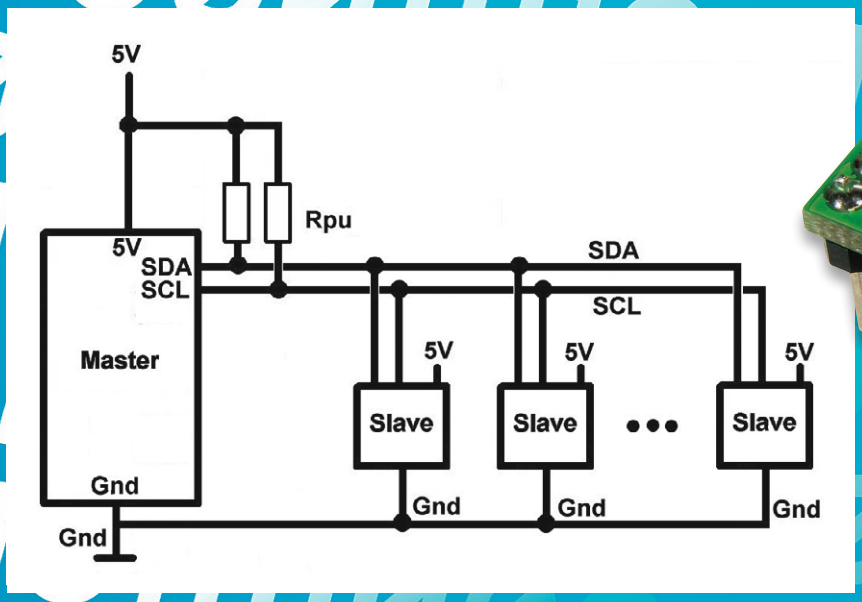
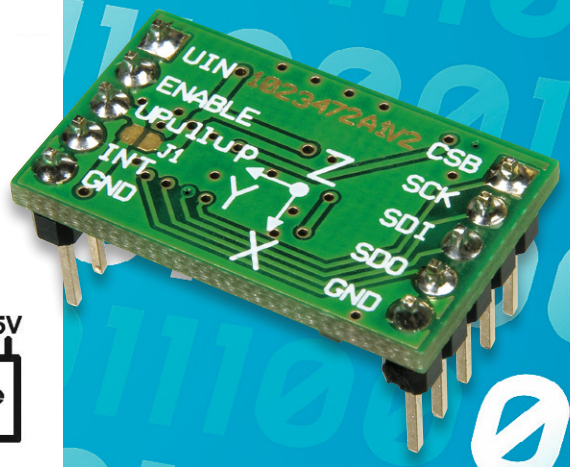
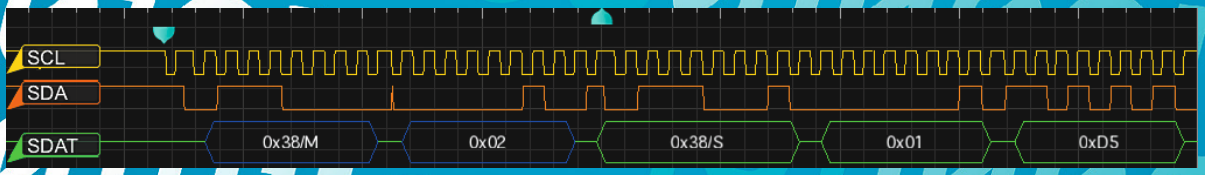
Digitale Hardwareschnittstellen

Teil 3: I²C







Die I²C-Schnittstelle ist eine synchrone serielle Schnittstelle, die in den frühen 1980er-Jahren von Philips Semiconductors (heute NXP Semiconductors) für die bidirektionale Datenübertragung zwischen Mikrocontrollern innerhalb eines Geräts (z. B. eines Fernsehers) erfunden wurde. Die Datenbits werden hintereinander (seriell) über eine Leitung mit dem Namen SDA (Serial Data) übertragen, wobei ein gemeinsamer, vom Master erzeugter Takt zwischen Sender und Empfänger für die Synchronisierung sorgt. Von einigen Herstellern, z. B. Atmel, wird aus lizenzrechtlichen Gründen von TWI (= Two Wire Interface) statt von I²C gesprochen, weil die Daten über zwei I/O-Leitungen übertragen werden.

Die I²C-Schnittstelle ist sehr interessant, weil von verschiedenen Herstellern sehr viele nützliche Sensoren bzw. Aktoren angeboten werden, die sich über I²C in eigene Anwendungen einbinden lassen. So gibt es beispielsweise 8-Bit-I/O-Expander-Bausteine (PCF8574), Echtzeituhrenbausteine (DS1307, DS3231 von Maxim), EEPROM-Bausteine (24C01, 24C02, 24C04, 24C08, 24C16) und eine ganze Reihe von I²C-Bausteinen zur Messung von Umweltgrößen wie z. B. Temperatur, Luftfeuchtigkeit und Luftdruck. Zur Messung der Temperatur gibt es beispielsweise LM75, TMP101, MCP9801. Ferner gibt es 8-Bit-A/D-D/A-Konverter (PCF8591), LED-Treiberbausteine (TLC59116, SAA1064), Bewegungssensoren (BMA020, LSM330) und vieles mehr [1]. ELV bietet einige I²C-Bausteine und Bausätze bzw. Fertigmodule mit I²C-Ansteuerung an, von denen einige Standard-I²C-Komponenten eingebaut haben und daher hier exemplarisch vorgestellt werden sollen.

Busaufbau

In Bild 1 sieht man die zwei I/O-Leitungen SCL und SDA, die typisch für den I²C-Bus sind. SCL steht für „Serial Clock Line" und dient der Übertragung eines gemeinsamen Takts zwischen Master und Slave. SDA steht für „Serial Data" und übernimmt die serielle Übertragung der Datenbits. Links im Bild sieht man den I²C-Master, der den Takt erzeugt und die Kommunikation mit einem der angeschlossenen Slaves beginnt. Die Slaves sind über die beiden I/O-Leitungen SCL und SDA sowie eine gemeinsame Gnd-Leitung verbunden und warten darauf, vom Master „angerufen" zu werden.

I²C ermöglicht eine bidirektionale Kommunikation zwischen einem Master und einem Slave. Dafür ist jede I²C-Komponente intern mit Open-Drain- bzw. Open-Collector-Schaltungen aufgebaut (Bild 2). Der I²C-Bus benötigt je einen Pull-up-Widerstand Rpu für die SCL- und die SDA-Leitung (Bild 1). Pro I²C-Bus sind nur zwei Pull-up-Widerstände zu verwenden, die in der Größenordnung von 1 bis 10 kΩ (typisch: 4,7 kΩ) liegen sollten. Durch die Pull-up-Widerstände liegen SCL und SDA im Ruhezustand (= Bus free) auf high (Betriebsspannung). Ein Master fragt ab, ob beide I/O-Leitungen high sind, bevor er eine Kommunikation startet. Er startet die Kommunikation durch eine fallende Flanke (High-low-Übergang) auf der SDA-Leitung, während die SCL-Leitung noch auf high liegt. Dann sendet der Master einen Takt (100 Kbit/s im Standard-Modus, teils 400 Kbit/s bis über 3,4 Mbit/s) auf SCL und seriell Datenbits auf SDA. Für bestimmte Situationen (Senden von Daten vom Slave, Taktverlangsamung) ist ein Slave in der Lage, seinerseits die Leitungen auf low (Gnd) zu ziehen (Bild 2). Nach dem Start einer Übertragung sendet der Master die Adresse (siehe Adressierung) eines Slaves auf die Datenleitung. Der Slave, der sich angesprochen fühlt, bestätigt durch Ziehen der Datenleitung auf low. Nach erfolgter Kommunikation zwischen Master und einem Slave erfolgt eine Freigabe des Busses durch eine steigende Flanke (low -> high) auf SDA, während SCL high ist.

Es ist durchaus möglich, dass Master und Slave unterschiedliche Betriebsspannungen haben – z. B. ein

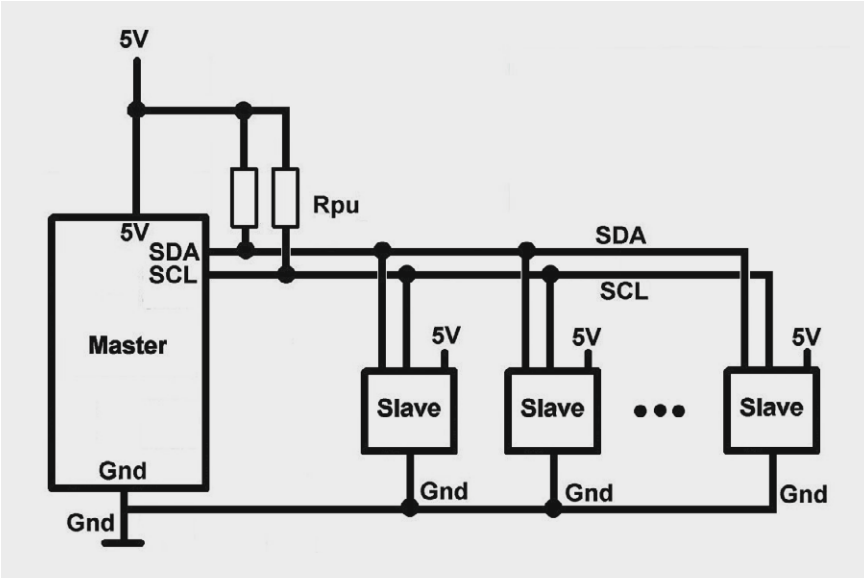


Bild 1: I²C-Bus

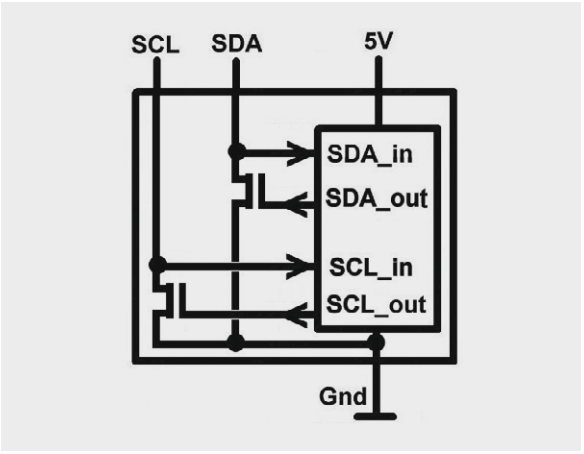


Bild 2: Aufbau eines I²C-Slaves

Sensor 3,3 V und der Master 5 V. In solchen Fällen muss ein sogenannter Levelshifter zwischengeschaltet werden.

Adressierung

Jeder Slave an einem Bus hat eine eindeutige Adresse, über die er vom Master angesprochen werden kann. Eine Slaveadresse besteht aus 7 Bits. Dadurch lassen sich insgesamt $2^7 = 128$ Adressen darstellen. Nach Abzug einiger reservierter Adressen lassen sich 112 Slaves in einem Bus adressieren! Wie in Bild 3 zu sehen ist, kommt zu der Adresse noch ein Bit hinzu, welches anzeigt, ob der Master zum Slave schreiben möchte (W = Write, Bit ist 0) oder Daten vom Slave lesen möchte (R = Read, Bit ist 1). Insgesamt sprechen wir von einem Adressbyte.

Oftmals lassen sich an gleichartigen Slaves unterschiedliche Adressen einstellen. Dadurch lassen sich z. B. mehrere gleichartige Temperatursensoren an einem Bus anschließen und über unterschiedliche Adressen ansprechen. Wie in Bild 3 angedeutet, haben Slaves dann meistens einen festen Adressteil von z. B. 4 Bits und einen variablen Adressbestandteil von z. B. 3 Bits, mit denen sich dann $2^3 = 8$ verschiedene Slaves eines Typs an einen Bus anschließen lassen. Welche Adresse ein I²C-Baustein hat, entnimmt man dem jeweiligen Datenblatt oder der Produktbeschreibung. Dabei muss man beachten, dass manchmal nur die 7 Bits der eigentlichen Slaveadresse zur Darstellung der Adresse verwendet werden und manchmal alle 8 Bits des Adressbytes (siehe Elektronikwissen).

Wer selbst in einer Programmiersprache Programme schreibt, kann die Adresse eines angeschlossenen Slaves leicht herausfinden, indem per Programm alle Schreibadressen (gerade Adressen, weil das R/W-Bit 0 ist) von 0 bis 128 einfach getestet werden. Wenn der angeschlossene Slave die entsprechende Adresse hat, dann antwortet er mit Acknowledge und man hat die unbekannte Adresse bzw. die Adresse, die man aus dem Datenblatt hatte, verifiziert. Man nennt so ein Testprogramm I²C-Adress-Scanner. Auf dem Raspberry kann man I2CDETECT eingeben, um die Adressen angeschlossener Slaves angezeigt zu bekommen.

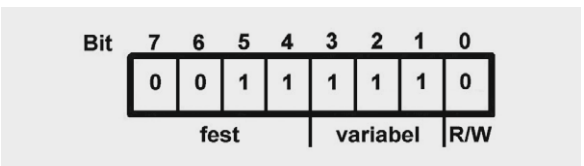


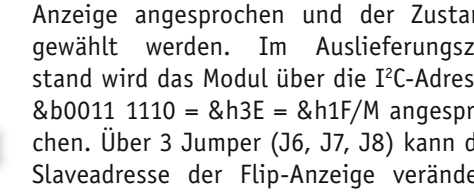
Bild 3: Slave-Adressbyte

I²C-Senden

Das Senden von Daten vom Master zu einem Slave erfolgt immer nach demselben Schema. Bild 4 zeigt das Senden eines Datenbytes. Nachdem der Master geprüft hat, ob der Bus frei ist, erzeugt er die Startbedingung. Dann sendet der Master die Adresse des anzusprechenden Slaves (zusammen mit dem Takt), und der Slave meldet sich durch Acknowledge. Danach wird das eigentliche Datenbyte vom Master zum Slave übertragen und die gesamte Übertragung mit der Stopp-Bedingung beendet, wodurch der Bus wieder freigegeben wird.

Flip-Anzeige (I²C-Senden)

Als einfaches Einstiegsbeispiel zur Verdeutlichung soll die (leider nicht mehr verfügbare) ELV-I²C-Flipanzeige dienen, welche eine Signalisierung zweier Zustände im Retro-Stil ermöglicht. Über I²C kann diese Anzeige angesprochen und der Zustand gewählt werden. Im Auslieferungszustand wird das Modul über die I²C-Adresse &b0011 1110 = &h3E = &h1F/M angesprochen. Über 3 Jumper (J6, J7, J8) kann die Slaveadresse der Flip-Anzeige verändert werden, wodurch dann mehrere Flip-Anzeigen an einem I²C-Bus über die nur zwei I²C-Datenleitungen steuerbar sind. Bild 5 zeigt den Anschluss eines Flip-Anzeigen-Moduls.



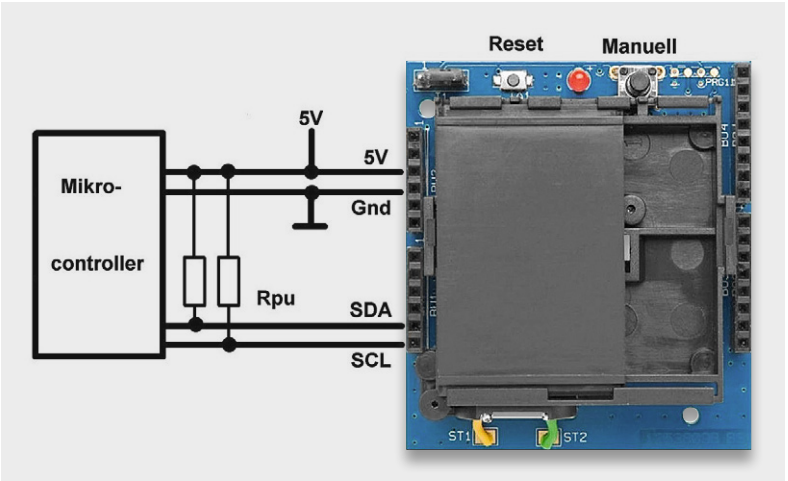


Bild 5: Anschluss Flip-Anzeige

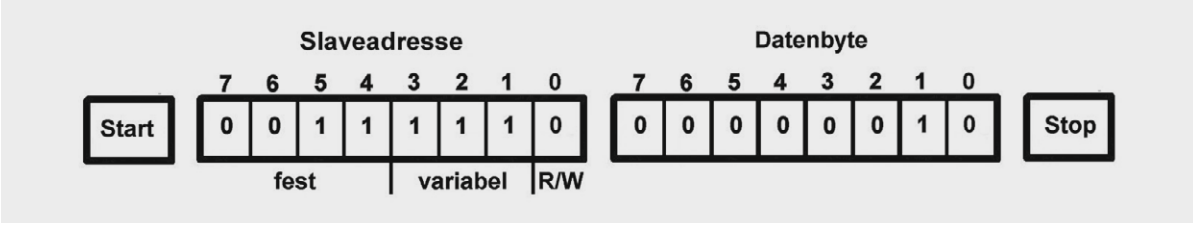


Bild 4: I²C-Senden

In Bild 6 sieht man eine I²C-Übertragung zu einer Flip-Anzeige inklusive der drei verschiedenen Darstellungsformen der Slaveadresse. Bild 7 zeigt den mit einem Logikanalysator gemessenen Signalverlauf. Die beiden oberen Linien zeigen jeweils die Taktleitung (SCL) und die Datenleitung (SDA). Fast jeder Logikanalysator kann Protokolle – wie hier I²C – dekodieren und gut lesbar darstellen. In Bild 7 sieht man jeweils in der dritten Linie die dekodierten Daten. Sehr gut sieht man, dass zunächst die Adresse (hier &h1F/M, dargestellt als 0x1F/M) übertragen wird und dann die Übertragung eines Datenbytes erfolgt. Bei der Flip-Anzeige bewirkt ein Datenbyte mit dem Wert &h02 das Bewegen der Anzeige nach „oben". Datenbyte &h04 schiebt die Anzeige nach „unten". Mit &h08 wechselt der jeweilige Zustand der Anzeige (Toggle).

Der Ablauf ist immer folgendermaßen:

- I²C-Startbedingung (fallende Flanke 1 -> 0 bzw. high -> low von SDA, während SCL 1 bzw. high ist)
- Slaveadresse (z. B. &h3E) senden
- Datenbyte senden (up: &h02, down: &h04, toggle: &h08)
- I²C-Stoppbedingung (steigende Flanke 0 -> 1 bzw. low -> high von SDA, während SCL 1 bzw. high ist)

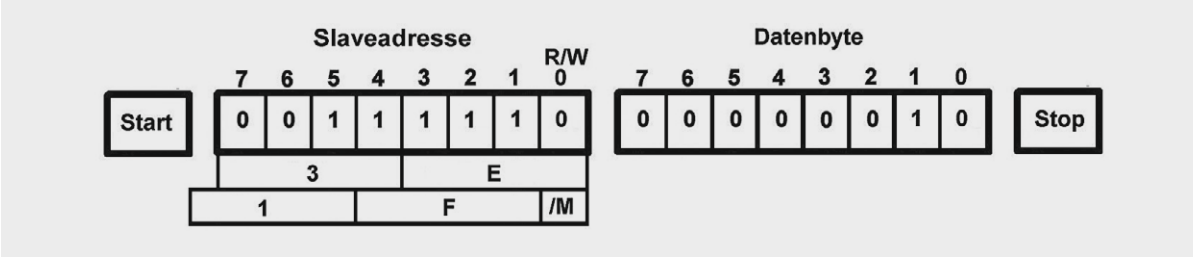


Bild 6: I²C-Senden Flip-Anzeige

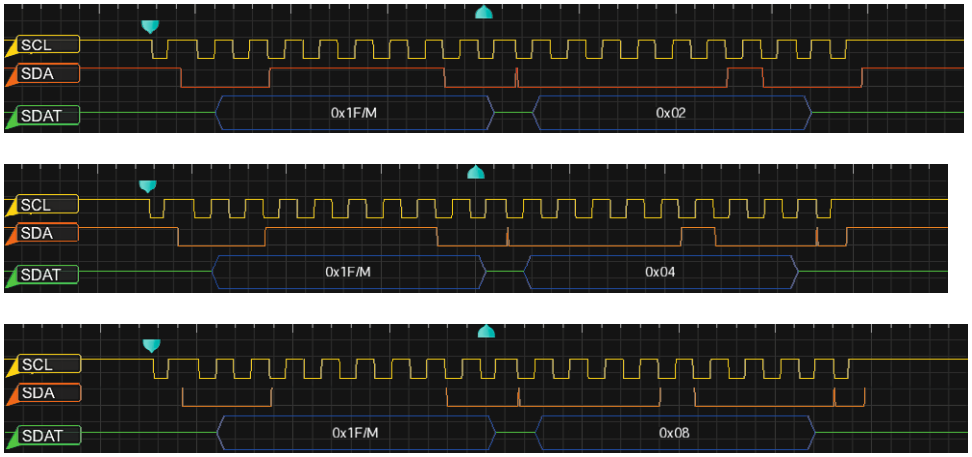
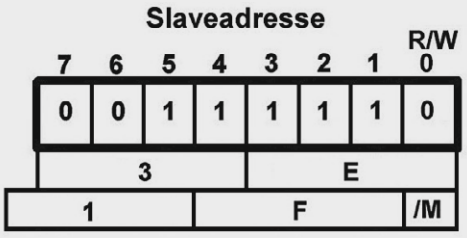


Bild 7: Signalverlauf Flip-Anzeige: up: &h02 down: &h04 toggle: &h08

Elektronikwissen

Darstellung von I²C-Adressen

Bei der Darstellung einer I²C-Slaveadresse findet man (leider) zwei verschiedene Arten. Dies muss bei der Benutzung von Programmiersprachen oder Tools, beim Lesen von Datenblättern und auch beim Benutzen eines Logikanalysators beachtet werden. Im Idealfall hat man ein Datenblatt, in dem das Adressbyte in binärer Darstellung – z. B. &b00111110 – zu sehen ist[1]. Diese Binärdarstellung wird oft abkürzend durch eine hexadezimale Darstellung ersetzt. Dabei werden die ersten 4 Bits des Adressbytes und die hinteren Bits des Adressbytes getrennt in hexadezimale Schreibweise umgesetzt. Aus &b0011 wird dann z. B. eine hexadezimale &h3 und aus binär &b1110 wird hexadezimal &hE. Insgesamt wird dann von &h3E als I²C-Adresse gesprochen. Wenn in Dokumentationen, Programmiersprachen, Tools oder Logikanalysatoren nur die eigentliche Slaveadresse dargestellt wird, dann werden nur die höherwertigen 7 Bits des Adressbytes betrachtet.



Die Interpretation der Binärdarstellung erfolgt also um ein Bit nach links verschoben. Im Beispielbild erhält man aus den linken Bits &b001 eine hexadezimale &h1 und aus den folgenden 4 Bits &b1111 eine hexadezimale &hF. Insgesamt also &h1F.

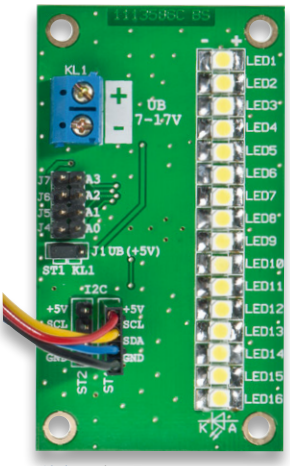
Das niedrigstwertige Bit, welches Write (0) bzw. Read (1) bedeutet, wird dann manchmal als /W bzw. /R dargestellt. Man sieht dann z. B. &h1F/M bzw. &h1F/S.

1) &b und &h nennt man Präfixe. Sie zeigen an, in welchem Zahlensystem eine Zahl dargestellt ist. Nach &b (oder 0b) folgt eine binär/dual dargestellte Zahl. Nach &h (oder 0x) folgt eine hexadezimal dargestellte Zahl. Dezimalzahlen schreibt man meist ohne Präfix.

LED-Treiber

Oft muss man mehrere Datenbytes hintereinander vom Master zum Slave übertragen. Beim ELV-Modul LED-I²C-Steuertreiber, 16 Kanäle (CM-09 83 77) lassen sich z. B. 16 (sechzehn!) LEDs individuell in ihrer Helligkeit über die zwei I²C-Steuerleitungen steuern (Bild 8).

Die Ansteuerung des Moduls beginnt mit mehreren vorbereitenden Schritten.



1. Reset

Zunächst wird entsprechend Abschnitt 9.3.5 im Datenblatt des TLC59116 [2] alles auf Standardwerte gesetzt (Software-Reset, Bild 9, Bild 10):

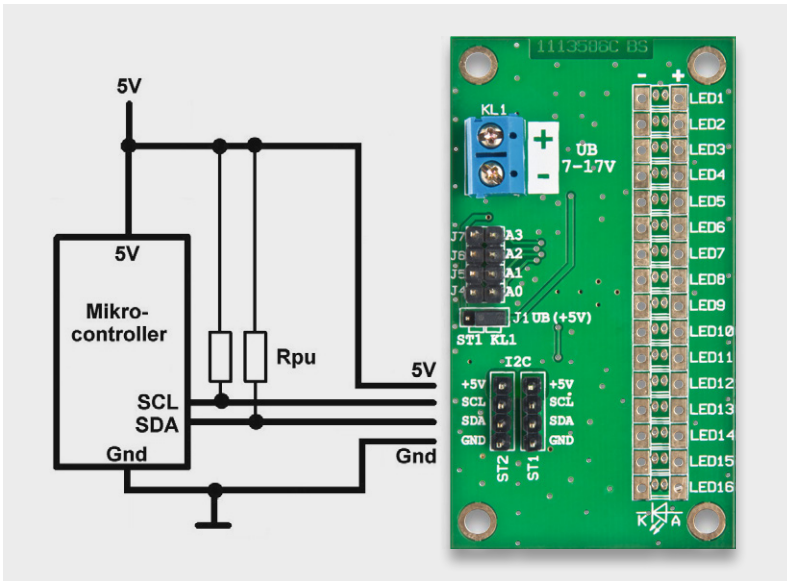


Bild 8: Anschluss LED-Treiber

- I²C-Adresse für Reset: &1101 0110 = &hD6 = &h6B/M
- Datenbytes: &hA5 &h5A

Das Ansprechen des LED-Treibers erfolgt dann immer nach dem Schema:

- I²C-Startbedingung senden (SDA geht auf low, während SCL high ist)
- I²C-Slaveadresse senden
- Kontrollregister senden: AAAD DDDD für Autoinkrement-Modus (AAA) (9.5.11. im Datenblatt [2]) und Start-Registeradresse (D_DDDD) (Tabelle 6 im Datenblatt [2])
- Wert schreiben, der in das jeweilige Register geschrieben werden soll
- Ggf. weitere Datenwerte senden
- I²C-Stoppbedingung senden (SDA geht auf high, während SCL high ist)

2. Init

Als zweite vorbereitende Maßnahme werden laut Seite 15 des Datenblatts alle LEDs initialisiert:

- I²C-Adresse für alle: &1101 0000 = &hD0 = &h68/M
- Datenbytes: &h80 &h01

&h80 = &b1000 0000 = Autoinkrement (100) ab Adresse 0

&h01 = Bit 0 auf 1 und alle anderen Bits des Mode-Registers 1 (Tabelle 4 in [2]) auf 0. Bit 4 im Mode-Register 1 ist nämlich im Lieferzustand 1, muss aber 0 sein für „Normal Mode".

Zitat aus dem Datenblatt: *„The Bit 4 must be set to 0 before any outputs will turn on. Proper operation requires this bit to be 0. Setting the bit to a 1 will turn all channels off."* (Bild 11).

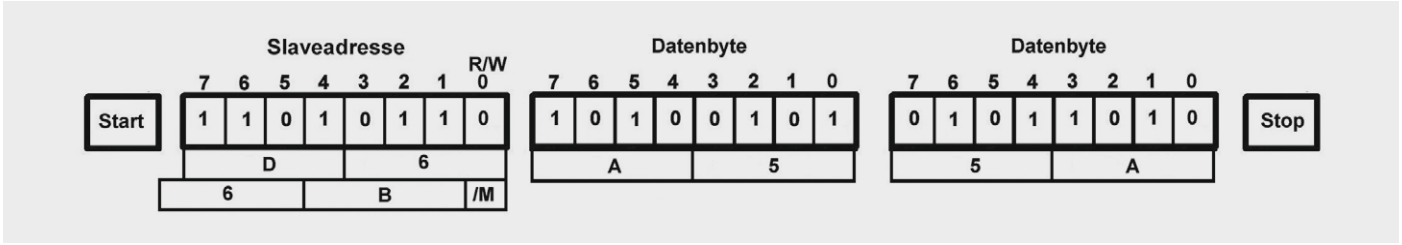


Bild 9: I²C-Senden LED-Treiber (Reset)

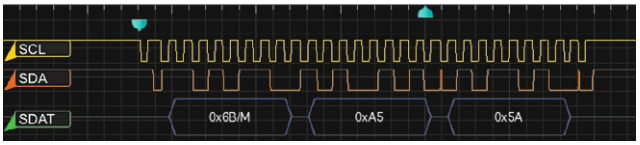


Bild 10: Signalverlauf LED-Treiber (Reset)

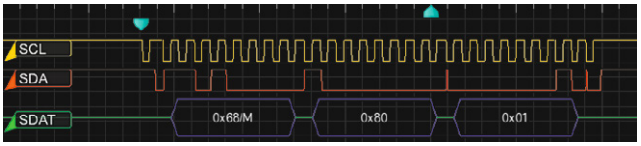


Bild 11: I²C-Senden LED-Treiber (alle initialisieren)

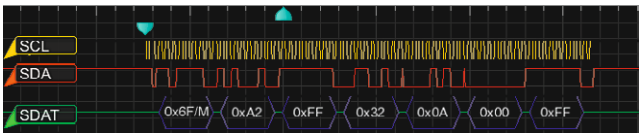


Bild 12: I²C-Senden LED-Treiber (an einzelnen Treiber mit der Slaveadresse 0x6F)

3. LED-Steuerung einschalten

Gemäß Tabelle 9 im Datenblatt [2] muss in Register 14 der Output-State der LEDs jeweils auf FF gesetzt werden. Im Auslieferungszustand sind die LEDs alle aus.

- I²C-Adresse für alle: &1101 0000 = &hD0 = &h68/M
- Datenbytes: 94 FF FF FF FF

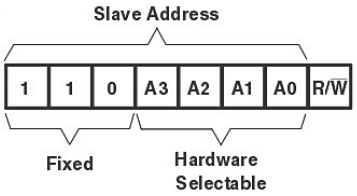
&h94 = &b1001 0100 Autoinkrement 1001 ab Register 14. FF FF FF FF Werte der Register

4. LEDs ansteuern

Nun können die 16 LEDs je Treiberbaustein einzeln auf individuelle Helligkeit eingestellt werden (Tabelle 6 im Datenblatt [2]). Im Auslieferungszustand hat so ein LED-Treibermodul die I²C-Adresse eines TLC59116: &1101 1110 = &hDE = &h6F/M.

Durch 4 Jumper lassen sich 2^4 = 16 Adressen einstellen. Nach Abzug einer festgelegten Resetadresse und ei-

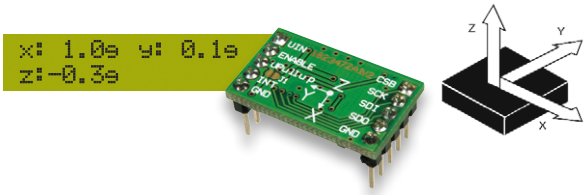
ner Alle_Adresse sind 14 Treiber adressierbar und dadurch 14 * 16 = 224 LEDs ansteuerbar – über zwei I²C-Datenleitungen!



Die Kommunikation sieht dann so aus:

- I²C-Adresse für einen TLC: &1101 1110 = &hDE = &h6F/M
- Datenbytes: A2 FF 32 0A 00 FF
  &hA2 = &b1010 0010, also Autoinkrement (101) ab Register 2.

Insgesamt sieht also das Ansprechen der LEDs in Kurzform so aus:
Adresse – Kontrollregister – PWM0-Wert – PWM1-Wert – PWM2-Wert – PWM3 – PWM4 – …
wobei 0 = aus, 255 = sehr hell (Bild 12)



3-Achsen-Bewegungssensor 3D-BS

Am Beispiel eines Beschleunigungssensors (BMA020) wird nun das Senden und Empfangen von Daten gezeigt. Mit dem ELV-3-Achsen-Beschleunigungssensor 3D-BS (Bausatz: CM-09 15 21 bzw. Fertiggerät: CM-10 48 93) lassen sich Beschleunigungswerte in drei Dimensionen elektronisch messen. Den Anschluss zeigt Bild 13.

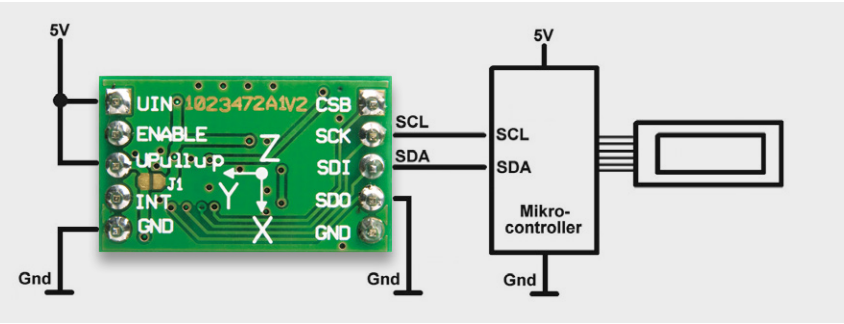


Bild 13: 3D-BS-Anschluss

Die Slaveadresse ist fest vorgegeben:
&b0111 0000 = &h70 = &h38/M

Zunächst einmal muss das Konfigurationsregister (hexadezimal 14, Bild 14) des im 3D-BS verbauten ICs BMA020 [3] beschrieben werden (Bild 15).

- I²C-Start
- Slaveadresse &b0111 0000 = &h70 = &h38/M senden
- Zu beschreibendes Register (&h14) senden
- Datenbyte (&h00) senden
- I²C-Stopp

Dadurch wird die Empfindlichkeit des Sensors festgelegt (vgl. 3.1.2 im Datenblatt [3]).

Zum Lesen der Beschleunigungswerte aus dem Sensor wird nun das I²C-Schema aus Bild 4 erweitert (Bild 17) und sieht folgendermaßen aus:

- I²C-Start
- Slave-Schreibadresse senden (&b0111 0000 = &h70 = &h38/M)
- Registernummer, ab der Werte gelesen werden sollen (&h02) (Bild 16)
- I²C-Start
- Slave-Leseadresse (&b0111 0001 = &h71 = &h38/S) schreiben
- Datenbytes empfangen
- I²C-Stopp

Register Address (hexadecimal)	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
14h	reserved			range<1:0>		bandwidth<2:0>		

Bild 14: Register für die Konfiguration des 3D-BS (Datenblattauszug)

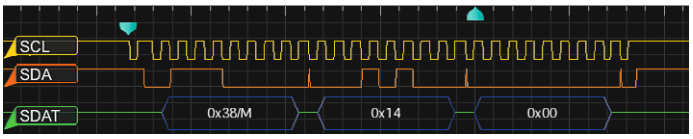


Bild 15: Signalverlauf 3D-BS bei Konfiguration

Register Address (hexadecimal)	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
07h	acc_z<9:2> (msb)							
06h	acc_z<1:0> (lsb)		unused					new_data_z
05h	acc_y<9:2> (msb)							
04h	acc_y<1:0> (lsb)		unused					new_data_y
03h	acc_x<9:2> (msb)							
02h	acc_x<1:0> (lsb)		unused					new_data_x

Bild 16: Register für die Sensorwerte des 3D-BS (Datenblattauszug)

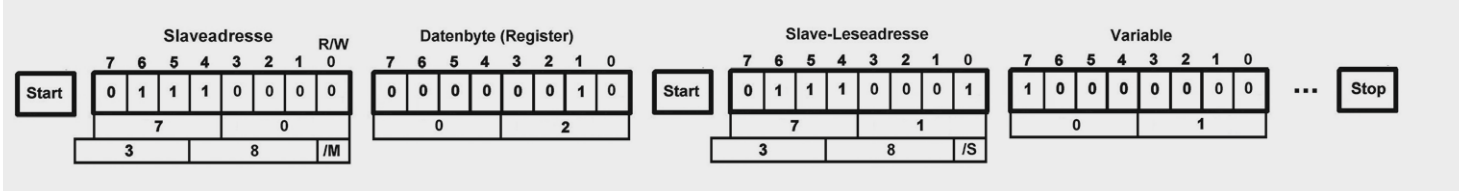


Bild 17: I²C-Lesen

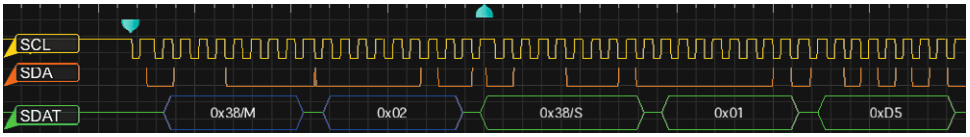


Bild 18: Signalverlauf 3D-BS Sensorwerte auslesen

In Bild 18 ist der Signalverlauf für diese I²C-Kommunikation zu sehen. In der vom Logikanalysator dekodierten Darstellung sieht man den Ablauf genau so, wie er auch im Datenblatt beschrieben ist und wie er typisch für jeden I²C-Lesevorgang ist. Nach der Startbedingung wird die Schreibadresse vom Master zum Slave gesendet. Danach wird die Startadresse gesendet, ab der im Nachfolgenden gelesen werden soll. Nach einem erneuten Start wird die Leseadresse gesendet und danach werden die Datenbytes vom Slave auf dem Master empfangen.

4-Digit-Anzeige und Temperatursensor

Unter der Bezeichnung I²C-4-Digit-LED-Display I2C-4DLED (CM-10 56 97) bietet ELV ein sehr interessantes, kompaktes Modul an, welches eine vierstellige 7-Segment-Anzeige, einen Temperatursensor und vier Taster umfasst. Zur Ansteuerung der Segmente und Dezimalpunkte der vierstelligen 7-Segment-Anzeige ist ein Standard-IC SAA1064 verbaut. Man könnte mit diesem IC statt einer vierstelligen 7-Segment-Anzeige auch anders angeordnete 4 x 8 = 32 LEDs ansteuern.

Die Slaveadresse des 7-Segment-Treibers ist &b0111 0000 = &h70 = &h38/M.

Es lassen sich vier verschiedene Slaveadressen einstellen und somit an einem I²C-Bus vier verschiedene dieser 7-Segment-Treiber-Module anschließen (Bild 19).

Gemäß Datenblatt [4] werden zunächst einmalig die Grundeinstellungen vorgenommen.

1. Initialisierung
- I²C-Start
- Slaveadresse schreiben
- Register für Controlbyte (0) schreiben
- &b01110111 schreiben
- I²C-Stopp

Dann können die vier Stellen der Anzeige nach dem nun bekannten I²C-Schema angesprochen werden.

2. Segmente beschreiben:
- I²C-Start
- Slaveadresse schreiben
- Wert für erstes Register (1) schreiben
- Datenbyte für erste Stelle schreiben
- Datenbyte für zweite Stelle schreiben
- Datenbyte für dritte Stelle schreiben
- Datenbyte für vierte Stelle schreiben
- I²C-Stopp

Der hier verbaute Temperatursensor ist vom Typ MCP9801 [5].

Um die Temperatur aus dem Sensor auszulesen, wird zunächst zur Initialisierung eine Schreibsequenz gesendet und dann (z. B. im Sekundenrhythmus) die Temperatur abgerufen.

1. Temperatursensor initialisieren:
- I²C-Start
- Slaveadresse schreiben
- Nummer des Kontrollregisters (1) schreiben
- Datenbyte (0) für 9-Bit-Auflösung etc. schreiben
- I²C-Stopp

2. Temperatursensor lesen:
- I²C-Start
- Slave-Schreibadresse schreiben
- Registernummer, ab der gelesen werden soll (0), schreiben
- I²C-Start
- Slave-Leseadresse schreiben
- Datenbyte für Vorkomma einlesen
- Datenbyte für Nachkomma einlesen
- I²C-Stopp

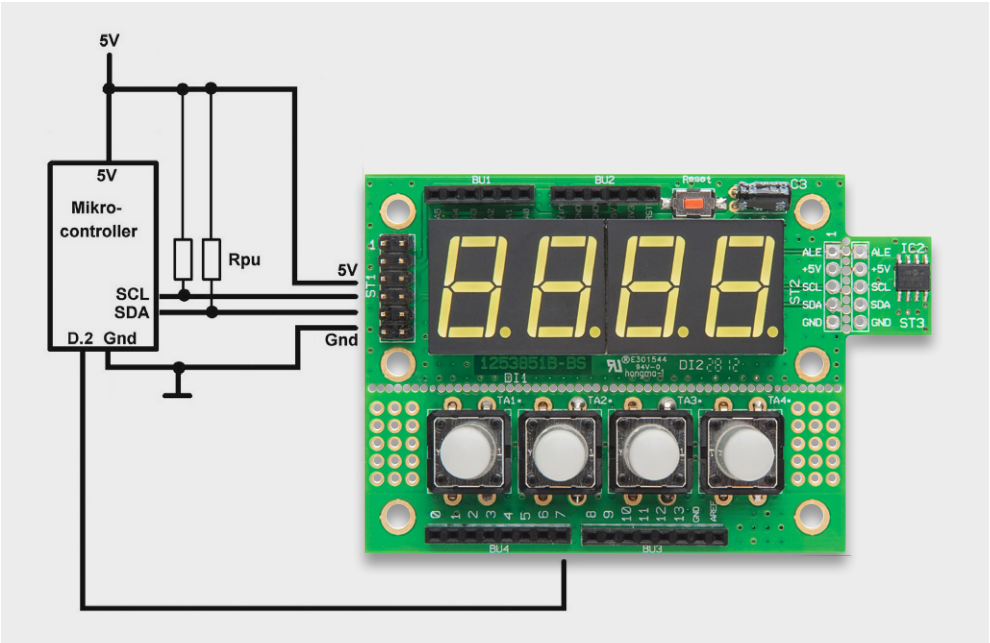
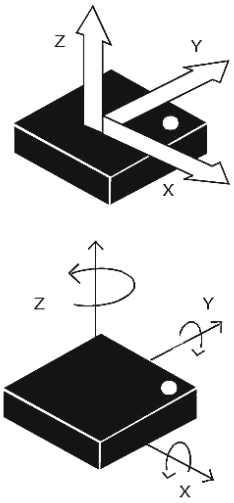


Bild 19: Anschluss 4-Digit-Modul



6-Achsen-Bewegungssensor 6D-BS

Ein weiteres I²C-Modul von ELV ist der 6-Achsen-Bewegungssensor 6D-BS (CM-13 05 98), der nicht nur Beschleunigungswerte in drei Dimensionen, sondern auch Rotationswerte misst. Verbaut ist hier ein IC vom Typ LSM330 [6]. Das Vorgehen ist wieder wie in den oben gezeigten Beispielen. Man muss sich also nur die entscheidenden Werte aus dem Datenblatt heraussuchen:
Slaveadresse schreiben: &b0011 0010 = &h32 = &h19/M
Slaveadresse lesen: &b0011 0011 = &h33 = &h19/S
Konfiguration: Control-Register (&h20) mit &h77 beschreiben
Auslesen: ab Register &hA8 sechs Werte einlesen

Ebenso ist bei den anderen ELV-I²C-Modulen wie RTC, DCF77-RTC etc. und auch bei allen sonstigen verfügbaren I²C-Modulen zu verfahren.

Weitere Infos:

[1] Liste von I²C-Bausteinen: www.digikey.ca/products/en?vendor=0&keywords=I2C
[2] LED-Treiber-IC für 16 LEDs TLC59116: www.ti.com/lit/ds/symlink/tlc59116.pdf
[3] Datenblatt 3D-Bewegungssensor BMA020: www.elv.de: Webcode #10072
[4] Datenblatt SAA1064 4-Digit-LED-Treiber: http://pdf.datasheetcatalog.com/datasheet/philips/SAA1064.pdf
[5] MCP9801-Temperatursensor Datenblatt: ww1.microchip.com/downloads/en/DeviceDoc/21909d.pdf
[6] Datenblatt 6D-Bewegungssensor LSM330: www.st.com/content/ccc/resource/technical/document/datasheet/c6/c5/7b/14/c6/10/42/2f/DM00059856.pdf/files/DM00059856.pdf/jcr:content/translations/en.DM00059856.pdf

Fazit

Die I²C-Schnittstelle ist sehr weit verbreitet – man findet sehr viele I²C-Komponenten am Markt – und ermöglicht es, sehr viele angeschlossene Slaves über nur zwei Datenleitungen schreibend und lesend anzubinden. I²C ist leicht zu verstehen, leicht zu benutzen und wird von praktisch jeder Programmiersprache unterstützt.

Im nächsten Teil dieser Artikelserie geht es um die 1-Wire-Schnittstelle, die vor allem durch die Ansteuerung der DS18(S/B)20-Temperatursensoren bekannt ist. ELV

Preisstellung Februar 2017 – aktuelle Preise im Web-Shop

Empfohlene Produkte	Best.-Nr.	Preis
LED-I²C-Steuertreiber, 16 Kanäle	CM-09 83 77	€ 12,95
I²C-4-Digit-LED-Display I2C-4LED	CM-10 56 97	€ 16,95
3-Achsen-Beschleunigungssensor 3D-BS – Bausatz	CM-09 15 21	€ 6,95
3-Achsen-Beschleunigungssensor 3D-BS – Fertiggerät	CM-10 48 93	€ 9,95
6-Achsen-Bewegungssensor 6D-BS	CM-13 05 98	€ 15,60
I²C-Realtime-Clock I2C-RTC	CM-10 34 13	€ 7,95
Real-Time-Clock-DCF-Modul mit I²C-, SPI- und UART-Schnittstelle RTC-DCF, Komplettbausatz	CM-13 05 41	€ 12,95
I²C-Bus-Displaymodul I2C-LCD	CM-09 92 53	€ 13,95
LED-Bussystem LED-B6	CM-08 53 20	€ 10,40
2-pol. Anschlussleitung, passend für Miniatur-Stiftbuchse	CM-07 60 55	€ 1,25
Verbindungskabel, 2 Module	CM-08 56 90	€ 2,05
Adapterplatine AP-Si4735	CM-10 34 39	€ 15,09
FM-Receiver-Modul mit SI4705 (UKW-Radio-Modul) FM-RM1	CM-14 09 84	€ 22,95
Intelligentes Schrittmotor-Treibermodul iSMT	CM-09 27 20	€ 24,95
USB-I²C-Interface USB-I2C	CM-09 22 55	€ 34,95
oder	CM-08 41 23	€ 24,95
2-Kanal-USB-Speicher-Oszilloskop LabNation Smartscope	CM-12 37 24	€ 229,–
USB-2.0-OTG-Kabel Micro-B/Mini-B für Android-Geräte, 0,5 m	CM-12 70 16	€ 5,95
Logikanalysator Oscium LogiScope	CM-11 53 40	€ 159,95
ELV-Triggergenerator TG1 für SPI/I²C/UART, Komplettbausatz	CM-14 21 24	€ 29,95