ESP32 + MicroPython + OLed II

engelbert gruber

20.10.2020
2ND CLASS GRADING

VORAUSSETZUNG

OLed-Display am ESP32 mit MicroPython.

Programmieren tun wir in Thonny.

Als Display haben wir ein kleines SSD1306 128x64 Pixel.

- Thonny installiert.
- Esp32-idf3-20200902-v1.13.bin von https://micropython.org/download/esp32/ ist installiert.
- OLed und ESP32 verbinden:

ESP32	OLED
3V3	VCC
GND	GND
22	SCL
21	SDA

OLED PYTHON MODUL

Für die Library (das python-Modul) dass die Ansteuerung des Displays ermöglicht habe ich vier Möglichkeiten gefunden:

a. https://randomnerdtutorials.com/micropython-oled-display-esp32-esp8266/ Ein kompletter Artikel und hier der Code: https://raw.githubusercontent.com/RuiSantosdotme/ESP-MicroPython/master/code/Others/OLED/ssd1306.py
b. Adafruit: https://github.com/adafruit/Adafruit_CircuitPython_SSD1306
Ob CircuitPython-Bibliotheken auf MicroPython funktionieren ?
c. Adafruit für displayIO
https://github.com/adafruit/Adafruit_CircuitPython_DisplayIO_SSD1306/

Heisst displayIO das der Python-Prompt ">>>" auf dem OLed ist ?

d. https://github.com/micropython/micropython/blob/master/drivers/display/ssd1306.py

"d." ist ein Treiber direkt bei **micropython** ... ich verwende den.

- Den github-Link öffnen
- auf den "Raw"-Button klicken, dann wird nur der Pythoncode angezeigt.
- Den Pythoncode in eine neue Datei im Thonny kopieren und in eine Datei mit Namen "ssd1306.py" speichern. Auf "This Computer" und im "MicroPython device".
-

TEST DES MODULS

- In der thonny-Shell "import ssd1306" eingeben. Es darf keine Meldung kommen.
- In thonny eine neue Datei "htl-oled.py" erstellen:

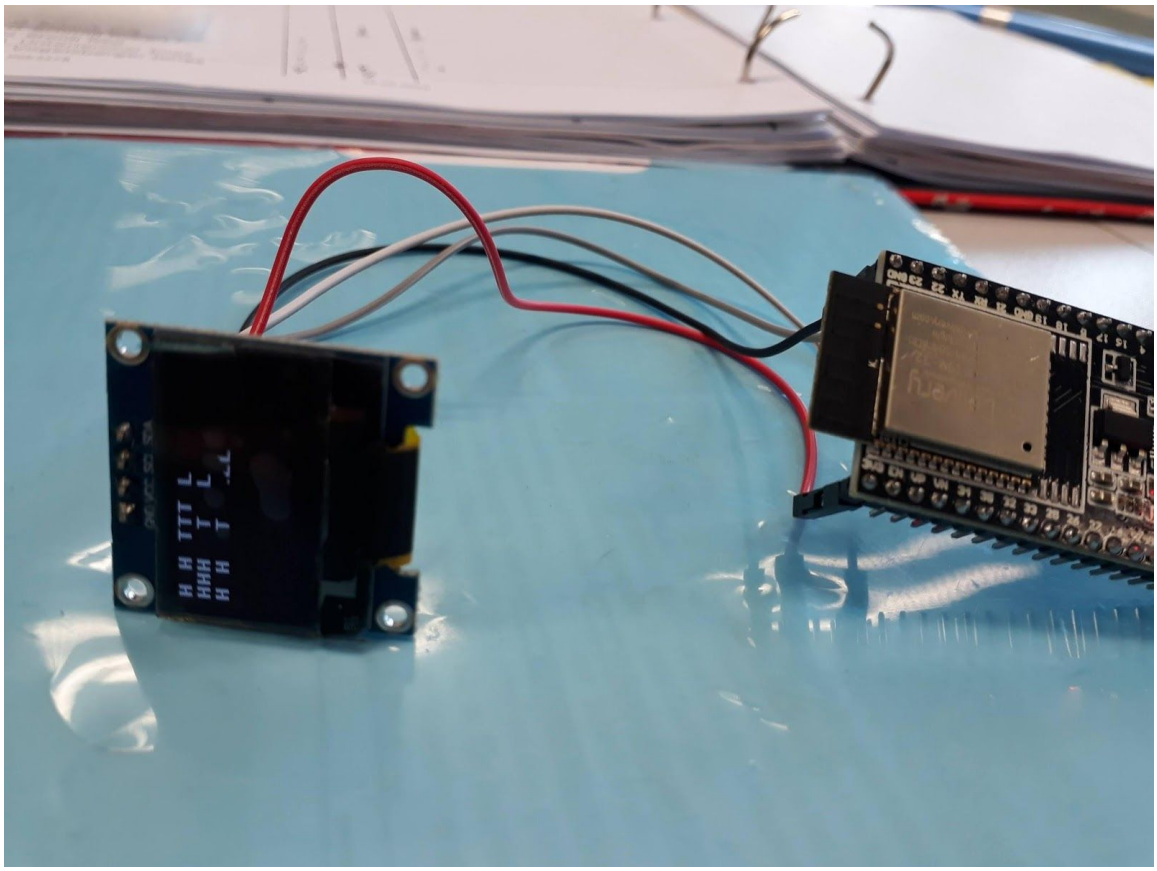
```
from machine import Pin, I2C
import ssd1306
i2c = I2C(-1, scl=Pin(22), sda=Pin(21))
oled_width = 128
oled_height = 64
oled = ssd1306.SSD1306_I2C(oled_width, oled_height, i2c)

oled.text('H H TTT L', 0, 0)
```

```
oled.text('HHH  T  L', 0, 10)
oled.text('H H  T  LLL', 0, 20)

oled.show()
```

- “Ausführen” - “Run current script F5”



ETWAS GENAUER

Das SSD1306 Modul baut auf dem framebuf-Modul von micropython auf. Das sieht man am Anfang der ssd1306.py Datei:

```
# MicroPython SSD1306 OLED driver, I2C and SPI interfaces
from micropython import const
import framebuf
```

Das Modul “`framebuf`” wird importiert.

Und etwas weiter unten, nach den Konstanten steht:

```
# Subclassing FrameBuffer provides support for graphics primitives
```

```
# http://docs.micropython.org/en/latest/pyboard/library/framebuf.html
class SSD1306(framebuf.FrameBuffer):
```

Der Ausdruck "`class SSD1306(framebuf.FrameBuffer):`" bedeutet, dass die Klasse SSD1306 auf der Klasse FrameBuffer aufbaut.

Die Klasse SSD1306 hat nur wenige Methoden (Funktionen) zur Initialisierung des Displays, die Zeichenroutinen sind in framebuf.FrameBuffer implementiert.

> https://docs.micropython.org/en/latest/library/framebuf.html

`fill(color)`

`pixel(x,y, color)`

`line(x1, y1, x2, y2, color)`

`hline(x, y, length, color)`

`vline(x, y, height, color)`

`rect(x, y, width, height, color)` ... nur ein Pixel Rahmen

`fill_rect(x, y, width, height, color)` ... ein ausgefülltes Rechteck

`test( string, x, y, color )`

`scroll( x_step, y_step )` ... verschiebt den Bildinhalt.

`blit` ... legt den Inhalt eines anderen FrameBuffer über diesen ... interessant, aber leichter in einem Beispiel zu erklären.

REFERENCES

1. http://micropython.org/
2. https://github.com/micropython/micropython
3. https://docs.micropython.org/en/latest/library/framebuf.html
4. Thonny : https://thonny.org/