



Über mich

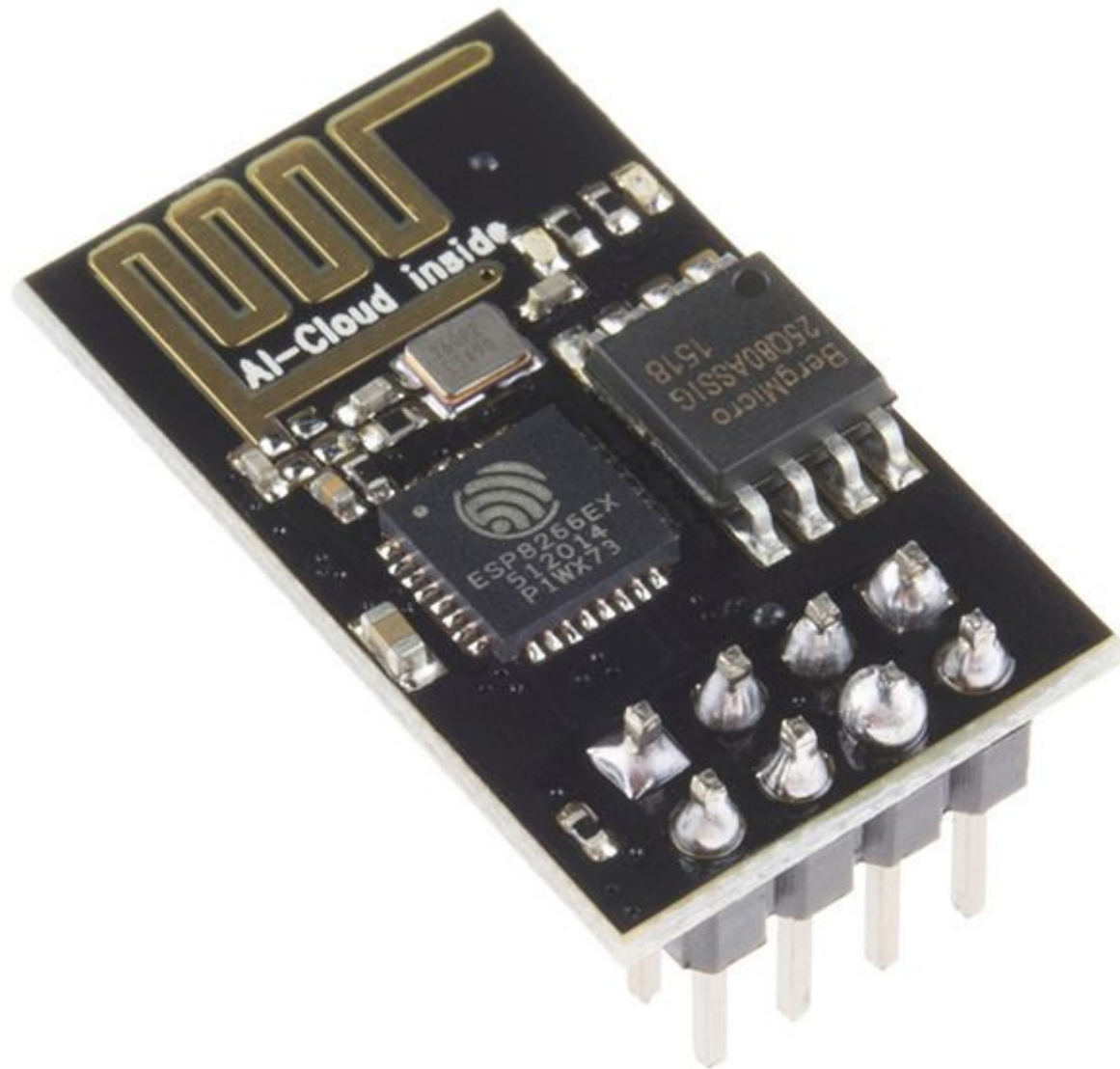


- Tam HANNA
- Tamoggemon Holding k.s.
- Instagram: tam.hanna
- Twitter: tamhanna

Vortrag wird GETAPED!

BITTE TAPE ANSEHEN!!!

Slides sind alleinstehend verwirrend!!!



Bildquelle: SparkFun

Vom ESP8266 zum ESP32

ESP8266 – was n das?

- 32-Bit-Prozessor
 - KEIN ARM (!!!)
- WLAN-Funkmodul
- GPIO-Besskram
- **BILLIG, WENIG BESCHALTUNG**

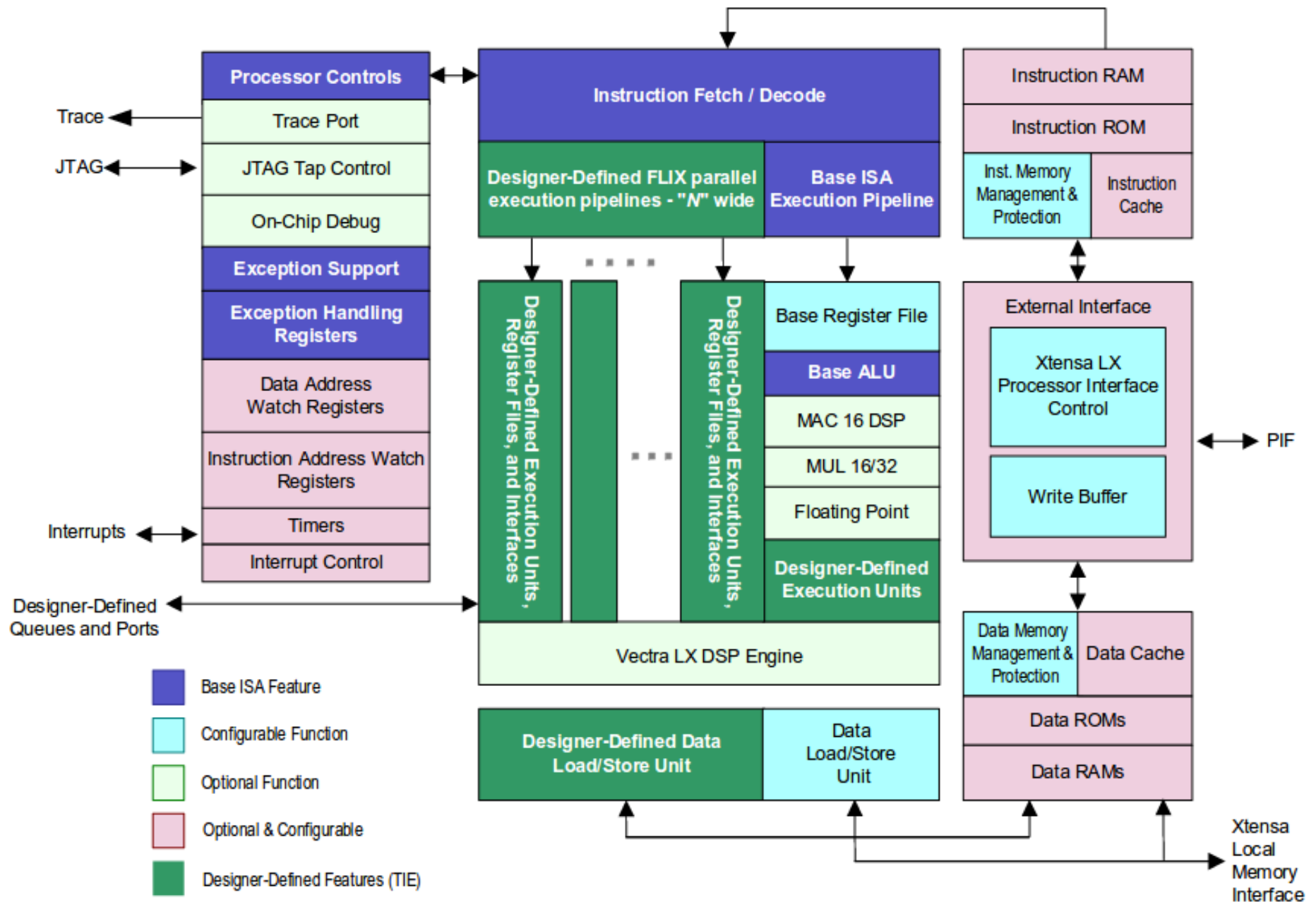
ESP8266-Probleme

- Kein Bluetooth
- Vergleichsweise wenig Power
 - 1 Core, 80 MhZ
- Nur Software-PWM
- Hoher Energieverbrauch

Cadence TenSilica

- Stark konfigurierbarer Softcore
 - Cadence nutzt ihn in hauseigenen DSPs
 - Beispiel Huffman-Encoding
- RISC-CPU mit 16/24bit-ISA
- <https://0x04.net/~mwk/doc/xtensa.pdf>

Cadence TenSilica - II



Neuerungen des ESP32

Attribut	ESP8266	ESP32
CPU-Aufbau	80 MhZ 1 Kern	160 MhZ 2 Kerne
RAM	160KBytes	512KBytes
GPIO-Pins (max)	17	36
Low-Power-Insel	NEIN	JA
ADC-Auflösung	10bit	12bit
Hardware-PWM	NEIN	JA
Besskram	keiner	CAN, Ethernet-MAC, mehr Hardwareeinheiten, Bluetooth-Transmitter

Probleme des ESP32

- Alte Revisionen des Siliziums “kursieren”
 - Achtung am Graumarkt!!!
- Darreichungsform
 - Von Hand “unleicht” zu löten
 - Verschiedene fertige Platinen
 - CAVE: Zertifikation (Bundesnetzagentur!!!)
 - CAVE: Mapping der Pins
 - CAVE: FTDI oder anderer Anbieter

ESP32 mit Windows / Mac

- Espressif ist ein “All-Linux-Shop”
- Windows und Mac “gehen”
 - Tut es besser nicht
 - IDF artet in Selbstflagellation aus
 - Glaubt dem rassistischen Halbaraber
 - Bitte ;) ;) ;)



Vom Wirsing ...

- “Abstraktionsebene für Hardware”
- Klassiker: setup() und loop()
- Vorteil: kennt jeder
- Nachteile: /me will nicht getötet werden ;)

ESP32-spezifisch ...

 **espressif** / **arduino-esp32**

 Watch **303**  Star **1,602**  Fork **627**

[Code](#) [Issues **324**](#) [Pull requests **11**](#) [Projects **1**](#) [Insights](#)

[Labels](#) [Milestones](#) [New issue](#)

[X](#) Clear current search query, filters, and sorts

 13 Open	 12 Closed	Author ▾	Labels ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
	Guru Meditation Error: Core 1 panic'ed (Cache disabled but cached memory region accessed)						 7
	ESP32 ARDUINO IDE for reference						
	#855 by igrr was closed on Nov 24, 2017						
	Websocket client support for reference						 5
	#713 opened on Oct 10, 2017 by paulfer747						
	Silent Start (no out message) - details available. for reference						 7
	#642 by luc-github was closed on Sep 29, 2017						
	E (588) SPIFFS: mount failed, -10025 for reference						 15
	#638 opened on Sep 16, 2017 by emilekm2142						
	[QUESTION] Is there a way to send custom data using SmartConfig for reference						 4
	#598 by Raienryu97 was closed on Sep 1, 2017						
	ESP32 suddenly stop appearing in serial port for reference						 15
	#570 by kosmic21 was closed on Oct 20, 2017						

Installation der Umgebung

- Unter Windows: Handarbeit
- Unter Linux: Shell-Skript abarbeiten
 - Artet auch in Arbeit aus
 - https://github.com/espressif/arduino-esp32/blob/master/docs/arduino-ide/debian_ubuntu.md

Startzeit: langsam

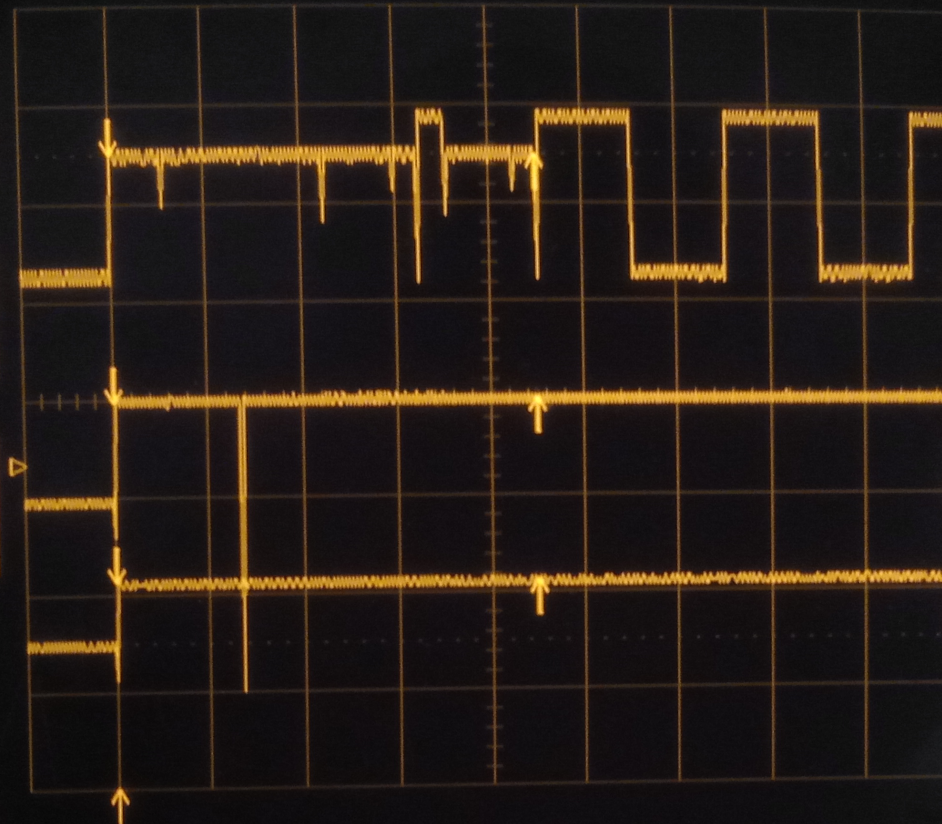
3-Apr-17
13:22:29

1
.5 s
2.46 V
2.46 V

2
.5 s
4.79 V
4.95 V

3
.5 s
3.32 V
3.32 V

.5 s BWL
1 .2 V DC X
2 .5 V DC X
3 .5 V DC X
4 5 V DC X



CURSORS

mode
Time
Amplitude

type
Relative
Absolute

show
Diff - Ref
Diff & Ref

Reference
cursor
Track OFF On

Difference
cursor

Δt 2.25424 s $\frac{1}{\Delta t}$ 443.61 MHz

20 ks/s

2 DC 1.7 V

STOPPED

analogWrite == Beef

- 16 channels LEDC (PWM)
- 8 channels SigmaDelta
- 2 channels DAC

DAC in easy

```
void loop()
{
    for(int i=0;i<255;i++)
    {
        dacWrite(25, i);
    }
}
```

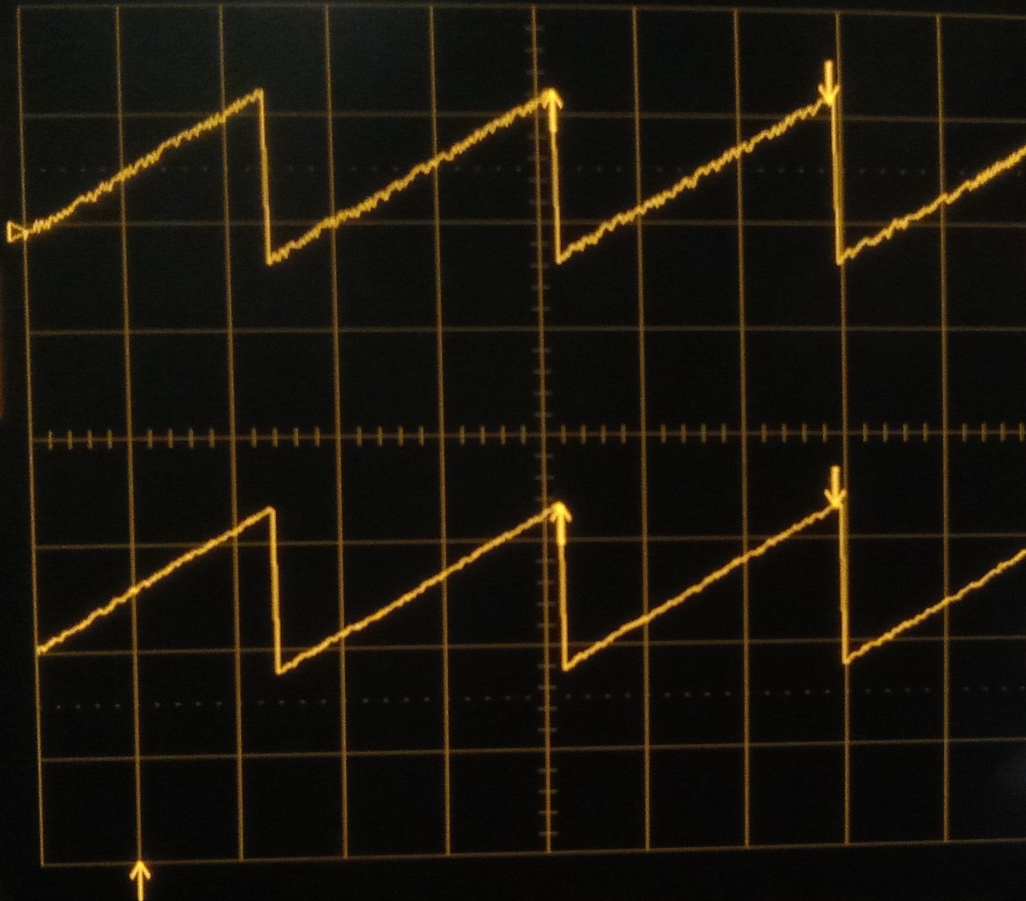
Schnell sind wir nicht...

3-Apr-17
19:06:35

CURSORS

1
.5 ms
3.01 V
3.14 V

2: Eres(1)
.5 ms
3.031 V
3.091 V



mode
Time
Amplitude

type
Relative
Absolute

show
Diff - Ref
Diff & Ref

Reference
cursor
Track OFF On

Difference
cursor

.5 ms BWL
1 .2 V DC X
2 .5 V DC X
3 .5 V DC X
4 5 V DC X

Δt -1.37696 ms $\frac{1}{\Delta t}$ 726.24 Hz

50 MS/s

1 DC 0.60 V

STOPPED

WLAN-API...<WiFi.h>

```
void setup()
{
    Serial.begin(115200);
    delay(10);

    Serial.println();
    Serial.println();
    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    Serial.println("");
    Serial.println("WLAN verbunden - IP lautet:");
    Serial.println(WiFi.localIP());
}
```


Arduino bleibt Arduino

```
void loop() {  
  WiFiClient client;  
  const int httpPort = 80;  
  if (!client.connect("tamoggemon.com", httpPort)) {  
    Serial.println("connection failed");  
    return;  
  }  
  String url="/test/2017/Elektorfile.txt";  
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +  
    "Host: " + "tamoggemon.com" + "\r\n" +  
    "Connection: close\r\n\r\n");  
  unsigned long timeout = millis();
```



Lieber nativ ...

IoT Development Framework

- Kurz ESP-IDF
- Native Entwicklungsumgebung
 - Bringt FreeRTOS mit
- Teilweise präkompiliert!!!
 - Ordner tamhan@tamhan-thinkpad:~/esp\$

Toolchain laden

- Archiv herunterladen
- In Unterordner von /esp entpacken
- `tar -xzf ~/Downloads/xtensa-esp32-elf-linux64-1.22.0-80-g6c4433a-5.2.0.tar.gz`
- `export PATH="$PATH:$HOME/esp/xtensa-esp32-elf/bin"`
- <http://esp-idf.readthedocs.io/en/latest/get-started/linux-setup.html>

ESP-Framework laden

- Entwicklung erfolgt auf GitHub
- `git clone --recursive https://github.com/espressif/esp-idf.git`
- `export IDF_PATH=~/.esp/esp-idf`

```
tamhan@tamhan-thinkpad:~/esp$ ls  
esp-idf  xtensa-esp32-elf  
tamhan@tamhan-thinkpad:~/esp$
```

Beispiel herbei

- `cp -r $IDF_PATH/examples/get-started/hello_world .`

```
tamhan@tamhan-thinkpad:~/esp/esp-idf/examples$ tree -L 1
.
├── bluetooth
├── ethernet
├── get-started
├── peripherals
├── protocols
├── README.md
├── storage
├── system
└── wifi
```

Gerät lokalisieren

- ESP32 hängt an USB-Seriell-Brücke
 - Verschiedene Anbieter
 - Treiber sollte vorhanden sein (Linux ;))

```
tamhan@tamhan-thinkpad:~$ lsusb
Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 004: ID 04f2:b2da Chicony Electronics Co., Ltd thinkpad t430s camera
Bus 001 Device 003: ID 0a5c:21e6 Broadcom Corp. BCM20702 Bluetooth 4.0 [ThinkPad]
Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 002: ID 145f:01bd Trust
Bus 003 Device 003: ID 0403:6015 Future Technology Devices International, Ltd Bridge(I2C/SPI/U
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

```
tamhan@tamhan-thinkpad:~$ dmesg | grep "ttyUSB"
[ 8957.489851] usb 3-1: FTDI USB Serial Device converter now attached to tttyUSB0
tamhan@tamhan-thinkpad:~$
```

Menügetriebene Konfiguration

- ESP-IDF “konfiguriert sich selbst”
- Nutzt menuconfig (bekannt vom Kernel)
- make menuconfig eingeben

```
/home/tamhan/esp/hello_world/sdkconfig - Espressif IoT Development Framework
C> Serial flasher config
```

Serial flasher config

Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*]

(/dev/ttyUSB0) Default serial port

Default baud rate (115200 baud) --->

[*] Use compressed upload

Flash SPI mode (DIO) --->

Flash SPI speed (40 MHz) --->

Flash size (2 MB) --->

[*] Detect flash size when flashing bootloader

Before flashing (Reset to bootloader) --->

After flashing (Reset after flashing) --->

'make monitor' baud rate (115200 bps) --->

<Select> < Exit > < Help > < Save > < Load >

Wichtige Felder

- Serial flasher config
 - Gerät und Geschwindigkeit
- Compiler options
 - Optimierungsgrad

Deployment

- User muss root oder Member von dialout sein
 - `sudo usermod -a -G dialout $USER`
 - Logout/Login bzw Reboot
- `make flash -j N`
- `-j` ist unbedingt erforderlich
 - ESP-IDF ist sehr groß

Der Code im Blick

```
/*  
#include <stdio.h>  
#include "freertos/FreeRTOS.h"  
#include "freertos/task.h"  
#include "esp_system.h"  
#include "esp_spi_flash.h"  
  
|  
void app_main()  
{  
    printf("Hello world!\n");  
}
```

Der Code im Blick - II

```
|  
void app_main()  
{  
    printf("Hello world!\n");  
  
    /* Print chip information */  
    esp_chip_info_t chip_info;  
    esp_chip_info(&chip_info);  
    printf("This is ESP32 chip with %d CPU cores, WiFi%s%s, ",  
           chip_info.cores,  
           (chip_info.features & CHIP_FEATURE_BT) ? "/BT" : "",  
           (chip_info.features & CHIP_FEATURE_BLE) ? "/BLE" : "");  
  
    printf("silicon revision %d, ", chip_info.revision);  
  
    printf("%dMB %s flash\n", spi_flash_get_chip_size() / (1024 * 1024),  
           (chip_info.features & CHIP_FEATURE_EMB_FLASH) ? "embedded" : "external");  
  
    for (int i = 10; i >= 0; i--) {  
        printf("Restarting in %d seconds...\n", i);  
        vTaskDelay(1000 / portTICK_PERIOD_MS);  
    }  
    printf("Restarting now.\n");  
    fflush(stdout);  
    esp_restart();  
}
```

make monitor

- Von logcat inspiriertes Werkzeug
- “Einfacher Debugger”
- Startet Programm automatisch neu!!!
- Ausbrechen mit Strg+]



Hardware-Interaktion

Wie man den DAC nutzt...

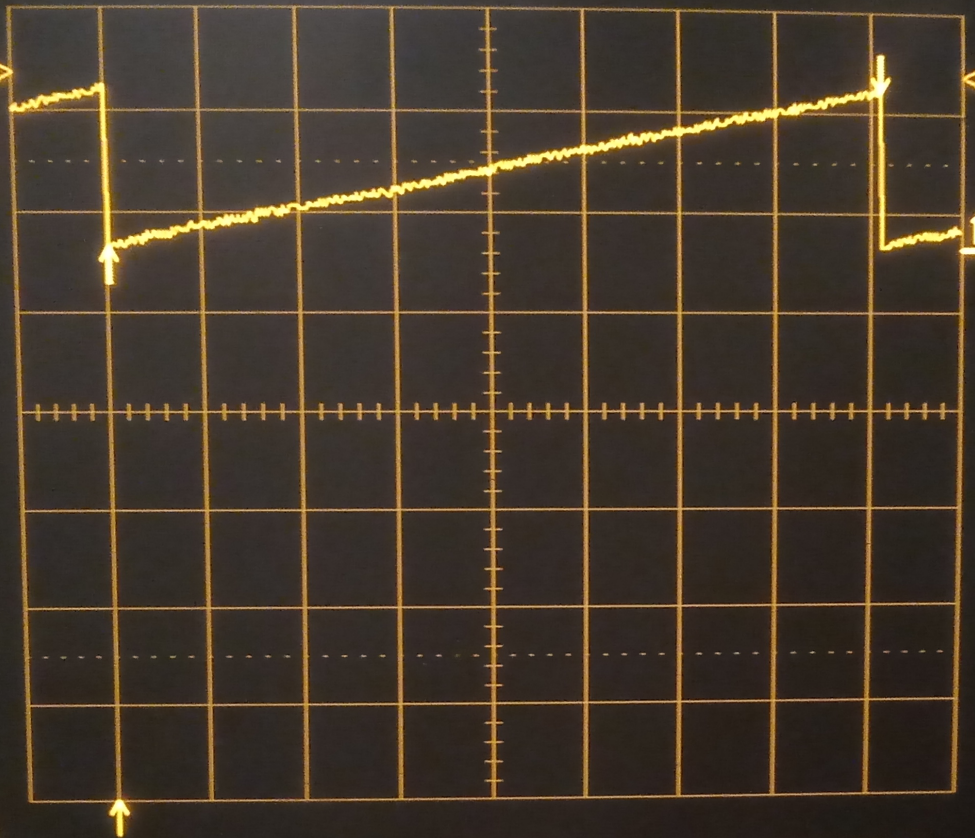
```
#include <driver/dac.h>
```

```
for(int i=0;i<255;i++){  
    dac_out_voltage(DAC_CHANNEL_1, i);  
}
```

Langsamer...

3-Apr-17
21:14:22

1
.5 ms
3.14 V
0.01 V



.5 ms BNL
1 .2 V DC X
2 .5 V DC X
3 .5 V DC X
4 5 V DC X

Δt -4.03698 ms $\frac{1}{\Delta t}$ 247.71 Hz

50 MS/s

1 DC 3.48 V

STOPPED

Achtung: komplex!

- Viele Header
- Vorlage nutzen!!!

```
#include "bt.h"  
#include "bta_api.h"  
  
#include "esp_gap_ble_api.h"  
#include "esp_gatts_api.h"  
#include "esp_bt_defs.h"  
#include "esp_bt_main.h"  
#include "esp_bt_main.h"
```

Läufer-Objekte

```
void app_main()
{
    esp_err_t ret;

    esp_bt_controller_config_t bt_cfg = BT_CONTROLLER_INIT_CONFIG_DEFAULT();
    ret = esp_bt_controller_init(&bt_cfg);
    if (ret) {
        ESP_LOGE(GATTS_TAG, "%s initialize controller failed\n", __func__);
        return;
    }

    ret = esp_bt_controller_enable(ESP_BT_MODE_BTDM);
    if (ret) {
        ESP_LOGE(GATTS_TAG, "%s enable controller failed\n", __func__);
        return;
    }
}
```

Asynchrones Programmieren

```
void app_main()
{
    esp_err_t ret;

    . . .|
    esp_ble_gatts_register_callback(gatts_event_handler);
    esp_ble_gap_register_callback(gap_event_handler);
    esp_ble_gatts_app_register(0);

    return;
}
```


“Mahlen” der Enums

```
static void gap_event_handler(esp_gap_ble_cb_event_t event,  
{  
    switch (event) {  
        case ESP_GAP_BLE_ADV_DATA_SET_COMPLETE_EVT:  
            esp_ble_gap_start_advertising(&test_adv_params);  
            break;  
        case ESP_GAP_BLE_ADV_DATA_RAW_SET_COMPLETE_EVT:  
            esp_ble_gap_start_advertising(&test_adv_params);  
            break;  
        case ESP_GAP_BLE_SCAN_RSP_DATA_RAW_SET_COMPLETE_EVT:  
            esp_ble_gap_start_advertising(&test_adv_params);  
            break;
```

Allgemeiner Prozess

- Beispiel aus ESP-IDF finden und klonen
- “Ausweiden”
- Eigene Logik einbauen
- Bei Fehlschlag: Prozessrechner holen!!!

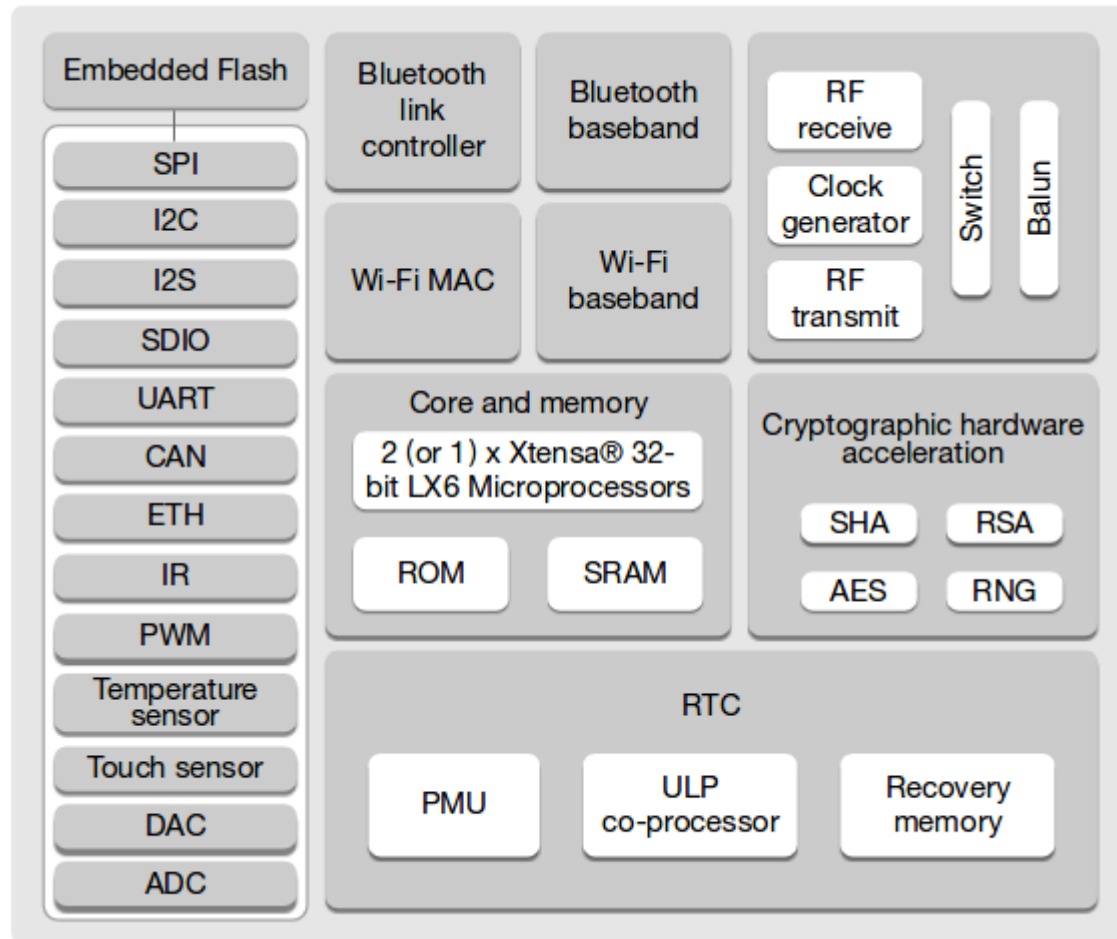


Stromsparen mit ULP

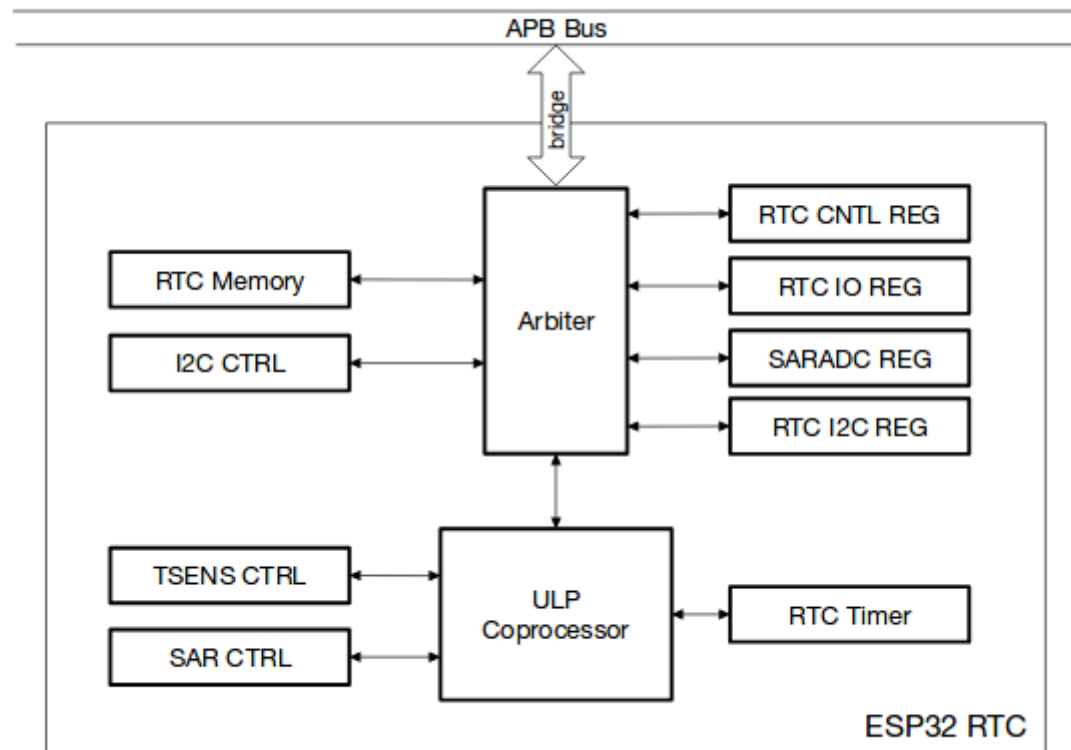
Stromverbrauch: recht hoch

Mode	Min	Typ	Max	Unit
Transmit 802.11b, DSSS 1 Mbps, POUT = +19.5 dBm	-	240	-	mA
Transmit 802.11b, OFDM 54 Mbps, POUT = +16 dBm	-	190	-	mA
Transmit 802.11g, OFDM MCS7, POUT = +14 dBm	-	180	-	mA
Receive 802.11b/g/n	-	95 ~ 100	-	mA
Transmit BT/BLE, POUT = 0 dBm	-	130	-	mA
Receive BT/BLE	-	95 ~ 100	-	mA

RealTime-Modul



Dedizierter Coprozessor



Was ist wichtig

- ULP teilt einen kleinen Speicherbereich
 - Auch “ein paar” GPIOs
- ACHTUNG: 16 vs 32 bit
- Programmierung in ASSEMBLER
 - Vier “Arbeitsregister” R0...R3

main/component.mk

```
ULP_APP_NAME ?= ulp_$(COMPONENT_NAME)
ULP_S_SOURCES = $(COMPONENT_PATH)/ulp/ulp_source_file.S
ULP_EXP_DEP_OBJECTS := main.o
include $(IDF_PATH)/components/ulp/component_ulp_common.mk
```

ulp/ulp_source_file.S

```
#include "soc/rtc_cntl_reg.h"
#include "soc/rtc_io_reg.h"
#include "soc/soc_ulp.h"

.bss

    .global acti_count
acti_count:
    .long 0

    /* Code goes into .text section */
    .text
    .global entry
entry:
```

```
gpio_num_t cpu_num = GPIO_NUM_25;
gpio_num_t ulp_num = GPIO_NUM_4; // RTC GPIO 10

extern const uint8_t ulp_main_bin_start[] asm("_binary_ulp_main_bin_start");
extern const uint8_t ulp_main_bin_end[] asm("_binary_ulp_main_bin_end");
void init_ulp_program()
{
    esp_err_t err = ulp_load_binary(0, ulp_main_bin_start,
    | (ulp_main_bin_end - ulp_main_bin_start) / sizeof(uint32_t));
    ESP_ERROR_CHECK(err);

    ulp_acti_count=0;
    rtc_gpio_init(cpu_num);
    rtc_gpio_set_direction(cpu_num, RTC_GPIO_MODE_OUTPUT_ONLY);
    rtc_gpio_set_level(cpu_num, 1);

    rtc_gpio_init(ulp_num);
    rtc_gpio_set_direction(ulp_num, RTC_GPIO_MODE_OUTPUT_ONLY);
    rtc_gpio_set_level(ulp_num, 1);
}
```


Lasst uns rebooten!!!

```
void app_main() {  
    //Init checken  
    esp_sleep_wakeup_cause_t cause = esp_sleep_get_wakeup_cause();  
    if (cause != ESP_SLEEP_WAKEUP_ULP) {  
        printf("Erststart\n");  
        init_ulp_program();  
    } else {  
        printf("ULP-verursachter Start\n");  
        printf("Counter %u \n", ulp_acti_count & UINT16_MAX);  
    }  
}
```

Lasst uns einschlafen!!!

```
ESP_ERROR_CHECK( esp_sleep_enable_ulp_wakeup() );  
ESP_ERROR_CHECK( ulp_run((&ulp_entry - RTC_SLOW_MEM) / sizeof(uint32_t)));  
esp_deep_sleep_start();  
}
```

Achtung vor “Gleichlauf”

```
/* Check if the SoC has said INRI already */
READ_RTC_REG(RTC_CNTL_DIAG0_REG, 19, 1)
and r0, r0, 1
jump exit, eq
wake
WRITE_RTC_FIELD(RTC_CNTL_STATE0_REG, RTC_CNTL_ULP_CP_SLP_TIMER_EN, 0)
halt
```



vs klassischen Arduino

+

- Mehr Leistung
- Preiswerter

-

- Einstieg schwieriger
- Board-Design
- RT-Eignung vs AVR8
- Stromverbrauch

vs OrangePi

+

- Kompakter
- “Offener”
- Stromverbrauch

-

- Kein Linux
- Weniger Speicher
- Nicht so viel teurer

Wann ESP32

- Wenn
 - Echtzeitanforderungen moderat
 - WLAN erforderlich
 - Kosten gering sein
- Wenn nicht
 - Linux erforderlich
 - Stromverbrauch extrem niedrig

Thank you!

?!?! → tamhan@tamoggemon.com



Instagram

Search

Get the app

Sign up | Log in



tam.hanna

Follow

368 posts

7,098 followers

4,281 following

Crazy Electronics Lab Random snapshots from my #electronics lab. I love to help - questions to tamhan@tamoggemon.com please, as I sometimes miss comments!
shop.heise.de/katalog/ix-tutorial-cross-plattform-entwicklung-mit-qt

