# 2.3 Funktionen

Julian Huber - Bussysteme 1 / 10

## **Funktionale Programmierung**

- Berechnung von Output aus Input wird in wieder aufrufbaren Funktionen gekapselt
- Funktion hat nur Input und Output aber keinen Speicher / Zustand

```
def add(a, b):
    return a + b

add(1, 2) # 3
add(3, 4) # 7
```

Julian Huber - Bussysteme 2/10

#### Steuerfunktion

$$L_{ ext{SET}} = (P_{ ext{ACT}} \wedge (H_{ ext{ROOM}} < ext{PAR}_{ ext{SETPT}})) ee L_{ ext{MAN}}$$

```
def l_set(p_act, h_room, PAR_SETPT, l_man):
    return (p_act and (h_room<PAR_SETPT)) or l_man</pre>
```

```
from tageslichtschaltung import l_set
from mapping import map_quat

while True:
    l_set_value = l_set(p_act, h_room, PAR_SETPT, l_man)
```

Julian Huber - Bussysteme 3 / 10

## 

- Implementieren Sie die Tageslichtschaltung in Python
- Stellen Sie zunächst sicher, dass LED, Button und Analog-Digital-Wandler korrekt angeschlossen sind
- Setzen Sie die manuelle Einstellung 1\_man dauerhaft auf False
- Setzen Sie den Sollwert PAR\_SETPT auf einen geigneten Wert
- Legen Sie die beiden Module tageslichtschaltung.py und mappings.py in den gleichen Ordner wie Ihre Hauptdatei
- PRecherchieren Sie einen geeigneten Sensor, den Sie für die Anwesenheitserkennung verwenden können

Julian Huber - Bussysteme 4 / 10

## Möglicher Startpunkt

```
import board
import analogio
import time
from mappings import map quat
from tageslichtschaltung import l_set
import digitalio
# Initialisierung des ADC (Analog-Digital Converter)
ldr = analogio.AnalogIn(board.A2)
# Initialisierung der LED
led pin = board.GP1
                        # Replace with the GPIO pin connected to your LED
led = digitalio.DigitalInOut(led_pin)
led.direction = digitalio.Direction.OUTPUT
# Initialisierung Button
button pin = board.GP0 # Replace with the GPIO pin connected to your button
button = digitalio.DigitalInOut(button pin)
button.direction = digitalio.Direction.INPUT
button.pull = digitalio.Pull.UP # Use pull-up resistor; change if using pull-down
# Parameter setzen
PAR SETPT = 100
1 man = False
# Wiederholung
while True:
    # ADC als Dezimalzahl lesen
    read = ldr.value
    # Ausgabe in der Kommandozeile/Shell
    print("ADC:", read)
    print("E in Lux", map quat(read))
```

Julian Huber - Bussysteme 5 / 10

## tageslichtschaltung.py

```
def l_set(p_act, h_room, PAR_SETPT, l_man):
    return (p_act and h_room<PAR_SETPT) or l_man</pre>
```

Julian Huber - Bussysteme 6 / 10

### mappings.py

```
def map_lin(z):
    E max = 1
    E min = 0
    z_{max} = 65535
    z \min = 0
    beta_0 = E_min
    beta_1 = (E_max - E_min) / (z_max - z_min)
    return beta 0 + beta 1 * z
def map_quat(x):
    s = 44000
   a = 0.0015
    return ((x-s)*a) **2
```

## ✓ Lösung

??? optional-class " 💡 anzeigen"

## 

- Welche Teile des Codes könnte man ebenfalls in Funktionen auslagern?
- Wie schätzen Sie den Aufwand ein, wenn man nun weitere Tageslicht-Schaltungen mit anderen LEDs und Sensoren auf der gleichen Platine realisieren möchte?

## √ Lösung

- Initialisierung, da die Code immer gleich ist und sich nur je nach Aufbau die Pins ändert
- Umrechnungen
- Einfacher, wenn mehr (z.B. auch die Zuweisung der Ein- und Ausgänge) in Funktionen ausgelagert wird

Julian Huber - Bussysteme 8 / 10

### Sichtbarkeit von Variablen

#### Lokale Variablen

- Variablen, die innerhalb einer Funktion definiert werden (z.B. s) sind außerhalb der Funktion nicht sichtbar (*Kapselung*)
- Dies gilt für die meisten Programmiersprachen und z.B. auch für Schleifen

```
def map_quat(x):
    s = 44000
    a = 0.0015
    return ((x-s)*a) **2

print(s)
# NameError Traceback (most recent call last)
# <ipython-input-11-76c4dd40fb41> in <module>
# ----> 1 print(s)

# NameError: name 's' is not defined
```

Julian Huber - Bussysteme 9 / 10

#### Globale Variablen

- Variablen, die (bewusst) überall im Programmcode aufrufbar sind (z.B.
   PAR\_SETPT ) sind globale Variablen
- in Python werden globale Variablen in Großbuchstaben geschrieben

```
A_GLOBAL_VAR = 1

def my_function():
    a_local_variable = 2
    return a_local_variable

another_variable = my_function()

print(A_GLOBAL_VAR) # 1
print(a_local_variable) # Error
print(another_variable) # 2
```

Julian Huber - Bussysteme 10 / 10