Computergestützte Experimente und Signalauswertung

PHY.W04UB/UNT.038UB

Hardwarevoraussetzungen

by Jan Enenkel

Übersicht

Mikrocontroller & Prozessoren

GPIO/ADC/DAC

Schnittstellen und Bussysteme

Timing/Polling/Interrupts

Mikrocontroller & Prozessoren

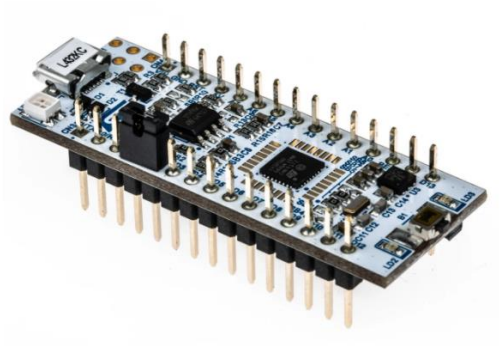
Unterschiede

Mikrocontroller	Prozessoren
Messen, Steuern, Regeln	Rechnen
Wenig Rechenleistung	Sehr viel Rechenleistung
8-16-32 Bit	32-64 Bit
Viele GPIOs	Interfaces
ADCs/DACs/PWM	Viel Speicher
12-100MHz	Hohe Taktraten ~2GHz
Kein Betriebssystem	Linux/MSFT ect.

Entwicklungsplattformen

Mikrokontroller Plattformen

- Hauptprogrammiersprache – „C“
- Board mit Mikrokontroller und jede menge I/Os
  - **Arduino** – ATMEL – ATMega328 – 8 Bit
  - **mBed** – Arm Cortex / NXP LPC1768 – 32 Bit
  - **Nucleo** – ST Micro – STM32 – 32 Bit

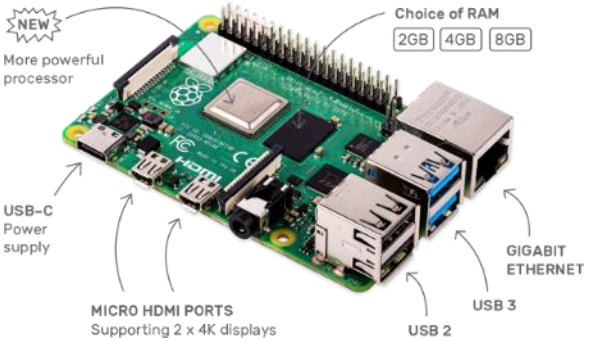


STM32 Nucleo 32 Board
Quelle: rs-components.com

Prozessoren / Entwicklungsplattformen

Programmiersprache: C++, Python, C#, QT, ect.

- Betriebssysteme: MSFT / Linux
- Professionellere Systeme: Intel/Nvidia/AMD
- **Raspberry Pi 4** ‚miniComputer‘ - 1.5 GHz
  - 2x Monitore / 4x USB / WLAN Bluetooth
  - 1x Kameraanschluss – 12MP Kamera
  - 2-8GB RAM
  - SD-Karte für Betriebssystem & Daten

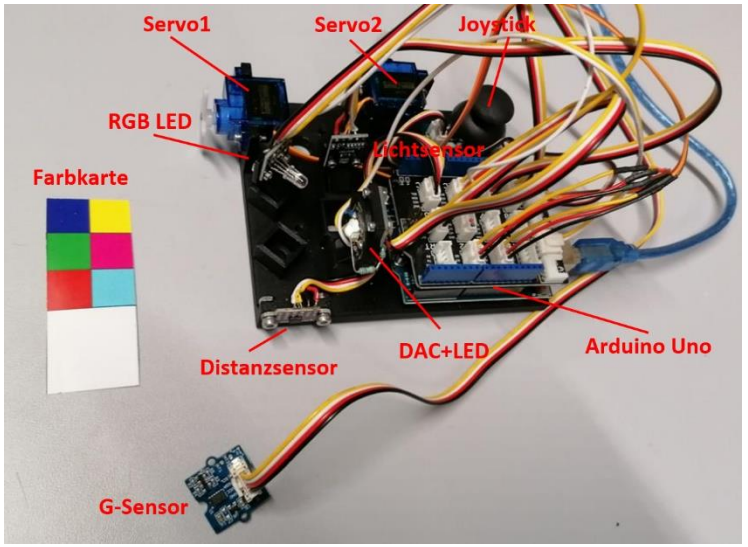


Raspberry Pi 4
Quelle: raspberrypi.org

In dieser VU

Basis - Arduino Uno & Nano

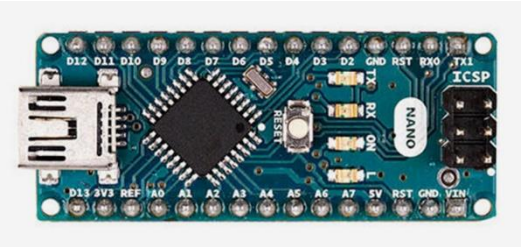
- ATMega328
- 8 Bit Microcontroller mit 12MHz
- GPIOs / ADCs / PWM Generatoren
- 32kb Speicher
- I²C / USB Anschluss (FTDI-UART)



Arduino Setup für Computergestützte Experimente und Signalverarbeitungs VU

Vorteile

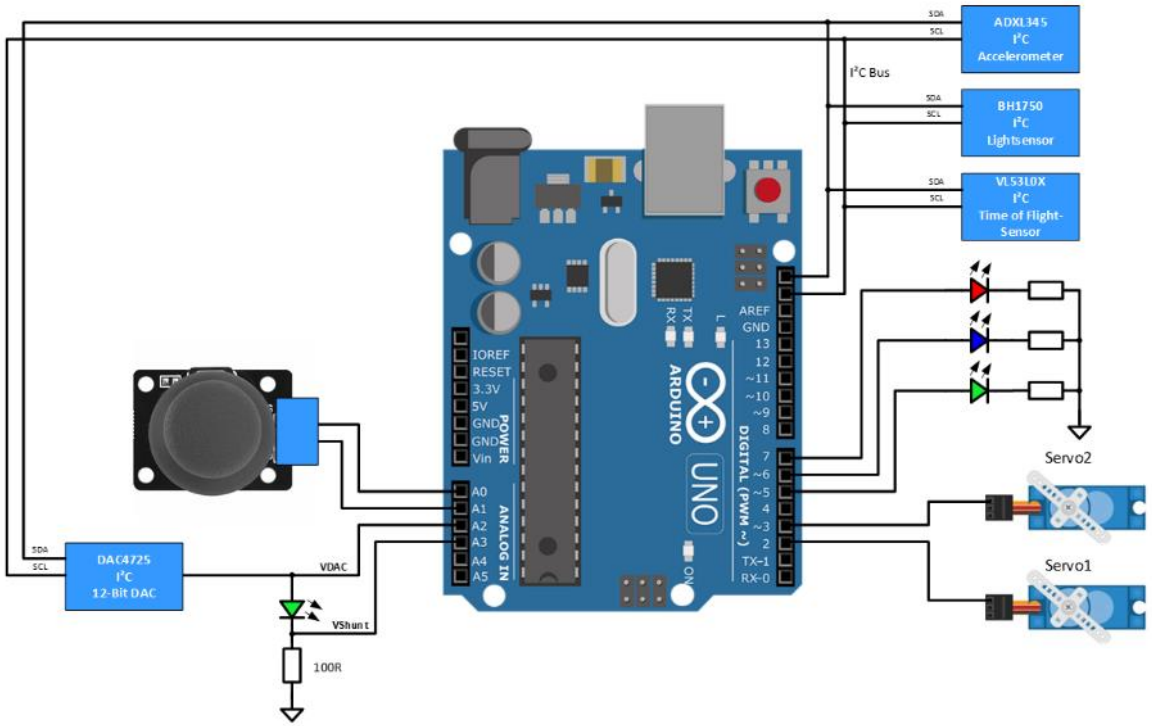
- Billig
- Leicht zu programmieren – „C“
- Billige Sensoren/Module
  - AZdelivery, Adafruit, Seeedstudio, Amazon Shops, ect.



Arduino Nano
Quelle: arduino.cc

Übungsboard

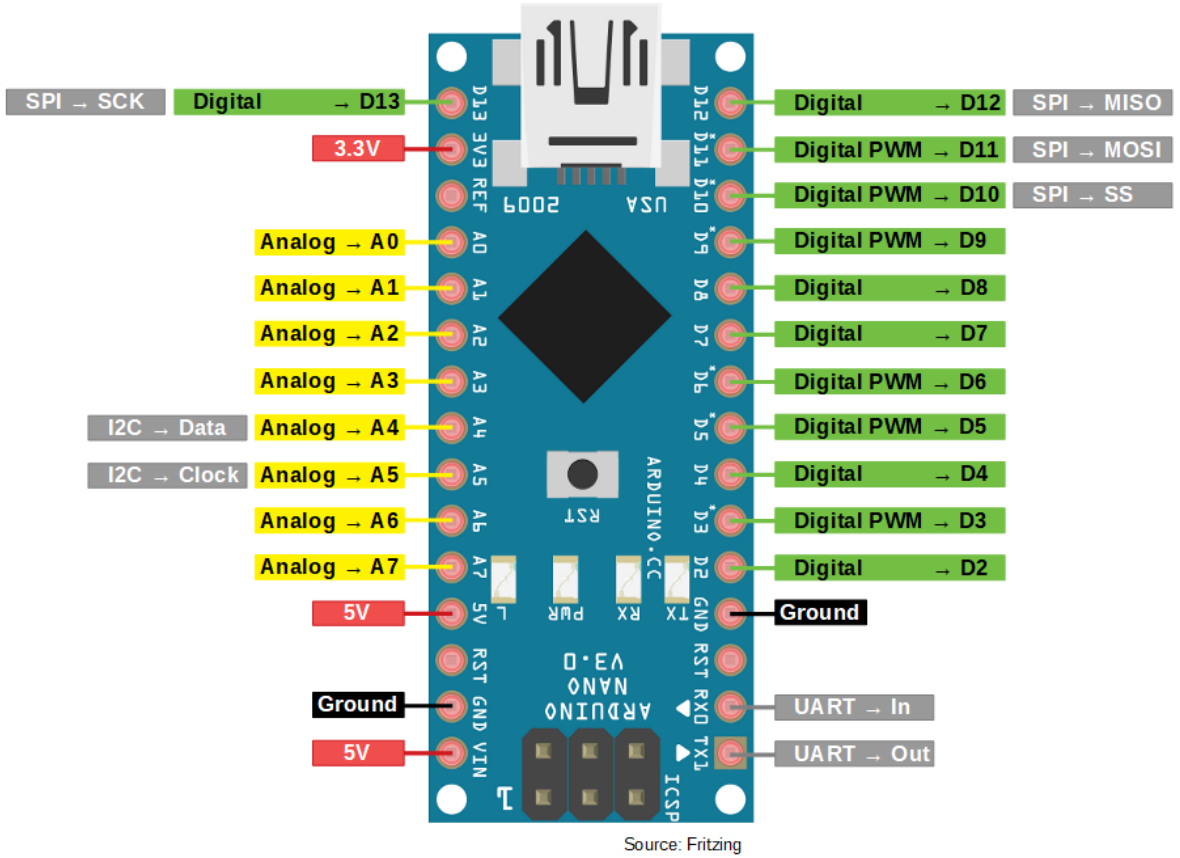
- Arduino Uno
- Seeed Sensor Shield
- Sensoren
  - Lichtsensor - Photometrisch
  - Joystick
  - Accelerometer
  - ADC
  - TOF Sensor
- Aktoren
  - 2x Servos
  - RGB LED
  - 1x DAC verbunden mit Power-LED
- Mechanik 3D Gedruckt



Dokumentation @ Moodle

GPIO/ADC/DAC

Arduino Nano Pinout



Arduino Nano Pinout
Quelle: arduino.cc Forum

GPIOs

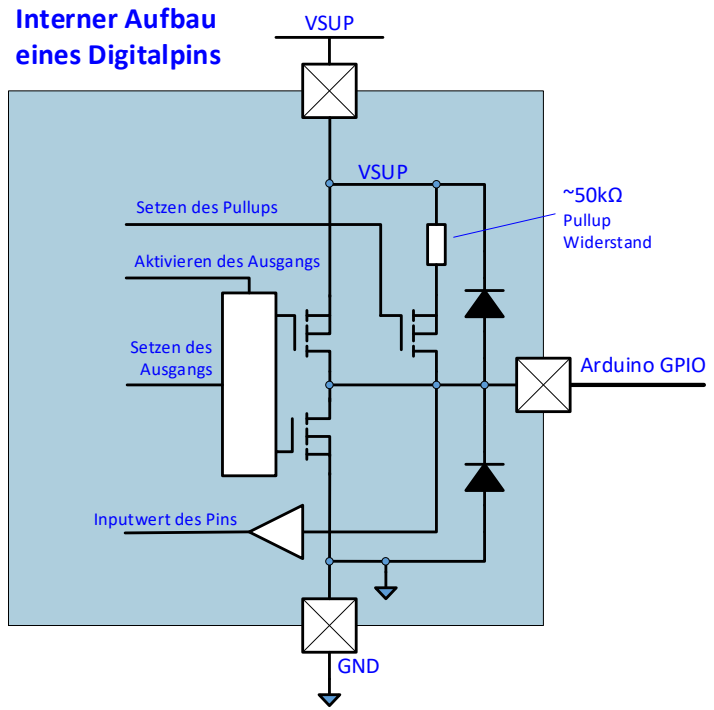
General **P**urpose **I**nput and **O**utput

AKA „Digitalpin“

- Pin konfigurierbar als INPUT oder OUTPUT
- *Input*: kann internen Pull-up benutzen ~ 50k
- *Output*: kann ca. ~40mA treiben als Senke oder Quelle
- Jeder Mikrocontroller hat andere GPIO Spezifikationen

Zu beachten:

- Inputs haben eine Schaltschwelle (Threshold)
  - Low < 1.4V
  - High > 3.4V
  - ADC kann Threshold Probleme umgehen
- Konkurrierende Hardware Blöcke am selben Pin
  - Wenn I²C/UART aktiviert ist kann man nicht mehr auf diese Pins zugreifen oder werden überschrieben
- „Default State“



Interner Aufbau eines Microcontroller GPIO

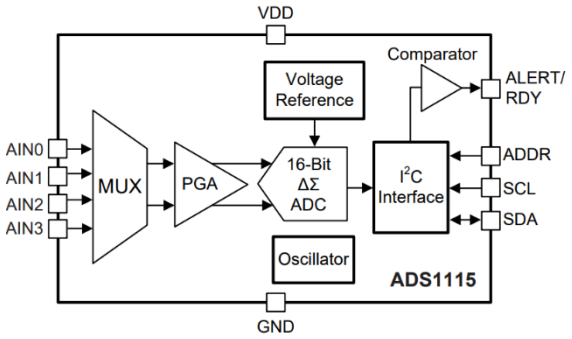
Coding Beispiel @ Moodle

ADC

Analog to **D**igital **C**onversion

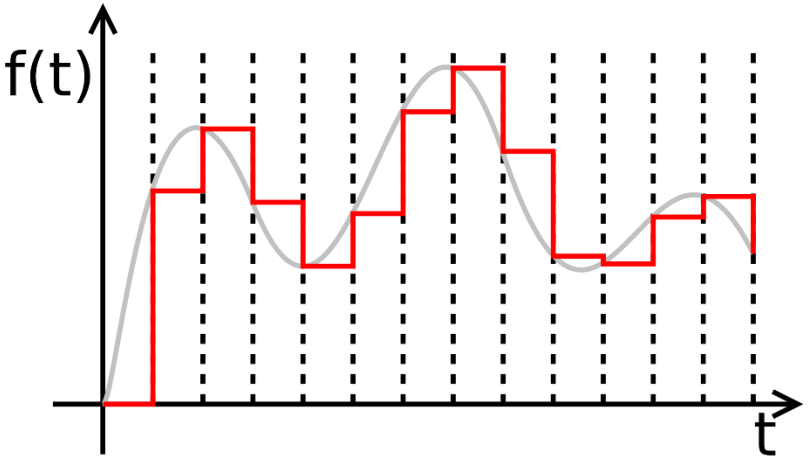
Wandelt eine analoge Spannung in einen digitalen Wert.

- Integriert oder als eigener Baustein/Chip
- Arduino Nano - Integriert
  - Auflösung 10-Bit
  - Messzeit pro Sample ~ 120µs
  - Messbereich: 0-5V → 5V Referenz
- SNR – **S**ignal to **N**oise **R**atio
  - Arduino → keine gute SNR
  - Externe ICs → gute SNR
  - Externe System → exzellent
- Limits
  - Frequenzen höher als Abtastrate
  - Signale kleiner als die Genauigkeit
- Dynamikbereich beeinflussen
  - Durch Verstärker
  - Durch Anpassung der Referenz
  - Höhere Auflösung



Texas Instruments ADS1115 - Quelle: ti.com

Adafruit ADS1115 Board
Quelle: adafruit.com



Signalabtastung mittels ADC - Quelle: wikipedia.de

Coding Beispiel @ Moodle

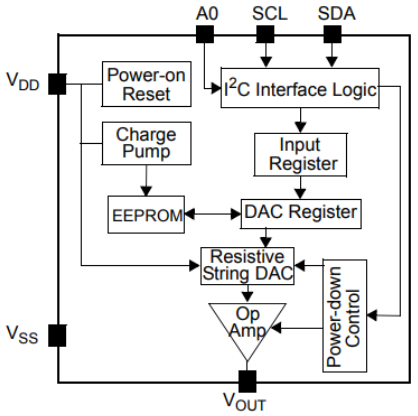
DAC

Digital to **A**nalog **C**onversion

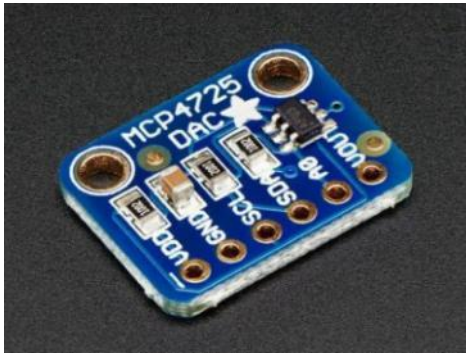
Wandelt einen digitalen Wert in eine analoge Spannung um.

Eigenschaften

- Integriert oder als eigener Baustein/Chip
- Arduino Nano hat keinen internen DAC
  - Nano hat nur PWM Generatoren!
  - Arduino Duo hat echten DAC
- Auflösung
  - Externe Kartensysteme oder Chips ~ 8-32 Bit
- Stellbereich / Dynamikbereich: zb. 0-5V
- Stromtreibefähigkeit: zb.: 15mA
- Linearität → y=**k***x+d
- Offset → y=k*x+**d**



Microchip MCP4725 - Quelle: microchip.com



Adafruit MC4725 Board
Quelle: adafruit.com

PC Mess Technik – ADC/DAC/GPIO

ADC / DAC / GPIO mit PC Verbunden

- Sehr viele Hersteller & Modelle verfügbar
  - National Instruments, Keysight, Agilent
  - USB-Oszilloskop
- Verschiedene Interfaces
  - GPIB / USB usw
- Combo Geräte, zB:
  - 8x ADCs / 2x DACs / 10x GPIOs
  - Verschiedenste Versionen verfügbar
- Vorteil
  - höhere Genauigkeit / Stabilität
  - höhere Datenrate als zb.: Arduino
  - Eigene SW/Labview Treiber/DLLs/Treiber ect.
  - USB3.0
  - Keine „schnelle“ Reaktion möglich ~25ms
- Nachteil
  - Einbindung in eigene Software oft Aufwändig
  - Kostenintensiv (100€-10.000€)

National Instruments NI USB6501 GPIO - Quelle: ni.com

LabJack USB Messlabor - Quelle: labjack.com

Weiteres USB Messlabor - Quelle: cesys.com

Rechenbeispiele

Rechenbeispiel #1 - Arduino ADC Genauigkeit

- Messbereich = 5V, Auflösung = 10 Bits
- Genauigkeit = 4.88mV/Bit

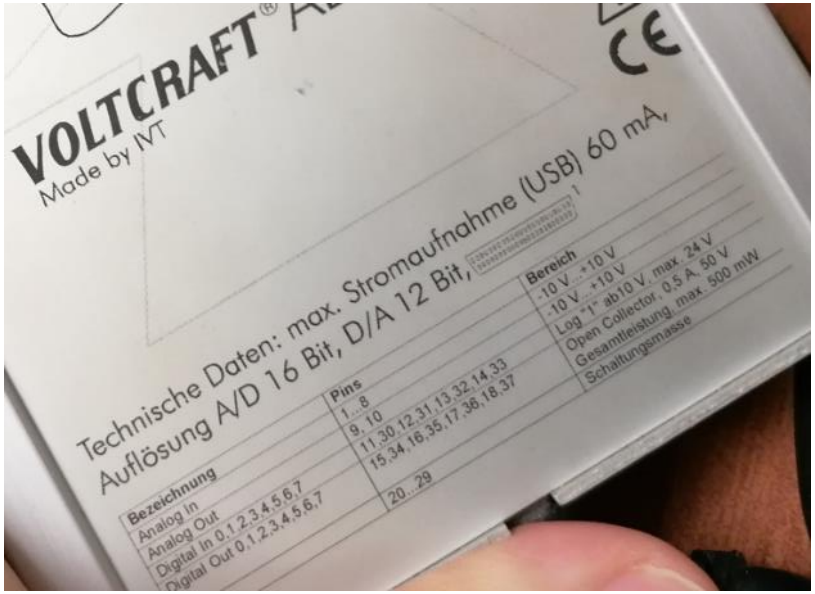
$$Genauigkeit\left[\frac{V}{bit}\right] = \frac{Messbereich[V]}{Auflösung[bits]} = \frac{5[V]}{2^{10}[bits]} = 4.88\left[\frac{mV}{bit}\right]$$

Rechenbeispiel #2 – Arduino ADC Messwert

- Messbereich = 5V, Messwert = 234 (dezimal)
- Messwert = 234*(5.0V/1023.0) = 1.14V

Rechenbeispiel #3

- Messbereich = +10V/-10V, Auflösung = 16 Bits
- Genauigkeit = 0.305mV/Bit



USB-Messsystem im Labor der Universität Graz

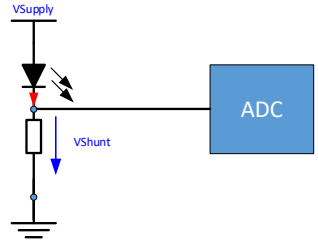
Mögliche Prüfungsfragen

- Sie bekommen von ihrem Arduino (5V Referenz, 10 Bit ADC) einen Messwert von 200 zurück. Welcher Spannung entspricht dies? Wie lautet die Formel dazu?

- Sie versuchen mit einem GPIO ein digitales Signal zu erfassen. Mit dem Oszilloskop messen Sie Signalwerte zwischen 0 und 0.5V, was ist der Grund das sie dieses Signal nicht Digital erfassen können? Und wie könnten sie dieses dann doch messen?

- Sie wollen den Strom durch einen Diodenlaser Messen. Dazu wird seriell zum Laser ein “Shunt”-Widerstand eingelötet welcher 25.0 Ohm hat. Sie messen die Analogspannung am Shunt-Widerstand mit einem 10Bit ADC welcher eine Referenzspannung von 5V hat.

1. Was ist der maximal messbare Strom?
2. Was ist der kleinste messbare Strom?

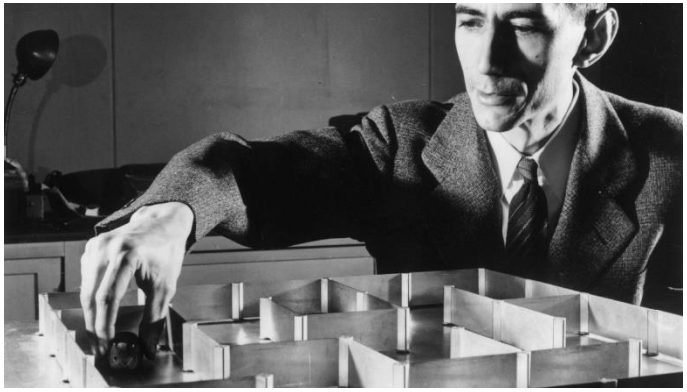


Schnittstellen und Bussysteme

Informationstheorie

Claude Shannon, 1948 ‚A Mathematical Theory of Communication‘:

„Eine Nachricht ist die ***Übertragung*** von Zeichen aus einem Vorrat der zuvor durch ***Übereinkunft*** zwischen Sender und Empfänger festgelegt wurde“



Claude Shannon
Quelle: spiegel.de

<u>Praktische Bedeutung:</u>

- Gleiches Protokoll
- Gleiche Geschwindigkeit
- Gleiche Spannungen
- Gleicher Zeichensatz/Sprache

Auch bei Menschen muss man die selbe Sprache sprechen um verstanden zu werden Quelle: Freepik.com

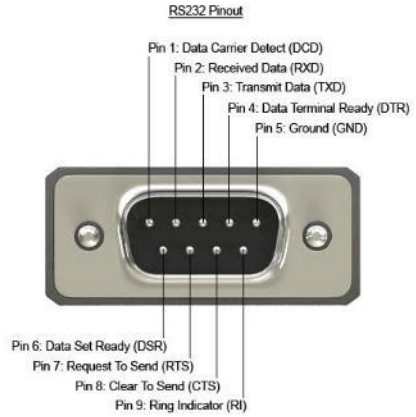
Übersicht der Schnittstellen

Schnittstelle	Anwendung
RS232 UART	PC zu Messgerät Mikrocontroller zu Mikrocontroller
I^2C	Mikrocontroller zu Sensor/Display/Aktor/DAC/ADC
Single Wire	Sensoren / Aktoren
USB	PC zu Mikrocontroller (oft FTDI/UART) HMI
SPI	Speicher / Sensoren / Displays
GPIB	PC zu Messgerät (Multimeter, Netzteil ect.)
Bluetooth	PC zu Mikrocontroller Mikrocontroller zu Mikrocontroller

RS232 / UART Basics

RS232 - Spannungsschnittstelle

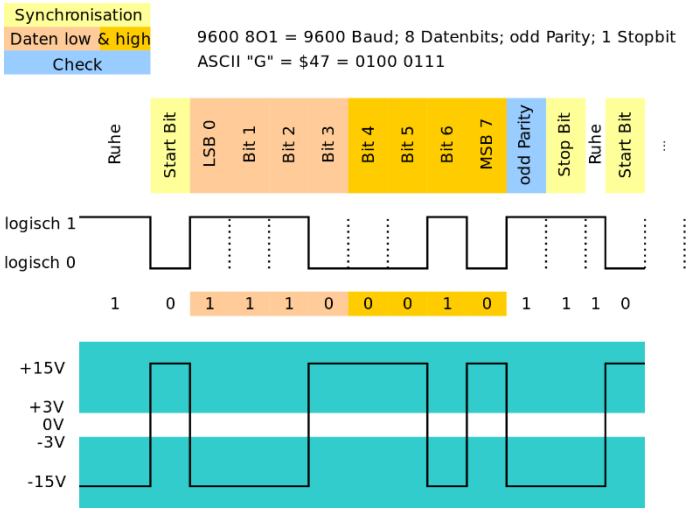
- Seit 1960
  - Seit 2010 nicht mehr auf PC Mainboards
- Meistens SubD9 Stecker
- Arbeitet mit +/-12V
- Verwendet den UART Standard
- Kurze Distanzen(2-3m)
  - je nach Baudrate & Störquellen



SubD-9 PinOut - Quelle: wikipedia.de

UART – Übertragungsstandard

- **U**niversal **A**synchronous **R**eceiver/**T**ransmitter
- Bidirektional – (meistens) nur Gerät zur Gerät
- Beide Geräte haben eigene Taktfrequenz
- Beispiele:
  - PC zu PC
  - Arduino zu Arduino
  - Arduino zu Bluetooth-Modul
  - Arduino zu Sensor

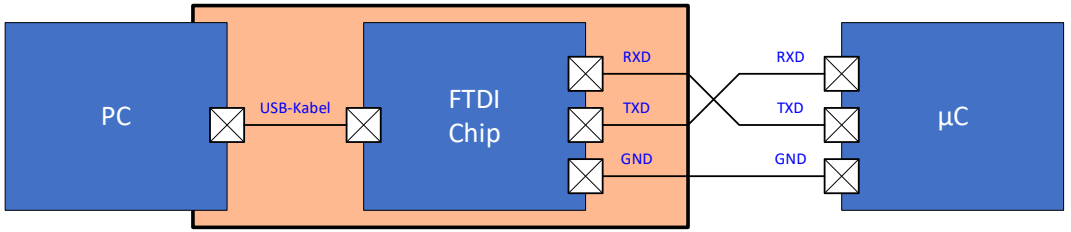


Datenübertragung bei RS232 - Quelle: wikipedia.de

UART

- Verbindung von PC zu Arduino
  - Per USB / FTDI mit TTL3.3V oder 5V.0
- USB-auf-UART Chip im Arduino verbaut - FTDI oder CH340
- FTDI/CH340 Chip meldet sich am PC (serielle Schnittstelle) an
- Minimalausführung 2x Leitungen – RXD, TXD
  - Größere Ausführungen möglich (CTS/RTS)
- Arduino hat eigenen „Hardware“ – UART Block
  - Spezielle Pins (D0 & D1)
- Auch Software UART möglich
  - Jeder GPIO möglich
  - Braucht viel Rechenleistung
  - Library verfügbar

USB-auf-UART Kabel - Quelle: rs-component.de

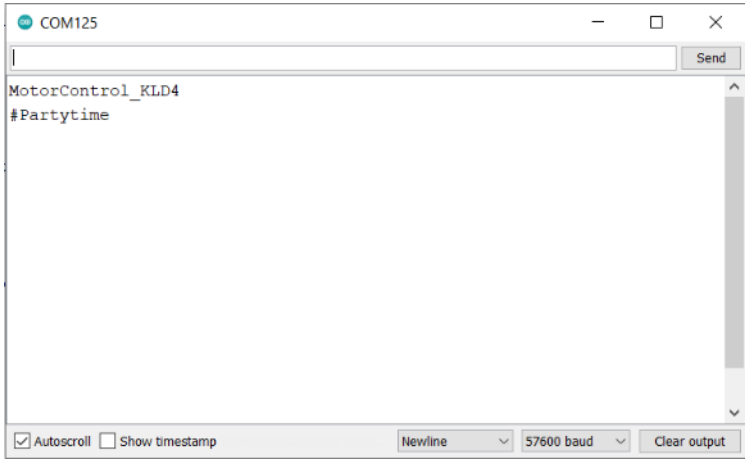


„UART - Kabel“

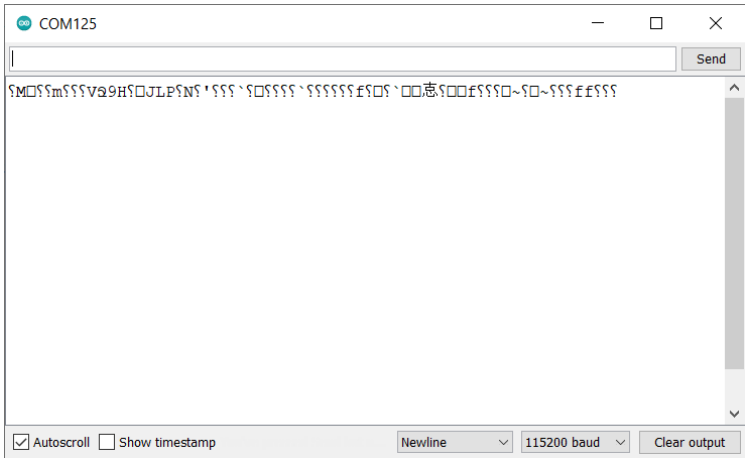
Coding Beispiel @ Moodle

Warum die Verbindung meistens nicht klappt

- Settings von beiden Geräten - **Übereinkunft**
  - Baud Rate
  - Anzahl Start/Stoppbits
  - Parity – Kann ‚Even' oder ‚Odd' sein
  - Protokoll
    - XON/XOFF
    - RTS/CTS
- Vergessen RXD und TXD Auszukreuzen
  - Nullmodem Kabel
  - Straight Kabel
- Falsche Spannungslevels
- Falsche Pins angeschlossen?
- „End of line" Signal (**Übereinkunft**)
  - CR & LF – Carriage Return & Line Feed
  - „0" – NULL
  - Manchmal auch andere Zeichen



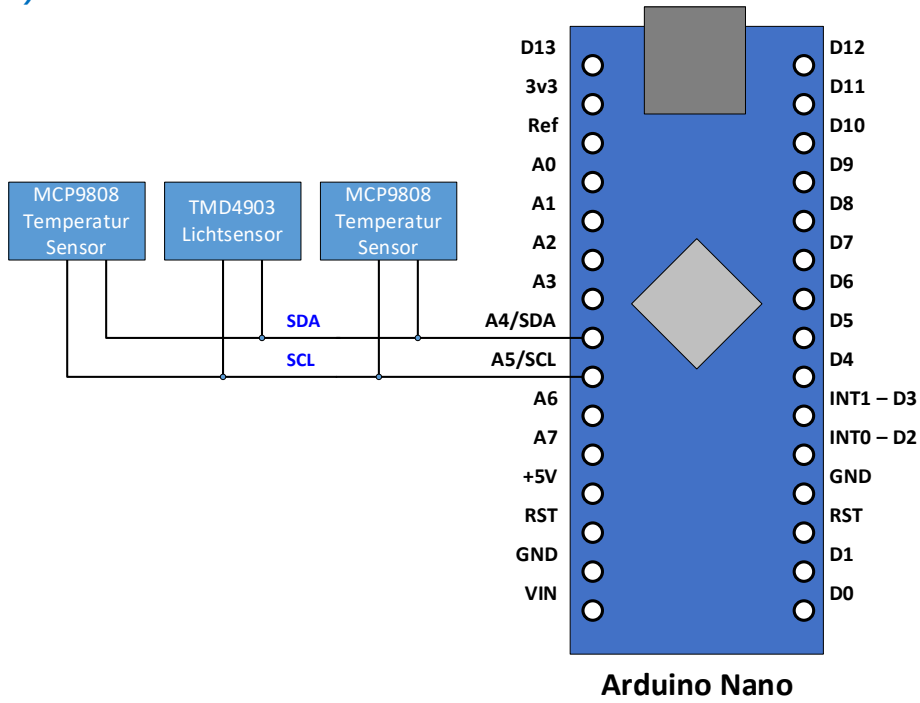
Arduino IDE – Serial Monitor – Korrekte Baudrate & End of line



Arduino IDE – Serial Monitor – Falsche Baudrate

I²C

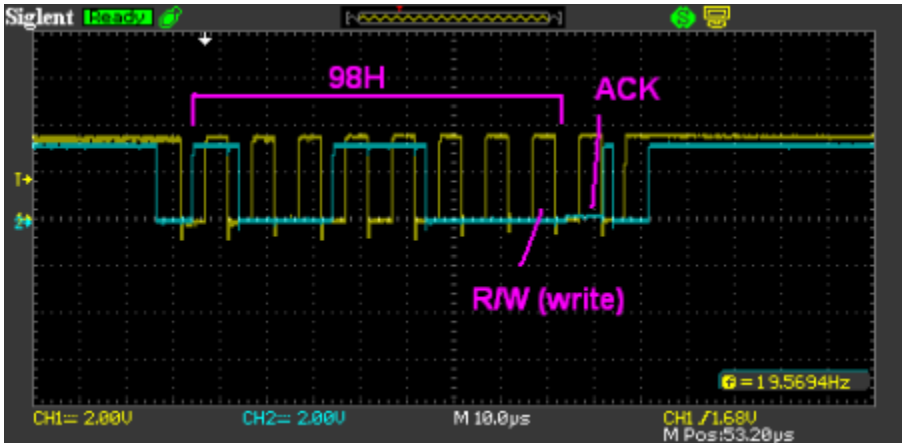
- Synchrones Serielles Kommunikationsprotokoll (1982)
- Mehrere Devices an einem Bus möglich
  - Sehr beliebt in Smartphones/Tablets/Portable Elektronik
  - 100kHz/400kHz/3400kHz
- Host/Client System (früher Master/Slave)
  - Host = Prozessor/Mikrocontroller – gibt den Takt vor
  - Client = Sensor/Aktor mit **eigener I²C Adresse**
- 2x Leitungen
  - SDA – **S**erial **Da**ta
  - SCL – **S**erial **Cl**ock
- Pull-up widerstände benötigt
  - Arduino hat integrierte Pull-up widerstände ~50kΩ
  - kleinere Pull-up Widerstände (2~3k) wirken oft wunder bei Störungen
- Sensor Datenblatt enthält
  - I²C - Registermap
  - Programmierinformationen
- Arduino auch als Sender/Receiver konfigurierbar



Coding Beispiel @ Moodle (Bluetooth Demo)

Mögliche Fehlerquellen bei I²C

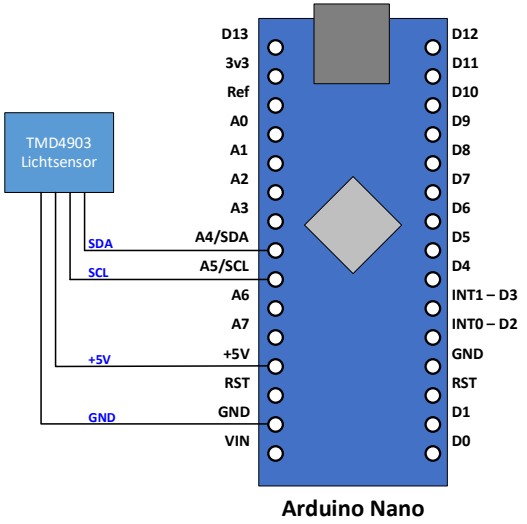
- Falsche I²C Adresse des Chips
- Spannungslevels passen nicht
  - Pullups?
  - Levelshifter?
- SDA und SCL vertauscht
- Vdd oder GND vergessen



I²C Adressierung - Quelle: Analog Devices Forum – ez.analog.com

Debugging:

- Oszilloskop nehmen
  - Signale messen
  - Hardware ausschließen
- Software
  - Durch-‘sweepen‘ aller I²C Adressen
  - Auf „ACK“ suche gehen
- I²C und Sensor getrennt voneinander testen
  - Mit bekannt funktionierenden Systemen



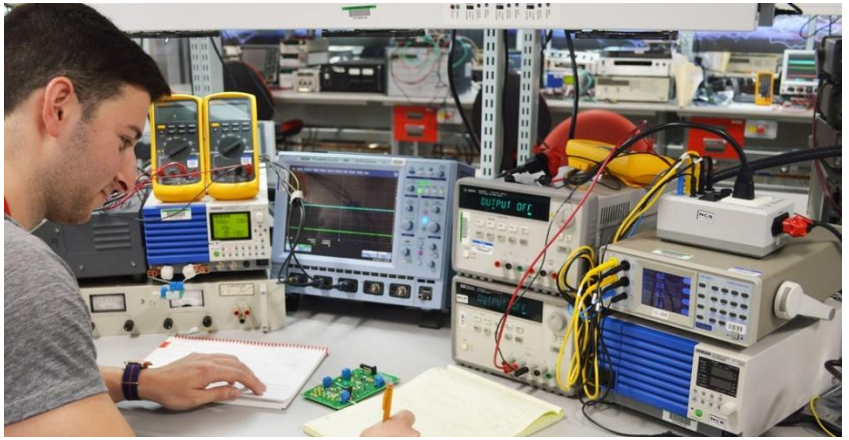
GPIB

Parallelschnittstelle (~1970)

- 1MByte/s
- 8-Bit Parallel Datenleitungen (geschirmt)
- Handshake betrieb
- Sehr Robust
- Beliebt bei Labor Messtechnik
  - Netzgeräte/Multimeter
  - Elektrische Messtechnik
  - Oszilloskope
- Vorteil:
  - Robustes Interface
  - Industriestandard / LabVIEW Treiber
  - Nicht tot zu kriegen
- Nachteil:
  - Teures Interface
  - SW Treiber / Eigene GUI
  - Nicht tot zu kriegen

Keithely GPIB Schnittstelle - Quelle: farnell.de

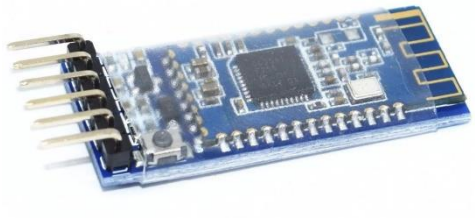
National Instruments – GPIB auf USB Quelle: farnell.de



Labor Messsetup - Quelle: Texas Instruments

Bluetooth

- Integriert in Smartphone/Laptop, Raspberry Pi ect.
- Peer-2-Peer Interface, 2.4GHz
- Reichweite ~10m
  - bis zu 1km durch Sendeleistung „boosten“ möglich
- UART-to-Bluetooth Module
  - RN42 / HC-05 / HC06
  - Anschließbar an Arduino als UART
- ESP32 – 32-Bit Mikrokontroller mit BT
- Arduino Nano 33 BLE / IoT



UART to Bluetooth Module –Quelle: roboter-bausatz.de

ESP32 – Arduino Kompatibler Mikrocontroller mit Bluetooth (BLE)

Integrationsmöglichkeiten

- Eigenen Stack programmieren
  - Sehr Aufwändig
- Vorgefertigte Chips/Protokolle nutzen
  - Bei UART to Bluetooth Modulen
  - Wenig aufwand in der Programmierung
  - Muss Konfiguriert werden

Arduino Nano 33 BLE Sense Rev.2

Quelle: Reichelt.at

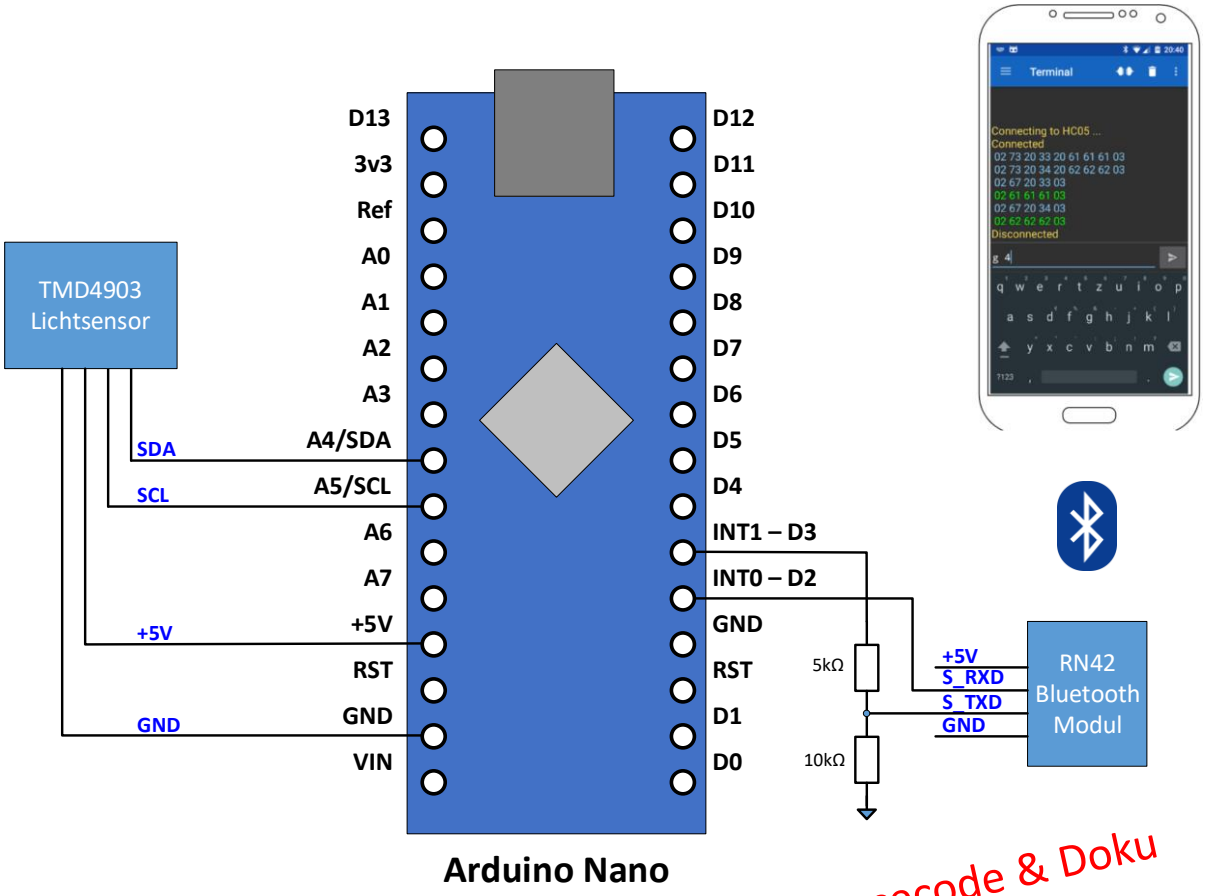
Bluetooth/UART/I²C Demo

Hardware

- Lichtsensor TMD4903
  - Per I²C
- Bluetooth Modul – HC05
  - Verbindung zum Smartphone
  - Software-UART Pin D2 und D3
- USB Stecker
  - Verbindung zum PC
  - Hardware-UART PIN D0 und D1

Software

- Von Appstore beliebiges Serial Interface App gewählt für Smartphone
- Lichtsensor Messdaten über Bluetooth schicken
- Bidirektionale Kommunikation zwischen PC und Smartphone möglich
- Messung ein/ausschaltbar mit „$“
- BT Modul muss einmalig konfiguriert werden über „AT-Commands“!



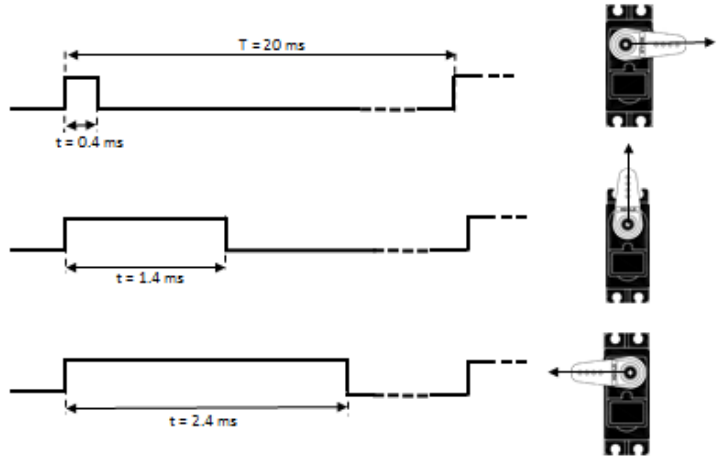
Sourcecode & Doku
@ Moodle

Single-Wire – Beispiel Servomotor

Single Wire:

- „Tastgrad“ Modulation
- Beliebt für
  - Servos
  - Temperatursensoren
  - klein Anwendungen
- Nur 1x Pin benutzt

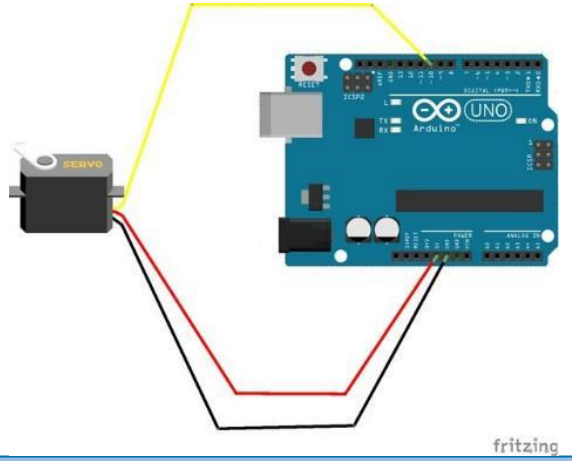
t	Duty Cycle	Direction
0.4 ms	0.4/20 = 2%	0 degs
1.4 ms	1.4/20 = 7%	90 degs
2.4 ms	2.4/20 = 12%	180 degs



Quelle: Python-exemplarisch.ch

Servomotor Beispiel

- Periodendauer = 20ms
- 0.4ms high time → 0°
- 2.4ms high time → 180°



Mögliche Prüfungsfragen

- Ein Kollege bittet Sie bei einem Computergestützten Experiment um Hilfe. Es “geht einfach nicht”. Dabei handelt es sich um ein UART-Interface zwischen einem PC und einem Piezotisch mit integriertem Mikrocontroller. Welche möglichen Optionen gibt es warum die Kommunikation “einfach nicht geht”.

- Sie haben in ihrer Masterarbeit einen I^2C-Beschleunigungs Sensor zum überwachen ihres Experiments an einen Arduino angeschlossen und bekommen kein Signal? Was sind mögliche Gründe? Was für Möglichkeiten gibt es die Fehler einzugrenzen?

- Was sind die Nachteile wenn sie ein Software UART Modul für ihr Computergestütztes Experiment verwenden? Was sind die Vorteile?

- Sie bekommen bei ihrem Serial Monitor Fenster folgende Zeichen: „????!“§435?????:YXC_“. Was kann der mögliche Grund sein?

Timer/Polling/Interupts

Timer

Anwendungen

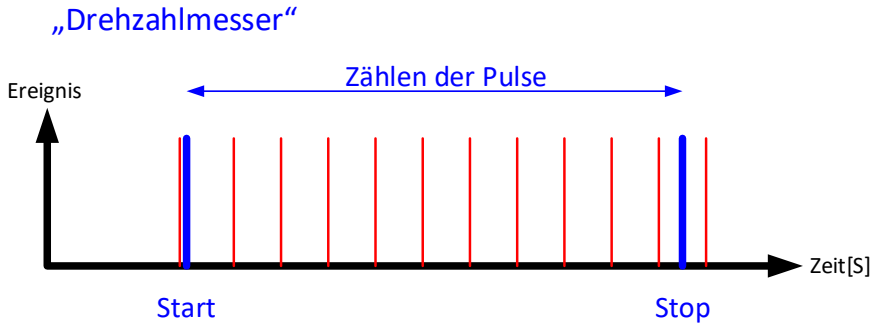
- Verzögerung
  - Auf Ereignis reagieren
  - Zb.: Relais 1s später ausschalten
- Kontinuierliche Wiederholung
  - Messschleifen
    - Im 50ms Takt von einem Sensor Daten holen
    - Nur alle 500ms einen Messwert an den PC schicken
- Zeit/Frequenzmessung
  - Sehr genaue Zeitmessungen mit dem Arduino schwer
  - Aufgrund von internen Interrupts → Jitter
  - Arduino Timer Taktfrequenz = 1MHz

Quelle: Shelvin.de

Zeitmessungen

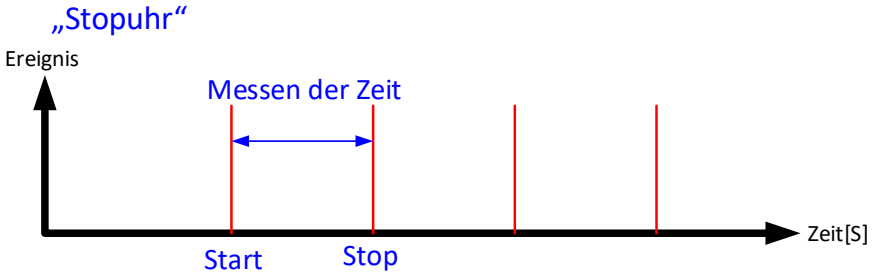
Verwendung für

- Drehzahl von zb.: Windrad Umdrehungen
- Laufzeiten von Kugeln in Flüssigkeiten
- Abklingkonstanten von Bierschaum ect.



Methoden

- Messung von Zeit zwischen zwei Ereignissen
  - „Stopuhr-Methode“
  - Erstes Ereignis →Starten des Timers
  - 2.tes Ereignis → Stoppen des Timers
  - Hohe Auflösung bei langer Messzeit

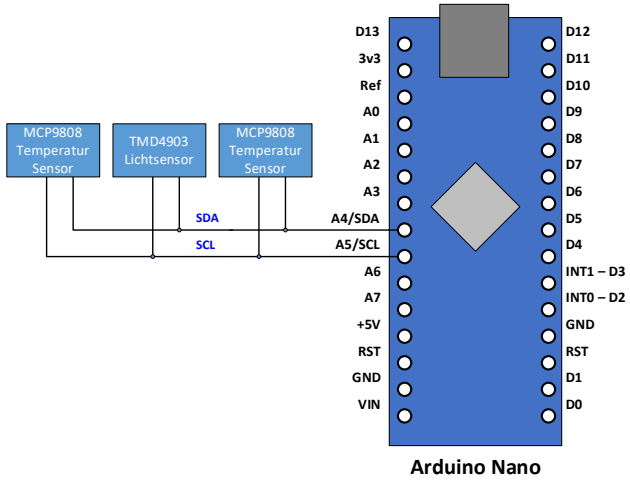


- Messung von „Anzahl der Ereignisse“ zwischen zwei fixen Zeitpunkten
  - „Drehzahlmesser“
  - Hohe Auflösung bei vielen Ereignissen pro Messzeit
  - Bei hoher Signalfrequenz: Überladen des Arduino möglich

2x Coding Beispiele
@ Moodle

Polling

Polling bedeutet kontinuierliches hereinholen von Daten von einem Sensor.



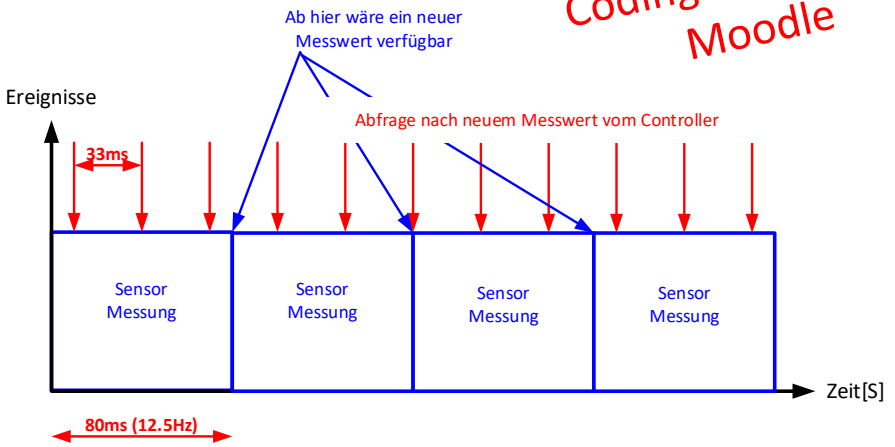
Synchronisation

- Sensor misst **kontinuierlich** und hinterlegt das Messresultat in einem Register im Sensor
- Sensor und Arduino haben eigenen Clock

Wie synchronisieren?

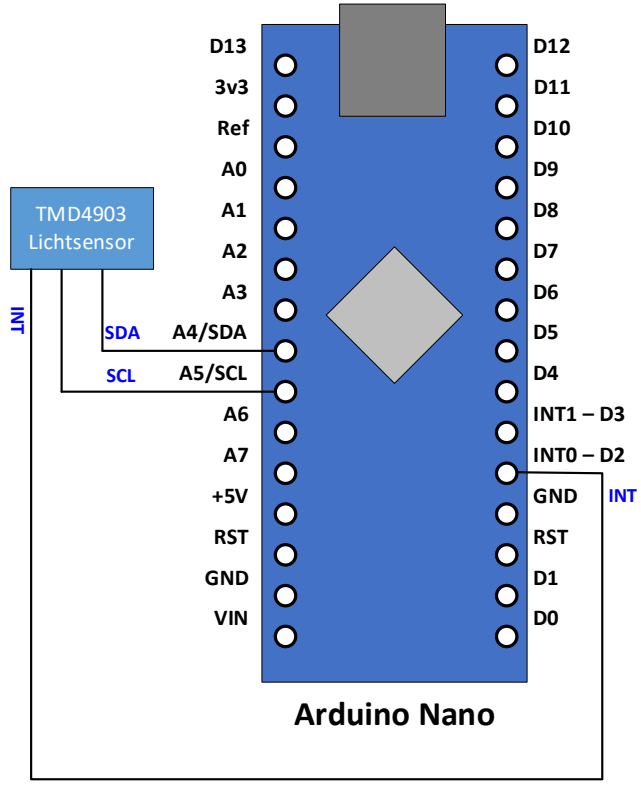
- Sensor bietet ein „Measurement-Finished-Bit“ an.
- Sensor wird bei jeder Messung neu gestartet
- Sensor hat eine Interrupt Leitung welche Anzeigt das die Messung fertig ist.
- Daten werden langsamer eingelesen als möglich wäre → keine Synchronisation nötig

Coding Beispiele @ Moodle



Interrupts – Unterbrechung des laufendes Programms

- Ereignis orientierte Programmierung
- Interne Interrupts
  - Timer
  - ADC Finished
  - UART hat Daten (fertig) empfangen
  - UART hat Daten (fertig) gesendet
- Externe Interrupts
  - INT0, INT1 Pin
    - Zum triggern einer „Stopuhr/Drehzahlmessers“
  - Verbunden mit einem Sensor
    - zb.: Sensor hat Messung fertig



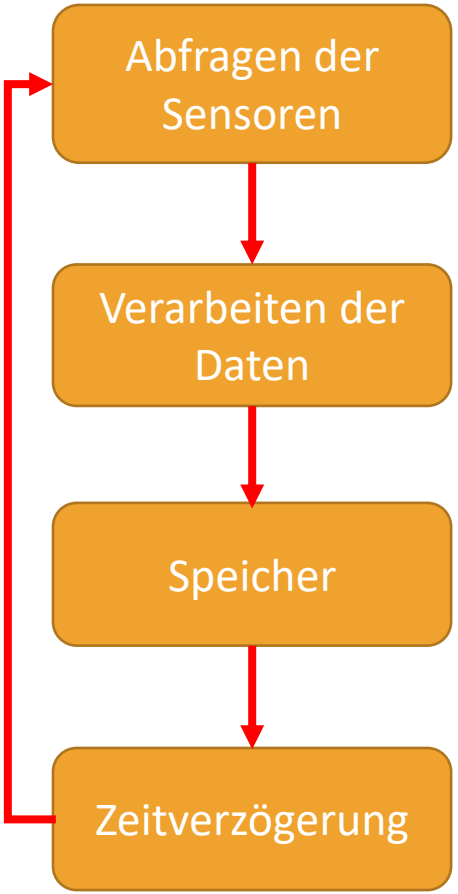
Messsystem - Durchlaufzeit

Beispiel:

- Sensorsampling benötigt 50ms
- Verarbeiten dauert 12µs
- Speichern dauert ~200ms!!! (SD-Karte)
  - ArduinoCC/ChatGPT Codes benötigen oft 200ms.
- Zeitverzögerung
  - Variabel
  - Fix
- ~4 Samples pro Sekunde möglich!

Unterschiede

- ‚Fire & Forget' befehle
- ‚Wait until finished' befehle



Mögliche Prüfungsfragen

- Sie haben für ihr Experiment einen Temperatursensor so konfiguriert das dieser kontinuierlich alle 250ms einen neuen Messwert bereitstellt. Was ist die minimale Zeit die sie wählen müssen das sie bei jeder Anfrage an den Sensor einen neuen Messwert bekommen? Welche Möglichkeiten gibt es um auszuschließen das sie einen Messwert Doppelt auslesen?

- Sie haben zwei Lichtschranken im Abstand von 25cm mit beiden Interrupt-Pins eines Arduino verbunden und versuchen nun die Austrittsgeschwindigkeit eines Cola+Mentos Experiments zu bestimmen. Welche Messmethode würden sie dazu verwenden, beschreiben sie grob den Messablauf.

- Sie arbeiten in einem Labor wo sie eine Vakuumpumpe für ein Experiment modifiziert haben, um die Drehzahl zu messen haben sie in den Rotor zwei Magnete eingebaut sowie im Stator einen Hall-Sensor. Sie bekommen also zwei pulse pro Umdrehung. Mit ihrem Arduino messen sie nun in 20 Millisekunden 75 pulse. Wie viele Umdrehungen pro Minute sind das?

The End