11 I2C Schnittstelle

Student Group

First Name	Surname	Matrikel Nr.

Table of Contents

10. I2C Schnittstelle

Ziele

Nach dieser Lektion sollten Sie:

1. wissen wie die Kommunikation zwischen I2C Master und Slave funktioniert

Video

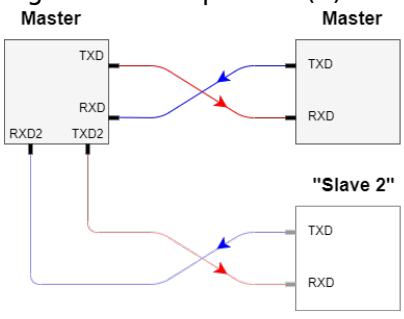
Dokumentation von Atmel

- Application Note: TWI Module as I2C Master
- Application Note: TWI Module as I2C Slave
- alternative Implementierung von Elia Ritterbusch

Übersicht über die am häufigsten verwendeten, seriellen Schnittstellen

USART

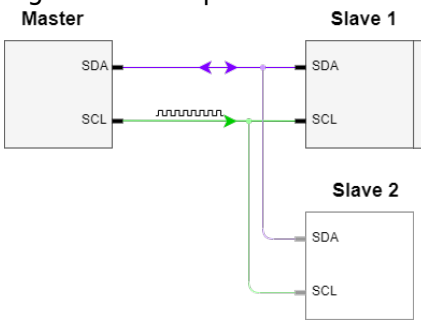
Fig. ##: Konzeptbild U(S)ART



- Keine gibt Takt vor. Es sind gleichberechtigte Kommunikationspartner (siehe figure ##).
- Jeder darf zu jederzeit senden.
- Senden und Empfangen geschieht über zwei

I2C

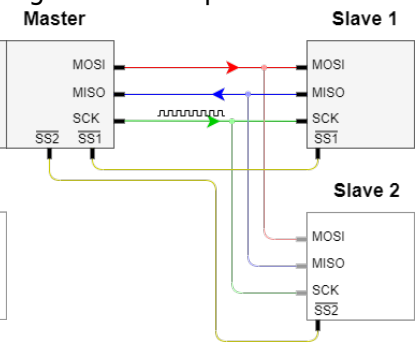
Fig. ##: Konzeptbild I2C



- Master gibt Takt vor (siehe figure ##).
- Slave darf nur zu bestimmten Zeiten senden und nur, wenn der Master dies anfordert.
- Senden und Empfangen geschieht über die

SPI

Fig. ##: Konzeptbild SPI



- Master gibt Takt vor (siehe figure ##).
- Slave darf nur zu bestimmten Zeiten senden und nur, wenn der Master dies anfordert.
- Senden und Empfangen

separate Leitungen.
- Kommunikation ist nur zwischen zwei Geräten möglich.
  Ein weiterer Slave würde eine weiteren U(S)ART-Bus benötigen.

gleiche Leitung.
- Alle Slaves hören am gleichen Bus mit und schreiben auf die gleiche Leitung.
- Jeder Slave muss anhand der Signale überprüfen, ob die Daten für ihn gemeint sind.

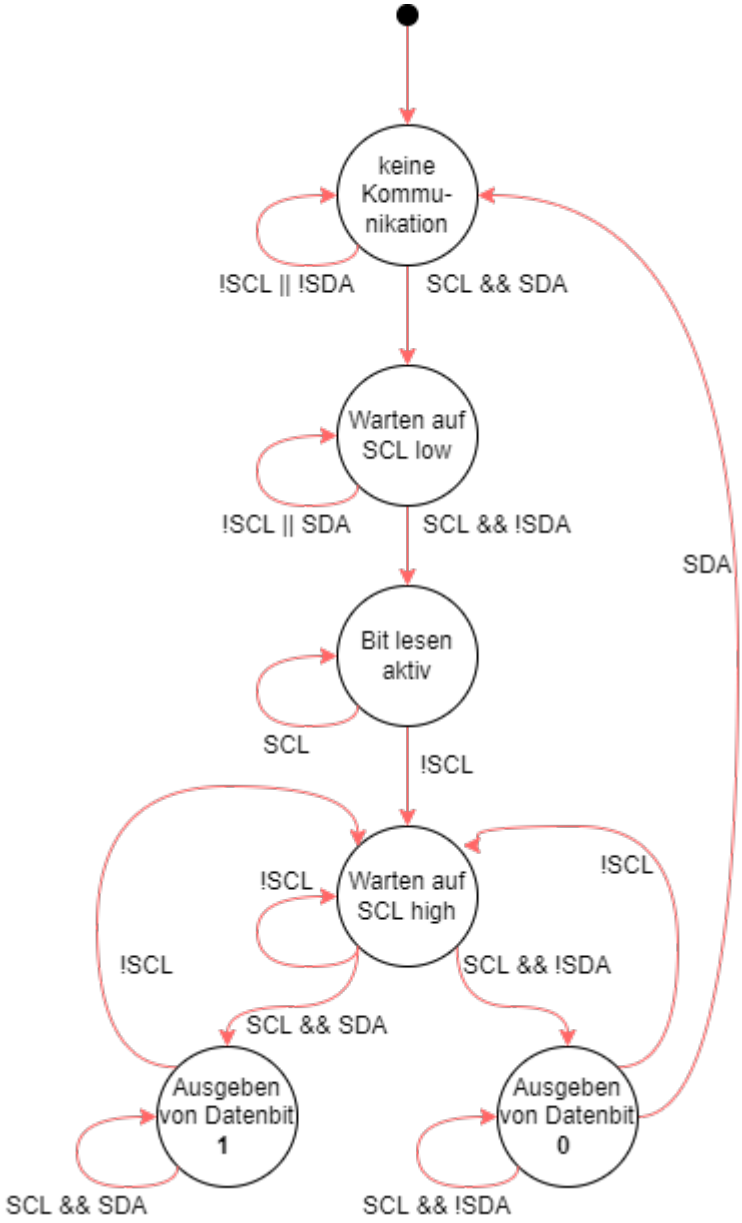
geschieht über zwei separate Leitungen.
- Alle Slaves hören auf der gleichen Leitung mit und schreiben auf die gleiche Leitung.
- Der gewünschte Slave wird über die Slave Select Leitung ausgewählt.

Statemachine für Datenpaket

Statemachine der I2C Kommunikation

Statemachine der I2C Kommunikation

Fig. 1: Statemachine der I2C Kommunikation

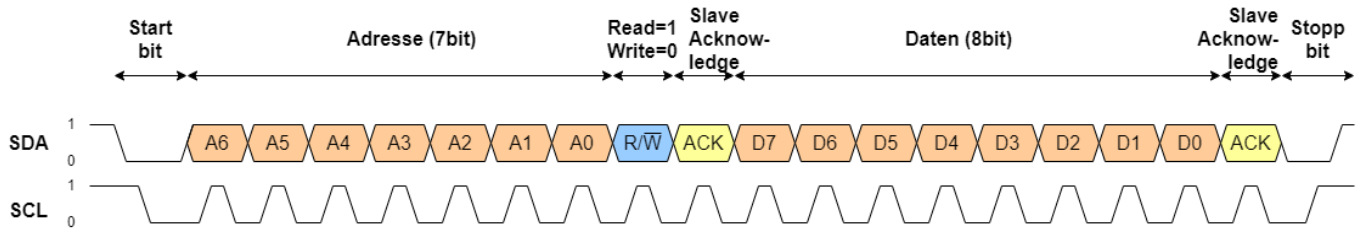


I2C in Kürze und Zeitverlaufsdiagramm

Übertragung

Für die I2C Übertragung “trommelt” der Master-IC auf der Taktleitung (SCL = Serial Clock). Bei jedem “Trommelschlag” (SCL=High), darf der Slave die Datenleitung (SDA = Serial Data) lesen.
D.h. während der Datenübertragung bleibt die Datenleitung bei SCL=High konstant.
Eine Flanke (=Signalwechsel) auf der Datenleitung während SCL=High definiert Beginn und Ende der Kommunikation.
Eine fallende Flanke auf SDA bei SCL=High stellt das Startbit dar, eine steigende Flanke das Stoppbit.
Läuft keine Kommunikation sind Daten- und Tankleitung auf High.

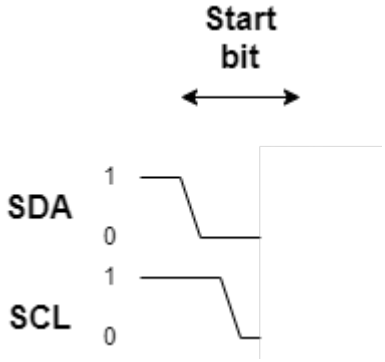
Fig. 3: Zeitverlaufsdiagramm der I2C Kommunikation



Startbedingung

Um die Übertragung zu beginnen muss die Startbedingung eingeleitet werden. Während SCL HIGH ist (a), geht SDA von HIGH auf LOW. Anschließend startet SCL mit LOW (b).

Fig. 4: Startbedingung



Für eine Startbedingung werden die Bits innerhalb des TWCR wie folgt gesetzt:

main.cpp

```
42. TWCR = (1<<TWINT)|(1<<TWEN);        //
    Setting TWINT clears interupt flag
43.                                     // to
    set the following state:
44.      | (1<<TWIE )                   //
    Enable TWI Interrupt.
45.      | (1<<TWSTA)|(0<<TWSTO);       //
    Initiate a START condition.
```

Übertragung

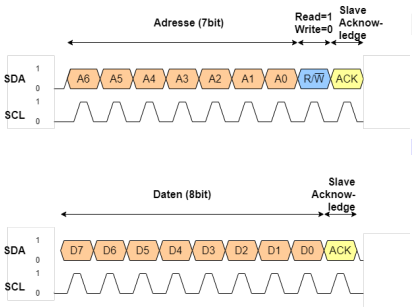
Die entscheidende Voraussetzung für eine erfolgreiche Bitübertragung ist, dass sich der Zustand von SDA nur ändern darf solange SCL auf LOW ist. Allerdings ist der Zustand von SDA erst gültig, wenn SCL auf HIGH ist.

Fig. 5: Übertragung

Für die Übertragung eines Bytes muss TWDR und TWCR wie folgt gesetzt werden.
Zunächst wird die Übertragung der Adresse (SLA_W) betrachtet:

main.cpp

```
42. TWDR = SLA_W;                       //
    Load SLA_W into TWDR
43. TWCR = (1<<TWINT)|(1<<TWEN);        //
    Setting TWINT clears interupt flag
44.                                     // to
    start transmission of address
```



Die Daten (DATA) werden in gleicher Weise übertragen:

main.cpp

```
42. TWDR = DATA;                          //
    Load DATA into TWDR
43. TWCR = (1<<TWINT)|(1<<TWEN);          //
    Setting TWINT clears interupt flag
44.                                       // to
    start transmission of address
```

Stoppbedingung

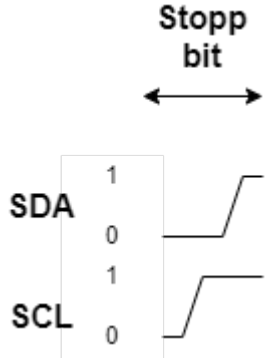
Die Stoppbedingung beendet die Übertragung. SCL geht auf HIGH (c), anschließend wechselt die SDA-Leitung von LOW nach HIGH (d).

Für eine Stoppbedingung werden die Bits innerhalb des TWCR wie folgt gesetzt:

main.cpp

```
42. TWCR = (1<<TWINT)|(1<<TWEN);          //
    Setting TWINT clears interupt flag
43.                                       // to
    set the following state:
44.       | (1<<TWIE )                    //
    Enable TWI Interrupt.
45.       | (0<<TWSTA)|(1<<TWSTO);        //
    Initiate a STOP condition.
```

Fig. 6: Stoppbedingung



Beispiele

- Simulide: …\share\simulide\examples\Arduino\sofware_i2c_lcd\i2c_lcd-arduino (hierbei wird Software I2C eingesetzt)
- Software I2C:
  - Library von Peter Fleury: library, Dokumentation