Mikrokontroller & I²C Bus

Raspberry Pi Pico
7 – Segment Anzeige
mit TM1637

Copyright

Sicherheitshinweise

Lesen Sie diese Gebrauchsanleitung, bevor Sie diesen Bausatz in Betrieb nehmen und bewahren Sie diese an einem für alle Benutzer jederzeit zugänglichen Platz auf. Bei Schäden, die durch Nichtbeachtung dieser Bedienungsanleitung verursacht werden, erlischt die Gewährleistung / Garantie. Für Folgeschäden übernehmen wir keine Haftung! Bei allen Geräten, die zu ihrem Betrieb eine elektrische Spannung benötigen, müssen die gültigen VDE-Vorschriften beachtet werden. Besonders relevant sind für diesen Bausatz die VDE-Richtlinien VDE 0100, VDE 0550/0551, VDE 0700, VDE 0711 und VDE 0860. Bitte beachten Sie auch nachfolgende Sicherheitshinweise:

- Nehmen Sie diesen Bausatz nur dann in Betrieb, wenn er zuvor berührungssicher in ein Gehäuse eingebaut wurde. Erst danach darf dieser an eine Spannungsversorgung angeschlossen werden.
- Lassen Sie Geräte, die mit einer Versorgungsspannung größer als 24 V- betrieben werden, nur durch eine fachkundige Person anschließen.
- In Schulen, Ausbildungseinrichtungen, Hobby- und Selbsthilfewerkstätten ist das Betreiben dieser Baugruppe durch geschultes Personal verantwortlich zu überwachen.
- In einer Umgebung in der brennbare Gase, Dämpfe oder Stäube vorhanden sind oder vorhanden sein können, darf diese Baugruppe nicht betrieben werden.
- Im Falle eine Reparatur dieser Baugruppe, dürfen nur Original-Ersatzteile verwendet werden! Die Verwendung abweichender Ersatzteile kann zu ernsthaften Sach- und Personenschäden führen. Eine Reparatur des Gerätes darf nur von fachkundigen Personen durchgeführt werden.
- Spannungsführende Teile an dieser Baugruppe dürfen nur dann berührt werden (gilt auch für Werkzeuge, Messinstrumente o.ä.), wenn sichergestellt ist, dass die Baugruppe von der Versorgungsspannung getrennt wurde und elektrische Ladungen, die in den in der Baugruppe befindlichen Bauteilen gespeichert sind, vorher entladen wurden.
- Sind Messungen bei geöffnetem Gehäuse unumgänglich, muss ein Trenntrafo zur Spannungsversorgung verwendet werden
- Spannungsführende Kabel oder Leitungen, mit denen die Baugruppe verbunden ist, müssen immer auf Isolationsfehler oder Bruchstellen kontrolliert werden. Bei einem Fehler muss das Gerät unverzüglich ausser Betrieb genommen werden, bis die defekte Leitung ausgewechselt worden ist.
- Es ist auf die genaue Einhaltung der genannten Kenndaten der Baugruppe und der in der Baugruppe verwendeten Bauteile zu achten. Gehen diese aus der beiliegenden Beschreibung nicht hervor, so ist eine fachkundige Person hinzuzuziehen

Bestimmungsgemäße Verwendung

- Auf keinen Fall darf 230 V~ Netzspannung angeschlossen werden. Es besteht dann Lebensgefahr!
- Dieser Bausatz ist nur zum Einsatz unter Lern- und Laborbedingungen konzipiert worden. Er ist nicht geeignet, reale Steuerungsaufgaben jeglicher Art zu übernehmen. Ein anderer Einsatz als angegeben ist nicht zulässig!
- Der Bausatz ist nur für den Gebrauch in trockenen und sauberen Räumen bestimmt.
- Wird dieser Bausatz nicht bestimmungsgemäß eingesetzt kann er beschädigt werden, was mit Gefahren, wie z.B. Kurzschluss, Brand, elektrischer Schlag etc. verbunden ist. Der Bausatz darf nicht geändert bzw. umgebaut werden!
- Für alle Personen- und Sachschäden, die aus nicht bestimmungsgemäßer Verwendung entstehen, ist nicht der Hersteller, sondern der Betreiber verantwortlich. Bitte beachten Sie, dass Bedien- und /oder Anschlussfehler außerhalb unseres Einflussbereiches liegen. Verständlicherweise können wir für Schäden, die daraus entstehen, keinerlei Haftung übernehmen.
- Der Autor dieses Tutorials übernimmt keine Haftung für Schäden. Die Nutzung der Hard- und Software erfolgt auf eigenes Risiko.

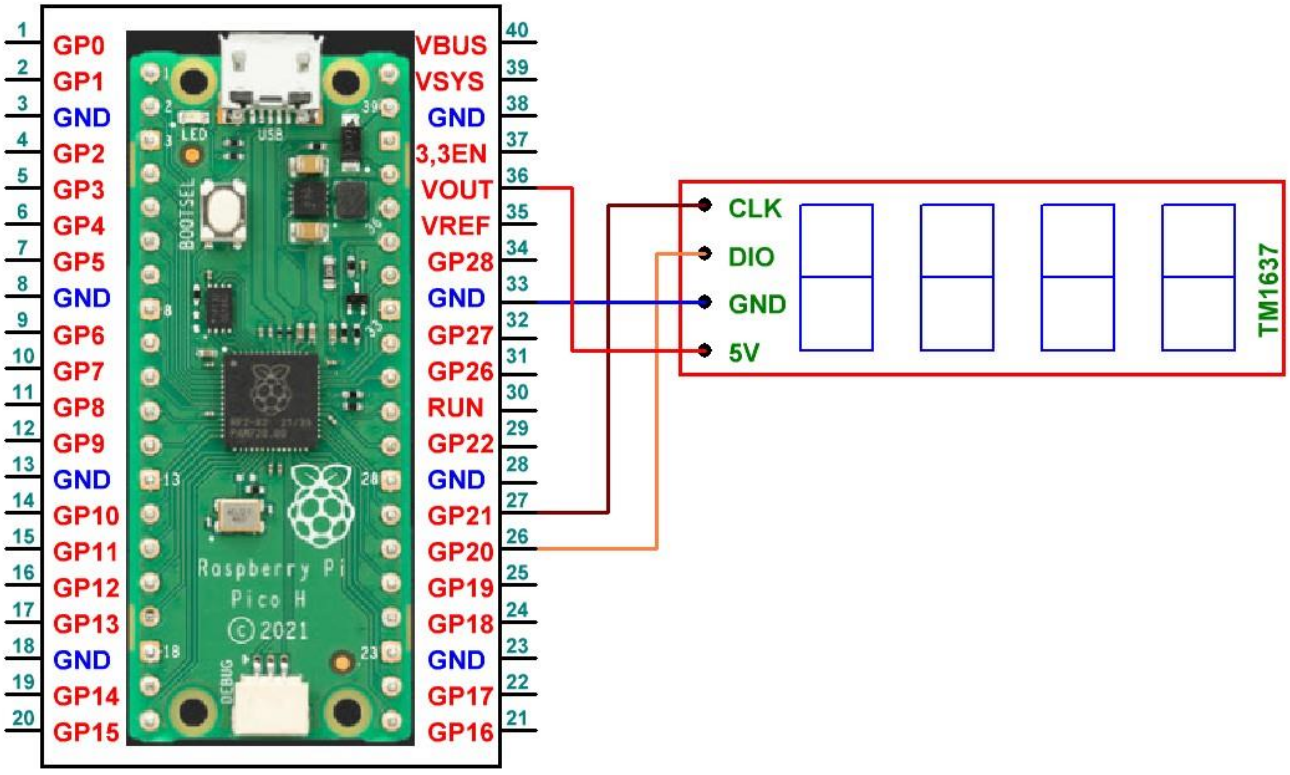
Raspberry Pi Pico – 7-Segment Anzeige mit TM1637

Neben dem HT16K33 gibt es noch andere ICs zur Anzeige einer 7-Segment Anzeige. Ein sehr beliebter IC ist der TM1637. In der Literatur und im Internet wird angegeben, dass er mit dem I^2C Bus angesteuert wird. Diese Angabe ist nicht korrekt, obwohl man die Bezeichnungen auf dem Modul so interpretieren könnte. Leider ist das Datenblatt nur auf Chinesisch bzw. Englisch zu finden. Wie weit die englische Übersetzung korrekt ist kann ich nicht beurteilen. Der TM1637 wird direkt von den Ports des Pico angesteuert.

Ansicht einer 7-Segment Anzeige mit TM1637.
Die Belegung der Pins ist auf der Rückseite.

Je nach Hersteller unterscheiden sich die Grösse und Aufbau der einzelnen Module.

Die Montage des Pico und der Platine mit dem TM1637 erfolgte auf einem Breadboard.



Belegung / Verbindung Pico zum TM1637

Pico	Platine mit TM1637
Pin 36 (V OUT 3,3V)	- 5V
Pin 33 (GND)	- GND
Pin 21 (GP21)	- CLK
Pin 20 (GP20)	- DIO

MicroPython-Bibliothek für TM1637

Zur Ansteuerung der 7-Segment-Anzeige TM1637 ist eine externe Bibliothek erforderlich, die heruntergeladen und mit dem Dateinamen „tm1637.py" auf dem Raspberry Pi Pico gespeichert werden muss.

Programmcode

Im Programmcode werden die Bibliotheken geladen und dann die 4-fache 7-Segment-Anzeige TM1637 initialisiert. Anschließend werden verschiedene sinnvolle Möglichkeiten der Darstellung beispielhaft realisiert:

- 7-Segment-Test: "88:88" (alle Segmente leuchten)
- Helligkeit einstellen (von dunkler nach heller)
- Anzeige löschen
- Ganzzahlen anzeigen / Wertebereich: -999 bis 9999
- Darstellung bei Überschreitung des Wertebereichs
- Verschiedene Texte anzeigen (ohne Unterscheidung von Groß- und Kleinschreibung)
- Laufschrift: Hallo Welt
- Aktuelle Uhrzeit
- Temperatur mit der Einheit °C
- Countdown von 9999 bis 0
- Countdown von 01:00 bis 00:00

Damit die Anzeige richtig interpretiert werden kann, wird zusätzlich auf der Kommandozeile angezeigt, was der Wert auf der Anzeige sein soll.
Die Variable „pause" gibt in Sekunden an, wie lange zwischen den unterschiedlichen Darstellungen gewartet werden soll.

```
# Bibliotheken laden
from machine import Pin, RTC, ADC, Timer
import utime as time
import tm1637

# Display initialisieren
display = tm1637.TM1637(clk=Pin(21), dio=Pin(20))

# Pause zwischen den unterschiedlichen Darstellungen
pause = 3
print('7-Segment-Test: "88:88" (alle Segmente leuchten)')
display.write([127, 255, 127, 127])

# Pause
time.sleep(pause)
print('Helligkeit einstellen (von dunkler nach heller)')
for i in range (0, 7+1):
    display.brightness(i)
    time.sleep(0.5)

# Pause
time.sleep(pause)
print('Anzeige löchen')
display.write([0, 0, 0, 0])
```

```
# Pause
time.sleep(pause)
print('Ganzzahlen anzeigen / Wertebereich: -999 bis 9999')
display.number(1)
time.sleep(1)
display.number(12)
time.sleep(1)
display.number(123)
time.sleep(1)
display.number(1234)
time.sleep(1)
print('Überschreitung des Wertebereichs')
display.number(12345)
time.sleep(2)
display.number(-1234)
time.sleep(2)
display.number(-123)
time.sleep(1)
display.number(-12)
time.sleep(1)
display.number(-1)

# Pause
time.sleep(pause)
print('Text anzeigen')
display.show('help')
time.sleep(2)
display.show('cool')
time.sleep(2)
display.show('  ja')
time.sleep(2)
display.show('nein')
time.sleep(2)
display.show('    ')

# Pause
time.sleep(pause)
print('Laufschrift: Hallo Welt') # Default: 4 FPS = 250
display.scroll('Hallo Welt', 250)

# Pause
time.sleep(pause)

# Echtzeituhr im Mikrocontroller initialisieren
rtc = RTC()

# Datum und Uhrzeit lesen
datetime = rtc.datetime()
print('Aktuelle Uhrzeit: %02d:%02d' % (datetime[4], datetime[5]))
display.numbers(datetime[4], datetime[5])
```

```
# Pause
time.sleep(pause)

# Initialisierung des ADC4
sensor = ADC(4)

# Temperatur messen
value = sensor.read_u16()
temp = int(27 - ( (value * 3.3 / 65535) - 0.706 ) / 0.001721)
print('Temperatur: %s°C' % temp) # Wertebereich -9 bis 99
display.temperature(temp)

# Pause
time.sleep(pause)
print('Countdown von 9999 bis 0')
for count in range (9999, -1, -1):
    display.number(count)
    time.sleep(0.00001)

# Pause
time.sleep(pause)
print('Countdown von 01:00 bis 00:00')
counter = 60
points = 1
timeValue = time.localtime(counter)
display.numbers(timeValue[4], timeValue[5], points)
time.sleep(1)

# Funktion: Countdown herunterzählen
def countdown(value):
    global counter
    global points
    # Blinkender Doppelpunkt im Wechsel
    if points == 0: points = 1
    else: points = 0
    # Countdown reduzieren
    counter -= 1
    # Countdown ist abgelaufen (0 = Ende)
    if counter == 0:
        clock.deinit()
        print('Countdown Ende')
    # Countdown in Minuten und Sekunden umrechnen und anzeigen
    timeValue = time.localtime(counter)
    display.numbers(timeValue[4], timeValue[5], points)

# Initialisierung Timer für Countdown
clock = Timer(freq=1, mode=Timer.PERIODIC, callback=countdown)
```

Im Programmcode sind die wichtigsten Methoden zur Darstellung von Werten beispielhaft verwendet. Es gibt aber noch weitaus mehr, die vielleicht nicht so häufig verwendet werden.

Einige Teile des Textes wurden zur besseren Übersicht farblich gestaltet.
Die Nutzung erfolgt auf eigenes Risiko.
Ich wünsche viel Spaß beim Bauen und programmieren
Achim

myroboter@web.de

Quellenangabe:
https://www.elektronik-kompendium.de/sites/raspberry-pi/2711111.htm

Download für tm1637.py und Beispiele:
https://github.com/mcauser/micropython-tm1637/blob/master/tm1637.py