MCP23017 Expander Modul

Beschreibung

Der MCP23017 ist die Bezeichnung des Chipsatzes, so gut wie immer findet man aber unter dem Namen auch passend vorkonfektionierte Module, z. B. dieses hier von Adafruit:

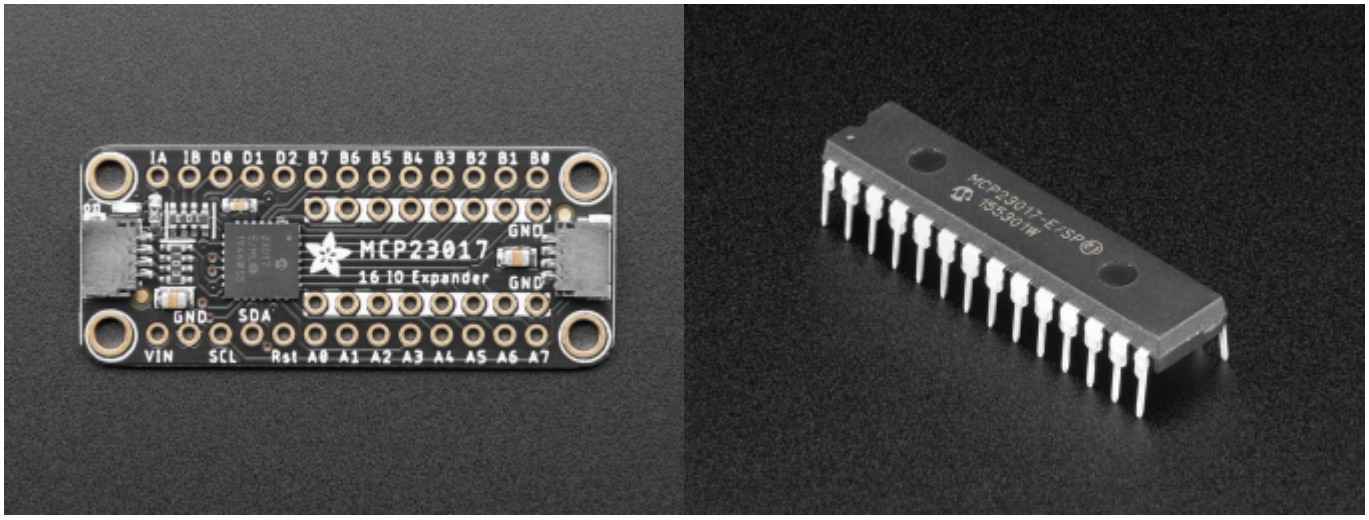


Abbildung: © https://learn.adafruit.com/adafruit-mcp23017-i2c-gpio-expander/overview

Ein Chip bzw. Modul kann 16 IOs bereitstellen. Je nach Konfiguration können diese als Ein- oder Ausgang dienen. Über 2 Interrupts (die jeweils an einen GPIO des Raspberry angeschlossen werden müssen), 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.

Datenblatt

MCP23017: http://ww1.microchip.com/downloads/en/DeviceDoc/20001952C.pdf

Weitere Informationen:
https://web.archive.org/web/20190826015652/http://www.netzmafia.de/skripten/hardware/RasPi/Projekt-I2C-Expander/index.html

Hardware-Konfiguration

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

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

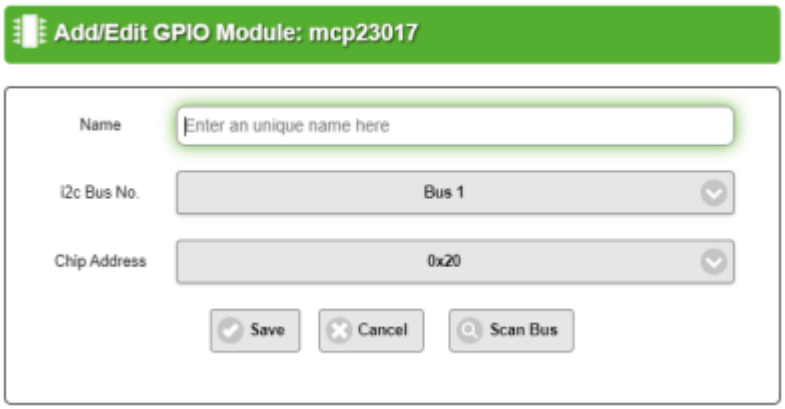
Welche Bus-Adresse das Modul verwendet, muss über 3 PINs auf dem Modul eingestellt werden. Möglich sind Adressen zwischen 0x20 und 0x27

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

Address (Hex) MCP23017	D2/A2	D1/A1	D0/A0
0x20	LOW	LOW	LOW
0x21	LOW	LOW	HIGH
0x22	LOW	HIGH	LOW
0x23	LOW	HIGH	HIGH
0x24	HIGH	LOW	LOW
0x25	HIGH	LOW	HIGH
0x26	HIGH	HIGH	LOW
0x27	HIGH	HIGH	HIGH

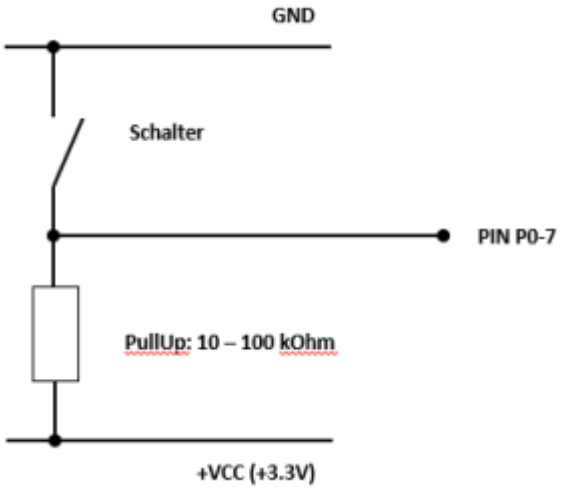
Software-Konfiguration

Modul



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.



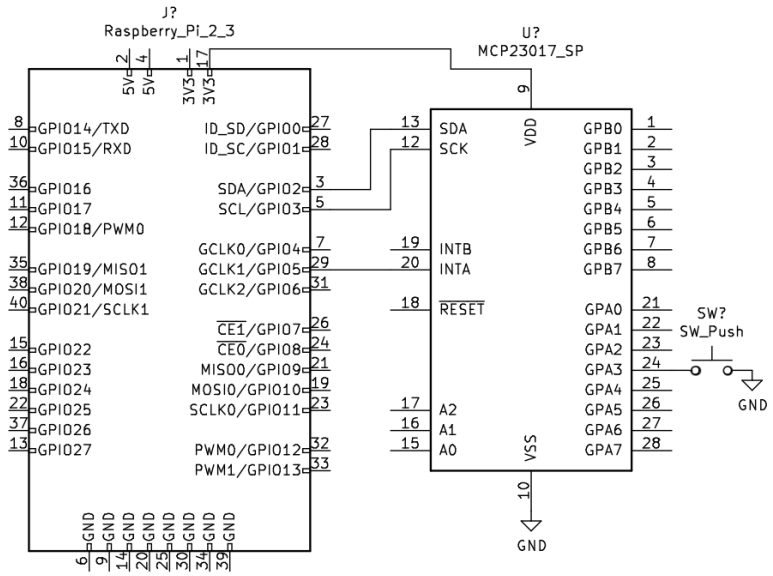
Verschaltung mit externem PullUp-Widerstand

Konfiguration mit Nutzung von Interrupts (empfohlen)

Standardmäßig werden die Zustände der Eingänge "gepollt", dass 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).

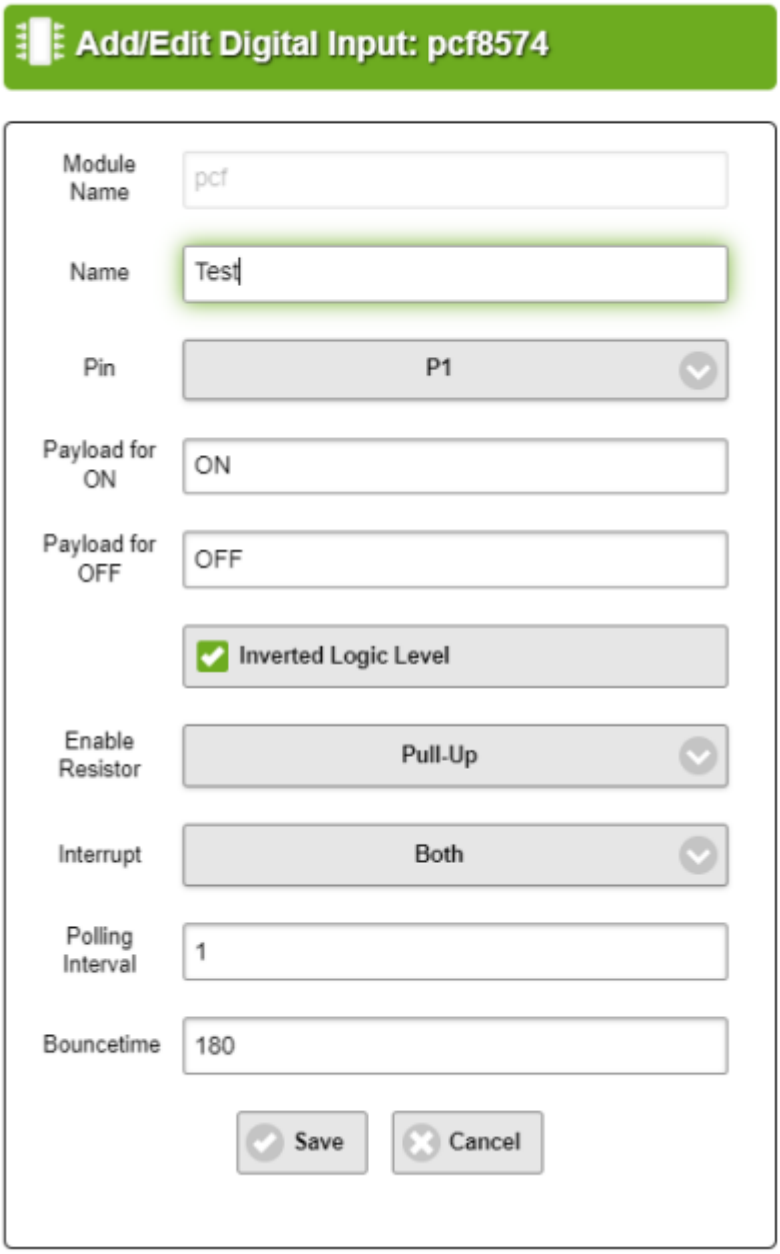
Die Module haben zu diesem Zweck einen INT-Pin, der auf LOW gesetzt wird, sobald sich der Zustand eines der PINs am Modul ändert. Der INT-Pin wird wieder auf HIGH gesetzt, sobald der Zustand aller Pins einmal abgefragt wurde. Mit Hilfe eines GPIO-Eingangs auf dem Raspberry kann man so eine Interrupt-gesteuerte 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, dass heisst das Betriebssystem bekommt eine Information, das sich er Status am GPIO geändert hat. Das Plugin erkennt das entsprechend und löst die Status-Abfrage des 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

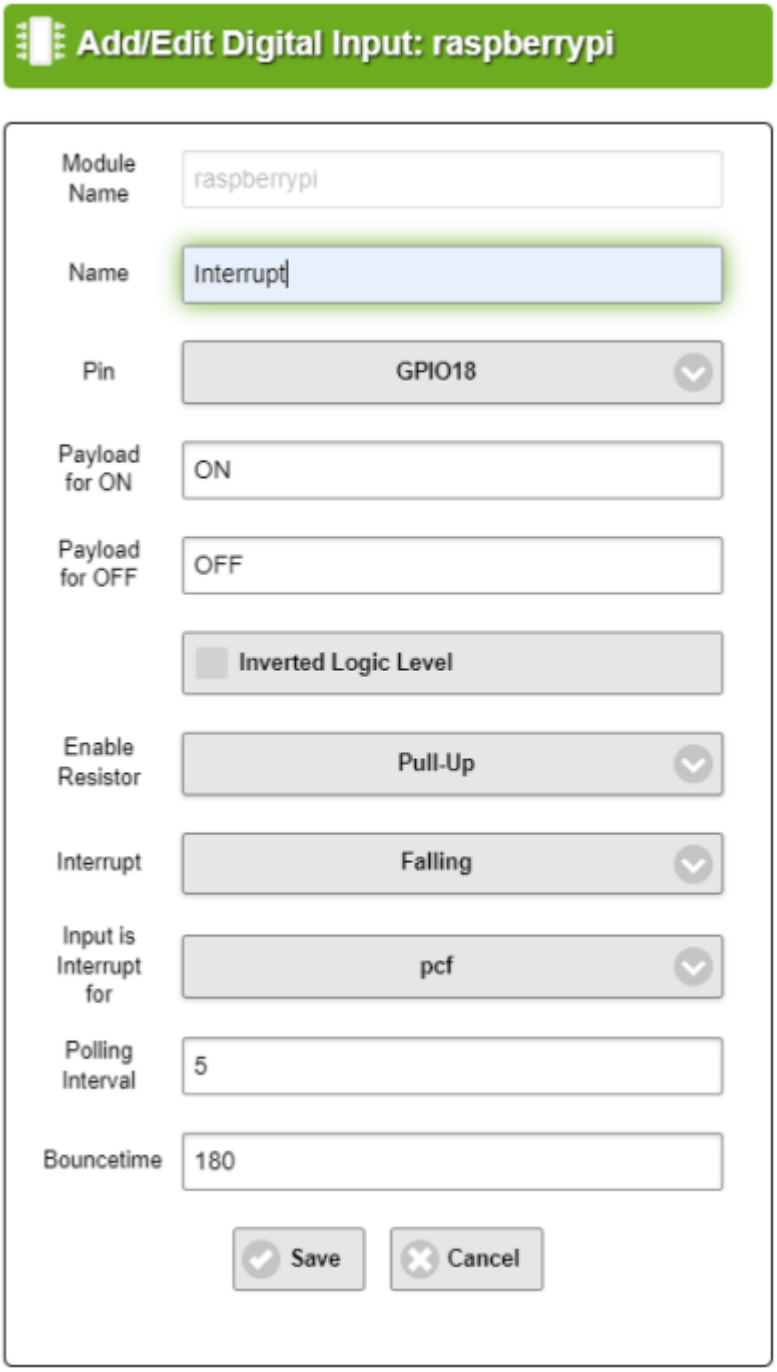
Zunächst konfiguriert man alle Eingänge am Modul:

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



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

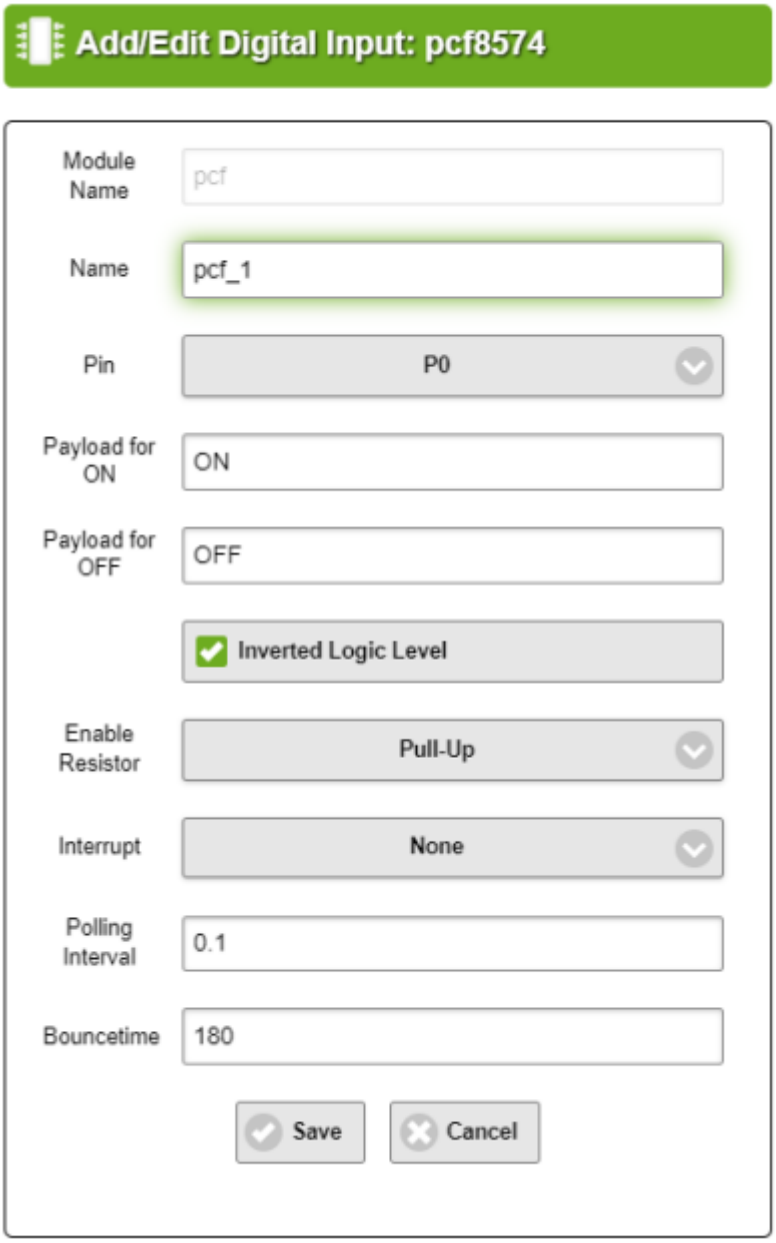
- Resistor: Pull-Up
- Interrupt: Falling
- Input is Interrupt for: <Modulname>



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



Ausgänge

Bitte ergänzen