# Mikrokontroller & I²C Bus

by AS

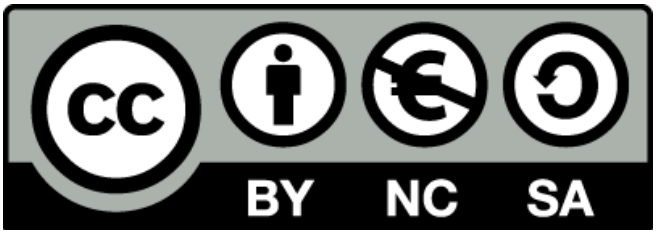
www.makerconnect.de

https://www.makerconnect.de/resource

## Raspberry Pi Pico
## I²C Bus mit HT16K33
## 8x8 Matrix Anzeige

## 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 – I²C Bus und dem HT16K33 8 x 8 LED Matrix Anzeige

In diesem Tutorial möchte ich euch die Anwendung mit dem HT16K33 und einer 8x8 LED Matrix vorstellen. Die notwendige Hardware habe ich bereits in einem anderen Tutorial beschrieben. Die Anwendung mit einer 7-Segment Anzeige habe ich bereits vorgestellt. Da für beide Anwendungen der HT16K33 verwendet wird, ist die Software sehr ähnlich.

Wie üblich müssen wir als erstes verschiedene Einstellungen vornehmen, in diesem Fall sagt man besser Register gesetzt werden.
Wenn die Anzeigeplatine eingeschaltet wird, ist die Anzeige dunkel. Es müssen drei Konfigurationswerte in den Chip geschrieben werden um ihn zum Laufen zu bringen.

```
i2c_Bus.writeto(I2C_Addr_1, bytes([0x21]) )   # Einstellung Oszillator
i2c_Bus.writeto(I2C_Addr_1, bytes([0xEF]) )   # Einstellung Helligkeit
i2c_Bus.writeto(I2C_Addr_1, bytes([0x81]) )   # Einstellung Blinken aus, Display ein
```

Einstellung Oszillator
Ein/aus
0x21 – 0 b 0010 0001
(Oscillator ein)

| Name | Command / Address / Data | | | | | | | | Option | Description | Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | | | |
| System set | 0 | 0 | 1 | 0 | X | X | X | S | {S} Write only | Defines internal system oscillator on/off<br>• {0}:Turn off System oscillator (standby mode)<br>• {1}:Turn on System oscillator (normal operation mode) | 20H |

Einstellung Helligkeit
0xEF – 0 b 1110 1111
(Helligkeit auf 16/16)

| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | ROW driver output pulse width | Def. |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | P3 | P2 | P1 | P0 | | |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 2/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 3/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 4/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 5/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 6/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7/16 duty | — |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 8/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 9/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 10/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 11/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 12/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 13/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 14/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 15/16 duty | — |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 16/16 duty | Y |

Einstellung Display ein/aus
Einstellung blinken ein/aus
0x81 – 0 b 1000 0001
(Display on, Blinking off)

(Auszug aus dem Datenblatt)

| Name | Command / Address / Data | | | | | | | | Option | Description | Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | | | |
| Display set | 1 | 0 | 0 | 0 | X | B1 | B0 | D | {D} Write only | Defines Display on/off status.<br>• {0}: Display off<br>• {1}: Display on | 80H |
| | | | | | | | | | {B1,B0} Write only | Defines the blinking frequency<br>• {0,0} = Blinking OFF<br>• {0,1} = 2HZ<br>• {1,0} = 1HZ<br>• {1,1} = 0.5HZ | |

Die Übertragung der Daten zwischen dem Pico und den Matrix Anzeigen erfolgt mit dem I²C Bus. Suchen wir als erstes nach den Adressen der einzelnen Anzeigen mit der Anweisung **i2c.scan()**. Damit werden uns 3 Module mit den folgenden Adressen angezeigt:

```
Scanne I2C Bus...                    (Bei meiner Hardware)
I2C-Geräte gefunden: 3
Dezimale Adresse: 117 | Hexadezimale Adresse: 0x75
Dezimale Adresse: 118 | Hexadezimale Adresse: 0x76
Dezimale Adresse: 119 | Hexadezimale Adresse: 0x77
```

Die Anwendung von **i2c.scan()** habe ich bereits in einem anderen Tutorial erläutert. Damit haben wir die Adressen unserer Matrix Anzeigen und können diese im Programm eintragen.

```
I2C_Addr_1 = 0x75  # Angabe Adresse 1 HT16K33
I2C_Addr_2 = 0x76  # Angabe Adresse 2 HT16K33
I2C_Addr_3 = 0x77  # Angabe Adresse 3 HT16K33
```

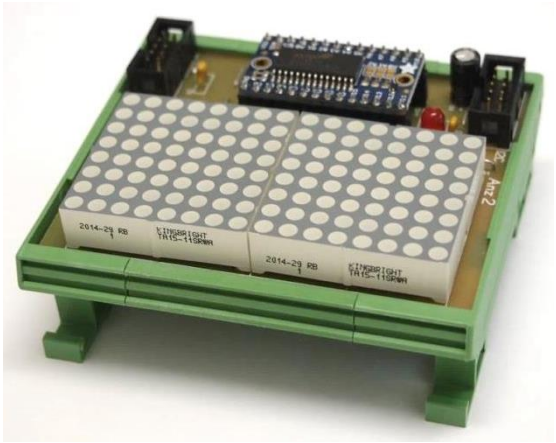
Als nächste erfolgt die Einstellung des I²C Bus:

```
SCL_Pin = 21 # nach der verwendeten Hardware, sonst 1
SDA_Pin = 20 # nach der verwendeten Hardware, sonst 0
Bus = 0 # Angabe Busnummer
# Initialisierung I2C, Bus 0, sda-0(20), scl-1(21)
i2c_Bus = I2C(Bus, scl = Pin(SCL_Pin), sda = Pin(SDA_Pin), freq = 400000)
```

Dann erfolgt die Einstellung der Register für jeden HT16K33:

```
i2c_Bus.writeto(I2C_Addr_1, bytes([0x21]))   # Einstellung Oscillator
i2c_Bus.writeto(I2C_Addr_1, bytes([0xEF]))   # Einstellung Helligkeit
i2c_Bus.writeto(I2C_Addr_1, bytes([0x81]))   # Einstellung Blinken aus, Display ein
```

Sehen wir uns als nächste das Modul mit 2x Matrix Anzeigen genauer an. Es besteht aus 2 x Matrix Anzeigen mit jeweils 8x8 LEDs und einer kleinen Platine mit dem HT16K33. Auf Grund der Grösse des HT16K33 kann ich 2 Matrix Anzeigen damit ansteuern.
Der Anschluss erfolgt mit dem 10 poligen Wannenstecker an den I²C Bus und die Stromversorgung. Die Stromversorgung erfolgt mit 5V. Beim Betrieb ist auf jeden Fall die Stromaufnahme zu beachten.



Der Aufbau jeder Matrix besteht aus 8 Zeile mit jeweils 8 LEDs.

| Anzeige 1 (links) | Anzeige 2 (rechts) |
|---|---|
| Zeile 0 | Zeile 1 |
| Zeile 2 | Zeile 3 |
| Zeile 4 | Zeile 5 |
| Zeile 6 | Zeile 7 |
| Zeile 8 | Zeile 9 |
| Zeile 10 | Zeile 11 |
| Zeile 12 | Zeile 13 |
| Zeile 14 | Zeile 15 |

Damit bestehen die 2 Matrix Anzeigen insgesamt aus 16 Zeilen. Gezählt wird von 0 bis 15 und entspricht damit 16 Zeilen.
Somit besteht die Anweisung zum Schalten der LEDs aus der folgenden Zeile:

**i2c_Bus.writeto(I2C_Addr_1, bytes([ 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 ]))**

Es werden insgesamt 17 Zahlen angegeben. Mit der ersten Zahl (**0**) wird angegeben mit welcher Zeile gestartet wird. Mit der Angabe 0 wird in der Zeile 0 gestartet, z.B. wird mit der Angabe 3 in der Anzeige 2 (rechts) in der 2. Zeile gestartet. Im oberen Bild sind die Zeilen angegeben. Alle weiteren Zahlen geben an wie viele LEDs in der jeweiligen Zeile leuchten sollen. Die Angabe 1 gibt an, das in der gewählten Zeile die LED 1 leuchtet. Diese Zahlen können von 0 (alle LEDs in der Zeile aus) bis 255 (alle LEDs in der Zeile an) gehen.

Kommen wir mal zum ersten Programm mit den Grundeinstellungen:

```
from machine import I2C, Pin # Laden der Lib
import time

# Vorbereitung Bus, Angabe Bus Nr. und Pins
I2C_Addr_1 = 0x75 # Angabe Adresse 1 HT16K33
SCL_Pin = 21      # oder 21 oder 1
SDA_Pin = 20      # oder 20 oder 0
Bus = 0           # Angabe Busnummer
# Initialisierung I2C, Bus 0, sda-0, scl-1
i2c_Bus = I2C(Bus, scl = Pin(SCL_Pin), sda = Pin(SDA_Pin), freq = 400000)
# Adresse 1
i2c_Bus.writeto(I2C_Addr_1, bytes([0x21]))    # Einstellung Oscillator
i2c_Bus.writeto(I2C_Addr_1, bytes([0xEF]))    # Einstellung Helligkeit
i2c_Bus.writeto(I2C_Addr_1, bytes([0x81]))    # Einstellung Blinken aus, Display ein

while True: # Endlos Schleife Beginn
    i2c_Bus.writeto(I2C_Addr_1, bytes([0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16])) # Display ein
    time.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes([0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])) # Display aus
    time.sleep(1)
```

Versuchen wir als nächstes ein Smiley auf unserer Matrix zu darzustellen.

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | X | X | X | X | | | 60 = 4 + 8 + 16 +32 |
| | X | | | | | X | | 66 = 2 + 64 |
| X | | X | | | X | | X | 165 = 1 + 4 + 32 + 128 |
| X | | | | | | | X | 129 = 1 + 128 |
| X | | X | | | X | | X | 165 = 1 + 4 + 32 + 128 |
| X | | | X | X | | | X | 153 = 1 + 8 + 16 + 128 |
| | X | | | | | X | | 66 = 2 + 64 |
| | | X | X | X | X | | | 60 = 4 + 8 + 16 + 32 |

Die Werte durch eine Addition der einzelnen Bits ermittelt.
Somit besteht das Smiley aus den Werten 60, 66, 165, 129, 165, 153, 66, 60. Leider sind diese Werte für uns nicht ausreichend. Das Smiley soll auf der Anzeige dargestellt werden.

In einem anderen Bild habe ich die Zuordnung der Zeile auf der Anzeige 1 und 2 dargestellt. Bei der Angabe der Werte muss das unbedingt beachtet werden. Bei der Anzeige 1 werden nur die Zeilen 0, 2, 4, 6, 8, 10, 12 und 14 benötigt. Somit ergibt sich:

**0, 60, 0, 66, 0, 165, 0, 129, 0, 165, 0, 153, 0, 66, 0, 60, 0**

Das kann man jetzt in diese Zeile eintragen:

i2c_Bus.writeto(I2C_Addr_1, bytes([**0,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0**]))

Damit wird auf dem Anzeige 1 ein Smiley dargestellt.
Es geht natürlich auch einfacher. Wenn man diese Werte in einen Data schreibt ergibt sich folgendes:

**smiley = [0,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]**

Im Programm wird diese Zeile genutzt:

**i2c_Bus.writeto(I2C_Addr_1, bytes(smiley))**

Das Programm:

```
from machine import I2C, Pin
import utime

data = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] # alle LED löschen

smiley = [0,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0] # Smiley
# Vorbereitung Bus, Angabe Bus Nr und Pins
I2C_Addr_1 = 0x75  # Angabe Adresse 1 HT16K33
SCL_Pin = 21 # 21 oder 1
SDA_Pin = 20 # 20 oder 0
Bus = 0 # Angabe Busnummer
# Initialisierung I2C, Bus 0, sda-0, scl-1
i2c_Bus = I2C(Bus, scl = Pin(SCL_Pin), sda = Pin(SDA_Pin), freq = 400000)
# Adresse 1
i2c_Bus.writeto(I2C_Addr_1, bytes([0x21]))   # Einstellung Oscillator
i2c_Bus.writeto(I2C_Addr_1, bytes([0xEF]))   # Einstellung Helligkeit
i2c_Bus.writeto(I2C_Addr_1, bytes([0x81]))   # Einstellung Blinken aus, Display ein
while True: # Endlos Schleife Beginn, es geht auch 0b10101101 oder 0x34
    i2c_Bus.writeto(I2C_Addr_1, bytes(data))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley))
    utime.sleep(1)
```

Natürlich kann man das einfach auf die Anzeige 2 bringen. Dazu muss einfach die erste **0** in eine **1** geändert werden

smiley = [**1**,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]

Dadurch wird in der Zeile 2 begonnen und alle anderen Zeilen werden angepasst. Man kann auch beide Anzeige abwechseln beschreiben mit:

Smiley_1 = [**0**,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]
Smiley_2 = [**1**,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]

Im Programm sieht es so aus:

```
i2c_Bus.writeto(I2C_Addr_1, bytes(data)) # alle LEDs aus
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_1)) # LEDs an
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(data)) # alle LEDs aus
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_2)) # LEDs an
utime.sleep(1)
```

Damit springt das Smiley zwischen der Anzeige 1 und 2 hin und her. Man kann das Smiley aber auch langsam und LED für LED schieben. Sehen wir uns dazu die Bits an:

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | X | X | X | X | | | 60 |
| | X | | | | | X | | 66 |
| X | | X | | | X | | X | 165 |
| X | | | | | | | X | 129 |
| X | | X | | | X | | X | 165 |
| X | | | X | X | | | X | 153 |
| | X | | | | | X | | 66 |
| | | X | X | X | X | | | 60 |

Schritt 1

Anzeige 1 (Smiley nur auf der Anzeige 1)

Schritt 2

Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | X | X | X | X | | 120 |
| | | X | | | | | X | 132 |
| | X | | X | | | X | | 74 |
| | X | | | | | | | 2 |
| | X | | X | | | X | | 74 |
| | X | | | X | X | | | 50 |
| | | X | | | | | X | 132 |
| | | | X | X | X | X | | 120 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 0 |
| | | | | | | | | 0 |
| X | | | | | | | | 1 |
| X | | | | | | | | 1 |
| X | | | | | | | | 1 |
| X | | | | | | | | 1 |
| | | | | | | | | 0 |
| | | | | | | | | 0 |

Schritt 3

Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | X | X | X | X | 240 |
| | | | X | | | | | 8 |
| | | X | | X | | | X | 148 |
| | | X | | | | | | 4 |
| | | X | | X | | | X | 148 |
| | | X | | | X | X | | 100 |
| | | | X | | | | | 8 |
| | | | | X | X | X | X | 240 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 0 |
| X | | | | | | | | 1 |
| | X | | | | | | | 2 |
| | X | | | | | | | 2 |
| | X | | | | | | | 2 |
| | X | | | | | | | 2 |
| X | | | | | | | | 1 |
| | | | | | | | | 0 |

Schritt 4 Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | X | X | X | 224 |
| | | | | X | | | | 16 |
| | | | X | | X | | | 40 |
| | | | X | | | | | 8 |
| | | | X | | X | | | 40 |
| | | | X | | | X | X | 200 |
| | | | | X | | | | 16 |
| | | | | | X | X | X | 224 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| X | | | | | | | | 1 |
| | X | | | | | | | 2 |
| X | | X | | | | | | 5 |
| | | X | | | | | | 4 |
| X | | X | | | | | | 5 |
| | | X | | | | | | 4 |
| | X | | | | | | | 2 |
| X | | | | | | | | 1 |

Schritt 5 Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | X | X | 192 |
| | | | | | X | | | 32 |
| | | | | X | | X | | 80 |
| | | | | X | | | | 16 |
| | | | | X | | X | | 80 |
| | | | | X | | | X | 144 |
| | | | | | X | | | 32 |
| | | | | | | X | X | 192 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| X | X | | | | | | | 3 |
| | | X | | | | | | 4 |
| | X | | X | | | | | 10 |
| | | | X | | | | | 8 |
| | X | | X | | | | | 10 |
| X | | | X | | | | | 9 |
| | | X | | | | | | 4 |
| X | X | | | | | | | 3 |

Schritt 6 Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | X | 128 |
| | | | | | | X | | 64 |
| | | | | | X | | X | 160 |
| | | | | | X | | | 32 |
| | | | | | X | | X | 160 |
| | | | | | X | | | 32 |
| | | | | | | X | | 64 |
| | | | | | | | X | 128 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| X | X | X | | | | | | 7 |
| | | | X | | | | | 8 |
| | | X | | X | | | | 20 |
| | | | | X | | | | 16 |
| | | X | | X | | | | 20 |
| X | X | | | X | | | | 19 |
| | | | X | | | | | 8 |
| X | X | X | | | | | | 7 |

Schritt 7 Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 0 |
| | | | | | | | X | 128 |
| | | | | | | X | | 64 |
| | | | | | | X | | 64 |
| | | | | | | X | | 64 |
| | | | | | | X | | 64 |
| | | | | | | | X | 128 |
| | | | | | | | | 0 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| X | X | X | X | | | | | 15 |
| | | | | X | | | | 16 |
| X | | | X | | X | | | 41 |
| | | | | | X | | | 32 |
| X | | | X | | X | | | 41 |
| | X | X | | | X | | | 38 |
| | | | | X | | | | 16 |
| X | X | X | X | | | | | 15 |

Schritt 8 Anzeige 1

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 0 |
| | | | | | | | | 0 |
| | | | | | | | X | 128 |
| | | | | | | | X | 128 |
| | | | | | | | X | 128 |
| | | | | | | | X | 128 |
| | | | | | | | | 0 |
| | | | | | | | | 0 |

Anzeige 2

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | X | X | X | X | | | | 30 |
| X | | | | | X | | | 33 |
| | X | | | X | | X | | 82 |
| | | | | | | X | | 64 |
| | X | | | X | | X | | 82 |
| | | X | X | | | X | | 76 |
| X | | | | | X | | | 33 |
| | X | X | X | X | | | | 30 |

| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | |
|---|---|---|---|---|---|---|---|---|
| | | X | X | X | X | | | 60 |
| | X | | | | | X | | 66 |
| X | | X | | | X | | X | 165 |
| X | | | | | | | X | 129 |
| X | | X | | | X | | X | 165 |
| X | | | X | X | | | X | 153 |
| | X | | | | | X | | 66 |
| | | X | X | X | X | | | 60 |

Schritt 9
Anzeige 2 (Smiley nur auf der Anzeige 2)

Somit ergeben sich die folgenden Werte:

**Schritt 1:** 0,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0
**Schritt 2:** 0,120,0,132,0,74,1,2,1,74,1,50,1,132,0,120,0
**Schritt 3:** 0,240,0,8,1,148,2,4,2,148,2,100,2,8,1,240,0
**Schritt 4:** 0,224,1,16,2,40,5,8,4,40,5,200,4,16,2,224,1
**Schritt 5:** 0,192,3,32,4,80,10,16,8,80,10,144,9,32,4,192,3
**Schritt 6:** 0,128,7,64,8,160,20,32,16,160,20,32,19,64,8,128,7
**Schritt 7:** 0,0,15,128,16,64,41,64,32,64,41,64,38,128,16,0,15
**Schritt 8:** 0,0,30,0,33,128,82,128,64,128,82,128,76,0,33,0,30
**Schritt 9:** 1,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0

Danach kann ich die Werte als Daten für das Smiley eintragen:

```
smiley_1 = [0,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]
smiley_2 = [0,120,0,132,0,74,1,2,1,74,1,50,1,132,0,120,0]
smiley_3 = [0,240,0,8,1,148,2,4,2,148,2,100,2,8,1,240,0]
smiley_4 = [0,224,1,16,2,40,5,8,4,40,5,200,4,16,2,224,1]
smiley_5 = [0,192,3,32,4,80,10,16,8,80,10,144,9,32,4,192,3]
smiley_6 = [0,128,7,64,8,160,20,32,16,160,20,32,19,64,8,128,7]
smiley_7 = [0,0,15,128,16,64,41,64,32,64,41,64,38,128,16,0,15]
smiley_8 = [0,0,30,0,33,128,82,128,64,128,82,128,76,0,33,0,30]
smiley_9 = [1,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]
```

Damit kann jeder für sich allein das gewünschte Bild ermitteln.
Mit dem Smiley 1-9 kann ich das Smiley von der linken Anzeige nach rechts wandern lassen.

Das vollständige Programm um das Smiley von der Anzeige 1 zur Anzeige 2 und wieder zurück laufen zu lassen:

```
from machine import I2C, Pin
import utime

smiley_1 = [0,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]
smiley_2 = [0,120,0,132,0,74,1,2,1,74,1,50,1,132,0,120,0]
smiley_3 = [0,240,0,8,1,148,2,4,2,148,2,100,2,8,1,240,0]
smiley_4 = [0,224,1,16,2,40,5,8,4,40,5,200,4,16,2,224,1]
smiley_5 = [0,192,3,32,4,80,10,16,8,80,10,144,9,32,4,192,3]
smiley_6 = [0,128,7,64,8,160,20,32,16,160,20,32,19,64,8,128,7]
smiley_7 = [0,0,15,128,16,64,41,64,32,64,41,64,38,128,16,0,15]
smiley_8 = [0,0,30,0,33,128,82,128,64,128,82,128,76,0,33,0,30]
smiley_9 = [1,60,0,66,0,165,0,129,0,165,0,153,0,66,0,60,0]

# Vorbereitung Bus, Angabe Bus Nr. und Pins
I2C_Addr_1 = 0x75  # Angabe Adresse 1 HT16K33
SCL_Pin = 21 # oder 21 oder 1
SDA_Pin = 20 # oder 20 oder 0
Bus = 0 # Angabe Busnummer
# Initialisierung I2C, Bus 0, sda-20, scl-21
i2c_Bus = I2C(Bus, scl = Pin(SCL_Pin), sda = Pin(SDA_Pin), freq = 400000)
# Einstellung Adresse 1
i2c_Bus.writeto(I2C_Addr_1, bytes([0x21]))   # Einstellung Oscillator
i2c_Bus.writeto(I2C_Addr_1, bytes([0xEF]))   # Einstellung Helligkeit
i2c_Bus.writeto(I2C_Addr_1, bytes([0x81]))   # Einstellung Blinken aus, Display ein

while True:
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_1))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_2))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_3))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_4))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_5))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_6))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_7))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_8))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_9))
    utime.sleep(1)
    i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_8))
    utime.sleep(1)
```

```
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_7))
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_6))
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_5))
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_4))
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_3))
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_2))
utime.sleep(1)
i2c_Bus.writeto(I2C_Addr_1, bytes(smiley_1))
utime.sleep(1)
```

Damit möchte ich die Vorstellung des HT16K33 mit einer Matrix Anzeige 8x8 am Pico beenden. Sicher gibt es noch andere Möglichkeiten einzelne LEDs an der Matrix anzusteuern.

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