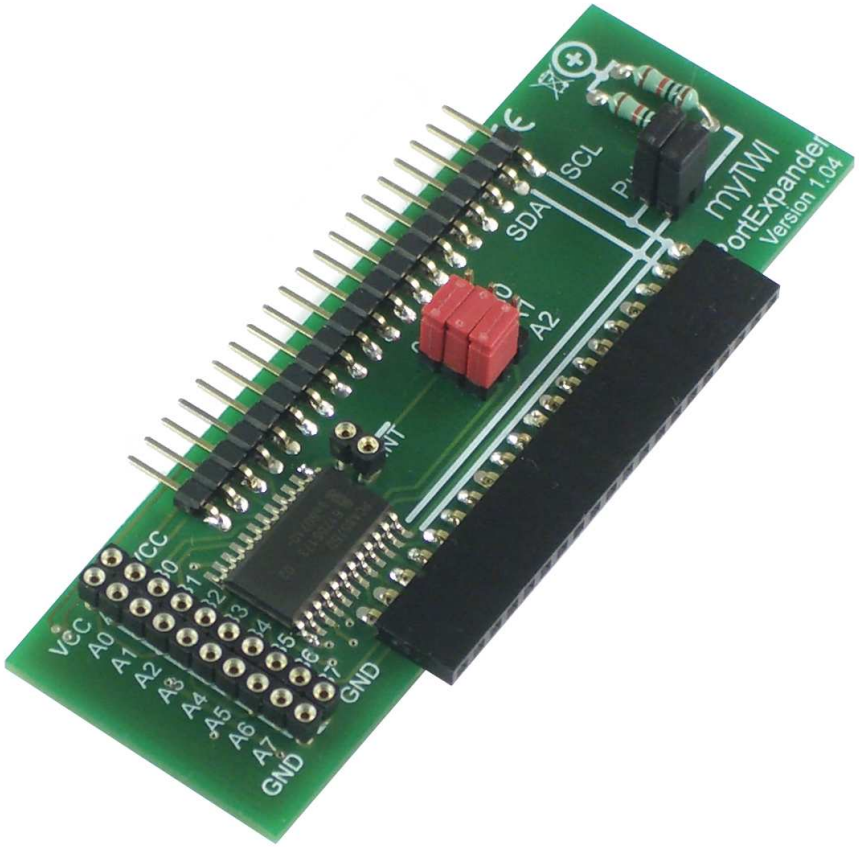


myTWI PortExpander
Version 1.04

Inhalt

Allgemeine Beschreibung ... 3
Eigenschaften ... 3
Prinzipschaltplan ... 3
Technische Daten ... 4
Betriebsdaten ... 4
Schnittstellendaten ... 4
Mechanische Daten ... 4
Pinbelegung ... 4
Schaltplan ... 5
Bestückungsplan ... 5
Bestücktes Board ... 5
Adresskonfiguration ... 6
Anwendungsscript mySmartUSB Terminal ... 7
Programmbeispiel LED-Ansteuerung ... 10
Versuchsaufbau mit Laborkarte ... 14
Allgemeine Sicherheitshinweise ... 14

Contents

General description ... 3
Properties ... 3
Principle circuit diagram ... 3
Technical Data ... 4
Operating Data, ... 4
Interface Data ... 4
Mechanical Data ... 4
Pin configuration ... 4
circuit diagram ... 5
layout diagram ... 5
board, equipped ... 5
Configuration of the address ... 6
Application script mySmartUSB Terminal ... 7
Programm example LED-control ... 10
Breadboard construction Prototyping board ... 14
Safety Guidelines ... 14

Die Informationen in diesem Produkt werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht.
Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt.
Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen.
Trotzdem können Fehler nicht vollständig ausgeschlossen werden.
Die Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.
Für Verbesserungsvorschläge und Hinweise auf Fehler sind die Autoren dankbar.

Fast alle Hardware- und Softwarebezeichnungen, die in diesem Dokument erwähnt werden, sind gleichzeitig auch eingetragene Warenzeichen und sollten als solche betrachtet werden.

© Laser & Co. Solutions GmbH
Promenadenring 8
02708 Löbau
Deutschland

www.myAVR.de
support@myavr.de

Tel: ++49 (0) 358 470 222
Fax: ++49 (0) 358 470 233

In spite of the great care taken while writing this document the author is not responsible for the topicality, correctness, completeness or quality of the information provided. Liability claims regarding damage caused by the use of any information provided, including any kind of information which is incomplete or incorrect,will therefore be rejected.

© Laser & Co. Solutions GmbH
Promenadenring 8
02708 Löbau
Germany

www.myAVR.com
support@myavr.com

Tel: ++49 (0) 358 470 222
Fax: ++49 (0) 358 470 233

Allgemeine Beschreibung

Die Zusatzplatine (Add-On) „myTWI PortExpander“ ist ein Teil der TWI-Serie der myAVR-Produktfamilie. Damit wird das myAVR-System um eine komfortable Lösung für die Ansteuerung weiterer I/O Ports, beispielsweise für die Verwendung zusätzlicher Peripheriegeräte, erweitert.

Es kann mit weiteren TWI (I²C) Add-Ons am myAVR Erweiterungsport angeschlossen werden.

Eigenschaften

- Universelle TWI-Porterweiterung mit 2 zusätzlichen I/O Ports für insgesamt 16 Pins
- Steckerleiste für den Anschluss an die myAVR Boards MK1, MK2, MK3 sowie mySmartControl
- Buchsenleiste für den Anschluss weiterer Module
- Robust, mit Dokumentationsdruck
- Industriefertigung
- Material: FR4, 1,5 mm; 0.35 µm Cu
- Gebohrt, verzinnt, Lötstoppmaske

General description

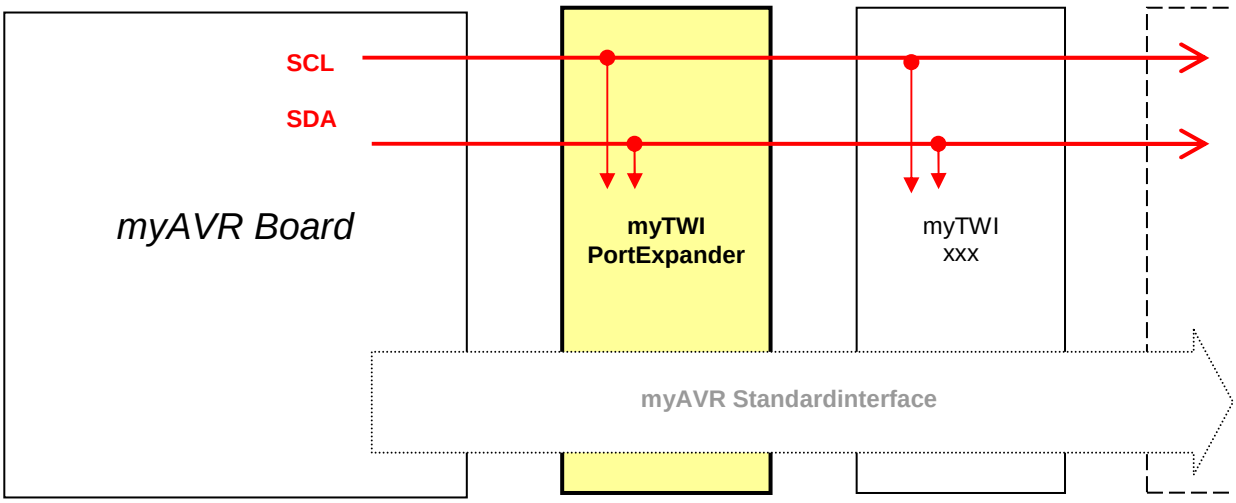
The additional board (add-on) “myTWI Port Expander” is a part of the TWI-production run, the myAVR-product line. With that it’s a comfortable solution to control more I/O ports, e.g. to use more peripheral equipment.

Its possible to use with one more TWI (I²C) add-on at the myAVR extension port.

Properties

- Universal TWI-extension port with 2 more I/O ports and a total of 16 pins
- Pin header for the access to the myAVR Board MK1, MK2, MK3 and also mySmartControl
- Female connector for more add-ons
- Solid, with documentation print
- Industrial prdoduction
- Material: FR4, 1,5; 0,35µm CU
- pre-drilled, tin-plated, solder resist mask

Prinzipschaltplan Principle circuit diagram



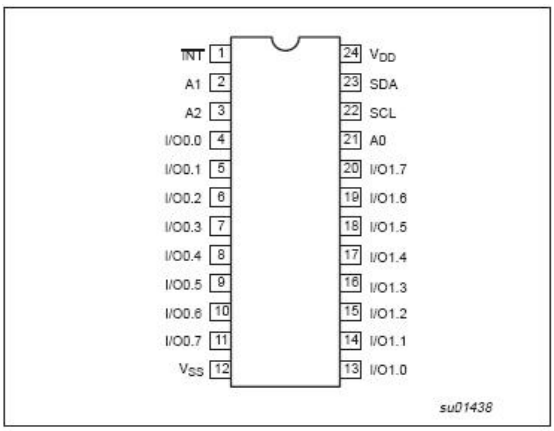
Technische Daten	
Betriebsdaten	
Versorgungsspannung	5V
Betriebsstrom	200mA
Betriebsspannung	2,3V - 5,5 V
Betriebstemperatur	-40 ℃ – 85 ℃
Lagertemperatur	-65 ℃ – 150 ℃
Schnittstellendaten	
Adresse	0b0100aaax a=Adressjumper x=read/write
Kommunikation	TWI (I²C)

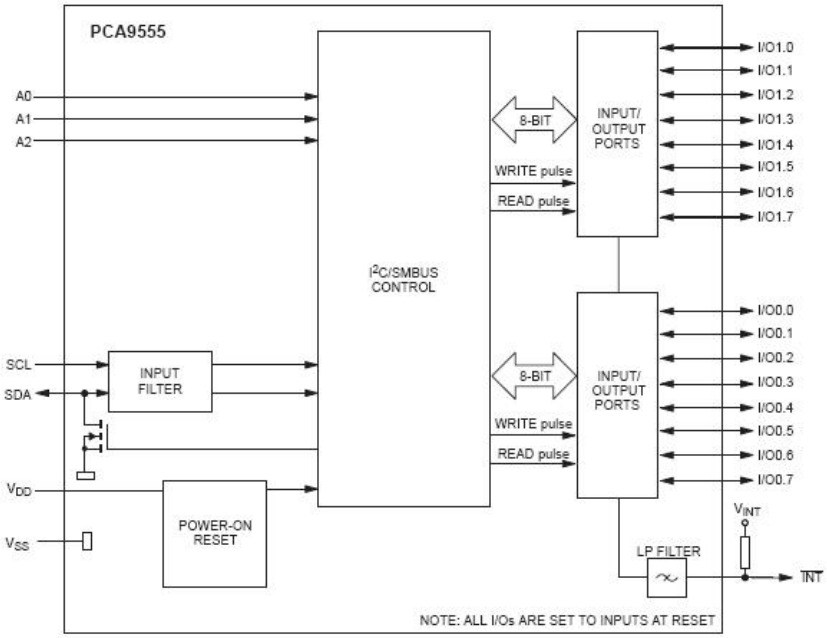
Technical Data	
Operating Data,	
Supply Voltage	5V
Operating Current	200mA
Operating Voltage	2,3V – 5,5V
Operating Temperature	-40 ℃ – 85 ℃
Storage Temperature	-65 ℃ – 150 ℃
Interface Data	
Address	0b0100aaax a=address jumper x=read/write
Comunication	TWI (I²C)

Mechanische Daten	
Abmessungen Platine (L x B x H) in mm	90 x 30 x 1,5
Masse in g	15
Rastermaß in mm	2,54
Leiterplattenmaterial:	FR4; 0,35 µm Cu

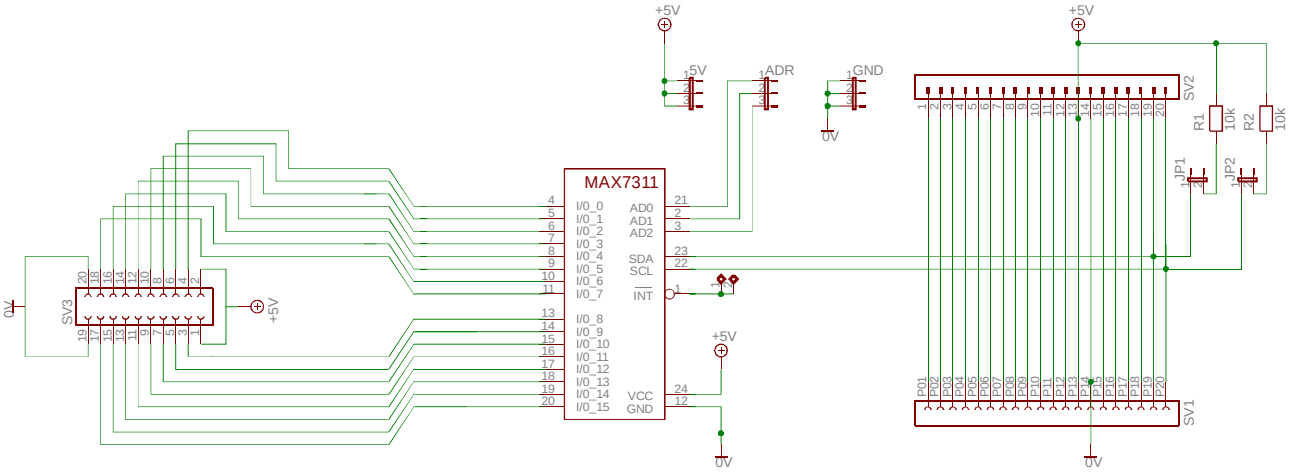
Mechanical Data	
Dimensions of the board (L x B x H) in mm	90 x 30 x 1,5
Weight in g	15
Grid dimensions in mm	2,54
PCB material	FR4; 0,35µm Cu

Pinbelegung Pin configuration

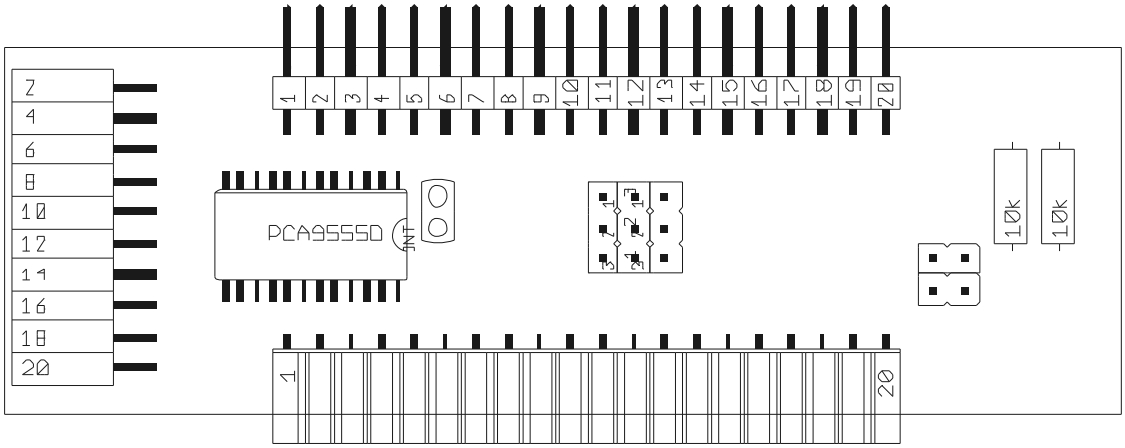




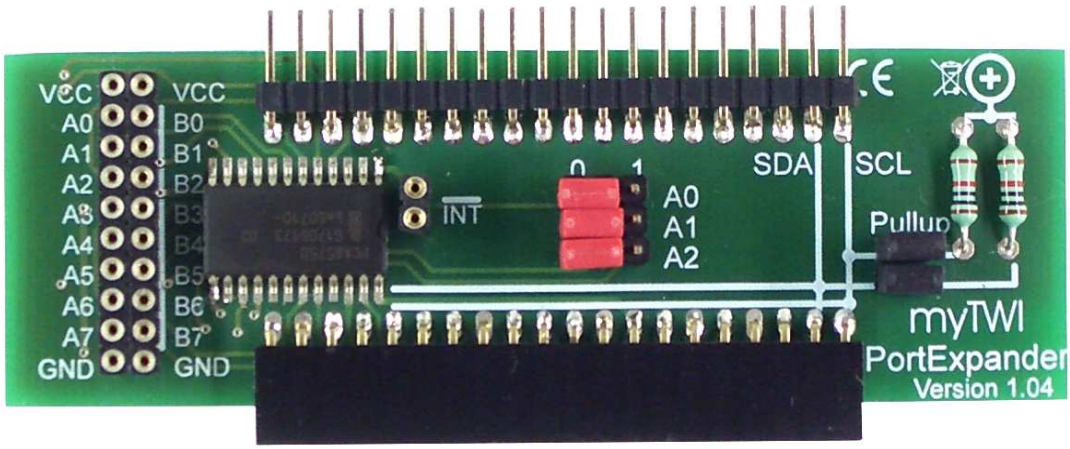
Schaltplan circuit diagram



Bestückungsplan layout diagram



Bestücktes Board board, equipped

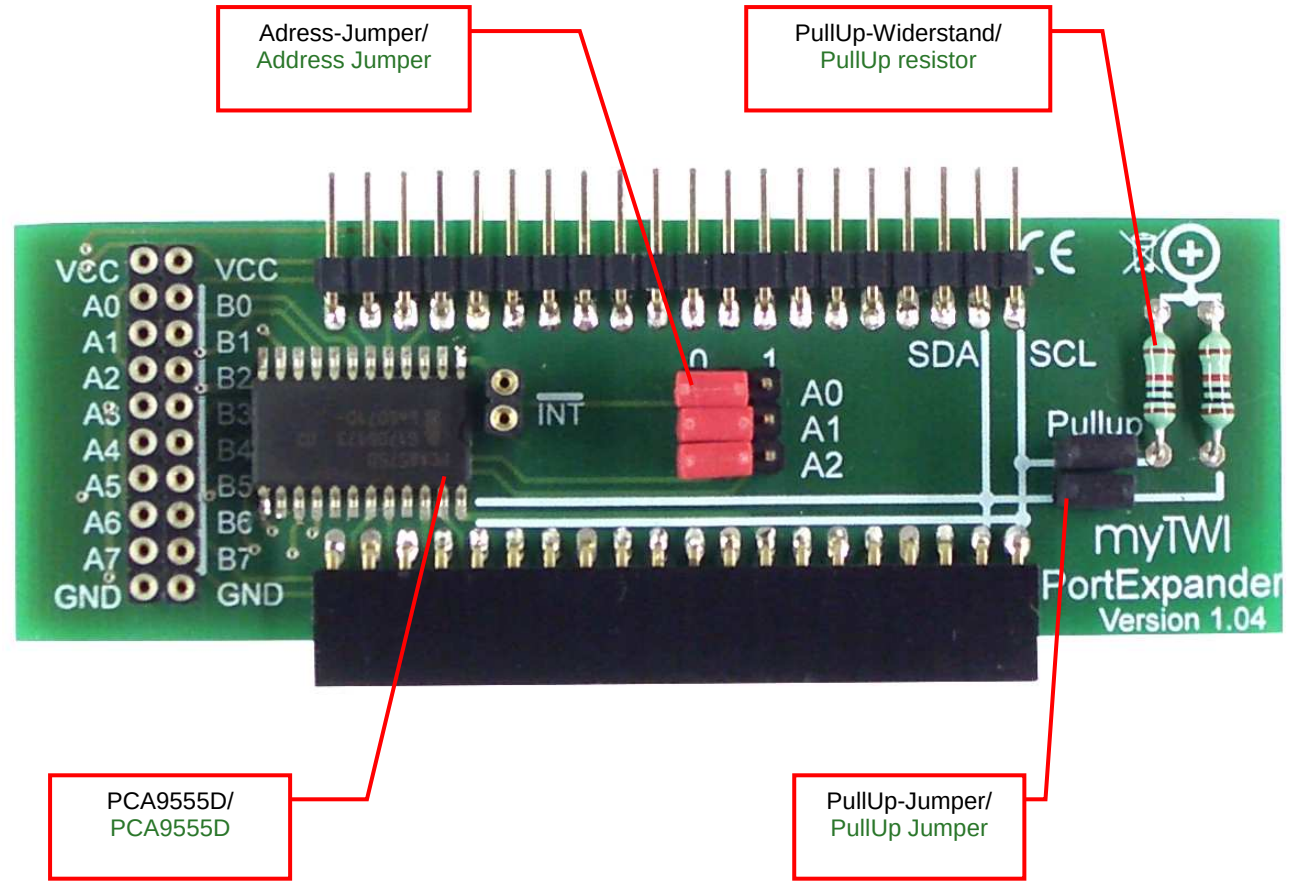


Adresskonfiguration

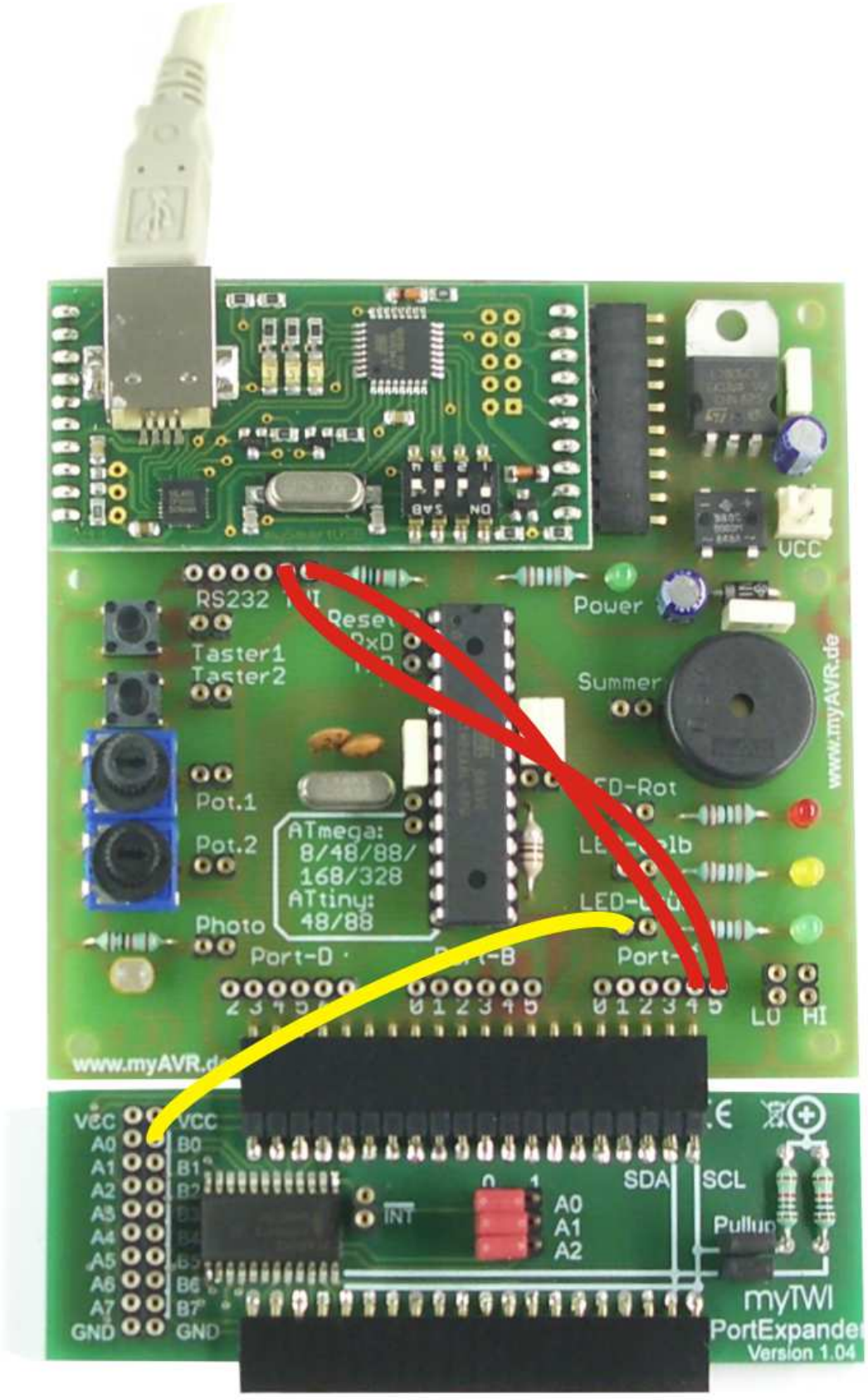
Ein myAVR TWI Add-On kann mit weiteren TWI Add-Ons in einem BUS betrieben werden. Ein TWI Gerät bildet aus seiner Geräte-ID und den möglichen Adresspins (A0-A2) seine Geräteadresse im Bus. Somit lassen sich auch mehrere gleiche Geräte in einem BUS betreiben. Auf jedem myTWI Add-On sind die Adresspins per Jumper konfigurierbar. Des weiteren muss der TWI Bus mit PullUp-Widerständen auf High gezogen werden. Dies sollte jeweils nur von einem Add-On erfolgen. Dazu verfügt jedes Add-On über entsprechende PullUp-Widerstände und Jumper, um diese zu aktivieren.

Configuration of the address

A myAVR TWI add-on can be used in a BUS System with other TWI add ons. One TWI add on create with the add on ID and the possibility adress pins (A0-A2) his adress in the BUS system. So, its possible to use several add ons. You can switch the address pin on every myTWI add on with jumper. Furthermore the TWI BUS must be changed to high with the help of the PullUp resistance. But this should be carried out only from one add on. Every add on dispose of the PullUp resistance and jumper to activate this.



Anwendungsscript mySmartUSB Terminal **Application script mySmartUSB Terminal**

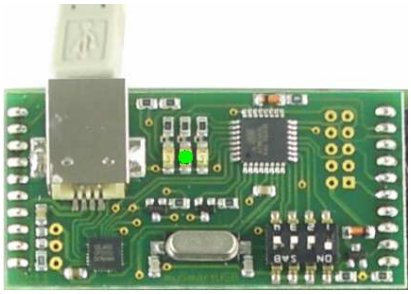


Mithilfe dieses Programmbeispiels lassen sich die einzelnen I/O Ports des myTWI PortExpanders als Ein- und Ausgänge schalten. Die korrekte Umsetzung wird durch das Leuchten einer zusätzlich angeschlossenen LED visualisiert. Zur Realisierung wird das Programm „mySmartUSB Terminal“ benötigt, dieses können Sie in der Downloadsektion von unserer Firmenwebseite herunterladen.

1. Stecken Sie das myTWI Add-On an das myAVR Board an. Achten Sie bitte darauf, das die Adresspins in der Grundstellung sind, das heißt A2, A1, A0 auf 0.
2. Verbinden Sie Port TWI-SCL mit Port C.5 sowie Port TWI-SDA mit Port C.4 mit jeweils einem Patchkabel (diese müssen sich überkreuzen). Weiterhin ist ein Anschluss einer LED vom myAVR Board mit Port B.0 des Portexpanders zur Visualisierung notwendig.
3. Verbinden Sie nun das Board mit dem PC.
4. Starten Sie das Programm „mySmartUSB-Terminal“ (Hinweis, die neuste Version wird empfohlen).
5. unter der Registerkarte „Start“ wählen Sie bitte die Schaltfläche „mySmartUSB suchen“ (bei erfolgreicher Suche erscheint rechts daneben die USB-Version in grüner Schrift).
6. Bei erfolgreicher Erkennung wechseln Sie bitte zur Registerkarte „Steuerung“ und wählen den „myMode“ aus. Dies erkennen Sie am Leuchten der mittleren grünen LED auf dem mySmartUSB MK2.

With this program example it’s possible to change the I/O ports of the myTWI PortExpander between combined in- and outputs. You see the correct conversion with the help of a additional affiliated LED, which is shining. For the realization the programm “mySmartUSB Terminal” is needed, which you can download from our company web page.

1. Connect the TWI add on at a myAVR Board. Please attend that the adress pins are in the starting position. (A2; A1; A0 set 0)
2. Connect the ports TWI-SCL and port C.5 as well as the port TWI-SDA and port C 4 with a patch cable (these must cross themselves over). Anymore a LED from the myAVR Board musst be conncet with port B 0 from the myTWI Portexpander to visualize.
3. Put the board on your PC.
4. Start the programm “mySmartUSB Terminal” (notice; newest version is recommended)
5. In the start tab choose the button “mySmartUSB suchen” (if the search are successful you see right aside this usb version in a green font).
6. After the successful search change to the “Steuerung” tab a chosse “myMode”. On your myAVR Board MK2 the centric LED must be shining.



7. Nun wechseln Sie zur Registerkarte „myMode“ und übertragen den auf der nächsten Seite aufgeführten Quelltext in das Ausführfenster hinein. Gehen Sie danach auf „Senden“
8. Bei erfolgreicher Ausführung leuchtet die angeschlossene LED und ist somit auf Ausgang geschalten.

7. Now, you open the tab “myMode” and copy the following source code into the execution window. After this click on the “Senden” button .
8. At a successful execution the attached LED is shining and the output is connected.

Dies ergibt folgende Scriptsequenz in der myMode Konsole:

This shows the following sequence in the myMode console:

```
m:main

mode mh
pwr 0
rst 1
pwr 1
m:twi
ima
sta
sla 0x40 w
wr 6
wr 0x00 0x00
sta
sla 0x40 w
wr 2
wr 0x00 0x01
sto
end
```

Den Controller auf dem myAVR Board deaktivieren durch Schaltung der Resetleitung nach Restart	Diseable the controller on the myAVR Board by Wiring of the reset track after restart
In das TWI (I²C) Menü wechseln	Changed into the TWI (I²C) menue.
mySmartUSB als Master initialisieren	Initialise the mySmartUSB as Master
TWI START	TWI start
TWI WRITE Adresse des PCA9555D	TWI WRITE address of the PC9555D
TWI WRITE Kommandoregister 6=CONFIG	TWI WRITE command register 6=CONFIG
TWI WRITE I/O-Konfiguration Port A+B	TWI WRITE I/O configuration port A+B
TWI START	TWI START
TWI WRITE Adresse des PCA9555D	TWI WRITE address of the PCA9555D
TWI WRITE Kommandoregister 2=OUTPUT	TWI WRITE command register 2=OUTPUT
TWI WRITE Wertzuweisung Port A+B	TWI WRITE value assignment port A+B
TWI STOP	TWI STOP
END	END

Programmbeispiel LED-Ansteuerung

Zielstellung dieses Anwendungsprogramms ist eine einfache Tasterabfrage, sowie eine daraus resultierende LED-Ansteuerung mit dem myAVR Workpad PLUS zu realisieren.

Notwendige Einstellungen:

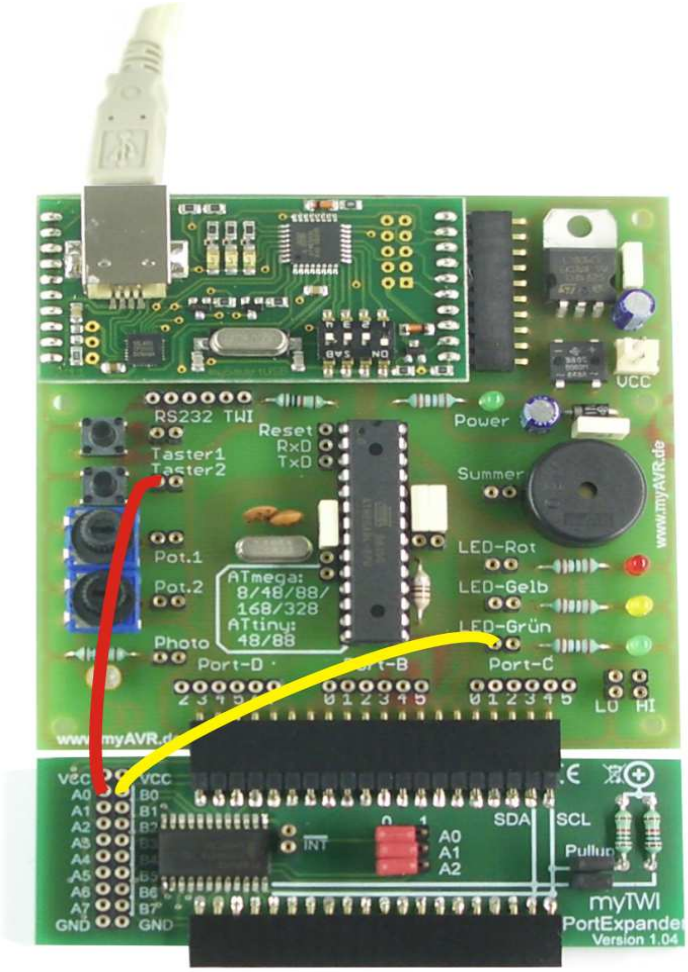
1. Stecken Sie das myTWI Add-On an das myAVR Board MK2 USB an. Achten Sie bitte darauf, dass die Adresspins in der Grundstellung sind (A2, A1, A0 auf 0).
2. Verbinden Sie einen Taster des myAVR Boards mit dem Port A.0 des PortExpanders und eine LED mit dem Port B.0 mithilfe eines Patchkabels.
3. Starten Sie das myAVR Workpad.
4. Klicken Sie im Begrüßungsmenü auf „neue Datei“ und wählen bitte „Grundgerüst“ unter der Programmsprache C/C++ aus.
5. Führen Sie nun die Hardwareerkennung unter der Registerkarte „Einstellung“ durch und speichern Sie diese.
6. Wechseln Sie nun zurück in die Entwicklungsumgebung und wählen unter „Vorlagen“: *9.TWI > twiInitMaster aus.*

Programm example LED-control

The goal of this programm is a simple test query, as well as realizing a LED drive resulting from it with the myAVR Workpad PLUS.

Necessary configurations

1. Put the myTWI add on into the myAVR Board MK2 USB. Please attend that the adress pins are in the starting position(A2; A1; A0 auf 0)
2. Connect a button from the myAVR Board with the port A.0 of the PortExpander and a LED with the port B.0 with a patch cable.
3. Start the myAVR Workpad
4. Click in the salutatory menu on “neue Datei” and choose the “Grundgerüst” in the programm language C/C++
5. proceed the hardware detection in the menu “Einstellungen and safe it.
6. Return to your development environment and chosse below “Vorlagen”: *9 TWI twiInitMaster*



Das Beispielscript umfasst folgende Komponente:

- Die Initialisierung des TWI-Master
- Auslösen der Start-Sequenz
- Die Adressierung des PCA955D im TWI-Bus sowie die Konfiguration der beiden I/O Ports (Port A Eingang, Port B Ausgang)
- Realisierung der eigentlichen Mainloop mit Abfrage der aktuellen Pinbelegung am Eingang und schalten der angeschlossenen LED bei Tasterbetätigung.

This example scrip enfolded following components

- To initialiise the TWI master
- To triggering the start sequence
- To addressing the PCA955D in a TWI Bus and to configurate the both I/O ports (port a input, port B output)
- To execution of the mainloop and the query of the pin assignment at the input and switching the connected LED with a push of the button.

Um dies realisieren zu können, sind weitere Definitionen im Programmkopf vorzunehmen, so muss das TWCR-Register sowie die Busgeschwindigkeit wie im unten aufgeführten Beispiel deklariert werden.
Weiterhin ist für den korrekten Ablauf die Initialisierung von eigenen Unterfunktionen notwendig. Konkret wird folgendes zusätzlich zu den Funktionen aus der Bibliothek benötigt:

To realise this, there are some more definitions in the program head are needed. So the TWCR register as well as the bus speed must be declared like in the example listed below.
Furthermore the initialization of insufficient functions of one's own is necessary for the correct expiry. The following concrete is needed in addition to the functions out of the library:

- `void twiInitPCA()`
  zur Initialisierung des PCA

- `void twiInitPCA()` to initialisation the PCA

- `void outPorts(uint8_t data, uint8_t ackn)`
  zur Übermittlung der gewünschten Pinbelegung (=data).

- `void outPorts(uint8_t data, uint8_tackn)`
  to transfer of the required pin assignment (=data)

- `uint8_t readPorts (uint8_t portNr)`
  zum Auslesen des gewünschten Ports (=PortNr) und Rückgabe der aktuellen Pinbelegung (data)

- `uint8_t readPorts (uint8_t portNr)`
  to read-out the ports (portNr) and return the state of the oin assignment

Am Ende sollte Ihr Programmkopf in dieser Form aussehen:

At the end the program descriptor should look in this form

```
//-------------------------------------------------------------------
// Titel     : Beispielprogramm Portexpander
//-------------------------------------------------------------------
// Funktion  : Ansteuerung einer LED via Tastereingabe
// Schaltung : ...
//-------------------------------------------------------------------
// Prozessor : ATmega8
// Takt      : 3.6864 MHz
// Sprache   : C
// Datum     : 24.09.09
// Version   : 1.0
// Autor     : Christian Müller
//-------------------------------------------------------------------
#define     F_CPU   3686400
#include <avr\io.h>
#define     BAUD    9600
//-------------------------------------------------------------------------------
// TWI-Funktionssammlung
#ifndef TWI_CLOCK
#define TWI_CLOCK 100000 // Geschwindigkeit des TWI-Busses
#endif
// TWCR - Control-Register-Bits
#define _TWINT 0b10000000
#define _TWEA 0b01000000
#define _TWSTA 0b00100000
#define _TWSTO 0b00010000
#define _TWWC 0b00001000
#define _TWEN 0b00000100
#define _TWIE 0b00000001

void twiInitMaster(uint8_t twiAdr);
void twiStart();
void twiStop();
void twiWriteByte(uint8_t data, uint8_t ackn);
uint8_t twiReadByte(uint8_t ackn);
void twiInitPCA();
void outPorts(uint8_t portNr, uint8_t data);
uint8_t readPorts(uint8_t portNr);
```

Folgend die eigentliche Mainloop mit den eingefügten Bibliotheksfunktionen:

Es wird der TWI-Master mit Geräte-ID 0x40 initialisiert und die I/O Belegung des PCA9555D konfiguriert.
Innerhalb der Endlosschleife wird die Pinbelegung am Port 0 abgefragt und mit dem Wert 0xFE (Bit 1-7 auf logisch 1, Bit 0 auf logisch 0 = Taster gedrückt) verglichen. Sollte der Taster geschlossen sein, werden am Port 1 alle Ausgänge auf High geschalten (LED leuchtet), sonst bleiben sie auf Low.

Furthermore the main loop with the included library function

There are initialised a TWI master with the device id 0x40 and an I/O configuration of the PCS 9555D.
Inside of the infinite loop the pin allocation is questioned at the port 0 and compared with 0xFE (bit 1-7 on logical 1, bit 0 on logical 0 = button closed). If the button are connect, at port 1 are all outputs get high (LED shining), otherwise they remain on Low.

```
//------------------------------------------------------------------------
// main
//------------------------------------------------------------------------
int main ()
{
    uint8_t data;
    twiInitMaster(0x40);
    twiInitPCA();
    do
    {
        data=readPorts(0);      // Abfrage Pins an PortA
        if(data == 0xFE)        // Taster gedrückt
        {
            outPorts(1,0xFF);   // alle Pins an PortB geschalten
        }
        else
        {
            outPorts(1,0x00);   // kein Pin an PortB geschalten
        }
        waitMs(100);            // Warte 100ms
    }
    while (true);
}

//------------------------------------------------------------------------
// twiInitMaster
//------------------------------------------------------------------------
void twiInitMaster(uint8_t twiAdr)
{
    // Clock
    TWBR=((F_CPU/TWI_CLOCK)-16)*2;
    // TWI-Status-Register (Vorteiler)
    TWSR=0;
    // Bus-Addr
    // TWAR=twiAdr;
    // Enable
    TWCR=_TWINT|_TWEN;
}
//------------------------------------------------------------------------
// Start TWI (ohne Interrupt)
//------------------------------------------------------------------------
void twiStart()
{
    uint8_t x = TWCR;
    x &= _TWEN|_TWIE;   // nur Beibehalten von Enable und InterruptJ/N
    TWCR = x|_TWINT|_TWSTA;
    // warten bis fertig
    while( !(TWCR & _TWINT))
    {}
}
//------------------------------------------------------------------------
// Stopp TWI (ohne Interrupt)
//------------------------------------------------------------------------
void twiStop()
{
    uint8_t x=TWCR;
    x &= _TWEN|_TWIE;   // nur Beibehalten von Enable und InterruptJ/N
    TWCR = x|_TWINT|_TWSTO;
}

//------------------------------------------------------------------------
// Write Byte per TWI (ohne Interrupt)
// PE: data = zu sendende Daten
//     ackn = wenn !=0 wird Acknowledge (=TWEA) gesetzt
//------------------------------------------------------------------------
```

```
void twiWriteByte(uint8_t data, uint8_t ackn)
{
    TWDR=data;      // Daten bereitlegen
    // Befehl zusammenstellen
    uint8_t x=TWCR;
    x&=_TWEN|_TWIE;     // nur Beibehalten von Enable und InterruptJ/N
    x|=_TWINT;
    if(ackn)
        x|=_TWEA;   // evt. TWEA setzen, für Datenanforderung
    TWCR=x;         // senden
    // warten bis fertig
    while( !(TWCR & _TWINT))
    {}
}
//---------------------------------------------------------------------------
// Read Byte per TWI (ohne Interrupt)
// PE: ackn = wenn !=0 wird Acknowledge (=TWEA) gesetzt
// PA: Data
//---------------------------------------------------------------------------
uint8_t twiReadByte(uint8_t ackn)
{
    // Befehl zusammenstellen
    uint8_t x=TWCR;
    x&=_TWEN|_TWIE;     // nur Beibehalten von Enable und InterruptJ/N
    x|=_TWINT;
    if(ackn)
        x|=_TWEA;   // evt. TWEA setzen, für Datenanforderung
    TWCR=x;         // senden
    // warten bis fertig
    while( !(TWCR & _TWINT))
    {}
    return TWDR;
}
//---------------------------------------------------------------------------
//  Funktionen für Portexpander
//---------------------------------------------------------------------------
// twiInitPCA
//---------------------------------------------------------------------------
void twiInitPCA()
{
    // I/O Definition (A Eingang, B Ausgang)
    twiStart();
    twiWriteByte(0x40,0);   // TWI-Adresse
    twiWriteByte(0x06,0);   // Adressierung des Kommandoregister = Config 0
    twiWriteByte(0xFF,0);   // ConfigDaten A = Eingang
    twiWriteByte(0x00,0);   // ConfigDaten B = Ausgang
    twiStop();
}
//---------------------------------------------------------------------------
// readPorts
//---------------------------------------------------------------------------
uint8_t readPorts(uint8_t port)
{
    twiStart();
    twiWriteByte(0x40,1);   // TWI-Adresse
    twiWriteByte(port,1);   // Kommandoregister 0/1 -> Input P0/P1
    twiStart();
    twiWriteByte(0x41,1);   // TWI-Adresse und Read
    return twiReadByte(0);
}
//---------------------------------------------------------------------------
// outPorts
//---------------------------------------------------------------------------
void outPorts(uint8_t port, uint8_t data)
{
    twiStart();
    twiWriteByte(0x40,1);       // TWI-Adresse
    twiWriteByte(0x02+port,1);  // Kommandoregister 2/3 -> Output P0/P1
    twiWriteByte(data,0);       //
}
//---------------------------------------------------------------------------
```

Beachte:
Die konkreten Porteinstellungen sind von der Rechnerkonfiguration abhängig. Besonders der USB Programmer mySmartUSB kann auf unterschiedlichen virtuellen COM Ports angemeldet werden. Es ist zu empfehlen, die COM Einstellung des mySmartUSB auf COM3 oder COM4 zu legen, da manche Werkzeuge wie das AVR Studio maximal COM4 zulässt. Die Zuweisung des COM Port erfolgt über den Gerätemanager.

Notice:
The precise port settings depend on the configuration of your PC. Especially the USB programmer mySmartUSB might be assigned to different virtual com ports. We recommend to use mySmartUSB with com 3 or com 4, as some tools (like AVR Studio) only support a com port up to com 4. You can change the com port settings in windows devia manager.

Versuchsaufbau mit Laborkarte

Breadboard construction Prototyping board

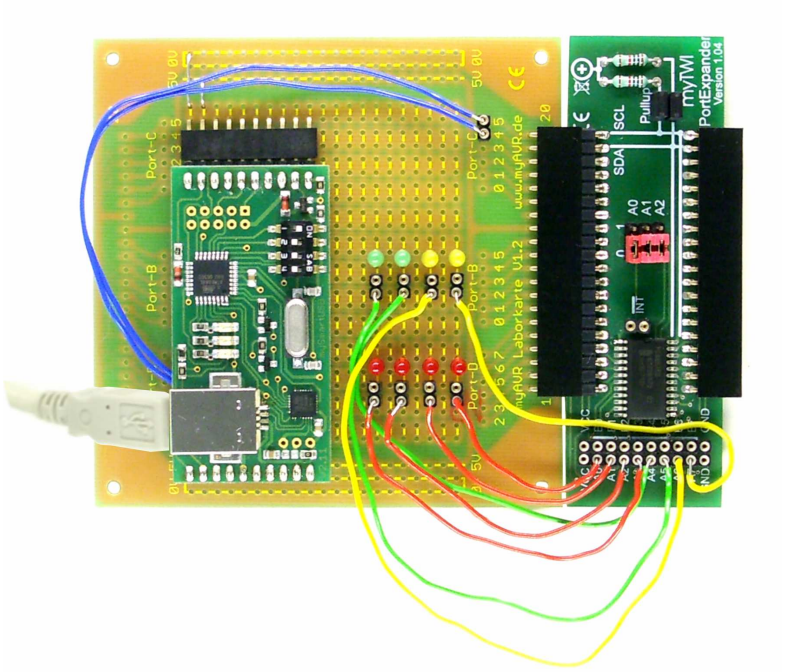


Abbildung: Beispielprogrammrealisierung verschiedener LED-Ansteuerungen durch mySmartUSB MK2 mit Laborkarte und myTWI PortExpander

Illustration: realization of program example of LED control via mySmartUSB MK2 with prototyping board and myTWI PortExpander

Allgemeine Sicherheitshinweise

Grundsätzlich ist der myTWI PortExpander nur zum Einsatz unter Lern- und Laborbedingungen konzipiert. Er ist nicht vorgesehen und nicht dimensioniert zur Steuerung realer Anlagen. Bei vorschriftsmäßigem Anschluss und Betrieb treten keine lebensgefährlichen Spannungen auf. Beachten Sie trotzdem die Vorschriften, die beim Betrieb elektrischer Geräte und Anlagen Gültigkeit haben. Wir versichern, dass die Leiterplatte durch den Hersteller getestet wurde. Für fehlerhaften und/oder vorschriftswidrigen Einsatz des Boards übernehmen wir keine Garantie.

Safety Guidelines

myTWI PortExpander is designed for educational and experimental use only. It is not intended and not dimensioned to control real industrial facilities. At correct use there will not occur extremely dangerous voltages. Nevertheless, be aware of general guidelines for using electronic devices. We assure that the PCB has been tested by the producer. For incorrect use and/or application contrary to technical regulations we are not liable.

Die aktuellsten Dokumente zum myTWI PortExpander finden Sie unter www.myAVR.de im Downloadbereich.
The latest documents for the myTWI PortExpander you can find at our homepage www.myAVR.com under „Download“.

Abbildungen können vom Inhalt abweichen. Änderungen im Sinne des technischen Fortschrittes behält sich der Hersteller vor.
Images may vary from the content. The manufacturers retains changes in terms of technical advances.