



Střední průmyslová škola a Vyšší odborná škola, Písek, Karla Čapka 402, Písek

18-20-M/01 Informační technologie

## Maturitní práce

# Dálkové ovládání zásuvek NETIO

Téma číslo 12

autor:

**Milan Jiříček, B4.I**

vedoucí maturitní práce:

**Ing. Břetislav Bakala**

Písek 2020/2021

## **Anotace**

Maturitní práce se zaměřuje na porovnání platforem ESP8266 a ESP32. Cílem je vytvořit ovladač pro ovládání zásuvek značky NETIO s webovou aplikací pro konfiguraci a zjistit, která platforma je vhodná pro realizaci funkčního vzorku z hlediska spotřeby energie a reakční doby.

## **Annotation**

The graduation thesis focuses on the comparison of the ESP8266 and ESP32 platforms. The goal is to create a driver for controlling NETIO sockets with a web application for configuration and to find out which platform is suitable for the implementation of a functional sample in terms of energy consumption and response time.

## Poděkování

Chtěl bych poděkovat panu učiteli Ing. Břetislavovi Bakalovi za odborné vedení práce a cenné rady, které mi pomohly tuto práci zkompletovat. Rád bych také poděkoval technickému řediteli Ing. Břetislavovi Bakalovi ml. společnosti NETIO products a.s. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování bakalářské práce. V neposlední řadě chci poděkovat Mgr. Haně Maříkové a Mgr. Vladimíře Špirhanzlové za pomoc při gramatické a stylistické kontrole.

# Obsah

<b>1</b>	<b>Úvod</b>	<b>5</b>
<b>2</b>	<b>Teoretický úvod do problematiky</b>	<b>6</b>
2.1	Zásuvka NETIO . . . . .	6
2.2	Platforma ESP . . . . .	6
2.2.1	ESP8266 . . . . .	6
2.2.2	ESP32 . . . . .	8
2.3	Komunikace mezi ESP a NETIO zásuvkou . . . . .	8
<b>3</b>	<b>Metodika postupu</b>	<b>10</b>
3.1	Tvorba rozhraní . . . . .	10
3.2	Měření připojení k WiFi a odeslání HTTP requestu . . . . .	10
3.2.1	Spotřeba . . . . .	11
3.2.2	Reakční čas . . . . .	11
3.3	Ustálené stavy . . . . .	11
3.3.1	Spotřeba . . . . .	12
3.3.2	Probuzení z ustálených časů . . . . .	12
<b>4</b>	<b>Měření spotřeby a času</b>	<b>14</b>
4.1	ESP8266 . . . . .	14
4.1.1	Spotřeba a reakční doba ustálených stavů . . . . .	14
4.1.2	Připojení k WiFi a odeslání HTTP requestu . . . . .	16
4.2	ESP32 . . . . .	18
4.2.1	Spotřeba ustálených stavů . . . . .	18
4.2.2	Reakční čas jednotlivých situací . . . . .	18
4.3	Porovnání získaných výsledků . . . . .	19
4.3.1	Reakční časy stavů . . . . .	19
<b>5</b>	<b>Vytvoření funkčních vzorků</b>	<b>21</b>

<b>6 Závěr</b>	<b>22</b>
<b>Přílohy</b>	<b>24</b>
<b>A Příloha</b>	<b>25</b>

# Kapitola 1

## Úvod

# Kapitola 2

## Teoretický úvod do problematiky

### 2.1 Zásuvka NETIO

### 2.2 Platforma ESP

ESP jsou rodina mikročipů od společnosti **Espressif Systems**.

#### 2.2.1 ESP8266

##### Historie

ESP8266 je levný mikročip, který umí využívat WiFi. První chip, který se dostal na světlo světa byl v modulu **ESP-01**. Tento modul dokázal připojit se na WiFi síť a provádět jednoduché TCP/IP spojení. Získal si velkou oblibu díky nízké ceně. Jsou vhodné pro IoT jako například automatizace, zabezpečení, chytré domy atd.

##### Specifikace

Pro tuto maturitní práci bude použit modul **WT8266-S1**, který je vytvořen společností **Wireless-Tag**. Je založen na mikročipu ESP8266.

ESP8266 integruje vylepšenou verzi procesoru **L106 Diamond series 32-bit** vytvořený firmou **Tensilica** s podporou frekvencí 80 MHz a 160 MHz a RTOS<sup>1</sup>. K dispozici je 36 KB RAM a 16 Mbit Flash paměti. Integrovaný v systému je také 10 bitový analog-digitální převodník. Modul podporuje standard **IEEE802.11 b/g/n**<sup>2</sup> a sadu protokolů TCP/IP. WiFi 2,4 GHz umožňuje WPA/WPA2<sup>3</sup>, rychlost až 72,2 Mbps a ke komunikaci využívá

---

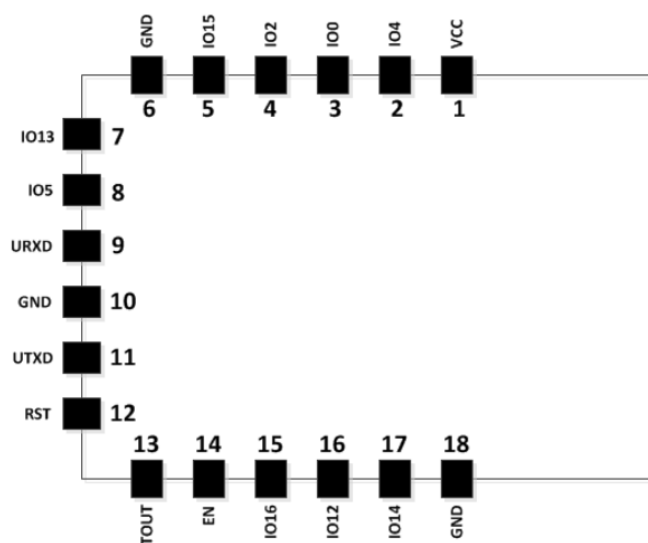
<sup>1</sup>Operační systém v reálném čase

<sup>2</sup>Standard pro lokální bezdrátové sítě

<sup>3</sup>Chráněný přístup k WiFi

anténu PCB. Zařízení má 16 GPIO pinů<sup>4</sup> (viz. obr. 2.1). Také obsahuje:

- **UART** - univerzální asynchronní přijímač-vysílač pro sériový přenos
- **I<sup>2</sup>C** - seriová sběrnice o dvou vodičích pro nízkorychlostní zařízení
- **I<sup>2</sup>S** - seriová sběrnice pro digitální audio
- **SPI** - seriové periferní rozhraní pro komunikaci mezi mikroprocesorem a integrovanými obvody
- **rozhraní HSPI** - externí SPI pro připojení displeje či externí Flash
- **PWM rozhraní** - 4 kanálová pulzně šířková modulace pro přenos analogového signálu pomocí dvouhodnotného signálu



Obrázek 2.1: ESP8266 pinout

## Deep sleep

Platforma ESP lze také uvést do úsporného režimu. Jednoduše to znamená vypnutí nejdůležitějších částí pro ustálený stav. ESP8266 celkově nabízí 3 druhy úsporných režimů.

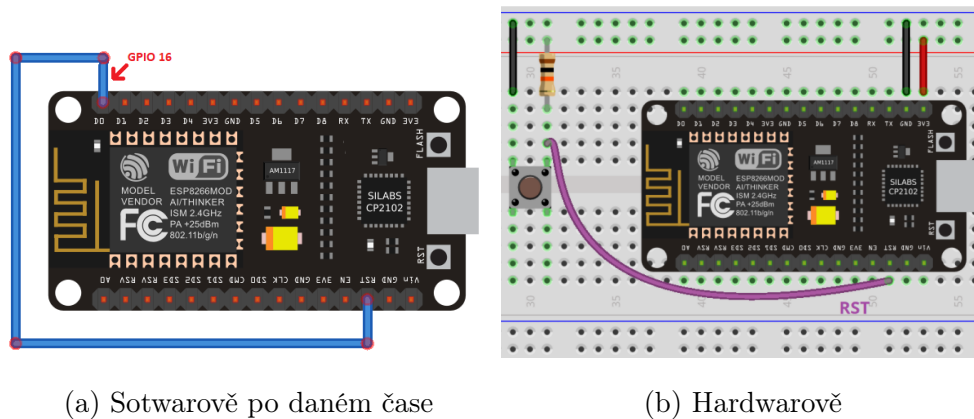
---

<sup>4</sup>Vstupně výstupní pin



Modem, light a deep. Pro moji maturitní práci použiji deep sleep, který je nejúspornějších. Kromě **RTC**<sup>5</sup> se vše vypne, včetně CPU či WiFi. V tomto režimu je také možné uchovat data v tzv. RTC paměti, která jsou dostupná i po obnovení. Průměrná spotřeba je 20  $\mu\text{A}$  a je vhodný pro jakýkoliv projekt na akumulátor či baterii.

Probuzení ESP probíhá, buď po nastaveném čase, kde je nutné připojit pin **GPIO 16** tzv. wake-up pin na **RESET** (viz. obr. 2.2a), nebo můžeme na **RESET** přivést krátce tlačítkem logickou nulu a obnovíme ESP hardwarově (viz. obr. 2.2b).



Obrázek 2.2: ESP8266 probuzení z deep sleep po daném čase

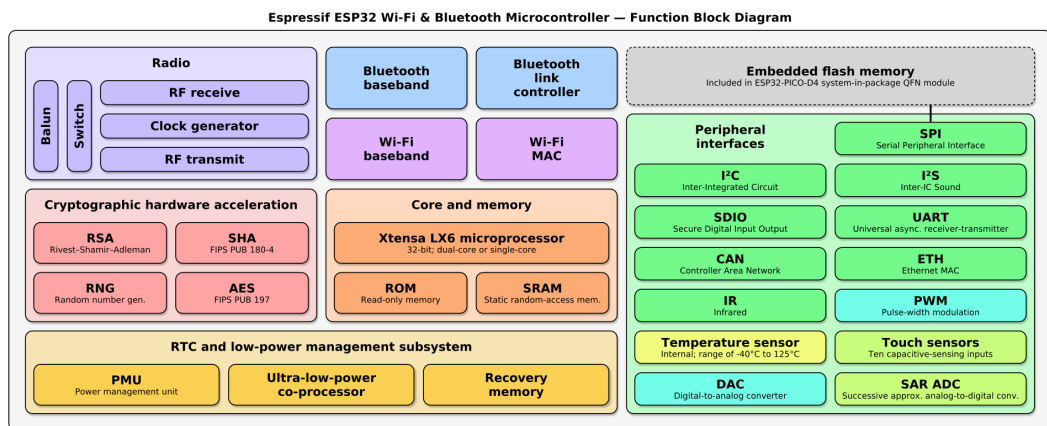
## 2.2.2 ESP32

ESP32 je starší model z řady ESP, který byl vydán v roce 2016. Tento mikročip má integrované WiFi i Bluetooth. Série ESP32 obsahuje mikroprocesor Tensilica Xtensa LX6 v dvou jádrové či jednojádrové verzi, který může operovat na 160 MHz nebo 240 MHz. SRAM je zde byla zvětšena na 520 kB. WiFi může dosáhnout rychlosti až 150 Mbps společně se zabezpečením WPA/WPA2. Kromě IPv4 je možnost použít i IPv6. ESP32 integruje 12 bitový analog-digitální převodník. Oproti ESP8266 je do tohoto modelu zabudován senzor teploty. ESP32 disponuje 36 GPIO piny. Další funkce jsou na obrázku 2.3.

## 2.3 Komunikace mezi ESP a NETIO zásuvkou

---

<sup>5</sup>Hodiny reálného času



Obrázek 2.3: ESP32 blokový diagram funkcí

# Kapitola 3

## Metodika postupu

### 3.1 Tvorba rozhraní

### 3.2 Měření připojení k WiFi a odeslání HTTP requestu

Cílem měření je zjištění rychlostí a spotřeby připojení různými způsoby k přístupovému bodu a následné porovnání případů. Do měření je také započítané odesílání HTTP requestu zásuvce. Jsou vytvořeny celkem 4 případy akcí:

- **statická IP adresa** - Zařízení dostane IP adresu, masku a bránu staticky staticky nakonfigurovanou. Na access pointu bude vyplé DHCP a komunikace nebyla zabezpečena.
- **dynamická IP adresa** - Bude využit DHCP protokol, kde zařízení požádá DHCP server o IP adresu, kterou mu access point přidělí společně s bránou, maskou a s časem, kdy tato adresa platí. Při měření nebyl přístupový bod zabezpečen.
- **Zabezpečené připojení** - Připojení na access point bude šifrované pomocí WPA2 - PSK. DHCP server bude zapnut. Toto je simulace klasického uživatelského používání.
- **Odeslání HTTP requestu** - Zařízení je připojené k WiFi. Odešle zprávu zásuvce a pokud dostane zpětnou vazbu od zásuvky, ukončí se činnost.

Ve všech scénářích bude měřeno zařízením **ANALOG DISCOVERY 2** od Digilent. Frekvence procesoru je 160 MHz a AP je vzdálen 3,5 m od zásuvky a funkčního vzorku ovladače.

### 3.2.1 Spotřeba

Měření bude úbytek napětí na bočníku<sup>1</sup> o velikosti určené u měření. V programu k měřicímu zařízení je možné stejně jako na osciloskopu vytvořit graf naměřených hodnot v dané vzorkovací frekvenci. Hodnoty napětí bude převedeno na hodnoty el. proudu pomocí Ohmova zákona  $I = \frac{U}{R}$ . Amplitudy el. proudu budou sečteny a vyděleny počtem vzorků [DOPLNIT VZOREC]. Tato operace nám poskytne zjištění průměrného el. proudu ve scénáři. Spotřebu již vypočítáme dle:

$$E = U \times \bar{I} \times \frac{t}{3600}$$

Čas  $t$  v sekundách je možné zjistit v následující kapitole.

### 3.2.2 Reakční čas

Z naměřeného případu bude vyjmuta klidová část a rozdíl mezi posledním a prvním vzorkem je poté reakční čas. V případě HTTP requestu nebude počítána část konfigurace WiFi. Společně s měřením byl na **serial monitor** odesílány zprávy, kde se program nachází. Poté se vyjmula část, která dle serial monitoru nepatří do scénáře.

## 3.3 Ustálené stavy

Pro ustálené stavy byly navrženy jednoduché programy pro co nejlepší porovnání. Ustálený stav je stav, kdy zařízení čeká na uživatelský podnět jako např. stisknutí tlačítka. Ideální ustálený stav musí splňovat:

1. nejkratší možnou reakční dobu
2. nejnižší možnou spotřebu pro co nejdelší výdrž baterií

Reakční doba musí být v rámci uživatelské přívětivosti tzn. probuzení z tohoto stavu nesmí trvat více jak 1,5 s. Zároveň se zařízení nesmí vypnout zdůvodu vybité baterie za krátkou dobu. Porovnání proběhne mezi třemi scénáři, které mají jiné vlastnosti. Vybrány byly následující:

---

<sup>1</sup>nízkoohmový rezistor

- **Kontinuální režim** - V tomto stanu není žádná funkce ESP vypnuta.
- **Deep sleep** - ESP vypne vše až na RTC (kapitola viz. 2.2.1 na straně 7).
- **Enable button** - ESP je vypnuto pomocí přivedené GROUND na pin ENABLE.

### 3.3.1 Spotřeba

Cílem měření je nalézt stav s nejmenší spotřebou energie, jelikož finální prvek bude napájen z baterií je důležité co nejdelší výdrž. Programy, které byly napsány pro měření:

- **Kontinuální režim** - Zařízení má nastavený určitý GPIO na tlačítko a monitoruje zda bylo zmáčknuto. ESP také má otevřenou sériovou komunikaci.
- **Deep sleep** - Program zajišťuje probuzení ESP každých 5 s a odesílání zprávy na serial monitoring.
- **Enable button** - Pro měření této situace není třeba specifický program

Pro měření bude použit **ANALOG DISCOVERY** s jeho funkcí osciloskopu. Na napájení bude přidán bočník o velikosti  $0,7 \Omega$ ,  $10,5 \Omega$  nebo  $4,7 \Omega$ <sup>2</sup>. Pomocí Ohmova zákona  $I = \frac{U}{R}$  je možné zjistit el. proud. Příkon, který bude porovnán vypočítáme dle vztahu:

$$P = U \times I$$

## OBRÁZKY ZAPOJENÍ

### 3.3.2 Probuzení z ustálených časů

Ovladač, který je již připojen k WiFi, po stisknutí tlačítka odešle HTTP request zásuvce, která zareaguje přepnutím stavu a odesláním zprávy zpět ovladači zda se povedlo či nikoliv.

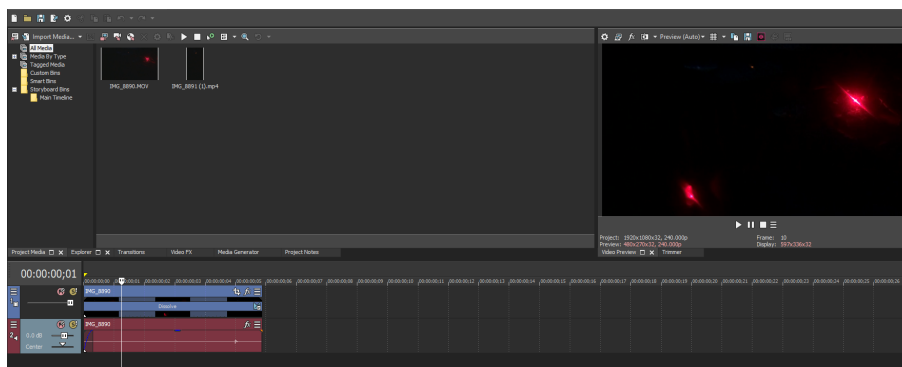
Reakční doba byla změřena pomocí kamery. K tlačítku jsem připojil LED, místnost jsem izoloval od světla a zmáčknutí tlačítka a reakci zásuvky jsem natočil ve zpomaleném režimu s **240 snímky za sekundu**. Pro upravení videa jsem použil trial verzi programu

---

<sup>2</sup>Je nutné zvolit takový odpor, aby byl úbytek napětí měřitelný.

**Sony Vegas** (viz. obr. 3.1). Našel jsem rozsvícení LED tlačítka a rozsvícení LED zásuvky ve videu a ustríhl jsem tento úsek od zbytku videa. Program poskytuje zobrazení počtu snímku daného úseku. Reakční čas je následovně možný zjistit pomocí:

$$t = \frac{\text{Počet snímků úseku}}{\text{Počet snímků za sekundu}}$$



Obrázek 3.1: Ukázka postupu pro měření reakčních časů

# Kapitola 4

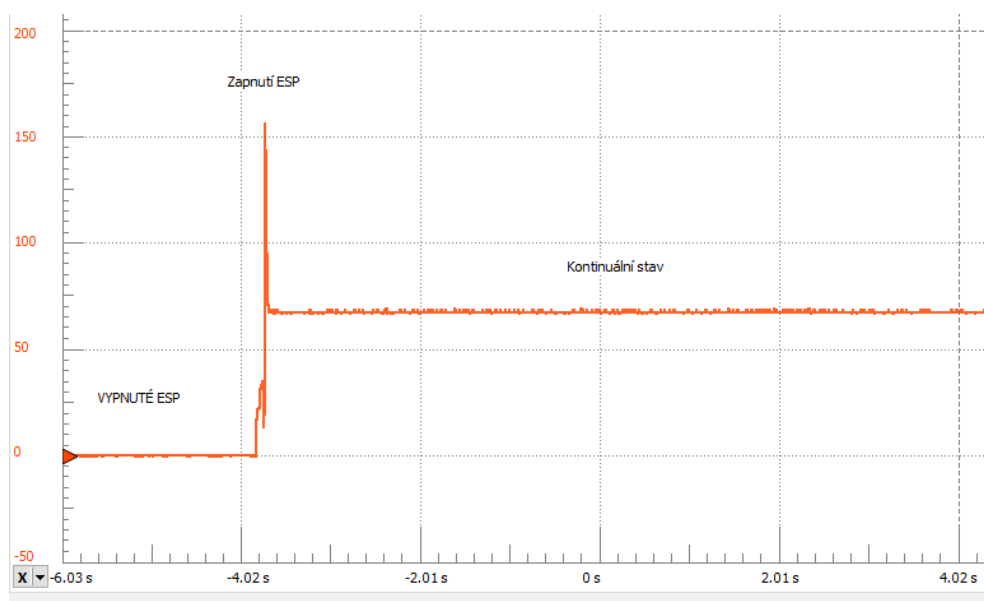
## Měření spotřeby a času

### 4.1 ESP8266

#### 4.1.1 Spotřeba a reakční doba ustálených stavů

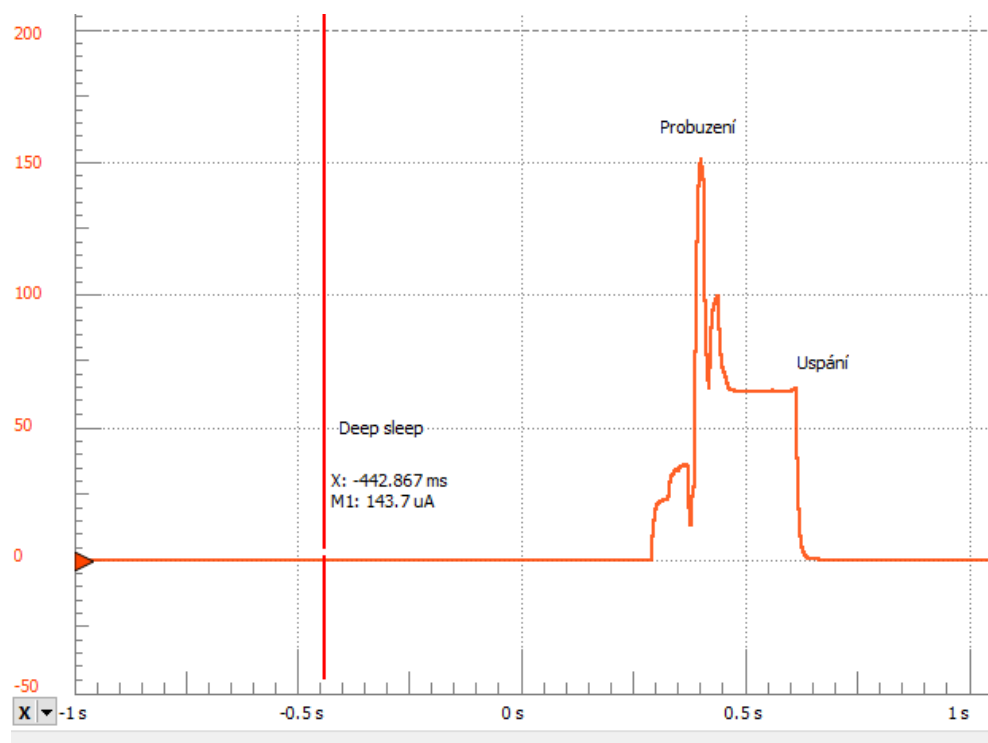
##### Spotřeba

Spotřeba jednotlivých scénářů je měřena dle metodiky 3.3.1 na straně 12. Bočník pro měření úbytku napětí byl o velikosti  $4,7\ \Omega$ .



Obrázek 4.1: ESP8266 graf kontinuálního ustáleného stavu

Situace **enable button** nebyla možná změřit zařízením **ANALOG DISCOVERY 2** ani na bočníku o velikosti  $0,7\ \Omega$  kvůli velice nízkému úbytku napětí, který se v datasheetu WT8266 pohybuje v jednotkách  $\mu\text{A}$ . V tomto případě pro výpočty je použita hodnota,



Obrázek 4.2: ESP8266 graf deep sleep ustáleného stavu

kterou určil výrobce v datasheetu.

Tabulka 4.1: ESP8266 porovnání spotřeby ustálených stavů

Ustálený stav	U(V)	I(A)	P(W)
Kontinuální	3,3	$67,66 \times 10^{-3}$	$223,3 \times 10^{-3}$
Deep sleep	3,3	$138,20 \times 10^{-6}$	$460,0 \times 10^{-6}$
Enable button	3,3	$3,00 \times 10^{-6}$	$9,9 \times 10^{-6}$

Dle tabulky 4.1 je možné vidět jednotlivé situace. Největším příkonem disponuje kontinuální ustálený stav z důvodu běžícího programu na kontrolu interakce. Nejmenší spotřeba je naopak vypnuté ESP přes enable pin.

## Reakční čas

Reakční čas byl měřen dle metodiky 3.3.2 na straně 12. Nejrychlejší ustálený stav je kontinuální, kvůli připravenosti na interakci. Nejdéle trvá vypnuté ESP, protože je nutné ESP zapnout, nastavit konfiguraci a připojit na WiFi, zatím co zapnuté ESP je již připraveno.



Tabulka 4.2: ESP8266 porovnání reakčního času ustálených stavů

Ustálený stav	t(ms)
Kontinuální	<b>196</b>
Deep sleep	<b>983</b>
Enable button	<b>3 208</b>

Tabulka 4.3: ESP8266 porovnání ustálených stavů

Ustálený stav	P[W]	$t_p[h]$	$t[ms]$
Kontinuální	$223,3 \times 10^{-3}$	75,6	196
Deep sleep	$460,0 \times 10^{-6}$	36 993,1	983
Enable button	$9,9 \times 10^{-6}$	1 704 545,0	3 208

## Porovnání

Pro porovnání spotřeb budeme předpokládat, že zařízení je napájeno  $3 \times$  NiZn akumulátorem typu AA, které společně mají kapacitu **3750 mAh**, což je možné převést na energii vztahem  $E = U_{aku} \times \text{kapacita}$ . a dále vztahem<sup>1</sup>  $t_p = \frac{E_{bat}}{P}$  zjistit provozní čas (viz. tabulka 4.3). Zařízení není schopno dosáhnout této výdrže protože pokud napětí klesne pod 3 V, stabilizátor napájení odpojí.

Kontinuální stav, zatímco má výdrž v rámci několika dní, reakční čas trvá necelých 200 ms. Enable button situace naprosto opačná. Reakční čas je pro uživatele dlouhý, ale výdrž v tomto stavu bez interakce by se pohybovala v letech. Dobrý kompromis je situace deep sleep, kdy zařízení reagovalo vždy do jedné sekundy a teoretická výdrž je v jednotkách let.

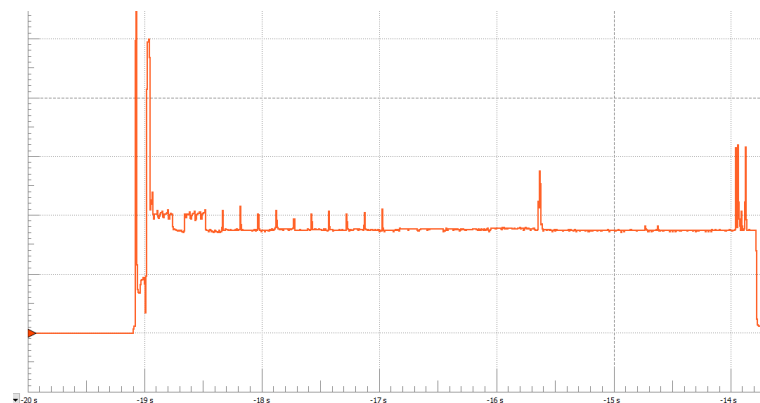
### 4.1.2 Připojení k WiFi a odeslání HTTP requestu

Měření proběhlo dle metodiky 3.2 na straně 10 na bočníku 0,7  $\Omega$ .

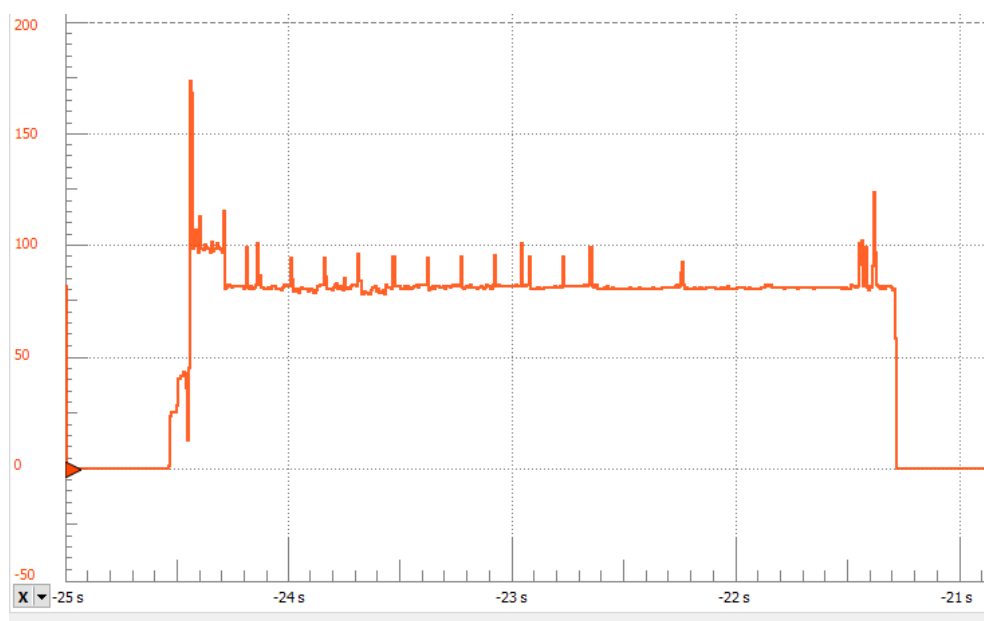
Z tabulky 4.4 je možné vidět, že **příkon P** je téměř totožný v každé operaci. Největší rozdíl, který ovlivňuje spotřebu, je čas, za který se úkon vykoná. Jelikož **připojení**

---

<sup>1</sup> $t_p$  - provoz v ustáleném režimu,  $E_{bat}$  - energie  $3 \times$  AA akumulátorů

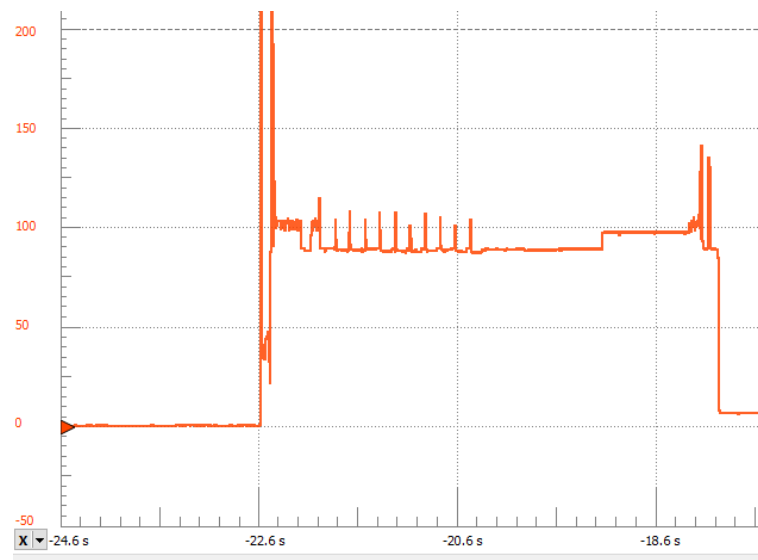


Obrázek 4.3: ESP8266 graf WiFi připojení s dynamickou IP



Obrázek 4.4: ESP8266 graf WiFi připojení se statickou IP

**pomocí statické IP adresy** trvalo nejkratší dobu, má také nejmenší spotřebu energie. Samotná **kommunikace HTTP** má nejnižší spotřebu ze všech definovaných operací a jeden cyklus operace spotřebuje zanedbatelné množství energie.



Obrázek 4.5: ESP8266 graf připojení na zabezpečenou WiFi

Tabulka 4.4: ESP8266 Spotřeba operací

Operace	U(V)	I[mA]	t[ms]	P[mW]	E[μWh]
Dynamické připojení	3,3	89,5	5 300	295	<b>435</b>
Statické připojení	3,3	83,0	3 143	274	<b>239</b>
Zabezpečené připojení	3,3	90,9	4 606	300	<b>383</b>
HTTP komunikace	3,3	84,8	732	280	<b>57</b>

## 4.2 ESP32

### 4.2.1 Spotřeba ustálených stavů

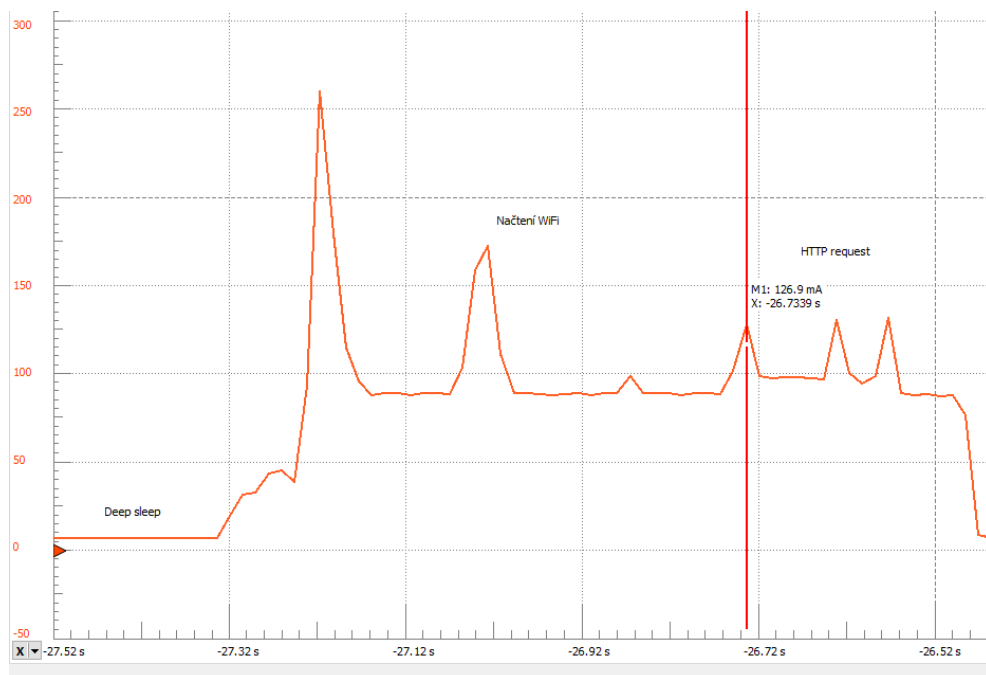
Kontinuální

Enable

Deep sleep

### 4.2.2 Reakční čas jednotlivých situací

Získání reakčních časů jednotlivých možností zapojení ESP32 proběhlo stejným způsobem jako u ESP8266 (viz. obr. 3.1 na straně 13) tzn. Náhrál jsem reakci na kameru v 240



Obrázek 4.6: ESP8266 graf HTTP requestu

snímcích za sekundu, vystříhl jsem úsek, kde zásuvka reaguje, a převedl jsem snímky na čas.

Tabulka 4.5: ESP32 porovnání reakčního času jednotlivých situací

Ustálený stav	t(ms)
Kontinuální	<b>188</b>
Enable	<b>2 333</b>
Deep sleep	<b>1 071</b>

## 4.3 Porovnání získaných výsledků

### 4.3.1 Reakční časy stavů

Z tabulky 4.6 je možné vidět jak si platformy vedly proti sobě.

Pokud ovladač neustále běží s připojenou WiFi čas reakce je velmi rychlý a v obou případech uživatel nepostřehne prodlevu mezi aktivací ovladače a sepnutí relé v zásuvce. Časy mezi platformami se liší minimálně a rozdíl se pohybuje v rámci milisekund, přesto je

ESP32 lehce rychlejší.

Zapnutí vypnutého ESP, následné načtení a připojení k WiFi je nejdelší ze všech scénářů, který trvá několik sekund. Pro uživatele je velice pomalé a v reálném produktu dle mého názoru zcela nepoužitelné. I přesto je ESP32 je mnohem rychlejší a to zhruba o 1 s.

Deep sleep je alternativa mezi kontinuálním a vypnutým stavem. ESP totiž vypne většinu svých částí. Díky tomu je schopno relativně rychle zareagovat na budící signál. Reakce obou ovladačů je cirka 1 s. Zde vede ESP8266, které je zhruba o 100 ms pomalejší.

Tabulka 4.6: Porovnání reakčních časů ustálených stavů platforem

Ustálený stav	ESP8266 (ms)	ESP32 (ms)
Kontinuální	<b>196</b>	<b>188</b>
Enable	<b>3 208</b>	<b>2 333</b>
Deep sleep	<b>983</b>	<b>1 071</b>

# Kapitola 5

## Vytvoření funkčních vzorků

# Kapitola 6

## Závěr

# Seznam tabulek

4.1	ESP8266 porovnání spotřeby ustálených stavů . . . . .	15
4.2	ESP8266 porovnání reakčního času ustálených stavů . . . . .	16
4.3	ESP8266 porovnání ustálených stavů . . . . .	16
4.4	ESP8266 Spotřeba operací . . . . .	18
4.5	ESP32 porovnání reakčního času jednotlivých situací . . . . .	19
4.6	Porovnání reakčních časů ustálených stavů platform . . . . .	20



# Seznam obrázků

2.1	ESP8266 pinout . . . . .	7
2.2	ESP8266 probuzení z deep sleep po daném čase . . . . .	8
2.3	ESP32 blokový diagram funkcí . . . . .	9
3.1	Ukázka postupu pro měření reakčních časů . . . . .	13
4.1	ESP8266 graf kontinuálního ustáleného stavu . . . . .	14
4.2	ESP8266 graf deep sleep ustáleného stavu . . . . .	15
4.3	ESP8266 graf WiFi připojení s dynamickou IP . . . . .	17
4.4	ESP8266 graf WiFi připojení se statickou IP . . . . .	17
4.5	ESP8266 graf připojení na zabezpečenou WiFi . . . . .	18
4.6	ESP8266 graf HTTP requestu . . . . .	19

Příloha A

Příloha

# Literatura

- [1] ESP8266. *Wireless-Tag Technology Co., Limited* [Online]. Irvine (CA): Wireless-Tag, 2021 [cit.2021-02-17]. Dostupné z: <https://www.tme.com/Document/12266bdd8a30eeb152305461b089a151/WT8266-S1.pdf>
- [2] DHCP PROTOKOL. In: *Wikipedia: the free encyclopedia* [Online]. San Francisco (CA): Wikimedia Foundation, 2021 [cit.2021-02-17]. Dostupné z: [https://en.wikipedia.org/wiki/Dynamic\\_Host\\_Configuration\\_Protocol](https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol)