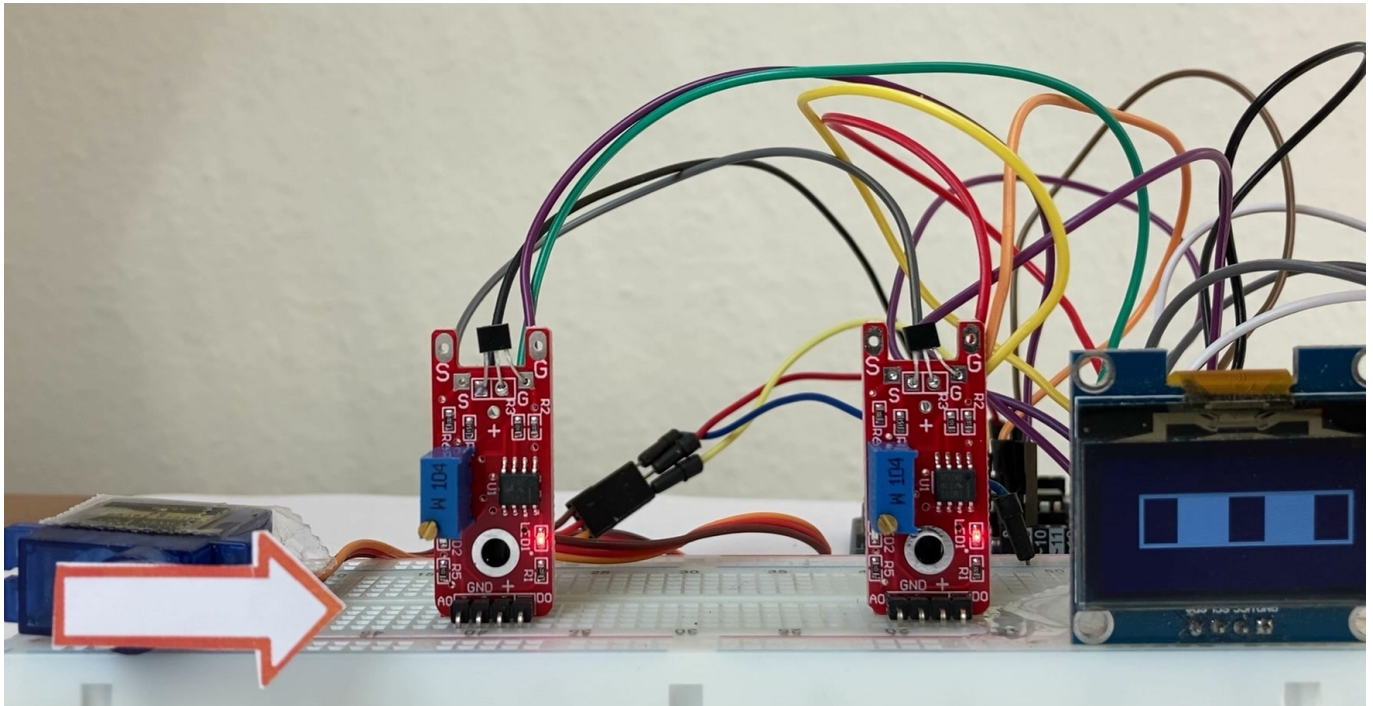


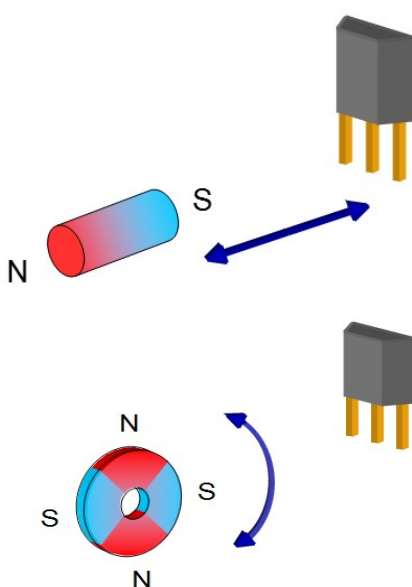
Das Programm simuliert eine automatische Tür. Wird der erste Hall-Sensor passiert, öffnet sich die Schranke. Beim „Weitergehen“ schließt der zweite Hall-Sensor die Tür.



Hall-Sensoren (nach Edwin Hall) bestehen aus einem stromdurchflossenen Halbleiter-Element und einem dahinter fest eingebauten Permanentmagneten. Dadurch ist das Halbleiter-Element magnetisch vorgespannt.

Wenn ein anderer Magnet in die Nähe dieses Magnetfeldes kommt, verändert sich die messbare Spannung im Halbleiter-Element.

Anwendungen von Hall-Sensoren:



Die An- oder Abwesenheit eines Magnetfeldes wird verwendet, um Geräte ein- oder auszuschalten, zum automatischen Öffnen/Schließen von Fenstern und Türen oder der Überwachung des Zustands von Fenstern/Türen bei Alarmanlagen.

Geschwindigkeit und Drehrichtung von Motoren bestimmen, Überwachung der korrekten Funktion von Motoren, besonders in der Fahrzeugtechnik

Quelle: <https://www.bba.ch/de/technische-infos/technische-infos/hallsensorbetaetigung> (z. T. Eigene Bearbeitung)

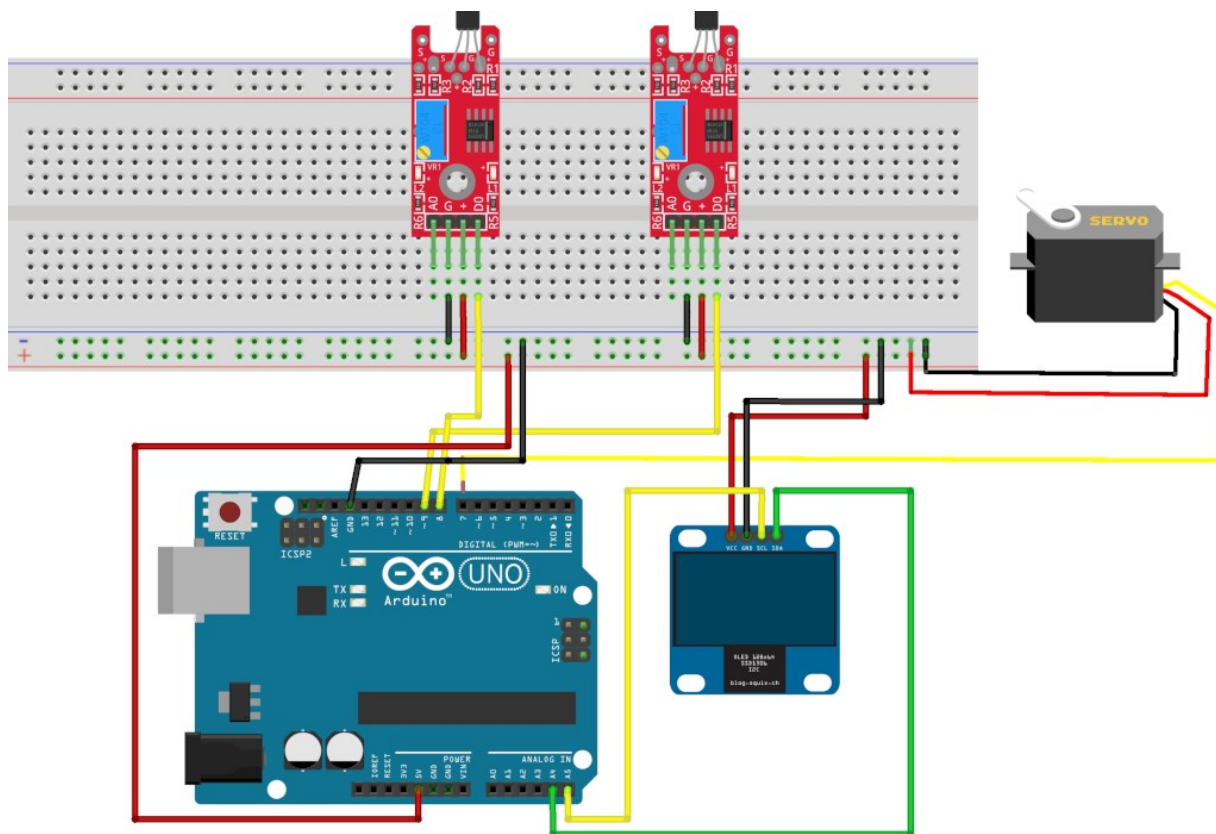
Hall-Sensoren unterscheiden sich in ihrer Funktionsweise:

- unipolar:** reagieren nur auf einen Pol des Magneten
- bipolar:** reagieren auf beide Pole eines Magneten
- latching:** bei der Anwesenheit eines Magnetfeldes wechselt der Zustand und wird auch nach der Entfernung des Magneten beibehalten, wird erneut ein Magnetfeld erkannt, wechselt der Zustand wiederum
- non-latching:** bei der Anwesenheit eines Magnetfeldes wechselt der Zustand
bei Entfernung des Magneten wird der Ausgangszustand wiederhergestellt

Benötigte Bauteile:

- 2 Hall-Sensoren KY-024 oder 2 Magnetsensoren KY-021/KY-025
- OLED-Display
- Servomotor
- Leitungsdrähte
- kleiner Magnet

Baue die Schaltung auf.

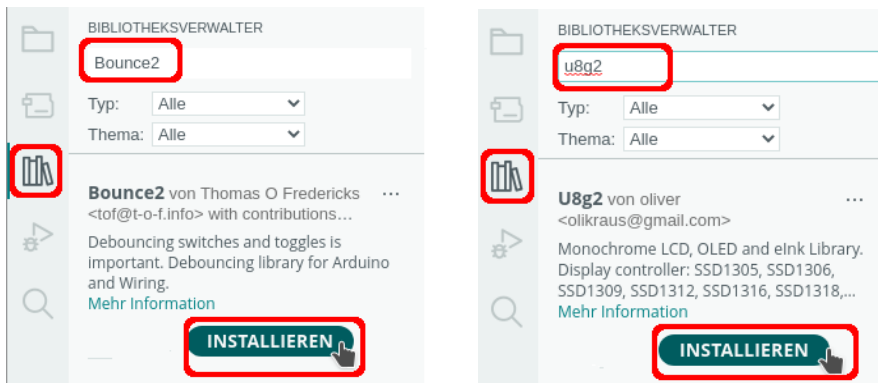


fritzing

Der Hall-Sensor KY-024 ist ein unipolarer Sensor.

Er reagiert nur auf einen Pol des Magneten. Bei Entfernung des Magneten wird der Ausgangszustand wiederhergestellt.

Benötigte Bibliotheken:



Binde die benötigten Bibliotheken ein und definiere die Variablen.

```
// eingebaute Bibliothek Servo einbinden
# include <Servo.h>

// Bounce2 -> Sensoren „entprellen“
# include <Bounce2.h>

# include <U8g2lib.h>

U8G2_SH1106_128X64_NONAME_1_HW_I2C u8g2(U8G2_R0, /* reset= */ U8X8_PIN_NONE);

// geschlossene Schranke
# define SchrankeGeschlossenBreite 100
# define SchrankeGeschlossenHoehe 28

static unsigned char SchrankeGeschlossen[] = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
    0xff, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
    0xff, 0xff, 0x0f, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff,
    0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc,
    0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00,
    0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01,
    0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff,
    0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff,
    0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc,
    0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00,
    0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03,
    0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c,
    0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00,
    0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03,
    0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff,
    0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff,
    0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc,
    0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00,
    0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x01,
    0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff,
    0x03, 0x00, 0x0c, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c, 0x03,
    0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00, 0x0c,
```

```

0x03, 0x00, 0xfc, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0x03, 0x00,
0x0c, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00
};
// offene Schranke
# define SchrankeOffenBreite 100
# define SchrankeOffenHoehe 50
static unsigned char SchrankeOffen[] = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0,
    0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0xfe, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0xe0, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0xfc, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x80, 0xff, 0x87, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0xf0, 0x7f, 0x80, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0xff, 0x0f, 0x00, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0xe0, 0xff, 0x01, 0x00, 0x0f, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0xfc, 0x3f, 0x00, 0x00, 0x0f, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0xc0, 0xff, 0x0f, 0x00, 0x00, 0x0e, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0x0f, 0x00, 0x00, 0x0e, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xff, 0x1f, 0x00, 0x00, 0x0e,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0xff, 0x1f, 0x00, 0x00,
    0x0c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfe, 0xff, 0xff, 0x1f, 0x00,
    0x00, 0x0c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xff, 0xff, 0x3f,
    0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff, 0xff,
    0x3f, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x87, 0xff,
    0xff, 0x7f, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xff, 0x80,
    0xff, 0xff, 0x7f, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x00, 0xfe, 0x1f,
    0x00, 0xff, 0xff, 0x7f, 0x00, 0x80, 0x0f, 0x00, 0x00, 0x00, 0xc0, 0xff,
    0x03, 0x00, 0xff, 0xff, 0xff, 0x00, 0xf0, 0x0f, 0x00, 0x00, 0x00, 0xf8,
    0xff, 0x00, 0x00, 0xfe, 0xff, 0xff, 0x00, 0xfe, 0x0f, 0x00, 0x00, 0x80,
    0xff, 0xff, 0x01, 0x00, 0xfe, 0xff, 0xff, 0xe0, 0xff, 0x01, 0x00, 0x00,
    0xf0, 0xff, 0xff, 0x01, 0x00, 0xfe, 0xff, 0xff, 0xfd, 0x3f, 0x00, 0x00,
    0x00, 0xfe, 0xff, 0xff, 0x01, 0x00, 0xfc, 0xff, 0xff, 0xff, 0x07, 0x00,
    0x00, 0xc0, 0xff, 0xff, 0xff, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x7f, 0x00,
    0x00, 0x00, 0xfc, 0xff, 0xff, 0xff, 0x03, 0x00, 0xfc, 0xff, 0xff, 0x0f,
    0x00, 0x00, 0x80, 0xff, 0xff, 0xff, 0xff, 0x07, 0x00, 0xf8, 0xff, 0xff,
    0x01, 0x00, 0x00, 0xf0, 0xff, 0xfc, 0xff, 0xff, 0x07, 0x00, 0xf8, 0xff,
    0x3f, 0x00, 0x00, 0x00, 0xff, 0x0f, 0xfc, 0xff, 0xff, 0x07, 0x00, 0xf0,
    0xff, 0x07, 0x00, 0x00, 0x00, 0xff, 0x01, 0xf8, 0xff, 0xff, 0x0f, 0x00,
    0xf0, 0xff, 0x00, 0x00, 0x00, 0x00, 0x3f, 0x00, 0xf8, 0xff, 0xff, 0x0f,
    0x00, 0xfc, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x03, 0x00, 0xf8, 0xff, 0xff,
    0x0f, 0x80, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0xf0, 0xff,
    0xff, 0x1f, 0xf0, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0xf0,
    0xff, 0xff, 0x1f, 0xfe, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00,
    0xe0, 0xff, 0xff, 0xff, 0xff, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03,
    0x00, 0xe0, 0xff, 0xff, 0xff, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x03, 0x00, 0xe0, 0xff, 0xff, 0xff, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x03, 0x00, 0xc0, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x07, 0x00, 0xc0, 0xff, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x07, 0x00, 0xc0, 0xff, 0xff, 0x01, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x80, 0xff, 0x3f, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x80, 0xff, 0x07, 0x00, 0x00,

```

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0x00, 0xe0, 0xff, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x00, 0xfc, 0x1f, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x80, 0xff, 0x03,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xf0, 0x7f,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xfe,
0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff,
0xff, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0xff, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0xff, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00
};
```

```
// Pins Hall-Sensoren
# define Sensor_1 8
# define Sensor_2 9

// Bounce starten
Bounce Start_1 = Bounce();
Bounce Start_2 = Bounce();

// Bezeichnung des Motors
Servo Motor;
```


[xbm mit GIMP erstellen](#)


Der setup-Teil. Beachte die Kommentare.

```
void setup()
{
  pinMode(Sensor_1, INPUT_PULLUP);
  pinMode(Sensor_2, INPUT_PULLUP);

  // Sensoren zuordnen
  Start_1.attach(Sensor_1);
  Start_2.attach(Sensor_2);

  // Intervall festlegen
  Start_1.interval(20);
  Start_2.interval(20);

  // Motor dem Pin zuordnen
  Motor.attach(7);

  // Motor in "geschlossen"-Position fahren
  Motor.write(5);

  // u8g2 starten
  u8g2.begin();

  // Farbe weiß
  u8g2.setDrawColor(1);

  // Display nicht rotieren
  u8g2.setDisplayRotation(U8G2_R0);
```

```
// geschlossene Schranke anzeigen
u8g2.firstPage();
do
{
    u8g2.drawXBM(10, 20, SchrankeGeschlossenBreite, SchrankeGeschlossenHoehe, SchrankeGeschlossen);
}
while (u8g2.nextPage());
}
```

Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
    // Sensor für das Öffnen der Schranke abfragen
    if (Start_1.update())
    {
        if (Start_1.read() == LOW)
        {
            // offene Schranke anzeigen
            u8g2.firstPage();
            do
            {
                u8g2.drawXBM(20, 1, SchrankeOffenBreite, SchrankeOffenHoehe, SchrankeOffen);
            }
            while (u8g2.nextPage());

            // Schranke öffnen
            Motor.write(90);
        }
    }

    // Sensor für das Schließen der Schranke abfragen
    if (Start_2.update())
    {
        if (Start_2.read() == LOW)
        {
            {
                // geschlossene Schranke anzeigen
                u8g2.firstPage();
                do
                {
                    u8g2.drawXBM(10, 20, SchrankeGeschlossenBreite, SchrankeGeschlossenHoehe, SchrankeGeschlossen);
                }
                while (u8g2.nextPage());

                // Schranke schließen
                Motor.write(5);
            }
        }
    }
}
```