TTGO ESP32 WATCH-20

Man kann diese Software nur selbst kompilieren.
z.B. mit ATOM und PlatformIO
Damit ist kompilieren und Hochladen sehr einfach.

Die Uhr hat eine Micro USB Buchse mit der man sie direkt seriell programmieren kann
Der Treiber für die serielle Schnittstelle ist derselbe wie für die meisten ESP32 boards

bei OSX
upload_port = /dev/cu.SLAB_USBtoUART

Bei Windows
muss man den entsprechenden COM Port finden

Nach dem Flashen hat man ganz normalen Zugriff auf die WEBUI
und die Uhr zeigt die Daten des Power Controllers
und die X,Y,Z Daten des Beschleunigungssensors an

Die Uhr ist jetzt bereit zur Programmierung mit Skripten

user_config_override.h
---

zusätzlich zu den Definitionen des eigenen WLANs:

am besten alle Treiber abschalten, die nicht gebraucht werden
um Platz zu sparen. Liste eventuell noch nicht vollständig

```
#undef USE_LM75AD
#undef USE_SHT
#undef USE_HTU
#undef USE_BME680
#undef USE_BH1750
#undef USE_MHZ19
#undef USE_SENSEAIR
#undef USE_PMS5003
#undef USE_IR_RECEIVE
#undef USE_WS2812
#undef USE_ARILUX_RF
#undef USE_NOVA_SDS
#undef USE_ENERGY_SENSOR
#undef USE_DOMOTICZ
#undef USE_HOME_ASSISTANT
#undef USE_DISCOVERY
#undef USE_DS18x20
#undef USE_SERIAL_BRIDGE
#undef USE_PZEM004T
```

```
#undef USE_SHT3X
#undef USE_SGP30
#undef USE_SPS0
#undef USE_SCD30
#undef USE_MLX90614
#undef USE_PZEM2
#undef USE_RF_FLASH
#undef USE_APDS9960
#undef USE_ADE7953
#undef USE_EMULATION_HUE
#undef USE_EMULATION_WEMO
#undef USE_DHT
#undef USE_BL0940
//#undef USE_COUNTER
#undef USE_TX20_WIND_SENSOR
#undef USE_IR_REMOTE
#undef USE_KNX
#undef USE_HOME_ASSISTANT
#undef USE_PCA9685
#undef USE_TUYA_DIMMER
#undef USE_RC_SWITCH
#undef USE_ARMTRONIX_DIMMERS
#undef USE_PS_16_DZ
#undef USE_SONOFF_IFAN
#undef USE_EMULATION_HUE
#undef USE_LIGHT
#undef USE_EMULATION
#undef USE_HX711
#undef USE_CSE7766
#undef USE_SR04
#undef USE_DS18x20
#undef USE_DS18B20
#undef USE_DS18x20_LEGACY
#undef USE_BMP
#undef USE_DS1820
#undef USE_PZEM004T
#undef USE_PZEM_AC
#undef USE_PZEM_DC
#undef USE_TIMERS
#undef USE_TIMERS_WEBINTERFACE
#undef USE_PN532_HSU
#undef USE_PN532_I2C
#undef USE_PWM_DIMMER
#undef USE_LIGHT
#undef USE_SONOFF_D1
#undef USE_MCP230xx
#undef USE_MCP39F501
#undef USE_COUNTER
#undef USE_BUZZER
#undef USE_SERIAL_BRIDGE
#undef USE_DEEPSLEEP
#undef USE_ADE7953
#undef USE_ENERGY_MARGIN_DETECTION
#undef USE_ENERGY_POWER_LIMIT
```

```
#undef USE_PZEM004T
#undef USE_PZEM_AC
#undef USE_PZEM_DC
#undef USE_MCP39F501
#undef USE_SONOFF_SC
#undef USE_ADC_VCC
#undef USE_COUNTER
#undef USE_SONOFF_RF
#undef USE_TUYA_MCU
#undef USE_BUZZER
#undef USE_SHUTTER
#undef USE_DEEPSLEEP
#undef USE_EXS_DIMMER
#undef USE_DEVICE_GROUPS
#undef USE_SM16716
#undef USE_SM2135
#undef USE_ELECTRIQ_MOODL
#undef USE_TELEINFO
#undef USE_OPENTHERM
#undef USE_IBEACON
#undef USE_GPS
#undef USE_HM10



// auf Deutsch stellen
#define MY_LANGUAGE            de_DE


// die script Sprache ist extrem modular aufgebaut
// aus Speicherplatz Gründen kann man viele Optionen nur einschalten
// wenn man sie auch wirklich braucht
// die unten angegebene Definition erlaubt sehr vielseitige Nutzung
// und passt bequem in den grossen ESP32 Speicher


// rules aus, script an
#undef USE_RULES
#define USE_SCRIPT
// Flash file System einschalten
// ergibt auf der Uhr 12 MB Platz
#define USE_SCRIPT_FATFS -1
#undef FAT_SCRIPT_SIZE
// script default Größe auf 8kB
#define FAT_SCRIPT_SIZE 8192
// zusätzliche Befehle für Dateizugriff
#define USE_SCRIPT_FATFS_EXT
// Dateiverzeichnis in WebUI anzeigen zum Datenaustausch mit dem
Dateisystem
#define SDCARD_DIR

// web display an
#define USE_SCRIPT_WEB_DISPLAY
```

```
// array Größe vergrößern für google charts
#define LARGE_ARRAYS
// mehr Platz für Variablennamen, bei großen Scripts sinnvoll
#define SCRIPT_LARGE_VNBUFF
// erlaubt globale Variablen
#define USE_SCRIPT_GLOBVARS
// i2c an für Uhr
#define USE_I2C
// Uhr einbinden
#define USE_TTGO_WATCH
// Uhren 3D Sensor (erkennt Doppelklick auf Display durch Vibration)
#define USE_BMA423
// Zeitansage erlauben
#define SAY_TIME
// webradio erlauben, mp3 abspielen und Text to Speech ist per
default an
#define USE_WEBRADIO
// multitasking erlauben
#define USE_SCRIPT_TASK
// mehr Arrays erlauben
#define MAXFILT 10
// bestimmte Funktionen im Script hinzunehmen
#define USE_ANGLE_FUNC
// Variablenanzahl im script erhöhen
#define MAXVARS 75
#define MAXSVARS 15
// Volle Webseite erlauben
#define SCRIPT_FULL_WEBPAGE
// Google Charts erlauben
#define USE_GOOGLE_CHARTS

// Display der Uhr erlauben
#define USE_DISPLAY
#define USE_SPI
#define USE_DISPLAY_ST7789
#undef USE_DISPLAY_MODES1TO5
// touch Controller erlauben
#define USE_FT5206
// Touch Buttons erlauben
#define USE_TOUCH_BUTTONS
// jpeg Bilder für Display erlauben
#define JPEG_PICTS

// mail für ESP32 erlauben
#define USE_SENDMAIL
#define USE_ESP32MAIL
// hier eigene email Daten eintragen
#define EMAIL_USER "user"
#define EMAIL_PASSWORD "passwd"
#define EMAIL_FROM "mr.x@gmail.com"
#define EMAIL_SERVER "smtp.gmail.com"
#define EMAIL_PORT 465
#undef EMAIL_FROM
#define EMAIL_FROM "mustermann@googlemail.com"
```

```
// friendly name etc z.B.
#undef FRIENDLY_NAME
#define FRIENDLY_NAME        "esp32_watch"
#undef MQTT_CLIENT_ID
#define MQTT_CLIENT_ID       "esp32_watch"
```

platformio_override.ini
---

z.B. tasmota32-DE einschalten

```
; hier esp32cam eintragen, weil es noch keine Definition für die Uhr
gibt
; die cam entspricht aber der Uhr bezüglich PSRAM Speicher etc
board                   = esp32cam
; eigenes Linkerfile erlaubt 12MB Flash file System
board_build.partitions  = esp32_partition_app1984k_ffat12M.csv
```