ADCs kurz und bündig

ADCs sind Analog-Digital-Converter, d.h. sie wandeln ein analoges Signal (eine elektrische Spannung zwischen 0 und 3.3 Volt) in einen diskreten ‚digitalen' Zahlenwert von 0 bis 65535 (= 16 Bit) um. 0 stellt dabei das minimal messbare Signal dar, 65535 das maximal messbare. Der Raspberry Pi Pico hat drei GPIOs, die gleichzeitig als ADC fungieren: GPIO26, GPIO27 und GPIO28 (ADC0 auf Pin 31, ADC1 auf 32 und ADC2 auf 34). Der GND auf Pin 33 dient auch als GND-Anschluss für die ADCs.

ADCs auslesen

Wie immer muss zuerst die Bibliothek des Pico geladen werden, um die Hardware direkt ansteuern zu können:

```
import machine
```

Lädt die notwendige Bibliothek und sollte am Beginn des Programms stehen.

Nun muss der ADC als Objekt initialisiert und einer Variablen zugewiesen werden:

```
adc0 = machine.ADC(26)
```

Weist der Variablen **adc0** den **GPIO26** (ADC0) als **Input** zu.

```
adc0.read_u16()   # liest den ADC-Wert als 16 Bit Ganzzahl
```

Beispiele für analoge Eingaben

Analoge Werte können aus vielen Quellen stammen. Typische Anwendungsbeispiele sind:

- Potentiometer – Drehregler mit einem Minimum und Maximumwert, dessen Stellung anhand des zurückgelieferten Wertes ausgelesen werden kann, und zum Beispiel als Regler oder zur Erfassung einer Achsausrichtung verwendet werden.
- Temperatursensoren (Thermistor), die den Widerstand und damit die Spannung in Abhängigkeit der Temperatur verändern.
- Helligkeitssensoren (Photoresistor), die den Widerstand und damit die Spannung in Abhängigkeit der Helligkeit verändern.

Beispiel: Temperatur messen

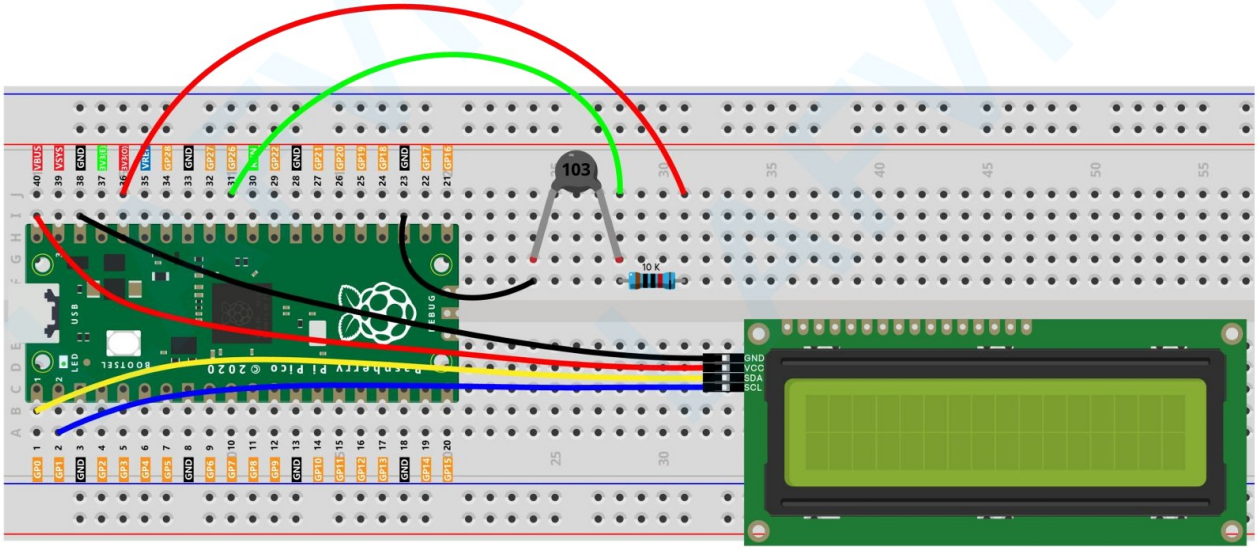

Bild 1: Schaltungsübersicht für Temperatur messen - um LCD erweitert (Bonusaufgabe

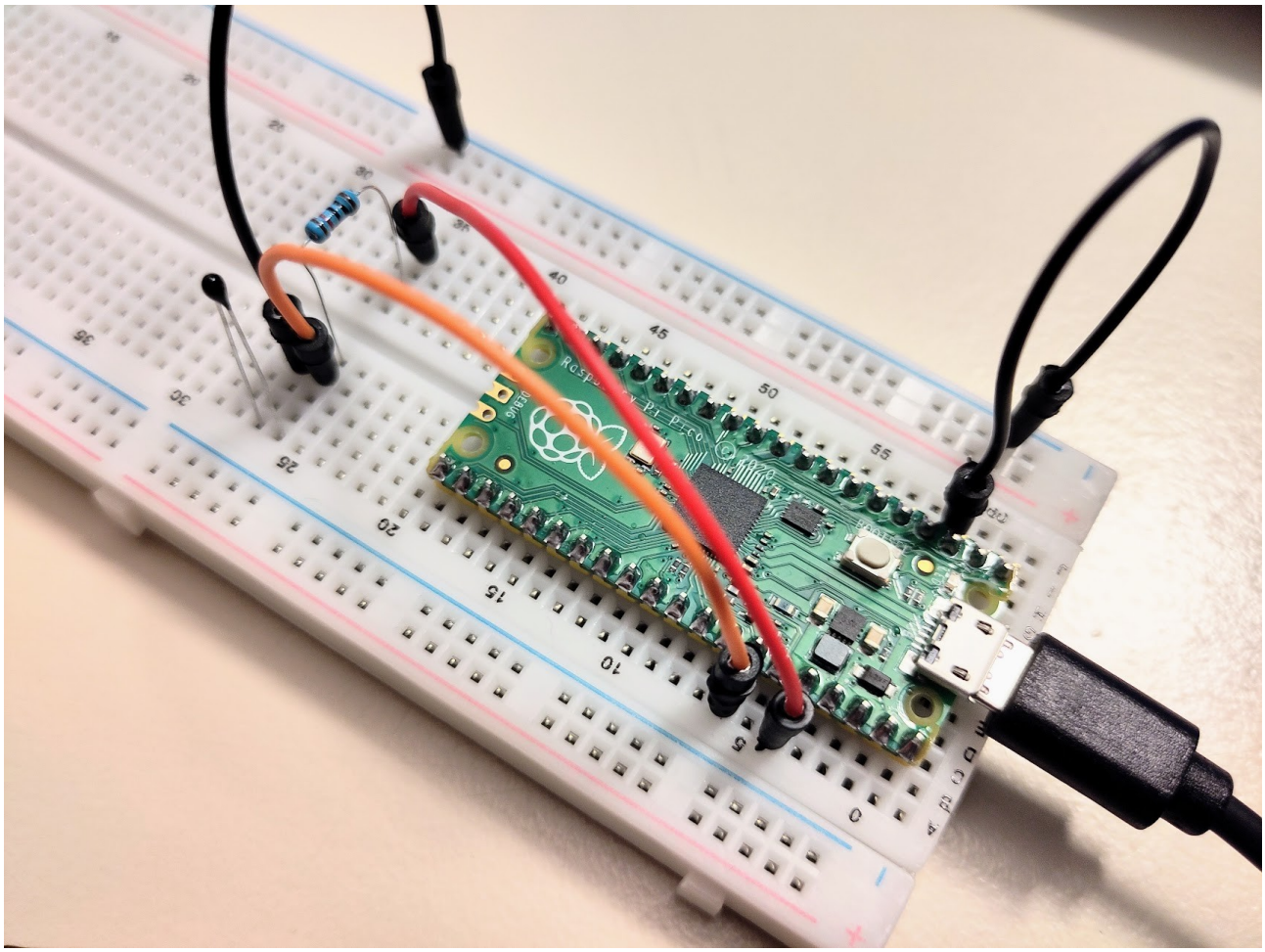

Bild 2: Schaltung auf Breadboard für Beispiel 1 (Temperatur-Messung)

Programmcode

```
import machine # Pico-Funktionen
import time    # Zeit-Funktionen
import math    # Mathe-Funktionen

# Pins initialisieren
adcTemperatur = machine.ADC(26)
ledOnBoard = machine.Pin(25, machine.Pin.OUT)

# Funktion, welche den ADC abfragt und als Temperatur in °C
# mit return zurueckgibt! Formel aus StarterKit-Tutorial
def getTemperatur():
    value = adcTemperatur.read_u16() # ADC lesen
    volt = value / 65535.0 * 3.3     # gemessene Spannung berechnen
    # Widerstand berechnen
    Rt = 10 * volt / (3.3 - volt)
    # Daraus die Temperatur in °K berechnen
    tempK = (1 / (1 / (273.15 + 25) + (math.log(Rt / 10)) / 3950))
    # In ganzzahlige Temperatur in °C umrechnen
    tempC = tempK - 273.15
    # Als formatierte Textzeile in Konsole ausgeben
    print("ADC: {}, V: {:.2f}, °C: {:.2f}".format(value, volt, tempC))
    return tempC

while True:
    # Temperatur mit obiger Funktion abfragen und ausgeben
    temp = getTemperatur()
    # onBoard LED toggeln als Betriebsanzeige
    ledOnBoard.toggle()
    # halbe Sekunde schlafen
    time.sleep(0.5)
```

LCDs ansteuern

Da der Pico nicht zwingend an einen Computer angeschlossen ist, um dessen Konsole zu nutzen und Informationen auszugeben (der `print()`-Befehl), gibt es die Möglichkeit direkt ein Display an den Raspi anzuschließen. Die einfachste Möglichkeit hierfür ist ein LCD-Display. Dem StarteKit liegt ein LCD mit Hintergrundbeleuchtung und zwei Zeilen à 16 Zeichen bei.

Damit das Display genutzt werden kann, muss eine zusätzliche Codedatei auf den Pico kopiert werden (`LCD1602.py`). Dies kann einfach mit Thonny geschehen. Diese enthält den notwendigen Code um Texte auf dem LCD anzuzeigen und wird ähnlich wie die bisherigen Funktionsbibliotheken einfach importiert.

Beispiel Countdown

Zählt einen Countdown runter und zeigt die verbleibende Zeit und eine Abschlussmeldung auf dem LCD an.

Programmcode

```
from LCD1602 import LCD # import aus beigefuegter Datei!
import time

# LCD initialisieren
lcd = LCD()

# Beleuchtung LCD aktivieren
while True:
    # von 10 bis null herunterzaehlen
    for value in range(10):
        countdown = 10 - value
        # Countdown auf LCD ausgeben
        lcd.message("{}...".format(countdown))
        time.sleep(1)
        # LCD bereinigen
        lcd.clear()

    # Countdown vorbei, BOOM! ausgeben
    lcd.message("BOOOOOOOOM!\nBOOOOOOOOM!")
    time.sleep(1)
    # LCD vor naechster Runde bereinigen
    lcd.clear()
```

Beispiel Temperaturmessen und per LCD anzeigen

Führt die beiden vorhergehenden Beispielprojekte zu einem zusammen. Jetzt zeigt der LCD die zuvor gemessene Temperatur an.

Programmcode

```
import machine # Pico-Funktionen
import time    # Zeit-Funktionen
import math    # Mathe-Funktionen
from LCD1602 import LCD # import aus beigefuegter Datei!

# LCD initialisieren
lcd = LCD()

# Pins initialisieren
adcTemperatur = machine.ADC(26)
ledOnBoard = machine.Pin(25, machine.Pin.OUT)

# Funktion, welche den ADC abfragt und als Temperatur in °C
# mit return zurueckgibt! Formel aus StarterKit-Tutorial
def getTemperatur():
    value = adcTemperatur.read_u16() # ADC lesen
    volt = value / 65535.0 * 3.3     # gemessene Spannung berechnen
    # Widerstand berechnen
    Rt = 10 * volt / (3.3 - volt)
    # Daraus die Temperatur in °K berechnen
    tempK = (1 / (1 / (273.15 + 25) + (math.log(Rt / 10)) / 3950))
    # In ganzzahlige Temperatur in °C umrechnen
    tempC = tempK - 273.15
    # Als formatierte Textzeile in Konsole ausgeben
    print("ADC: {}, V: {:.2f}, °C: {:.2f}".format(value, volt, tempC))
    return tempC

while True:
    # LCD bereinigen
    lcd.clear()
    # Temperatur mit obiger Funktion abfragen
    temp = getTemperatur()
    # Text auf LCD ausgeben
    lcd.message("Temperatur:\n{:.2f} Celsius".format(temp))
    # onBoard LED toggeln als Betriebsanzeige
    ledOnBoard.toggle()
    # halbe Sekunde schlafen
    time.sleep(0.5)
```