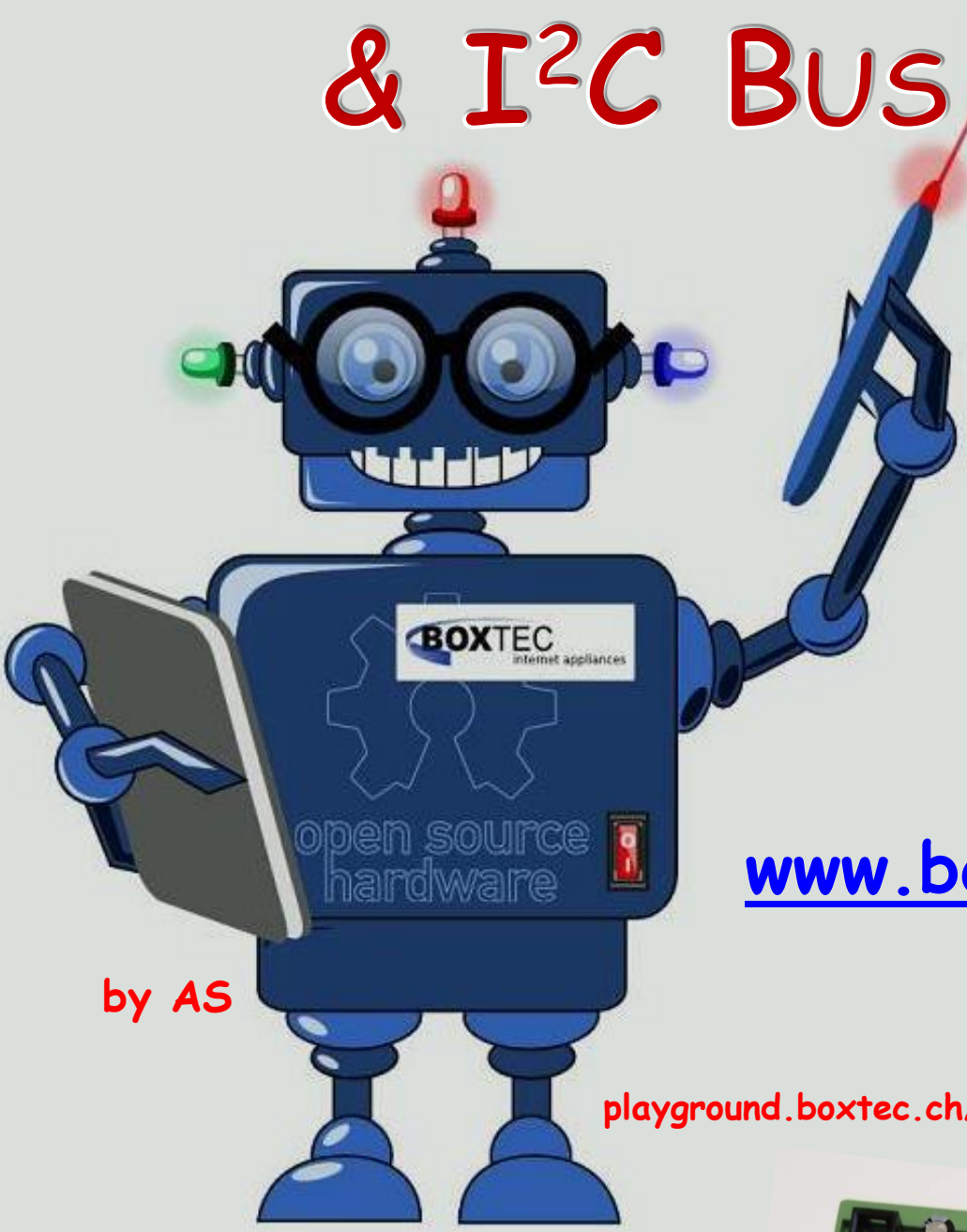
Mikrokontroller & I²C Bus

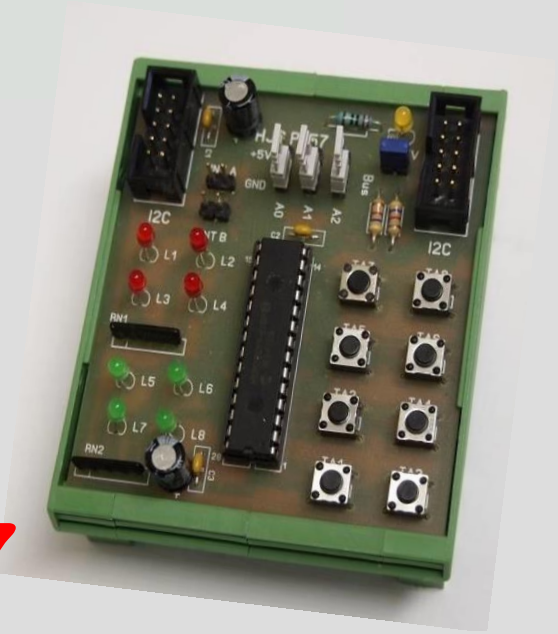


www.boxtec.ch

by AS

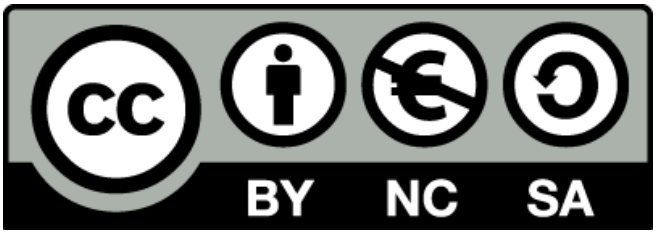
playground.boxtec.ch/doku.php/tutorial

I²C Bus und der MCP23017
Teil 2 - Software



I²C Bus und der MCP 23017

Copyright



Sicherheitshinweise

Lesen Sie diese Gebrauchsanleitung, bevor Sie diesen Bausatz in Betrieb nehmen und bewahren Sie diese an einem für alle Benutzer jederzeit zugänglichen Platz auf. Bei Schäden, die durch Nichtbeachtung dieser Bedienungsanleitung verursacht werden, erlischt die Gewährleistung / Garantie. Für Folgeschäden übernehmen wir keine Haftung! Bei allen Geräten, die zu ihrem Betrieb eine elektrische Spannung benötigen, müssen die gültigen VDE-Vorschriften beachtet werden. Besonders relevant sind für diesen Bausatz die VDE-Richtlinien VDE 0100, VDE 0550/0551, VDE 0700, VDE 0711 und VDE 0860. Bitte beachten Sie auch nachfolgende Sicherheitshinweise:

- Nehmen Sie diesen Bausatz nur dann in Betrieb, wenn er zuvor berührungssicher in ein Gehäuse eingebaut wurde. Erst danach darf dieser an eine Spannungsversorgung angeschlossen werden.
- Lassen Sie Geräte, die mit einer Versorgungsspannung größer als 24 V- betrieben werden, nur durch eine fachkundige Person anschließen.
- In Schulen, Ausbildungseinrichtungen, Hobby- und Selbsthilfewerkstätten ist das Betreiben dieser Baugruppe durch geschultes Personal verantwortlich zu überwachen.
- In einer Umgebung in der brennbare Gase, Dämpfe oder Stäube vorhanden sind oder vorhanden sein können, darf diese Baugruppe nicht betrieben werden.
- Im Falle eine Reparatur dieser Baugruppe, dürfen nur Original-Ersatzteile verwendet werden! Die Verwendung abweichender Ersatzteile kann zu ernsthaften Sach- und Personenschäden führen. Eine Reparatur des Gerätes darf nur von fachkundigen Personen durchgeführt werden.
- Spannungsführende Teile an dieser Baugruppe dürfen nur dann berührt werden (gilt auch für Werkzeuge, Messinstrumente o.ä.), wenn sichergestellt ist, dass die Baugruppe von der Versorgungsspannung getrennt wurde und elektrische Ladungen, die in den in der Baugruppe befindlichen Bauteilen gespeichert sind, vorher entladen wurden.
- Sind Messungen bei geöffnetem Gehäuse unumgänglich, muss ein Trenntrafo zur Spannungsversorgung verwendet werden
- Spannungsführende Kabel oder Leitungen, mit denen die Baugruppe verbunden ist, müssen immer auf Isolationsfehler oder Bruchstellen kontrolliert werden. Bei einem Fehler muss das Gerät unverzüglich ausser Betrieb genommen werden, bis die defekte Leitung ausgewechselt worden ist.
- Es ist auf die genaue Einhaltung der genannten Kenndaten der Baugruppe und der in der Baugruppe verwendeten Bauteile zu achten. Gehen diese aus der beiliegenden Beschreibung nicht hervor, so ist eine fachkundige Person hinzuzuziehen

Bestimmungsgemäße Verwendung

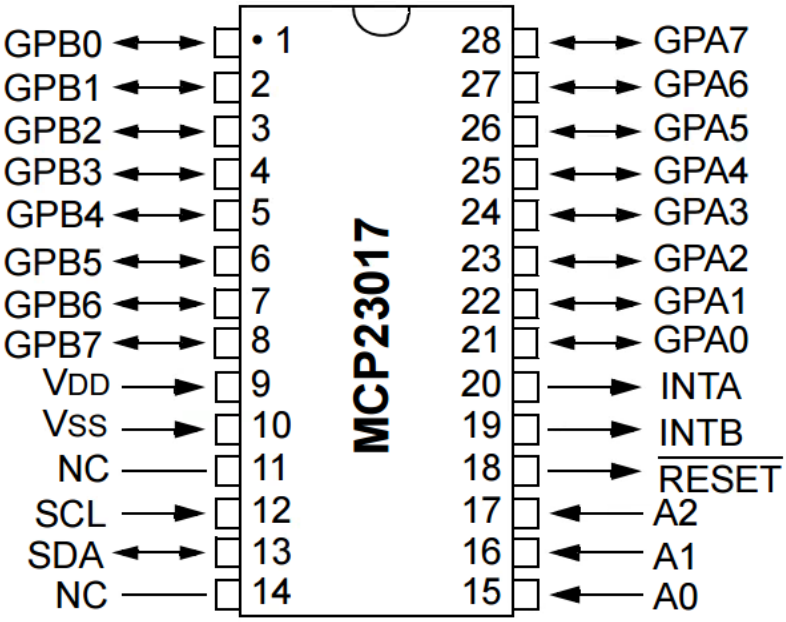
- Auf keinen Fall darf 230 V~ Netzspannung angeschlossen werden. Es besteht dann Lebensgefahr!
- Dieser Bausatz ist nur zum Einsatz unter Lern- und Laborbedingungen konzipiert worden. Er ist nicht geeignet, reale Steuerungsaufgaben jeglicher Art zu übernehmen. Ein anderer Einsatz als angegeben ist nicht zulässig!
- Der Bausatz ist nur für den Gebrauch in trockenen und sauberen Räumen bestimmt.
- Wird dieser Bausatz nicht bestimmungsgemäß eingesetzt kann er beschädigt werden, was mit Gefahren, wie z.B. Kurzschluss, Brand, elektrischer Schlag etc. verbunden ist. Der Bausatz darf nicht geändert bzw. umgebaut werden!
- Für alle Personen- und Sachschäden, die aus nicht bestimmungsgemäßer Verwendung entstehen, ist nicht der Hersteller, sondern der Betreiber verantwortlich. Bitte beachten Sie, dass Bedien- und /oder Anschlussfehler außerhalb unseres Einflussbereiches liegen. Verständlicherweise können wir für Schäden, die daraus entstehen, keinerlei Haftung übernehmen.
- Der Autor dieses Tutorials übernimmt keine Haftung für Schäden. Die Nutzung der Hard- und Software erfolgt auf eigenes Risiko.

Der I2C Bus und der MCP 23017 – Teil 2

Der MCP23017 hat Aufgrund seiner Register einige gewöhnungsbedürftige Einstellungen gegenüber seinen Vorgängern.

MCP 23017 auf der Platine 157 mit 2 Anschlüssen für den I²C Bus, 8 LEDs zur Anzeige am Port A, 8 Taster zur Eingabe am Port B, 3 Stecker zur Auswahl der Adresse, 2 Stecker zur Schaltung der Interrups, Stecker zum Anlegen der Vcc an den Bus und der Anzeige der Betriebs-spannung

Ansicht des ICs von oben mit Pinbelegung



Die einzelnen Pins die wir benutzen:

Port A – GPA0 bis GPA7 – Pin 21 bis 28
Port B – GPB0 bis GPB7 – Pin 1 bis 8

Bus SCL – Pin 12
Bus SDA – Pin 13

Angabe Adresse A0 / A1 / A2 – PIN 15 / 16 / 17

Betriebsspannung Vcc / GND – Pin 9 / 10

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

Angabe der Register mit Namen, Beschreibung, Bank 0, Bank 1 und Werten

Angabe der Register mit der entsprechenden Werten im Programm:

```
#define MCP23017_IODIRA   0x00          // Steuert die Richtung der Daten am Port A, Pin
                                           gesetzt=Eingang, nicht gesetzt=Ausgang
#define MCP23017_IODIRB   0x01          // Steuert die Richtung der Daten am Port B, Pin
                                           gesetzt=Eingang, nicht gesetzt=Ausgang
#define MCP23017_IPOLA    0x02          // konfiguriert die Polarität GPIO Port Bits
#define MCP23017_IPOLB    0x03
#define MCP23017_GPINTENA 0x04          // steuert die Interrupt-On-Change
#define MCP23017_GPINTENB 0x05
#define MCP23017_DEFVALA  0x06          // Standardvorgabe für Eingangspins
#define MCP23017_DEFVALB  0x07
#define MCP23017_INTCONA  0x08          // legt fest wie der zugehörige Pin-Wert für die
                                           Interrupt-On-Change-Funktion verglichen wird
#define MCP23017_INTCONB  0x09
#define MCP23017_IOCON    0x0A          // enthält etliche Bits für die Konfiguration
#define MCP23017_GPPUA    0x0C          // steuert die Pull-UP-Widerstände für Port-Pins
#define MCP23017_GPPUB    0x0D
#define MCP23017_INTFA    0x0E          // spiegelt die Interrupt-Bedingungen auf den Port-
                                            Pins wieder
#define MCP23017_INTFB    0x0F
#define MCP23017_INTCAPA  0x10          // erfassen den GPIO-Port-Wert zum Zeitpunkt des
                                           Interrups
#define MCP23017_INTCAPB  0x11
#define MCP23017_GPIOA    0x12          // geben den Wert der Portleitungen zurück. Ein
                                           lesen entspricht dem lesen der Portleitungen
#define MCP23017_GPIOB    0x13          // ein schreiben entspricht dem schreiben in die
                                           Ausgangslatches
#define MCP23017_OLATA    0x14          // erlaubt den Zugriff auf die als Ausgang
                                           konfigurierten Pins Register A
#define MCP23017_OLATB    0x15          // erlaubt den Zugriff auf die als Ausgang
                                           konfigurierten Pins Register B
```

Im oberen Teil habe ich den Inhalt der Datei **MCP23017.h** dargestellt. Ein Teil der Kommentare habe ich entfernt. Wer mehr wissen möchte, bitte die Datei anschauen.
Mit der Angabe der Register und der Bank 0 und 1 können wird die Werte zuordnen. In unserem Hauptprogramm benutzen wird dann nur noch die Namen der Register für eine bessere Übersicht.
Die Angabe der Frequenz unseres Prozessors und der **I²C** Bus Adresse erfolgt in der Datei **main.h**. Für die Kommunikation des **I²C** Busses verwende ich die Dateien von Peter Fleury.

Zum Start des Hauptprogrammes verwende ich diese Dateien:

```
#include <stdbool.h>
#include <avr/pgmspace.h>
#include "main.h"
#include "i2cmaster.h"
#include "avr/io.h"
#include "util/delay.h"
#include "avr/interrupt.h"
#include "stdlib.h"
#include "MCP23017.h"
```

Ob sie alle benötigt werden habe ich nicht getestet.
In diesem Teil erfolgt die Einstellung der Register:

```
//setzt Port A als Ausgang mit IODIRA
i2c_start(MCP23017_ADDR);               // Angabe Adresse
i2c_write(MCP23017_IODIRA);             // Steuert die Richtung der Daten am Port
i2c_write(0x00);                        // Angabe Ausgang  0x00-alle, 0xff-alle aus
i2c_stop();

// setzt Port B alle R mit GPPUB als Eingang
i2c_start(MCP23017_ADDR);               // Angabe Adresse
i2c_write(MCP23017_GPPUB);              // aktivierung der internen Pullups Widerstände
i2c_write(0xff);                        // schalte alle Eingänge auf 5V
i2c_stop();

// Alle Eingänge Port B invertieren
i2c_start(MCP23017_ADDR);               // Angabe Adresse
i2c_write(MCP23017_IPOLB);              // Taster schalten Masse (active-low)
i2c_write(0xff);                        // alle Eingänge invertieren
i2c_stop();
```

In diesem Teil werden die entsprechenden LEDs ein- oder ausgeschaltet mit den Pausen:
(Für das Programm „Blinken")

```
// schreiben der Register für LEDs
i2c_start(MCP23017_ADDR);         // Angabe Adresse
i2c_write(MCP23017_OLATA);        // schalte Ausgänge
i2c_write(0x04);                  // Auswahl der LED
i2c_stop();                       // zweite Zahl - LED 1-4

_delay_ms(1500);                  // Pause

i2c_start(MCP23017_ADDR);         // Angabe Adresse
i2c_write(MCP23017_OLATA);        // schalte Ausgänge
i2c_write(0x40);                  // Auswahl der LED
i2c_stop();                       // erste Zahl - LED 5-8

_delay_ms(1500);                  // Pause
```

Es wird einfach mit `i2c_write(0x04);` und `i2c_write(0x40);` 2 LEDs umgeschaltet. Vorher wird die Adresse aufgerufen und im Ausgangsregister A die gewählten Ausgänge eingeschaltet.

```
i2c_start(MCP23017_ADDR);         // Angabe Adresse
i2c_write(MCP23017_OLATA);        // schalte Ausgänge
i2c_write(0x40);
```

Im zweiten Programm möchte ich euch zeigen wie „einfach" (wenn man weiss wie) es ist einen Taster auszulesen und eine LED zu schalten.
Dazu verwende ich wieder den Teil aus dem Start des Programmes und die Einstellung der Register. Als erstes dazu erfolgt das durch `i2c_write(MCP23017_GPPUB);` das schalten der der Pull-Up-Widerstände auf Vcc (+5V). Danach erfolgt durch `i2c_write(MCP23017_GPIOB);` ein lesen des Register. Das Ergebnis wird in `Data = i2c_readNak();` gespeichert.

```
// Abfrage der Eingänge am MCP
i2c_start(MCP23017_ADDR);               // Angabe Adresse
i2c_write(MCP23017_GPIOB);
i2c_stop();

i2c_start(MCP23017_ADDR +1);            // Auslesen Adresse + 1
Data = i2c_readNak();                   // liest alle Eingänge ein
i2c_stop();

// Abfrage welcher Taster
if (!(Data & 0x05))                     // Abfrage DATA, welcher Taster
  {                                     // Angabe Taster mit z.B. mit 0x05
    i2c_start(MCP23017_ADDR);           // Angabe Adresse
    i2c_write(MCP23017_OLATA);          // Schalte Ausgang
    i2c_write(0x20);                    // Auswahl der LED
    i2c_stop();
  }
else
  {
    i2c_start(MCP23017_ADDR);           // Angabe Adresse
    i2c_write(MCP23017_OLATA);          // Schalte Ausgang
    i2c_write(0x02);                    // Auswahl der LED
    i2c_stop();
  }
```

Danach erfolgt mit `if (!(Data & 0x05))` die Abfrage welcher Taster gedrückt wurde. Das schalten der LED habe ich bereits oben beschrieben.
Es ist ebenfalls ein gemischter Betrieb möglich.

Ansicht des fertigen Modules

Einige Teile des Textes wurden zur besseren Übersicht farblich gestaltet.
Die Nutzung erfolgt auf eigenes Risiko.
Ich wünsche viel Spaß beim Bauen und programmieren
Achim

myroboter@web.de