

PCF8575 Expander Modul

Beschreibung

Der PCF8575 ist eigentlich die Bezeichnung des Chipsatzes, aber es gibt auch fertig konfektionierte Breakout-Boards, wie z. B. dieses hier:

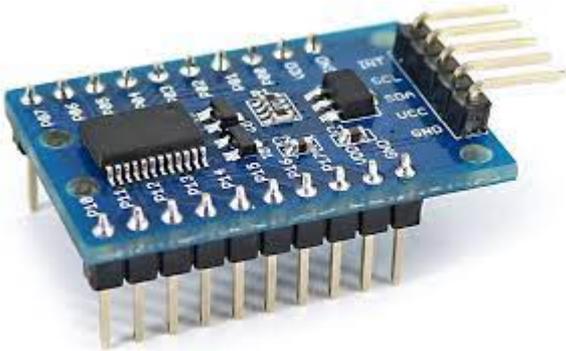


Abbildung: © FXCO

Ein Chip bzw. Modul kann 16 IOs bereitstellen. Je nach Konfiguration können diese als Ein- oder Ausgang dienen. Über einen Interrupt (der an einen GPIO des Raspberry angeschlossen werden muss), werden bei Eingängen Änderungen in Echtzeit eingelesen. Es können bis zu 8 Module parallel betrieben werden (also insgesamt 128 IOs). Dazu werden diese im "Daisy Chain"-Verfahren einfach hintereinander geschaltet. Wenn das nicht ausreicht, können weitere 8 Module des [Schwester-Chipsatzes PCF8574\(A\)](#) verwendet werden und so bis zu bis zu 256 IOs an den Raspberry angeschlossen werden!

Datenblatt

PCF8575: <https://www.ti.com/lit/ds/symlink/pcf8575.pdf>

Hardware-Konfiguration

Das Modul wird an den I2C-Bus des Raspberry angeschlossen (PIN SDA, SCL) und benötigt zusätzlich die 3.3V Spannungsversorgung des Rasperrys (Anschluss an VCC und GND). Das Modul darf **nicht** an die 5V Spannungsversorgung des Rasperrys angeschlossen werden!

Wenn das Modul als Eingangskarte verwendet werden soll, so muss die Interrupt-Leitung (PIN INT) des Moduls noch mit einem GPIO des Rasperrys verbunden werden.

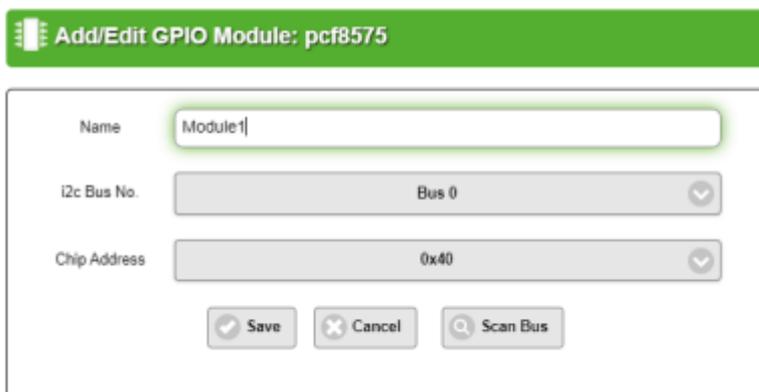
Welche Bus-Adresse das Modul verwendet, muss über 3 Jumper eingestellt werden. Möglich sind Adressen zwischen 0x40 und 0x4E

Die Jumper werden wie folgt für die einzelnen Bus-Adressen gesetzt:

Address (Hex) PCF8575	A2	A1	A0
0x40	LOW	LOW	LOW
0x42	LOW	LOW	HIGH
0x44	LOW	HIGH	LOW
0x46	LOW	HIGH	HIGH
0x48	HIGH	LOW	LOW
0x4A	HIGH	LOW	HIGH
0x4C	HIGH	HIGH	LOW
0x4E	HIGH	HIGH	HIGH

Software-Konfiguration

Modul



Add/Edit GPIO Module: pcf8575

Name: Module1

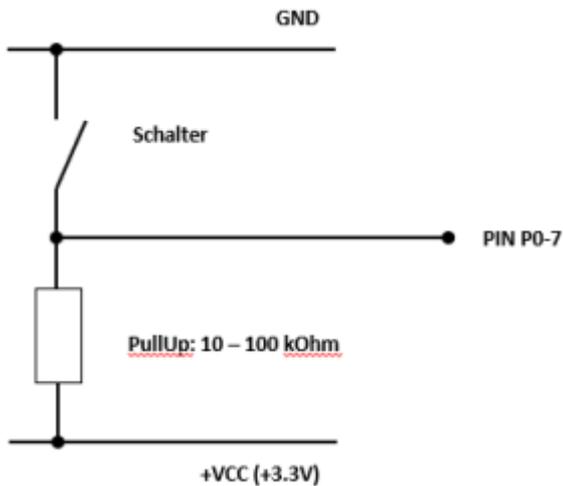
I2c Bus No.: Bus 0

Chip Address: 0x40

Save Cancel Scan Bus

Eingänge

Das Modul hat eingebaute PullUp-Widerstände (keine Pulldown-Widerstände). Man verschaltet die Eingänge also so, dass sie im geschlossenen Zustand den PIN auf GND ziehen. **Ich empfehle dringend zusätzlich zu den internen Widerständen externe PullUp-Widerstände zu verwenden!** Ohne externe Widerstände konnte ich keine stabile Erkennung der Eingangszustände realisieren.



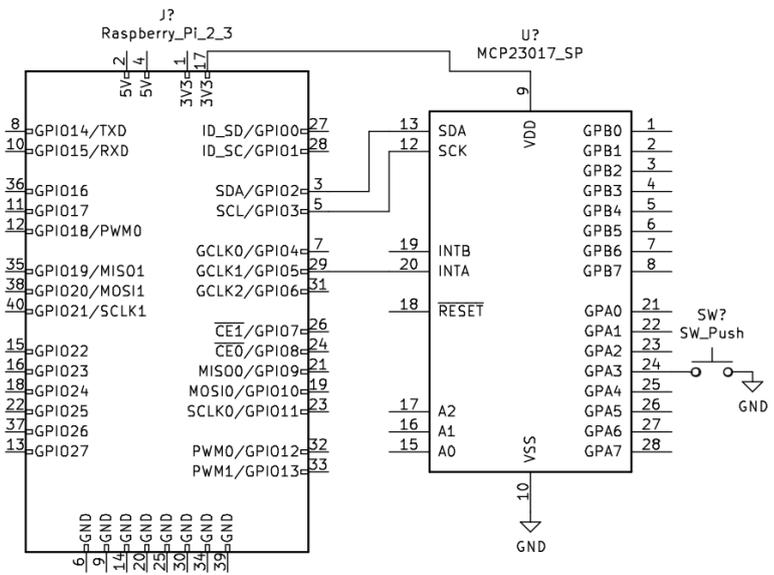
Verschaltung mit externem PullUp-Widerstand

Konfiguration mit Nutzung von Interrupts (empfohlen)

Standardmäßig werden die Zustände der Eingänge "gepollt", das heißt in einem gewissen Abstand fragt das Plugin den Zustand der Eingänge ab (ob EIN oder AUS). Das ist nicht nur sehr ineffektiv (CPU Last), es birgt bei sehr kurzen Eingangssignalen (z. B. ein kurzer Impuls) auch die Gefahr, dass man ein Eingangsimpuls "verpasst", wenn dieser gerade zwischen dem Poll-Intervall liegt.

Die Lösung ist die Verwendung von sogenannten Interrupts. Interrupts signalisieren dem Betriebssystem eine Statusveränderung z. B. eines PINs. So ist es möglich den Status der Eingänge nur abzufragen, wenn sich der Status geändert hat (im Gegensatz zum permanenten, zyklischen Abfragen).

Die Module haben zu diesem Zweck einen INT-Pin, der auf LOW gesetzt wird, sobald sich der Zustand eines der PINs am Modul ändert. Der INT-Pin wird wieder auf HIGH gesetzt, sobald der Zustand aller Pins einmal abgefragt wurde. Mit Hilfe eines GPIO-Eingangs auf dem Raspberry kann man so eine Interrupt-gesteuerte Abfrage aufbauen. Dazu verbindet man den INT-Pin mit einem freien GPIO des Raspberry und konfiguriert diesen GPIO dann als "Interrupt für das PCF-Modul". Die GPIOs des Raspberry unterstützen echte Interrupts, das heißt das Betriebssystem bekommt eine Information, das sich der Status am GPIO geändert hat. Das Plugin erkennt das entsprechend und löst die Status-Abfrage des PCF-Moduls aus. Sobald sich also ein Pin des Moduls im Status ändert, wird der GPIO des Raspberry getriggert und dieser löst die Statusabfrage der einzelnen Pins des Moduls aus.



Verschaltung am Beispiel des MCP23017 - die Verschaltung ist beim PCF identisch

Zunächst konfiguriert man alle Eingänge am Modul:

- Inverted: true
- Resistor: Pull-Up
- Interrupt: Both

 **Add/Edit Digital Input: pcf8574**

Module Name	<input type="text" value="pcf"/>
Name	<input type="text" value="Test"/>
Pin	<input type="text" value="P1"/>
Payload for ON	<input type="text" value="ON"/>
Payload for OFF	<input type="text" value="OFF"/>
	<input checked="" type="checkbox"/> Inverted Logic Level
Enable Resistor	<input type="text" value="Pull-Up"/>
Interrupt	<input type="text" value="Both"/>
Polling Interval	<input type="text" value="1"/>
Bouncetime	<input type="text" value="180"/>
<input type="button" value="Save"/> <input type="button" value="Cancel"/>	

Und dann den GPIO des Raspberry, der mit dem INT-Pin des Moduls verbunden ist wie folgt:

- Resistor: Pull-Up
- Interrupt: Falling
- Input is Interrupt for: <PCF Modulname>

Add/Edit Digital Input: raspberrypi

Module Name	<input type="text" value="raspberrypi"/>
Name	<input type="text" value="Interrupt"/>
Pin	<input type="text" value="GPIO18"/>
Payload for ON	<input type="text" value="ON"/>
Payload for OFF	<input type="text" value="OFF"/>
	<input type="checkbox"/> Inverted Logic Level
Enable Resistor	<input type="text" value="Pull-Up"/>
Interrupt	<input type="text" value="Falling"/>
Input is Interrupt for	<input type="text" value="pcf"/>
Polling Interval	<input type="text" value="5"/>
Bouncetime	<input type="text" value="180"/>
	<input type="button" value="Save"/> <input type="button" value="Cancel"/>

Konfiguration ohne Nutzung von Interrupts (Polling)

Bei dieser Konfiguration benötigt man keinen zusätzlichen GPIO-Pin des Raspberry. Die Stati der einzelnen Pins auf dem Modul werden zyklisch abgerufen. Es muss sichergestellt sein, dass Eingangsimpulse nicht zu kurz sind, damit kein Impuls verpasst wird.

- Inverted: true
- Resistor: Pull-Up
- Interrupt: None
- Polling Interval: 0.1

Add/Edit Digital Input: pcf8574

Module Name	<input type="text" value="pcf"/>
Name	<input type="text" value="pcf_1"/>
Pin	<input type="text" value="P0"/>
Payload for ON	<input type="text" value="ON"/>
Payload for OFF	<input type="text" value="OFF"/>
	<input checked="" type="checkbox"/> Inverted Logic Level
Enable Resistor	<input type="text" value="Pull-Up"/>
Interrupt	<input type="text" value="None"/>
Polling Interval	<input type="text" value="0.1"/>
Bouncetime	<input type="text" value="180"/>
	<input type="button" value="Save"/> <input type="button" value="Cancel"/>

Ausgänge

- initial: high
- inverted: true

Add/Edit Digital Output: pcf8575

Module Name	<input type="text" value="Module2"/>
Name	<input type="text" value="Enter an unique name here"/>
Pin	<input type="text" value="P0"/>
Payload for ON	<input type="text" value="ON"/>
Payload for OFF	<input type="text" value="OFF"/>
	<input checked="" type="checkbox"/> Inverted Logic Level
Initial State on Startup	<input type="text" value="HIGH"/>
Timed (in ms)	<input type="text" value="0"/>
<input type="button" value="Save"/> <input type="button" value="Cancel"/>	

From: <https://wiki.loxberry.de/> - **LoxBerry Wiki - BEYOND THE LIMITS**

Permanent link: https://wiki.loxberry.de/plugins/multi_io/gpio_module/pcf8575_expansion_module

Last update: **2023/07/29 12:00**