

Eingebettete Systeme

Prof. Dr. rer. nat. Peter Gerwinski

16. November 2022

Eingebettete Systeme

<https://gitlab.cvh-server.de/pgerwinski/es>

1 Einführung

2 Einführung in Unix

3 TCP/IP in der Praxis

...

3.6 Netzwerkanalyse

3.7 SSH

3.8 X11 und VNC

3.9 Programmierung

4 Bus-Systeme

4.1 Was sind Bus-Systeme?

4.1 RS-232

4.1 I²C (TWI)

4.1 SPI

4.1 PWM

4.1 Sonstiges

5 Echtzeit



Änderungen
vorbehalten

3 TCP/IP in der Praxis

3.6 Netzwerkanalyse

- tcpdump
- wireshark
- ettercap

3 TCP/IP in der Praxis

3.9 Programmierung

Beispiel: Programmierung eines VNC-Servers per Web-Interface

- Grundlagen: RFC 6143
- Praxis: Untersuchung mit Wireshark
- Implementation: JavaScript, WebSockets, [websockify](#), Callbacks

Weitere Beispiele:

- Shell-Skripte: `nc` ([traditional](#) oder [OpenBSD](#))
- C: Sockets, [select\(\)](#)
- C++: Callbacks

4 Bus-Systeme

4.1 Was sind Bus-Systeme?

Ein Bus ist ein System zur Datenübertragung zwischen mehreren Teilnehmern über einen gemeinsamen Übertragungsweg. Findet eine Datenübertragung zwischen zwei Teilnehmern statt, so müssen die übrigen Teilnehmer schweigen, da sie sich sonst gegenseitig stören würden. Umgangssprachlich werden mitunter – oft aus historischen Gründen – auch Datenübertragungssysteme als „Bus“ bezeichnet, die technisch eigentlich eine andere Topologie besitzen.

[https://de.wikipedia.org/wiki/Bus_\(Datenverarbeitung\)](https://de.wikipedia.org/wiki/Bus_(Datenverarbeitung))

Beispiele:

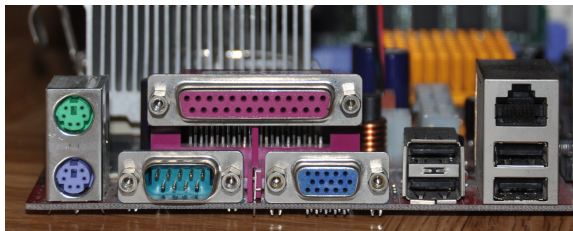
- Computer kommuniziert mit Peripherie
- Computer kommunizieren (direkt) miteinander
- Prozessor kommuniziert mit externem Speicher
- Teile eines Prozessors kommunizieren miteinander

4 Bus-Systeme

4.1 Was sind Bus-Systeme?

Standard-Personal-Computer:

- Einsteckkarten: PCI (und Vorgänger)
- Festplatten: SATA (und Vorgänger)
- USB, FireWire, ...
- Ethernet, CAN-Bus, ...
- WLAN, BlueTooth, IR, ...
- PS/2, RS-232, Centronics



4 Bus-Systeme

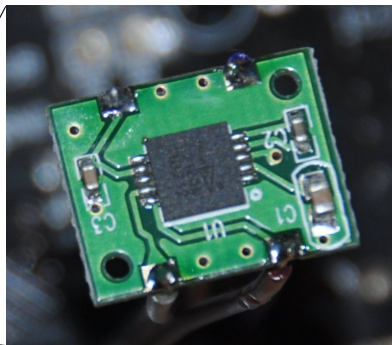
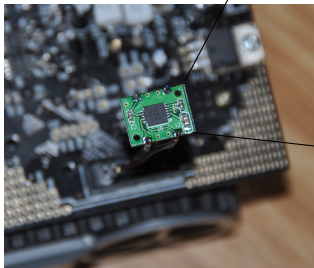
4.1 Was sind Bus-Systeme?

Standard-Personal-Computer:

- Einsteckkarten: PCI (und Vorgänger)
- Festplatten: SATA (und Vorgänger)
- USB, FireWire, ...
- Ethernet, CAN-Bus, ...
- WLAN, BlueTooth, IR, ...
- PS/2, RS-232, Centronics

Minimal-Hardware:

- RS-232
- I²C (TWI)
- SPI



4 Bus-Systeme

4.1 Was sind Bus-Systeme?

<i>seriell</i>	jedes Bit einzeln übertragen
<i>parallel</i>	mehrere Bits gleichzeitig
<i>synchron</i>	Abgleich über Steuerleitung: <i>Takt</i>
<i>asynchron</i>	Abgleich über Zeitvereinbarungen
<i>Punkt-zu-Punkt</i>	genau zwei Teilnehmer
<i>busfähig</i>	mehrere Teilnehmer, mit <i>Adressierung</i>

- I²C: seriell, synchron, mit Adressierung
- RS-232: seriell, asynchron, Punkt-zu-Punkt
- RS-485, USB, CAN: seriell, asynchron, mit Adressierung
- SPI: seriell, synchron, Punkt-zu-Punkt oder mit Adressierung

4 Bus-Systeme

4.2 RS-232

seriell

- *TX*: 1 Leitung für Daten
- *RX*: ggf. 1 Leitung für Daten in der anderen Richtung
- *GND*: gemeinsame *Masse*
- evtl. zusätzliche Steuerleitungen

asynchron

- *keine* Taktleitung für Abgleich, wann Daten anliegen
- Stattdessen: Abgleich über Zeitvereinbarungen

→ Jeder Teilnehmer braucht eine eigene Zeitbasis.

Punkt-zu-Punkt

- nur 2 Teilnehmer vorgesehen

4.2 RS-232

Synchronisation

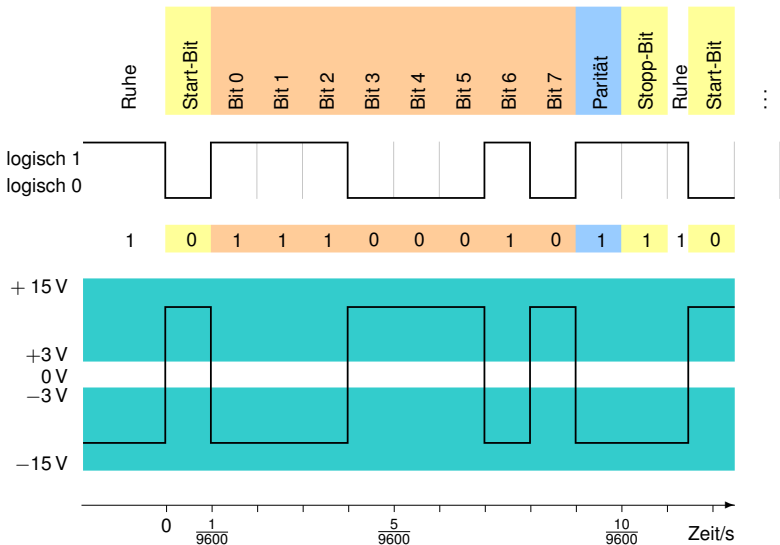
Daten

Check

9600 Baud, 8 Daten-Bits, ungerade Parität, 1 Stopp-Bit

Beispiel-Daten: ASCII „G“ = 71 = 0100 0111 binär

Übertragung der Daten von rechts (Bit 0) nach links (Bit 7)



4 Bus-Systeme

4.3 I²C (TWI)

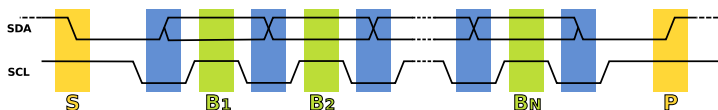
I²C = Inter-IC; TWI = Two-Wire-Interface

seriell

- *SDA*: 1 Leitung für Daten (in beiden Richtungen)
- *SCL*: Taktleitung (Clock)
- *GND*: gemeinsame Masse
- evtl. *VCC*: Stromversorgung für Peripheriegerät

synchron

- Abgleich über Taktleitung



busfähig

- *Master* initiiert Kommunikation und steuert Taktleitung
- erstes gesendetes Byte: *Adresse* des Teilnehmers
- 2 Adressen pro Teilnehmer: Lesen/Schreiben

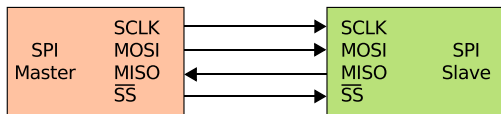
4 Bus-Systeme

4.4 SPI

Serial Peripheral Interface

seriell

- *MOSI*: Master Out, Slave In
- *MISO*: Master In, Slave Out
- *SCLK*: Taktleitung (Clock)
- \overline{SS} : Slave Select (invertiert)
- *GND*: gemeinsame Masse
- evtl. *VCC*: Stromversorgung für Peripheriegerät



synchron

- Abgleich über Taktleitung

busfähig

- *Master* initiiert Kommunikation und steuert Taktleitung
- *Slave* wird über *Slave Select* ausgewählt

4 Bus-Systeme

4.4 SPI

Serial Peripheral Interface

seriell

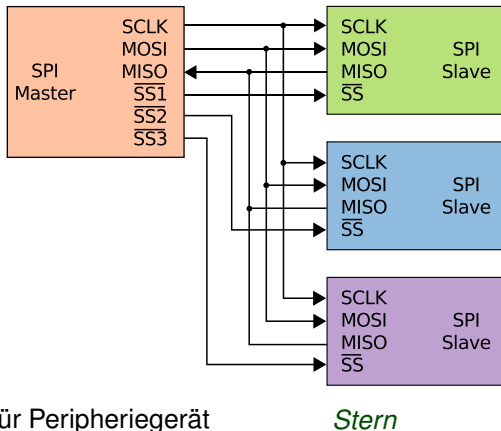
- *MOSI*: Master Out, Slave In
- *MISO*: Master In, Slave Out
- *SCLK*: Taktleitung (Clock)
- \overline{SS} : Slave Select (invertiert)
- *GND*: gemeinsame Masse
- evtl. *VCC*: Stromversorgung für Peripheriegerät

synchron

- Abgleich über Taktleitung

busfähig

- *Master* initiiert Kommunikation und steuert Taktleitung
- *Slave* wird über *Slave Select* ausgewählt



4 Bus-Systeme

4.4 SPI

Serial Peripheral Interface

seriell

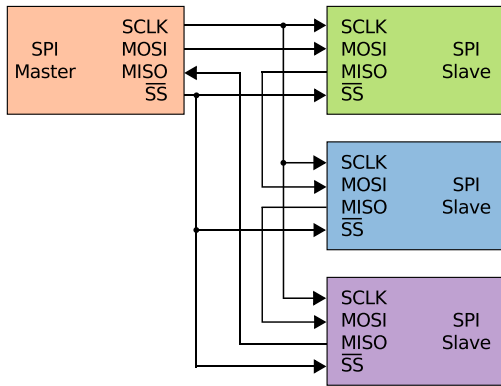
- *MOSI*: Master Out, Slave In
- *MISO*: Master In, Slave Out
- *SCLK*: Taktleitung (Clock)
- \overline{SS} : Slave Select (invertiert)
- *GND*: gemeinsame Masse
- evtl. *VCC*: Stromversorgung für Peripheriegerät

synchron

- Abgleich über Taktleitung

busfähig

- *Master* initiiert Kommunikation und steuert Taktleitung
- *Slave* wird über *Slave Select* ausgewählt



*Kaskade
Daisy Chain*

4 Bus-Systeme

4.4 SPI

Serial Peripheral Interface

seriell

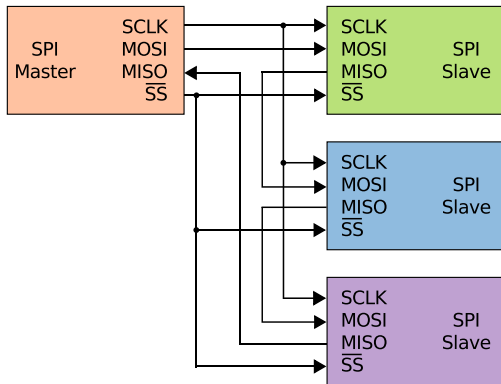
- *MOSI*: Master Out, Slave In
- *MISO*: Master In, Slave Out
- *SCLK*: Taktleitung (Clock)
- \overline{SS} : Slave Select (invertiert)
- *GND*: gemeinsame Masse
- evtl. *VCC*: Stromversorgung für Peripheriegerät

synchron

- Abgleich über Taktleitung

busfähig

- *Master* initiiert Kommunikation und steuert Taktleitung
- *Slave* wird über *Slave Select* ausgewählt



*Kaskade
Daisy Chain*

Slave gibt MOSI-Input um 1 Takt verzögert an MISO aus \rightarrow Master setzt „im richtigen Moment“ \overline{SS}

4 Bus-Systeme

4.5 PWM

Pulsweitenmodulation – *pulse-width modulation*

- Steuerung von Motoren
- Nutzung als allgemeines Protokoll zur Übertragung analoger Werte

4 Bus-Systeme

4.6 Sonstiges

Matrix-Schaltung

- möglichst viele Aktoren/Sensoren
über möglichst wenige digitals Inputs abfragen
bzw. über möglichst wenige digitale Outputs steuern
- Beispiele: LED-Felder, Tastaturen

4 Bus-Systeme

4.6 Sonstiges

Matrix-Schaltung

- möglichst viele Aktoren/Sensoren
über möglichst wenige digital Inputs abfragen
bzw. über möglichst wenige digitale Outputs steuern
- Beispiele: LED-Felder, Tastaturen

R/2R-Netzwerk

- möglichst viele digitale Inputs
über einen einzigen analogen Input abfragen
- Beispiele: Tastaturen

Eingebettete Systeme

<https://gitlab.cvh-server.de/pgerwinski/es>

1 Einführung

2 Einführung in Unix

3 TCP/IP in der Praxis

4 Bus-Systeme

4.1 Was sind Bus-Systeme?

4.1 RS-232

4.1 I²C (TWI)

4.1 SPI

4.1 PWM

4.1 Sonstiges

5 Echtzeit

5.1 Was ist Echtzeit?

5.2 Echtzeitprogrammierung

5.3 Multitasking

5.4 Ressourcen

5.5 Prioritäten



Änderungen
vorbehalten