

IT-Systeme / Mobile-Systeme (MOSY-Teil)

Prof. Dr. Torsten Edeler

HAW-Hamburg, Medientechnik

27. März 2017

Python

Raspberry
Pi

Grundlagen
EL

GPIO

① Python

② Raspberry Pi

Raspberry Pi Laborübung

③ Grundlagen Elektronik

Der elektrische Stromkreis

Das OHMsche Gesetz

Kirchhoffsche Gesetze

LED an einen Computer

Tasteranschluss an einen Computer

④ GPIO

Laboraübung

Python

Raspberry
Pi

Grundlagen
EL

GPIO

- 1 Python
- 2 Raspberry Pi
- 3 Grundlagen Elektronik
- 4 GPIO

Python

Raspberry
Pi

Grundlagen
EL

GPIO

- Skriptsprache
- Fokus auf einfache Erlernbarkeit
- Große Nutzerbasis (Viel Hilfreiches im Internet)
- Auf dem RPI quasi die standard Skriptsprache

Nachteile:

- Langsamer als C (da interpretiert)

Python

Raspberry
PiGrundlagen
EL

GPIO

- Entwicklung seit 1983
- Derzeit sind aktuell die Version 2.7 und 3.5
- Version 2 und 3 sind inkompatibel
 - Die Unterschiede sind marginal, jedoch relevant für die Ausführung.
 - Wir arbeiten mit Version 3

Die Entwicklung in Python basiert stark auf der Nutzung von sog. **Paketen**. Die meisten Pakete sind für Version 2 und 3 verfügbar.

- Sie können Python zu Hause installieren.
 - Laden Sie sich dafür das Paket Anaconda <https://www.continuum.io/downloads> herunter.
 - Verfügbar für Linux, MacOS und Windows
 - Achten Sie auf Python Version 3!
 - Videos:
 - Windows:
<https://www.youtube.com/watch?v=LvmpDyFyS7o>
- Pakete installieren:
 - Kommandozeile¹ `conda install <paketname>`
- Python auf aktuellen Stand bringen:
 - Kommandozeile `conda update anaconda`
- Informationen über das Programm conda:
<https://www.youtube.com/watch?v=YJC6ldI3hWk>

¹Unter Windows erreichen Sie die Kommandozeile (Dosfenster) durch Windowstaste → cmd

Python installieren (Auf dem Raspberry Pi)

Python

Raspberry
Pi

Grundlagen
EL

GPIO

- Auf dem Standardsystem ist Python immer installiert!
- Pythonversionen:
 - Version 2: `python`
 - Version 3: `python3`
- Pakete installieren (für Version 3):
 - `sudo pip3 update` (einmalig)
`sudo pip3 install <packetname>`
 - Derzeit gibt es ca. 100.000 Pakete für Python.
 - Übersicht hier: <https://pypi.python.org/pypi>

Python

Raspberry Pi

Grundlagen EL

GPIO

- Bücher

- Raspberry Pi programmieren mit Python

- Autor: Michael Weigend
 - Verlag: mitp; Auflage: 3. Auflage 2016
(16. Mai 2016)
 - ISBN-13: 978-3958454293



- Internet

- Ein ganz nettes Tutorial finden Sie hier:
http://www.python-kurs.eu/python3_kurs.php
 - Noch ein Tutorial für das *Jupyter-Notebook*:
http://nbviewer.jupyter.org/github/ehmatthes/intro_programming/blob/master/notebooks/hello_world.ipynb
 - Mit Google erhalten Sie so gut wie immer eine Antwort.
 - Youtube-Videos sind teilweise sehr hilfreich.

Übungsthemen Stichwortartig. Bitte machen Sie sich Notizen:

- Starten einer Python-Shell
- Starten Jupyter Notebook
- Ausgabe mit `print()`
- Variablen
- Funktionen
- Datentypen
 - Zahlen (Integer, Float)
 - Zeichenketten (Strings)
 - Sammelobjekte (Listen, Tuple, Dictionary)
- `if`-Abfragen
- `for`-Schleife
- `while`-Schleife

Python

Raspberry
Pi

Laborübung

Grundlagen
EL

GPIO

① Python

② Raspberry Pi

Raspberry Pi Laborübung

③ Grundlagen Elektronik

④ GPIO

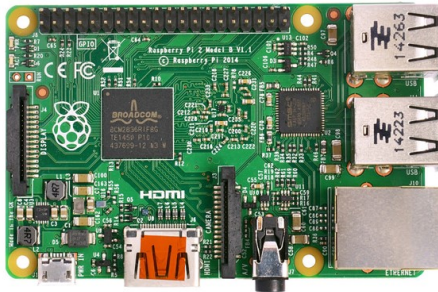
Python

Raspberry
Pi

Laborübung

Grundlagen
EL

GPIO



- Eine Platine
- Komplette funktionsfähiger PC mit WLAN, USB, Ethernet etc.
- Linux Betriebssystem

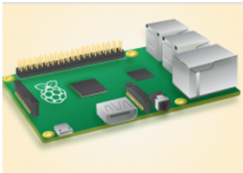
- Einplatinencomputer
 - Sehr günstig (ca. 40Euro)
 - Betriebssystem Linux
 - Einfache elektrische Schnittstelle (GPIO)
 - Leistungsbedarf ca. 3Watt
 - Sehr viele Sensoren / Aktoren (Zusatzhardware)
 - Sehr gute Softwareunterstützung
 - Viel Hilfe und Tutorials im Netz
-
- Sie finden sehr viele „Copy-Paste“-fähige Lösungen, um einzelne Sensoren und Aktoren anzusteuern.

Entwickelt durch britische Stiftung „Raspberry Pi Foundation“

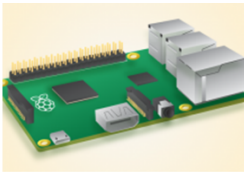
- 2012 Modell B; 700MHz, 512MB Ram
- 2013 Modell A; 700MHz, 256MB Ram
- 2014 Modell A+; 700MHz, 256MB Ram
- 2014 Modell B+; 700MHz, 512MB Ram
- 2015 Rasp. 2 Model B; 900MHz Vierkern-CPU 1GB Ram
- 2015 Raspberry Pi Zero (1GHz Einkern-CPU)
- 2016 Rasp. 3 Model B; 1,2GHz, Vierkern-CPU, 1GB Ram, 64bit, WLAN, Bluetooth



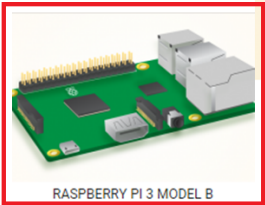
RASPBERRY PI 1 MODEL A+



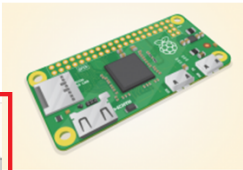
RASPBERRY PI 1 MODEL B+



RASPBERRY PI 2 MODEL B



RASPBERRY PI 3 MODEL B



RASPBERRY PI ZERO

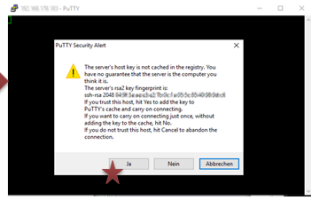
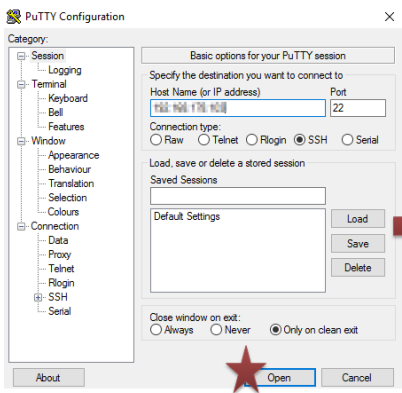
- Normale PCs haben eine Festplatte
- Der RPI nutzt hierfür eine SD-Karte $\geq 8\text{GB}$ empfohlen



- Auf dieser sind das Betriebssystem und alle sonstigen Daten gespeichert.
- Das Betriebssystem samt Anleitung zur Erstellung einer SD-Karte finden Sie unter „<https://www.raspbian.org/>“
- Im Labor ist schon alles installiert

- Normale PCs haben eine Tastatur und Bildschirm
- Grundsätzlich ist das auch beim RPI möglich (→usb, hdmi)
- Üblicher ist allerdings die Verbindung über eine Netzwerkshell (ssh)
- Unter Windows kann mit dem Programm „Putty“ (<http://www.putty.org/>) eine solche ssh-Verbindung aufgebaut werden.
- Weiteres dazu im Abschnitt „Raspberry Pi Laborübung“

Verwendung von Putty im Labor (Verbindungsaufbau)

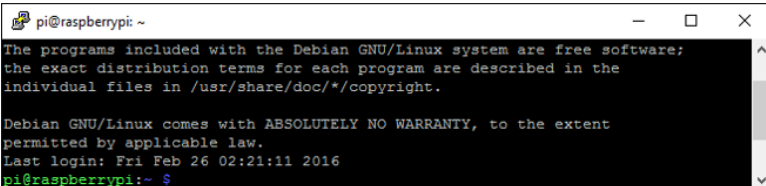


Verwendung von Putty im Labor (Login)

Benutzername: pi

Password: raspberry

Nach dem Login erscheint das Konsolenfenster. Hier können nun Linuxbefehle eingegeben werden



```
pi@raspberrypi: ~  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Fri Feb 26 02:21:11 2016  
pi@raspberrypi:~ $
```

Python

Raspberry
Pi

Laborübung

Grundlagen
EL

GPIO

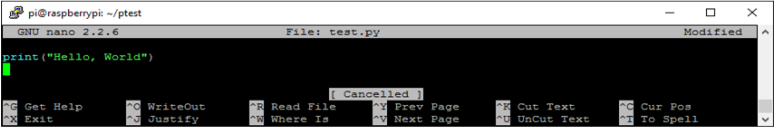
Schreiben Sie bitte mit!

- SD-Karte (Größe / Belegung)
- Verzeichnis erstellen
- Textdatei erstellen
- Textdatei editieren
- Textdatei anzeigen
- Verzeichnis anzeigen
- Verzeichnis löschen
- Textdatei löschen

Erstellen Sie zunächst eine Textdatei mit Namen `test.py`:



```
pi@raspberrypi: ~/ptest
pi@raspberrypi:~ $ mkdir ptest
pi@raspberrypi:~ $ cd ptest
pi@raspberrypi:~/ptest $ nano test.py
```



```
GNU nano 2.2.6 File: test.py Modified
print("Hello, World")
[Cancelled]
```

~G Get Help ~C WriteOut ~R Read File ~Y Prev Page ~K Cut Text ~C Cur Pos
~X Exit ~O Justify ~W Where Is ~V Next Page ~U UnCut Text ~T To Spell



```
GNU nano 2.2.6 File: test.py Modified
print("Hello, World")
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
```

Y Yes
N No ^C Cancel

Führen Sie das Programm aus:

A terminal window titled 'pi@raspberrypi: ~/ptest' with standard window controls. The prompt is 'pi@raspberrypi:~/ptest \$'. The command 'python3 test.py' has been entered and executed, resulting in the output 'Hello, World'. A new prompt 'pi@raspberrypi:~/ptest \$' is shown with a green cursor.

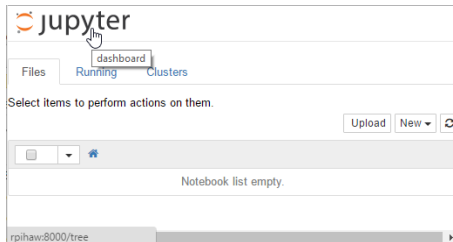
```
pi@raspberrypi: ~/ptest
pi@raspberrypi:~/ptest $ python3 test.py
Hello, World
pi@raspberrypi:~/ptest $
```

- `python3` ist der sog. Interpreter
- Dieses Programm interpretiert (führt aus) jede Zeile in Ihrem Programm
- Das Programm besteht nur aus der einen Anweisung `print("Hello, World")` → Bildschirmausgabe

Auf den RPIs im Labor ist ein spezieller Webserver installiert, über den Sie ebenfalls Python-Programme ausführen können.

Gehen Sie wie folgt vor:

- Öffnen Sie mit dem Webbrowser die Seite `http://raspi<nr>.local:8000`
 - <nr> ersetzen Sie bitte durch die auf dem Gehäuse aufgedruckte Nummer
 - :8000 bedeutet, dass der Port 8000 angewählt wird



Python

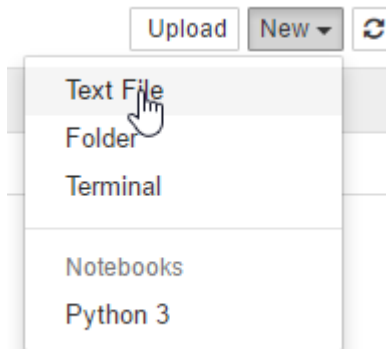
Raspberry
Pi

Laborübung

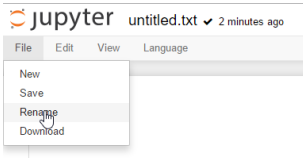
Grundlagen
EL

GPIO

Erstellen Sie eine neue Datei:

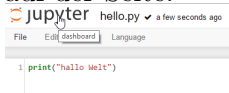


- Benennen Sie die Datei um in hello.py

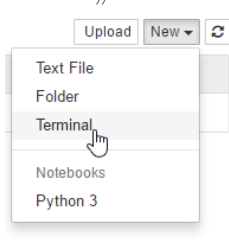


- Schreiben Sie ein einfaches Programm zur Textausgabe
- Speichern Sie mit `<strg> + <s>` oder im Menü mit File→Save

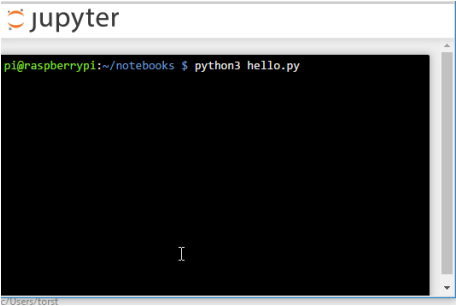
- Gehen Sie zurück zur Ausgangsseite (dashboard). Diese erreichen Sie durch ein Klick auf „jupyter“ links oben auf der Seite:



- Wählen Sie anschließend unter dem Punkt „New“ den Punkt „Terminal“:



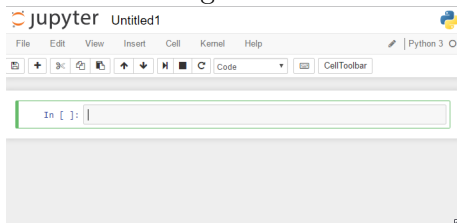
Im nun erscheinenden Terminal-Fenster können Sie das Programm gewohnt ausführen:



```
jupyter  
pi@raspberrypi:~/notebooks $ python3 hello.py  
I
```

Der einfachste Weg zur Ausführung von Programmen ist über das Notebook. Sie starten ein Notebook von der Ausgangsseite (dashboard) mit **New** → **Python3**.

Es erscheint folgendes Fenster:



- Notebooks sind organisiert in **cells**. Klicken Sie mit der Maus auf eine Zelle erscheint ein grüner Rahmen -

Der Editmode:

```
In [ ]: print("Hallo Notebook")
```

- Im Editmode können Sie die Zelle ausführen mit dem Druck auf **<shift> + <enter>**

```
In [1]: print("Hallo Notebook")
```

Hallo Notebook

```
In [ ]:
```

- Die Ausgabe wird angezeigt und eine neue Zelle im Kommandomodus (blauer Rahmen) erzeugt. Drücken Sie **<enter>**, um in den Editmode zu gelangen

```
In [1]: print("Hallo Notebook")
```

Hallo Notebook

```
In [ ]: |
```

Wichtige Hinweise zum Arbeiten mit dem Notebook:

- Im Hintergrund läuft immer der selbe Python-Interpreter.
- Das bedeutet, dass Variablen etc. aus vorhergehenden Programmabläufen vorhanden bleiben.
- Sie können den Interpreter neu starten, wenn Sie aus dem Menü **Kernel -> Restart** wählen.
- Deswegen: Verwenden Sie das Notebook für kleine Tests und schreiben Sie richtige Programme in eine **.py**-Datei und führen Sie dieses über das Terminal aus.

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

① Python

② Raspberry Pi

③ Grundlagen Elektronik

Der elektrische Stromkreis

Das OHMsche Gesetz

Kirchhoffsche Gesetze

LED an einen Computer

Tasteranschluss an einen Computer

④ GPIO

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

<http://starthardware.org/>

- Sehr schöne Einführung in Elektronik mit Videos

Nachbereitung zur Vorlesung

Lesen Sie bis zum nächsten Mal:

- <http://starthardware.org/lektion-3-stromkreis/>
- <http://starthardware.org/lektion-4-unser-erster-eigener-stromkreis/>
- <http://starthardware.org/lektion-5-wie-funktioniert-das-breadboard/>
- <http://starthardware.org/lektion-6-digital-out-vorbereitung/>
- <http://starthardware.org/lektion-6-digital-out-vorbereitung/>

Python

Raspberry
Pi

Grundlagen
EL

**Der elektrische
Stromkreis**

Das OHMsche
Gesetz

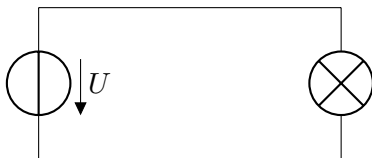
Kirchhoffsche
Gesetze

LED

Taster

GPIO

- Gegeben sei dieser Stromkreis:
- Bestehend aus



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

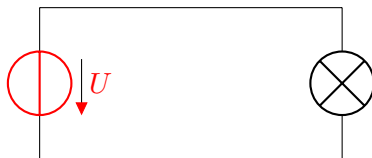
Kirchhoffsche
Gesetze

LED

Taster

GPIO

- Gegeben sei dieser Stromkreis:
- Bestehend aus
 - Spannungsquelle



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

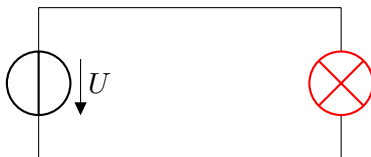
Kirchhoffsche
Gesetze

LED

Taster

GPIO

- Gegeben sei dieser Stromkreis:
- Bestehend aus
 - Spannungsquelle
 - Verbraucher (Lampe in diesem Fall)



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

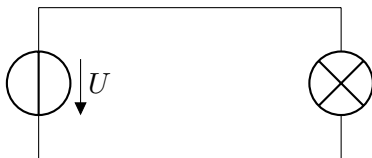
Kirchhoffsche
Gesetze

LED

Taster

GPIO

- Gegeben sei dieser Stromkreis:
- Bestehend aus
 - Spannungsquelle
 - Verbraucher (Lampe in diesem Fall)
 - Verbindungsleitungen



Python

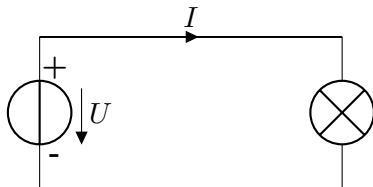
Raspberry
PiGrundlagen
ELDer elektrische
StromkreisDas OHMsche
GesetzKirchhoffsche
Gesetze

LED

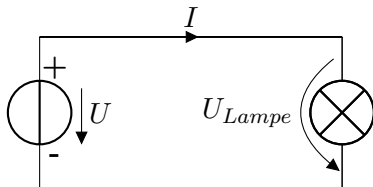
Taster

GPIO

- Gegeben sei dieser Stromkreis:
- Bestehend aus
 - Spannungsquelle
 - Verbraucher (Lampe in diesem Fall)
 - Verbindungsleitungen
 - Strom fließt von $+$ nach $-$. Ausgehend von der Spannungsquelle **durch** den Verbraucher.



- Gegeben sei dieser Stromkreis:
- Bestehend aus
 - Spannungsquelle
 - Verbraucher (Lampe in diesem Fall)
 - Verbindungsleitungen
 - Strom fließt von $+$ nach $-$. Ausgehend von der Spannungsquelle **durch** den Verbraucher.
 - Der Stromfluss sorgt für einen *Spannungsabfall* über dem Verbraucher.
 - Die Lampe leuchtet. Es wird Leistung $P = U_{Lampe} \cdot I$ umgesetzt.



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

① Python

② Raspberry Pi

③ Grundlagen Elektronik

Der elektrische Stromkreis

Das OHMsche Gesetz

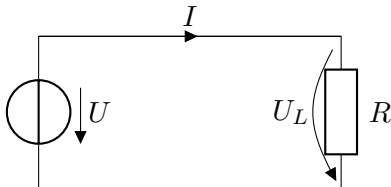
Kirchhoffsche Gesetze

LED an einen Computer

Tasteranschluss an einen Computer

④ GPIO

Der elektrische Stromkreis mit Ohmschem Verbraucher



Das Ohmsche Gesetz

- In einem Stromkreis mit Widerstand gilt:
- $U = I \cdot R$ bzw. $I = \frac{U}{R}$ bzw. $R = \frac{U}{I}$

Spannungen

- Die Spannungsquelle ist direkt mit dem Widerstand verbunden. Daher gilt $U = U_L$.

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

① Python

② Raspberry Pi

③ Grundlagen Elektronik

Der elektrische Stromkreis

Das OHMsche Gesetz

Kirchhoffsche Gesetze

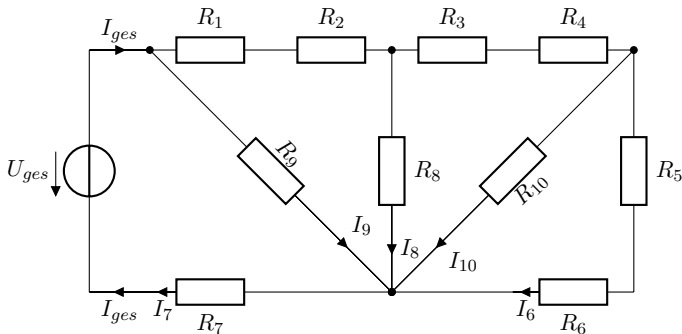
LED an einen Computer

Tasteranschluss an einen Computer

④ GPIO

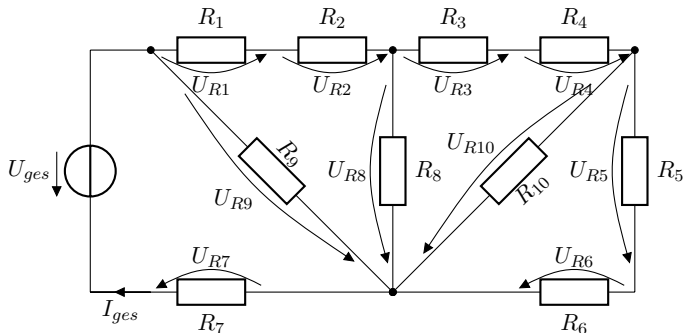
Die Knotenregel (1. Kirchhoffsche Gesetz)

- In einem Knotenpunkt ist die Summe aller Ströme gleich null.
- Beispiel $I_6 - I_7 + I_8 + I_9 + I_{10} = 0A$
Entsprechend $I_{ges} = I_7 = I_6 + I_8 + I_9 + I_{10}$



Die Maschenregel (2. Kirchhoffsche Gesetz)

- In einer geschlossenen Masche ist die Summe der Spannungen gleich Null.

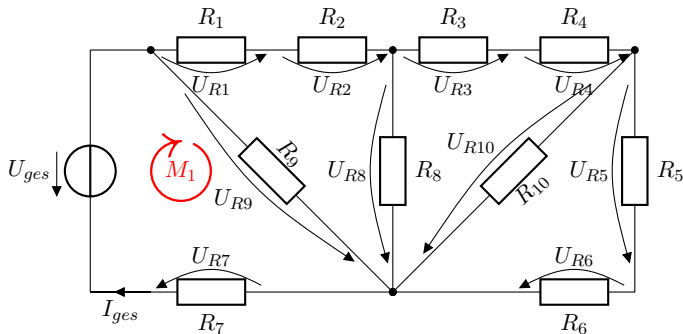


Die Maschenregel (2. Kirchhoffsche Gesetz)

- In einer geschlossenen Masche ist die Summe der Spannungen gleich Null.

- Beispiel für M_1

$$U_{R9} + U_{R7} - U_{ges} = 0V$$



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

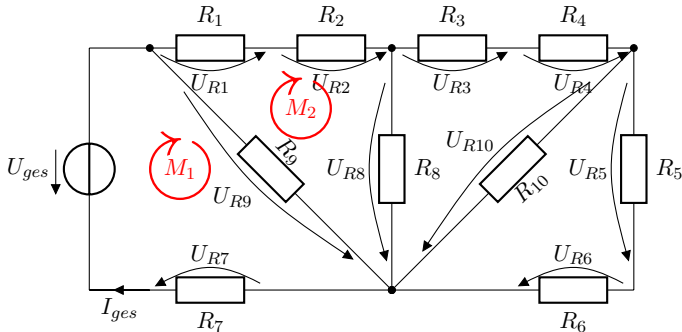
GPIO

Die Maschenregel (2. Kirchhoffsche Gesetz)

- In einer geschlossenen Masche ist die Summe der Spannungen gleich Null.

- Beispiel für M_2

$$U_{R1} + U_{R2} + U_{R8} - U_{R9} = 0V$$

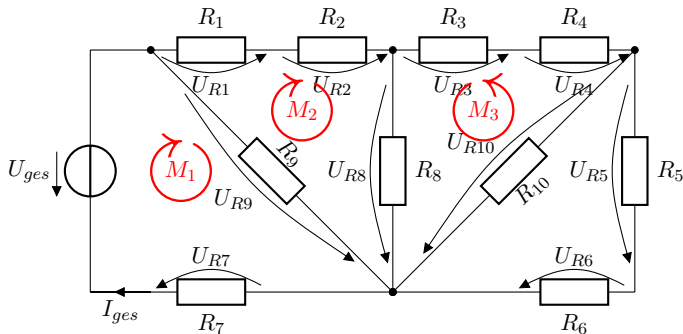


Die Maschenregel (2. Kirchhoffsche Gesetz)

- In einer geschlossenen Masche ist die Summe der Spannungen gleich Null.

- Beispiel für M_3

$$-U_{R3} - U_{R4} - U_{R10} + U_{R8} = 0V$$

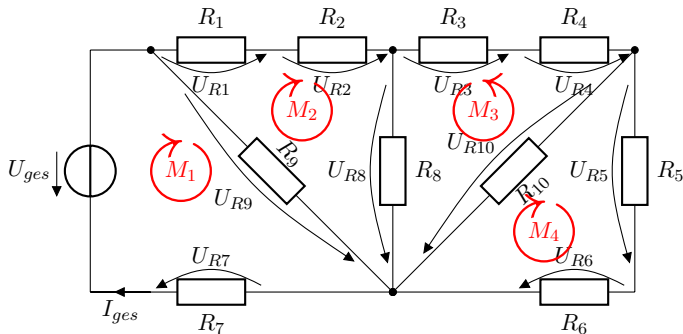


Die Maschenregel (2. Kirchhoffsche Gesetz)

- In einer geschlossenen Masche ist die Summe der Spannungen gleich Null.

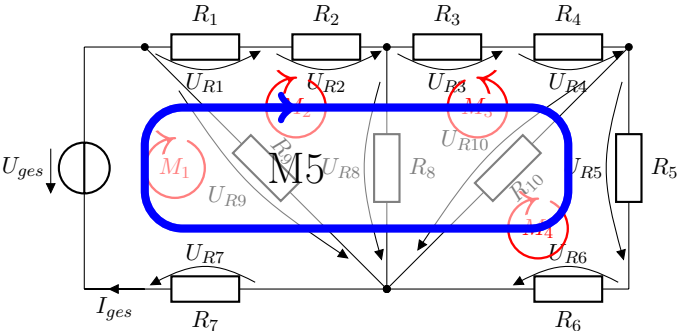
- Beispiel für M_4

$$U_{R5} + U_{R6} - U_{R10} = 0V$$



Die Maschenregel (2. Kirchhoffsche Gesetz)

- In einer geschlossenen Masche ist die Summe der Spannungen gleich Null.
- Auch große Maschen sind möglich wie $M5$
$$-U_{ges} + U_{R1} + U_{R2} + U_{R3} + U_{R4} + U_{R5} + U_{R6} + U_{R7} = 0V$$



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

① Python

② Raspberry Pi

③ Grundlagen Elektronik

Der elektrische Stromkreis

Das OHMsche Gesetz

Kirchhoffsche Gesetze

LED an einen Computer

Tasteranschluss an einen Computer

④ GPIO

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

Ziel

- Verwendung einer LED (Light Emitting Diode, Lichtaussendende Diode) als Lichtquelle.
- Nutzung von Software für den Wechsel zwischen zwei Zuständen:
 - Licht an
 - Licht aus

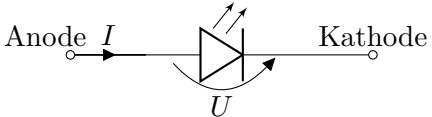
Umsetzung

- Verwendung eines Computers mit digitalem Ausgang.

Fragen

- Wie verbindet man einen Computer mit einer LED?

- Die LED ist ein elektronisches Bauteil, welches bei Stromfluss leuchtet.
- Eine LED hat zwei Anschlüsse
 - Anode (Plus)
 - Kathode (Minus)
- Schaltzeichen:



- Reale Bauteile:



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

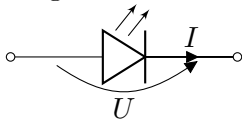
Kirchhoffsche
Gesetze

LED

Taster

GPIO

Frage: Unter welchen Voraussetzungen leuchtet eine LED?



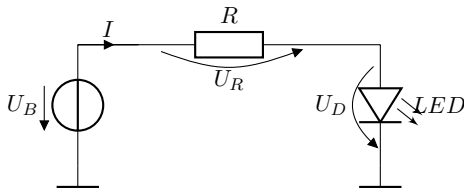
Typische Werte:

rot: 20mA@2.1V
grün: 20mA@2.1V
blau: 20mA@2.9V

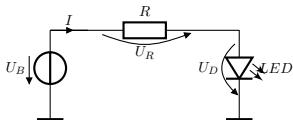
Überschreiten Sie NIE die
Werte für die Spannung!!!

Verwenden Sie immer einen
Vorwiderstand

- Die folgende Schaltung zeigt den Anschluss einer LED an einer Spannungsquelle (U_B)
- Der Strom I ist durch alle Bauteile gleich (Reihenschaltung)
- Die Spannungen teilen sich auf, so dass gilt
$$U_B = U_R + U_D$$



- Der Hersteller einer LED gibt den Nennstrom und Normspannung an.



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

Beispielwerte

- Angabe des LED Herstellers (Datenblatt) $U_D = 2.1V$ und $I_D = 20mA$
- $U_B = 5V$

Lösung für den Widerstandswert R

- $U_R = U_B - U_D = 5V - 2.1V = 2.9V$
- $I_R = I_D = I$
- $R = \frac{U_R}{I_R} = \frac{2.9V}{20mA} = 145\Omega$

Anschluss einer LED am Digital-Pin eines Computers

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

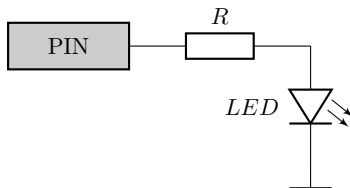
Kirchhoffsche
Gesetze

LED

Taster

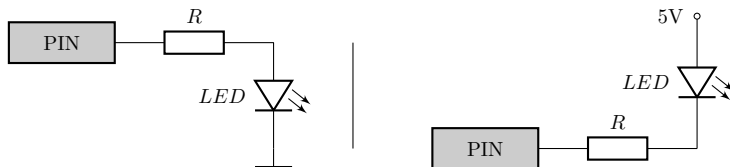
GPIO

- Beim Anschluss an einen Digital-Pin können zwei Zustände (LOW=0V und HIGH=5V) über Software eingestellt werden.
- Wann leuchtet die LED?
- Berechnen Sie den Vorwiderstand für eine LED mit (20mA@2.4V)
- Berechnen Sie den Vorwiderstand für eine LED mit (2mA@2.6V)



Anschluss einer einfachen LED (III)

- Worin unterscheiden sich die beiden Schaltungen?



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

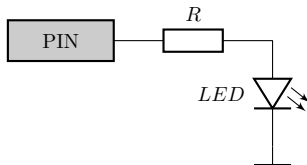
LED

Taster

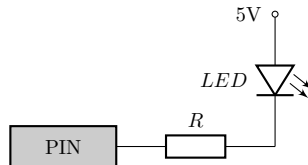
GPIO

Anschluss einer einfachen LED (III)

- Worin unterscheiden sich die beiden Schaltungen?



Active HIGH



Active LOW

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

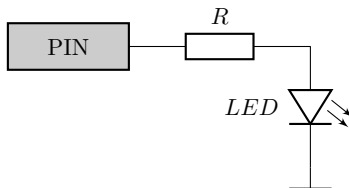
Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO



- Kann man bei dieser Schaltung auch die Helligkeit der LED einstellen?

Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

① Python

② Raspberry Pi

③ Grundlagen Elektronik

Der elektrische Stromkreis

Das OHMsche Gesetz

Kirchhoffsche Gesetze

LED an einen Computer

Tasteranschluss an einen Computer

④ GPIO

Python

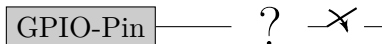
Raspberry
PiGrundlagen
ELDer elektrische
StromkreisDas OHMsche
GesetzKirchhoffsche
Gesetze

LED

Taster

GPIO

- Die Eingabe durch einen Taster (Schließen eines elektrischen Kontakts bei Betätigung) ist die einfachste Art der Eingabe in ein Computersystem.
- Wie schließt man einen solchen Taster an?



Python

Raspberry
Pi

Grundlagen
EL

Der elektrische
Stromkreis

Das OHMsche
Gesetz

Kirchhoffsche
Gesetze

LED

Taster

GPIO

- Taster besitzen zwei Zustände
- IO-Pins können zwei Zustände „lesen“
- Wie bringt man beides zusammen?
- Ein IO-Pin im Eingangsmodus erkennt (3.3V System)
 - Eine Null bei Spannungen von 0V-0.8V
 - Eine Eins bei Spannungen von 2.3V-3.3V

Python

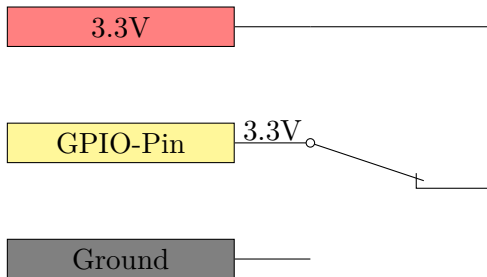
Raspberry
PiGrundlagen
ELDer elektrische
StromkreisDas OHMsche
GesetzKirchhoffsche
Gesetze

LED

Taster

GPIO

- Verwendung eines Tasters
- Direkte Verbindung mit 3.3V → Logisch 1



Python

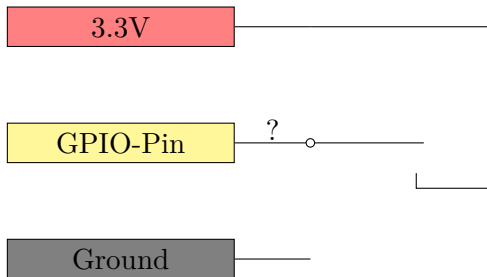
Raspberry
PiGrundlagen
ELDer elektrische
StromkreisDas OHMsche
GesetzKirchhoffsche
Gesetze

LED

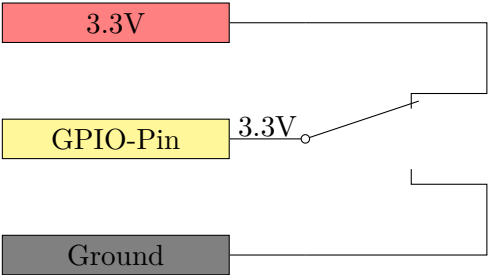
Taster

GPIO

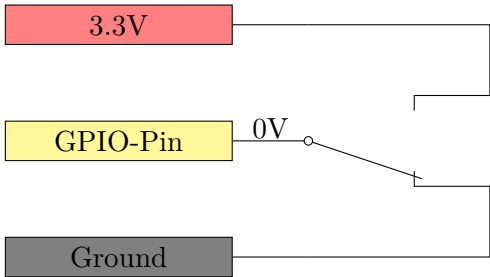
- Verwendung eines Tasters
- Offener Eingangsport \rightarrow undefinierter Zustand



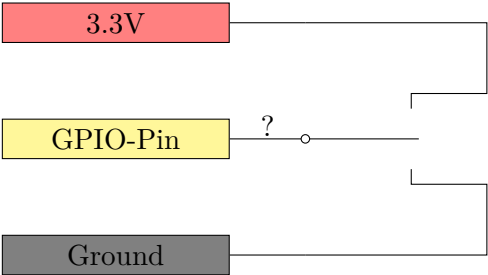
- Verwendung eines Wechseltasters
- Direkte Verbindung mit 3.3V → Logisch 1



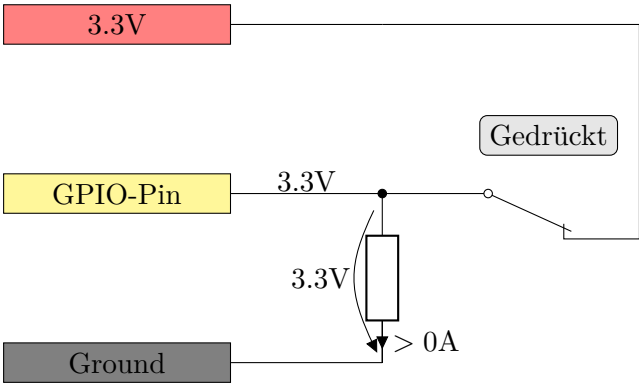
- Verwendung eines Wechseltasters
- Direkte Verbindung mit 0V → Logisch 0



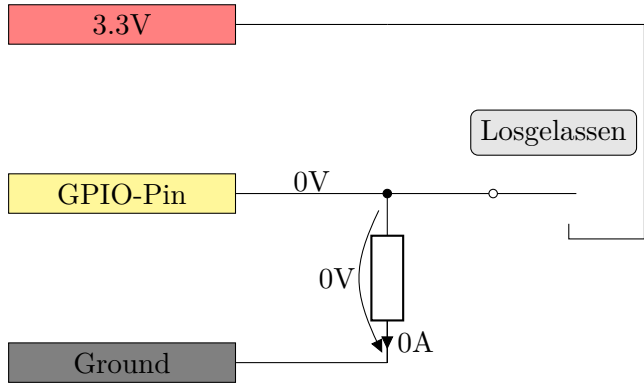
- Verwendung eines Wechseltasters
- Offener Eingangsport → undefinierter Zustand



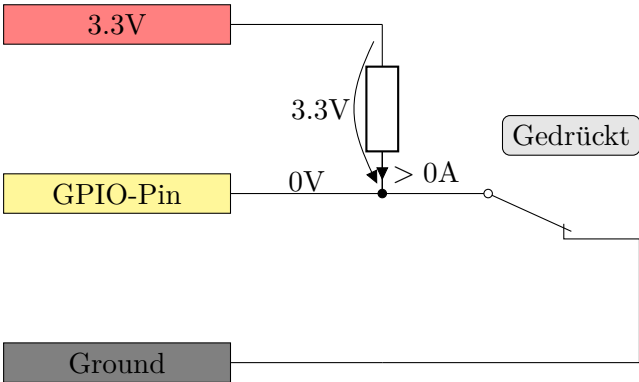
- Verwendung eines Pulldown-Widerstand
- Direkte Verbindung mit 3.3V → Logisch 1. Durch den Widerstand fließt ein Strom.



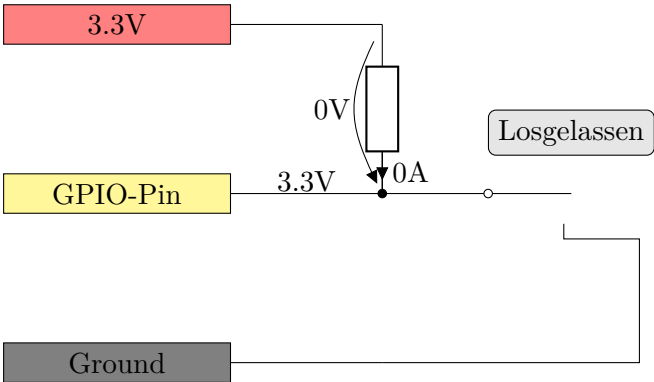
- Verwendung eines Pulldown-Widerstand
- Weiterhin Verbindung über Widerstand zu Ground → Logisch 0. Durch den Widerstand fließt kein Strom.



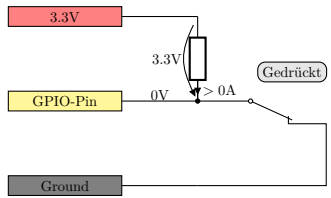
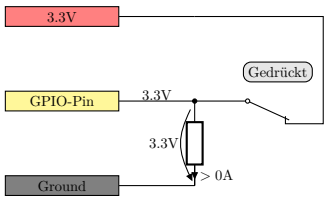
- Verwendung eines Pullup-Widerstand
- Direkte Verbindung mit Ground (0V) → Logisch 0.
Durch den Widerstand fließt ein Strom.



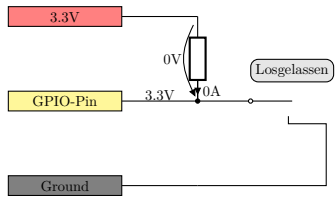
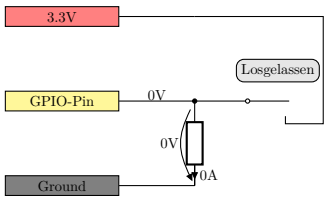
- Verwendung eines Pullup-Widerstand
- Weiterhin Verbindung über Widerstand zu 3.3V → Logisch 1. Durch den Widerstand fließt kein Strom.



- Verwendung von Pullup- oder Pulldown-Widerstand unterscheiden sich in der Polarität des Auslesens.
- In der Regel liegt der Widerstandswert zwischen $1k\Omega$ und $10k\Omega$.



- Verwendung von Pullup- oder Pulldown-Widerstand unterscheiden sich in der Polarität des Auslesens.
- In der Regel liegt der Widerstandswert zwischen $1k\Omega$ und $10k\Omega$.



Python

Raspberry
Pi

Grundlagen
EL

GPIO

Laboraufgabe

① Python

② Raspberry Pi

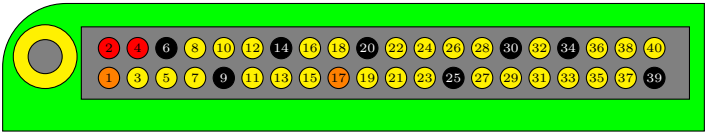
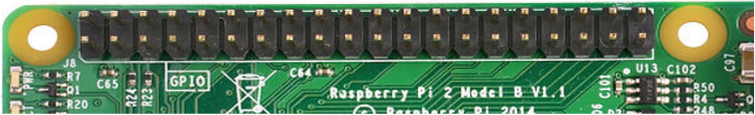
③ Grundlagen Elektronik

④ GPIO

Laboraufgabe

Anschluss von Peripherie an den RPI

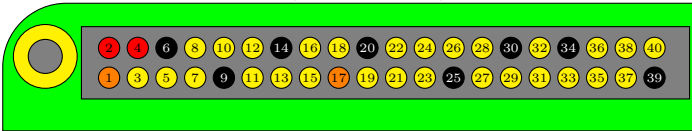
Neben den gängigen Schnittstellen (usb, hdmi, ethernet, etc.) bietet der Raspberry Pi die Möglichkeit gängige Elektronikschnittstellen (SPI, I2C, RS232, GPIO) zu verwenden. Dafür steht ein 40 Pin Sockel zur Verfügung:



● 5V ● 3.3V ● Ground (0V) ● GPIO

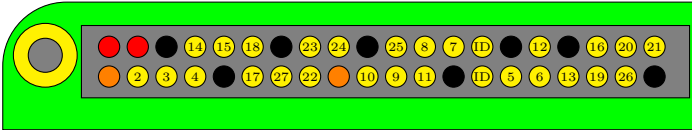
- Auf dem RPI gibt es zwei verschiedene Nummernsysteme für die Programmansteuerung der Stiftleiste:

- BOARD: `GPIO.setmode(GPIO.BOARD)`



● 5V ● 3.3V ● Ground (0V) ● GPIO

- BCM: `GPIO.setmode(GPIO.BCM)`



● 5V ● 3.3V ● Ground (0V) ● GPIO

Python

Raspberry
Pi

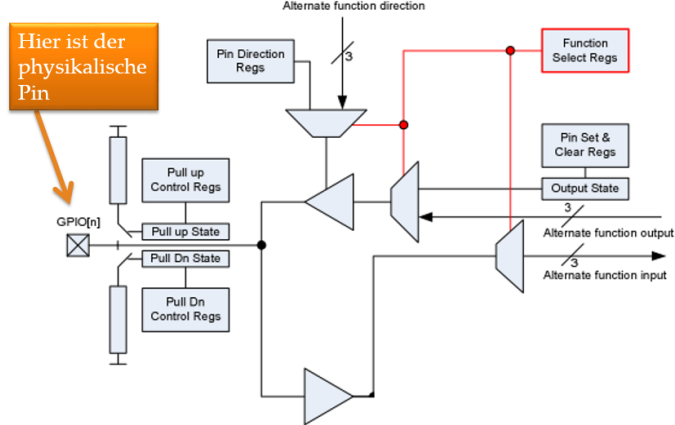
Grundlagen
EL

GPIO

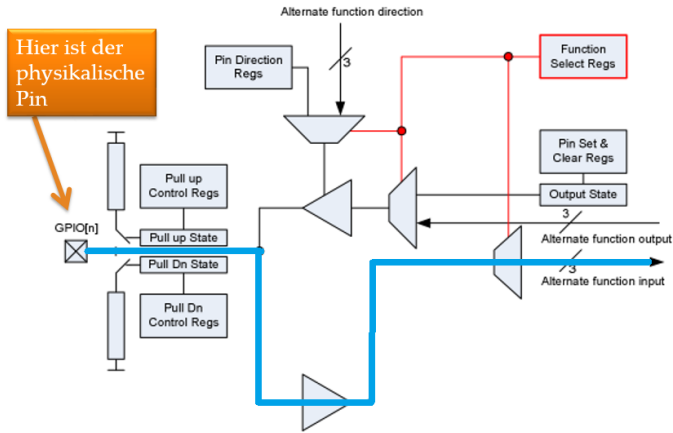
Laboraufgabe

- Diese Schnittstelle ist eine Möglichkeit mit der Außenwelt Kontakt aufzunehmen.
- 27 Inputs / Outputs
 - Die Richtung wird in Software festgelegt
- Spannungspegel für Ausgänge (Outputs)
 - LOW : 0V
 - HIGH : 3.3V
- Spannungspegel für Eingänge (Inputs)
 - Erlaubter Spannungsbereich (!0...3.3V!)
 - LOW < 0.8V
 - HIGH > 2.0V

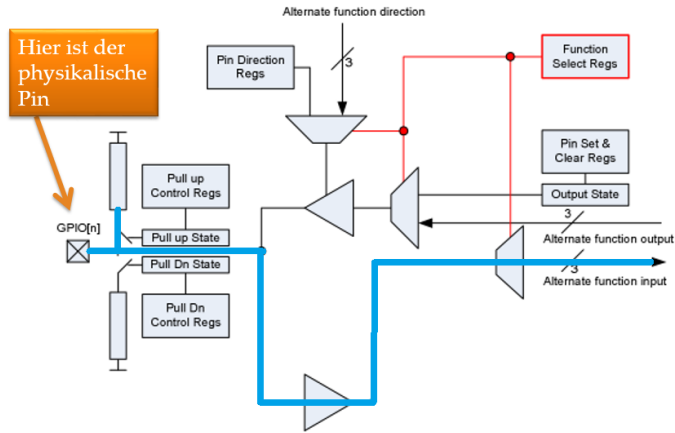
Dies ist der interne Aufbau eines einzelnen GPIO-Pins:



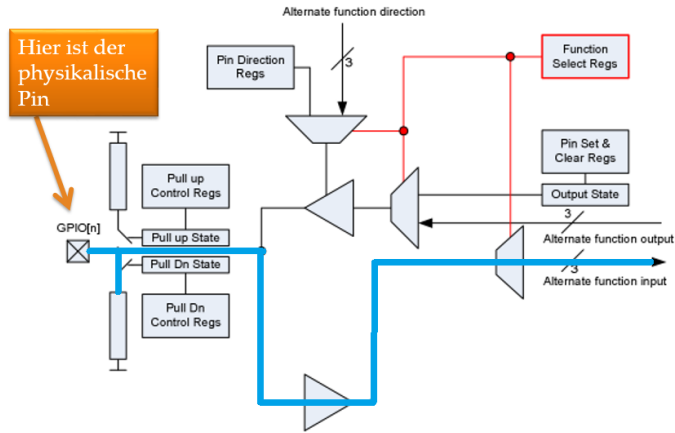
Beispiel für die Konfiguration als Eingangspin:



Beispiel für die Konfiguration als Eingangspin mit Pullup:



Beispiel für die Konfiguration als Eingangspin mit Pulldown:

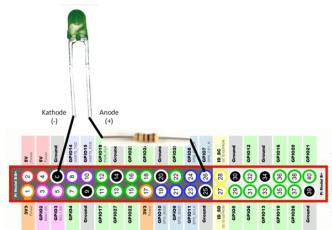
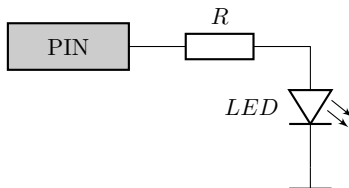


Anschluss einer einfachen LED (I)

```

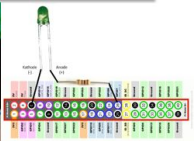
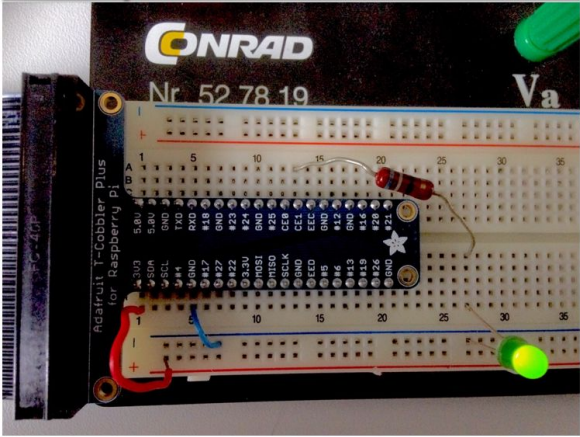
1 import RPi.GPIO as GPIO    # GPIO Bibliothek importieren
2 import time                 # Modul time importieren
3
4 GPIO.setmode(GPIO.BOARD)    # Verwende Board-Pinnummern
5 GPIO.setup(26, GPIO.OUT)     # Setze Pin 26 (GPIO7) als Ausg
6 GPIO.output(26, True)       # Lege 3.3V auf Pin 26
7 time.sleep(0.5)             # Warte 500ms
8 GPIO.output(26, False)      # Lege 0V auf Pin 26
9 GPIO.cleanup()              # Aufräumen

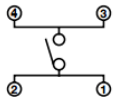
```



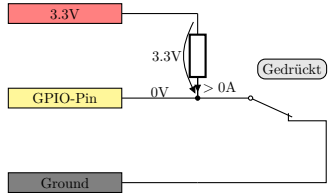
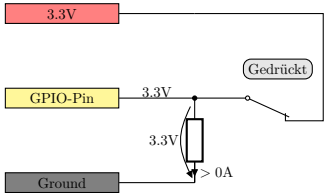
Anschluss einer einfachen LED (II)

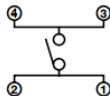
Und so sieht es im Labor aus:



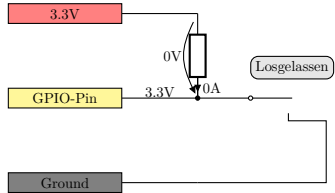
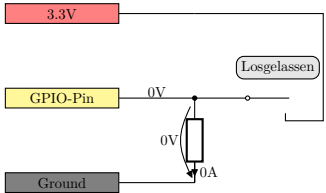


- Verwendung von Pullup- oder Pulldown-Widerstand unterscheiden sich in der Polarität des Auslesens.
- In der Regel liegt der Widerstandswert zwischen $1k\Omega$ und $10k\Omega$.





- Verwendung von Pullup- oder Pulldown-Widerstand unterscheiden sich in der Polarität des Auslesens.
- In der Regel liegt der Widerstandswert zwischen $1k\Omega$ und $10k\Omega$.



Anschluss eines Tasters (II)

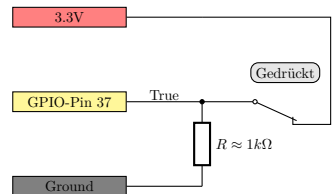
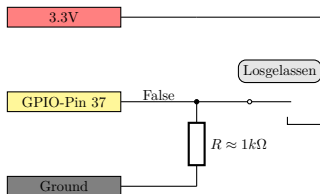
Python

Raspberry
PiGrundlagen
EL

GPIO

Laboraufgabe

```
1 import RPi.GPIO as GPIO # GPIO Bibliothek importieren
2 import time              # Modul time importieren
3
4 P = 37                   # Verwende Pin 37
5
6 GPIO.setmode(GPIO.BOARD) # Verwende Board-Pinnummern
7 GPIO.setup(P, GPIO.IN)   # Setze Pin P als Eingang
8
9 for i in range(100):     # 100 mal laufen
10     value = GPIO.input(P) # Wert auslesen
11     print(value)          # Wert ausgeben
12     time.sleep(0.1)       # 100ms warten
13
14 GPIO.cleanup()          # Aufräumen
```



Python

Raspberry
Pi

Grundlagen
EL

GPIO
Laboraufgabe

- ❶ Schließen Sie eine LED am Raspberry-Pi an und lassen Sie diese blinken.
- ❷ Schließen Sie einen Taster am Raspberry-Pi an und zeigen Sie den Wert auf der Konsole an.
- ❸ Lassen Sie die LED bei jeden Tastendruck 10 mal schnell blinken.