE);
11 }
```

8 Demonstration: PHOTO_IRLED

(Inhalt)

8.1 Zielstellung

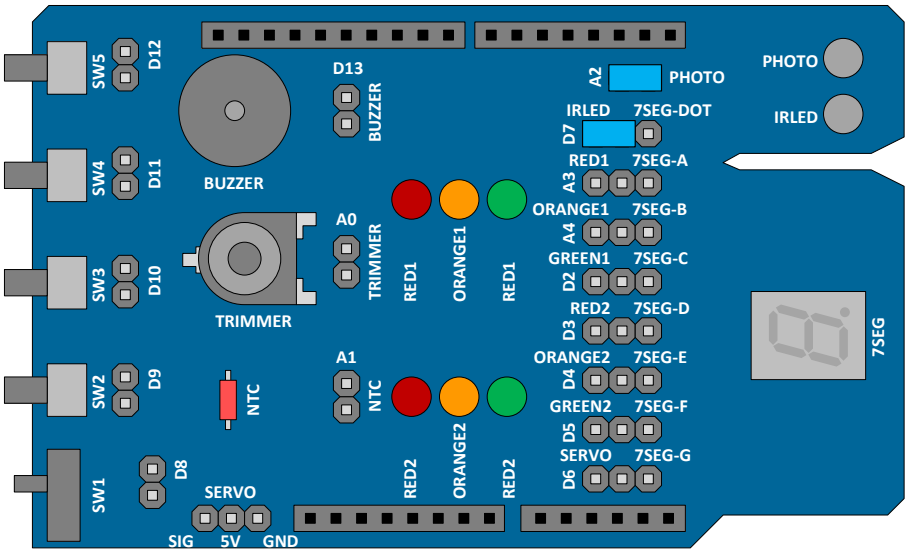
(Thema)

Die Infrarot-LED erhält HIGH-Pegel und leuchtet dauerhaft im nicht sichtbaren Bereich. Der Fototransistor liefert im Abstand von 2 Sekunden einen Wert für die Lichtstärke. Hält man ein Blatt vor die Komponenten PHOTO und IRLED steigt der Wert für die Lichtstärke deutlich, das das Blatt das infrarote Licht reflektiert.

8.2 Vorbereitung

(Thema)

Verwendete Komponenten: PHOTO, IRLED.
Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



8.3 Lösung

(Thema)

PHOTO_IRLED

```
const int PAUSE = 2000;
const int IRLED = 7;
const int PHOTO = A2;

void setup() {
  Serial.begin(9600);
  pinMode(IRLED, OUTPUT);
  digitalWrite(IRLED, HIGH);
}

void loop() {
  Serial.println(analogRead(PHOTO));
  delay(PAUSE);
}
```

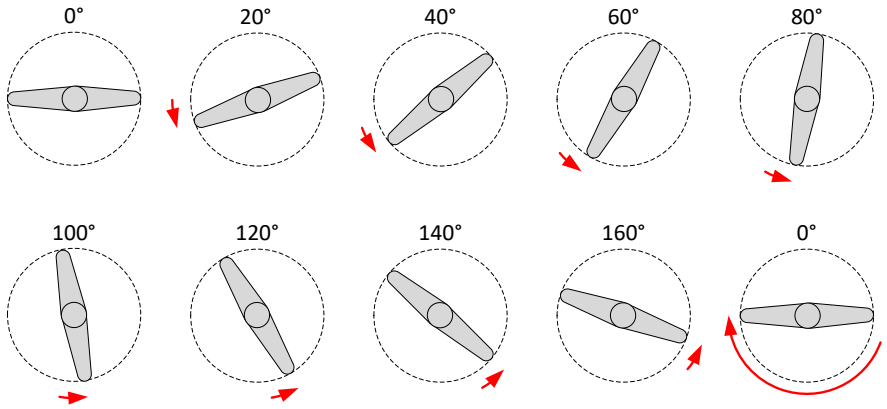
9 Demonstration: SERVO

(Inhalt)

9.1 Zielstellung

(Thema)

Der angeschlossene Servo-Motor bewegt sich endlos nach jeweils 500 Millisekunden zu folgenden Winkeln.

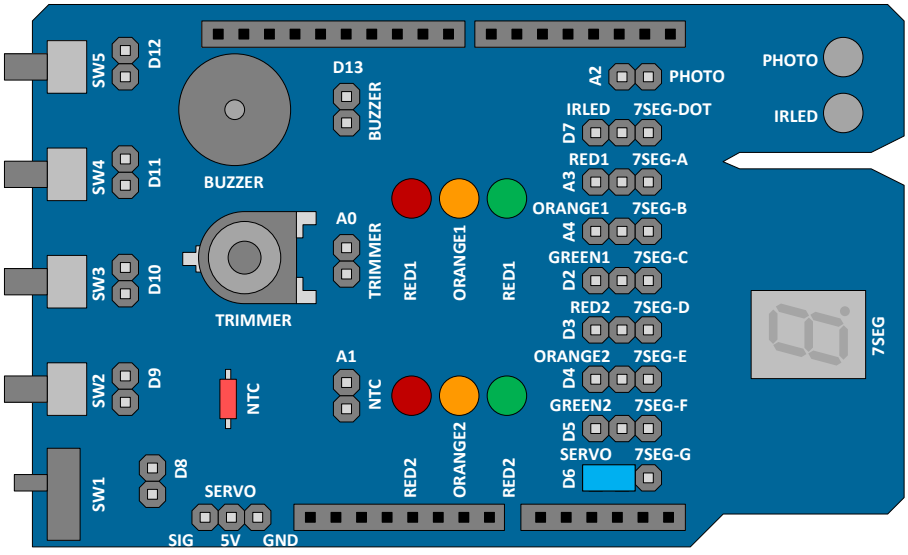


9.2 Vorbereitung

(Thema)

Verwendete Komponenten: SERVO (Anschluss).

Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



9.3 Lösung

(Thema)

Die Arduino IDE bietet im Menü: „Sketch/Bibliothek einbinden“ die Bibliothek Servo (Servo.h) an, die hier für die Realisierung genutzt wird.

SERVO

```
#include <Servo.h>

Servo servo;
const int PAUSE = 500;
const int SERVO = 6;
int i;

void setup() {
  servo.attach(SERVO);
}

void loop() {
  for (i = 0; i < 9; i++) {
    servo.write(i * 20);
    delay(PAUSE);
  }
}
```

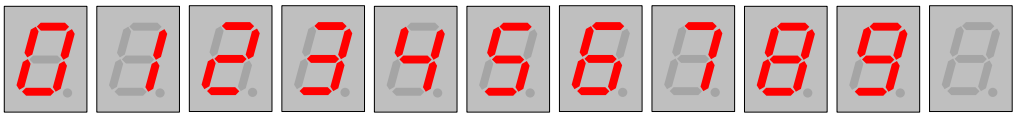
10 Demonstration: SEGMENT_B

(Inhalt)

10.1 Zielstellung

(Thema)

Es werden endlos die die folgenden Segmente der 7-Segmentanzeige im Abstand von einer Sekunde angezeigt.

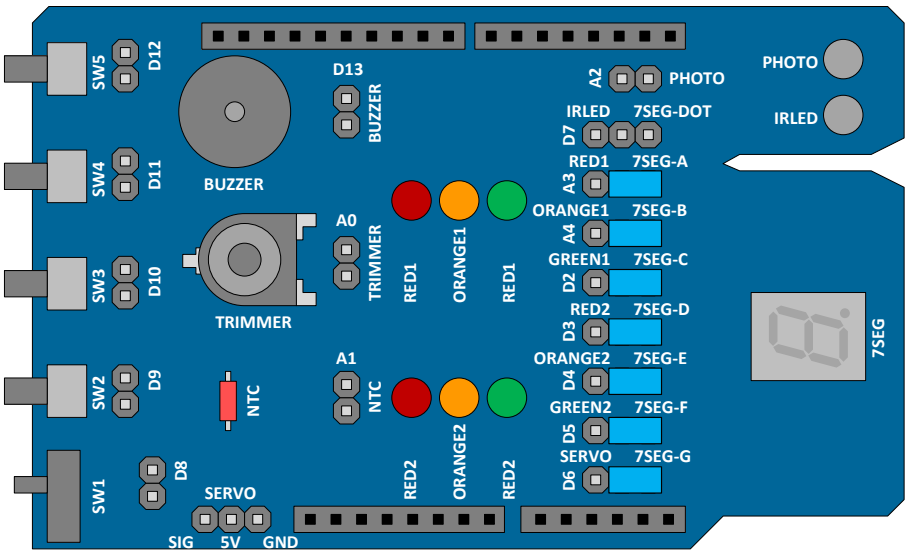


Dabei handelt es sich um die Darstellung der Ziffern 0..9 und eine nicht leuchtende Anzeige.

10.2 Vorbereitung

(Thema)

Verwendete Komponenten:
7SEG-A, 7SEG-B, 7SEG-C, 7SEG-D, 7SEG-E, 7SEG-F, 7SEG-G.
Die Abbildung zeigt die Verwendung der benötigten Kontaktbrücken. Die Position der anderen Steckbrücken hat keinen Einfluss auf das Programm.



10.3 Lösung

(Thema)

Zum Anzeigen der Ziffern kommt die Funktion `void show(int digit)` zum Einsatz. Mit dem Aufruf `show(0)` erfolgt zum Beispiel die Anzeige der Ziffer 0. Das gilt analog für 1...9. Alle anderen Zahlen (zum Beispiel. 10) liefern eine Anzeige, die nicht leuchtet.

SEGMENT_B

```
const int PAUSE = 1000;
const int SEG_A = A3;
const int SEG_B = A4;
const int SEG_C = 2;
const int SEG_D = 3;
const int SEG_E = 4;
const int SEG_F = 5;
const int SEG_G = 6;
int i;

void setup() {
  pinMode(SEG_A, OUTPUT);
  pinMode(SEG_B, OUTPUT);
  pinMode(SEG_C, OUTPUT);
  pinMode(SEG_D, OUTPUT);
  pinMode(SEG_E, OUTPUT);
  pinMode(SEG_F, OUTPUT);
  pinMode(SEG_G, OUTPUT);
}

void loop() {
  for (i = 0; i < 11; i++) {
    show(i);
    delay(PAUSE);
  }
}

void show(int digit) {
  String data;
  int segment[] = {SEG_A, SEG_B, SEG_C, SEG_D, SEG_E, SEG_F, SEG_G};
  int index;

  for (index = 0; index < 7; index++) {
    digitalWrite(segment[index], HIGH);
  }
```

```
  switch (digit) {
    case 0:
      data = "1111110";
      break;
    case 1:
      data = "0110000";
      break;
    case 2:
      data = "1101101";
      break;
    case 3:
      data = "1111001";
      break;
    case 4:
      data = "0110011";
      break;
    case 5:
      data = "1011011";
      break;
    case 6:
      data = "1011111";
      break;
    case 7:
      data = "1110000";
      break;
    case 8:
      data = "1111111";
      break;
    case 9:
      data = "1111011";
      break;
    default:
      data = "0000000";
      break;
  }

  for (index = 0; index < 7; index++) {
    if (data[index] == '1') {
      digitalWrite(segment[index], LOW);
    }
  }
}
```