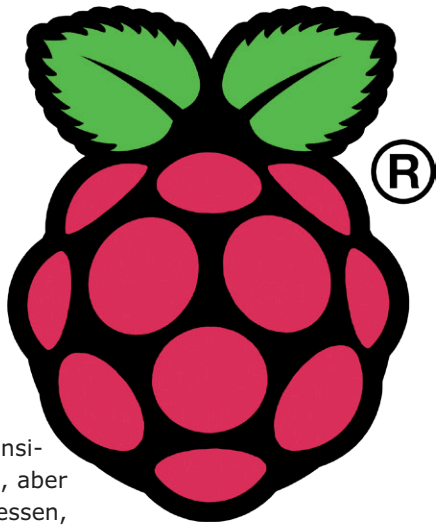
Raspberry Pi Rezepte Teil 5

Von **Tony Dixon** (UK)

I²C: Zwischen zwei ICs



In den letzten beiden Folgen von Elektor.POST ging es um die seriellen Schnittstellen UART und SPI auf dem Expansion Header von RPi. Nun vervollständigen wir das Ganze mit dem letzten seriellen Interface: I²C.

Die Inter-IC- oder I²C-Schnittstelle ist die letzte der drei auf dem Expansion Header von RPi befindlichen seriellen Interfaces. Bei den anderen beiden handelt es sich um den gewöhnlichen UART und die SPI-Schnittstelle, die schon in Teil 3 und 4 beschrieben wurden.

I²C-Details

Tabelle 1 listet die Signale des Expansion Headers auf. Die I²C-Anschlüsse sind Pin 3 (SDA) und Pin 5 (SCL).

Wie SPI wurde auch das I²C-Interface zum Anschluss anderer Geräte mit möglichst wenigen Leitungen konzipiert. Bei I²C gibt es nur zwei bidirektionale Open-Drain-Leitungen: SDA (Serial Data Line) und SCL (Serial Clock). Beide sind typischerweise über Pullup-Widerstände mit der 3,3-V-Versorgung verbunden. Bei RPi sind das zwei 1,8-kΩ-Widerstände. I²C ist zwar nicht so schnell wie z.B. SPI, aber 100 Kbit/s im Standard-Mode und 400 Kbit/s im Fast-Mode sind so langsam nicht.

Der von RPi genutzte Broadcom-Controller bietet zwei I²C-Interfaces. Die ursprüngliche Version von RPi verfügte nur über ein I²C-Interface auf dem Expansion Header (I2C_SDA0 und I2C_SCL0). Die zweite RPi-Revision ist mit einem weiteren kleinen Expansion Header ausgestattet, was Zugang zu einem zweiten I²C-Interface ermöglicht. Dabei gab es etwas Chaos: Bei Revision 2 befindet sich auf dem normalen Expansion Header jetzt das zweite I²C-Interface (I2C_SDA1 und I2C_SCL1). Das erste I²C-Interface (I2C_SDA0 und I2C_SCL0) wurde auf den neuen, kleineren Expansion-Header verlegt – nicht dramatisch, aber man sollte diese Änderung nicht vergessen, wenn es um I²C mit RPi geht.

Tabelle 1. Pin-Belegung des Expansion-Headers

Pin-Name	Pin-Funktion	Alternative	RPi.GPIO
P1-02	5,0V	-	-
P1-04	5,0V	-	-
P1-06	GND	-	-
P1-08	GPIO14	UART0_TXD	RPi.GPIO8
P1-10	GPIO15	UART0_RXD	RPi.GPIO10
P1-12	GPIO18	PWM0	RPi.GPIO12
P1-14	GND	-	-
P1-16	GPIO23		RPi.GPIO16
P1-18	GPIO24		RPi.GPIO18
P1-20	GND	-	-
P1-22	GPIO25		RPi.GPIO22
P1-24	GPIO8	SPI0_CE0_N	RPi.GPIO24
P1-26	GPIO7	SPI0_CE1_N	RPi.GPIO26

Pin-Name	Board Revision 1		Board Revision 2	
	Pin-Funktion	Alternative	Pin-Funktion	Alternative
P1-01	3,3V	-	3,3V	-
P1-03	GPIO0	**I2C0_SDA**	GPIO2	**I2C1_SDA**
P1-05	GPIO1	**I2C0_SCL**	GPIO3	**I2C1_SCL**
P1-07	GPIO4	GPCLK0	GPIO4	GPCLK0
P1-09	GND	-	GND	-
P1-11	GPIO17	RTS0	GPIO17	RTS0
P1-13	GPIO21		GPIO27	
P1-15	GPIO22		GPIO22	
P1-17	3,3V	-	3,3V	-
P1-19	GPIO10	SPI0_MOSI	GPIO10	SPI0_MOSI
P1-21	GPIO9	SPI0_MISO	GPIO9	SPI0_MISO
P1-23	GPIO11	SPI0_SCLK	GPIO11	SPI0_SCLK
P1-25	GND	-	GND	-

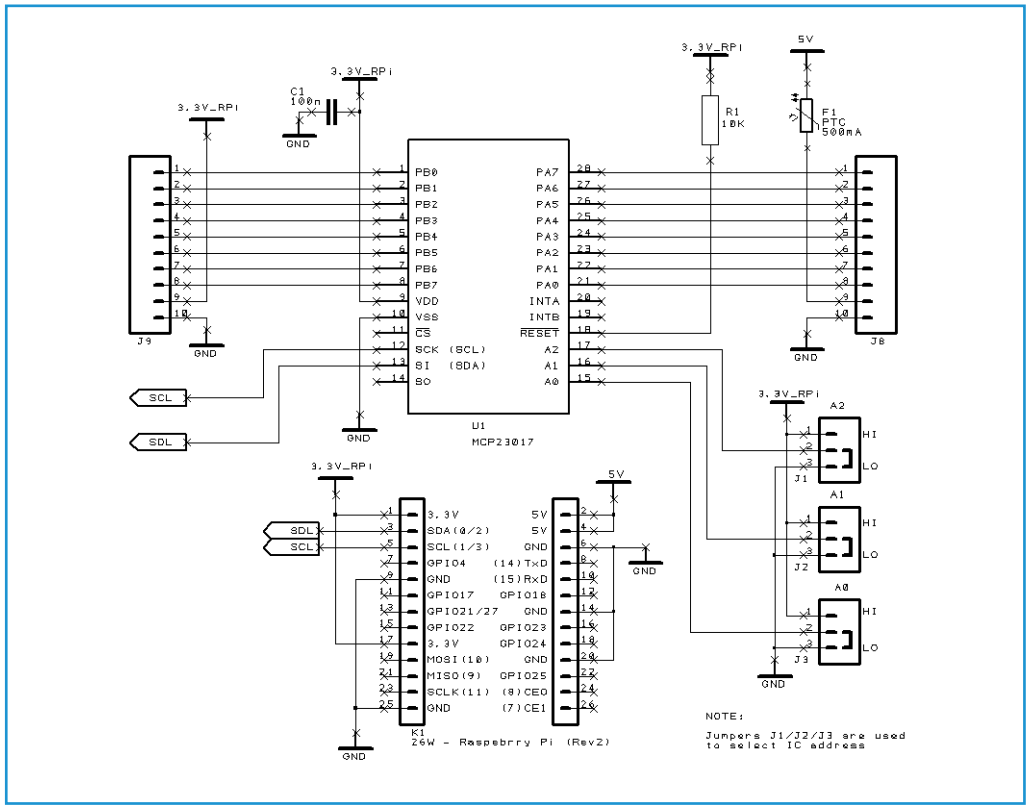


Bild 1.
Schaltung des RPi-Port-Expanders mit MCP23017.

Port Expander (die 2te)

Auch bei unserem RPi-I²C-Projekt werden die GPIO-Pins wieder durch einen zusätzlichen Port-Expander erweitert. Auch dieses Mal kommt ein Port-Expander von Microchip zum Einsatz, aber nun die 16-kanalige Variante MCP23017 [1] – quasi die I²C-Ausführung des im SPI-Projekt eingesetzten Chips MCP23S17.

Bild 1 zeigt eine vereinfachte MCP23017-Schaltung. Der Chip ist mit dem I²C-Interface des RPi verbunden. Anders als bei der SPI-Version ist hier kein Chip-Enable-Signal notwendig, was die Schaltung vereinfacht. Mit den Jumpern J1, J2 und J3 wird dem Port-Expander eine Adresse zur Verfügung gestellt. Auf diese Weise können mehrere Port-Expander pro RPi angeschlossen werden.

Bild 2 zeigt den Hardware-Aufbau; der MC23017 befindet sich auf einer kleinen Zusatzplatine. Vielleicht ist es schon aufgefallen: Es ist genau das gleiche Board wie zuvor [2]. Das ist kein Versehen, denn diese Platine kann sowohl per I²C mit dem IC MCP23017 als auch per SPI mit dem Chip MCP23S17 betrieben werden. Zwischen beiden Modi lässt sich per Jumper wählen.

I²C-Tools

Vor der Installation der I²C-Tools muss man etwas aufräumen und Raspbian mitteilen, dass jetzt das Hardware-I²C-Interface verwendet werden soll. Standardmäßig ist nämlich I²C deaktiviert. Man muss dies zunächst durch Editieren der Blacklist-Datei aktivieren:

```
sudo nano /etc/modules
```

Man suche den Text „blacklist i2c-bcm2708“ und füge am Anfang dieser Zeile ein „#“ (Doppelkreuz) ein und kommentiere sie so aus. Dann wird die Datei gesichert.
Nun editiert man die Module-Datei:

```
sudo nano /etc/Module
```

Der Text „i2c-dev“ kommt in eine neue Zeile, dann wird die Datei gesichert.
Jetzt wird das Paket mit den I²C-Tools installiert:

```
sudo apt-get update
sudo apt-get install i2c-tools
```

Nachdem dies geschehen ist, muss zur „i2c group“ ein neuer User hinzugefügt werden:

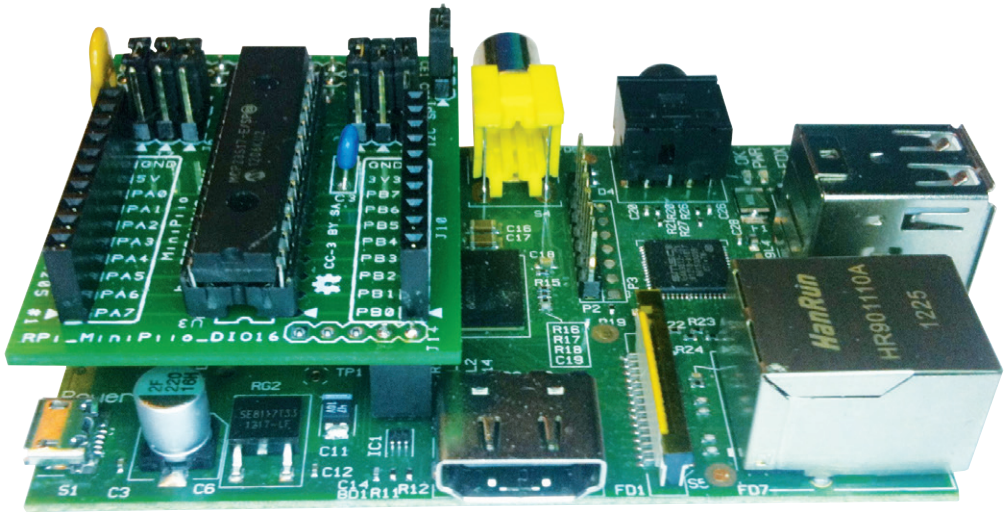


Bild 2.
RPi und Zusatzplatine mit MCP23017.

```
sudo adduser pi i2c
```

Ein Neustart geht so:

```
sudo reboot
```

Nach dem Booten können die I²C-Interfaces überprüft werden. Hierzu startet man eine neue LXTerminal-Session und tippt:

```
ls /dev/i2c*
```

Die beiden I²C-Geräte sollten so gelistet werden (ein Gerät pro I²C-Interface):

```
/dev/i2c-0
/dev/i2c-1
```

Ein Test mit RPi Rev 1 geht so:

```
sudo i2cdetect -y 0
```

Mit einen RPi Rev. 2 wird so getestet:

```
sudo i2cdetect -y 1
```

Nun sollte man so etwas wie in **Bild 3** sehen können.

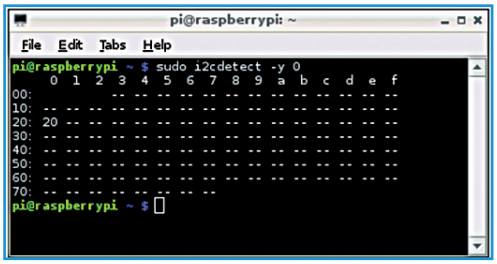


Bild 3.
Ergebnisse von „i2cdetect“.

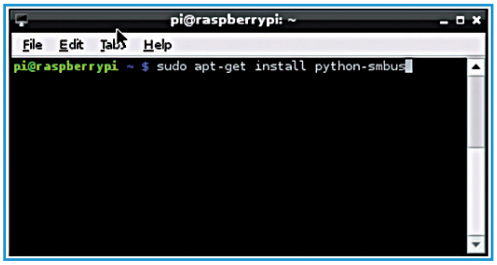


Bild 4.
LXTerminal.

Smbus-I²C-Library

Die Beispiele dieses Projekts basieren auf Python 2. Wie schon aus den bisherigen Folgen klar sein dürfte, ist Python schon als Teil der Raspbian-Distribution installiert.

Leider gibt es da aber keine Unterstützung für das I²C-Interface. Zwecks Nachrüstung installiert man die I²C-Python-Wrapper-Library. Hierzu startet man eine LXterminal-Session wie in **Bild 4** und gibt den folgenden Befehl ein:

```
sudo apt-get install python-smbus
```

Nach dieser Installation kann man I²C mit Python nutzen.

Beispiel mcp23017.py

Nach der Installation der Library braucht man ein kleines Testprogramm, um die an den Port-Expander angeschlossenen LEDs leuchten zu lassen. Hierzu beginnen wir mit einem Doppelklick auf das IDLE-Icon auf dem Desktop des RPi, um die Python-Shell und IDLE zu starten (siehe **Bild 5**). Jetzt wird per File-Menü ein neues Programm erstellt. Hierdurch wird der IDLE-Editor gestartet.

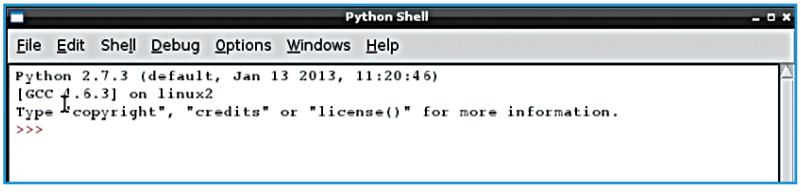


Bild 5.
IDLE-Shell von Python.

Im IDLE-Editor (**Bild 6**) wird dann der Code von **Listing 1** eingegeben. Nach der Eingabe wird das Programm gesichert. Nun wechseln wir zu LXTerminal und geben dort die folgenden Befehle ein, um das Programm ausführbar zu machen:

```
sudo chmod +x mcp23017.py
```

Anschließend kann das Programm mit dem folgenden Befehl gestartet werden:

```
sudo ./mcp23017.py
```

Tabelle 2 enthält einen Überblick über die Control-Register des MCP23x17.

(130236)

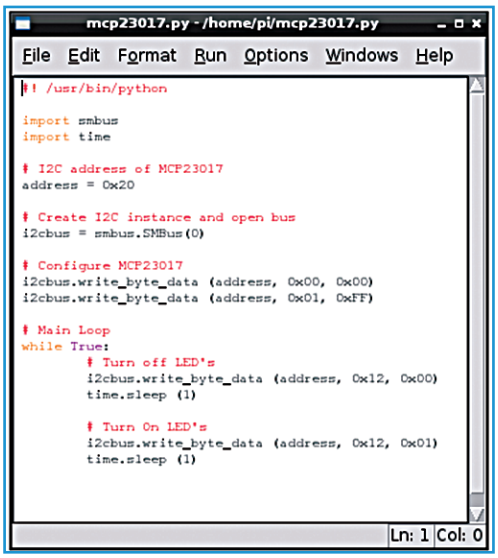


Bild 6.
IDLE-Editor.

Listing

```
#! /usr/bin/python

import smbus
import time

# I2C address of MCP23017
address = 0x20

# Create I2C instance and open bus
i2cbus = smbus.SMBus(0)

# Configure MCP23017
i2cbus.write_byte_data(address,0x00,0x00) # Set Bank A to outputs
i2cbus.write_byte_data(address,0x01,0xFF) # Set Bank B to inputs

# Main loop
while True:
  # Turn off LEDs
  i2cbus.write_byte_data (address,0x12,0x00)
  time.sleep(1)

  # Turn on PortA.0
  i2cbus.write_byte_data (address,0x12,0x01)
  time.sleep(1)
```

Hinweis: Für RPi Rev 2 Pi ändert man die Zeile:

`i2cbus = smbus.SMBus(0)` **in:** `i2cbus = smbus.SMBus(1)`

Tabelle 2. Register-Adressen der MCP23x17-Serie

Adresse IOCON.BANK = 1	Adresse IOCON.BANK = 0	Register	Beschreibung
0x00 / 0 dec	0x00 / 0 dec	IODIRA	I/O Direction Register Port A
0x10 / 16 dec	0x01 / 1 dec	IODIRB	I/O Direction Register Port B
0x01 / 1 dec	0x02 / 2 dec	IPOLA	Input Polarity Port Register Port A
0x11 / 17 dec	0x03 / 3 dec	IPOLB	Input Polarity Port Register Port B
0x02 / 2 dec	0x04 / 4 dec	GPINTENA	Interrupt-n-Change Control Register Port A
0x12 / 18 dec	0x05 / 5 dec	GPINTENB	Interrupt-n-Change Control Register Port B
0x03 / 3 dec	0x06 / 6 dec	DEFVALA	Default Compare Register GPINTENA
0x13 / 19 dec	0x07 / 7 dec	DEFVALB	Default Compare Register GPINTENB
0x04 / 4 dec	0x08 / 8 dec	INTCONA	Interrupt Control Register Port A
0x14 / 20 dec	0x09 / 9 dec	INTCONB	Interrupt Control Register Port B
0x05 / 5 dec	0x0A / 10 dec	IOCON	I/O Expander Configuration Register
0x15 / 21 dec	0x0B / 11 dec	IOCON	I/O Expander Configuration Register
0x06 / 6 dec	0x0C / 12 dec	GPPUA	Pull-Up Resistor Configuration Register Port A
0x16 / 22 dec	0x0D / 13 dec	GPPUB	Pull-Up Resistor Configuration Register Port B
0x07 / 7 dec	0x0E / 14 dec	INTFA	Interrupt Flag Register Port A
0x17 / 23 dec	0x0F / 15 dec	INTFB	Interrupt Flag Register Port B
0x08 / 8 dec	0x10 / 16 dec	INTCAPA	Interrupt Capture Register Port A
0x18 / 24 dec	0x11 / 17 dec	INTCAPB	Interrupt Capture Register Port B
0x09 / 9 dec	0x12 / 18 dec	GPIOA	Port Register Port A
0x19 / 25 dec	0x13 / 19 dec	GPIOB	Port Register Port B
0x0A / 10 dec	0x14 / 20 dec	OLATA	Output Latch Register Port A
0x1A / 26 dec	0x15 / 21 dec	OLATB	Output Latch Register Port B

Weblinks

[1] ww1.microchip.com/downloads/en/devicedoc/21952b.pdf

[2] www.dtronixs.com