



1

Agenda

SIEMENS
Ingenuity for life

- Teil 1: Grundlagen
 - IoT
 - Elektronik
- Teil 2: Vorstellung Raspberry Pi mit Beispielen und Übungen
 - Grundlagen Raspberry Pi Zero W{H}
 - HATs und Sensoren
 - Cloud Computing: ThingSpeak, IFTTT, Node-RED
- Teil 3: Mini Hackathon - DiY

2

Sachdienlicher Hinweis: Git-Repository für das Tutorium

SIEMENS
Ingenuity for life

- Git-Repository
 - <https://github.com/ms1963/IoTDeeperDiveOOP2020.git>
- Klonen auf dem Raspberry Pi mit dem Kommando:
 - `git clone https://github.com/ms1963/IoTDeeperDiveOOP2020.git`



- Das Repository enthält
 - Quelldateien der Lösungen => src
 - Datasheets/Informationen => datasheets
 - Handouts der vorliegenden Folien => handouts
 - In datasheets befindet sich auch ein Bild mit dem GPIO-Header-Layout vom Raspberry Pi {ZERO }

© Michael Stal,2020
Page 3

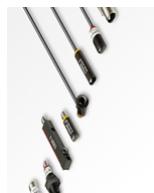
3

IoT Geräte und Komponenten

SIEMENS
Ingenuity for life

Wichtige Themen:

- Konzeptioneller Aufbau
- Autonomes Embedded Gerät basierend auf Microcontroller/CPU
- Sensoren und Aktuatoren zum Beobachten und Interagieren mit der Umgebung
- Integration in Mesh-Netzwerke (lokal) / Internet (global)
- Lokale Verarbeitung (Edge) und/oder am Server
- Kommunikation mit Cloud-Diensten (Melden/Empfangen von Ereignissen/Daten/Kommandos)



© Michael Stal,2020
Page 4

4

IoT Praxis in diesem Tutorium

SIEMENS
Ingenuity for life

Hardware

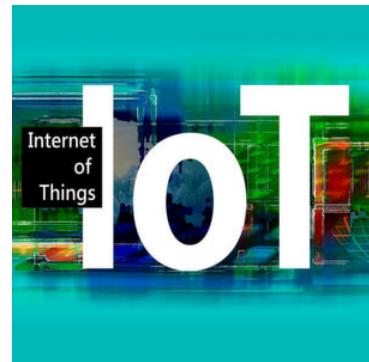
Raspberry Pi Zero WH plus Netzteil
BME280-Sensor von Bosch
PIR – Bewegungserkennungssensor
PCF8591 A/D Wandler Board
Breadboard, Widerstände, Drähte/Kabeln, LEDs, Push Button
Raspberry Pi Cam
Micro-USB auf USB-A Kabel

Software

Visual Studio Code
Raspberry Pi
Python und Bibliotheken

Dienste

Clouddienst ThingSpeak, um Messwerte zu visualisieren



© Michael Stal, 2020
Page 5

5

Kommunikation Protokolle



Zum einen Nutzung von TCP/IP und HTTP. Lokal:
ZigBee, Remote: LoRa



Besser: anwendungsnahe standardisierte Protokolle



Im IoT gibt es
zwei verbreitete
Standards

MQTT
REST/CoAP

© Michael Stal, 2020
Page 6

6

Verwendung leichtgewichtiger Kommunikations-Protokolle

Nachrichtenbasierte Kommunikation:
Versenden von Nachrichten 1:1, 1:n

Ereignisgetriebene Kommunikation: n:m
Ereignismeldungen mit Publish/Subscribe-Modell

Dateitransfers

Streaming von Audio/Video/Echtzeitdaten

SIEMENS
Ingenuity for life

Unterstützte Semantiken
Asynchrone Kommunikation über Nachrichten/Ereignisse
Dienstgüten (Quality of Service):
Garantiert eine Zustellung
Nach bestem Bemühen
Mindestens eine Zustellung

Beispiele für Protokolle/Standards
Internetkommunikation: RESTful, CoAP
Eventing: DDS
Messaging: MQTT
Short Range/Low Energy: Bluetooth, NFC, Zigbee, ...
On-Device: I ² C, CAN, Profibus, ...)

© Michael Stal, 2020
Page 7

Gerätetopologien

1:1 Konfiguration: Client connected to IoT Gerät.

Bus: IoT Geräte connected to a central Bus.

Baum: IoT Geräte organized into a hierarchical tree structure under Gesamt system, which is further divided into Gruppe 1 and Gruppe 2.

Backend: IoT Geräte 1, 2, and 3 connected to a central Backend.

Statische Konfiguration: IoT Geräte 1, 2, and 3 connected to a central Backend.

Dynamische Vernetzung: IoT Geräte X, Y, and Z connected to each other in a peer-to-peer network.

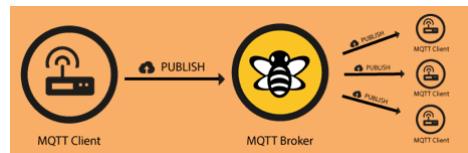
SIEMENS
Ingenuity for life

© Michael Stal, 2020
Page 8

MQTT

SIEMENS
Ingenuity for life

- MQTT hieß ursprünglich MQ Telemetry Transport. Heute ist es ein Standard: OASIS- bzw. ISO Standard (ISO/IEC PRF 20922)
 - Folgt dem Publisher/Subscriber Pattern
 - Erfordert einen MQTT Broker, um Nachrichten und Topics zu speichern und zu verwalten
 - Dienstgüten (Quality of Service): fire'n forget, at-least-once, exactly-once
 - Nachrichten: volatile oder persistent, last will, good known state
 - Benutzt verschiedene Ports abhängig von den Sicherheitsanforderungen (Nicht gesicherter Port: 1883)
 - Ignoriert Nachrichteninhalte



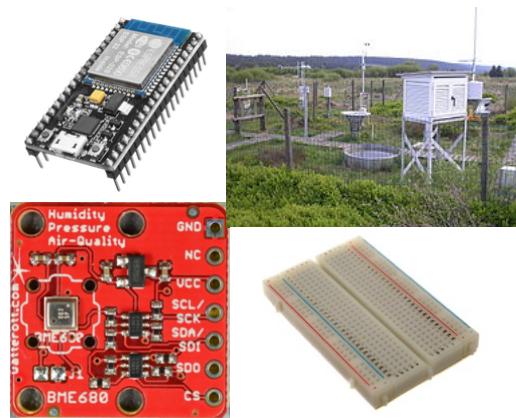
© Michael Stal, 2020
Page 9

9

Projektbeispiel: Wetterstation Light

SIEMENS
Ingenuity for life

Embedded Microcontroller, der über Sensorik Wetterdaten erfassst (Temperatur, Luftdruck, Feuchtigkeit, Luftqualität, ...)
Diese Messungen erfolgen periodisch. Die Daten sendet der Controller jeweils über WiFi (oder LoRa, SigFox) an einen Clouddienst
Mit einem Webbrowser lassen sich die Messdaten und ihr zeitlicher Verlauf beobachten

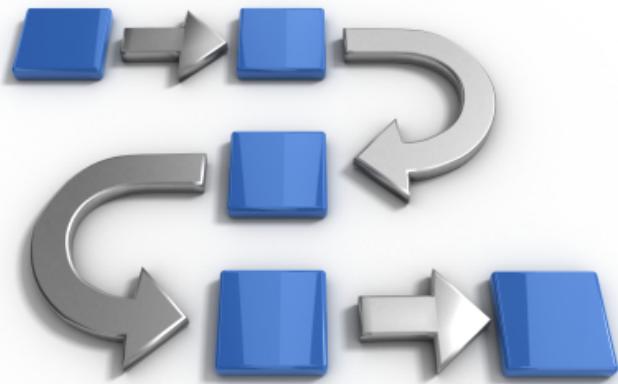


© Michael Stal, 2020
Page 10

10

Elektronik Grundlagen

SIEMENS
Ingenuity for life

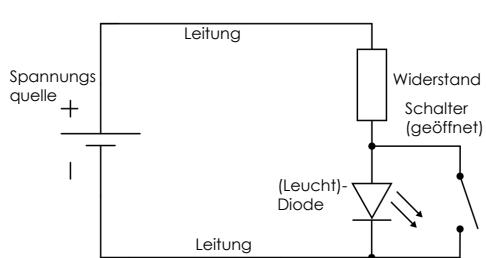


© Michael Stal, 2020
Page 11

11

Typische Elemente einer Schaltung

SIEMENS
Ingenuity for life



Batterie/Zelle/Netzdose/USB-Ausgang:
Spannungsquelle
Lampe, Widerstand, Diode: Verbraucher
Schalter: Um Verbindung herzustellen/zu trennen
Leitungen verbinden Spannungsquellen und Verbraucher
Strom fließt nur in geschlossenen Stromkreisen
Grundannahme: Wir benutzen nur „Independent Voltage Sources“ (erzeugen konstante Spannung)
• Es gibt auch „Independent Current Sources“!

© Michael Stal, 2020
Page 12

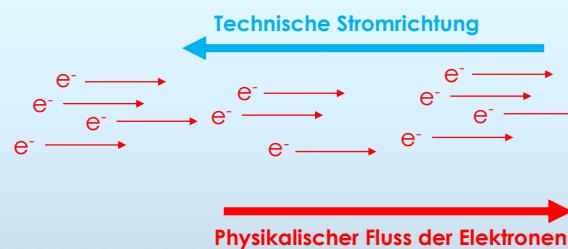
12

Stromflussrichtung

SIEMENS
Ingenuity for life

Im Gegensatz zur Physik (Flussrichtung der Elektronen) fließt Strom technisch von

der Anode (+) zur Kathode (-)



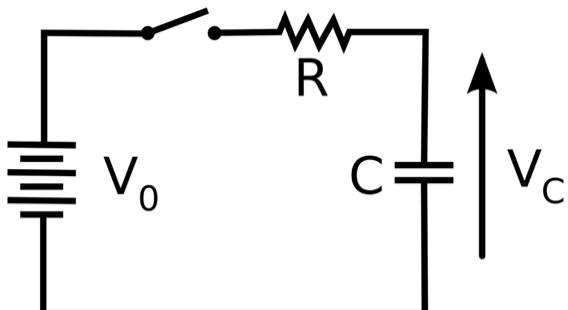
© Michael Stal, 2020
Page 13

13

Widerstand ist nicht zwecklos

SIEMENS
Ingenuity for life

- **Problem:** Verbraucher, die nur eine gewisse Stromstärke tolerieren
- **Lösung:** Widerstände als Verbraucher (Wandlung von Strom in Wärme) zur Reduktion der Stromstärke
- Alle Teile einer Schaltung haben Widerstand, sogar Leitungen (meistens vernachlässigbar)
- Ein geöffneter Schalter hat unendlichen Widerstand



© Michael Stal, 2020
Page 14

14

Ohm's Gesetz

$$\text{Spannung} = \text{Widerstand} \times \text{Stromstärke}$$

$$U = R \times I$$

Serielle Schaltung: $R_{\text{seriell}} = R_1 + R_2 + \dots$

Hinweis: Es ist allgemeine Praxis, Widerstände zu kombinieren, um sich einem gewünschten Widerstandswert zu nähern

Parallelenschaltung: $\frac{1}{R_{\text{parallel}}} = \frac{1}{R_1} + \frac{1}{R_2} + \dots$



Gleiche Stromstärke, Verschiedene Spannungen



Gleiche Spannung, Verschiedene Stromstärken



© Michael Stal, 2020
Page 15

15

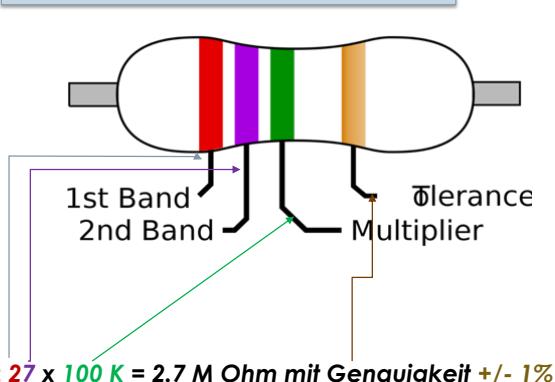
Herr der Ringe

Farbe	Multiplikator	Toleranz
Braun	$\times 10$	$\pm 1\%$
Rot	$\times 100$	$\pm 2\%$
Orange	$\times 1 K$	
Gelb	$\times 10 K$	
Grün	$\times 100 K$	$\pm .5\%$
Blau	$\times 1 M$	$\pm .25\%$
Violett	$\times 10 M$	$\pm .1\%$
Grau		$\pm .05\%$
Gold	$\times .1$	$\pm 5\%$
Silber	$\times .01$	$\pm 10\%$

Farbcodierung:

- █ 0
- █ 1
- █ 2
- █ 3
- █ 4
- █ 5
- █ 6
- █ 7
- █ 8
- █ 9

3 Ringe: Ring 1,2 sind Zahlenwerte, Ring 3 ist Multiplikator, Zusatzring: Genauigkeit
 4 Ringe: Ringe 1,2,3 sind Zahlenwerte, Ring 4 ist Multiplikator, Zusatzring: Genauigkeit.



Beispiel 3 Ringe: $27 \times 100 K = 2.7 M \Omega$ mit Genauigkeit $\pm 1\%$

© Michael Stal, 2020
Page 16

16

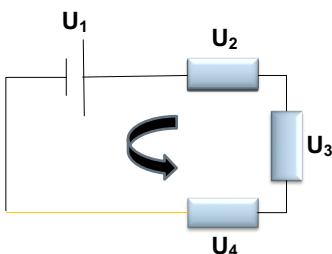
Kirchhoff's Spannungsgesetz

SIEMENS
Ingenuity for life

Kirchhoff's Voltage Law:

In einer geschlossenen Schleife ist die Summe aller Spannungs erzeuger gleich der Summe aller Spannungsverbraucher.

Summe aller Spannungen in einer geschlossenen Schleife ist immer 0 (egal ob im oder gegen den Uhrzeigersinn aufsummiert)



In Stromrichtung (+ => -)
 Erzeuger: negative Spannung
 Verbraucher: positive Spannung
 Gegen Stromrichtung (- => +)
 Erzeuger: positive Spannung
 Verbraucher: negative Spannung

$$U_1 + U_2 + U_3 + U_4 = 0$$

© Michael Stal, 2020
Page 17

17

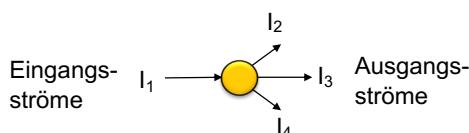
Kirchhoff's Stromstärkengesetz

SIEMENS
Ingenuity for life

Kirchhoff's Current Law:

Summe aller Eingangsströme an einem Knoten ist gleich der Summe aller Ausgangsströme.

Summe aller in einem Knoten ein- und abgehenden Ströme ist immer 0



$$\begin{aligned} I_1 &= I_2 + I_3 + I_4 \Rightarrow \\ I_1 - (I_2 + I_3 + I_4) &= 0 \end{aligned}$$

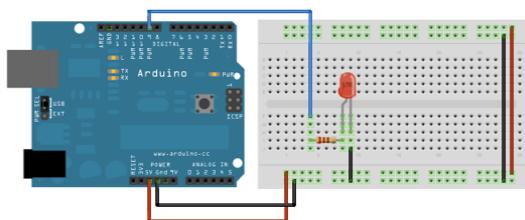
© Michael Stal, 2020
Page 18

18

Praxis

SIEMENS
Ingenuity for life

- Verbindung einer Spannungsquelle mit Raspberry Pi-Board, LED und Widerstand
- Ein Raspberries arbeitet mit 3.3V, sollte maximal 8 mA Stromstärke an einem I/O-Pin nutzen
- Die Durchlassspannung einer roten LED liegt bei 2V und die Stromstärke sollte etwa bei 8 mA liegen
- Um die restlichen 1.3V zu „konsumieren“, verwenden wir einen Widerstand (Kirchhoff's Voltage Law)
- Laut Ohm's Gesetz: $U = R * I \Rightarrow R = U / I = 1.3 V / 8 mA = 162.5 \Omega$ => Entweder 150 Ohm oder nächsthöheren Widerstand verwenden, z.B. 220 Ohm (Rot/Rot/Braun)



3-Band-Farbcodes: Rot/Rot/Braun = 220 Ohm
4-Band-Farbcodes Rot/Rot/Schwarz/Schwarz

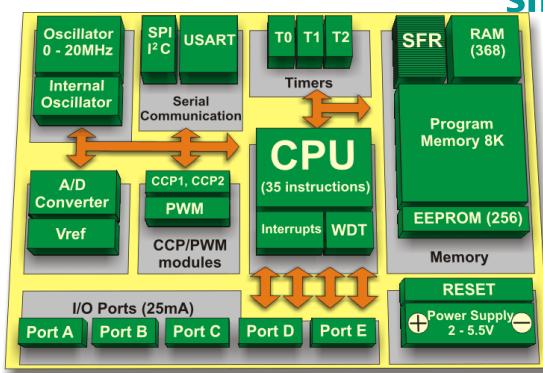
© Michael Stal, 2020
Page 19

19

Microcontroller im Überblick

SIEMENS
Ingenuity for life

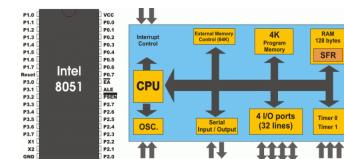
- Microcontrollers (bzw. µC, uC oder MCU) sind nicht CPUs, sondern komplett „Systems on a Chip“ (SoC) mit
 - Prozessor
 - Bussystemen
 - Speicher
 - Ein-/Auszabe-Peripherie
- Noch immer haben weit über die Hälfte der Microcontroller 8 Bits
- Meistens erfolgt die Programmierung der Embedded Systeme in C/C++, oder sogar Assembler
- Inzwischen kommen Skriptsprachen wie Python, Lua hinzu



Quelle: <http://www.mikroe.com/chapters/view/16/chapter-3-pic16f887-microcontroller/>

Geheimsprache des µC

USART	Universal Synchronous/Asynchronous Receiver/Transmitter
SPI	Serial Peripheral Interface
I²C	Inter-Integrated-Bus
SFR	Special Function Registers
WDT	WDT
PWM	PWM
CCP	Capture, Compare, and Pulse Width Modulation- PIC16 µC
Vref	Reference Voltage



© Michael Stal, 2020
Page 20

20

Sensoren und Aktuatoren

SIEMENS
Ingenuity for life

Sensoren

- Temperatur
- Gyro
- Luftdruck
- Gasgehalt
- Feuchtigkeit
- Wind
- Berührung
- Höhe
- GPS
- ...



Aktuatoren

- Motoren
- LEDs bzw. Licht
- Steckdosen
- Displays
- Lautsprecher
- Spannung
- Kommunikation



© Michael Stal, 2020
Page 21

21

HAT (steht bei Raspberry Pis für Hardware on Top)

SIEMENS
Ingenuity for life

- Raspberry Pi Hats lassen sich auf Raspberry Pi Boards aufstecken, um die Funktionalität zu erweitern
- Dafür gibt es zwei spezielle Pins zur I²C-Kommunikation (siehe spätere Folien)
- Dazu muss HAT ein EEPROM mit genaueren Daten (Herstellerinformation, Art der Erweiterung,) zur Verfügung stellen



Beispiel: Sense Hat

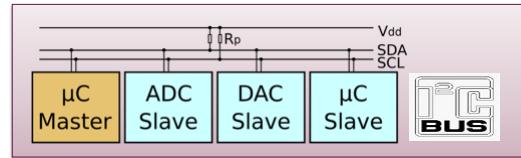
© Michael Stal, 2020
Page 22

22

I²C* – Low Speed Bussystem

SIEMENS
Ingenuity for life

Von Philips Semiconductors erfundenes Multi-Master Serial Single-ended Bussystem
In meisten Hardwareboards eingesetzt um Peripherie mit kleiner Geschwindigkeit anzuschließen (Geschwindigkeit 10 Kb/s-100Kb/s; neuere Versionen bis zu 3.6 Mb/s)
Zwei bidirektionale Open-Drain Verbindungen : Serial Data Line (SDA) und Serial Clock (SCL)
Pull-up Widerstände um fluktuierende Signale zu vermeiden
Es gibt Bibliotheken, um einfacher auf I²C zuzugreifen



Master generiert Takt und initiiert Kommunikation
Slave empfängt Takt und reagiert, sobald ihn der Master adressiert
Master sendet nur, wenn er auf der Leitung kein Start- oder Stop-Bit eines anderen Masters liest
Austausch einzelner oder mehrerer Nachrichten mit Slave

* I²C = Inter-Integrated Circuit, erfunden von Philips Semiconductor Division, heute NXP

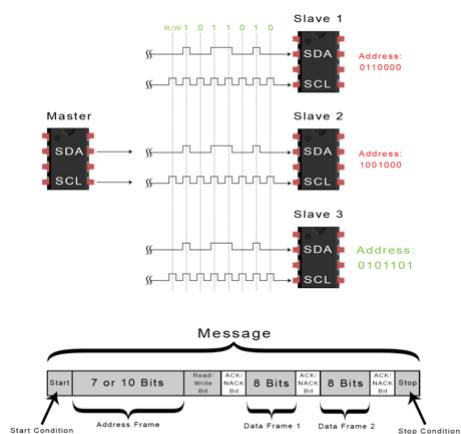
© Michael Stal, 2020
Page 23

23

I²C Kommunikation zwischen Master und Slaves (1)

SIEMENS
Ingenuity for life

- ▶ Ein Master kommuniziert mit mehreren Slaves (Multi-Master ebenfalls möglich)
- ▶ Pins SDA, SCL
- ▶ Synchronisation durch Master
- ▶ Nachrichten
 - ▶ Start Condition Bit: SDA von HIGH auf LOW, dann SCL von HIGH auf LOW
 - ▶ Address Frame: 7 – 10 Bits. Damit adressiert der Master den gewünschten Slave
 - ▶ Read/Write Bit: Master will senden (LOW) oder empfangen (HIGH)
 - ▶ ACK/NACK Bit: Address/Data Frames werden von einem ACK/NACK-Bit abgeschlossen. ACK vom Empfänger an Sender der Nachricht (bei Erfolg)
 - ▶ Stop Condition Bit: SDA von LOW auf HIGH, dann SCL von LOW auf HIGH



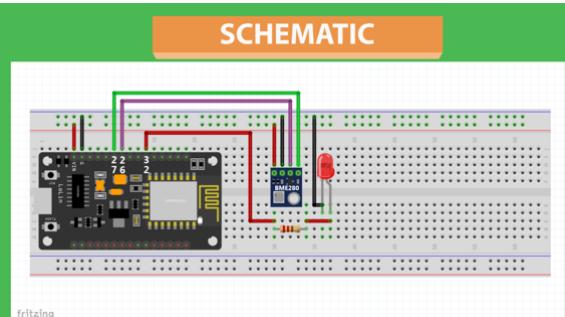
© Michael Stal, 2020
Page 24

24

I²C Kommunikation zwischen Master und Slaves (2)

SIEMENS
Ingenuity for life

Beispiel: ESP32 mit I²C-Anbindung eines BME280



Quelle: educ8s.tv

Ablauf

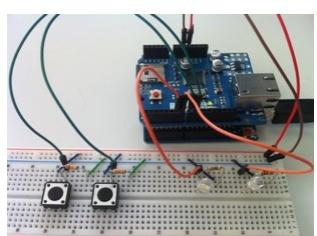
- ▶ Sobald Master bei der Suche nach dem Slave (Address Frame) erfolgreich ist, erhält er ein ACK des Slaves
- ▶ Nun kann er die eigentlichen Nachrichten an den Slave senden (Data Frames)
- ▶ ... oder Information vom Slave erhalten

© Michael Stal, 2020
Page 25

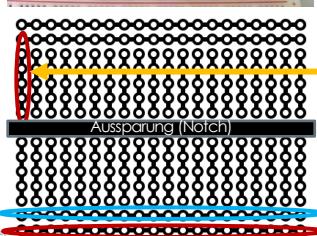
25

Breadboards

SIEMENS
Ingenuity for life



Es ist sinnvoll, das Microcontroller-Board für das Prototyping zusammen mit einem Breadboard zu benutzen. Dadurch entfällt das Löten



Ist die Schaltung fertig (getestet), lässt sich ein PCB (Printed Circuit Board) verwenden

Breadboards haben Löcher (um Bauteile aufzunehmen), die sich in Terminal Strips oder Bus Strips organisieren

Terminal Strip: Vertikaler Verbindungsstreifen

Kontaktlose Aussparung (Notch) in der Mitte (um ICs mit DIPs zu platzieren)

Bus Strips zur Stromversorgung, blau = Erde, rot = Spannung

©
Page 26

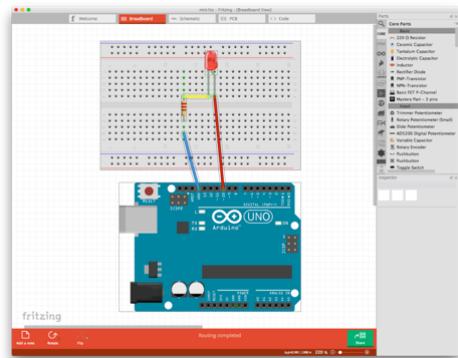
26

Fritzing zum Entwerfen von Schaltungen

SIEMENS
Ingenuity for life

Open Source Fritzing Editor zum Zeichnen von Schaltungen, und mehr

Download verfügbar über <http://fritzing.org> (Mac OS, Windows, Linux) – frei, aber kleine Spende empfohlen



© Michael Stal,2020
Page 27

27

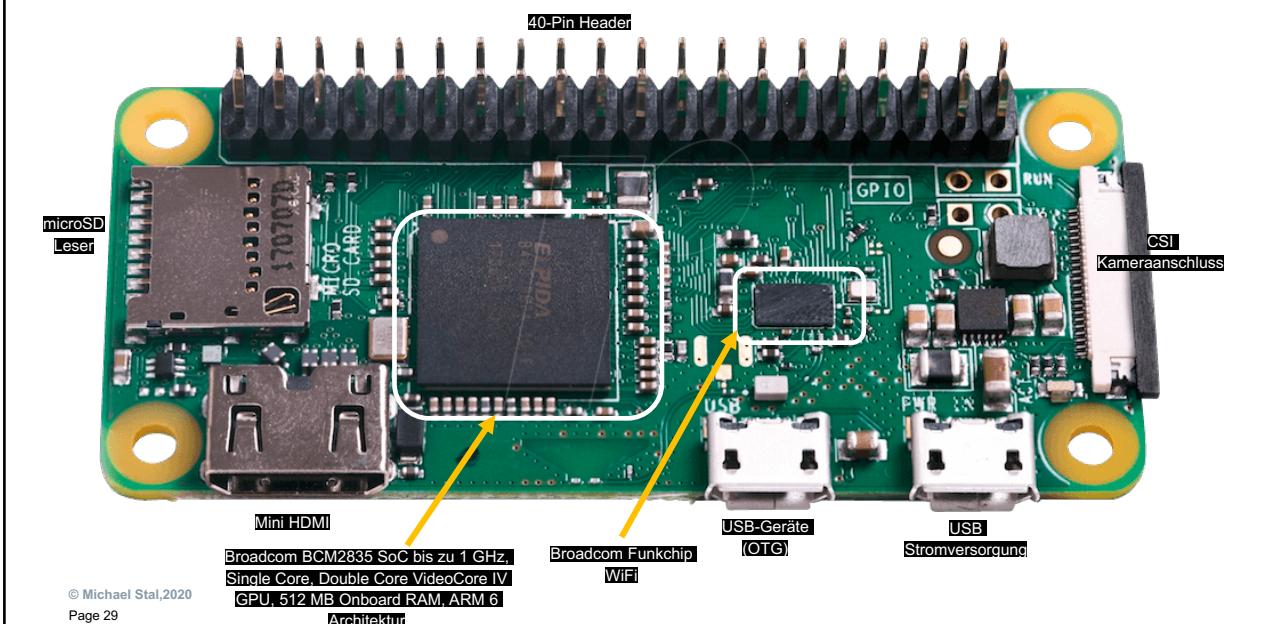
Und jetzt zur Hardware

SIEMENS
Ingenuity for life

© Michael Stal,2020
Page 28

28

Raspberry Pi Zero WH (W = WiFi, H = vorinstallierter Header)



29

Header mit GPIOs

SIEMENS
Ingenuity for life

- 802.11 b/g/n wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)
- 1GHz, Einkern-CPU
- 512MB RAM
- Mini HDMI und USB On-The-Go Ports
- Micro USB Stromversorgung
- HAT-kompatibler 40-Pin Header (HAT = Hardware on Top)
- Composite Video plus Reset Headers
- CSI Camera Anschluss



RP Model B+		RP Model B+	
Power	3V3	1	2
SDA I2C	GPIO2	3	4
SCL I2C	GPIO3	5	6
		7	8
		9	10
		11	12
		13	14
		15	16
		17	18
Power	3V3	19	20
MOSI	GPIO10	21	22
MISO	GPIO9	23	24
SCLK	GPIO11	25	26
I2C ID EEPROM	ID_SD	27	28
	GPIO5	29	30
	GPIO6	31	32
	GPIO13	33	34
	GPIO19	35	36
	GPIO26	37	38
		39	40
Ground		Ground	
ID_SC		I2C ID EEPROM	
Power		Power	
Power		Power	
Ground		Ground	
UART0_TXD		UART0_RXD	
PCM_CLK		PCM_CLK	
Ground		Ground	
GPIO23		GPIO24	
Ground		Ground	
GPIO25		GPIO26	
GPIO12		GPIO13	
Ground		Ground	
GPIO16		GPIO17	
GPIO20		GPIO18	
GPIO21		GPIO19	
Ground		Ground	

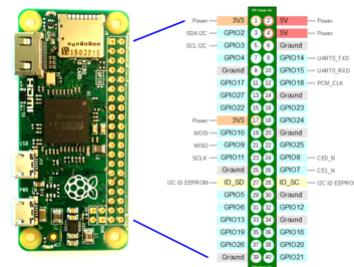
© Michael Stal, 2020
Page 30

30

Technische Spezifikation GPIO Header

SIEMENS
Ingenuity for life

- 2 Pins mit 5 Volt, um Board mit Strom zu versorgen
- 2 Pins mit Spannung von 3,3 Volt
- 2 Pins zur Identifikation des HAT über I²C (EEPROM mit DeviceTree)
- 8 Pins als Masse
- 17 Pins bzw. 26 Pins frei programmierbar (für Spannung von 3,3 Volt ausgelegt)
- 5 Pins als SPI-Schnittstelle
- 2 Pins mit 1,8-kΩ-Pull-up-Widerstand (auf 3,3 V) - für I²C nutzbar
- 2+ Pins als UART-Schnittstelle
- GPIO-Schnittstelle P6 erlaubt Rücksetzen/Start des Raspberry Pi
- Zur Steuerung der GPIOs: Bibliotheken für diverse Programmiersprachen. Auch Steuerung durch Terminal oder Webinterfaces möglich
- Empfehlung: maximal 8 mA pro GPIO-Port



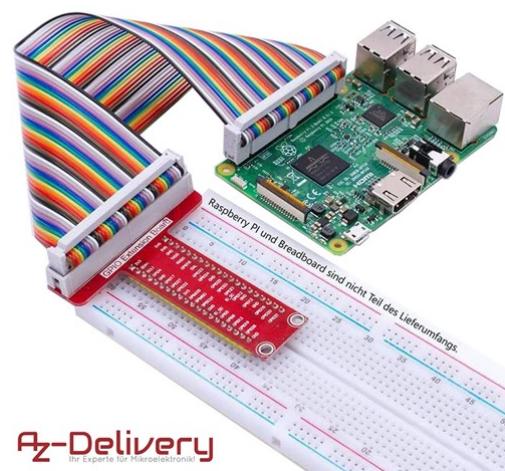
© Michael Stal, 2020
Page 31

31

Mögliche Option für Maker: T-Coppler (GPIO Breakout-Board)

SIEMENS
Ingenuity for life

- Direkte Verbindung des Raspberry Pi Headers über Flachbandkabel mit dem Breadboard
- Dadurch stehen die GPIO-Pins direkt auf dem Breadboard zur Verfügung
- Zusätzliche externe Stromquellen lassen sich mit den roten/blauen Strängen des Breadboards verbinden



AZ-Delivery
Ihr Experte für Mikroelektronik

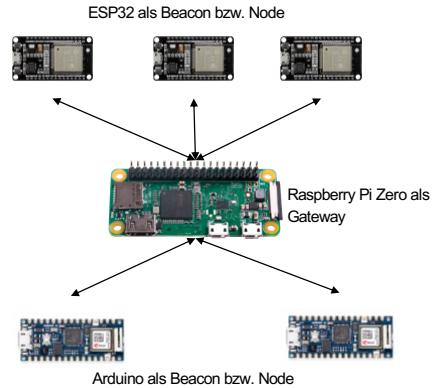
© Michael Stal, 2020
Page 32

32

Mögliche Konfiguration für IoT-Anwendungen

SIEMENS
Ingenuity for life

- Raspberry Pi leistungsstark aber nicht wirklich fähig zu Echtzeit/Embedded
 - verfügt nur über wenige I/O-Schnittstellen
 - durch Aufsetzboards (HATs) wie Gert Board oder Gertduino kompensierbar (dort sind „Arduino“-Microcontroller verbaut)
- ESP32, Arduino weit weniger performant, aber dafür eingebettet, echtzeitfähig, mit vielen I/O-Schnittstellen
- Konstellation mit Gateway als Edge-Knoten, Beacons als Sensornetzwerk
- Nutzung von Tensorflow Lite auf Raspberry Pi zur Analytik von Messdaten
- Kommunikation zwischen Beacons und Gateway im lokalen Mesh z.B. über serielle Ports, Bluetooth, LoRA, WiFi, IR, Funk



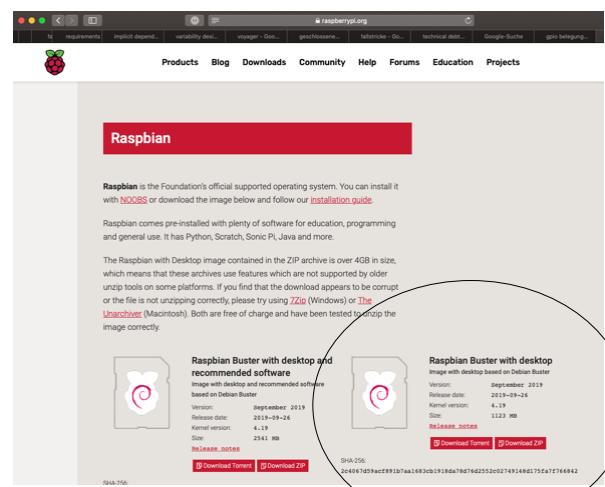
© Michael Stal,2020
Page 33

33

Installation von Embedded Linux/Raspbian

SIEMENS
Ingenuity for life

- Download von *Raspbian Buster with desktop* von Seite <https://www.raspberrypi.org/downloads/>



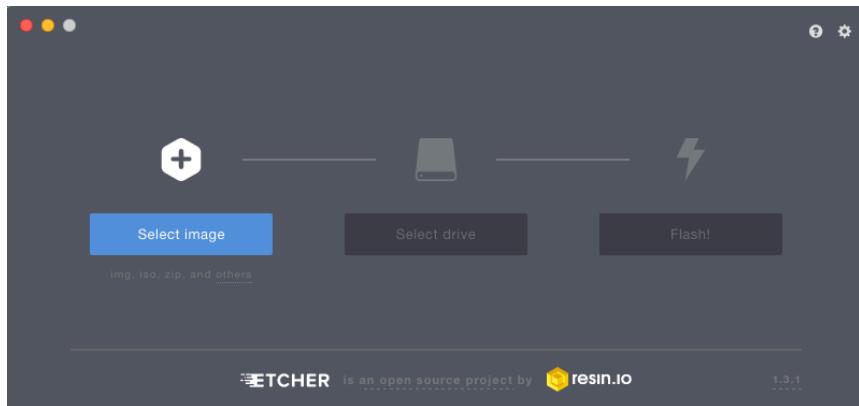
© Michael Stal,2020
Page 34

34

Image auf microSD-Karte kopieren

SIEMENS
Ingenuity for life

Übertragung des Raspbian-Images auf Raspbian über Werkzeug *balenaEtcher*
Downloadseite: <https://www.balena.io/etcher>



© Michael Stal,2020
Page 35

35

Kopflose (headless) Installation

SIEMENS
Ingenuity for life

- Zwei Optionen:
 - Installation als SBC (Single Board Computer) mit Maus, Tastatur, Bildschirm (*im Tutorial nicht machbar*)
 - **Installation zum Zugriff über SSH**
 - Zwei Dateien auf Boot-Verzeichnis der microSD-Karte: ssh (leere Datei) und wpa_supplicant.conf (siehe rechts)
 - Linux verschiebt diese Dateien bei Systemstart an die richtigen Verzeichnisse
 - Achtung: Bitte auf Linux/Unix-Zeichenenden achten. Keine Windowszeilenenden!

```
country=DE
ctrl_interface=DIR=/var/run/wpa_supplicant
GROUP=netdev update_config=1
network={
    ssid="meine WLAN SSID"
    scan_ssids=1
    psk="mein WLAN Passwort"
    key_mgmt=WPA-PSK
}
```

© Michael Stal,2020
Page 36

36

Weitere Option PiBakery

SIEMENS
Ingenuity for life

- Lohnt sich bei häufigen Änderungen oder bei Beschreiben unterschiedlicher SD-Karten mit eigenen Konfigurationen
- Speziell von Vorteil bei Installation vieler Images
- Rezepte (recipes) lassen sich hinzufügen, ändern
- Konfigurationen importierbar und exportierbar
- PiBakery schreibt Image mit Konfiguration auf (micro-)SD-Karte
- Download über: <https://www.pibakery.org/>



© Michael Stal,2020
Page 37

37

Nach Booten von Raspbian

SIEMENS
Ingenuity for life

- Zugriff auf Raspberry Pi Zero WH über ssh von Windows, Linux, Mac, Android, iOS, ...
- *Rechtes Bild:* Zugriff auf Rechner über App termius vom iPad
- Rechnername: raspberrypi
- User: pi
- Initiales Passwort: raspberry
 - Passwort bitte ändern
- Eigene IP-Adresse ermitteln über hostname -I
- **Geregeltes Herunterfahren*** mit sudo shutdown -H 0

```
Linux raspberrypi 4.19.75+ #1278 Tue Sep 24 18:38:54 BST 2019 armv6l
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
any such warranty is allowed by applicable law.

last login: Tue Dec 31 17:08:52 2019 from 192.168.176.85
pi@raspberrypi: ~ ls -al
total 12
drwxr-xr-x 17 pi pi 4096 Dec 31 17:08 .
drwxr-xr-x 3 root root 4096 Sep 26 01:09 ..
-rw-r--r-- 1 pi pi 932 Sep 26 01:09 .bash_history
-rw-r--r-- 1 pi pi 323 Sep 26 01:09 .bash_logout
-rw-r--r-- 1 pi pi 3523 Sep 26 01:09 .bashrc
-rw-r--r-- 8 pi pi 4096 Sep 23 01:32 .cache
-rw-r--r-- 1 pi pi 4096 Sep 23 01:32 .config
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 Desktop
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 Documents
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 Downloads
drwxr-xr-x 3 pi pi 4096 Sep 26 01:31 .empty
drwxr-xr-x 1 pi pi 4096 Dec 23 18:01 .local
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 .local/share
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 Music
drwxr-xr-x 5 pi pi 4096 Dec 31 16:47 ootp2020
drwxr-xr-x 1 pi pi 4096 Sep 26 01:32 Pictures
drwxr-xr-x 1 pi pi 4096 Sep 26 01:32 Public
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 Public/ssh
drwxr-xr-x 1 pi pi 4096 Dec 31 15:42 .recently-used
drwxr-xr-x 1 pi pi 4096 Dec 31 15:42 .recently-used.xbel
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 .ssh
drwxr-xr-x 1 pi pi 4096 Sep 26 01:32 .ssh/known_hosts
drwxr-xr-x 1 pi pi 4096 Sep 26 01:32 .ssh/known_hosts.old
drwxr-xr-x 6 pi pi 4096 Dec 25 20:06 .ttk-ttk
drwxr-xr-x 2 pi pi 4096 Sep 26 01:32 Videos
drwxr-xr-x 1 pi pi 4096 Sep 26 01:32 .wheezy-backups
drwxr-xr-x 1 pi pi 56 Dec 31 17:08 .xauthORITY
drwxr-xr-x 1 pi pi 2014 Dec 31 17:08 .xsession-errors
drwxr-xr-x 1 pi pi 2014 Dec 30 00:57 .xsession-errors.old
pi@raspberrypi: ~ hostname 2
192.168.176.88
pi@raspberrypi: ~
```

Einfaches Ziehen des Netzsteckers kann zu Fehlern im Dateisystem führen

© Michael Stal,2020
Page 38

38

Immer aktuell

Jetzt sollte Raspbian auf den aktuellen Stand gebracht werden:

```
pi@raspberrypi:~ $ sudo apt update
pi@raspberrypi:~ $ sudo apt full-upgrade
```

Warnung: Gerade letzteres kann ein paar Minuten dauern



SIEMENS
Ingenuity for life

```
michael - pi@raspberrypi: ~ — ssh pi@raspberrypi — 80x24
pi@raspberrypi's password:
Linux raspberrypi 4.19.75+ #1270 Tue Sep 24 18:38:54 BST 2019 armv6l

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: Wed Jan  8 08:54:20 2020
pi@raspberrypi:~ $ raspistill -o test.jpg
pi@raspberrypi:~ $ raspistill -o test.jpg
pi@raspberrypi:~ $ nodered
pi@raspberrypi:~ $ node-red
pi@raspberrypi:~ $ node-red
pi@raspberrypi:~ $ sudo apt update
Get:1 http://archive.raspberrypi.org/debian buster InRelease [25.2 kB]
Get:2 http://raspbian.raspberrypi.org/raspbian buster InRelease [15.0 kB]
Get:3 http://raspbian.raspberrypi.org/raspbian buster/main armhf Packages [13.0
MB]
Get:4 http://archive.raspberrypi.org/debian buster/main armhf Packages [260 kB]
68% [3 Packages 7797 kB/13.0 MB 60%] 572 kB/s 9s
pi@raspberrypi:~ $
```

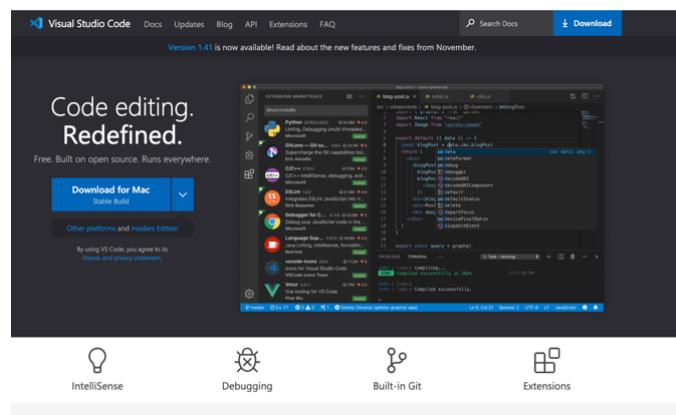
© Michael Stal, 2020
Page 39

39

Editieren von Programmdateien

SIEMENS
Ingenuity for life

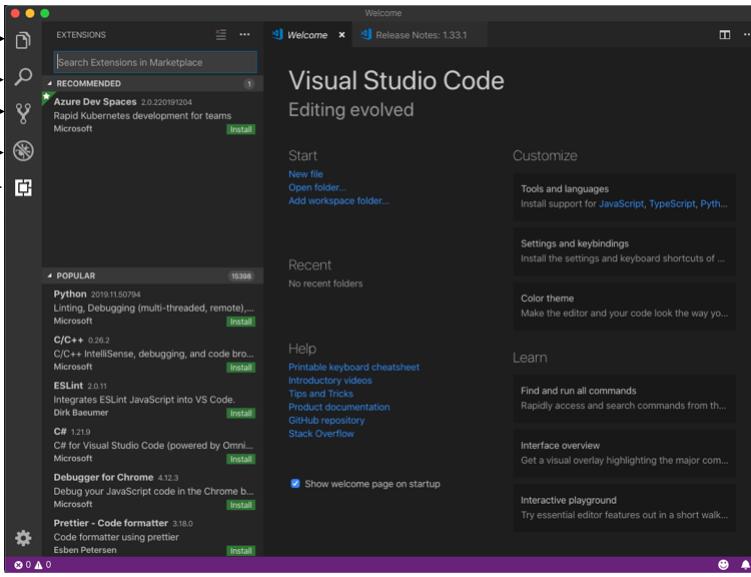
- Eine Möglichkeit: mit vim, vi, emacs, nano direkt auf dem Raspi editieren
- Eine andere Möglichkeit: VSCode (auf Windows, Mac, Linux) plus rmate (auf Raspbian)
 - Download über code.visualstudio.com



© Michael Stal, 2020
Page 40

40

Visual Studio Code lässt sich mittels Extensions erweitern



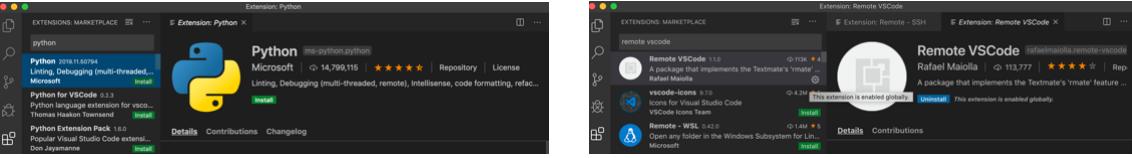
The screenshot shows the Visual Studio Code interface. On the left, there is a sidebar with icons for Explorer, Suchen (Search), Versionskontrolle (Version Control), Debugging, and Extensions. The main area displays the 'Welcome' screen with sections for Start, Customize, Recent, Help, Learn, and a list of popular extensions like Python, C/C++, ESLint, and C#.

SIEMENS
Ingenuity for life

© Michael Stal, 2020
Page 41

41

Installation Python und Remote VSCode



The screenshot shows the Visual Studio Code Extension Marketplace. It displays two extensions: 'Python' by Microsoft and 'Remote VSCode' by Rafael Malolla. Both extensions have high ratings and are installed.

SIEMENS
Ingenuity for life

Python

- Stellt Python IDE zur Verfügung

Remote VSCode

- Erlaubt Fern-Editieren auf Raspberry Pi

© Michael Stal, 2020
Page 42

42

Konfigurieren von Remote VSCode und rmate

SIEMENS
Ingenuity for life

Schritt 1: Host mit VSCode (Mac, Linux, Windows)

- Bei Erweiterung Einstellungen anklicken (Zahnrad von Remote VSCode in der Übersicht installierter Extensions)
- Configure Extension Settings auswählen
 - Remote Don't Show Port Already in Use Error-Kästchen selektieren
 - Remote Host: 127.0.0.1
 - Remote: Onstartup: Kästchen auswählen
 - In Remote: Port 52698 eingeben
- In Visual Studio Code auf F1-Taste drücken und Remote: Start Server wählen
- Auf Konsole/Terminal einen SSH-Tunnel errichten:
\$> ssh -R 52698:127.0.0.1:52698 pi@<ip-adresse>

Schritt 2: Raspberry Pi Zero WH

- rmate (Python-Version) installieren:
pip install rmate
- Zu editierende Datei selektieren mit
rmate -P 52698 meine datei.py
- Nun erscheint die Datei im Editor-Fenster von Visual Studio Code

Anmerkung: Leider unterstützt die wesentlich ausgereiftere Extension Remote Development (momentan nur lauffähig unter VS Code Insider Versionen!) (noch) nicht die ARM61-Architektur des Raspberry Pi Zero WH

© Michael Stal,2020
Page 43

43

Python – print("Wir nutzen Python 3")

SIEMENS
Ingenuity for life

- Das Tutorium nutzt Python 3 als Sprache
- Deutsches Tutorial z.B. auf <https://py-tutorial-de.readthedocs.io/de/python-3.3/>
- Auf Raspberry Pi heißt der Interpreter python3
- ... und das zugehörige Paketinstallationsprogramm heißt pip3



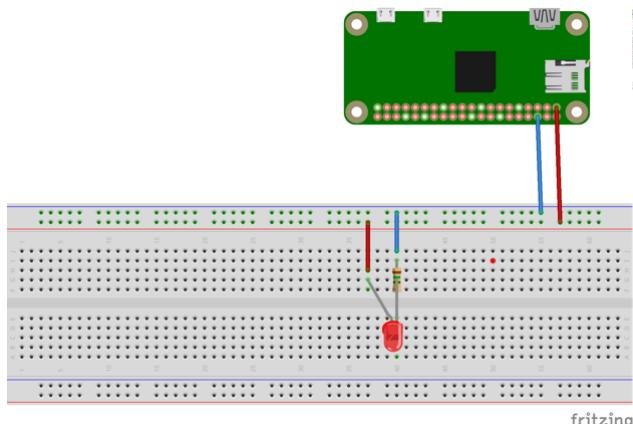
© Michael Stal,2020
Page 44

44

Kleiner Test des Raspberry Pi GPIO

SIEMENS
Ingenuity for life

- Anschluss einer LED über Pin 1 (3.3V) und Pin 6 (Erde/Ground)
- Widerstand, z.B. 150 Ohm, 220 Ohm
- Merke:
 - Pins mit maximal 8 mA beladen
 - LED braucht ≥ 2 V
 - $\Rightarrow 1.3$ V Spannungsabfall am Widerstand
 - Wegen $U = R \cdot I \Rightarrow$
 - $R = U / I = 1.3 \text{ V} / 0.008 \text{ A} = 162,5 \text{ Ohm}$
- Für einen kurzen Test lässt sich eine LED auch ohne Widerstand anschließen Das ist aber nicht für längere Zeit ratsam



© Michael Stal, 2020
Page 45

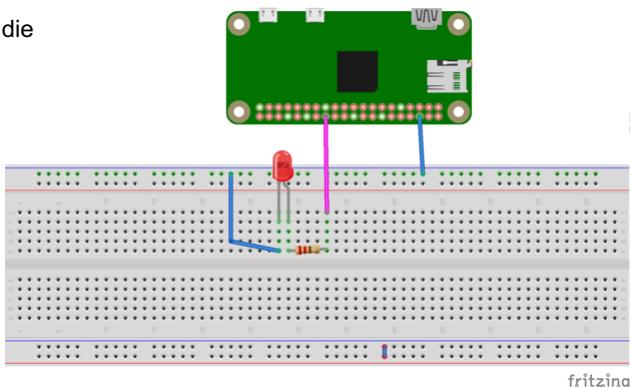
45

Blinkende LED – das übliche Beispiel 😊

SIEMENS
Ingenuity for life

- Schaltung über Pin 26
- Zunächst ohne Programmierung über die Kommandozeile mit
 - Setzen des Ausgabemodus für Pin 26
 - Schreiben von 1 auf Pin 26 (LED an)
 - Schreiben von 0 auf Pin 26 (LED aus)
 - Lesen des Wertes an Pin 26

```
gpio -1 mode 26 out
gpio -1 write 26 1
gpio -1 write 26 0
gpio -1 read 26
```



© Michael Stal, 2020
Page 46

46

Programmatischer Ansatz

- Die LED blinks 10 mal jeweils im Sekundentakt
- Das Programm verwendet dafür die Bibliothek `RPi.GPIO`
- Der Modus `GPIO.BCM` sorgt dafür, dass Entwickler die physikalischen Pins angeben können
- Alternative wäre `GPIO.BCM` (Broadcom GPIO Numbers). In diesem Fall entspricht die Angabe von 7 dem Port `GPIO7`, was wiederum dem physikalischen Pin 26 entspricht

SIEMENS
Ingenuity for life

```
#!/usr/bin/python3
import RPi.GPIO as GPIO
import time
# Die Nummerierung der Pins erfolgt
# anhand der Position am Board
GPIO.setmode(GPIO.BCM)

# LED an Pin 26
LEDPIN = 26

# LEDPIN ist Ausgabe-Port
GPIO.setup(LEDPIN, GPIO.OUT)

for i in range(10):
    print("Schleifendurchgang ", i+1)
    GPIO.output(LEDPIN, GPIO.HIGH)
    print("An")
    time.sleep(0.5)
    GPIO.output(LEDPIN, GPIO.LOW)
    time.sleep(0.5)
    print("Aus")
GPIO.cleanup()
```

© Michael Stal,2020
Page 47

47

Software-mäßiges Dimmen über PWM

SIEMENS
Ingenuity for life

- Über PWM lässt sich die LED auch dimmen
- Als Frequenz nutzen wir 100 Hz (`pwm = GPIO.PWM(PIN, 100)`)
- Die Helligkeit wird über `pwm.ChangeDutyCycle(percentage)` gesteuert
- Das Argument legt fest, zu wieviel Prozent eines Zyklus die volle Spannung anlegt => Mittelwert per Zyklus definiert Helligkeit
- Je höher dieser Wert desto heller leuchtet die LED
- Das Programm rechts erhöht in der `while`-Schleife zunächst die Helligkeit in 4er-Schritten, um sie dann wieder schrittweise zu verringern

```
#!/usr/bin/python3
from time import sleep
import RPi.GPIO as GPIO

PIN = 26
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(PIN, GPIO.OUT)
pwm=GPIO.PWM(PIN, 100);
pwm.start(0)
maxreps = 10
repetitions = 0
while repetitions < maxreps:
    for i in range(0,100,4):
        pwm.ChangeDutyCycle(i)
        sleep(0.1)
    for i in range(0,101,4):
        pwm.ChangeDutyCycle(100-i)
        sleep(0.1)
    repetitions += 1

GPIO.cleanup()
```

© Michael Stal,2020
Page 48

48

Hardware-mäßiges Dimmen über PWM

SIEMENS
Ingenuity for life

- Ist weniger ratsam, lässt sich aber testen
- Nicht alle GPIO-Pins sind PWM-fähig!
- Installation der Bibliothek `wiringpi` notwendig
`pip3 install wiringpi2`

```
#!/usr/bin/python3
import wiringpi
import time

LED_PIN = 12
wiringpi.wiringPiSetupPhys()

wiringpi.pinMode(LED_PIN,2)

maxreps = 10
repetitions = 0

while repetitions < maxreps:
    for i in range(0,1001,50):
        wiringpi.pwmWrite(LED_PIN, i)
        time.sleep(0.1)
    for i in range(0,1001,50):
        wiringpi.pwmWrite(LED_PIN, 1000-i)
        time.sleep(0.1)

    repetitions += 1

wiringpi.pwmWrite(LED_PIN,0)
```

© Michael Stal,2020
Page 49

49

Hinweis zu GPIO-Bibliotheken

- Eine weitere alternative Bibliothek ist übrigens `gpiozero` (siehe URL <https://gpiozero.readthedocs.io/>)
- Deren Installation erfolgt über

```
sudo pip3 install gpiozero
```

© Michael Stal,2020
Page 50

50

Anschluss eines PIR-Sensors

SIEMENS
Ingenuity for life

- Steht für Pyroelectric Infrared Sensor
- Typ: HC-SR501
- Erkennt Bewegungen
- Über zwei Potis sind regulierbar:
 - Verzögerungszeit und
 - Dauer der Aktivierung
- Benötigt 5V Versorgungsspannung
- Funktioniert aber mit 3.3V-Signalen an seinen Dateneingängen und -ausgängen
- Optionale Dip-Switches erlauben 2 Arbeitsmodi:
 - Bewegungsmeldung und dann Signal wieder auf 0
 - Solange Bewegungen vorhanden bleibt Sensor alarmiert (Signal = 1)



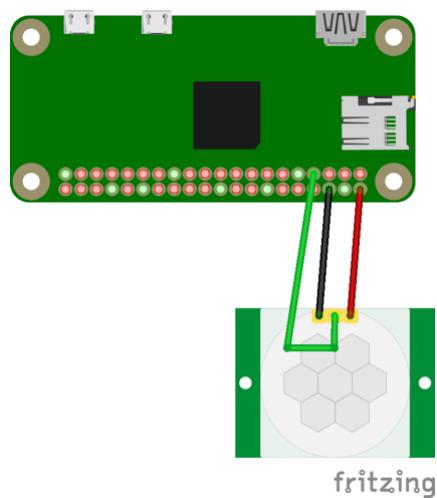
© Michael Stal, 2020
Page 51

51

Beispiele-Schaltung

SIEMENS
Ingenuity for life

- Rot: 5V des Raspi an Pin 2
- Schwarz: GND des Raspi an Pin 6
- Grün: Dateneingang des Sensors an Pin 7



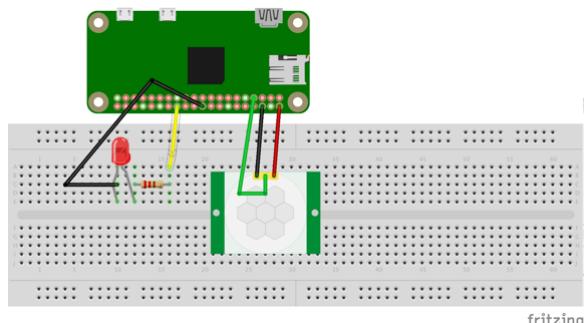
© Michael Stal, 2020
Page 52

52

Ein kleines Experiment: Bewegungserkennung & LED

SIEMENS
Ingenuity for life

- Rot: 5V des Raspi an Pin 2
- Schwarz: GND des Raspi an Pin 6 bzw. Ground an Pin 20
- Grün: Dateneingang des Sensors an Pin 7
- Gelb: Steuerung der LED über Pin 26
- Widerstand 220 V
- Ziel: Sobald Bewegung erkannt, soll LED für eine Zeit leuchten



© Michael Stal, 2020
Page 53

53

Das Programm: Deklarationen

SIEMENS
Ingenuity for life

- Wir nutzen an Pin 7 einen Pull-Down-Widerstand, um versehentliches internes Auslösen zu verhindern
- Eine LED ist an Pin 26 angebracht
- Der PIR Sensor liegt an Pin 7

```
#!/usr/bin/python3

import sys
from time import sleep
import RPi.GPIO as GPIO

# Zeit, während der die LED leuchtet
SLEEP_TIME = 10
# Verwendete LED an Pin 26
LEDPIN = 26

# Verwendet werden die physikalischen Pins
GPIO.setmode(GPIO.BOARD)
# LEDPIN ist Ausgabeport
GPIO.setup(LEDPIN, GPIO.OUT)

# Pin 7 greift den Status des PIR-Sensors ab
GPIO.setup(7, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)

# Fortsetzung auf nächster Folie
```

© Michael Stal, 2020
Page 54

54

Das Programm: Ausführungslogik

SIEMENS
Ingenuity for life

- Der Callback-Handler `bewegung_erkann()` wird bei GPIO angemeldet
 - reagiert auf steigende Signalflanken
- Behandlung der Ereignisse im Callback-Handler
 - Handler schaltet für eine konstante Zeit das Licht ein, dann wieder aus
- Zusätzlich wird Meldung am Bildschirm ausgegeben

```
# Callback bei Statusänderungen am PIR
def bewegung_erkann(pin):
    print("Bewegung registriert")
    # LED an bei Bewegungserkennung
    GPIO.output(LEDPIN, GPIO.HIGH)
    sleep(SLEEP_TIME)
    GPIO.output(LEDPIN, GPIO.LOW)
    return

# Auslösung der Ereignisse durch
# aufsteigende Signalflanken
GPIO.add_event_detect(7, GPIO.RISING)
GPIO.add_event_callback(7, bewegung_erkann)

print("CONTROL-C beendet das Programm")
try:
    while True: # Endlosschleife
        sleep(0.5)
except KeyboardInterrupt: # Schleifenende bei Interrupt
    GPIO.cleanup()
    sys.exit()

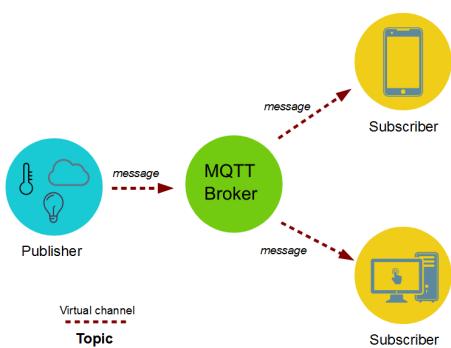
# Ende
```

© Michael Stal,2020
Page 55

55

Raspberry Pi und MQTT

SIEMENS
Ingenuity for life



© Michael Stal,2020
Page 56

56

Mosquitto als MQTT Broker

- Spricht MQTT
- Stellt Topics zur Verfügung
- Ist Ansprechpartner für Sender (Publisher) und Receiver (Subscriber)
- Wir nutzen *Eclipse Mosquitto*
- Installation
 - sudo apt update
 - sudo apt install -y mosquitto-clients
- Auto-Start beim Booten:
 - sudo apt update
 - sudo systemctl enable mosquitto.service
- Abfrage der installierten Version:
mosquitto -v



© Michael Stal, 2020
Page 57

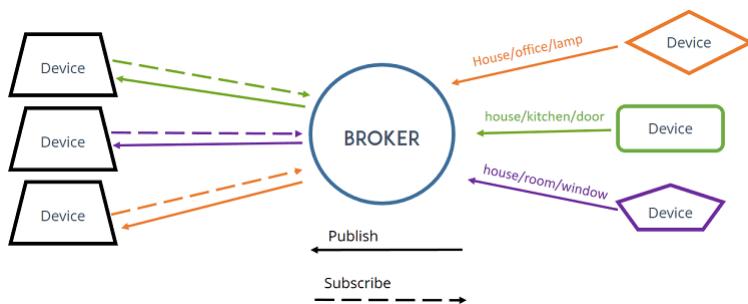
57

Test der Installation



Auf dem Raspi-Terminal geben wir ein:

```
mosquitto_sub -d -t meinTestTopic&  
  
mosquitto_pub -d t meinTestTopic -m "Hallo, OOP 2020!"
```



© Michael Stal, 2020
Page 58

58

MQTT Beispiel mit Python - Subscriber

SIEMENS
Ingenuity for life

- Installation von Bibliothek notwendig
`sudo pip3 install paho-mqtt`
- Alternativ
`git clone https://github.com/eclipse/paho.mqtt.python.git`
`cd paho.mqtt.python`
`python3 setup.py install`
- Start des Programmes mit
`python3 mqttsub_bsp.py&`

```
import paho.mqtt.client as mqtt

# Callback für Verbindungsaufbau
def on_connect(client, userdata, flags, rc):
    print("Verbindung aufgebaut mit Fehlercode " + str(rc))
    client.subscribe("oop2020/test")
# Callback für Nachrichtenempfang
def on_message(client, userdata, msg):
    print("Neue Nachricht im Topic " + msg.topic + " Inhalt: " + str(msg.payload))
# MQTT Client anlegen
client = mqtt.Client()
# Callbackhandler zuweisen
client.on_connect = on_connect
client.on_message = on_message
# Verbindung zum Broker aufbauen
client.connect("localhost", 1883, 60)
# Auf eingehende Nachrichten warten
client.loop_forever()
```

© Michael Stal,2020
Page 59

59

MQTT Beispiel mit Python - Publisher

SIEMENS
Ingenuity for life

- Start des Programmes mit mit
`python3 mqttpub_bsp.py`

```
import paho.mqtt.client as mqtt

# Einen Client kreieren
client = mqtt.Client("node")
# Mit dem lokalen Broker an Port 1883 verbinden:
client.connect("localhost")
# topic: Nachrichtenwarteschlange, dazu Inhalt = payload
# QoS (0:at most once 1: at least once 2: exactly once)
# Parameter 4: retained? (Nachricht wird jedem zugesandt)
client.publish("oop2020/test", "Diese Nachricht bekommen alle zu sehen", 1, True)
client.publish("oop2020/test", "Hallo liebe Tutorialbesucher", 1, False)
client.publish("oop2020/test", "Ich hoffe, alle geniessen die Arbeit", 1, False)
```

© Michael Stal,2020
Page 60

60

SIEMENS
Ingenuity for life

ThingSpeak

– Cloudservice für's IoT

ThingSpeak Features

- Collect data in private channels
- Share data with public channels
- RESTful and MQTT APIs
- MATLAB® analytics and visualizations
- Alerts
- Event scheduling
- App integrations
- Worldwide community

Works With

- Arduino®
- Particle Photon and Electron
- ESP8266 WiFi Module
- Raspberry Pi™
- Mobile and web apps
- Twitter®
- Twilio®
- MATLAB®

61

ThingSpeak Features

SIEMENS
Ingenuity for life

Daten in privaten Kanälen speichern
Mit öffentlichen Kanälen Daten publizieren
RESTful und MQTT APIs
MATLAB® Analytiken und Visualisierungen
Alarne
Ereignisverteilung
Integration von Apps
Weltweite Community

© Michael Stal, 2020
Page 62

62

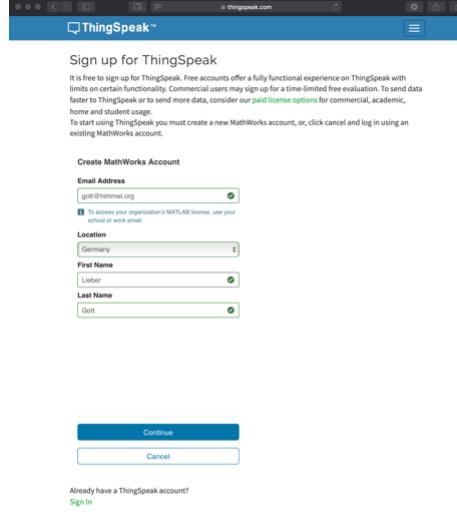
Registrierung



- ▶ Auf initialer Webseite "Get Started For Free" wählen

Dann MathWorks Konto anlegen (siehe Bild rechts)

- ▶ Zuletzt anmelden

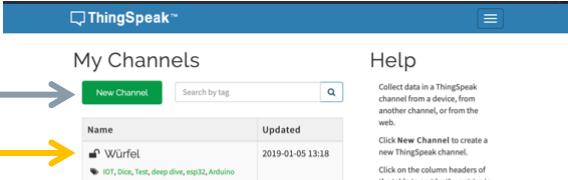


The screenshot shows the 'Sign up for ThingSpeak' page. It includes fields for 'Email Address' (gott@siemens.org), 'Location' (Germany), and 'First Name' (Lüder). Below these are 'Continue' and 'Cancel' buttons. At the bottom, there's a link 'Already have a ThingSpeak account? Sign In'.

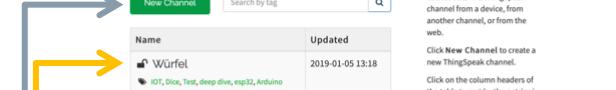
© Michael Stal, 2020
Page 63

63

Kanäle verwalten



Nach der Anmeldung sehen Sie die Webseite mit Ihren Kanälen
Mit New Channel lässt sich ein neuer Kanal anlegen
Existierende Kanäle werden ebenfalls angezeigt



The screenshot shows the 'My Channels' section of the ThingSpeak interface. It features a table with columns 'Name' and 'Updated'. Two rows are listed: 'Würfel' (updated 2019-01-05 13:18) and 'IOT_Dice_Test_deep_dive_esp32_Arduino'. To the right of the table is a 'Help' section with instructions on creating new channels and using tags, and an 'Examples' section listing various Arduino-based projects.

© Michael Stal, 2020
Page 64

64

Anlegen eines neuen Kanals

SIEMENS
Ingenuity for life

Geben Sie die Daten für Ihren Kanal ein und klicken Sie anschließend ganz unten auf Save Channel

Save Channel

© Michael Stal, 2020
Page 65

Visualisierung gestalten

- Auf der nun angezeigten Seite können Sie die Visualisierung Ihres neuen Kanals konfigurieren
- Wichtig für den späteren programmatischen Zugriff ist die Channel ID (hier: 669466)
- Gehen Sie anschließend auf API Keys

© Michael Stal, 2020
Page 66

Zugriffsschlüssel

In den Feldern finden Sie die API Keys zum Lesen und Schreiben (hier unkenntlich gemacht)

Die benötigen Sie später zum programmatischen Zugriff auf den Kanal

© Michael Stal, 2020
Page 67

67

Going Public

- ▶ Standardmäßig ist der Kanal nur privat für Sie sichtbar
- ▶ Mittels des Registertabs Sharing lässt sich das einstellen
- ▶ Wir machen den Kanal öffentlich

© Michael Stal, 2020
Page 68

68

Trockendock - Würfelbeispiel

Eine App greift auf den zuvor
kreierten Kanal zu
Sie ermittelt die Ergebnisse
zweier Würfel und schickt
diese als Felder an
ThingSpeak
Achtung: In der freien Version
darf ein Kanal maximal einmal
pro 20 Sekunden aktualisiert
werden



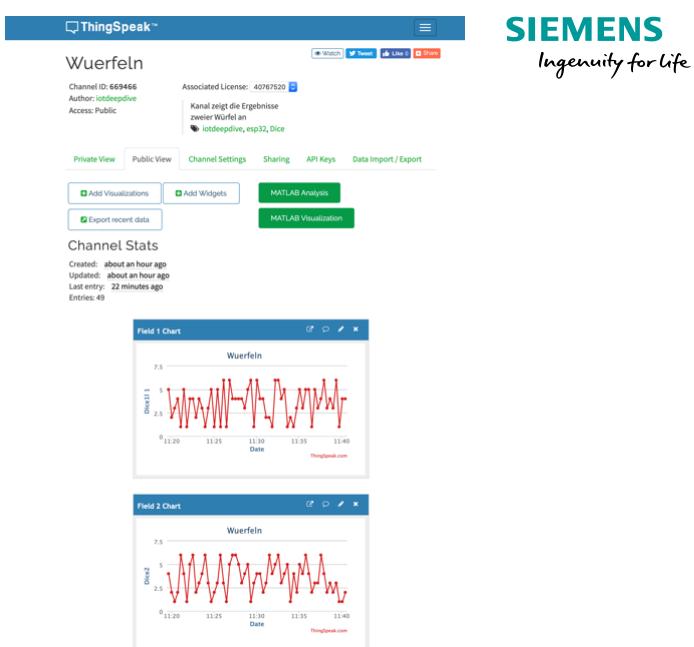
© Michael Stal, 2020
Page 69

69

Trockendock - Visualisierung auf ThingSpeak

Wir sehen auf der Webseite den
Vorlauf der Würfelergebnisse bei
Würfel 1 (oben) und Würfel 2 (unten)

Keine Angst – wir sehen bald ein
richtiges Beispiel mit Code



© Michael Stal, 2020
Page 70

70

Programmbeispiel Hardwaredatenvisualisierung auf ThingSpeak

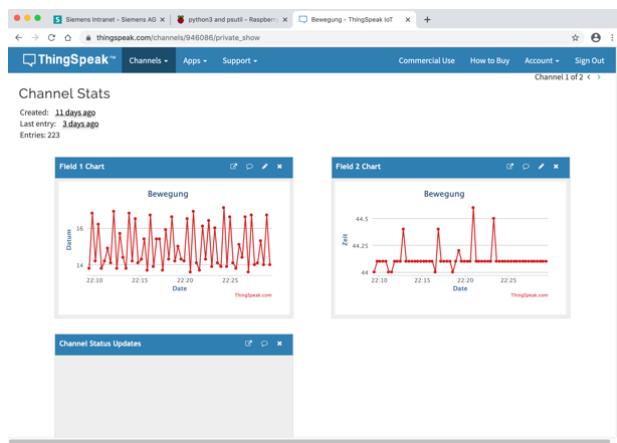
SIEMENS
Ingenuity for life

- Dazu soll auf dem RasPi Zero die Erfassung von Werten wie RAM-Verbrauch erfolgen
- Installation der Python-Bibliothek psutil nötig: sudo pip3 install psutil

```
import psutil
...
cpuPercent =
psutil.cpu_percent(interval=20)

ramPercent =
psutil.virtual_memory().percent
cpuFrequency = psutil.cpu_freq()
```

- Datenpakete werden mittels MQTT an den eigenen ThingSpeak-Channel übertragen



© Michael Stal, 2020
Page 71

71

Programmatischer Zugriff (1)

SIEMENS
Ingenuity for life

- Installation ThingSpeak Bibliothek
 - dient zum Zugriff auf ThingSpeak
 - sudo pip3 install thingspeak
- Installation der psutil-Bibliothek
 - dient zum Anzeigen von Hardwaredaten des Raspi
 - sudo apt-get update
 - sudo pip3 install psutil

```
# ThingSpeak Update Using MQTT
# Copyright 2016, MathWorks, Inc

# This is an example of publishing to multiple fields
# simultaneously.
# Connections over standard TCP, websocket or SSL are
possible by setting
# the parameters below.
#
# CPU and RAM usage is collected every 20 seconds and
published to a
# ThingSpeak channel using an MQTT Publish
#
# This example requires the Paho MQTT client package which
# is available at: http://eclipse.org/paho/clients/python

from __future__ import print_function
import paho.mqtt.publish as publish
import psutil
# Fortsetzung auf nächster Folie
```

© Michael Stal, 2020
Page 72

72

Programmatischer Zugriff (2)

SIEMENS
Ingenuity for life

- Hier eigenen Channel ID und eigenen API Key eintragen

```
### Start of user configuration ###

# ThingSpeak Channel Settings

# The ThingSpeak Channel ID
# Replace this with your Channel ID
channelID = myChannelID

# The Write API Key for the channel
# Replace this with your Write API key
apiKey = myAPIKey

# MQTT Connection Methods
# Set useUnsecuredTCP to True to use the default MQTT port
# of 1883
# This type of unsecured MQTT connection uses the least
# amount of system resources.
useUnsecuredTCP = True

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 73

73

Programmatischer Zugriff (3)

SIEMENS
Ingenuity for life

- Wie wird der Zugriff auf ThingSpeak konfiguriert

```
# Set useUnsecuredWebSockets to True to use MQTT over an
# unsecured websocket on port 80.
# Try this if port 1883 is blocked on your network.
useUnsecuredWebsockets = False

# Set useSSLWebsockets to True to use MQTT over a secure
# websocket on port 443.
# This type of connection will use slightly more system
# resources, but the connection
# will be secured by SSL.
useSSLWebsockets = False

### End of user configuration ###

# The Hostname of the ThingSpeak MQTT service
mqttHost = "mqtt.thingspeak.com"

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 74

74

Programmatischer Zugriff (4)

SIEMENS
Ingenuity for life

```
# Set up the connection parameters based on the connection
# type
if useUnsecuredTCP:
    tTransport = "tcp"
    tPort = 1883
    tTLS = None

if useUnsecuredWebsockets:
    tTransport = "websockets"
    tPort = 80
    tTLS = None

if useSSLWebsockets:
    import ssl
    tTransport = "websockets"
    tTLS = {'ca_certs':"/etc/ssl/certs/ca-certificates.crt",'tls_version':ssl.PROTOCOL_TLSv1}
    tPort = 443

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 75

75

Programmatischer Zugriff (5)

SIEMENS
Ingenuity for life

- Eigentliche Erfassung der Daten

```
# Create the topic string
topic = "channels/" + channelID + "/publish/" + apiKey

# Run a loop which calculates the system performance every
# 20 seconds and published that to a ThingSpeak channel
# using MQTT.
while(True):

    # get the system performance data
    cpuPercent = psutil.cpu_percent(interval=20)
    ramPercent = psutil.virtual_memory().percent
    cpuFrequency = psutil.cpu_freq()
    print (" CPU =",cpuPercent," RAM =",ramPercent, "
FREQ =", cpuFrequency)

    # build the payload string
    tPayload = "field1=" + str(cpuPercent) + "&field2=" +
str(ramPercent)

    # Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 76

76

Programmatischer Zugriff (6)

SIEMENS
Ingenuity for life

- Versenden des Datenpakets an ThingSpeak

```
# attempt to publish this data to the topic
try:
    publish.single(topic, payload=tPayload,
    hostname=mqttHost, port=tPort, tls=tTLS,
    transport=tTransport)

except (KeyboardInterrupt):
    break

except:
    print ("There was an error while publishing
the data.")

# Ende
```

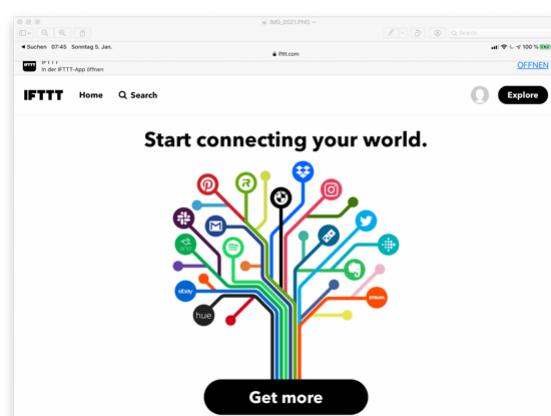
© Michael Stal,2020
Page 77

77

Zurück ans PIR mit IFTTT

SIEMENS
Ingenuity for life

- IFTTT ist ein Cloud-Dienst
- Es steht für **If This Then That**
- Es verbindet Geräte und Dienste
 - Etwa ein praktisches Beispiel
 - This = Wenn NASA ein neues Astronomie-Foto bereitstellt
 - That = Dann sende dieses Foto an meine E-Mail-Adresse
 - Oder
 - This = Wenn jemand einen Knopf meiner Elektronikschaltung drückt
 - That = Aktiviere meine Hue-Lampen
- Sowohl das This als auch das That lassen sich selbst bereitstellen oder aus der Bibliothek zahlreicher IFTTT-Komponenten nutzen



© Michael Stal,2020
Page 78

78

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Wir bauen erneut unsere PIR-Schaltung auf
- Immer wenn der Sensor eine Bewegung erkennt, soll IFTTT eine E-Mail an eine von uns festgelegte Adresse versenden
- Dazu wählen wir die Option Create your own auf der IFTTT-Webseite IFTTT.COM
- Zunächst müssen wir das This definieren



If
+ This
Then
That

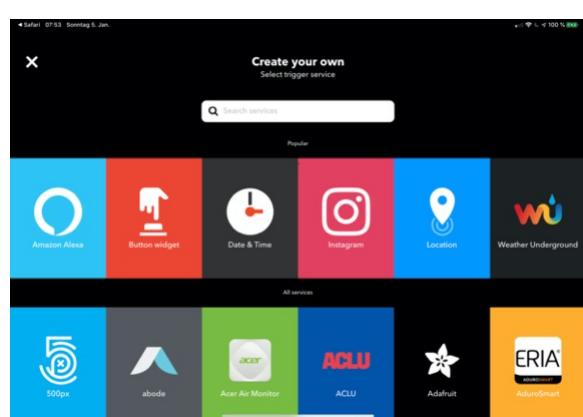
© Michael Stal,2020
Page 79

79

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Nach Anklicken von This erscheint eine Auswahlspalette
- Im Suchfeld geben wir Webhooks ein
- Mittels Webhooks ist es möglich, eigenständig Ereignisse an IFTTT zu senden (ein Trigger), das daraufhin die gewünschte Aktion auslöst



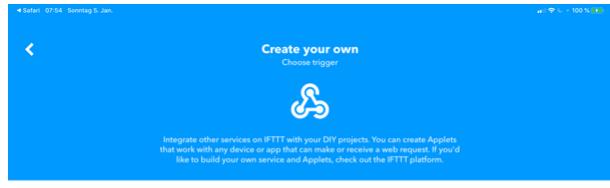
© Michael Stal,2020
Page 80

80

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Innerhalb des Webhooks-Dialogs lassen sich Web-Requests festlegen



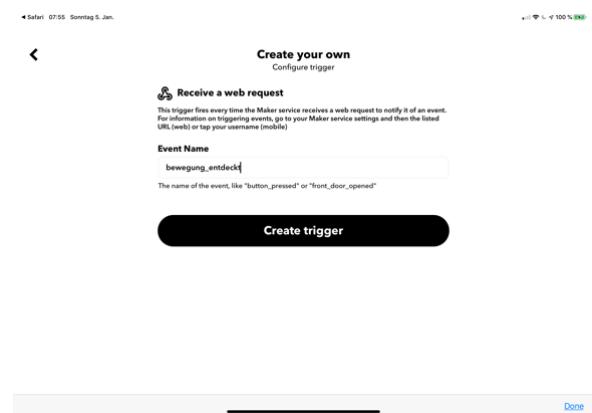
© Michael Stal,2020
Page 81

81

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Unser Event heißt `bewegung_entdeckt`



© Michael Stal,2020
Page 82

82

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Nach Betätigen von Create Trigger erhalten wir weitere Information wie der Aufruf aussehen soll, unter anderem einen geheimen Schlüssel
- Nun legen wir fest, was passieren soll, wenn der Webhook unseren Request erhält
- Kurz gesagt, wir definieren das That



If
Then
+That

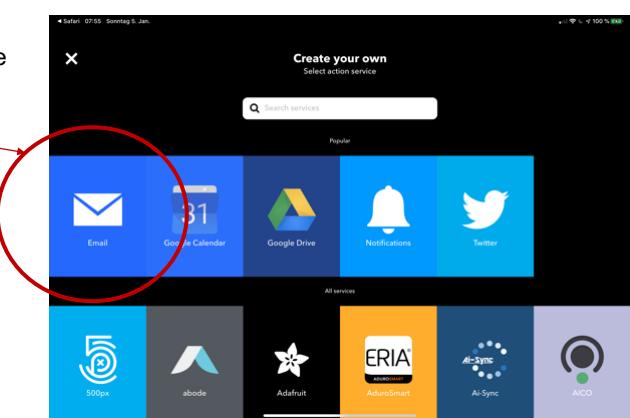
© Michael Stal,2020
Page 83

83

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Wir könnten IFTTT veranlassen einen Tweet zu senden oder eine Nachricht an unser Smartphone
- ... oder eine E-Mail



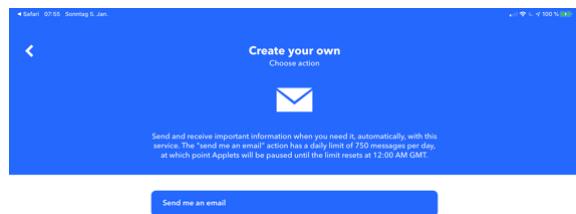
© Michael Stal,2020
Page 84

84

Erzeugen eines eigenen IFTTT-Dienstes

SIEMENS
Ingenuity for life

- Danach erhalten wir den Dialog
- Klicken Sie auf Send me an email



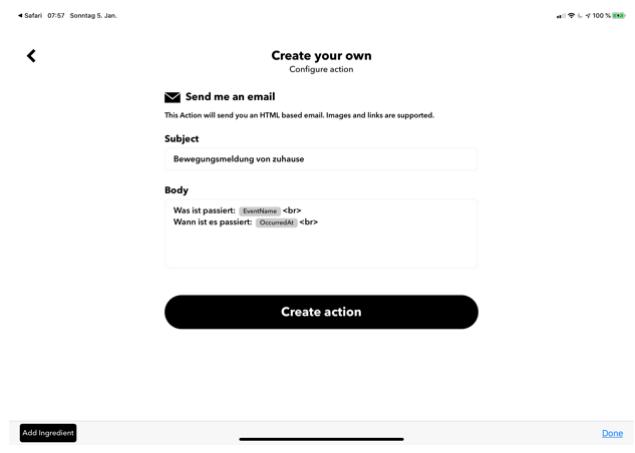
© Michael Stal,2020
Page 85

85

Gerüst der gewünschten Mail

SIEMENS
Ingenuity for life

- Geben Sie nun die gewünschte Überschrift an
- Im Body spezifizieren Sie die Inhaltszeilen Was ist passiert? und Wo ist es passiert
- Betätigen Sie nun Create action
- Und jetzt heißt es programmieren



© Michael Stal,2020
Page 86

86

Programmierung mit PIR und IFTTT (1)

SIEMENS
Ingenuity for life

- Für das Programm benötigen wir diverse Bibliotheken, unter anderem requests
 - Solte das noch nicht installiert sein, dann sudo pip3 install requests
- Zudem erfolgt die Festlegung
 - Leuchtdauer (LED) bei Bewegung
 - Pin für PIR
 - Pin für LED

```
import sys
from time import sleep
import RPi.GPIO as GPIO
import requests

# Leuchtdauer bei Bewegungserkennung
LED_ON_TIME = 10
# Verwendete LED
LEDPIN = 26
# PIR Sensor
PIRPIN = 7

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 87

87

Programmierung mit PIR und IFTTT (2)

SIEMENS
Ingenuity for life

- Definition der GPIO-Ein- und Ausgänge

```
# Verwendet werden die physikalischen Pins
GPIO.setmode(GPIO.BOARD)
# Keine Warnungen wie "Port ist bereits belegt"
GPIO.setwarnings(False)
# LEDPIN ist Ausgabeport
GPIO.setup(LEDPIN, GPIO.OUT)

# PIRPIN greift den Status des PIR-Sensors ab
GPIO.setup(PIRPIN, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 88

88

Programmierung mit PIR und IFTTT (3)

SIEMENS
Ingenuity for life

- Definition des „Eventhandlers“
- Der Eventhandler sendet einen POST-Request an den IFTTT-Webhook-Service
- ... und lässt die LED für die festgelegte Dauer leuchten

```
# Ereignisbehandlung, wenn Bewegung erkannt
def bewegung_erkannet(pin):
    print("Bewegung registriert")
    # Code fuer IFTTT-Post
    data =
    requests.post("https://maker.ifttt.com/trigger/bewegungserkannt/with/key/<<Ihr
IFTTT-Webhook-Key>>")
    print (data.text)
    # LED an bei Bewegungserkennung
    GPIO.output(LEDPIN, GPIO.HIGH)
    sleep(LED_ON_TIME)
    GPIO.output(LEDPIN, GPIO.LOW)
    return
# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 89

89

Programmierung mit PIR und IFTTT (4)

SIEMENS
Ingenuity for life

- Nun tragen wir die Methode `bewegung_erkannet` bei der GPIO eine
- Dieser haben wir zuvor bekanntgegeben, dass wir nur bei steigender Signallanke am `PIRPIN` reagieren wollen
- Am Schluss erfolgt eine „unendliche“ `while`-Schleife
- Bei Programmlauf müsste jede Bewegungserkennung über IFTTT den Versand einer Mail auslösen

```
# Auslösung der Ereignisse durch
# aufsteigende Signallanken
GPIO.add_event_detect(PIRPIN, GPIO.RISING)
GPIO.add_event_callback(PIRPIN, bewegung_erkannet)

print("CONTROL-C beendet das Programm")

try:
    while True: # Endlosschleife
        sleep(0.5)
except KeyboardInterrupt: # Schleifenende bei Interrupt
    GPIO.cleanup()
    sys.exit()

# Ende
```

© Michael Stal,2020
Page 90

90



Bosch Sensortec BME280



- ▶ **Baustein** mit integrierten Umweltsensoren
- ▶ **Druck:** Barometrischer Druck
- ▶ **Relative Feuchtigkeit:** Schnelle Messung der relativen Feuchtigkeit
- ▶ **Temperatur**

© Michael Stal, 2020
Page 91

91

Technische Daten

- **Sonstiges**
 - Temperaturbereich
 - -40 ... 85 °C
- **Ausführung**
 - Luftfeuchtigkeit
 - 0-100%
- **Allgemeines**
 - Bauform
 - 8-Pin LGA
 - Ausführung
 - I²C und SPI
- **Elektrische Werte**
 - Spannung
 - 1,71 ... 3,60 V

Technical data	BME280 (prelim.)
Package dimensions	8-Pin LGA with metal 2.5 x 2.5 x 0.93 mm ³
Operation range (full accuracy)	Pressure: 300...1100 hPa Temperature: -40...+85 °C
Supply voltage V _{DIO} Supply voltage V _{DV}	1.2 ... 3.6 V 1.71 ... 3.6V
Interface	I ² C and SPI
Average current consumption (typ.) (1Hz data refresh rate)	1.8 µA @ 1 Hz (H, T) 2.8 µA @ 1 Hz (P, T) 3.6 µA @ 1 Hz (H, P, T) T=temperature
Average current con- sumption in sleep mode	0.1 µA
Humidity sensor	
Response time ($\tau_{63\%}$)	1s
Accuracy tolerance	±3% relative humidity
Hysteresis	≤ 2% relative humidity
Pressure sensor	
RMS Noise	0.2 Pa (equiv. to 1.7 cm)
Sensitivity Error	±0.25% (equiv. to 1 m at 400 m height change) ±11.5 Pa/K
Temperature coefficient offset	(equiv. to ±12.6 cm at 1°C temperature change)
RoHS compliant, halogen-free, MSL1	



Datenblätter:

- https://cdn-reichelt.de/documents/datenblatt/B400/BST-BME280_DS001-10.pdf
- https://cdn-reichelt.de/documents/datenblatt/B400/BOSCH SENSORTEC FLYER_BME280.pdf

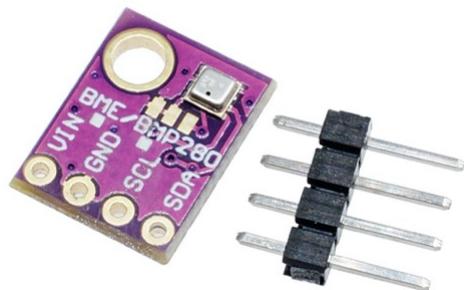
© Michael Stal, 2020
Page 92

92

Das verwendete BME280-Sensor-Modul

SIEMENS
Ingenuity for life

- Das Board nutzt I²C
- Pins:
 - VIN: Eingangsspannung
 - GND: Erde
 - SCL: I²C Control
 - SDA: I²C Data



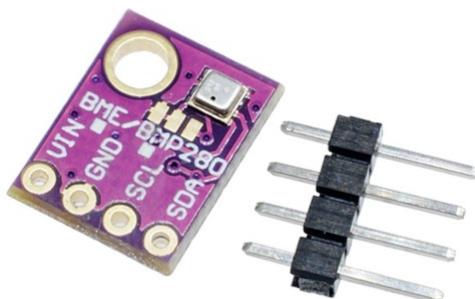
© Michael Stal, 2020
Page 93

93

Löten der Header-Pins an das BME240 Breakout-Board

SIEMENS
Ingenuity for life

- Header-Pins mit den kurzen Seiten von unten in das Board stecken (d.h. kurze Pinseite zeigt nach oben)
- Beides auf Breadboard stecken
- Gegenstand (z.B. Münzen) unterlegen, damit das Board parallel zum Breadboard liegt
- Jetzt die 4 Header-Pins von oben auf das Board löten



© Michael Stal, 2020
Page 94

94

BME280 Schaltung

SIEMENS
Ingenuity for life

Achtung: Auf dem rechts in der Schaltung abgebildeten BME280-Modul ist die Reihenfolge der Pins anders:
 GND
 VIN
 SDA
 SCL

© Michael Stal, 2020
 Page 95

95

Konfiguration von I²C auf dem Raspberry Pi: Schritt 1

SIEMENS
Ingenuity for life

Aufruf der Konfiguration
`sudo raspi-config`

Drei Schritte durchführen (siehe Screenshots), um I²C zu aktivieren

Zuerst: Im Hauptmenü Interface Options auswählen

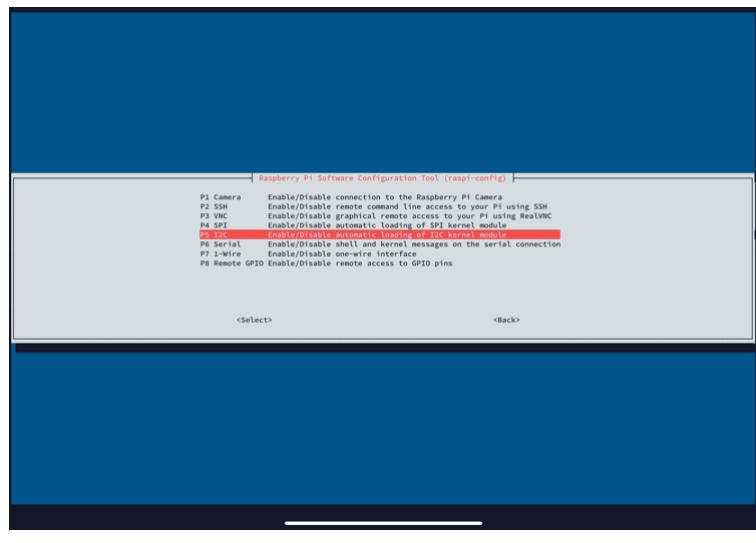
© Michael Stal, 2020
 Page 96

96

Konfiguration von I²C auf dem Raspberry Pi: Schritt 2

SIEMENS
Ingenuity for life

Im Submenü I²C auswählen



© Michael Stal,2020
Page 97

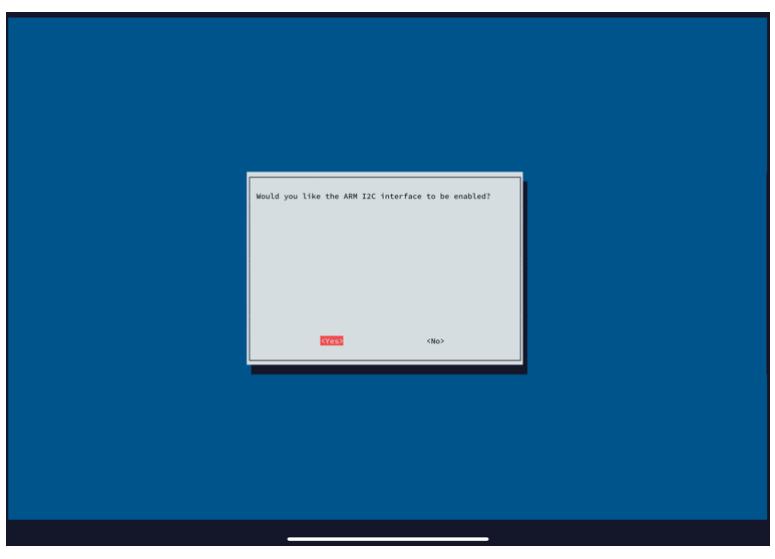
97

Konfiguration von I²C auf dem Raspberry Pi: Schritt 3

SIEMENS
Ingenuity for life

Im Dialog YES auswählen

Danach raspi-config verlassen und rebooten!



© Michael Stal,2020
Page 98

98

Wo genau hängt das BME-Modul am I²C?

- i2cdetect -y 1
=> Tabelle rechts
- Das BME280-Modul verwendet Adresse 0x76
- Für nachfolgende Programmierung des I²C (auch bekannt als SMBus) ist die Bibliothek smbus bzw smbus2 nötig
 - Installation auf Raspi über pip3 install smbus2

```
pi@raspberrypi:~ $ i2cdetect -y 1
  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: --
10: --
20: --
30: --
40: --
50: --
60: --
70: -- 76 --
pi@raspberrypi:~ $
```

© Michael Stal,2020
Page 99

99

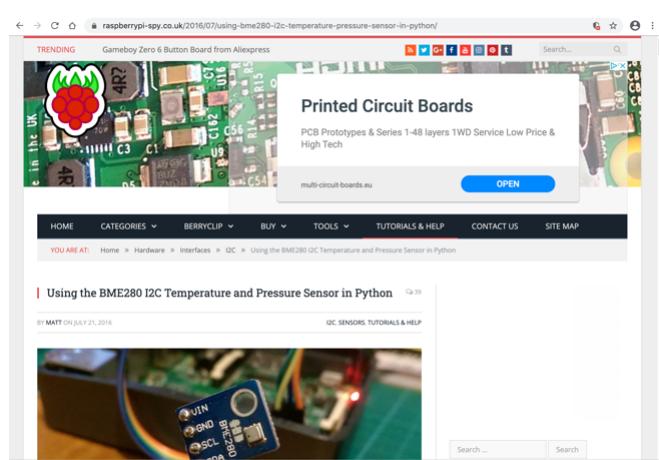
Bibliothek für BME280

SIEMENS
Ingenuity for life

Laden über

```
wget https://bitbucket.org/MattHawkinsUK/rpispky-misc/raw/master/python/bme280.py
```

Bibliothek auf Verzeichnis kopieren, in dem die BME280-Programmdateien liegen



© Michael Stal,2020
Page 100

100

Testen des BME280 über einfaches Programm

SIEMENS
Ingenuity for life

Durch Nutzen der
Bibliothek `bme280`
für den BM280
gestaltet sich der
Zugriff sehr einfach

```
from bme280 import readBME280All
from time import sleep

print( "---- STARTEN DER MESSWERTERFASSUNG VOM BME280 ----")
while True:

    temperature,pressure,humidity = readBME280All()
    print( "Temperatur : " + str(temperature) + " C" )
    print( "Luftdruck : " + str(pressure) + " hPa" )
    print( "Feuchtigkeit : " + str(humidity) + " %" )
    print( "-----");
    sleep(2)
```

© Michael Stal,2020
Page 101

101

Ein etwas komplexeres Beispiel (1)

SIEMENS
Ingenuity for life

- Periodisches Übertragen von Wetterdaten an ThingSpeak
 - Kanal einrichten
 - ChannelID und Writer API Key vermerken
 - Nachfolgendes Programm sendet alle 20 Sekunden die Messdaten an den ThingSpeak-Kanal

```
from __future__ import print_function
import paho.mqtt.publish as publish

from bme280 import readBME280All
from time import sleep

### Start of user configuration #####
# ThingSpeak Channel Settings
# The ThingSpeak Channel ID
# Replace this with your Channel ID
channelID = << Your Channel ID>>

# The Write API Key for the channel
# Replace this with your Write API key
apiKey = << Your Write API Key >>
# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 102

102

Ein etwas komplexeres Beispiel (2)



- Wegen des Mehraufwands für die Integration von SSI-Keys verwenden wir ungesichertes TCP zur Datenübertragung
- Für sensitive Information natürlich nicht empfehlenswert!

```
# MQTT Connection Methods
# Set useUnsecuredTCP to True to use the default MQTT port of 1883
# This type of unsecured MQTT connection uses the least amount of system
resources.

useUnsecuredTCP = True

# Set useUnsecuredWebSockets to True to use MQTT over an unsecured
websocket on port 80.

### End of user configuration ###

# The Hostname of the ThingSpeak MQTT service
mqttHost = "mqtt.thingspeak.com"

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 103

103

Ein etwas komplexeres Beispiel (3)



- Der genutzte Port für den MQTT-Broker (von ThingSpeak) lautet 1883

```
# Set up the connection parameters based on the connection type
if useUnsecuredTCP:
    tTransport = "tcp"
    tPort = 1883
    tTLS = None

# Create the topic string
topic = "channels/" + channelID + "/publish/" + apiKey

# Run a loop which calculates the system performance every
# 20 seconds and published that to a ThingSpeak channel
# using MQTT.
# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 104

104

Ein etwas komplexeres Beispiel (4)

SIEMENS
Ingenuity for life

- In der `while`-Schleife liest das Programm die Daten des BME280, baut die Nutzdaten auf und sendet sie an den MQTT-Broker
- Beenden des Programmes über `<control> + <C>`

```
while(True):
    temperature, pressure, humidity = readBME280All()
    # build the payload string
    print ("Temperature in C %5.2f Humidity in Percentage %5.2f Air Pressure in
hPa %5.2f" %(temperature, humidity, pressure))
    tPayload = "field1=" + str(temperature) + "&field2=" + str(humidity) + "&field3="
+ str(pressure)
    # attempt to publish this data to the topic
    try:
        publish.single(topic, payload=tPayload, hostname=mqttHost, port=tPort,
tls=tTLS, transport=tTransport)
    except (KeyboardInterrupt):
        break
    except:
        pass
    # Ende
```

© Michael Stal,2020
Page 105

105

Raspberry Pi Zero WH plus Raspi Cam

SIEMENS
Ingenuity for life



- Camera mit 2 Megapixeln
- Anschluss über mitgeliefertes Kabel
- Kabelaustausch: Dazu an den CSI-Ein/Ausgängen schwarze Plastikklammer vorsichtig nach vorne (d.h. von dem Anschluss weg-) ziehen
- Neues Kabel einstecken, zur Befestigung Klammer wieder nach innen drücken, fertig!

© Michael Stal,2020
Page 106

106

Konfiguration der Kamera über Raspbian (1)

SIEMENS
Ingenuity for life

- Auf Raspberry Pi Zero:
`sudo raspi-config`
- Auf Raspberry Pi Zero:
 5 Interfacing Options wählen
- Beim Folgedialog:
 P1 Camera wählen



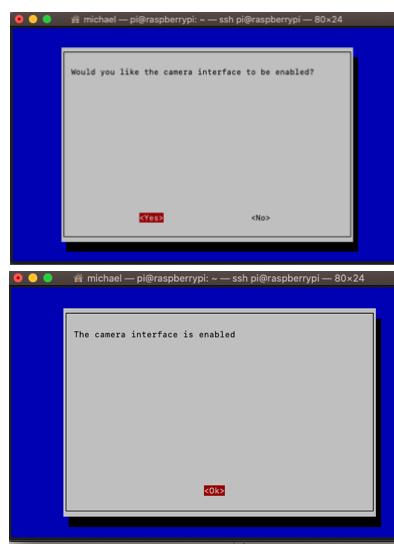
© Michael Stal,2020
Page 107

107

Konfiguration der Kamera über Raspbian (2)

SIEMENS
Ingenuity for life

- Jetzt bei Frage nach Aktivierung der Kamera:
 <Yes> wählen
- Im Dialog:
 <Ok> wählen
- Konfigurationstool verlassen und Reboot
`sudo reboot`
- Nach Neustart Testaufnahme mit:
`raspistill -o test.jpg`



© Michael Stal,2020
Page 108

108

Demoanwendung (3)

- Bei Programmabbruch sollte die Anwendung ebenfalls den Port zur Kamera schließen



```
GPIO.add_event_detect(PIR_PIN, GPIO.RISING)
GPIO.add_event_callback(PIR_PIN, bewegungsmeldung)
```

try:

```
    while True:
        time.sleep(1)
    except KeyboardInterrupt:
        GPIO.cleanup()
        cam.close()
        sys.exit()
```

Ende

© Michael Stal,2020
Page 109

109

Demoanwendung (1)



- Der PIR-Sensor kommt wieder zum Einsatz
- Bibliothek picamera ist nötig
sudo pip3 picamera
- Es kommen die üblichen Pins der PIR-Schaltung zum Einsatz
- Neben GPIO muss jetzt auch die Kamera initialisiert werden

```
#!/usr/bin/python3
import picamera, sys, time
import RPi.GPIO as GPIO

PIR_PIN = 7
SLEEP_TIME = 10
LED_PIN = 26

GPIO.setmode(GPIO.BOARD)
cam = picamera.PiCamera()
cam.resolution = (1920, 1080)
GPIO.setup(PIR_PIN, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)
GPIO.setup(LED_PIN, GPIO.OUT)

# Fortsetzung auf nächster Folie
```

© Michael Stal,2020
Page 110

110

Demoanwendung (2)

SIEMENS
Ingenuity for life

- Im Eventhandler ist nun zusätzlich ein Trigger, der die Kamera auslöst
cam.capture()
- Der Name des Fotos enthält Datum und Uhrzeit
- Das Foto wird im Programmordner abgelegt. Es ließe sich aber auch ein absoluter Pfad angeben

```
def bewegungsmeldung(pin):
    tme = time.strftime("%Y_%m_%d-%H:%M:%S")
    cam.capture("Foto-%s.jpg" %tme)
    GPIO.output(LED_PIN, GPIO.HIGH)
    print("Foto gespeichert")
    time.sleep(SLEEP_TIME)
    GPIO.output(LED_PIN, GPIO.LOW)
    return

GPIO.add_event_detect(PIR_PIN, GPIO.RISING)
GPIO.add_event_callback(PIR_PIN, bewegungsmeldung)

# Fortsetzung auf nächster Folie
```

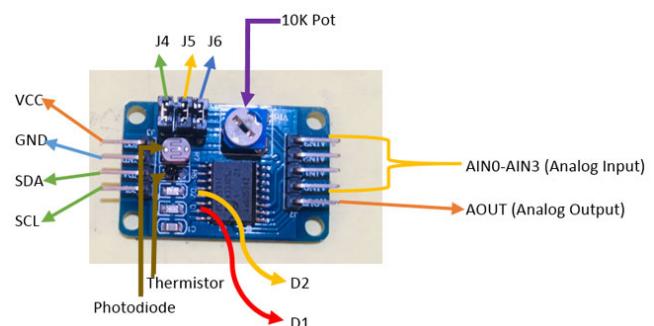
© Michael Stal,2020
Page 111

111

A/D-Wandler - Gut muss nicht teuer sein

SIEMENS
Ingenuity for life

- Der Raspberry Pi besitzt keine analogen GPIO-Eingänge bzw. -Ausgänge
- Viele Sensoren funktionieren analog
- Mögliche Lösung: Kauf eines Gert-Boards oder eines Gertduino (teuer!!!)
- Die kostengünstige Alternative heißt **PCF8591**
 - < 10 Euronen
 - Anschluss über I²C-Bus, 3.3V
 - Enthält A/D-Ausgänge und -Eingänge
 - Zusätzlich vorhanden: Poti (Regelbarer Widerstand), ein temperaturabhängiger Widerstand, teilweise ein Temperatursensor

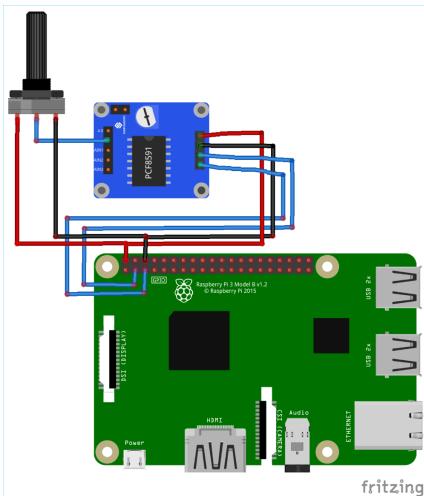


© Michael Stal,2020
Page 112

112

Beispielsschaltung

SIEMENS
Ingenuity for life



- Links ist eine Beispielschaltung zu sehen
- Ein Poti fungiert als analoger Eingabesensor
- Der PCF8591 lässt sich an 5V oder 3.3V und GND anschließen (besser an 3.3V)
- Verbinden über I²C erfordert Pins 3 und 5
- Voraussetzung: I²C installiert: siehe Folien 96 ff.
- Test des Anschlusskanals mit `i2cdetect -y 1`

© Michael Stal, 2020
Page 113

113

Zugriff auf PCF8591 über Python (1)

SIEMENS
Ingenuity for life

- Import der Bibliotheken
- Initialisierung des I²C-Busses
- Und Definition einer Hilfsmethode zum Lesen von Daten

```
#!/usr/bin/python3

from smbus2 import SMBus # i2c
import time

i2cbus = SMBus(1)
adr = 0x48

def read(cmd):
    write = i2cbus.write_byte_data(adr, cmd, 0)
    read = i2cbus.read_byte(adr)
    return read

# Fortsetzung auf nächster Folie
```

© Michael Stal, 2020
Page 114

114

Zugriff auf PCF8591 über Python (2)

SIEMENS
Ingenuity for life

- Es werden Bytes empfangen
- Alle Werte liegen zwischen 0 und 255
- Beispielsweise wird nicht die tatsächliche Spannung am Poti ausgegeben sondern ein Wert zwischen 0 (dunkel) und 255 (hell)
- Mehr Infos zum PCF8581 finden Sie in dem Datenblätter-Verzeichnis

```
for i in range(100):
    licht = read(0x41)
    print("Lichtwiderstand " + str(licht))
    temp = read(0x42)
    print("Temperatur " + str(temp))
    ain2 = read(0x43)
    print("Analogeingang " + str(ain2))
    poti = read(0x40)
    print("Poti-Spannung " + str(poti))
    time.sleep(2)

# Ende
```

© Michael Stal,2020
Page 115

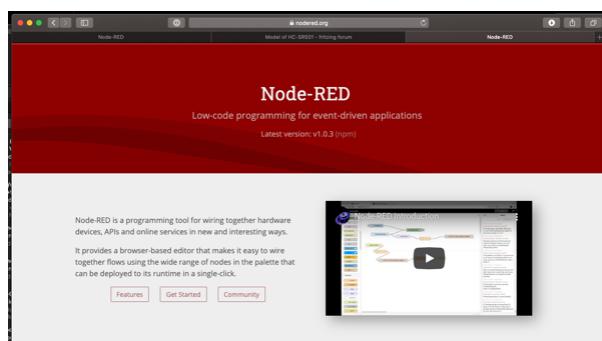
115

Integration über Node-RED

SIEMENS
Ingenuity for life

„Node-RED ist ein Programmierwerkzeug, um Hardwaregeräte, APIs und Online-Dienste auf neue und interessante Weise miteinander zu verbinden.“

Es bietet einen Browser-basierten Editor, der es einfach macht, Flows aus einer breiten Palette von Knoten (Nodes) zu kreieren, die sich mit nur einem einzigen Klick in seiner Laufzeitumgebung bereitstellen lassen“



Node-RED basiert auf JavaScript und Node.js

© Michael Stal,2020
Page 116

116

Node-RED Installation

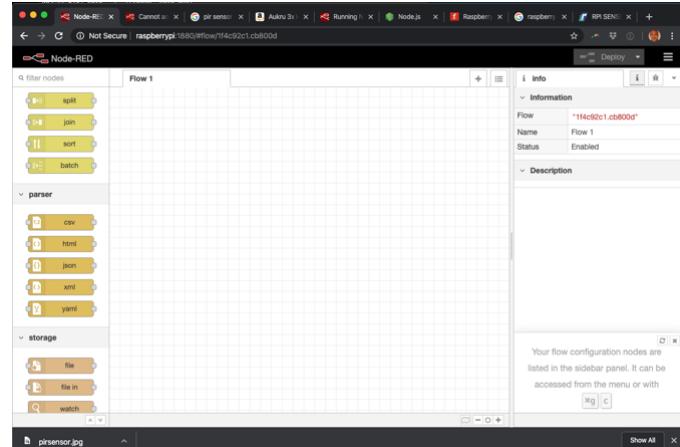
SIEMENS
Ingenuity for life

- Installation, wenn noch nicht vorhanden
(bitte vorher testen über Kommando node-red-pi)
 - sudo apt-get install build-essential
 - sudo apt-get install nodered
 - node-red-pi

Nun über PC (Windows, Mac, Linux) und Browser auf

`http://<raspiname>:1880`
navigieren (sobald Node-RED hochgefahren ist)

Es erscheint ein Menü wie rechts im Bild zu sehen



© Michael Stal, 2020
Page 117

117

Mini-Tutorial auf der Raspberry Pi Projekt-Seite

SIEMENS
Ingenuity for life

Durchlaufen Sie das Tutorial auf der folgenden Webseite:

<https://projects.raspberrypi.org/en/projects/getting-started-with-node-red/>

Hinweise:

- Starten von Node-RED mit `node-red-pi`
- Das Tutorial spricht von einem 330 Ohm Widerstand. Es genügt jeder Widerstand ≥ 162.5 Ohm

© Michael Stal, 2020
Page 118

118

Mini-Hackathon

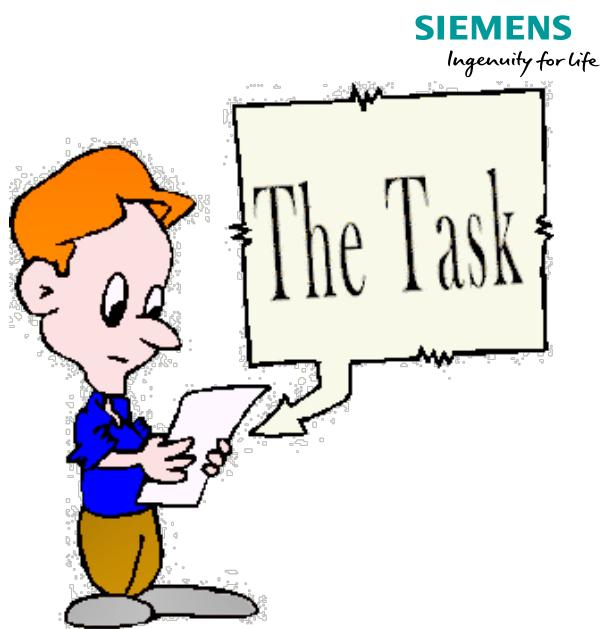
Unter der Verwendung der verfügbaren Bauteile bauen Sie in Gruppen (2 - 4 Personen) ein kleines IoT-System

Es ist natürlich erlaubt, sämtliche Komponenten aller beteiligten Personen zu nutzen

Die Gruppen entscheiden selbst, was sie bauen

Es gibt keine Bewertung. Die einzige Vorgabe ist Edutainment, d.h. Spaß

© Michael Stal,2020
Page 119



119



Das Finale

© Michael Stal,2020
Page 120

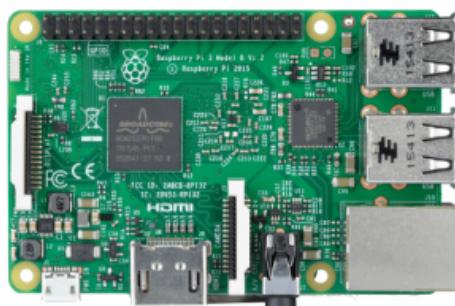
120

Es gibt viele Ideen für weitere Projekte

- Tensorflow Lite zur Objekterkennung über Raspi-CAM
- Messung von Feinstaub, Regen, Wind, UV, Radioaktivität, CO₂
- Tensorflow-Lite zur Anomalieerkennung, z.B. bei Wetterdaten
Empfehlung: RNNs (Recurrent Neural Networks) mit LSTM-Zellen (= Long-Short-Term-Memory) oder Autoencoder oder schlicht Gaußsche Verteilung für mehrdimensionale Räume
- Roboterbau (Drohnen, Fahrzeuge)
- Heimautomatisierung
- Mobiler Einsatz (d.h., externe Stromversorgung mit Akkus)
- LoRA-Nutzung
- AlexaPi: Konstruktion eines eigenen Amazon Echo-Geräts (benötigt Sound- bzw. Audio-HAT sowie LED-Ring)
- Nutzung von Docker: Achtung auf der ARM6-Architektur sind nicht alle Container lauffähig
- ... und vieles mehr!

SIEMENS
Ingenuity for life

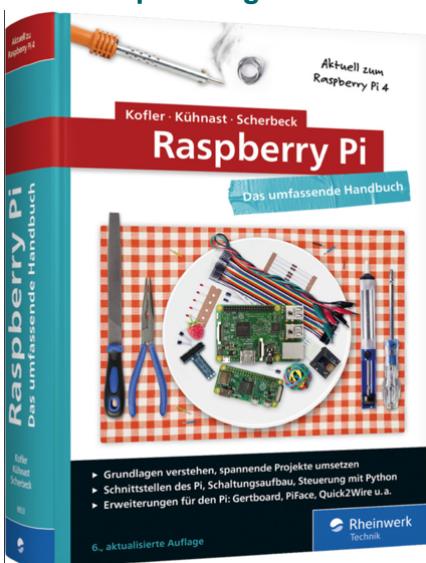
Raspberry Pi Projects



© Michael Stal, 2020
Page 121

121

Buchempfehlung



SIEMENS
Ingenuity for life

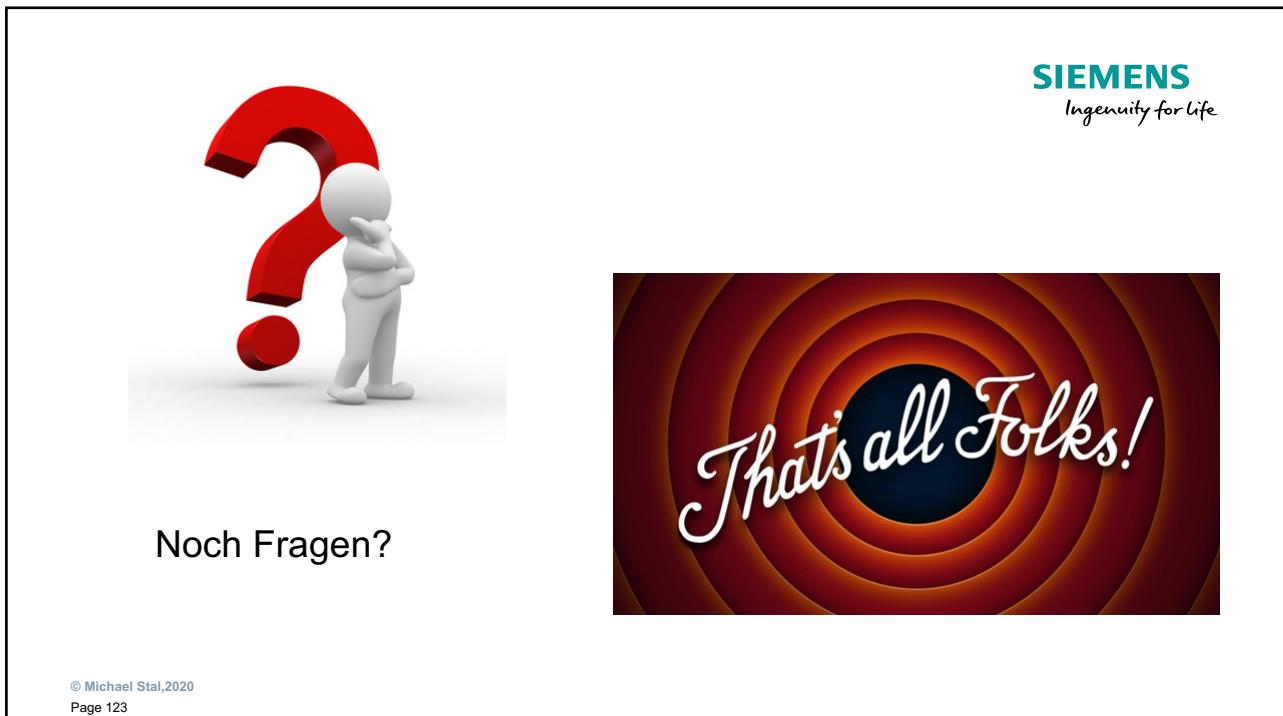
Bezug zum Beispiel über

https://www.amazon.de/Raspberry-Pi-umfassende-Handbuch-Tekkies/dp/3836269333/ref=sr_1_1?_mk_de_DE=AMA%CC%84Z&keywords=raspberry+pi+kofler&qid=1578223545&sr=8-1

Preis: 44,90 Euro

© Michael Stal, 2020
Page 122

122



Noch Fragen?

© Michael Stal, 2020
Page 123

123



124

SIEMENS
Ingenuity for life

Anhänge A-C

© Michael Stal,2020
Page 125

125

Addendum A:
MVP - Most
Valuable
Powertools

SIEMENS
for life



© Michael Stal,2020
Page 126

126

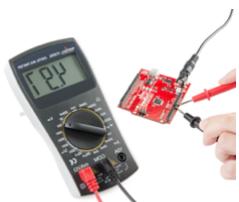
Addendum A (Fortsetzung)

Multimeter: 20-40€
Strom, Spannung (AC/DC, DC), Widerstand, Diodentester, Kapazität, Induktion

Oszilloskop: ca. 300€
Nohe Bandbreite, Frequenzzähler, Hohe Sampling-Rate, 2 Kanäle plus ein externer Kanal

Flexible Stromversorgung: 50-80€
Variables Einstellen von Spannung & Stromstärke (z.B. 0-30V bzw. 0-5A)



→ 

→ 

→ 

© Michael Stal, 2020
Page 127

127

Addendum B: Kondensatoren, Induktoren, und mehr ...

- Kondensatoren speichern Energie & glätten Signale, verfügbar in diversen Materialien (Keramik => Polung irrelevant, Elektrolyt. => Polung relevant). Wichtig für Hochpass/Tiefpass/Bandbass Filter
- Induktoren sind Spulen, die ein Magnetfeld erzeugen, sobald Strom fließt. Umgekehrt erzeugen sie Strom bei Bewegung durch ein Magnetfeld. Wichtig für Wandler und Motoren
- Dioden (LEDs sind auch Dioden) lassen Strom in nur einer Richtung passieren
- Transistoren nötig für Logikgates (z.B. Flip-Flops) und Schalter (um eine Schaltung mit hohen Strömen zu kontrollieren bzw. von einer Schaltung mit niedrigen Stromstärken zu trennen)



Kondensatoren, Induktoren haben einen dynamischen Widerstand (Impedanz)



© Michael Stal, 2020
Page 128

128

Addendum C: Nützliche ICs

Beispiele

- **555 Timer IC**: Timer, Taktgenerator, Oszillator
- **Half-Bridge** (SN754410 hat z.B. 4 Halbbrücken)* erlaubt Kontrolle über Stromrichtung (z.B. Motorbewegung in/gegen Uhrzeigersinn). Kann auch Schaltungen mit hohen Stromstärken kontrollieren.
- **Operationsverstärker** (z.B. 741, LM 358, LM386)* : verstärken Eingangssignal um das mehrere Hundert-/Tausend-fache => Als Vergleicher einsetzbar. Im Rückkopplungsmodus lassen sich verschiedene Transferfunktionen realisieren.
- **Schieberegister** (HC74HC595)*: nehmen ein Byte, konvertieren es zu einem Bitfeld, bei dem jedes Bit ein Ausgangsignal kontrolliert.
Beispieldwendung: Zahl der von einem Microcontroller benutzten Ausgänge reduzieren.
- **Optokoppler**(PC817)*: nutzen Licht, um von einem Stromkreis verbindungslos einen anderen zu kontrollieren.
- **Darlington Transistor**(ULN2003)* kombiniert 2 Transistoren, um die Verstärkung noch weiter zu erhöhen: genutzt zur Motorkontrolle.

SIEMENS
Ingenuity for life



Für weitere Details:

<http://arduino-info.wikispaces.com/Popular-ICs>

*In Klammern stehen jeweils
Beispield-ICs

© Michael Stal, 2020
Page 129

129