IoT Elektronik

Wie verwende ich Mikrocontroller Boards?

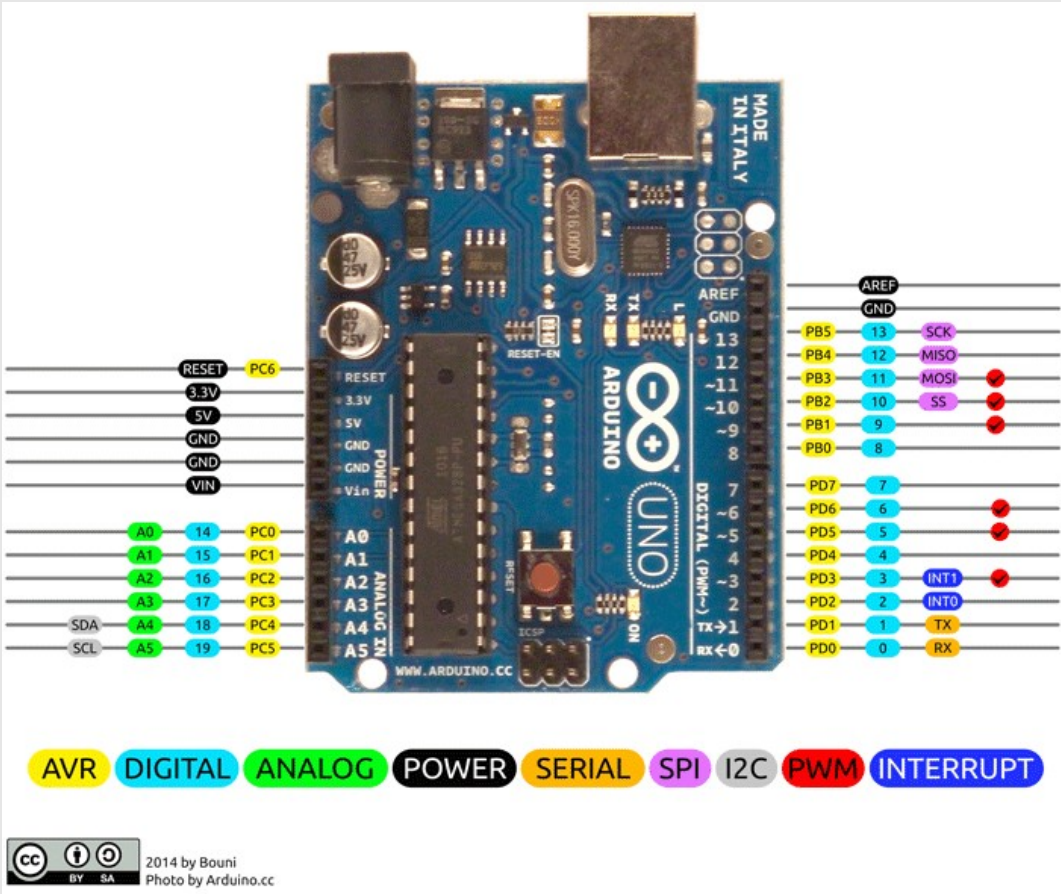
Chris C (chca@fsfe.org)

Inhalt

- Mikrocontroller-Boards: ESP8266, ESP32, Arduino, pyboard, ... $\rightarrow$ Anschlussmöglichkeiten
- Was ist ein Pin und wie funktioniert er?
- Sensoren: digital (Taster), analog (resistiv, kapazitiv, induktiv)
- Sensoren mit Bus: SPI, I2C
- Aktoren: Schalten von Lasten (LED, Relais, Geräte, ...)
- Leitungen: Entfernte Sensoren / Aktoren, Kommunikation (RS232, RS485)

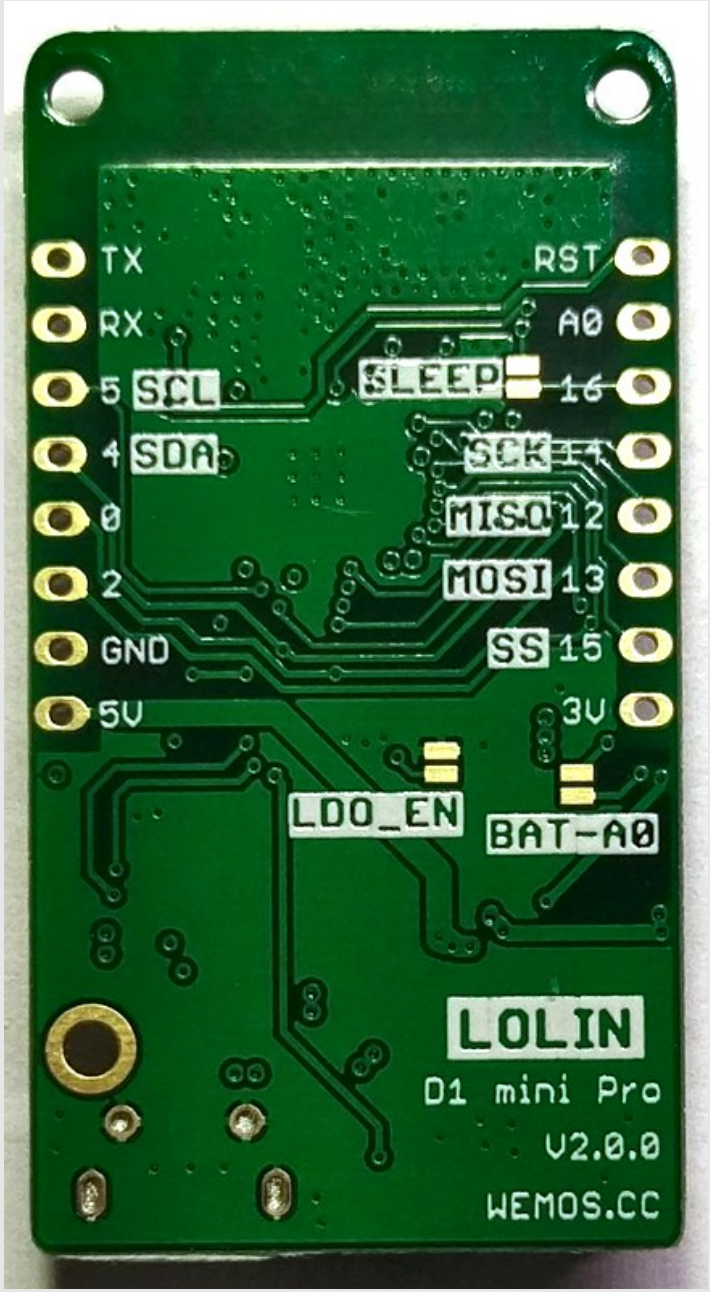
Arduino

- Anschlussmöglichkeiten:
  - 12 Digital I/O (Pin 2-13)
  - 2 eingeschränkt (Pin 0,1 $\rightarrow$ Tx/Rx)
  - 5 Analog In oder Digital I/O
- Außer Tx/Rx keine Einschränkungen
- Einige Pins mit Sonderfunktionen
  - PWM
  - I2C
  - SPI
  - Ext. Interrupts



ESP8266

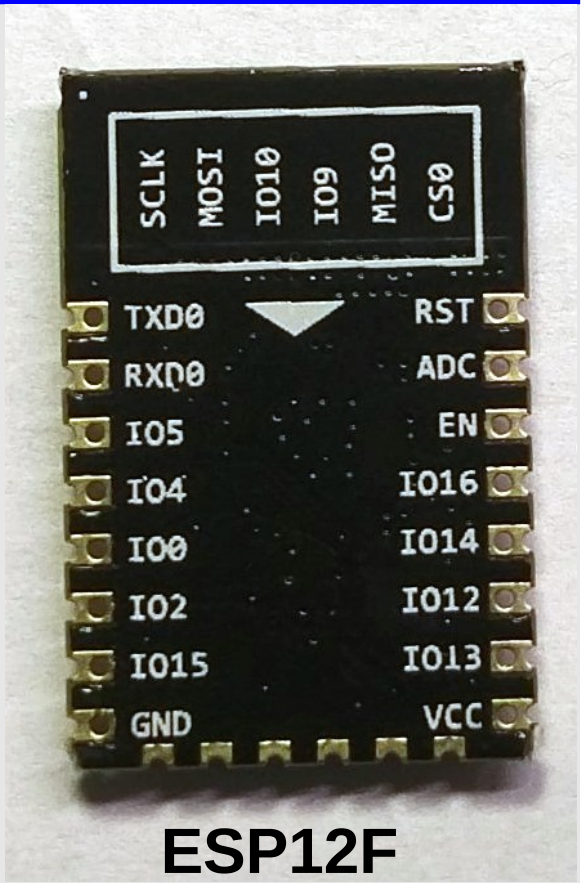
- Sehr günstig (beim Ali ab $1.20, Wemos D1 mini ca. $3)
- Relativ leistungsstark: Tensilica L106 32-bit RISC, bis 160 MHz
- mit WLAN (2.4 Ghz, b/g/n)
- Anschlussmöglichkeiten:
  - Bis zu 12 Digital I/O
  - 1 Analog In
  - 1 SPI
- Leider viele Pins nicht nutzbar



ESP8266 Pins

- Nur 5 Pins wirklich gut nutzbar
- Rest mit vielen Fallstricken:

GPIO	Funktion	Bemerkung
0	Boot Mode	Low: Bootloader, High: Flash Boot
1	UART TX	Bootloader, TTL RS232
2	Boot Mode	Built-in LED (active low), hat pull-up
3	UART RX	Bootloader, TTL RS232
4	keine	Uneingeschränkt nutzbar
5	keine	Uneingeschränkt nutzbar
6-11	SDIO	Flash, gar nicht nutzbar
12	SPI MISO	SPI, während Boot für ~100 ms high
13	SPI MOSI	SPI, während Boot für ~100 ms high
14	SPI SCK	SPI, während Boot für ~100 ms high
15	SPI CS	Muss beim Booten high sein
16	Deep sleep wakeup	Normalerweise mit Reset Pin verbunden, um aufwachen zu können



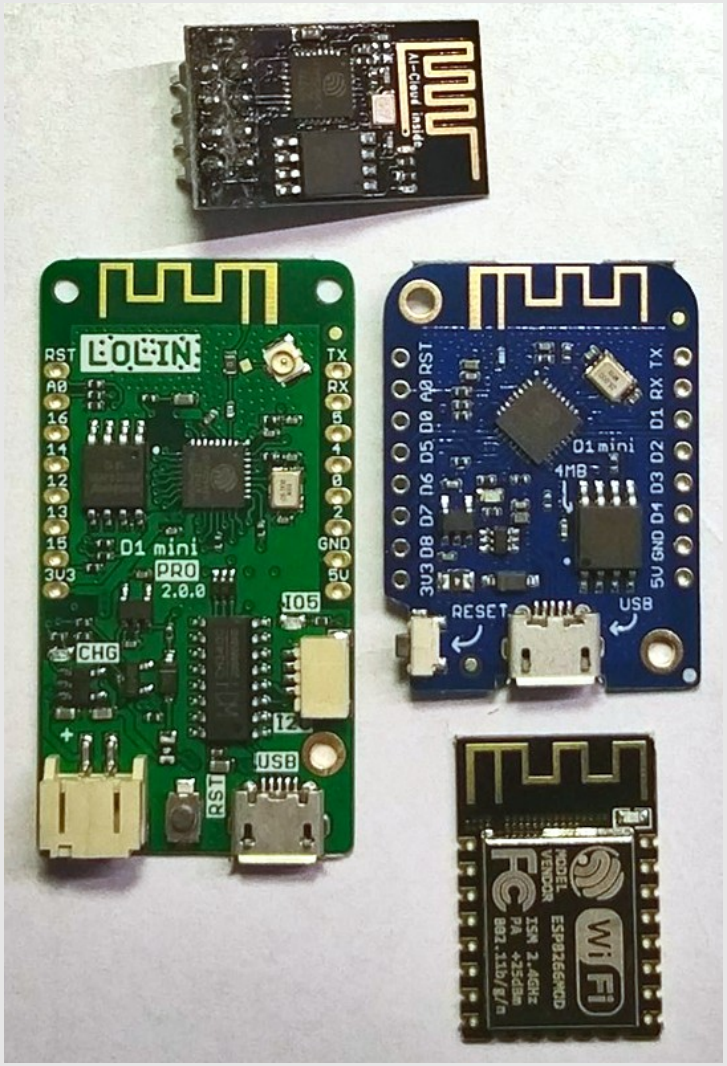
ESP12F

Bootmodes über Pin 0,2,15:

15	0	2	Boot
L	L	H	UART
L	H	H	Flash
H	X	X	SDIO

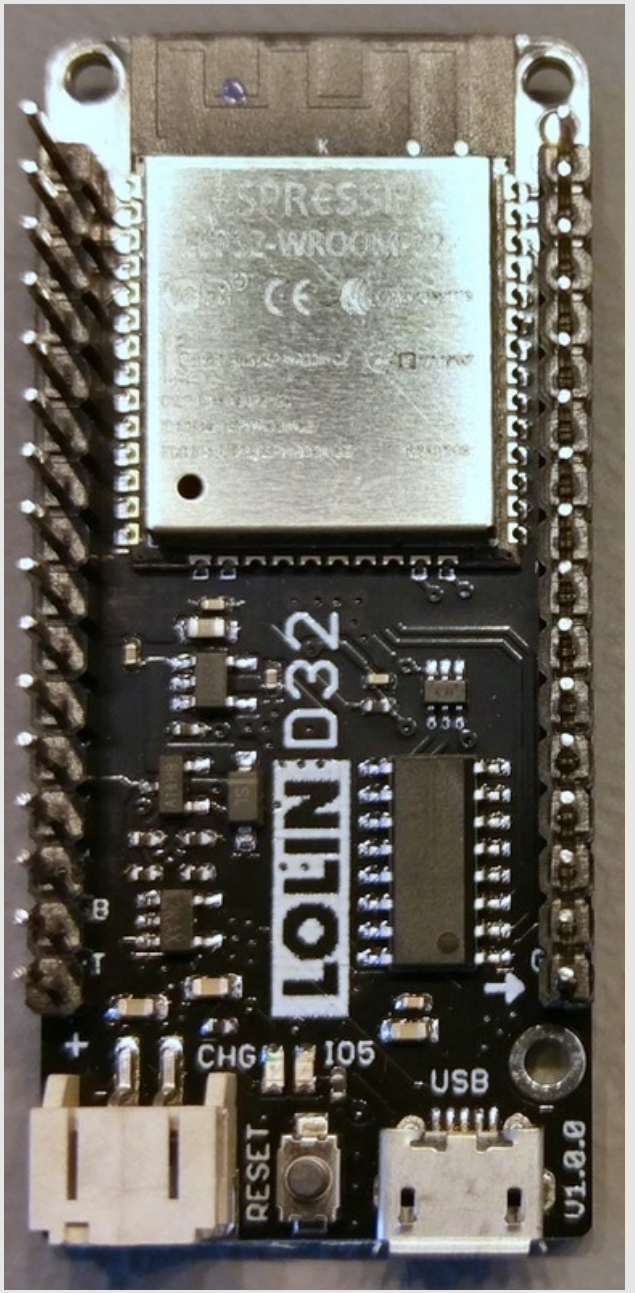
ESP8266 Pins

- Bootmodes sollten bei Modulen (ESP12F, usw.) extern fest eingestellt werden
- Boards (NodeMCU, Wemos, etc.) sind meist schon entsprechend eingestellt, Boot über UART wird automatisch über USB-UART Bridge bedient (RTS/CTS)



ESP32

- Immer noch sehr günstig
  (beim Ali ab ~$4 mit USB,
  nur Modul ab ~$2.80)
- Sehr leistungsstark:
  Tensilica LX6, bis 240 MHz
- mit WLAN+BT (2.4 Ghz, b/g/n)
- Anschlussmöglichkeiten:
  - ADC (viele), 2xDAC
  - 2xSPI, I2C(software)
  - 10x Touch Sensor
  - 19 uneingeschränkt nutzbare I/Os



ESP32 Pins (DevKitC)

Sonstiges	ADC	Digital	Pin
			3.3V
			EN
Sensor C	ADC0	Nur Input	GPIO36
Sensor C	ADC3	Nur Input	GPIO39
	ADC6	Nur Input	GPIO34
	ADC7	Nur Input	GPIO35
TOUCH9	ADC4	Nur I/O	GPIO32
TOUCH8	ADC5	Nur I/O	GPIO33
DAC1	ADC18	Nur I/O	GPIO25
DAC2	ADC19	Nur I/O	GPIO26
TOUCH7	ADC17	Nur I/O	GPIO27
TOUCH6	ADC16	H SCK	GPIO14
TOUCH5	ADC15	H MISO	GPIO12
			GND
TOUCH4	ADC14	H MOSI	GPIO13
		SD2	GPIO9
		SD3	GPIO10
		SDCMD	GPIO11
			5V

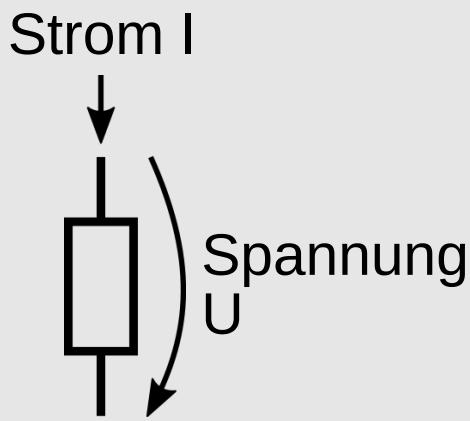
Pin	Digital	ADC	Sonstiges
GND			
GPIO23	V MOSI		
GPIO22	Nur I/O		U0 RTS
GPIO1	U0 TX		
GPIO3	U0 RX		
GPIO21			
GND			
GPIO19	V MISO		U0 CTS
GPIO18	V SCK		
GPIO5	V SS		
GPIO17	U2 TX		
GPIO16	U2 RX		
GPIO4		ADC10	
GPIO0	BOOT	ADC11	
GPIO2		ADC12	
GPIO15	H SS	ADC13	
GPIO8	SD1		
GPIO7	SD0		
GPIO6	SDCK		

Grundlagen: Widerstand

- Spannung über Widerstand hat Strom zur Folge:

$$U = R \cdot I$$

[U]: Volt (V)
[R]: Ohm (Ω)
[I]: Ampere (A)



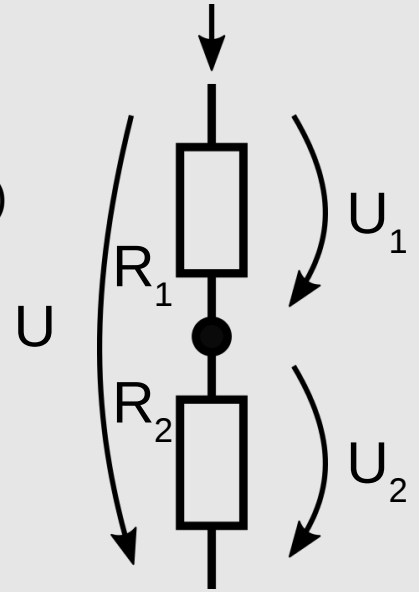
- Spannung bzw. Strom über/durch Widerstand führt zu Leistungsumsatz:

$$P = U \cdot I = R \cdot I^2 = \frac{U^2}{R}$$

[P]: Watt (W)

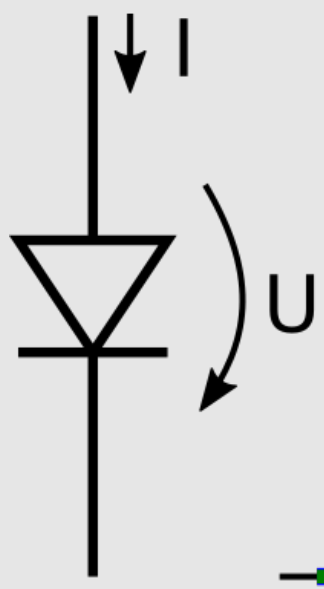
- Spannungsteiler:

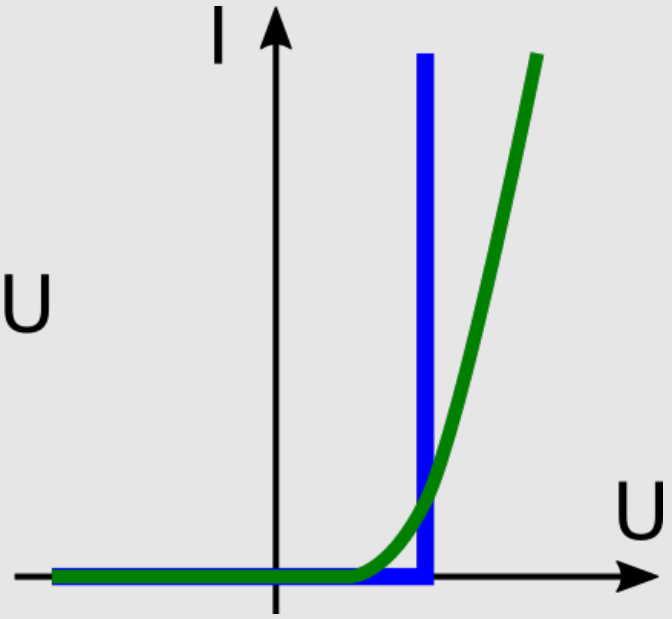
$$\frac{U_1}{R_1} = \frac{U_2}{R_2} \qquad U_2 = \frac{R_2}{R_1 + R_2} \cdot U$$



Grundlagen: Diode

- Leitet Strom nur in eine Richtung (Pfeilrichtung)
- Mindestspannung, ab der Strom fließt, danach starker Anstieg
  - Für Rechungen: Konstante Schwellspannung sobald Diode leitend
  - Schwellspannung meist 0.3 V bis 0.7 V
- Spannungsfestigkeit in die andere Richtung ist jedoch begrenzt ($\rightarrow$ Datenblatt)



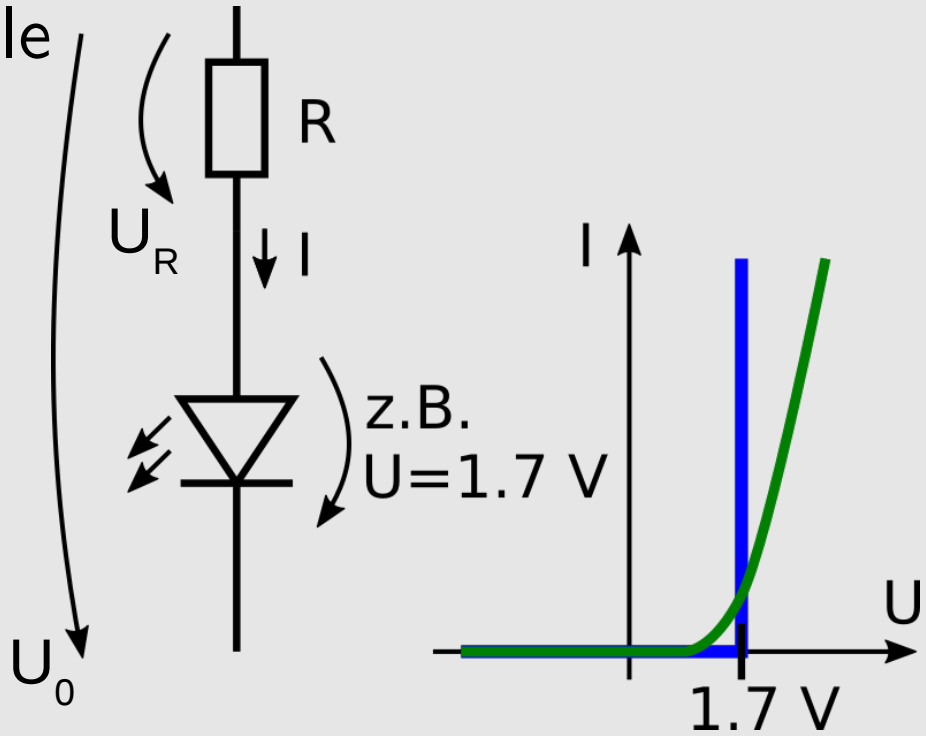


Grundlagen: LED

- Diode erzeugt Licht wenn Strom fließt
- Schwellspannung höher (meist 1.3 V ... 4 V)
- Spannungsfestigkeit in andere Richtung geringer
- Strombelastbarkeit stark begrenzt
  - Ideal: Betrieb mit Stromquelle
  - Kompromiss: Vorwiderstand
- Berechnung:

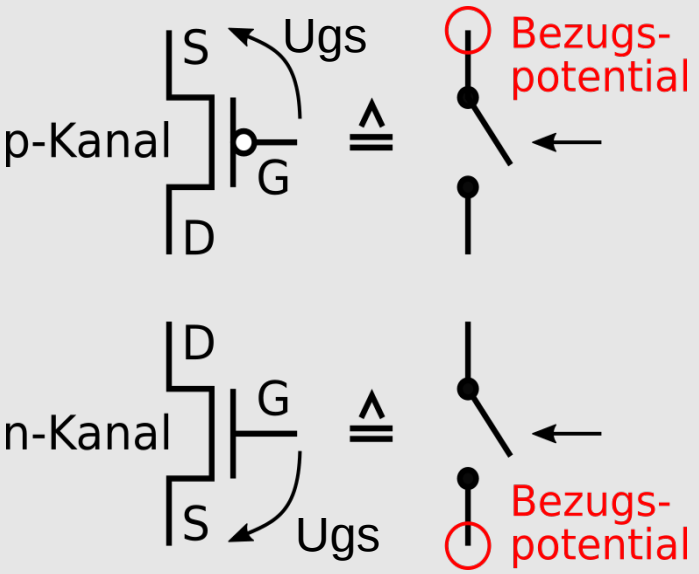
$$U_R = U_0 - U = R \cdot I$$

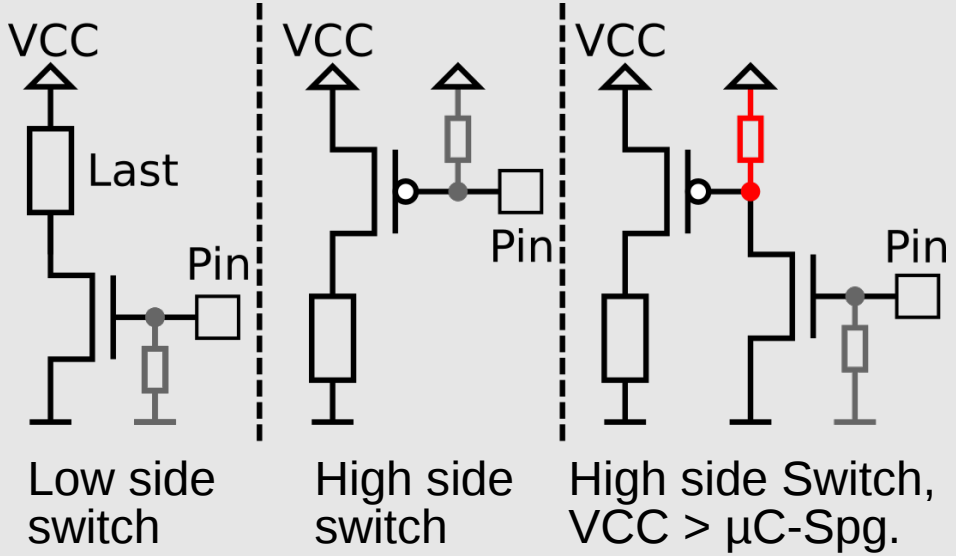
$$R = \frac{U_0 - U}{I}$$



Grundlagen: Transistor

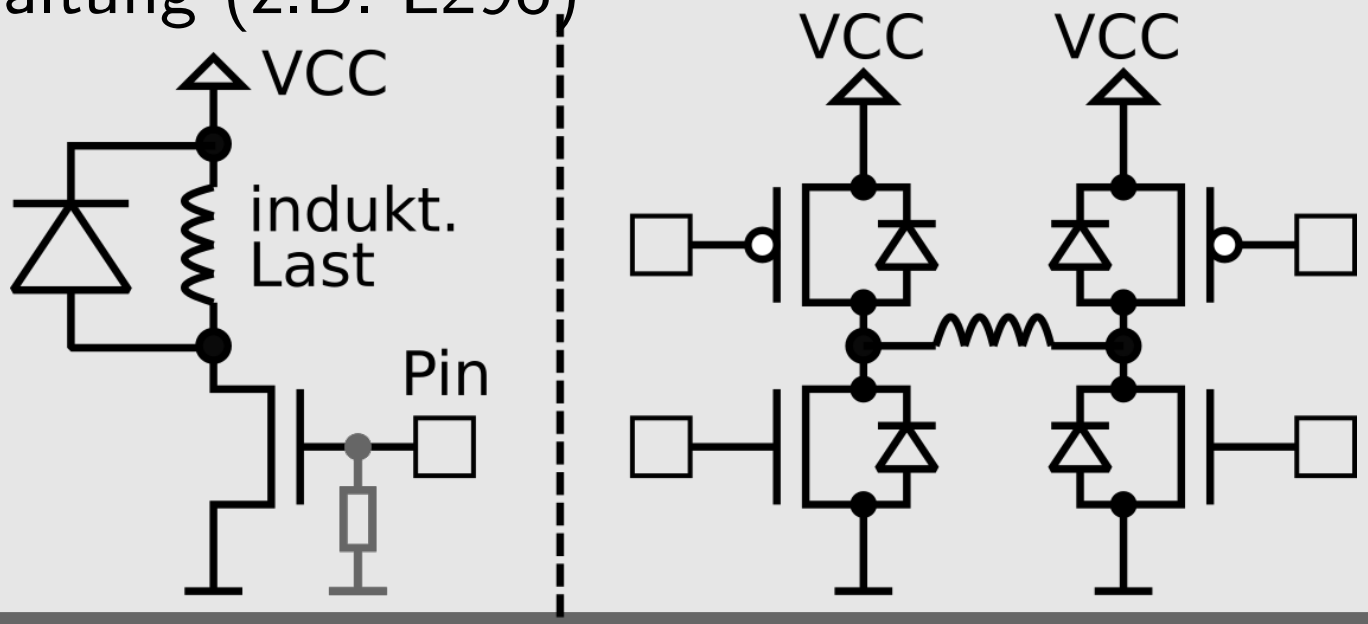
- Hier: FET (Feldeffekttransistor)
- Prinzip (vereinfacht): Spannungsgesteuerter Schalter
- Schalter hat Innenwiderstand („Ron“)
- Schließt, wenn Spannung Ugs Schwelle überschreitet
- 2 wesentliche Varianten:
  - P-Kanal: High-side Switch
  - N-Kanal: Low-side Sw.





Grundlagen: Schalter

- Induktive Lasten (z.B. Motor, Relais): „Freilaufdiode“ wichtig
- Für Motoren: https://www.mikrocontroller.net/articles/Motoransteuerung_mit_PWM
  - Kondensator (10 … 100 nF), Keramik-/ Folienkondensator zwischen VCC und Masse
  - Für Motoren: H-Brücke, am besten als integrierte Schaltung (z.B. L298)

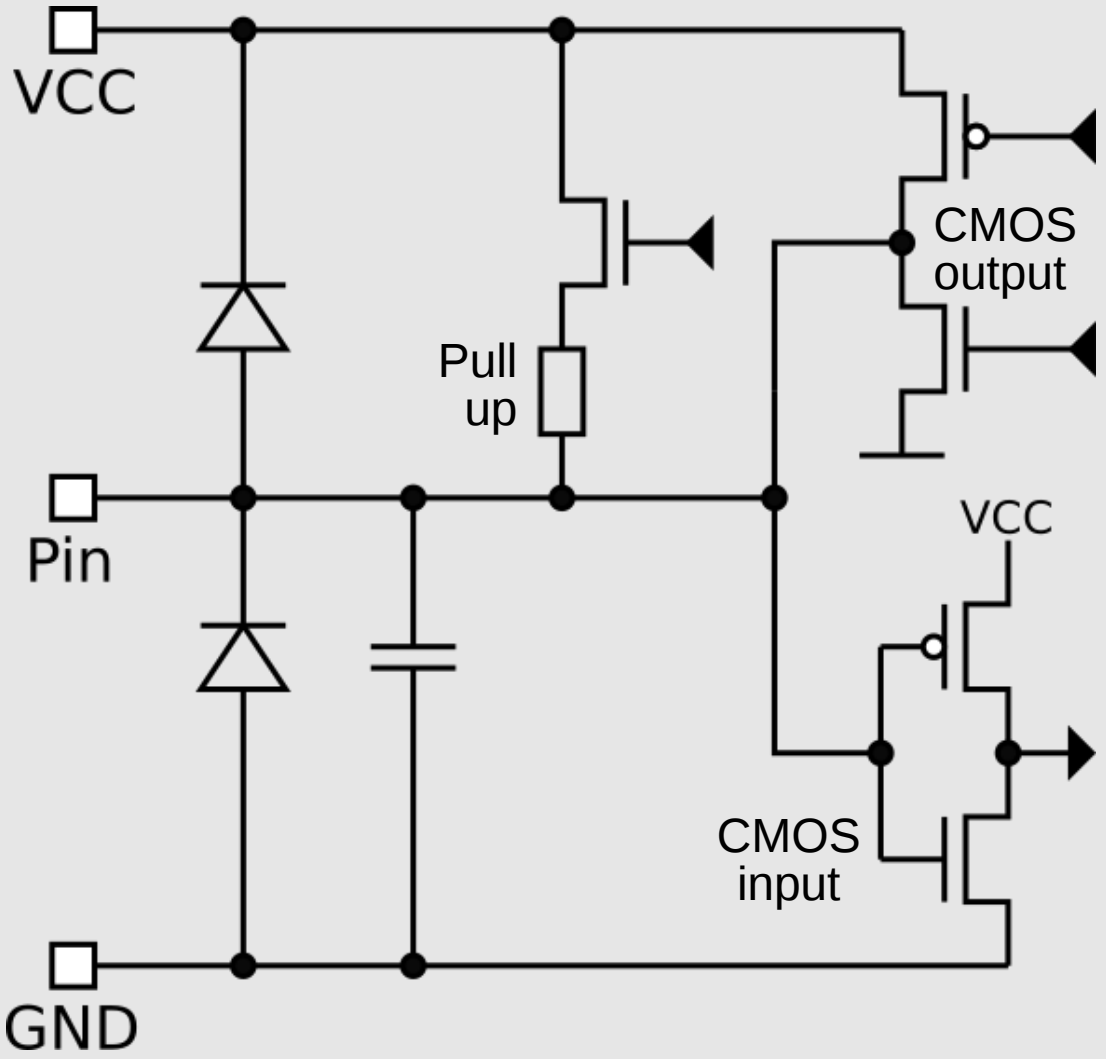


Grundlagen: Schalter

- Kapazitive Lasten: Hoher Strom beim Einschalten!
  - Beispiel: Einschalten eines Moduls mit vielen Kondensatoren zur Spannungsglättung
  - Tödlich für Mikrocontroller-Pins
- Maximalen Spitzenstrom im Datenblatt beachten
- Im Zweifel Transistor großzügiger auslegen
- Maßnahme zur Stromreduzierung:
  - NTC („Heißleiter“) bei ausreichend hohen Strömen
  - PWM mit steigendem Duty Cycle, sofern Spitzenstrom handhabbar
  - Filter (mit Drossel), sehr vorsichtig auslegen!

Digital I/O Pins

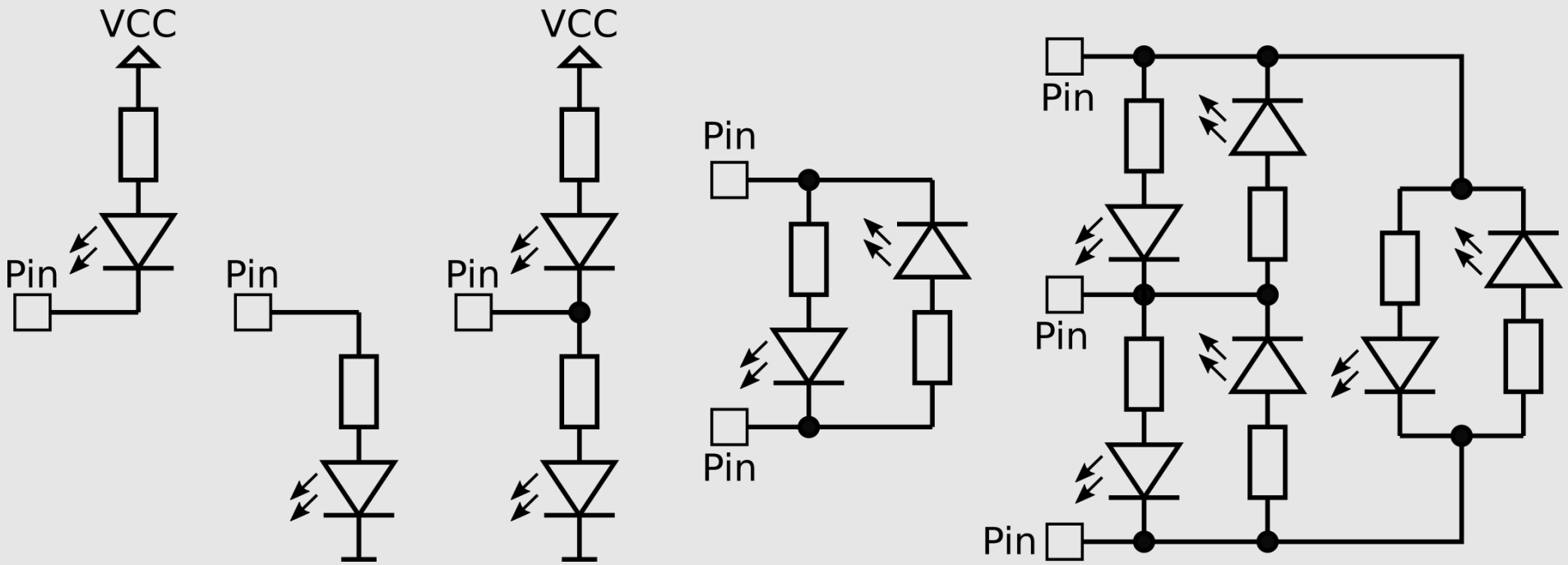
- Verschiedene Betriebsarten
  - Input (high-Z)
  - Output (push-pull)
  - Open drain/collector
  - Pull up/down
- Belastbarkeit: meist 10 ... 20 mA, kann abweichen für sink/source → Datenblatt!



LEDs direkt am I/O Pin

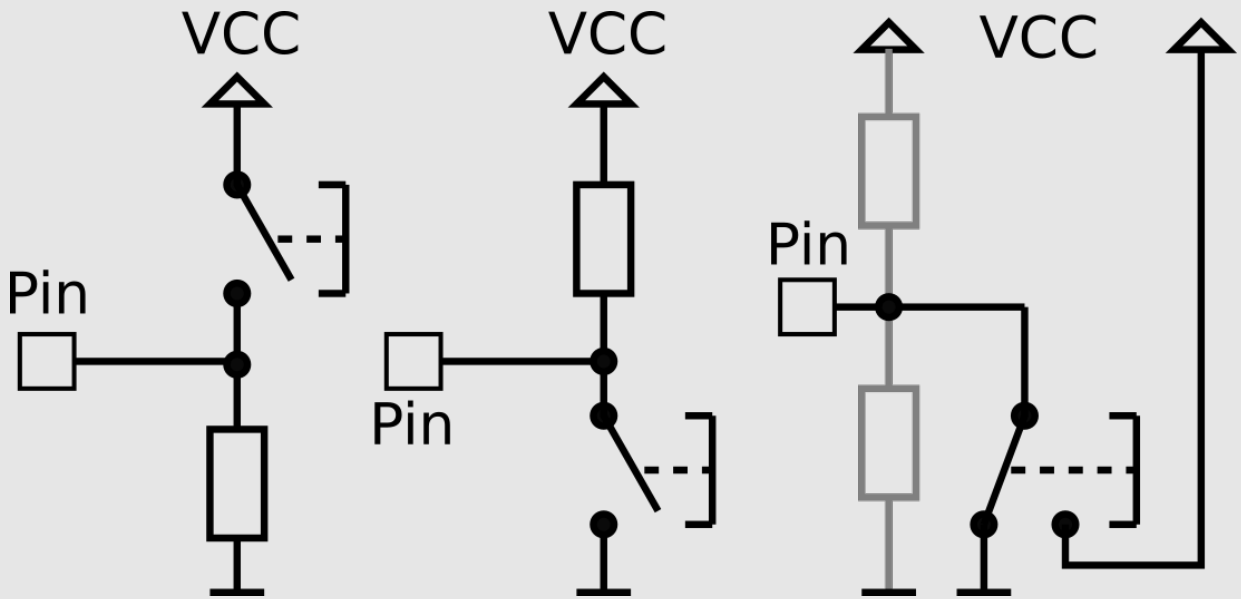
- Übung zum Verständnis: Anschlussmöglichkeiten für LEDs („tri-state logic“):
  - Gegen Masse / gegen VCC
  - Zwischen zwei Pins
  - Zwischen mehreren Pins („Charlieplexing“)

n^2-n LED für n Pins
vs. n^2 für 2n Pins bei Matrix



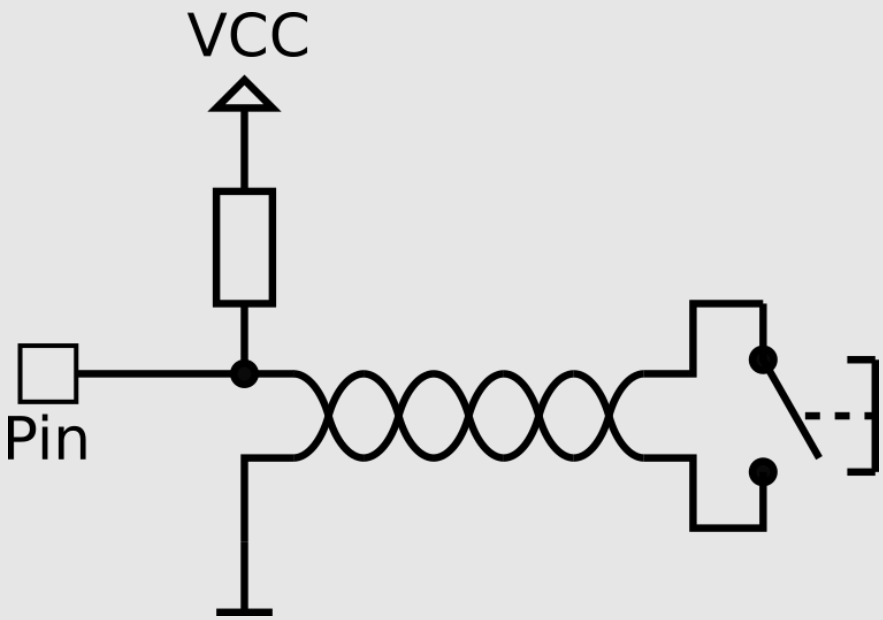
Eingänge: Schalter / Taster

- Es gibt Mono- und Bistabile Öffner, Schließer und Umschalter (monostabil: oft „Taster“ genannt)
- Achtung: Schalter „prellen“ ca. 200 ms lang (schnelle Folge von ein/aus Zuständen)
- Stromfluss = störfester (z.B. 10 mA), v.a. mit Leitungen



Schalter an Leitung

- Leitung hat parasitäre Eigenschaften (Induktivität und Kapazität)
- Verzerrt Signalverlauf beim Schalten
- Reduktion der Störempfindlichkeit:
  - Verdrillte Leitung
  - Pullup möglichst klein (viel Stromfluss)

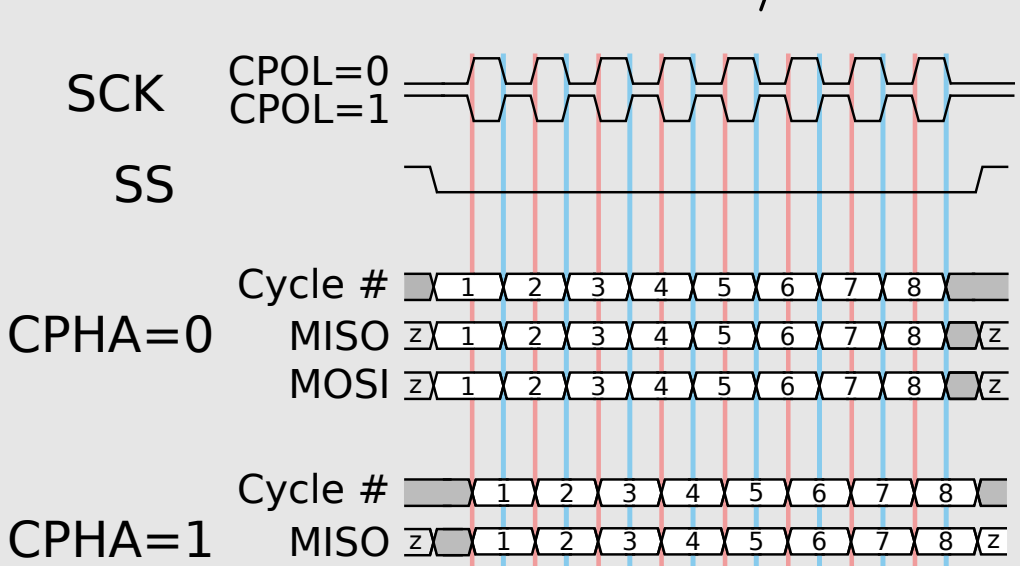


Digitale Sensoren

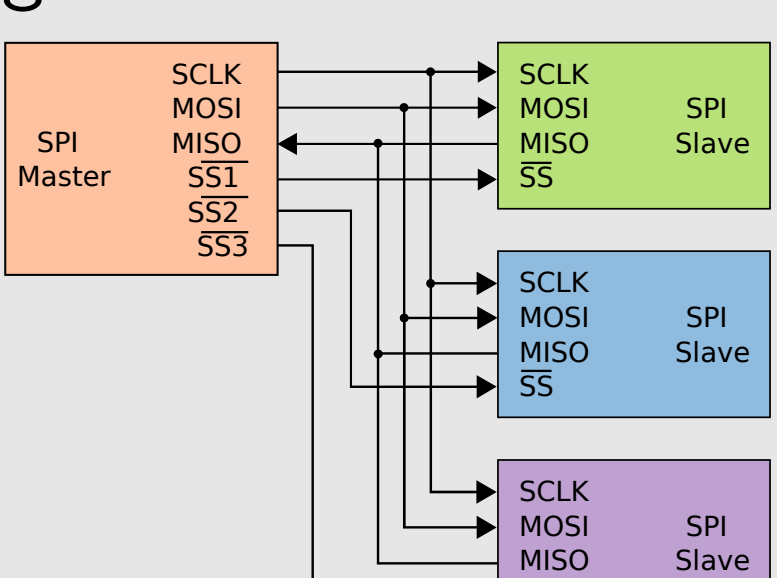
- Einfachste Möglichkeit für Anschluss
- Meist über serielle Schnittstelle (SPI, I2C, UART)
- Zuleitung begrenzt, $<<$ 1m, möglichst dicht an Board anschließen
- Benötigt Spannungsversorgung
  - Ideal: benötigte Spannung = Mikrocontroller Versorgungsspannung
  - Sonst Pegelwandler oder „Schaltungstricks“ notwendig

SPI

- SPI = Serial Peripheral Interface
- Übertragung mit separatem Taktsignal
- Flanken-“Belegung“ konfigurierbar, muss an Slave angepasst werden
- Oft 10 ... 50 Mbit/s oder sogar mehr



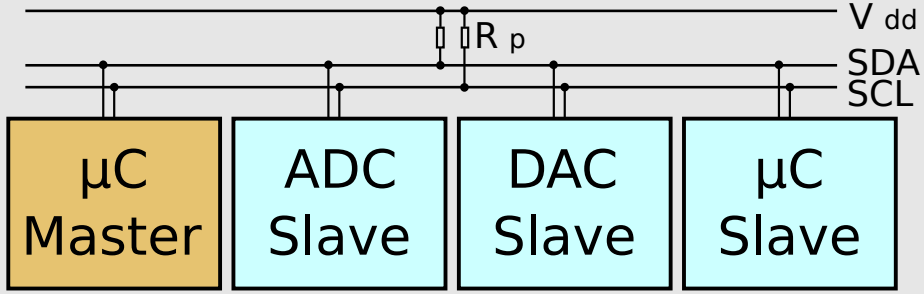
SPI_timing_diagram.svg: en:User:Cburnett derivative work: Jordsan (talk) (https://commons.wikimedia.org/wiki/File:SPI_timing_diagram2.svg), „SPI timing diagram2“, https://creativecommons.org/licenses/by-sa/3.0/legalcode



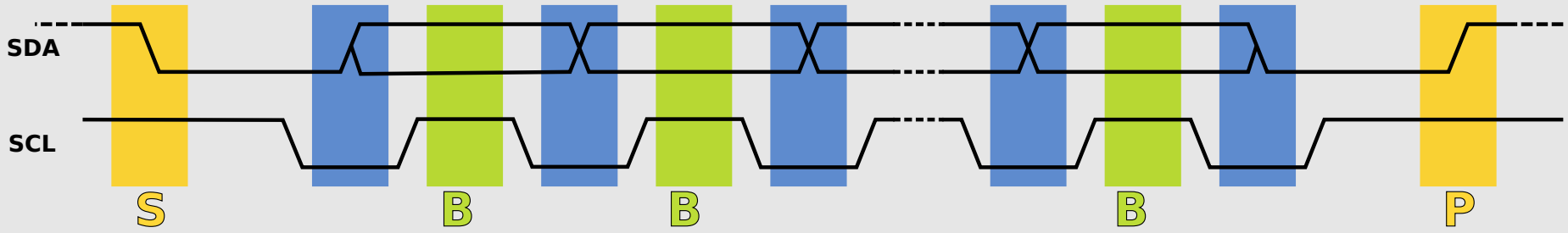
en:User:Cburnett (https://commons.wikimedia.org/wiki/File:SPI_three_slaves.svg), „SPI three slaves“, https://creativecommons.org/licenses/by-sa/3.0/legalcode

I2C

- I²C = Inter-Integrated Circuit
- 2-Draht Bus mit „Wired-AND“ Prinzip
- Verwendet Open Collector / Drain Ausgänge
- Meist 100 kb/s oder 400 kb/s
- Wichtig: Pullups vorsehen
- z.B. 10 kOhm



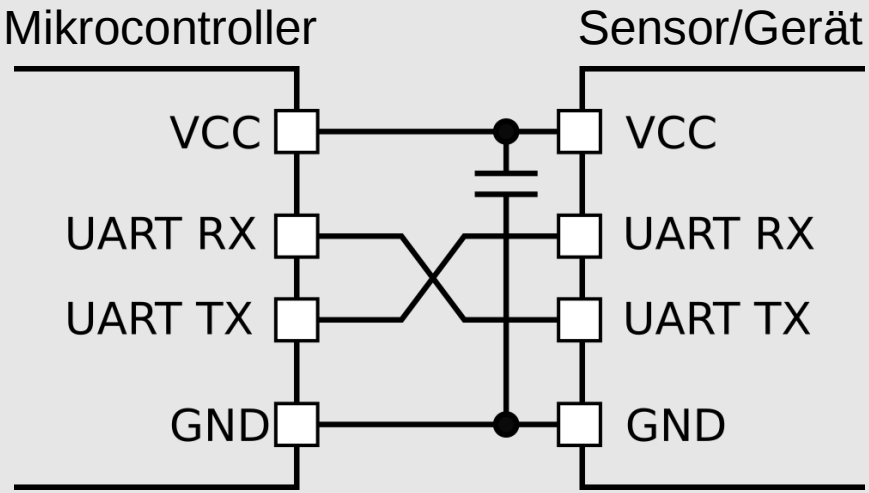
en:user:Cburnett (https://commons.wikimedia.org/wiki/File:I2C.svg), „I2C“, https://creativecommons.org/licenses/by-sa/3.0/legalcode

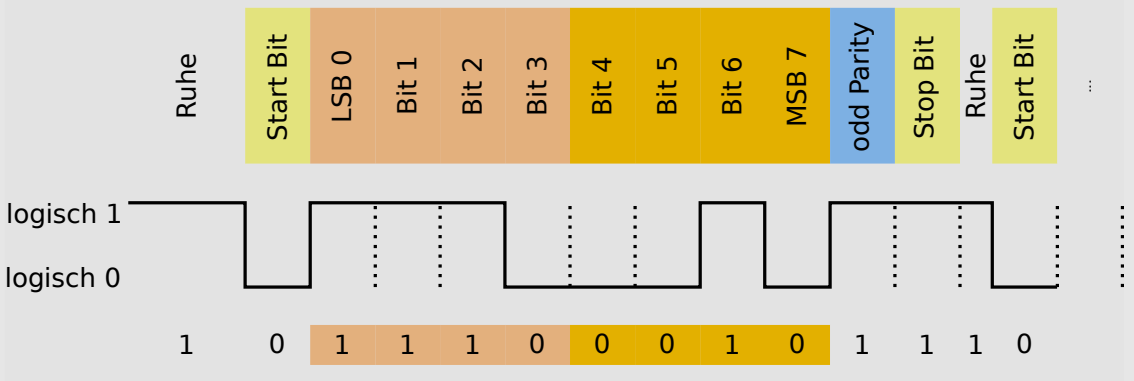


Marcin Floryan (https://commons.wikimedia.org/wiki/File:I2C_data_transfer.svg), „I2C data transfer“, als gemeinfrei gekennzeichnet, Details auf Wikimedia Commons: https://commons.wikimedia.org/wiki/Template:PD-self

UART

- UART = Universal Asynchronous Receiver Transmitter
- Vollduplex mit zwei Leitungen
- Synchronisation über fallende Flanke
- Übertragung in Zeichen je 8 Bit
- Meist 9600 ... 115200 Baud

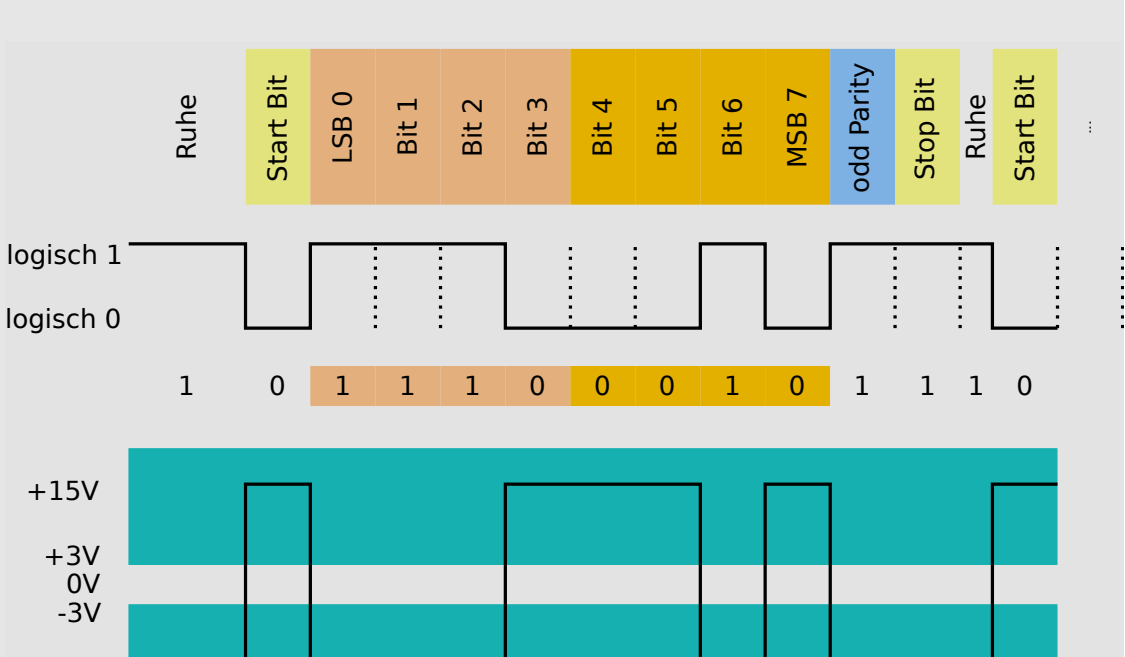




Chris828 (https://commons.wikimedia.org/wiki/File:RS-232_timing.svg), Ausschnitt

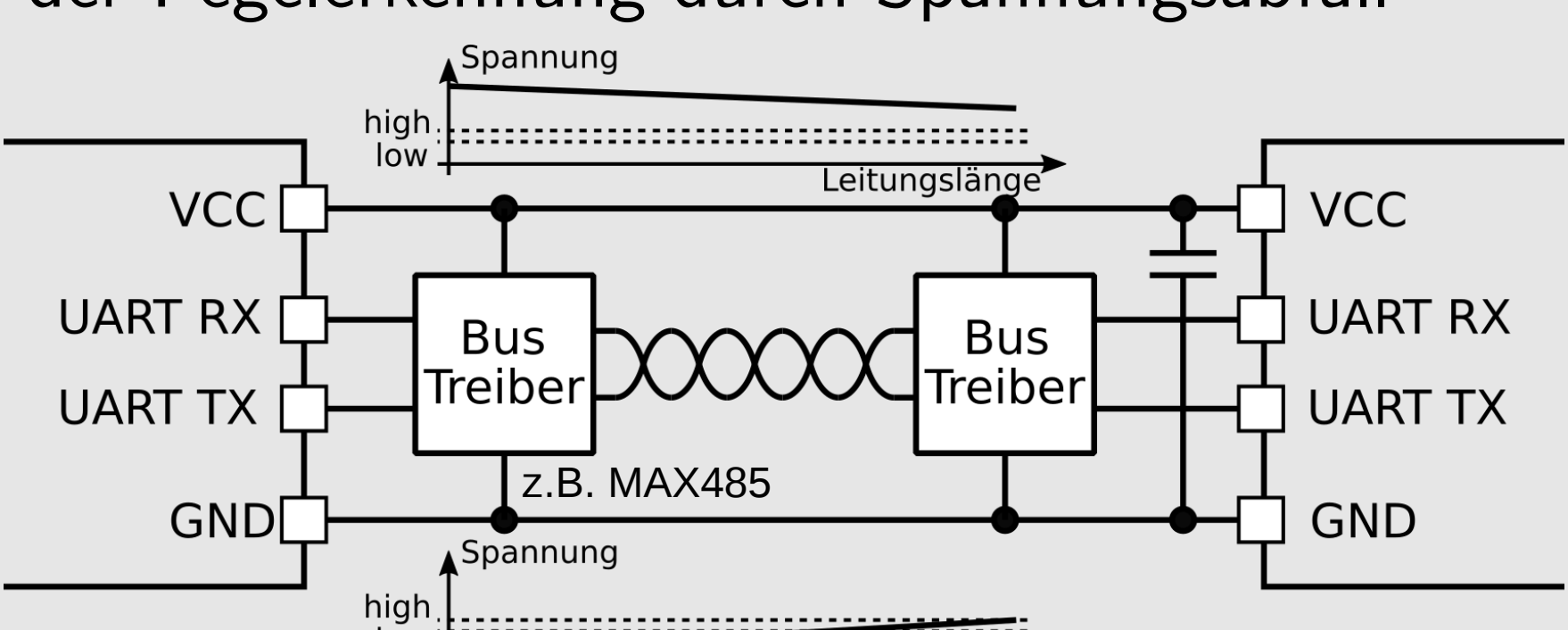
RS232

- Ist im Prinzip ein UART mit zusätzlichen Funktionen
  - z.B. Handshake (RTS, CTS)
- Spannungen höher, bipolar (bis +/- 15 V)
- Standard für Stecker
- Kabel bis etwa 15m
- Längere Kabel haben zu hohen Kapazitäts-belag



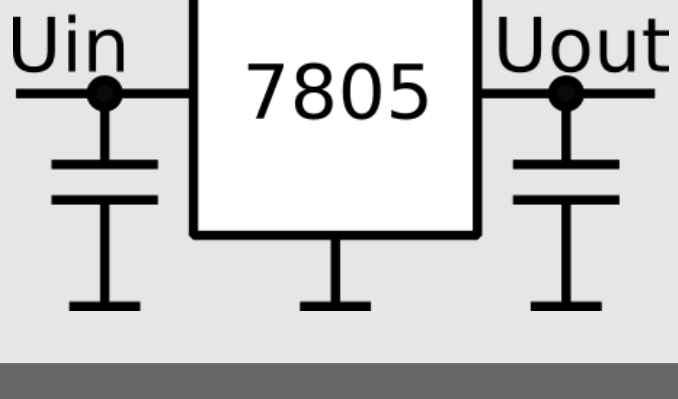
RS485

- Wird häufig im IoT Bereich für Hobby-Anwendungen mit längeren Leitungen eingesetzt (neben CAN), meist Halb-Duplex, auch als Bus
- Differentielle Übertragung meidet Probleme mit der Pegelerkennung durch Spannungsabfall



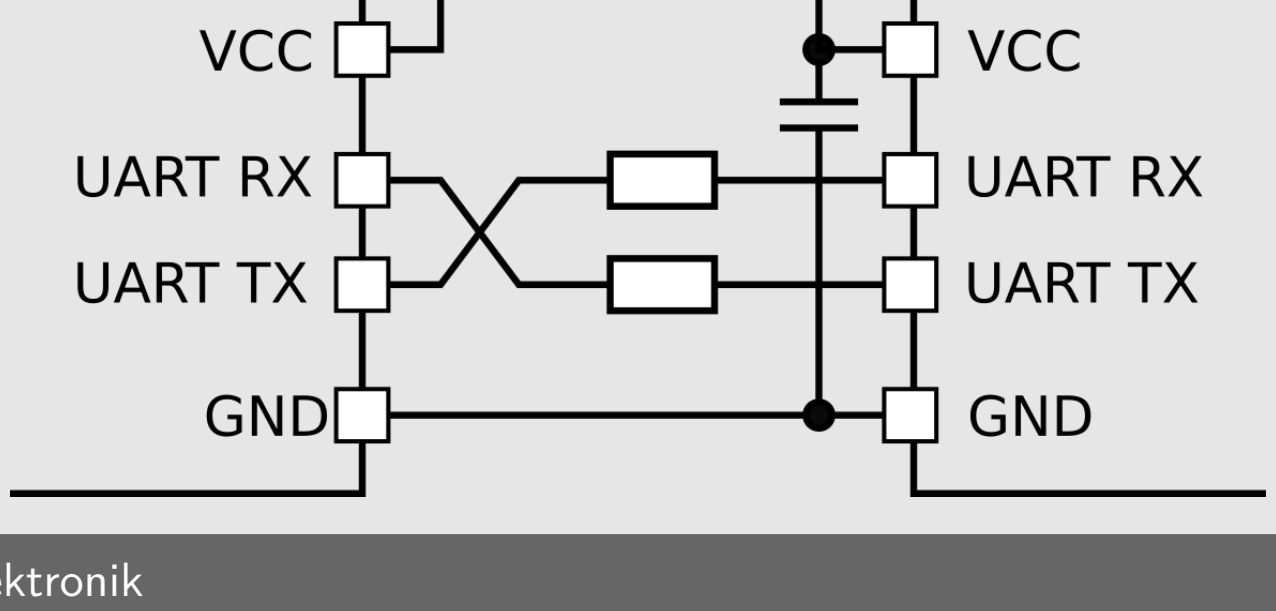
Spannungsversorgung

- Meist entweder Batterie/Akku oder Netzteil
- Wenn Spannung zu hoch: Regler
  - Linearregler (z.B. 7805 oder LM1117 für Netzteil, MCP1703 für Akku): Differenz = Verlust, aber einfach zu verwendet, meist stabil mit 1uF Kondensatoren
  - Schaltregler: bessere Effizienz, Schalten verursacht jedoch Störungen, teurer, höherer Eigenverbrauch („quiescent current“)



Pegelwandlung

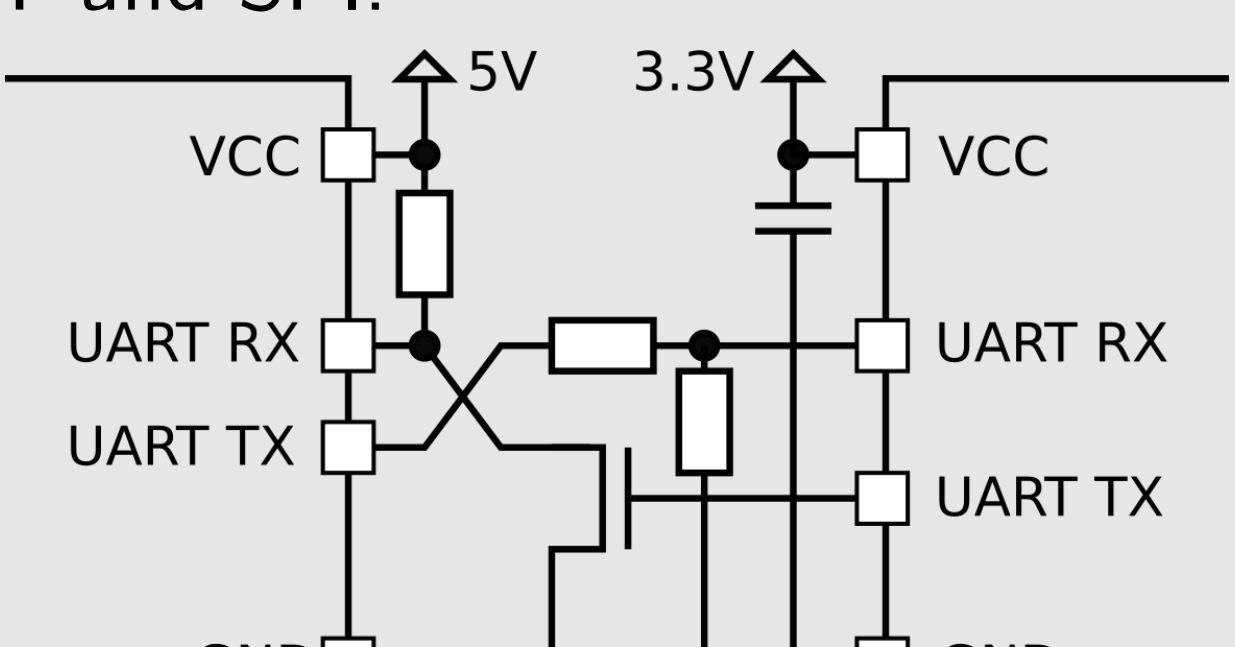
- z.B. 5V Mikrocontroller, 3.3V Sensor
- Problem: 3.3V Sensor würde durch 5V i.d.R. zerstört, Pegelumsetzung nötig
- Quick & Dirty: Falls ESD Dioden beidseitig vorhanden Widerstände einsetzen, die Strom auf zulässiges Maximum begrenzen, Spannung sinkt



Achtung: Hier ist Vorsicht geboten, das geht nicht mit jedem Board! Im Zweifel noch mal nachmessen, ob Dioden vorhanden

Pegelwandlung (II)

- Nächstbessere Variante: Spannungsteiler und Open Drain Stufe
- Bei I2C klappt oft Pullup mit niedrigerer Spannung (hat schon Open Drain)
- UART und SPI:



Bauelemente

- Wie finde ich Bauelemente? Wo gibt es weitere Informationen?
- Gute Ressource: www.mikrocontroller.net
- Standardbauelemente: https://www.mikrocontroller.net/articles/Standardbauelemente
- Auch viele weitere Schaltungskonzepte und Tipps