ESP32, ESP8266, Hardware, Nuetzliche Links

ESP32 / ESP8266

Nützliche Links

- ESP32 Espressif Systems
- ESP32 Series Datasheet
- ESP32 Pinbelegung
- ESP32 Pinout Reference: Which GPIO pins should you use?

ESP Hardware

- ESP32 vs ESP8266

	ESP8266	**ESP32**
CPU-Kerne	1	2
Taktrate	160 MHz	240 MHz
Flash-Speicher	4 MB	4 MB
SRAM	160 KB	512 KB
GPIO-Pins	17	36
Touchsceeen-Sensoren	nein	10
ADC-Kanäle	1	16
Bluetooth	nein	ja
Hardware-PWM-Kanäle	nein	ja
Software-PWM-Kanäle	8	16
Verbrauch	80 mA	260 mA

Wer ist besser geeignet für simple Sensoren?

Beide Micro-PCs besitzen zahlreiche GPIO-Pins, unterstützen die Funkverbindung per WLAN, lassen sich (sofern du die NodeMCU-Variante der Boards verwendest) per micro USB Kabel mit Strom versorgen und können mit der Arduino IDE programmiert werden. Während beide Boards je nach Marktplatz für ca 3-6€ erhältlich sind, ist der ESP8266 aufgrund des fehlenden zweitren Kerns und da er kein Bluetooth unterstützt in der Regel ein wenig günstiger.

Wer ist besser geeignet für komplexe Geräte?

Für komplexere Geräte, die Multithreading oder eine Bluetooth-Verbindung voraussetzen ist hingegen der ESP32 vermutlich die sinnvollere Wahl, da der ESP8266 diese Funktionen schlicht einfach nicht bietet.

Peripherie

- ESP32 TTGO T-Display
- ESP32-CAM Dokumentation
- ESP32-CAM-MB
- SSD1306 OLED Display
- Human Static Presence Module Lite (MR24HPC1)
- ESP Muse Luxe
- sensoren

ESP Projekte

- ESP32 Sromversorgung
- ESP32/ESP8266 Deep Sleep
- Frigate
- ESP32 Bluetooth Presence Detection

esptool

- Esptool.py Documentation
- `esptool.py –port /dev/ttyUSB0 erase_flash`
- `esptool.py –chip esp32 –port /dev/ttyUSB0 write_flash -z 0x1000 /home/cs/Downloads/esp32-20220117-v1.18.bin`
- die Befehle werden **fehlerhafte aus DokuWiki** kopiert am besten von hier kopieren.

Serielle Terminals

- **picocom** /dev/ttyUSB0 -b115200
- **uPyLoader** uPyLoader git

- **mpfshell** mpfshell Tutorial
- **rshell**
  - https://github.com/dhylands/rshell
  - Aufruf: ins Arbeitsverzeichnis wechselm und dann `rshell –buffer-size=30 -p/dev/ttyUSB0 -a -e nano`
  - Dann ähnlich wie bash. Der ESP32 wird mit /pyboard angesprochen, z.B. `cd /pyboard`.
  - repl ist möglich
  - `cp file.py /pyboard` kopiert file.py auf den ESP32
  - `edit /pyboard/start.py` editiert auf dem ESP32
  - Aktuelles Verzeichnis synchronisieren: `rsync . /pyboard`

µPyLoader

- µPyLoader

Jupyter

- Jupyter MicroPython Kernel „pip install jupyter_micropython_kernel“, und „python -m jupyter_micropython_kernel.install“
- im Terminal „jupyter notebook“ starten und mit `%serialconnect to –port=/dev/ttyUSB0 –baud=115200` zum ESP32 verbinden

Nützliche Links

- ESPHome Core Configuration
- ESPHome Configuration Types (Substitution etc.)
- ESPHome Display Component
- ESPHome Binary Sensor Component
- ESPHome Logger Component