

# Einführung Gebäudetechnik: IT & Bussysteme

Dr. Julian Huber

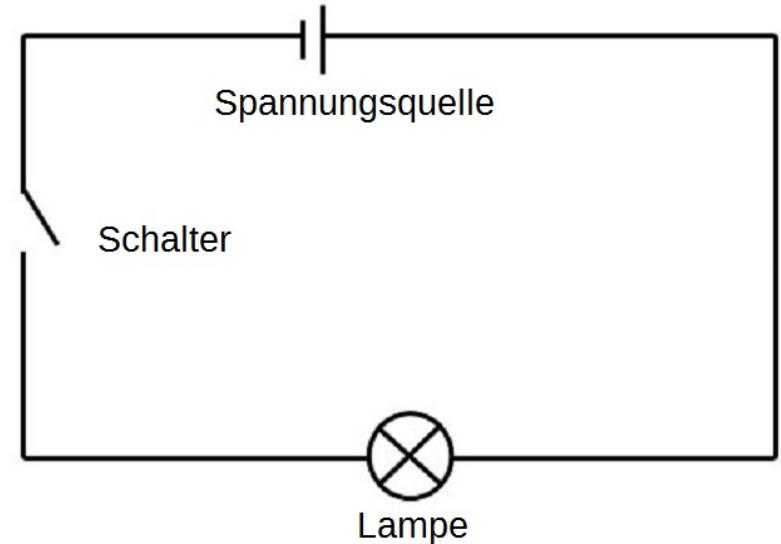
*Management Center Innsbruck*

# Beispiel Glühbirne



## Klassische Glühbirne

- **Energieeffizienz:** 5%
- **Lebensdauer:** 1.000 Stunden
- **Farbtemperatur:** fix z.B. 2.700 K
- **dimmbare Variante:** z.B. durch Unterspannung
- **Preis:** 1 €



code.py    diagram.json

```
1 # No code needed
```

Simulation

The screenshot shows a software interface for simulating a Raspberry Pi Pico. On the left, there's a code editor with a single line of code: "# No code needed". Above the code editor is a tab labeled "code.py" with a back arrow icon. To the right of the code editor is a dropdown menu showing "diagram.json" with a downward arrow icon. On the far right, there's a small circular icon with a link symbol. The main area contains a "Simulation" section with three buttons: a green circular button with a white circular arrow, a grey square button, and a grey button with a double vertical line. Below these buttons is a schematic diagram of a Raspberry Pi Pico board. The board is green with various pins and components labeled. A red LED is connected between pin 18 (labeled "LED") and ground. A push button is connected between pin 27 and ground. The breadboard setup is shown with green wires connecting the board pins to the respective components.



# Moderne LED-Glühbirne



## Moderne LED-Glühbirne



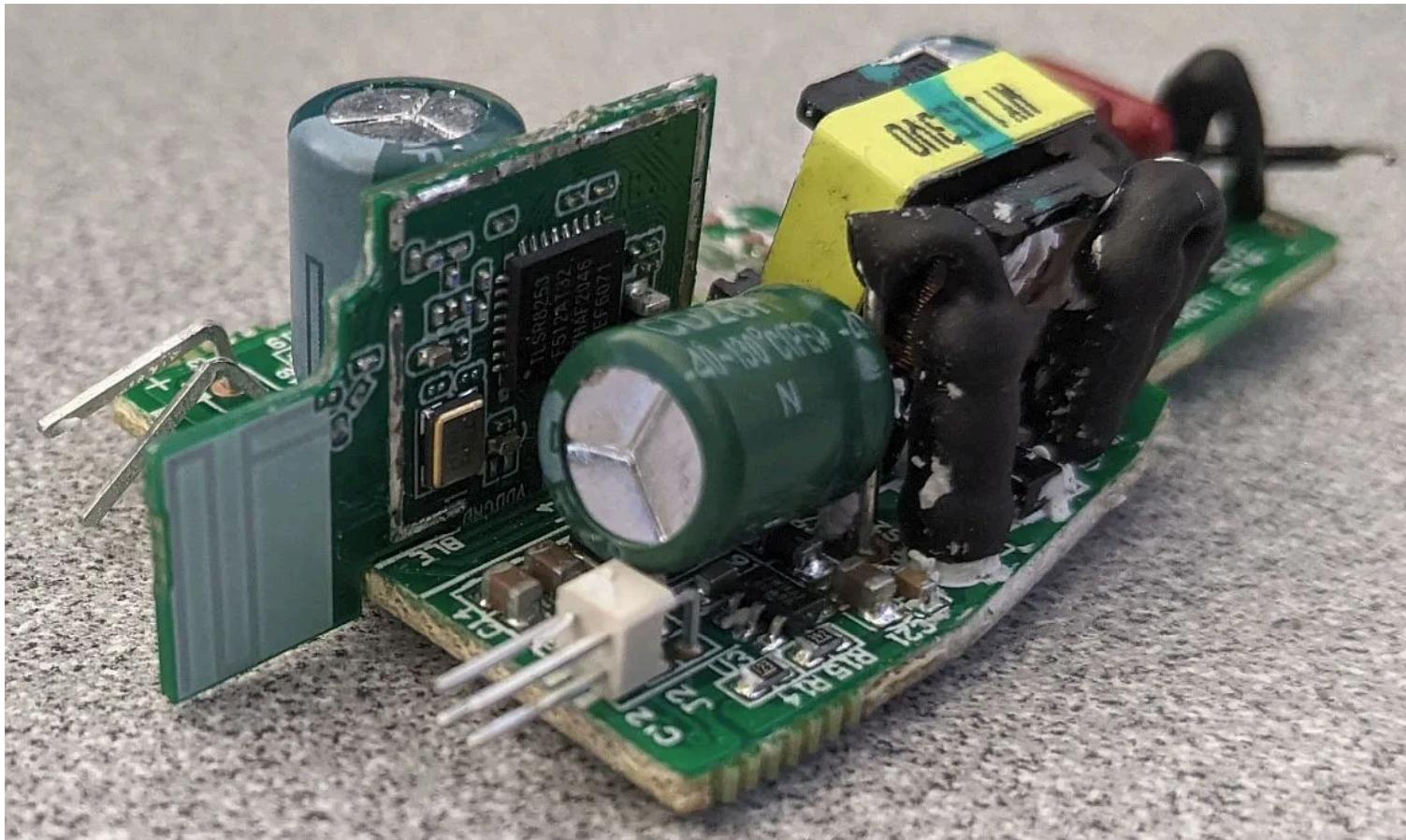
## Moderne LED-Glühbirne

- **Energieeffizienz:** 80%
- **Lebensdauer:** 25.000 Stunden
- **Farbe/temperatur:** variabel
- **dimmbare Variante:** z.B. durch Pulsweitenmodulation
- **Preis:** 10 €

5. Nachhaltigkeit und Energieeffizienz

## Eingebettetes System

- jedes Gerät hat einen eigenen Mikrocontroller
- **Software-Code** steuert die Farbe, Helligkeit, ...



- Trennung von Ein- und Ausgängen
- diese werden erst durch Software verknüpft

WOKwi

code.py diagram.json

```

1 import board
2 import digitalio
3 import time
4
5 # Define the LED
6 led = digitalio.DigitalInOut(board.GP28)
7 led.direction = digitalio.Direction.OUTPUT
8
9 # Define the button
10 button = digitalio.DigitalInOut(board.GP21) # Assuming GP21 is the button pin
11 button.direction = digitalio.Direction.INPUT
12 button.pull = digitalio.Pull.DOWN
13
14 # State variable to track button presses
15 button_last_state = False
16
17 # LED state (False = Off, True = On)
18 led_state = False
19
20 while True:
21     # Read the current state of the button
22     button_state = button.value
23
24     # Check if the button was pressed
25     if button_state and not button_last_state:
26         # If the LED is off, turn it on immediately
27         if not led_state:
28             led_state = True
29             led.value = led_state
30         # If the LED is on, delay switching it off by 2 seconds
31         else:
32             time.sleep(2)
33             led_state = False
34             led.value = led_state
35
36     # Update last button state
37     button_last_state = button_state
38
39     # Small delay to debounce the button
40     time.sleep(0.05)
41

```

Simulation





## Lernziele

- Konzeption von Schaltregeln als **Endliche Automaten** oder **Wahrheitstabellen**
- Anschluss von Sensoren und Aktoren an **Mikrocontroller** und  
**Speicherprogrammierbaren Steuerungen**
- Umsetzung in **Software-Programmen** mittels Kontrollstrukturen

# 3-Kanal-LED-Controller mit Pulsweitenmodulation

```
code.py    diagram.json
```

```
1 import board
2 import pwmio
3
4 # Initialize PWM outputs for each color channel
5 red_pwm = pwmio.PWMOut(board.GP28, frequency=5000, duty_cycle=0)
6 green_pwm = pwmio.PWMOut(board.GP27, frequency=5000, duty_cycle=0)
7 blue_pwm = pwmio.PWMOut(board.GP26, frequency=5000, duty_cycle=0)
8
9 # Function to set the color of the RGB LED
10 def set_rgb_color(red_value, green_value, blue_value):
11     # Convert 0-255 range to 0-65535 for PWM
12     red_pwm.duty_cycle = int((red_value / 255) * 65535)
13     green_pwm.duty_cycle = int((green_value / 255) * 65535)
14     blue_pwm.duty_cycle = int((blue_value / 255) * 65535)
15
16 # Example usage: Set initial color to purple (Red + Blue)
17 red_value = 255 # Example value for red channel (0-255)
18 green_value = 255 # Example value for green channel (0-255)
19 blue_value = 255 # Example value for blue channel (0-255)
20
21 set_rgb_color(red_value, green_value, blue_value)
22
23 while True:
24     # The main loop can be used to update the color dynamically if needed
25     set_rgb_color(red_value, green_value, blue_value)
```

Simulation

00:47.716 32°

The simulation shows the waveforms for the three PWM outputs. The red output starts at 0, goes to 100% duty cycle, then drops back to 0. The green output follows a similar pattern, starting at 0, going to 100%, then dropping back to 0. The blue output starts at 0, goes to 100%, then drops back to 0. This results in a purple color.





## Lernziele

- Einsatz von **Zahlensysteme** und **Bit-Operationen**
- Unterschiede zwischen **Analogen** und **Digitalen Signalen**
- Auswahl von **Ein- und Ausgabegeräten**

# Verbindung von Sensoren und Aktoren

WOKWi SAVE SHARE 6\_LED\_Bewegungsmelder Docs Profile

code.py diagram.json

```
1 import board
2 import digitalio
3 import time
4
5 # Set up the PIR motion sensor
6 pir_sensor = digitalio.DigitalInOut(board.GP26)
7 pir_sensor.direction = digitalio.Direction.INPUT
8
9 # Set up the LED
10 led = digitalio.DigitalInOut(board.GP28)
11 led.direction = digitalio.Direction.OUTPUT
12
13 while True:
14     if pir_sensor.value:
15         led.value = True # Turn on LED when motion is detected
16         time.sleep(2)
17         led.value = False
18     else:
19         led.value = False # Turn off LED when no motion is detected
20
21     time.sleep(0.1) # Small delay to debounce the sensor
```

Simulation 00:36.443 99%

The screenshot shows a Wokwi project interface. On the left, the code editor contains Python code for a Raspberry Pi Pico that reads from a PIR motion sensor and controls a red LED. On the right, the simulation window displays a schematic of the hardware setup. A Raspberry Pi Pico is connected to a PIR motion sensor and a red LED. The PICO\_GND pin is connected to the GND pin of the PIR and the cathode of the LED. The PICO\_GP26 pin is connected to the PIR's VCC pin and the anode of the LED via a 220 ohm resistor. The PICO\_GP28 pin is connected to the LED's anode. The simulation shows the LED turning on when motion is detected and turning off when no motion is detected.

4. Gebäudeautomation und Steuerung



## Lernziele

- **Informationsübertragung mittels Bussystemen** und verschiedenen Randbedingungen  
z.B. Echtzeitfähigkeit, Teilnehmerzahl, Störsicherheit
- Grundkonzepte der **Steuerungs- und Regelungstechnik** z.B. PID-Regler für Konstantlichtregelung

# Vernetzung in Gebäuden

SOFTWARE-AUSFALL

## Seit 1,5 Jahren brennt das Licht an einer Schule

Seit Sommer 2021 leuchten Tag und Nacht 7.000 [Lampen](#) in einer US-Schule.  
Auslöser dafür ist ein Software-Ausfall.



22. Januar 2023, 10:35 Uhr, Ingo Pakalski



(Bild: Minnechaug Regional High School)

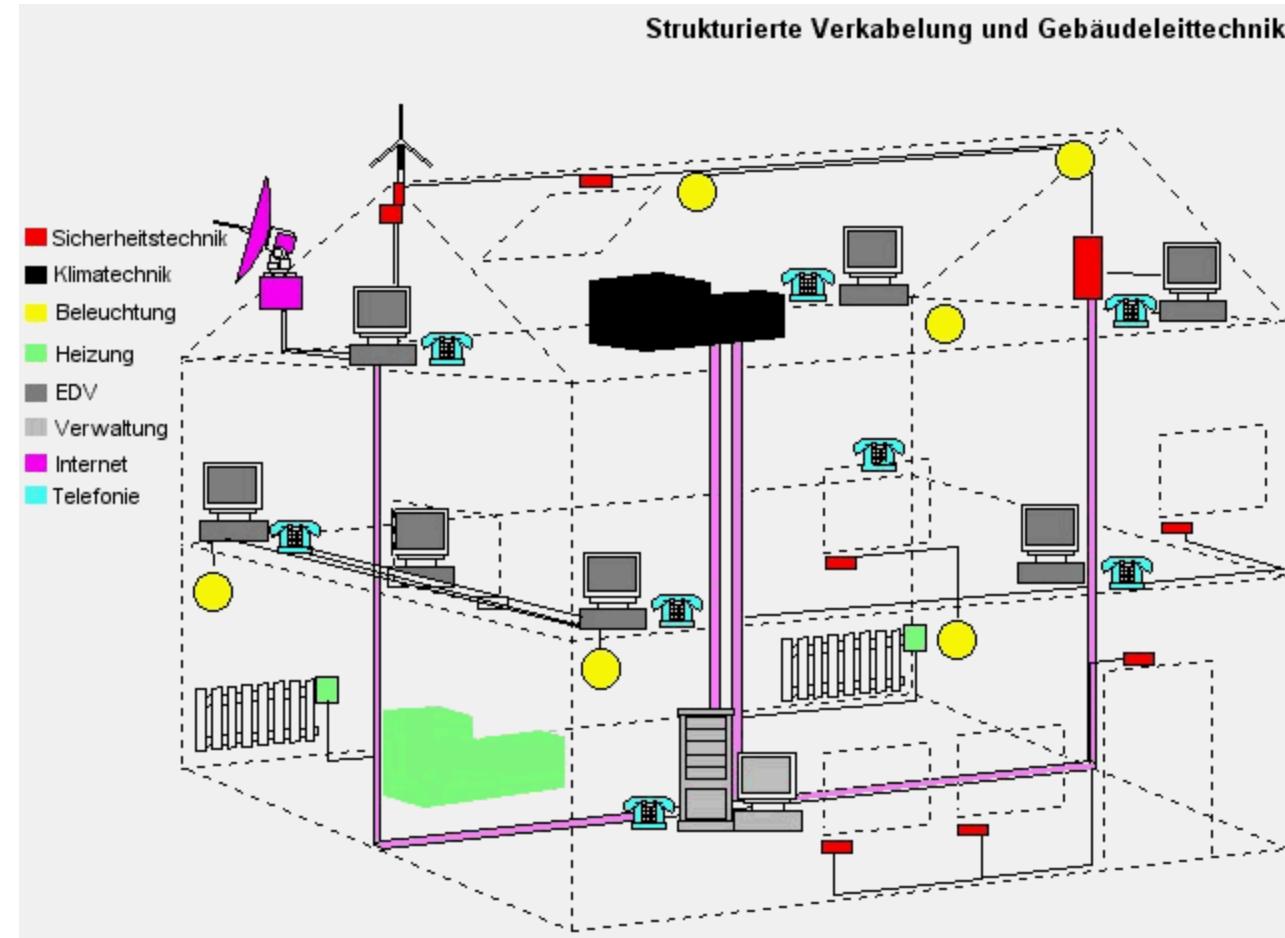
## 6. Instandhaltung und Betrieb



## Lernziele

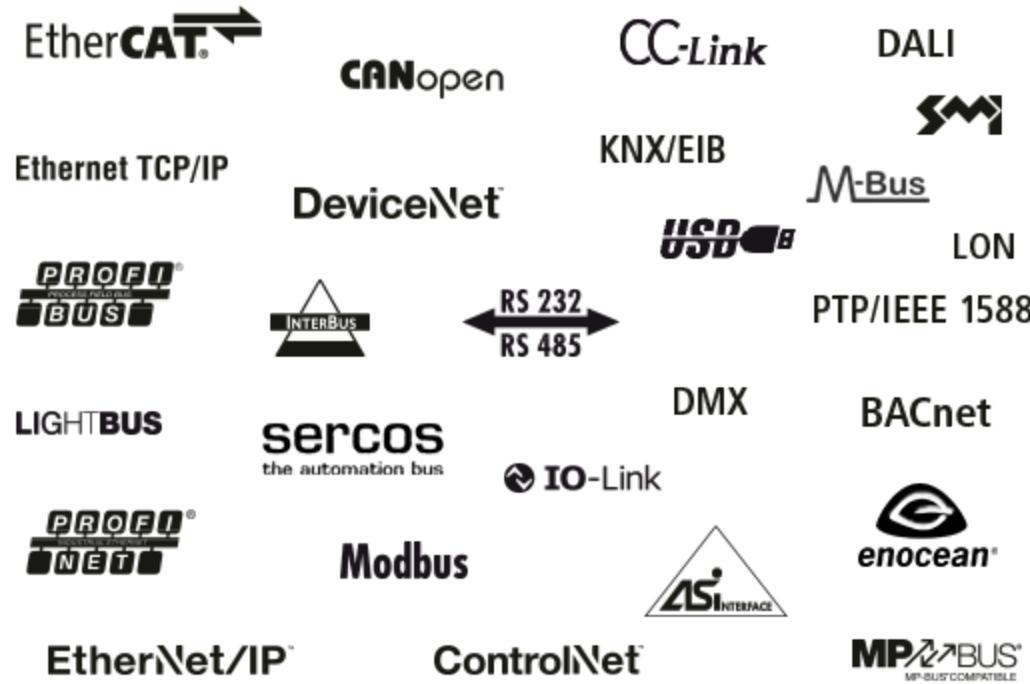
- **Netzwerkkonfiguration** für TCP/IP-Netzwerke
- **Sicherheitsaspekte** z.B. Firewall, VPN, Verschlüsselung
- **Datenspeicherung** und **Datensicherung** im Betrieb und bei Störungen

# Bussysteme als Zentrales Nervensystem

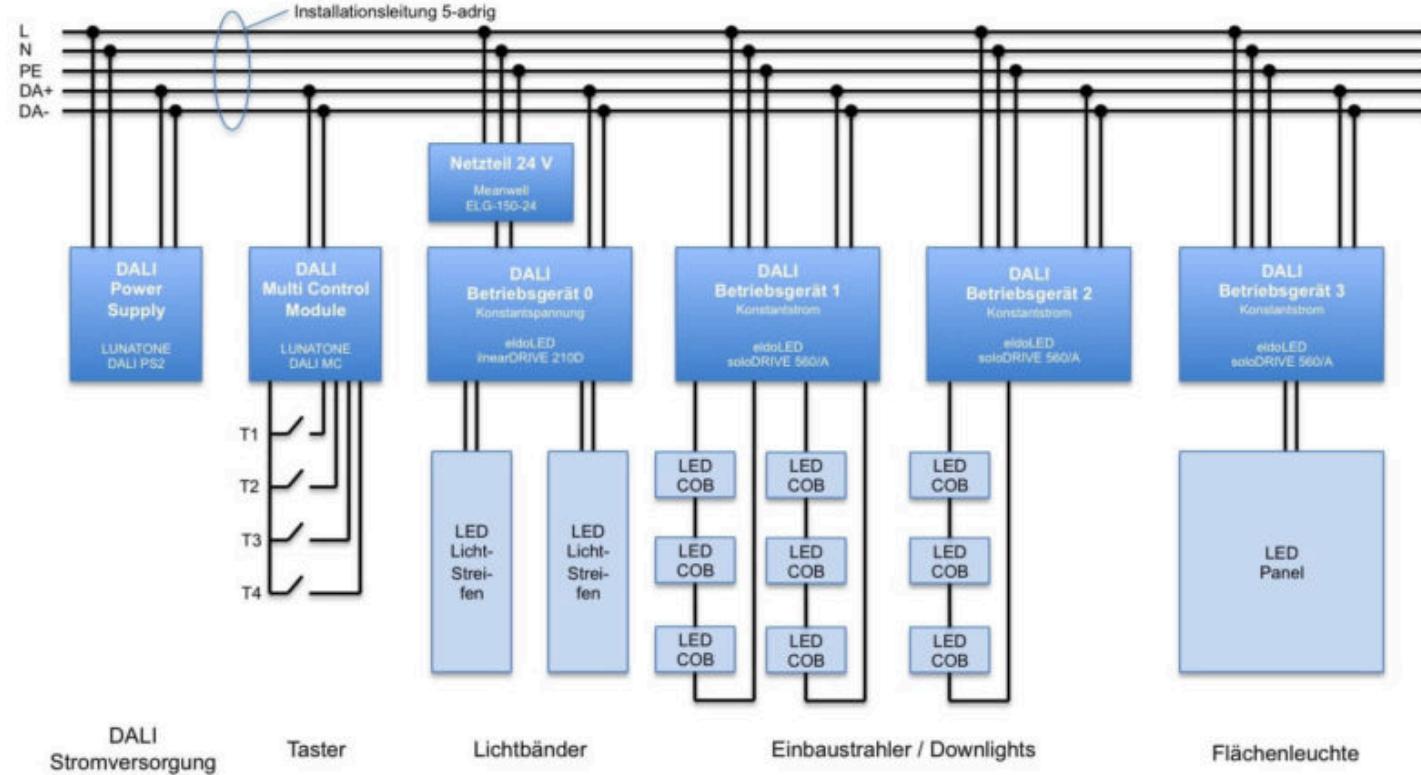


2. Klimatisierung und Raumkomfort

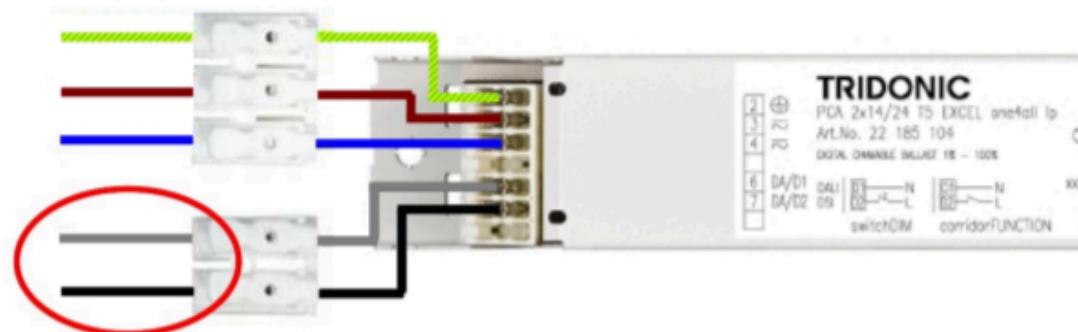
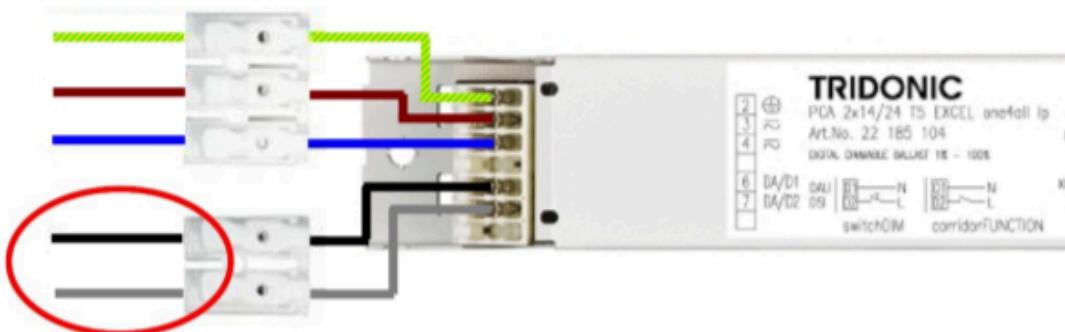
# Bussysteme für verschiedene Gewerke



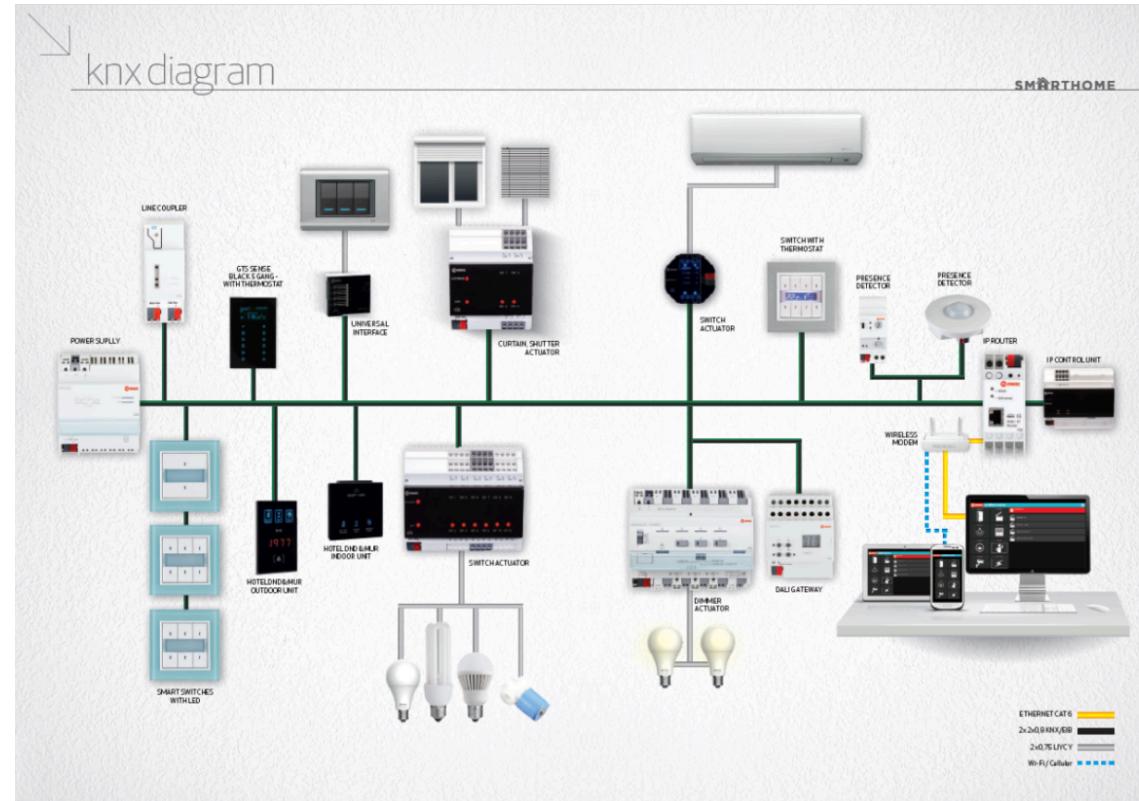
# 🎯 DALI - Intelligente Lichtsteuerung



- **Verkabelung und Addressierung** von bis zu 64 Leuchten und Zusammenfassen in Gruppen
- Konfiguration von **Szenen** für verschiedene Anforderungen



# 🎯 KNX & BACnet - Übergreifende Gebäudeautomation



- **Einsatzgebiete** und **Topologien** verschiedener Bus-Systeme
- Planung von **Raumautomations-Funktionen** mittels **Schemata**

3. Sicherheits- und Brandschutz

# Anlagenautomation



# Klassische Automatisierungstechnik



- Verdrahtung von **Aufbau von SPS-Sytemen**
- Programmierung mittels **Funktionsplänen** oder **Strukturiertem Text**

# Ethernet/IP

**Note**  
Support for these integrations is provided by the Home Assistant community.

All (2447) Search integrations...

**Featured**

Added in:

- 3D Printing (2)
- Alarm (36)
- Alarm Control Panel (12)
- Automation (26)
- Binary Sensor (196)
- Button (49)
- Calendar (10)
- Camera (49)

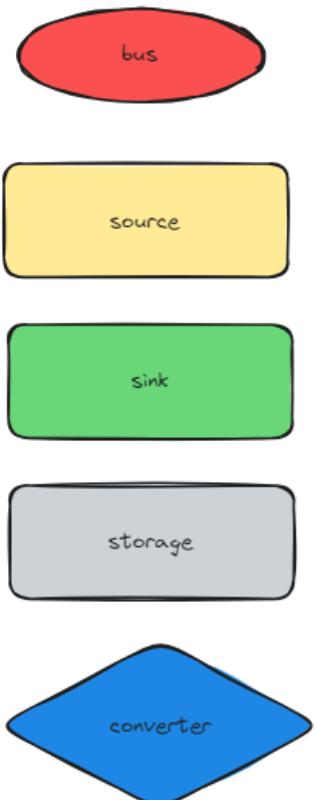
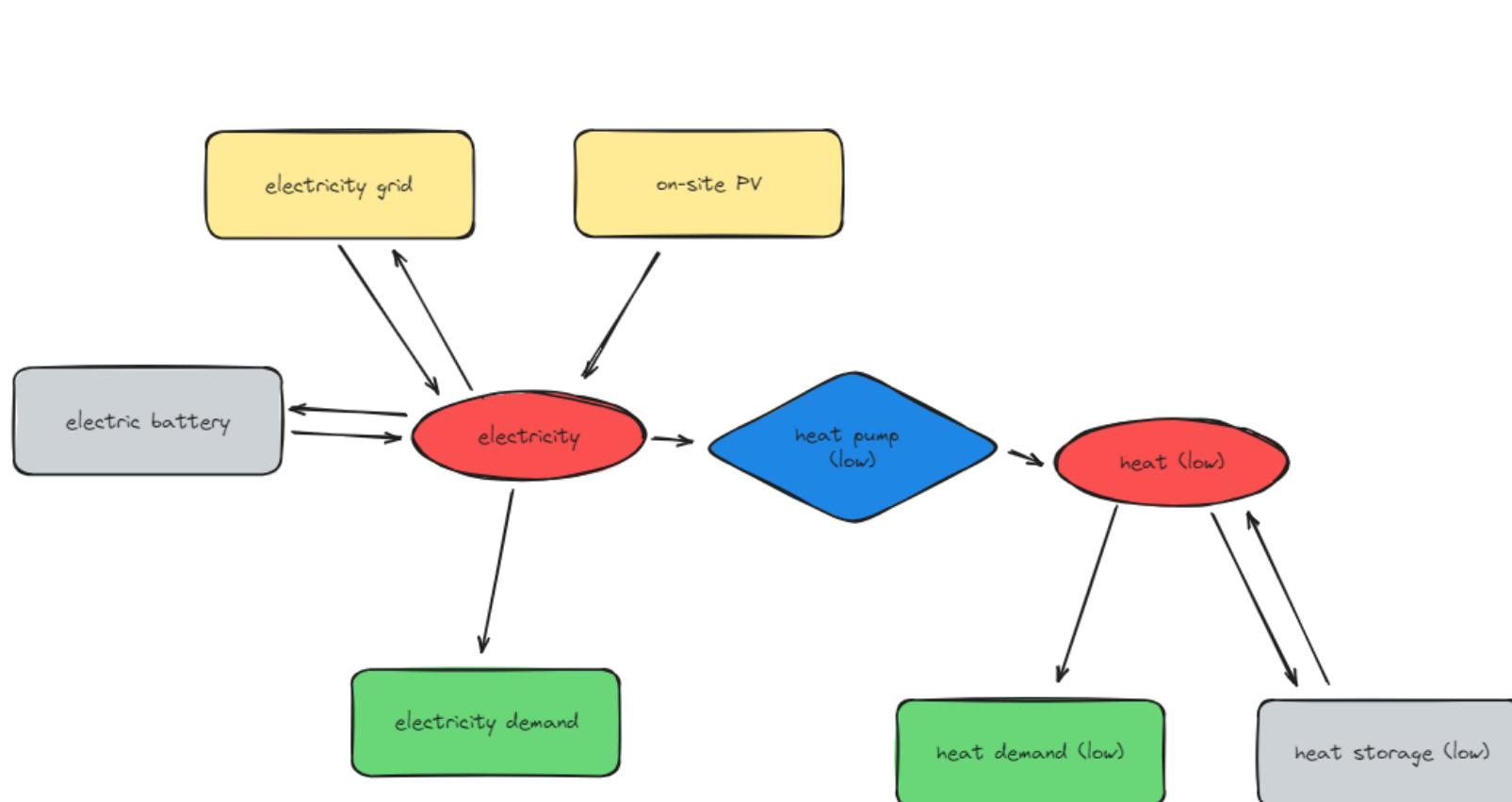
 amazon alexa Amazon Alexa	 ecobee ecobee	 ESPHome ESPHome	 Google Assistant Google Assistant
 Google Cast Google Cast	 HomeKit HomeKit	 IKEA TRÅDFRI IKEA TRÅDFRI	 LEVITON. Leviton Z-Wave
 Lutron Caséta Lutron Caséta	 MQTT MQTT	 hue personal wireless lighting Philips Hue	 plex Plex Media Server

- Grundtechnologien des **Internets**, z.B. **TCP/IP**, **JavaScript**, **HTML**
- Datenübertragung mittels **HTTP**, **MQTT**

8. Komfort und Nutzerfreundlichkeit

# Energiemanagement & Smart Metering

## Energy Management of a Single Family Home



1. Sicherstellung von Versorgung und Entsorgung

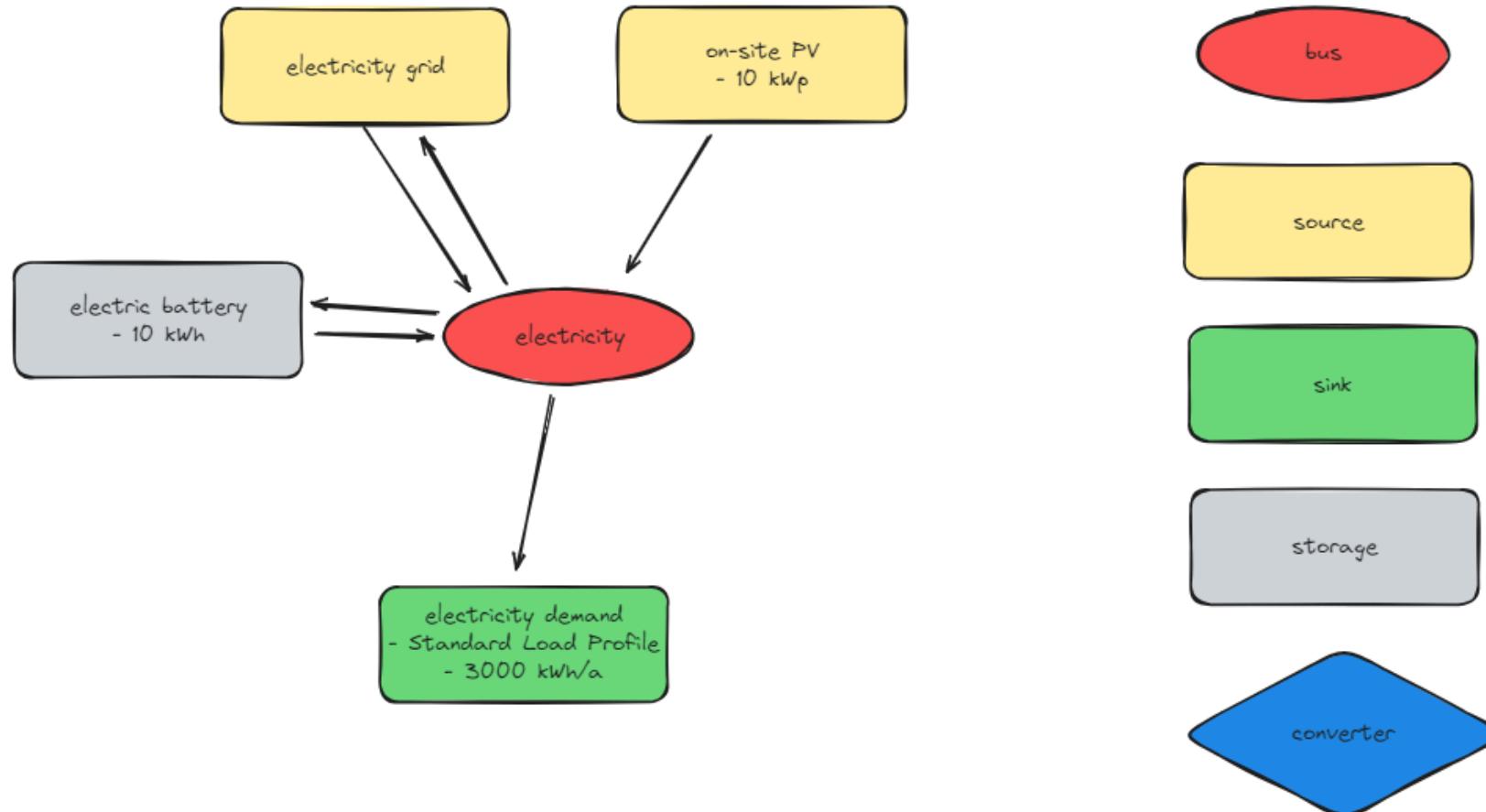
## Energiesysteme

- **Versorgung**, Erzeugung, Speicherung, Verteilung, Verbrauch
- **Skalen**: Gebäude, Stadt, Region, Land, Welt
- **Sektoren**: Strom, Wärme, Mobilität
- **Technologien**: Erneuerbare, Speicher, Wärme, Elektrolyse, ...
- **Zielkonflikte**: Klimaneutralität, Versorgungssicherheit, Wirtschaftlichkeit

## Energiemanagement

- **Planung**: Welche Technologien, Standorte, Größen
- **Betrieb**: Welche Steuerung, Regelung, Optimierung
- **Optimierung**: Potenziale aufdecken, Kosten minimieren

## Beispiel: Optimierung eines EFH mit PV und Speicher



# Lasten und Ertäge



# Speicherbewirtschaftung

□ × + - × ☰





## Dafür benötigt

- Zentrales System zur **Steuerung** und **Optimierung**
- **Smart Metering** für genaue Verbrauchsdaten und Abrechnung
- Schnittstelle oder Bus-Systeme für **Speicher-Management** und
- externe **Datenquellen** für Wetterdaten, Strompreise, ...

# Semester 1: Grundlagen Informationstechnologie & Datensicherheit

1. Rechnersysteme
2. Informationsdarstellung
3. Compiler und Algorithmen
4. Programmieren
5. Datenspeicherung
6. Kommunikation
7. IT-Sicherheit und Datenschutz

## Semester 2: Bussysteme

1. Gebäudeautomation und Planung
2. Messkette und Computer-Systeme
3. Steuerung- und Regelungstechnik
4. Speicherprogrammierbare Steuerung
5. Bussysteme
6. Smart Metering

## Semester 2: Labor Bussysteme

