April 21, 2023

# 1 Systemaufbau

Das System ist auf einer Experimentiersteckbrett (Breadboard) aufgebaut.

Der IO expander ist ein Microchip MCP 23017. Die Adresse ist auf 0 (0x20) festgelegt (A0 bis A2 auf GND). Die Bus-Leitungen (SCK, SDA) sind am MCP 23017 nicht mit pull_ups auf VCC gezogen. Am Pico sind die internen pullup-Widerstände aktiviert (siehe Listing3.)

An b0 ist ein BC550 angeschlossen, daran ein Finder 52.09.012 Relais. Das Relais ist an +12V (a0) angeschlossen, a1 ist am Kollektor des BC550 angeschlossen, der Emitter des BC550 ist an GND.

Die Tests sollen einen Einsatz als IO-Expander für einen Raspberry Pi vorbereiten.

Interessante Infos sind hier: https://netzmafia.ee.hm.edu/skripten/hardware/RasPi/Projekt-I2C-Expander/index.html

# 2 Test Pi Pico

Als Testgerät wird ein Raspberry Pi Pico W eingesetzt.

Der Extender ist an Pin 38 mit GND verbunden, Pin 36 mit 3,3V.

Das Testprogramm initialisiert den Bus i2c an den default-Pins (CSK an Pin 7 / GP5, SDA an Pin 6 / GP4).

allgemeine Inits:

in main:

Anschließend sendet das Programm die Befehlsfolge:

```
1 #include <stdio.h>
  #include "pico/stdlib.h"
3 #include "hardware/i2c.h"
  #include "pico/cyw43_arch.h"
```

Listing 1: Includes (Pi Pico)

```
uint8_t src[2];
uint8_t result = 2;
```

Listing 2: Variablen für Kommunikation

```
i2c_init(i2c_default, 100 * 1000);// 48000);
gpio_set_function(PICO_DEFAULT_I2C_SDA_PIN, GPIO_FUNC_I2C);
gpio_set_function(PICO_DEFAULT_I2C_SCL_PIN, GPIO_FUNC_I2C);
gpio_pull_up(PICO_DEFAULT_I2C_SDA_PIN);
gpio_pull_up(PICO_DEFAULT_I2C_SCL_PIN);
const int io_expander0 = 0x20;
```

Listing 3: init I2C-Bus

```
src[0] = 0x00;
src[1] = 0x00;
result = i2c_write_blocking(i2c_default, io_expander0, src, 2, false);
src[0] = 0x01;
src[1] = 0x00;
result = i2c_write_blocking(i2c_default, io_expander0, src, 2, false);
```

Listing 4: Init der Ports a und b auf Ausgang

```
{
    src[0] = 0x15;
    src[1] = 0x00;
    result = i2c_write_blocking(i2c_default, io_expander0, src, 2, false);
    src[0] = 0x14;
    src[1] = 0x01;
    result = i2c_write_blocking(i2c_default, io_expander0, src, 2, false);

    printf("i2c led on  result: %d\n", result);

    gpio_put(LED_PIN, 1); // IO an pico
    cyw43_arch_gpio_put(CYW43_WL_GPIO_LED_PIN, 1); // LED des pico, Modell w
    sleep_ms(1000);
    src[0] = 0x15;
    src[1] = 0x01;
    result = i2c_write_blocking(i2c_default, io_expander0, src, 2, false);
    src[0] = 0x14;
    src[1] = 0x00;
    result = i2c_write_blocking(i2c_default, io_expander0, src, 2, false);

    printf("i2c led off result: %d\n", result);
    gpio_put(LED_PIN, 0);
    cyw43_arch_gpio_put(CYW43_WL_GPIO_LED_PIN, 0);
    sleep_ms(1000);
    printf("blink\n");
}
```

Listing 5: Endlosschleife. Schaltet a0 und b0 zyklisch ein und aus

und initialisiert damit die Ports a und b als Ausgänge (alle Pins). mit printf("i2c result: %d\n", result); wird via Terminal ( minicom -b 115200 -o -D /dev/ttyACM0 ) die Rückmeldung des i2c-Bus ausgegeben. 2 = okay, 254 = nok.

In einer Endlosschleife werden dann die Pins a0 und b0 wechselnd auf 1 und 0 gesetzt: Das Ergebnis auf dem Bus zeigt Bild 1. Im Bild sind die Register 0x12 und 0x13 adressiert, dies ist im Script in 0x14 und 0x15 korrigiert.

Der Test war überwiegend erfolgreich verlaufen. Wenn durch einen Wackelkontakt Pin SCK oder SDA unterbrochen wurde, wurde der Kontakt nicht wieder hergestellt. Es musste durch trennen der Versorgung des Pico und ca. 30 - 60 sec. Warten das System neu initialisiert werden.

## 2.1 Zweiter Test Pi Pico

Testaufbau: Sender Pi Pico, Programm wie oben. Empfänger: 2x MCP 23017. Ein Expander (links) auf Adr. 0x20, der andere auf Adr. Oszilloskop

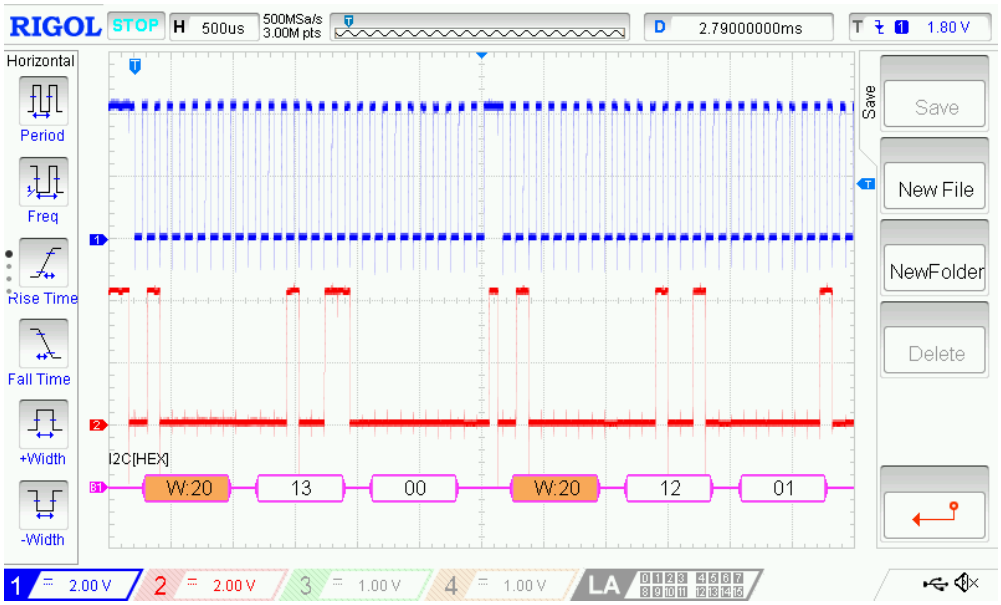


Figure 1: Ausgabe des I2C-Bus (Adr. 0x20, Register 0x13 und 0x12 sind im Code auf 0x15 und 0x14 korrigiert)

**Kanal 1** SCK an Pico,

**Kanal 2** SDA an Pico,

**Kanal 3** SCK an Expander links (0x22),

**Kanal 4** SDA an Expander links (0x22)

Der Expander links lässt sich nicht korrekt ansprechen. Das Signal am Pico ist in Ordnung, beim Signal am Expander links fehlt bei einigen Tests SCK.

Einspeisung erfolgt zuerst am Expander links. Dann am Expander rechts. Bei Einspeisung rechts ist links ist zeitweise kein SCK messbar.

Nach Vertauschen der Reihenfolge im Programm (zuerst an 0x22, dann 0x20) reagiert der Expander 0x22 auch nicht. siehe Bild 2

Bei beiden Expandern /Reset auf VCC gezogen, das behebt das Problem, von undefinierten ausfällen. Vorher war /Reset nicht beschaltet.

# 3 Test Pi 3b+

Als Pi wird ein Raspberry Pi 3b+ mit Raspian verwendet. Am Pi ist der i2c Bus aktiviert. zum Testen ist der Takt des i2c auf 1kHz begrenzt. Bei Messung mit dem Oszilloskop (Rigol DS 1054Z) ergibt sich, dass der Takt bei 100kHz bleibt (siehe Bild 3).

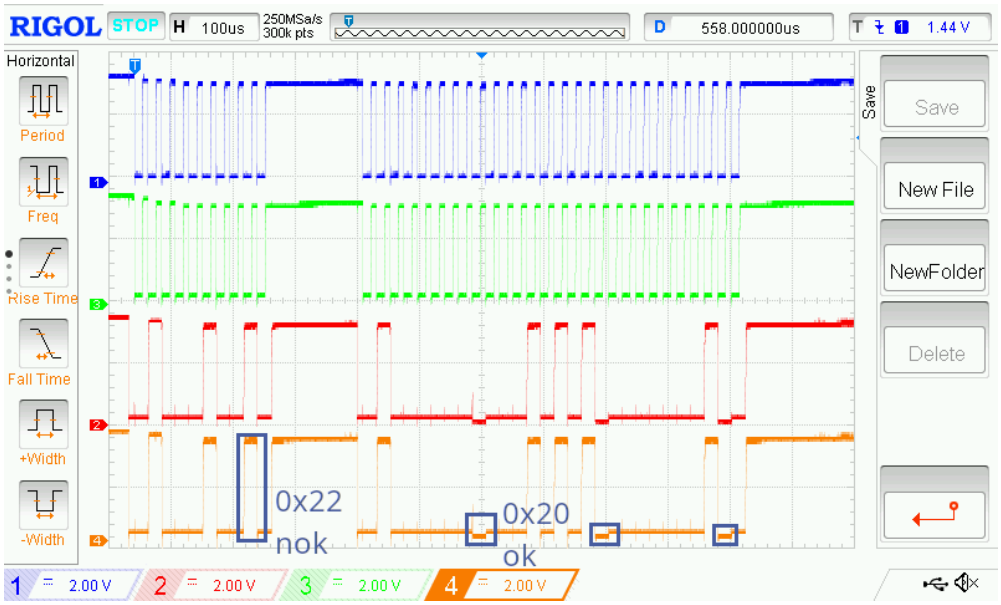


Figure 2: Zwei Expander. Sende Reihenfolge 0x22, dann 0x20

Ein Test mit i2cdetect −y 1 0x20 0x27 ergab folgende Situation: siehe Bild 3 An der Adresse 0x20 antwortete der IC planmäßig. An Adresse 0x21 kam keine Antwort (gekennzeichnet durch das "'?"') danach bricht das Signal unerwartet zusammen. Diese Zusammenbrüche sind bei Test mit dem Raspberry Pi mehrfach festzustellen, eine Ursache ist mir noch nicht bekannt. Ein erneuter Test mit detect ergab Bild 4. Der Expander konnte die Adresse nicht dekodieren und der Detect schlug damit fehl.

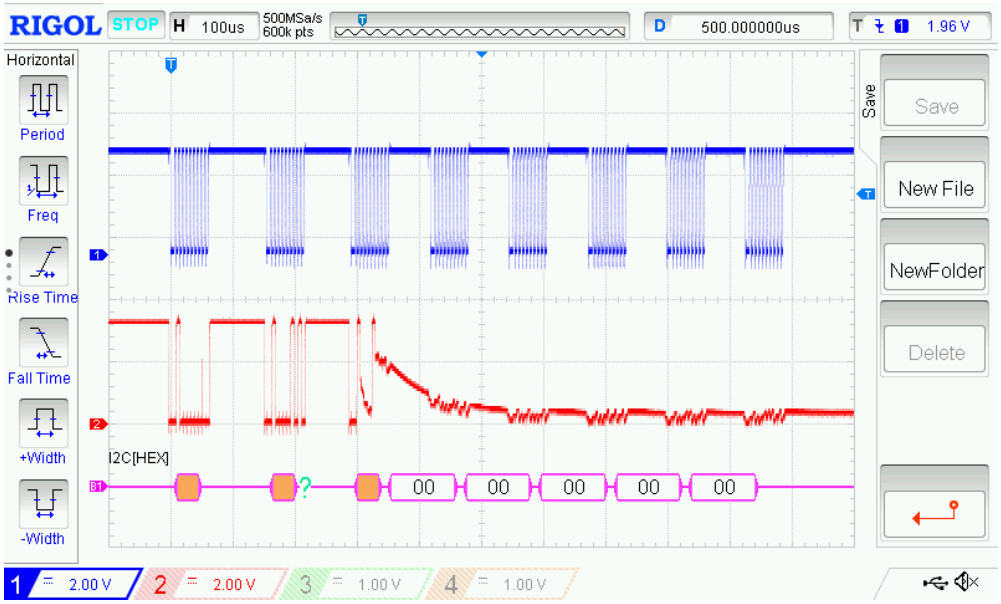


Figure 3: detect vom Raspberry Pi, Bereich 0x20 bis 0x27. Der Expander ist hier auf 0x20 eingestellt.

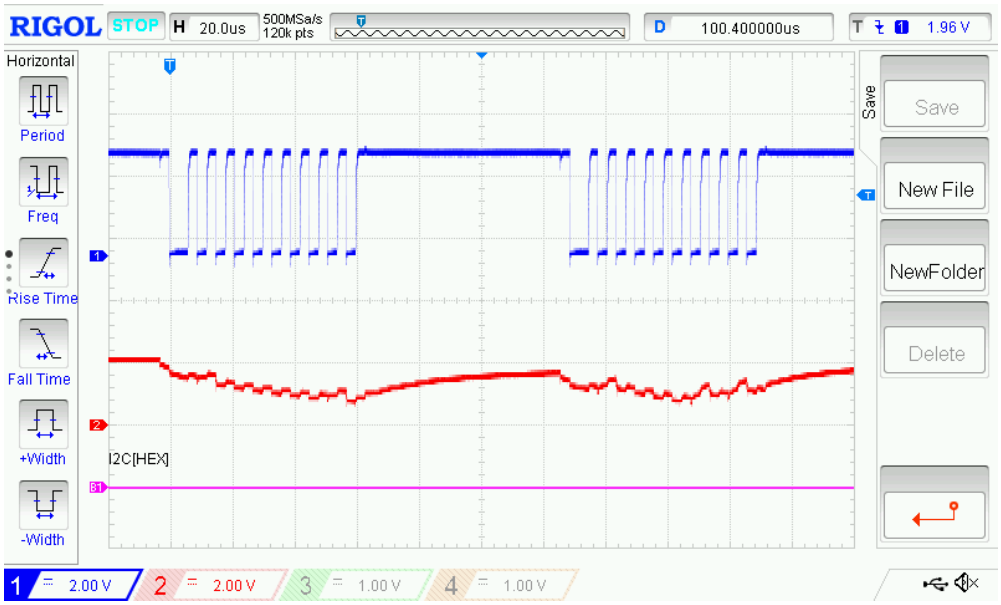


Figure 4: Detect am Pi. Signal ist nicht korrekt auf 0V gezogen.