MicroPython-Programmierung am Raspberry Pi Pico

Vorbereitung:

Öffne die Homepage Wokwi und wähle Micropython on Pi Pico (Auszug aus Homepage, s. unten)

Code schreiben

Schritt1: Um zu testen, ob dein Pico richtig funktioniert, schreiben wir eine einfache Funktion, die "Hello World" sagt. Drücken Sie anschließend auf Kompilieren, um den Code auszuführen. Sie sehen nun eine Aktualisierung der Python-Shell (auch bekannt als REPL, Read, Eval, Print, Loop), um Ihnen mitzuteilen, dass Ihr Pico bereit ist und arbeitet.

```
print("Hallo Welt")
```

Schritt 2: Importieren Sie die erforderlichen Bibliotheken: Wir werden unseren Code auf dem leeren Platz auf der linken Seite schreiben. Für den Anfang importieren wir zwei MicroPython-Bibliotheken - Pin class und utime. Die Pin-Klasse können Sie aus der Maschinenbibliothek beziehen und dient zur Steuerung von I/O-Pins (auch bekannt als GPIO - general-purpose input/output). Utime wird hauptsächlich dazu verwendet, die Geschwindigkeit unseres Codes zu kontrollieren.

```
from machine import Pin

import utime
```

Schritt 3: Verbindung des Codes mit der physischen Welt: Fügen Sie ein Objekt mit dem Namen "led" hinzu, um den GPIO-Pin mit unserem Code zu verbinden. Hier wird GPIO 25 (Pin 34 auf dem physischen Board) als Ausgangspin verwendet, wobei der Strom nun vom GPIO des Raspberry Pi Pico zur Onboard-LED fließt. Verwenden wir unser Objekt, um den GPIO-Pin auf "pull low" zu setzen. Damit stellen wir sicher, dass unser GPIO-Pin zu Beginn des Projekts ausgeschaltet ist.

```
led = Pin(25, Pin.OUT)

led.low()
```

Schritt 4: Zeit, dass unsere LED blinken: Schreiben Sie eine while-true-Schleife, eine Schleife, die niemals endet, um unsere LED ein- und auszuschalten. Um eine Verzögerung von einer Sekunde zwischen den einzelnen Ein- und Ausschaltvorgängen zu erzeugen, fügen wir dem Code für jeden Zyklus der Schleife eine Sekunde „sleep“ hinzu.

Wichtig: Beachten Sie beim Code die Einrückungen, diese ersetzen die Klammern und Semikolons.

```
while True:
    led.toggle()
    print("Toggle")
    utime.sleep(1)
```

Komplett sieht der Code folgendermaßen aus:

```
from machine import Pin
import utime
led = Pin(25, Pin.OUT)
led.low()
while True:
    led.toggle()
    print("Toggling LED")
    utime.sleep(1)
```

Schritt 5: Wenn Sie Ihren Code ausführen, klicken Sie auf die grüne Schaltfläche "Play" und Sie sehen jede Sekunde ein „Toggle“, was bedeutet, dass Ihre LED jede Sekunde ein- und ausgeschaltet wird.

Aufgabe

Programmieren Sie die Ampelschaltung, welche Sie bereits in Arduino programmiert haben, in der Programmiersprache MicroPython mit einem Raspberry Pi Pico!

> ***Info***
> *Warum nicht gleich in Python programmieren? Was hat es mit MicoPython auf sich? Informationen dazu: [Micro Python: Ein Python für Microcontroller | heise online]*

Anhang

Befehle als Beispiel

Bezeichnung	**Befehl**
Output	`Pin(3,Pin.OUT)`
Input	`Pin(4,Pin.IN)`
Wertabfrage (von Variable a)	`a.value()`
high-Signal (von Variable a)	`a.high()`
low-Signal (von Variable a)	`a.low()`
Verzögerung	`utime.sleep(2)`

If-else-Condition	`temperature = 15`
	`target = 10`
	`if temperature > target:`
	`  print("Too High!")`
	`elif temperature < target:`
	`  print("Too Low!")`
	`else:`
	`  print("Just right!")`

For-Schleife	`x = 0`
	`for y in range(0, 9):`
	`  x += 1`
	`print(x)`

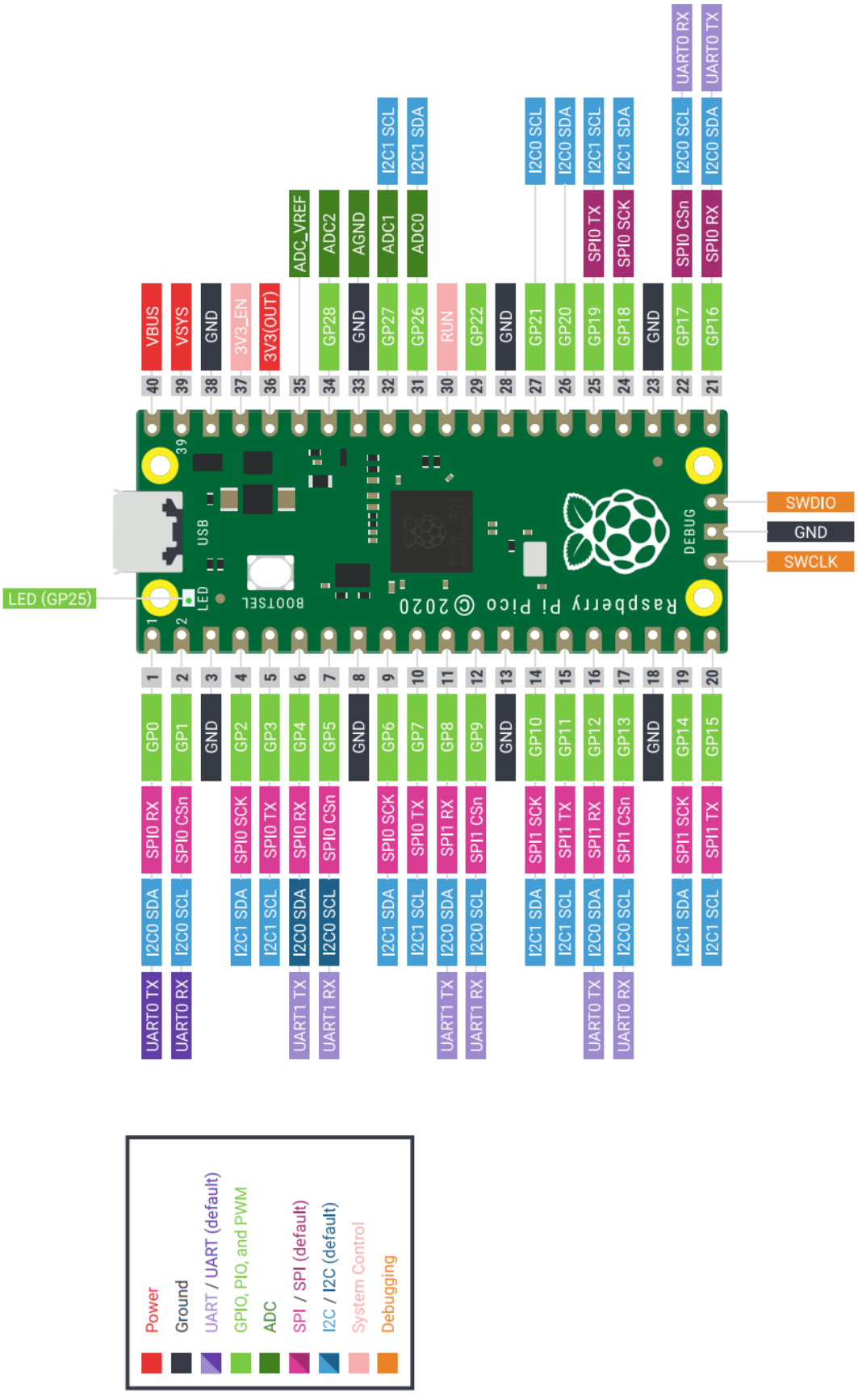
While-Schleife	`x = 0`
	`while x < 9:`
	`  x += 1`
	`print(x)`

Funktion mit Zahlen	`def add(number1, number2):`
	`  return number1 + number2`
	
	`add(1, 2) # expect a result of 3`

Funktion mit String	`def welcome(name):`
	`  welcome_phrase = "Hello, " + name + "!"`
	`  print(welcome_phrase)`
	
	`welcome("Alex") # expect "Hello, Alex!"`

Liste	`networks = ['lora', 'sigfox', 'wifi', 'bluetooth', 'lte-m']` `print(networks[2]) # expect 'wifi'`
Wörterbuch	`address_book = {'Alex':'2604 Crosswind Drive','Joe':'1301 Hillview Drive','Chris':'3236 Goldleaf Lane'}` `print(address_book['Alex']) # expect '2604 Crosswind Drive'`
Tupel	`pycom_devices = ('wipy', 'lopy', 'sipy', 'gpy', 'fipy')` `print(pycom_devices[0]) # expect 'wipy'`
Bibliothek „B“ einfügen	`import B`

GPIO-Belegung Raspberry PI Pico



Quelle: raspberrypi.com