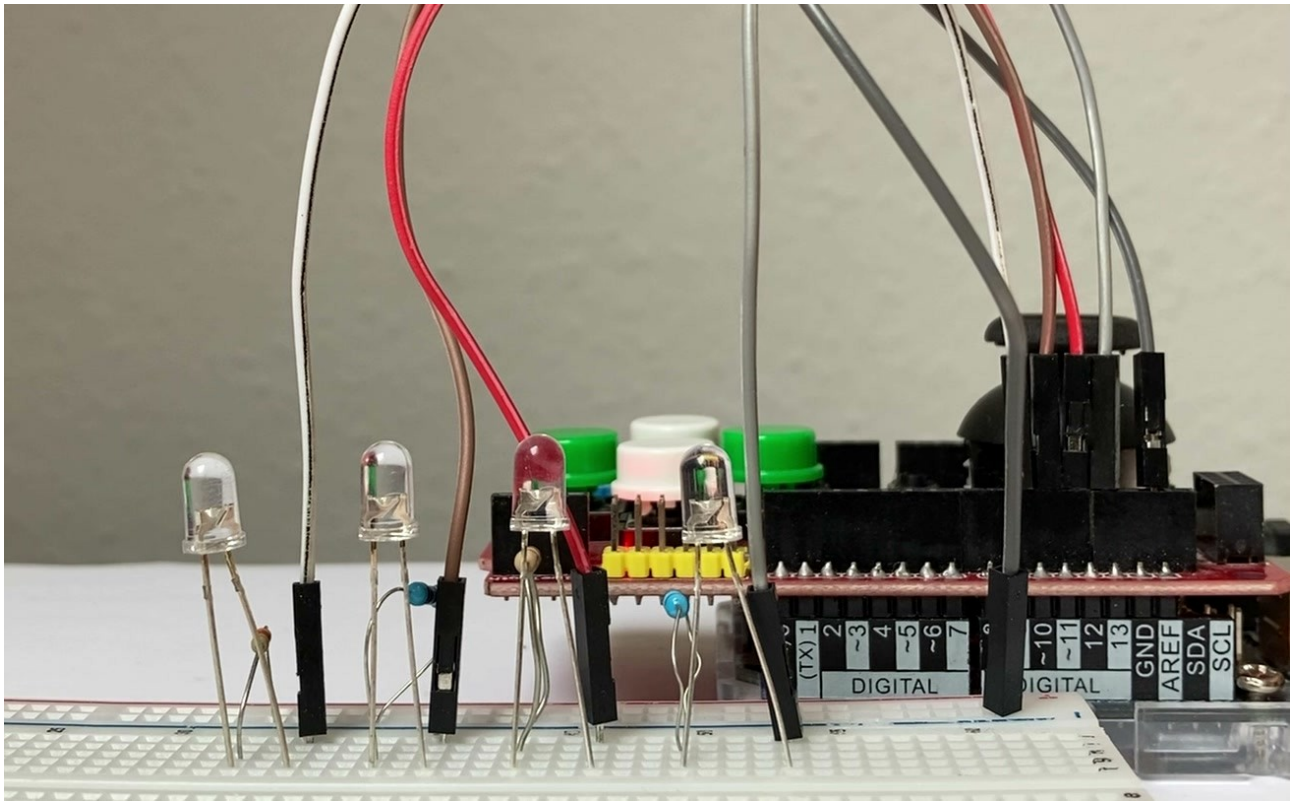
Inhaltsverzeichnis

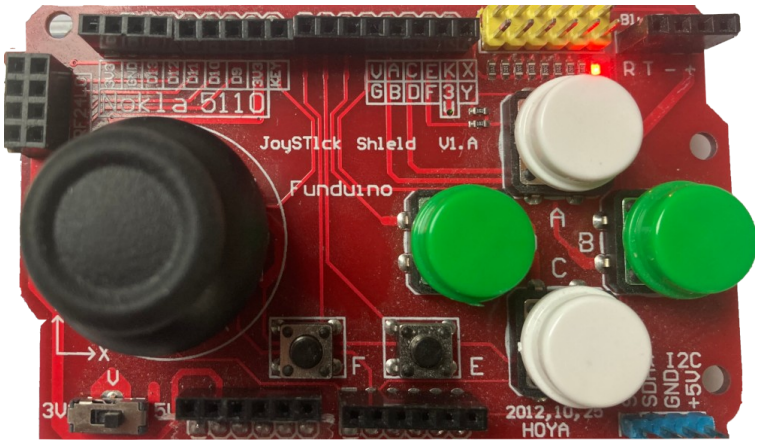
Aufbau des Joystick-Shields....2
Benötigte Bauteile....2
Pinbelegung des Joystick-Shields....2
Taster und die Bewegung des Joysticks abfragen....3
Erläuterungen zum Programm:....5
Arrays....5
sizeof....5
for-Schleife....5
if-Abfrage....6
Die Steuerung der LEDs mit Taster und Joystick....6
Definition der Variablen und Arrays....6
Der setup-Teil....7
Der loop-Teil....7

LEDs mit einem Joystick-Shield schalten

Mit einem Joystick-Shield sollen vier LEDs mit den Tastern geschaltet werden. Der Joystick steuert ein Lauflicht in beide Richtungen.



Aufbau des Joystick-Shields



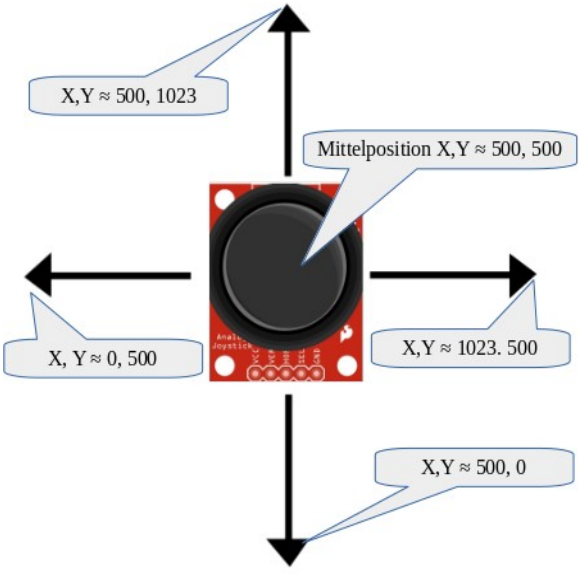
Das Joystick-Shield besitzt insgesamt sechs Taster, die mit den Buchstaben A bis E gekennzeichnet sind. Ein Joystick vervollständigt das Shield. Es wird einfach auf den Arduino aufgesteckt.

Benötigte Bauteile

- Joystick-Shield
- 4 LEDs
- 3 Widerstände 220 Ω (gelb, rot und grüne LEDs)
- 2 Widerstände 100 Ω (blaue und weiße LEDs)
- Leitungsdrähte

Pinbelegung des Joystick-Shields

Die digitalen Pins D2 bis D8 sind mit den Tastern und dem Joystick-Knopf belegt. Die LEDs werden deshalb an den digitalen Pins D9 bis D12 angeschlossen. Der Joystick belegt die analogen Anschlüsse A0 und A1.



A0 liest die Werte auf der X-Achse, A1 die Werte auf der Y-Achse.
Um die Bewegungen des Joysticks auszuwerten, reicht es aus, die Werte der jeweiligen Achsen abzufragen.

Taster und die Bewegung des Joysticks abfragen

Als Erstes sollen die Taster mit ihrem Namen und die Bewegung des Joysticks im Seriellen Monitor angezeigt werden.

Wird ein Taster längere Zeit gedrückt, kommt es zum sogenannten „Prellen“ des Tasters. In diesem Fall wird der Name des Tasters mehrfach angezeigt.

Das dazugehörige Programm:

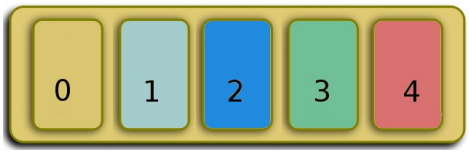
```
// Array Taster
// Reihenfolge A B C D
int Taster[] = { 2, 3, 4, 5, 8 };

// Namen der Taster
String NameTaster[] = { "A", "B", "C", "D" };

// analoge Pins Joystick
byte XAchse = A0;
byte YAchse = A1;

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

  // pinMode der 4 Taster festlegen: sizeof = Größe des Arrays
  // INPUT_PULLUP -> Vorwiderstand einschalten
  for (int i = 0; i <= sizeof(Taster); i++)
  {
    pinMode(Taster[i], INPUT_PULLUP);
  }
}
```

```
void loop()
{
  // Taster der Reihe nach abfragen
  for (int i = 0; i < sizeof(Taster); i++)
  {
    // wenn der jeweilige Taster gedrückt wird
    if (digitalRead(Taster[i]) == LOW)
    {
      // kurzes delay -> Prellen des Tasters verringern
      delay(200);

      // Name des Tasters zuordnen
      Serial.println("Taste: " + NameTaster[i]);
    }
  }

  // Bewegung der Y-Achse lesen
  int PositionX = analogRead(XAchse);

  // Bewegung X-Achse nach links
  if (PositionX < 300)
  {
    delay(200);
    Serial.println("Bewegung X-Achse nach links");
  }

  // Bewegung X-Achse nach rechts
  if (PositionX > 600)
  {
    delay(200);
    Serial.println("Bewegung X-Achse nach rechts");
  }

  // Bewegung der Y-Achse lesen
  int PositionY = analogRead(YAchse);

  // Bewegung Y-Achse nach oben
  if (PositionY > 600)
  {
    delay(200);
    Serial.println("Bewegung Y-Achse nach oben");
  }

  // Bewegung Y-Achse nach unten
  if (PositionY < 300)
  {
    delay(200);
    Serial.println("Bewegung Y-Achse nach unten");
  }
}
```

Erläuterungen zum Programm:

Arrays

Im Kopf des Programms werden zwei Arrays definiert.



Du kannst dir ein Array wie einen Schrank mit Schubladen vorstellen. In jeder Schublade befindet sich ein Element. Mit Hilfe von Arrays können Daten eines einheitlichen Typs (z. B. int oder Strings) im Speicher abgelegt und jederzeit wieder hervorgeholt werden.

Arrays können bereits bei der Definition mit Elementen gefüllt werden.

Array mit int:

```
int Taster[] = { 2, 3, 4, 5 };
```

Array mit String:

```
String NameTaster[] = { "A", "B", "C", "D" };
```

Der Zugriff auf ein Element des Arrays erfolgt über seine Position im Array.

Die Zählung beginnt mit 0!

Taster[0] hat den Wert 2, Taster[3] hat den Wert 5.

NameTaster[0] hat den Wert A, NameTaster[3] hat den Wert D.

sizeof

sizeof bestimmt die Größe eines Arrays

```
int Groesse = sizeof(Taster);
```

schreibt die Größe des Arrays Taster in die Variable Groesse.

for-Schleife

```
  for (int i = 0; i < sizeof(Taster); i++)
  {
    // wenn der jeweilige Taster gedrückt wird
    if (digitalRead(Taster[i]) == LOW)
    {
      // kurzes delay -> Prellen des Tasters verringern
      delay(200);

      // Name des Tasters zuordnen
      Serial.println("Taste: " + NameTaster[i]);
    }
  }
```

Die for-Schleife startet mit 0 und fragt den Taster im Array an der Position 0 (Taster an D2) ab digitalRead().

Weil der Vorwiderstand eingeschaltet ist (INPUT_PULLUP) wird gefragt, ob der digital gelesene Wert LOW ist. Ist das der Fall, wird der Name des Tasters ausgegeben, der sich an der gleichen Position im Array befindet:

Taster[0] wird NameTaster[0] zugeordnet: digitaler Pin D2 → A
Taster[3] wird NameTaster[0] zugeordnet: digitaler Pin D5 → D

if-Abfrage

Der gelesene Wert des Tasters wird mit den Werten HIGH oder LOW verglichen.

Wenn (if) er den Wert LOW hat, zeigt der Serielle Monitor den Namen des Tasters an.

Der Joystick wird ebenfalls mit einer if-Abfrage ausgewertet. Bei der Bewegung auf der X-Achse wird nur der Wert auf der X-Achse verglichen. Ist er kleiner als 300, wurde der Joystick nach links bewegt. Ist er größer als 600, wurde er nach rechts bewegt.

Auch die if-Abfrage der Y-Achse vergleicht nur die gelesenen Werte der Y-Achse.

Die Steuerung der LEDs mit Taster und Joystick

Jede der Taster soll eine LED schalten. Der Joystick soll in der X-Achse nach rechts ein Lauflicht nach rechts, die Bewegung nach links soll das Lauflicht nach links steuern.

Definition der Variablen und Arrays

Füge das Array für die LEDs hinzu:

```
// Array Taster
// Reihenfolge A B C D
byte Taster[] = { 2, 3, 4, 5, 8 };

// analoge Pins Joystick
byte XAchse = A0;
byte YAchse = A1;

// Array für die LEDs
byte LED[5] = { 9, 10, 11, 12 };
```

Der setup-Teil

Im setup-Teil wird mit einer for-Schleife der pinMode für die LEDs ergänzt.

```
void setup()
{
  // Seriellen Monitor starten
  Serial.begin(9600);

  // pinMode der 4 Taster festlegen: sizeof = Größe des Arrays
  // INPUT_PULLUP -> Vorwiderstand einschalten
  for (int i = 0; i <= sizeof(Taster); i++)
  {
    // INPUT_PULLUP -> Vorwiderstand einschalten
    pinMode(Taster[i], INPUT_PULLUP);
  }

  // pinMode der LEDs festlegen
  for (int i = 0; i < sizeof(LED); i++)
  {
    pinMode(LED[i], OUTPUT);
  }
}
```

Innerhalb der for-Schleife wird jetzt die dem Taster zugeordnete LED eingeschaltet.

Taster[0] wird LED[0] am digitalen Pin D9 zugeordnet
Taster[3] wird LED[3] am digitalen Pin D12 zugeordnet

Die Abfrage der Bewegung des Joysticks wird mit dem Lauflicht ergänzt.

Der loop-Teil

```
void loop()
{
  // Zustand der Taster der Reihe nach abfragen
  for (int i = 0; i < sizeof(Taster); i++)
  {
    // wenn der jeweilige Taster gedrückt wird
    // wird der Zustand der jeweiligen LED true
    // -> LED wird eingschaltet
    if (digitalRead(Taster[i]) == LOW)
    {
      digitalWrite(LED[i], HIGH);
      Serial.println("Taste: " + String(Taster[i]));
    }
    // wenn der Taster nicht gedrückt wurde
    // -> LED bleibt ausgeschaltet
    else digitalWrite(LED[i], LOW);
  }
```

```
  // Bewegung der Y-Achse lesen
  int PositionX = analogRead(XAchse);

  // Bewegung X-Achse nach links
  // Lauflicht nach links,
  // for-Schleife läuft rückwärts
  if (PositionX > 600)
  {
    for (int i = 0; i < sizeof(LED); i++)
    {
      digitalWrite(LED[i], HIGH);
      delay(200);
      digitalWrite(LED[i], LOW);
    }
  }

  // Bewegung X-Achse nach links
  // Lauflicht nach links
  if (PositionX < 300)
  {
    for (int i = sizeof(LED); i >= 0; i--)
    {
      digitalWrite(LED[i], HIGH);
      delay(200);
      digitalWrite(LED[i], LOW);
    }
  }
}
```