Diplomová práce

České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra mikroelektroniky

Systém pro automatizaci s rozpoznáváním gest

Bc. Petr Vilím

Vedoucí práce: prof. Ing. Miroslav Husák, CSc.
Obor: Elektronika
Specializace: Elektronika a komunikace
Srpen 2020

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Vilím** Jméno: **Petr** Osobní číslo: **393020**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra mikroelektroniky**

Studijní program: **Elektronika a komunikace**

Specializace: **Elektronika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Systém pro automatizaci s rozpoznáváním gest

Název diplomové práce anglicky:

System for Automation with Gesture Recognition

Pokyny pro vypracování:

1. Proveďte analýzu současného stavu využití integrovaných senzorů pro rozpoznávání gest, bezdrátových komunikačních rozhraní, mikrokontrolerů a embedded Linux platforem. Zvažte, které z těchto komponent bude pro danou aplikaci vhodné použít z hlediska provozních parametrů.
2. Navrhněte systém pro automatizaci s rozpoznáváním gest, který se bude skládat ze tří hlavních komponent – ovládací rozhraní schopné rozpoznávat základní gesta, řídicí část zpracovávající data a akční člen pro ovládání např. osvětlení. Vzájemná komunikace všech komponent bude zajištěna vhodným bezdrátovým rozhraním. Realizujte jednoduchý model navrženého systému.
3. Zjistěte základní parametry realizovaného systému, navrhněte možné úpravy pro vylepšení parametrů. Zhodnoťte finanční stránku realizace daného systému.

Seznam doporučené literatury:

[1] HOROWITZ et al. The Art of Electronics. Harvard University, 2015. ISBN 978-0-521-80926-9
[2] VOBECKÝ, V. a ZÁHLAVA, V. Elektronika - Součástky a obvody, principy a příklady. Grada 2005. ISBN 978-80-247-1241-3
[3] ZÁHLAVA, V. Návrh a konstrukce desek plošných spojů. BEN 2011. ISBN 978-80-7300-266-4.

Jméno a pracoviště vedoucí(ho) diplomové práce:

prof. Ing. Miroslav Husák, CSc., katedra mikroelektroniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **17.02.2020** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce: **19.02.2022**

prof. Ing. Miroslav Husák, CSc. podpis vedoucí(ho) práce	prof. Ing. Pavel Hazdra, CSc. podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

22. 2. 2020
Datum převzetí zadání

Podpis studenta

Poděkování

Děkuji svému vedoucímu za trpělivost a všem kteří mě podporovali.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 11. srpna 2020

.....

Abstrakt

Cílem této práce je analýza současného stavu využití integrovaných senzorů pro rozpoznávání gest, jejich provozních parametrů a funkcí, dále pak analýza současného stavu využití bezdrátových komunikačních rozhraní, jejich provozních parametrů a funkcí, a analýza současného stavu využití mikrokontrolerů a embedded Linux platforem. Dále je cílem této práce návrh systému pro automatizaci s rozpoznáváním gest, který se bude skládat ze tří hlavních komponent – ovládací rozhraní schopné rozpoznávat základní gesta, řídící centrála zpracovávající data a aktivní člen, kterým může být například ovládání osvětlení. Všechny zmíněné komponenty mezi sebou budou komunikovat pomocí vhodného bezdrátového rozhraní. Závěrečným cílem této práce fyzická modelová realizace tohoto systému a zhodnocení jeho parametrů po funkční a finanční stránce.

Klíčová slova: Senzory pro rozpoznávání gest, automatizace, vestavěné systémy

Vedoucí práce: prof. Ing. Miroslav Husák, CSc.
ČVUT FEL,
Technická 2,
166 27 Praha 6 - Dejvice

Abstract

First goal of this thesis is a research of available sensors for gesture recognition with their operational parameters and functions, a research of wireless communication interfaces with their operational parameters and functions and a research of microcontrollers and embedded Linux platforms. Second goal is a design of system for automation with gesture recognition consisting of: control interface with gesture recognition capability, main control unit and active device controlling for example room lights. All previously mentioned component will communicate using appropriate wireless communication protocol. Final goal of this thesis is a physical realization of a model system and analysis of its properties and design costs.

Keywords: Sensors for gesture recognition, automatization, embedded systems

Obsah

1 Úvod 1
1.1 Aktuální trendy 1
1.2 Předcházející práce 1

2 Analýza dostupných komponent 3
2.1 Mikrokontroléry 3
2.1.1 8-bitové architektury 3
2.1.2 16-bitové architektury 4
2.1.3 32-bitové 4
2.1.4 Programovací jazyky 5
2.2 Bezdrátová komunikace 5
2.2.1 Systémy na čipu 6
2.2.2 Komunikační moduly 6
2.3 Embedded Linux platformy 8
2.4 Senzory pro rozpoznávání gest . . . 9

3 Výsledná podoba systému 11
3.1 Testování APDS9960 12
3.2 Testování ESP32 13

4 Hardware 15
4.1 Bloková schémata 15
4.1.1 Sense 15
4.1.2 Control 16
4.2 Funkční schémata 16
4.2.1 Sense 17
4.2.2 Control 17
4.3 Návrh DPS 18
4.4 Osazení a zprovoznění 18

5 Software 21
5.1 Celková struktura systému 22
5.2 Komunikační rámec 23
5.3 Control 23
5.4 Server 25
5.5 Sense 26
5.5.1 Rozpoznávání gest 26

6 Testování parametrů systému 29
6.1 Odezva systému 29
6.2 Ovládání 30

7 Finanční stránka realizace 31

8 Závěr 33
8.1 Zhodnocení výsledků 33
8.2 Potenciální vylepšení 33

Literatura 35

Seznam zkratek a symbolů 39

Přílohy 41
Příloha A1 - funkční schéma Sense . 41
Příloha A2 - funkční schéma Control 47
Příloha B1 - rozložení DPS Sense . . 52
Příloha B1 - rozložení DPS Control 54
Příloha C - zdrojové kódy 56

Obrázky

1.1 Schéma navrhovaného systému .. 2

2.1 Bezdrátové komunikační moduly . 8
2.2 Jednodeskové počítače 9

3.1 Testovací přípravek APDS-9960 12
3.2 Testovací přípravek ESP32 13

4.1 Blokové schéma SENSE 16
4.2 Blokové schéma CONTROL 16
4.3 Osazené DPS 19

5.1 Blokový diagram celého systému 22
5.2 Komunikační rámec 23
5.3 Blokový diagram CONTROL ... 24
5.4 Blokový diagram SERVER 25
5.5 Blokový diagram SENSE 27

Tabulky

3.1 Navrhované varianty 11

6.1 Měření ping a ACK 29
6.2 Měření chybovosti 30

7.1 Cenová kalkulace 31

Kapitola 1
Úvod

1.1 Aktuální trendy

V současné době je nadále velkým tématem internet věcí a lze předpokládat i rozsáhlý budoucí rozvoj. Aktuálně mimo jiné dochází k prvním implementacím 5G sítí i v České Republice a již se běžně můžeme setkat s rozsáhlými systémy na bázi NB-IoT a WB-IoT, tedy úzkopásmových a širokopásmových zařízení internetu věcí. Rozšiřuje se též pokrytí datovými sítěmi obecně, včetně lokalit s dlouhodobě špatným pokrytím od vzdálených venkovských oblastí až po tunely pražského metra. Vše probíhá s cílem zajistit dostupnost internetu ve všech oblastech a umožnit tak další vývoj potenciálních aplikací.

Mezi tyto aplikace patří zejména pro tuto práci relevantní senzorové sítě, domovní automatizace nebo inteligentní zařízení. V souvislosti s těmito aplikacemi narůstá zájem též o ovládací rozhraní která nevyžadují fyzickou interakci uživatele - typicky se jedná o různé systémy na rozpoznávání hlasu nebo systémy rozpoznávání gest, což je součástí této práce. Současně s tím narůstají nároky na zázemí těchto aplikací, tedy servery ve smyslu jak softwaru tak hardwaru.

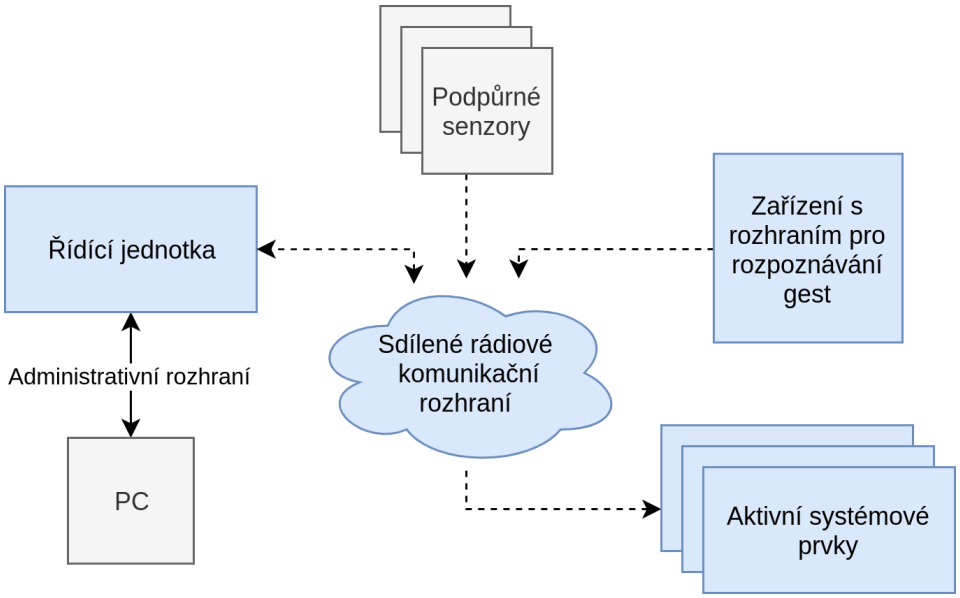
Jako vždy nový vývoj přináší přináší nové možnosti ale i nové výzvy a nároky. Za zmínku stojí, že v důsledku těchto požadavků došlo k vývoji v mnoha jiných souvisejících odvětví elektroniky. Například se značně rozšiřují modularizovaná řešení pro bezdrátovou komunikaci, vznikají nové platformy pro embedded systémy a na popularitě získávají relativně nové programovací jazyky, čemuž se též budu věnovat v této práci.

Vzhledem k tomu, že tato práce ideově navazuje na moji předcházející práci, tak bych ji rád pro kontext věnoval pár slov v následující úvodní podkapitole.

1.2 Předcházející práce

Ve své bakalářské práci [4], která předchází této diplomové práci, jsem navrhl vlastní prototyp systému pro rozpoznávání gest. Mnou navržený systém byl založen na matici senzorů pro měření vzdálenosti, který se osvědčil, ale zároveň se v průběhu času projevily některé jeho nedostatky. Také se v průběhu posledních dvou let na trhu objevily nové typy senzorů, které nabízí potenciální možnosti vylepšení původní konstrukce.

Již původní idea byla vytvořit celý systém, což se ale ukázalo jako příliš rozsáhlé téma na rámec bakalářské práce. Nyní ale na tento návrh navazuji s cílem rozšířit původní koncept na rozsáhlejší systém řízení nebo automatizace skládající se z nezávislých komponent s oddělenou funkcí a bezdrátovým komunikačním rozhraním - nikoliv jako v původním návrhu, kde jedno zařízení zajišťovalo funkci ovládacího rozhraní, řídící jednotky a aktivních systémových prvků v jednom. Nový přístup výrazně zvýší flexibilitu celého systému a zároveň přinese možnost zcela nových funkcí. Následující obrázek 1.1 reprezentuje zjednodušené schéma takového systému.



Obrázek 1.1: Zjednodušené schéma navrhovaného systému

Mým osobním zaměřením je obecně návrh a konstrukce hardwaru, tudíž hlavním cílem v této práci je demonstrace fyzické realizovatelnosti celého systému. Mimo toho jsou dílčí cíle této následující:

- analýza aktuálního stavu v souvisejících oblastech elektroniky
- návrh modelového systému na základě předcházející analýzy
- realizace systému v podobě funkčního vzorku
- zhodnocení parametrů funkčního vzorku
- zhodnocení finančních nákladů realizace

Kapitola 2
Analýza dostupných komponent

Od dokončení mé bakalářské práce proběhl nezanedbatelný vývoj elektronických komponent, zejména v oblasti senzorů, mikrokontrolérů a prvků internetu věcí. Též došlo k vývoji v oblasti používaných programovacích jazyků a na trhu se objevují s nimi spojené vývojové nástroje. Tato celková změna situace na trhu mě přivedla k zásadnímu přehodnocení použité součástkové základny.

Následující podkapitoly se snaží popsat současnou situaci pomocí reprezentativního vzorku se zaměřením na komponenty vhodné do mobilních zařízení, zejména pro zařízení s nízkou spotřebou. Nejedná se tedy o kompletní výčet veškerých dostupných komponent, který by byl mnohonásobně rozsáhlejší. Pro obecné vysvětlení je však podle mne tento přístup dostačující. V jednotlivých podkapitolách, kde je to třeba, se zmíním též například o komunikačních standardech.

2.1 Mikrokontroléry

V současnosti jsou na trhu dostupné mikrokontroléry s 8, 16 a 32 bitovými procesory, jejichž podíl je poměrně rovnoměrný [7] navzdory narůstajícím zájmu o platformu 32 bitovou platformu s jádry od společnosti ARM. Za zmínku stojí nadále velký zájem o 16 bit platformy na Americkém trhu [8].

2.1.1 8-bitové architektury

- **AVR8** - Do této skupiny patří zejména řady ATtiny (např.: ATtiny85 [10]) a ATmega (např.: ATmega328 [9]). Jedná se o dlouhodobě prověřenou a rozšířenou sérii mikrokontrolérů, z nichž ATmega328p tvoří základ známé vývojové platformy Arduino™Uno. V nedávné době byly uvedeny na trh i nové modely (např.: ATtiny817 [11]), které rozšířily množství dostupných periferií. Vzhledem k popularitě těchto mikrokontrolerů mají rozsáhlou podporu ve vývojových nástrojích a lze pro ně použít mnoho různých vývojových prostředí.
- **PIC18** - Mezi zástupce této skupiny patří zejména série PIC18F (např.: PIC18F4520 [12]). Tyto mikrokotroléry obecně vynikají velkým množstvím periferií a analogově-digitálním převodníkem s vyšším rozlišením než konkurenční architetury. Jedná se též o dlouhodobě prověřenou architekturu s kvalitní zázemím a dostupnými vývojovými nástroji.

- **STM8** - Novější zástupce 8-bitové architektury STMicroelectronics, který cílí primárně na automobilový průmysl. Standardní mikrokotroléry této série (např.: STM8AF [13]) tak běžně mají CAN sběrnice a jsou certifikované pro vyšší provozní teploty. Vzhledem k cílovému trhu a novosti těchto mikrokontrolerů nemají velkou softwarovou podporu jako mikrokontroléry AVR a PIC, konfigurační nástroj STMCubeMX je volně dostupný, ale vývojová prostředí a kompilátory jsou licencované. Jejich výhodou je ale výrazně nižší cena.

2.1.2 16-bitové architektury

- **MSP430** - Pravděpodobně nejzajímavější zástupce 16-bitových architektur je MSP430 od společnosti Texas Instruments [15]. Tento mikrokontrolér vyniká extrémně nízkou spotřebou, možností napájení nízkým napětím 1,8V, širokým portfoliem analogově-digitálních rozhraní jako analogově-digitální převodníky s vysokým rozlišením a integrované operační zesilovače. Díky těmto vlastnostem je využíván zejména v senzorových systémech.

2.1.3 32-bitové

- **STM32L4** - Zástupcem této skupiny je například STM32L476RE [16], který jsem použil v již zmíněném senzorovém systému ve své bakalářské práci [4]. Jedná se o mikrokontrolér s nízkou spotřebou s jádrem ARM M4. Mezi produkty STMicroelectronics se jedná o výpočetně nejvýkonnější mikrokotrolér, který má zároveň nízkou spotřebu. Jeho součástí je FPU (koprocesor pro výpočty s desetinnou čárkou) a je vhodný pro výpočetně náročné aplikace - v případě mého předchozího použití tím bylo vyhodnocování výstupů čtyř senzorů současně a vyhodnocování gest pomocí matematického modelu.

- **STM32L0** - Tato série mikrokontrolerů vyniká nízkou spotřeba a je postavena na jádru ARM M0+ (např.: STM32L031K6 [17]. Na rozdíl od STM32L4 nemá FPU, vyznačuje se ale nižší spotřebou a nižší pořizovací cennou. Je tedy vhodný pro malá zařízení u kterých je primárním požadavkem právě spotřeba energie a tedy i výdrž baterie.

- **STM32G0** - Relativně novým zástupcem této série mikrokontrolerů je STM32G031J6, který do portfolia STMicroelectronics přináší několik nových možností. Mimo snížení spotřeby oproti STM32L4 a STM32L1 je to itegrovaná napěťová reference a dedikované obvody pro monitorování stavu baterie. Dále je dostupný v provedení pouzdra SO8N (pouze 8 vývodů a minimální velikost). Tyto mikrokontroléry jsou specificky navržené pro aplikace provozované na bateriovém napájení a lze je považovat za přímou konkurenci 8-bitovým architekturám.

2.1.4 Programovací jazyky

Současně s vývojem mikrokontrolerů narůstají i požadavky na jejich programovaní. Rozsáhlejší a obtížněji spravovatelný kód nebo požadavek na vícevláknové aplikace komplikují vývoj softwaru a zvyšují riziko poruch. Tyto nedostatky se snaží řešit následující dva programovací jazyky.

- **Rust** - Tento programovací jazyk se primárně zaměřuje na bezpečnost a stabilitu aplikací [5]. Svojí syntaxí vychází z jazyka C/C++ avšak snaží se řešit jeho nedostatky v oblasti správy paměti - například známá rizika spojená s dynamickou alokací nebo se správou vícevláknových aplikací. Rust byl vyvinut společností Mozilla a byl původně určen k vývoji PC aplikací, aktuálně tak standardní knihovny Rust nepodporují mikrokontroléry a některé architektury (např.: ARM M0+ [6]) nejsou plně podporované ani kompilátorem. Rust ale prochází velmi dynamickým vývojem, získal si značnou pozornost a tak lze v dohledné době očekávat jeho velké uplatnění ve vestavěných systémech.

- **MicroPython** - Odvozením od již velmi rozšířeného jazyka Python 3 [14] vznikla jeho odlehčená verze určená pro vestavěné systémy. MicroPython má hlavní cíl zrychlit vývoj nových aplikací a v současnosti již podporuje mnoho platforem mikrokontrolerů. Stejně jako Python umožňuje komplexní správu vícevláknových aplikací nebo import velkého množství standardních knihoven, kterými vyniká nad ostatními programovacími jazyky.

2.2 Bezdrátová komunikace

Odvětví IoT se stále více rozvíjí a s ním i požadavky na možnosti bezdrátové komunikace, zejména je kladen důraz na nízkou spotřebu, spolehlivost, velký dosah a jednoduchost implementace. Na trhu se tak objevují nové standardy komunikačních rozhraní nebo modifikace již rozšířených standardů.

Vzhledem k tom, že návrh vlastního přijímače/vysílače s anténou na tištěném spoji je časově i technicky náročný a je tedy snaha zjednodušit konstrukci hardwaru, tak se na trhu objevují dedikované systémy na čipu nebo přímo i celé moduly. Tyto systémy nebo moduly používají pásma ISM pásma 433MHz, 868MHz, 915MHz a 2,4GHz a můžeme se u nich setkat s následujícími standardy.

- **Bluetooth** - Již dlouhou dobu používaný standard, který byl navržen pro přímou komunikaci mezi dvěma zařízeními. V současnosti je již definována verze 5, která v sobě zahrnuje funkce Bluetooth Low Power a Bluetooth Mesh [21], která rozšiřuje možnost komunikace na celou síť zařízení a výrazně tak zvyšuje možný dosah komunikace. Komunikace probíhá v pásmu 2,4GHz.

- **Wi-Fi** - Standard primárně používaný pro bezdrátový přístup k síti LAN a internetu. Největším požadavkem je tedy v prvé řadě velká přenosová rychlost, ale existují modifikace umožňující provozovat takzvaný promiskuitní mód (přímá komunikace mezi více uživateli bez centrálního přístupového bodu), nízkoenergetický mód v rámci standardu IEEE 802.11ah nebo vytvořit i síť Wi-Fi Mesh. Komunikace probíhá v pásmech 900Mhz, 2,4Ghz a 5Ghz.
- **Sigfox a LoRa** - Jedná se o dva standardy dedikované specificky pro IoT aplikace. Oba jsou navrženy pro komunikaci na větší vzdálenosti (dosah až několik kilometrů) a využívají pásma 868Mhz a 915Mhz. Hlavním rozdílem mezi nimi je přístup vytváření sítě pokrytí, Sigfox se snaží vytvořit globální proprietární síť, zatímco LoRa je otevřeným standardem umožnující uživatelům vytvořit si vlastní lokální LoRa WAN sítě.
- **ZigBee** - Tento komunikační protokol byl navržen především pro účely domovní automatizace [22], ale existují i zařízení s tímto protokole schopná komunikace na vzdálenosti řádově několik kilometrů v úsporném režimu. Obecně se jedná o prověřený a stále se rozvíjející standard a na trhu existuje velké množství kompatibilních zařízení. Jedná se o variaci standardu IEEE 802.15.4 a komunikace probíhá v pásmu 2,4GHz.

Mimo zmíněné standardy existuje mnoho proprietárních standardů různých výrobců, které se liší výrobce od výrobce. Většina z nich využívá již zmíněná ISM pásma 433MHz, 868MHz, 915MHz a 2,4GHz.

2.2.1 Systémy na čipu

Následující systémy na čipu se snaží snížit komplexnost návrhu bezdrátového rozhraní integrací RF rozhraní přímo na čipu. Celý hardwarový návrh bezdrátového rozhraní se tím redukuje pouze na návrh antény. Spojením s mikroprocesorem se zároveň redukuje velikost celého zařízení.

- **NRF52** - Bezdrátový mikrokontrolér společnosti Nordic Semiconductor [23]. Z již zmíněných komunikačních standardů podporuje protokoly Bluetooth 5.0, Bluetooth Mesh a proprietární protokol ANT na 2,4GHz. Dále pak podporuje další bezdrátové funkce jako NFC.
- **STM32WB** - Bezdrátový mikrokontrolér společnosti STMicroelectronics [24]. Podporuje protokoly Bluetooth 5.0, Bluetooth Mesh a ZigBee.
- **ESP32** - Zajímavý bezdrátový mikrokontrolér společnosti Espressif Systems, který získal velkou pozornost díky dobrému poměru mezi funkčností a cenou. Podporuje protokoly Wifi a Bluetooth 4.2.

2.2.2 Komunikační moduly

Komunikační moduly se snaží redukovat potřebu návrhu RF částí zařízení na absolutní minimum. Mnohé z nich mají integrované antény nebo dedikovaný konektor pro anténu.

- **ZETAPLUS** - Velmi dostupný a rozšířený modul společnosti RF Solutions, existují verze pro tři různé frekvence [19] 433MHz, 868MHz a 915MHz. Modul má integrovanou anténu a k jeho použití je tedy potřeba minimální znalost RF komunikace. Pro komunikaci jsou použity proprietární protokoly.

- **RF-LORA** - Je dalším modulem od společnosti RF Solutions. Tento modul je srovnatelný s ZETAPLUS modulem, avšak narozdíl od něj má kompletně implementované protokoly standardu LoRa. Existují dvě varianty pro pásma 868Mhz a 915Mhz.

- **SPSGRFC a SPBTLE-S1** - Jedná se o dva na první pohled velmi podobné moduly [25][26], které se ale shodují pouze ve fyzických rozměrech a uspořádání pinů. Modul SPSGRFC je bezdrátovým modulem pro pásma 433MHz, 868MHz a 915MHz, zatímco SPBTLE-S1 je modul pro Bluetooth verze 4.2. Jejich shodné fyzické rozměry mohou představovat zajímavou výhodu - bez jakékoliv změny návrhu tištěného spoje je možné vytvořit modifikace téhož zařízení pro Bluetoth nebo pro komunikaci na delší vzdálenost v ISM pásmech.

- **ESP32-WROOM** - Modul založený na již zmíněném bezdrátovém mikrokontroléru ESP32 [27]. Jedná se o velice flexibiní řešení, může sloužit pouze jako komunikační modul, kdy integrovaný procesor zajišťuje zpracování bezdrátové komunikace, nebo přímo může tvořit jádro navrhovaného systému, kdy je procesor plně naprogramován uživatelem. V případě použití jako bezdrátového modulu dodává výrobce oficiální firmware po jehož naprogramování komunikuje modul po sériové lince a je ovládán pomocí dedikovaných AT příkazů. Je-li třeba modul pro účely aplikace plně naprogramovat, tak je možné použít oficiální knihovny v jazyce C nebo pro rychlý návrh komunitní firmware a knihovny Micropython.

Na následujících obrázcích 2.1a, 2.1b, 2.1c a 2.1d na straně 8 je možné vidět uvedené bezdrátové moduly.

(a) : Modul ZETAPLUS

(b) : Modul ESP32-WROOM

(c) : Modul SPSGRFC

(d) : Modul SPBTLE-S1

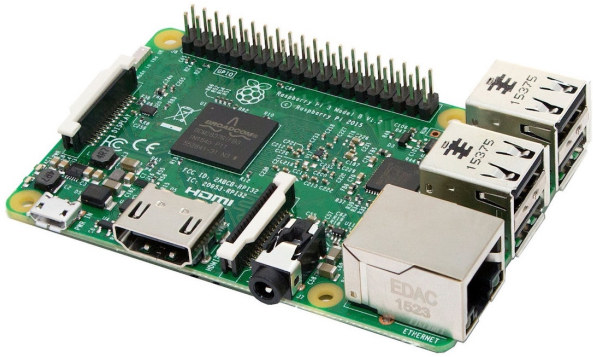
Obrázek 2.1: Bezdrátové komunikační moduly [28]

2.3 Embedded Linux platformy

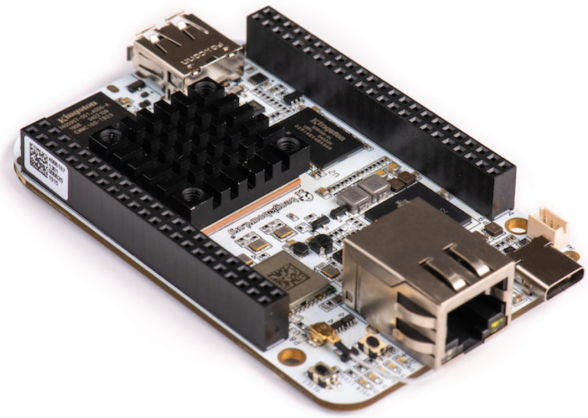
S narůstajícími nároky na výpočetní výkon malých zařízení a s rozšiřujícím se uplatněním operačního systému Linux, na trhu se objevuje stále více platforem pro embedded Linux. Ve většině případů se jedná průmyslové moduly, které vyžadují vlastní návrh zařízení, kde je tento modul umístěn. Mimo tyto moduly se ale zároveň rozšiřují takzvané SBC (Single board computer - jednodeskový počítač), které najdou uplatnění v menších projektech, kde návrh vlastního hardwaru není finančně výhodný. Tyto počítače narozdíl od embedded Linux modulů mají typicky sadu rozhraní obsahují USB, ethernet, Wi-Fi nebo HDMI pro připojení displeje.

- **Raspberry Pi** - Pravděpodobně nejrozšířenější platforma pro embedded Linux. Původně tato platforma vznikla pro výukové účely, ale již se běžně objevuje v některých průmyslových aplikacích. V současnosti již existuje verze 4, jejíž jádro tvoří čtyřjádrový procesor a mezi periferiemi můžeme najít 2x mini HDMI, 2x USB 3.0, 1x USB Typ C a gigabitový ethernet.

- **Banana Pi** - Modifikace Raspberry Pi 3, oproti běžným periferiím přidává SATA rozhraní pro připojení disku. Tato modifikace je vhodná pro vytváření serverových aplikací.
- **ASUS Tinkerboard** - Modifikace Raspberry Pi 3 navržená firmou ASUS s cílem zvýšit výpočetní výkon.
- **BeagleBone** - Zajímavá platforma s procesorem firmy Texas Instruments. Její hlavní výhodou oproti jiným platformám je nižší spotřeba - embedded Linux platformy jsou obecně energeticky náročné na provoz.



(a) : Raspberry Pi 3B+ [30]



(b) : BeagleBone AI [31]

Obrázek 2.2: Jednodeskové počítače s embeded Linux

2.4 Senzory pro rozpoznávání gest

Již před zahájením této rešeršní práce jsem měl otestovaný senzor APDS-9960 firmy Broadcom. Vzhledem k tomu, že se mi tento senzor osvědčil a vzhledem

k tomu, že mám k tomuto senzoru kompletní knihovny funkcí, tak byl tento senzor téměř jasnou volbou. V průběhu rešerše jsem neobjevil žádný senzor se srovnatenými parametry mimo jeho novější verze APDS-9500.

- **APDS-9960** - Infračevený senzor pro rozpoznávání gest, osvětlení a barev. Tento systém využívá čtyřzónový přijímač a interní paměť do které se ukládá časový průběh příjmu ze všech čtyř směrů. Z těchto časových průběhů je poté možné vyhodnotit směr nebo rychlost pohybujícího se předmětu, případně i složitější gesta. Vlastní vyhodnocení musí provádět mikrokontrolér, ke kterému je tento senzor připojen.
- **APDS-9500** - Novější verze senzoru APDS-9960. Oproti svému předchůdci využívá matici pixelů 16x16 a obsahuje integrovanou vyhodnocovací logiku. Jedná se ale o výrazně dražší senzor.

Kapitola 3

Výsledná podoba systému

Na základě analýzy uvedených součástek a úvaze nad možnými řešeními jsem vytvořil následujících pět možností realizace navrhovaného systému.

	MCU	RF rozhraní	RF protokol	Řídící jednotka
1	STM32WB	*Integrované*	Zigbee	Raspberry Pi 3
2	nRF52840	*Integrované*	ANT/Bluetooth	Raspberry Pi 3
3	STM32G0	ESP32-WROOM	Wi-Fi	Raspberry Pi 4
4	STM32G0	SPSGFRC/SPBTLE	ISM/Bluetooth	Raspberry Pi 4
5	STM32G0	RF-LORA/SX1272	LoRaWAN	LoRa Gateway

Tabulka 3.1: Navrhované varianty

Uvedené varianty jsem prověřil podrobněji z hlediska softwaru a vyřazovací metodou jsem postupně zhodnotil proveditelnost jejich realizace. Důvody vyřazení jednotlivých variant byly následující:

- **Varianta 1** - Mikrokontrolér ST32WB vyžaduje naprogramování dedikovaným firmwarem pro rádiový koprocesor. Tento firmware vyžaduje licenci a není velmi dobře zdokumentovaný. Nejsou dostupné HAL funkce jako pro ostatní periferie. Lze předpokládat, že absence dokumentace a knihoven je důsledkem faktu, že série mikrokontrolerů ST32WB je nová a časem se tato situace změní.

- **Varianta 2** - Obdobný problém jako ve variantě jedna. Navíc by tato varianta vyžadovala kopletní změnu ekosystému (jsem dlouhodobě uživatelem vývojových prostředků STMCubeMX). Tato varianta by byla výrazně více časově náročná.

- **Varianta 4** - Moduly SPSGFRC a SPBTLE postrádají aplikační knihovny, to by reálně znamenalo velké množství programátorské práce pro vytvoření knihovny funkcí. Příliš časově náročná varianta.

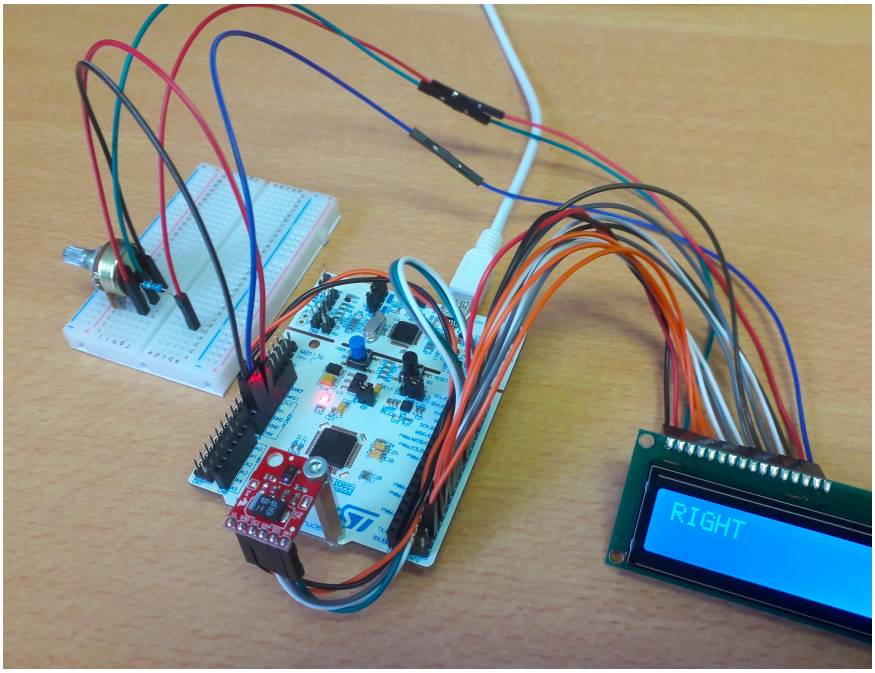
- **Varianta 5** - Hlavním faktorem vyřazení je v tomto případě cena. Bezdrátové moduly LoRa jsou výrazně dražší (až dvojnásobně) než jiné bezdrátové moduly. Pro plné využití je navíc potřeba LoRa Gateway, která vysoce svojí cennou převyšuje veškeré ostatní komponenty.

Po vyřazení ostatních variant jsem zvolil modifikovanou **variantu 3**, která nad ostatními vyniká v prvé řadě cenou, dále jednoduchostí konfigurace a díky použití Wi-Fi možností využít existující infrastrukturu (ještě nižší cena a jednoduší konfigurace). Modifikací bylo využití mikrokontrolerů série STM32L476 pro zařízení rozpoznávající gesta. Ostatní prvky využívají komponenty dle původní **varianty 3**.

Pro vybranou variantu jsem otestoval všechny dílčí prvky pomocí vývojových kitů, abych prověřil i jejich reálné parametry. Zejména jsem jako první testoval senzor APDS9960 v kombinaci s STM32L476, a v druhém testu konfiguraci a komunikaci s ESP32 v kombinaci s programovacím jazykem MicroPython.

3.1 Testování APDS9960

Senzor APDS9960 jsem již dříve testoval v jednoduchém zapojení pro demonstraci jeho funkcí a tedy jsem měl již k dispozici i zdrojové kódy pro jeho zprovoznění. Jako základ posloužila volně dostupná knihovna pro Arduino od společnosti Sparkfun [33]. Tuto knihovnu jsem očistil od funkcí, které jsem nepotřeboval, a přepsal jsem ji z jazyka C++ do jazyka C společně s knihovnami HAL (knihovny STMicroelectronics pro hardwarovou abstrakci). Následně jsem sestavil testovací přípravek, jak je možné vidět na obrázku 3.1



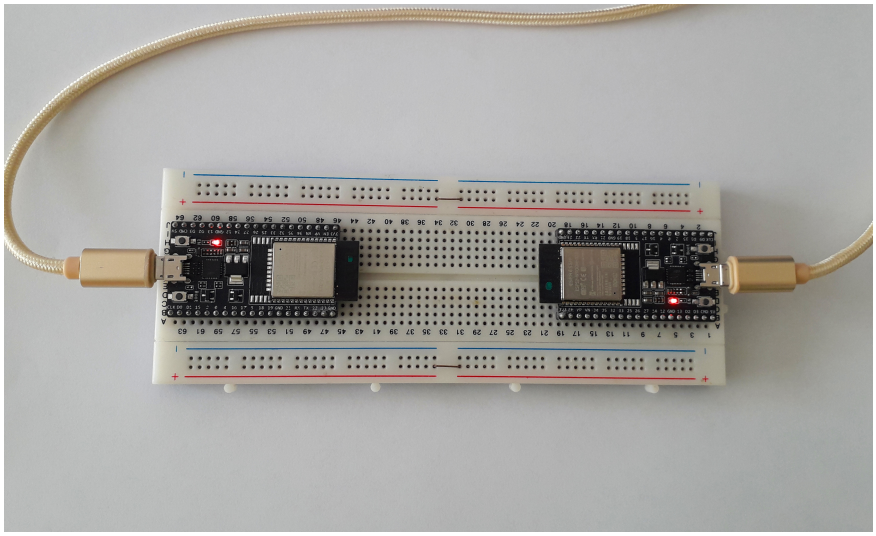
Obrázek 3.1: Testovací přípravek APDS-9960

Vlastní ověření knihoven bylo poměrně rychlé, konfigurace vývojového kitu mikrokontroléru i vlastního senzoru proběhla bez větších komplikací. Pro pohodlnost jsem přípravek doplnil od znakový display a podrobnější výstup

jsem vypisoval přes virtuální sériový port na PC. Tímto jsem si ověřil, že jsou zvolené komponenty skutečně použitelné a nehrozí větší riziko budoucích komplikací a případného zdržení navazujících úkonů.

3.2 Testování ESP32

Pro ověření funkčnosti modulů ESP32-WROOM-32D jsem pořídil dva vývojové kity, viz. obrázek 3.1. Do těchto kitů jsem nahrál zdrojový kód pro MicroPython [14] ve verzi 1.12 a následně jsem otestoval dílčí funkcionality.



Obrázek 3.2: Testovací přípravek ESP32-WROOM-32D

Hlavní funkcionalitou, kterou jsem potřeboval ověřit, bylo připojení k Wi-Fi a komunikace mezi moduly, případně mezi modulem a PC. Test jsem provedl konfigurací přes REPL (interaktivní terminál) na virtuálním sériovém portu modulu ESP32. Všechny dílčí testy proběhly bez technických komplikací a pro stručnost je dále uvádím pouze v bodech:

- Připojení modulů ESP32 k lokální Wi-Fi síti
- Ping modulů z PC
- Otevření soketu pro režimy UDP nebo TCP
- Jednosměrné odeslání testovacího paketu
- Odeslání paketu s potvrzením příjmu

Po tomto testu jsem již měl ověřenou funkci všech komponent, které jsem se rozhodl použít a mohl jsem tímto přejít k návrhu hardwaru.

Kapitola 4
Hardware

Vzhledem tomu, že se jedná o celý systém skládající ze z několika různých prvků (viz. obrázek 1.1 v úvodní kapitole, str. 2), tak pro přehlednost budu nadále jednotlivé prvky systému označovat následujícím způsobem:

- *Sense* - Zařízení s rozhraním pro rozpoznávání gest
- *Control* - Aktivní systémový prvek řídící osvětlení (RGB LED pásek)
- *Server* - Řídící jednotka systému

Tuto názvovou konvenci i udržuji jak ve schématech a podkladech pro návrh DPS, tak ve zdrojovém kódu. Vlastní návrh hardwaru jsem provedl v následujících třech krocích:

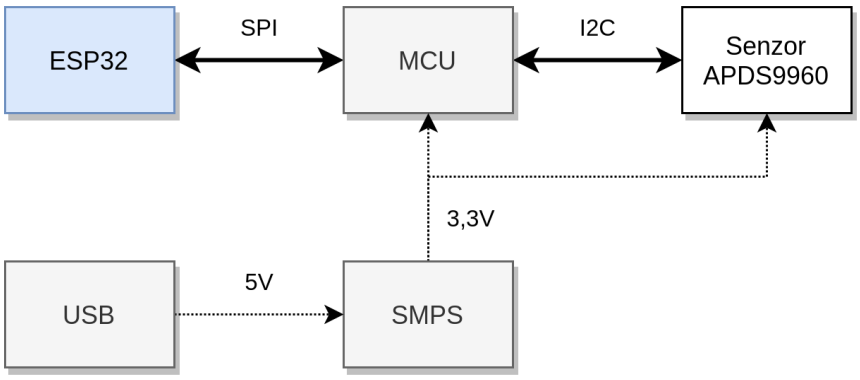
- návrh základního blokového schématu s jednotlivými funkční bloky, propojení mezi nimi a napájením
- kreslení funkčního schématu s výběrem konkrétních elektronických komponent
- návrh DPS a generování výrobních dat

Protože *Server* nepoužívá žádný dedikovaný hardware, tak se v této kapitole nadále věnuji pouze systémovým prvkům *Sense* a *Control*.

4.1 Bloková schémata

4.1.1 Sense

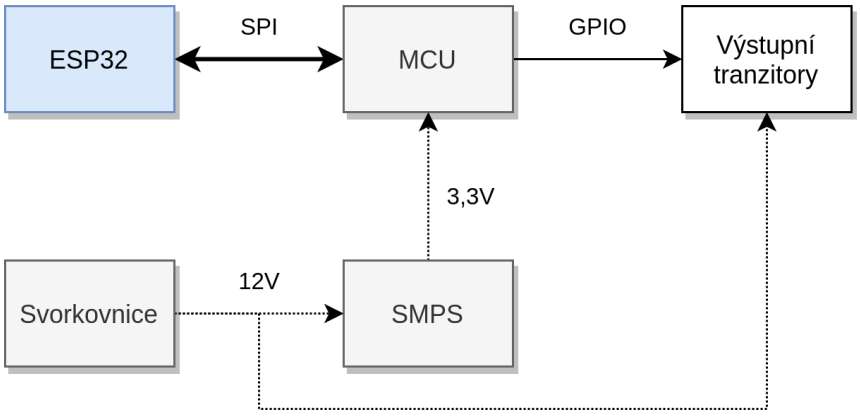
Zařízení *Sense* zajišťje detekci gest a následně je odesílá, z toho vyplývá, že musí mimo MCU obsahovat Wi-Fi modul a vlastní senzor. Samozřejmostí je též i napájení, kde jsem zvolil pro praktičnost USB, které mi mimo napájení umožní vytvářet kontrolní výstupy přes virtuální sériový port. Jednoduché blokové schéma lze vidět na obrázku 4.1.



Obrázek 4.1: Blokové schéma *Sense*

4.1.2 Control

Zařízení *Control* přijímá příkazy a následně dle nich řídí výstupy. Pro účely modelové realizace jsem se rozhodl ovládat RGB LED pásek a výstupy tak tvoří tranzistorové pole. Jako zdroj napájení slouží 12V shodné s napětím pro LED pásek. Jednoduché blokové schéma lze vidět na obrázku 4.2



Obrázek 4.2: Blokové schéma *Control*

4.2 Funkční schémata

Pro návrh zařízení jsem použil návrhový systém KiCad, který je přístupný volně ke stažení, viz. [34]. Hlavní motivací byla otevřenost programu a dostatečná funkcionalita, která se dlouhodobě rozšiřuje. Pro všechna schémata jsem využil funkci hierarchických schémat, kde je možné části zapojení uzavřít do bloků se vstupy a výstupy, a tyto bloky lze poté zapojit na vyšší úrovni.

4.2.1 Sense

MCU + Wi-Fi

Nejprve jsem se zaměřil na blok MCU + Wi-Fi, kde bylo nejprve třeba navrhnout zapojení SPI mezi mikrokontrolérem STM32L476 a modulem ESP32. Toto zapojení jsem dále doplnil o signalizační linky a také sdílený konektor se sériovým rozhraním jak z mikrokontrolérem tak z ESP32. Tento konektor slouží jako programovací konektor nebo může po spojení jumpery fungovat jako přímá propojka mezi mikrokontrolérem a ESP32, což umožní případnou komunikaci po sériové lince mezi nimi. V závěru jsem již doplnil pouze reset tlačítka, signalizační led a standardní blokovací kondenzátory. Výsledné zapojení je k nahlédnutí v příloze A1, list 4/5.

Senzor

V další části schématu jsem se věnoval senzoru APDS9960. Ten je zapojen z větší části dle zkušebního modulu, který jsem měl k dispozici. Pouze jsem zde nahradil velký tantalový kondenzátor kaskádou menších keramických kondenzátorů kvůli úspoře místa v okolí senzoru. Výsledné zapojení je k nahlédnutí v příloze A1, list 5/5.

USB

Na závěr jsem se věnoval bloku napájení z USB. Zde jsem použil již osvědčené zapojení s integrovaným obvodem ST1S03, které jsem použil ve své předchozí práci [4]. Výstupní napětí je v tomto případě 3,3V. Výsledné zapojení je k nahlédnutí v příloze A1, list 3/5. Ve schématu je možné vidět i zapojení pro nabíjení Li-Pol baterií, které nyní nebude využito, slouží pouze jako příprava pro budoucí potenciální modifikace.

4.2.2 Control

MCU + ESP32

Prvním blokem, kterému jsem se v návrhu věnoval byl stejně jako u zařízení *Sense* mikrokontrolér a Wi-Fi modul. Zapojení je v tomto případě principiálně shodné, pouze jsem použil jiné MCU, konkrétně STM32G031 v menším pouzdru. Výsledné zapojení je k nahlédnutí v příloze A2, list 2/4.

Výstupy a vstupy

Jako druhý blok jsem navrhoval zapojení výstupů a vstupů. Pro řízení výstupů jsem použil tranzistorové pole TPL7407LDR, které je přímou náhradou běžných tranzistorových polí ULN2003. Oproti nim se jedná o MOSFETy nikoliv Darligtonovy tranzistory, díky čemuž mají menší úbytek napětí na výstupu v sepnutém stavu. Dále jsem do obvodu přidal přípravu na dvě tlačítka s pull-up rezistorem, filtračním kondenzátorem a diodami BAT54S

pro základní ochranu proti ESD. Výsledné zapojení je k nahlédnutí v příloze A2, list 4/4.

Napájení

Posledním navrhovaným blokem byla část napájení. Zde jsem pro jednoduchost použil kompletní modul spínaného zdroje TSR 1-2433, jehož maximálním vstupním napětím je 24V a výstupním napětím jsou 3,3V. Dále jsem pro minimální ochranu obvodu přidal polymerovou pojistku a transil SMAJ24CA do 24V. Výsledné zapojení je k nahlédnutí v příloze A2, list 3/4.

4.3 Návrh DPS

Funkční vzorky zařízení jsem se rozhodl navrhnout tak, aby byly se nejednalo o holé DPS, ale aby měly reprezentativní vzhled odpovídající jejich použití. Návrh DPS jsem tedy zahájil výběrem vhodných konstrukčních krabic. Pro zařízení *Sense* jsem hledal konstrukční krabici vhodnou pro přenosná zařízení a zvolil jsem krabičku TEKO 10007.5 [35]. Pro zařízení *Control* jsem hledal naopak krabici vhodnou pro elektroinstalace a zvolil jsem tedy propojovací krabici SPELBERG MINI-25 [36].

Poté co jsem měl jasně vymezen rozměr navrhovaných DPS a mohl jsem přejít k vlastnímu návrhu. Mimo běžných návrhových pravidel bylo nutné pouze uvažovat správné umístění Wi-Fi modulu. Zde jsem se již v minulosti setkal s komplikacemi u rádiových modulů a proto jsem se držel doporučení výrobce dle dokumentace.

4.4 Osazení a zprovoznění

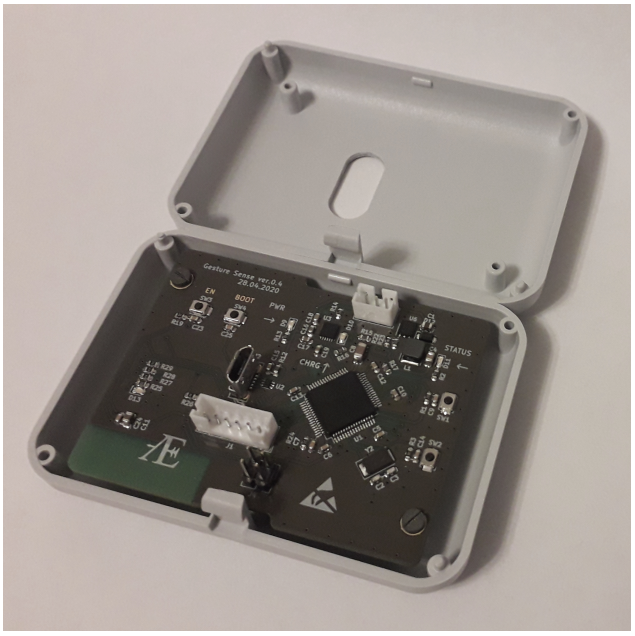
Vzhledem k technologickým prostředkům, které jsem měl k dispozici jsem se rozhodl pro ruční osazení DPS. Zapájení většiny součástek jak na desce *Sense* tak na desce *Control* nepředstavovalo komplikaci, větší pozornost vyžadovala pouze QFN pouzdra integrovaných obvodů. Speciální přístup byl nutný u senzoru APDS9960, který jako optický senzor musí být pájen takzvaně suchými metodami, tedy bez přidaných tavidel ani jiných kapalin, které by mohly proniknout dovnitř senzoru a poškodit jej. Tento senzor jsem tedy na deskách osazoval jako poslední za použití horkovzduchu až po kompletním omytí desek po předcházejícím pájení.

Po osazení jsem prověřil funkčnost zařízení v následujících krocích, po jejichž dokončení jsem přešel do fáze návrhu softwaru:

- Přeměření hodnot napájecích napětí zdrojů
- Nahrání zdrojového kódu MicroPython do ESP32
- Ověření možnosti konfigurace ESP32 po sériovém rozhraní
- Nahrání testovací konfigurace do mikrokontrolerů

- Test komunikace senzoru přes I2C

Fotografie osazených a zprovozněných DPS včetně jejich konstrukčních krabic lze vidět na následujícím obrázku 4.3.



(a) : Zařízení Sense

(b) : Zařízení Control

Obrázek 4.3: Osazené DPS s kontrukčními krabicemi

Kapitola 5
Software

Návrh softwaru byl relativně rozsáhlou činností, kterou bylo třeba rozdělit na dílčí části. Dobrou praxí je v tomto případě nejprve definovat obecně fungování systému jako celku, použitých komunikačních protokolů a příkazů v rámci komunikačního rámce a poté přejít k podrobnostem implementace dílčích zařízení.

V některých částech systému je použit zdrojový kód v jazyce C a v jiných v jazyce Python. To vyžaduje dostatečně hlubokou úvahu nad tím, jak vhodně implementovat dílčí funkce s ohledem na součinnost dvou různých programovacích jazyků, které následují odlišná paradigmata.

Použití dvou programovacích jazyků ale v některých momentech přináší nezanedbatelné výhody, což bylo mojí motivací takto celý systém navrhovat. Například v jazyce C bylo výrazně snazší implementovat nízkoúrovňové funkce, řízení hardwarových výstupů a případně i úsporných režimů. Naproti tomu v jazyce Python bylo velmi jednoduché konfigurovat komunikaci přes Wi-Fi a zprovoznit potřebné síťové protokoly, protože většina těchto funkcí je již součástí standardních knihoven.

Při návrhu softwaru jsem v souladu s předchozími poznámkami postupoval v následujících krocích:

- Návrh celkové struktury systému
- Definice základního komunikačního protokolu
- Návrh a programování *Control*
- Návrh a programování *Server*
- Návrh a programování *Sense*

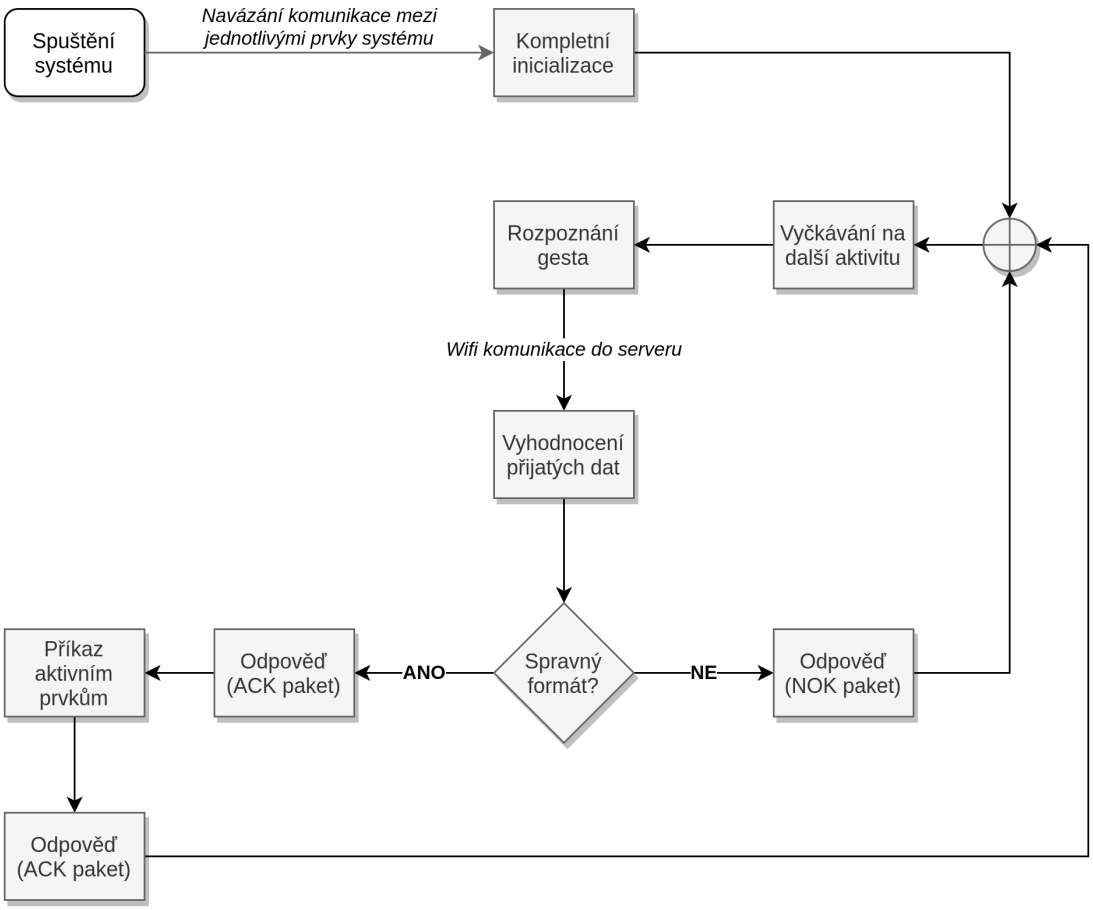
Těmto dílčím krokům se věnuji následujících podkapitolách.

5.1 Celková struktura systému

Po prvotní inicializaci celý systém vyčkává na aktivační událost v podobě detekce gesta v zařízení *Sense*. To po zjištění známého gesta zašle informační paket na *Server*, který vytvoří odpovídající příkazový paket do cílového zařízení *Control*. Tato implementace zároveň umožní připojení více aktivních i ovládacích prvků.

Celý systém bude pro jednoduchost statický, tedy všechna zařízení budou mít pevně dané IP adresy. Z tohoto důvodu musí mít všechna zařízení informaci o použitých IP adresách. To zajišťuje jednotný konfigurační soubor, který je shodný pro všechna zařízení a obsahuje informaci o přihlašovacích údajích na Wi-Fi sít, tak důležité IP adresy jako například adresu serveru i použité komunikační porty.

Při jakékoliv komunikaci bude pro rychlost použit protokol UDP a zařízení které obdrží zprávu vždy posílá potvrzovací zprávu. Přesný formát komunikačního rámce, který k tomuto je použit uvádím v následující podkapitole.

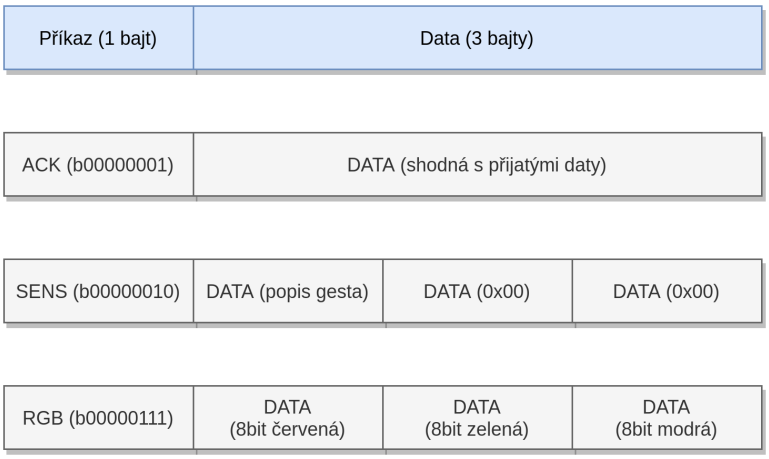


Obrázek 5.1: Blokový diagram celého systému

5.2 Komunikační rámec

Pro komunikaci mezi zařízeními je použit jednoduchý komunikační rámec, který má pevně danou délku čtyři bajty. Jeho vnitřní strukturu pro příklad tří nejdůležitějších paketů je možné vidět na následujícím obrázku 5.2. Funkce jednotlivých paketů je následující:

- ACK - potvrzovací paket, cílové zařízení jím v komunikaci potvrzuje přijetí předchozího paketu
- SENS - informační paket, který zasílá zařízení *Sense*, obsahující informaci o detekovaném gestu
- RGB - příkazový paket, který nastavuje hodnoty



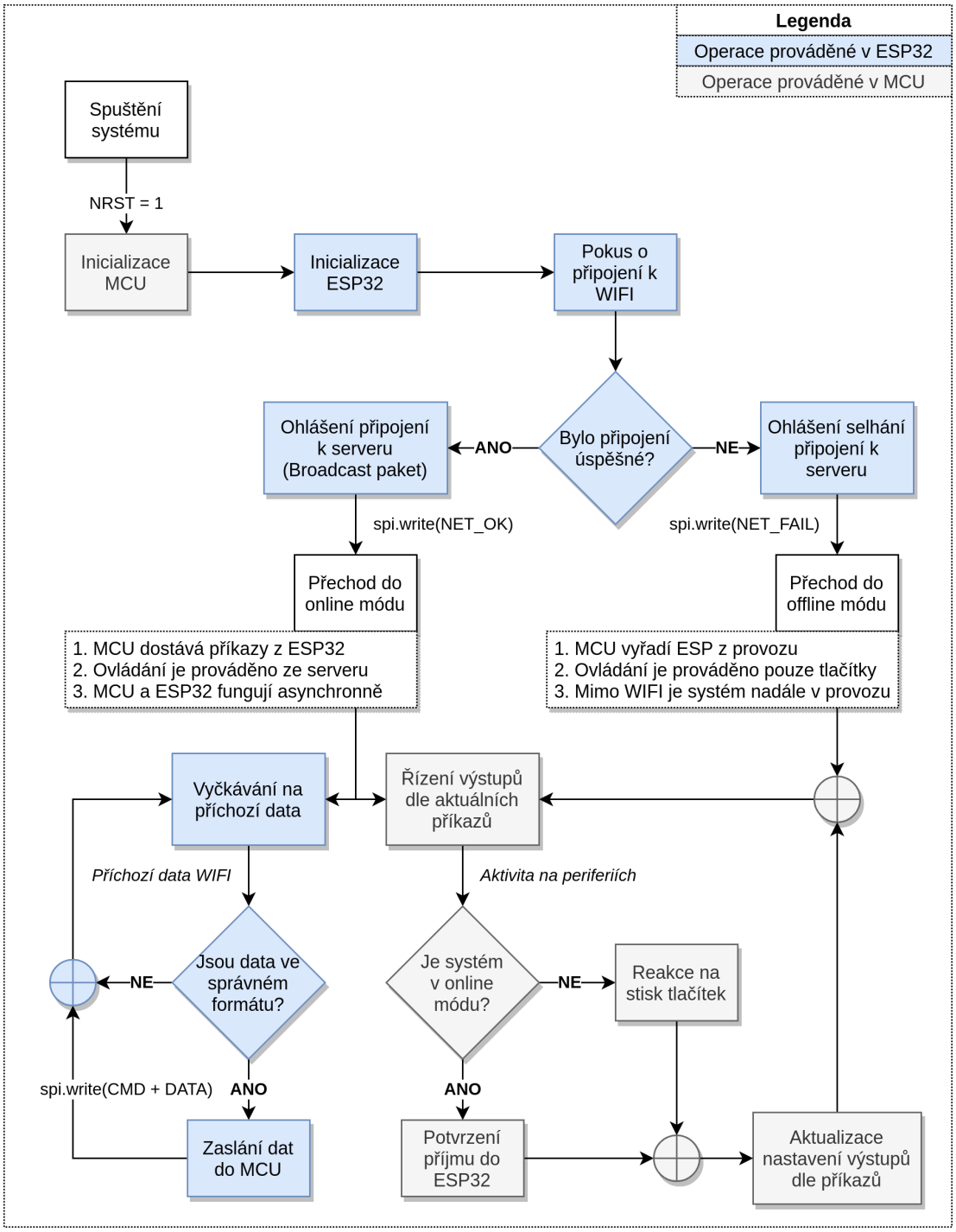
Obrázek 5.2: Komunikační rámec včetně příkladu paketů ACK, SENS a RGB

5.3 Control

Zařízení *Control* má celkem dva procesory, jedním z nich je mikrokontrolér STM32G031 a druhý je koprocesor rádiového modulu ESP32-WROOM. Tomuto odpovídá i struktura programu, který obsahuje dvě části vzájemně spolupracující v asynchronním režimu. Předávání dat mezi těmito dvěma procesory zajišťuje sběrnice SPI. Blokový diagram popisující funkci celého softwaru je na obrázku 5.3 na straně 24.

Modul ESP32 zajišťuje veškerou funkcionalitu týkající se připojení k síti Wi-Fi, zpracovává příchozí data a vytváří rámce ACK v případě správně doručených příkazů. Doručená data poté předává mikrokontroléru.

Mikrokontrolér STM32G0 zajišťuje řízení výstupů a zajišťuje nízkoúrovňové funkce. Při připojení napájení se spouští jako první a následně aktivuje modul ESP32, který vyžaduje správnou bootovací sekvencí vstupů a spouští se výrazně déle.

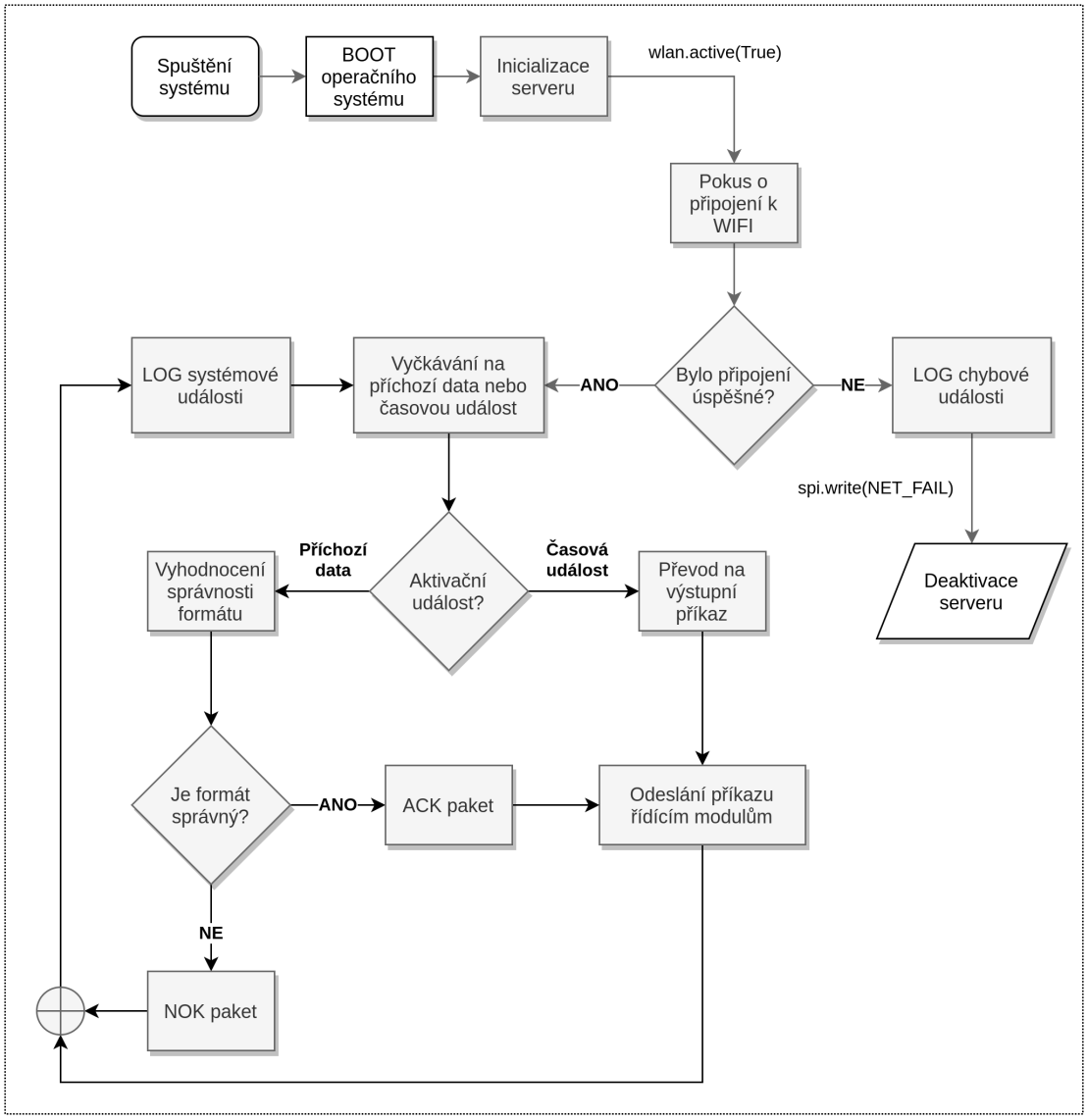


Obrázek 5.3: Blokový diagram *Control*

5.4 Server

Server je plně virtuální implementovaný v jazyce Python a není tak vázaný na konkrétní hardware. Během návrhu jsem jej provozoval na svém PC s Linux Mint 19.3 a následně jsem jej přesunul na Raspberry PI 4.

Jádro *Serveru* tvoří směrovací tabulku příkazů tabulka implementovaná jako knihovna. Knihovny v jazyce Python představují datový typ, který je optimalizovaný na vyhledávání vazeb mezi dvěma prvky (klíč a hodnota). V tomto případě knihovna obsahuje jako klíč vstupní příkaz, který odpovídá datům přicházejícím z komponenty *Sense* a hodnotou jsou IP adresy a příkazy určené zařízení *Control*. Pro vytváření konfigurace jsem vytvořil konfigurační skript, který tyto vazby uloží do souboru pro *Server*.



Obrázek 5.4: Blokový diagram *Server*

5.5 Sense

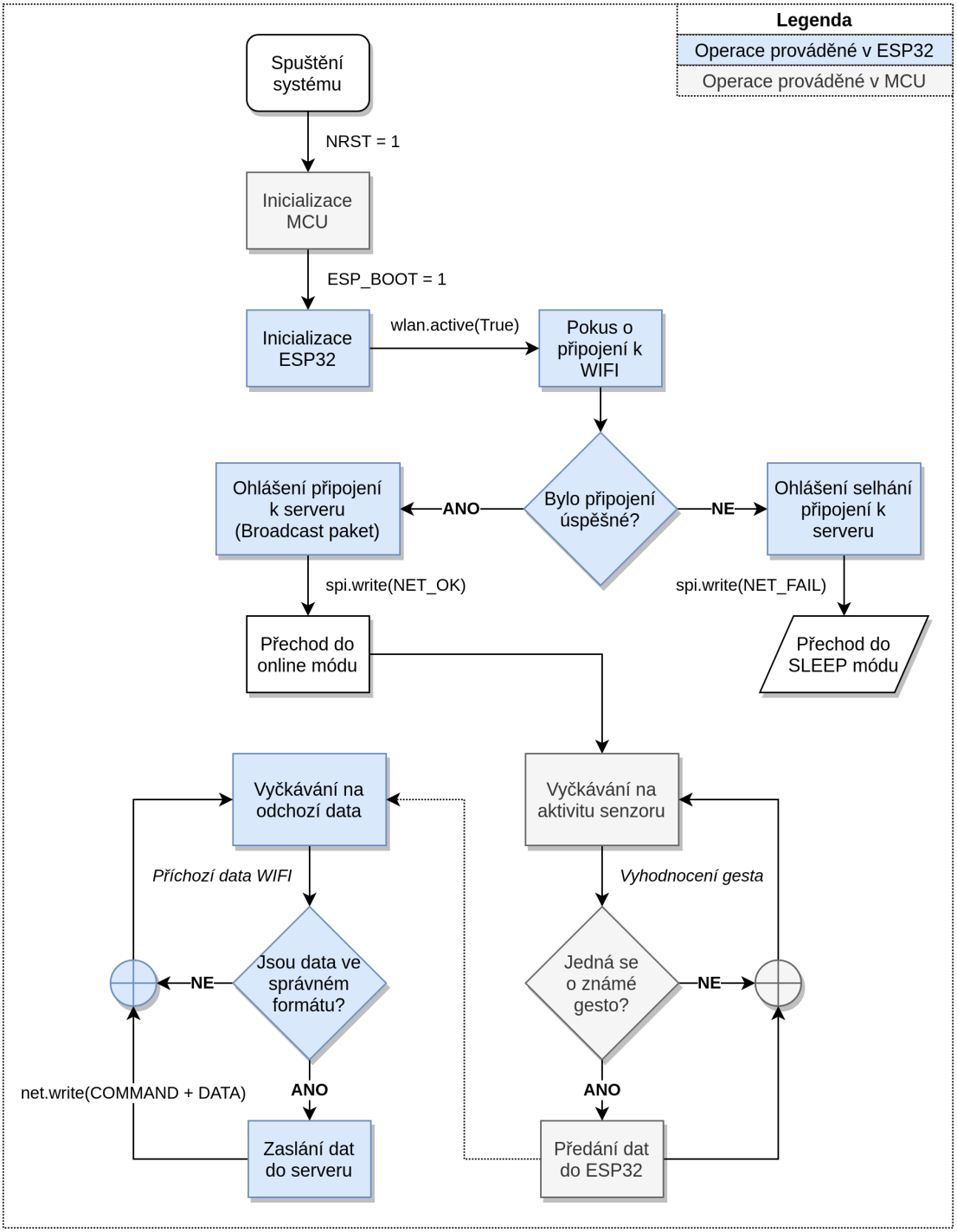
Zařízení *Sense* má shodně jako zařízení *Control* celkem dva procesory, jedním z nich je mikrokontrolér STM32L476 a druhý je koprocesor rádiového modulu ESP32-WROOM. Způsob předávání dat mezi nimi je shodný a též funkce modulu ESP32 je totožná. Jiný zdrojový kód je použit pro mikrokontrolér, který nyní neřídí výstupy, ale přijímá data ze senzoru. Blokový diagram popisující funkci celého softwaru je na obrázku 5.5 na straně 27.

Mikrokontrolér v tomto zařízení vyhodnocuje data přijatá ze senzoru a na základě nich vyhodnocuje gesta. V případě úspěšné detekce, předá data do modulu ESP32 a ten je odešle v patřičném rámci na *Server*. Princip algoritmu rozpoznávání gest vysvětlím v následující části.

5.5.1 Rozpoznávání gest

Základní princip fungování senzoru APDS9960 je založen na čtyřech zónách, jak jsem již zmínil v analýze dostupných komponent. V těchto čtyřech zónách kontinuálně měří míru odraženého infračerveného záření a v momentě, kdy je přesažena mezní hodnota a je tedy vysoce pravděpodobné že se před senzorem pohybuje objekt/dlaň, začne tento průběh ukládat. Jakmile je detekce dokončena, tedy v momentě, kdy míra přijatého infračerveného záření klesne pod mezní hodnotu nebo dojde k naplnění bufferu, signalizuje senzor pomocí přerušení dokončenou detekci. V tomto momentě je možné se senzoru vyčíst čtyři časové průběhy.

Získané časové průběhy je poté možné analyzovat a vyhodnotit z nich, zdali se jedná o časové průběhy odpovídající definovaným gestům. V tomto případě jsem jsem se rozhodl použít jednoduchá gesta definovaná jako pohyb dlaně zdola-nahoru, shora-dolů, zleva-doprava a zprava-doleva a rozeznání těchto gest je relativně jednoduché. Čspoasové průběhy ze všech zón jsou v tomto případě podobné pouze se liší v časovém posunu. Programově je tedy potřeba najít v časových průbězích náběžné hrany a z jejich posunu je možné určit směr pohybu. Například gesto pohybu zprava-doleva bude mít nejprve náběžnou hranu v pravé zóně senzoru, poté v horní a dolní zóně senzoru a jako poslední v levé zóně senzoru.



Obrázek 5.5: Blokový diagram *Sense*

Kapitola 6

Testování parametrů systému

6.1 Odezva systému

Po zprovoznění a zapojení celého systému jsem jako první ověřil odezvu celého systému. Tento parametr je velmi důležitý z praktického důvodu, není přijatelné aby případný uživatel čekal po provedení gesta příliš dlouhou dobu na aktivaci výstupního prvku.

Vzhledem k tomu, že k nastavení výstupů po přijetí příkazů dojte v podstatě okamžitě a samotná detekce gesta je též velmi rychlou operací v řádu milisekund, tak jsem přijal předpoklad, že pro rámcovou představu stačí změřit pouze odezvu na úrovni sítě příkazem ping. Pro rozšíření jsem též změřil odezvu v rámci definovaného komunikačního protokolu, tedy odesláním příkazu a změřením doby do obdržení ACK paketu.

Měření příkazem ping i vyčkáváním na paket ACK jsem provedl celkem 100x pomocí automatického skriptu a následně jsem vyhodnotil minimální, maximální a průměrnou odezvu. Celé měření probíhalo v podmínkách nevytížené sítě, tedy v síti byla celou dobu připojena mimo routeru pouze měřená zařízení a PC, kterým jsem přes SSH přistupoval k serveru. Během testu neprobíhalo žádné význačné stahování nebo nahrávání dat.

	min	**max**	**průměr**
PING z *Sense* na *Server*	10ms	27ms	13ms
PING z *Server* na *Control*	27ms	132ms	78ms
ACK z *Sense* na *Server*	<1ms	<1ms	<1ms
ACK z *Server* na *Control*	58ms	144ms	124ms

Tabulka 6.1: Měření odezvy příkazem ping a vyčkáváním na ACK

Měření pomocí příkazů ping i měřením pomocí ACK mělo poměrně velký rozptyl výsledků. Celkem z měření vyplývají následující poznatky:

- Změřené hodnoty pomocí ping a ACK se výrazně liší jak bylo možné očekávat.
- Odezva serveru je signifikantně rychlejší než odezva koncového zařízení. Tyto rozdíly si vysvětluji rozdílným výpočetním výkonem, potenciálně rozdílnou systémovou implementací odezvy na ping a v neposlední řadě též možným využitím úsporných režimů v zařízení *Control*.

V součtu byla nejhorší možná doba odezvy z bodu *Sense* až do aktivace v zařízení *Control* menší než 200 ms, což je v rámci možností přijatelné.

6.2 Ovládání

Dalším měřením jsem ověřil fungování systému již pohledem uživatele. V tomto měření jsem postupně prováděl různé pohyby dlaní a sledoval jsem odezvu systému ve smyslu očekávaného výstupu. Pro tento test byl systém nakonfigurován tak, aby gesto pohybu směrem nahoru zapnulo osvětlení LED páskem, gesta ve směru doleva a doprava měnila barvu RGB LED pásku z vybraných barev a gesto pohybu směrem dolů vypnulo osvětlení. Pohyb ve všech směrem jsem provedl celkem 100x a a výstupem měření byla celková chybovost systému, kterou je možné vidět v následující tabulce.

Doleva	Doprava	Shora dolů	Nahoru
14%	16%	19%	15%

Tabulka 6.2: Měření chybovosti rozpoznání gesta

Téměř ve všech případech se jednalo o chybu, kdy nebylo gesto vůbec detekováno a pouze v jednom případě bylo detekováno gesto chybné. Změřené hodnoty jsou poměrně vysoké a mnohem akceptovatelnější by byly hodnoty o řád nižší, na druhou stranu vzhledem k tomu, jak jednoduchý algoritmus byl použit, tak lze tento výsledek považovat za uspokojivý. S využitím komplexnějších algoritmů lze očekávat výrazné zlepšení.

Kapitola 7

Finanční stránka realizace

Posledním důležitým cílem této práce je zhodnocení finanční stránky realizace celého systému. Pro tento účel jsem si zvolil modelový příklad domácnosti v bytě 2+1 s oddělenou koupelnou a toaletou. V tomto modelovém bytě jsem zvážil možná rozmístění ovládacích prvků a osvětlovacích prvků a došel jsem k následující sestavě:

- *Sense* - 8 krát (v každé místnosti jednou, obývák a ložnice po dvou)
- *Control* - 8 krát (v každé místnosti jednou, obývák a ložnice po dvou)
- *Server* na Raspberry Pi 4 - 1 krát

Na základě obvyklých cen jsem určil orientační náklady na dílčích částí výroby, které uvádím v následující tabulce:

	DPS	**Součátky**	**Osazení**	**Krabice**	**Jiné**	**Celkem**
Sense	140,-	250,-	200,-	40,-	0,-	630,-
Control	140,-	200,-	100,-	25,-	0,-	465,-
Server	0,-	0,-	0,-	0,-	1200,-	1200,-

Tabulka 7.1: Cenová kalkulace

Po sečtení všech dílčích položek vycházejí rámcové náklady na vybavení jedné domácnosti tímto systémem na téměř 10000,- Rámcová cena po zahrnutí nákladů na vývoj, pozáručního servisu a marže by se reálně mohla pohybovat okolo 15000,- až 20000,- v závislosti na celkovém rozsahu výroby. Je otázkou, zdali je takováto částka úměrná potenciálnímu přínosu tohoto systému, na druhou stranu by ale stále bylo možné cenu podstatně snížit optimalizací použitých komponent.

Kapitola 8
Závěr

8.1 Zhodnocení výsledků

Na základě analýzy současného stavu využití integrovaných senzorů pro rozpoznávání gest, jejich provozních parametrů a funkcí, dále pak analýzy současného stavu využití bezdrátových komunikačních rozhraní, jejich provozních parametrů a funkcí, a analýzy současného stavu využití mikrokontrolerů a embedded Linux platforem, jsem zvolil vhodné prvky k použití v navrhovaném systému.

Následně jsem navrhl a realizoval model tohoto systému spojením tří hlavních komponent – ovládacího rozhraní s funkcí rozpoznávání gest, řídící centrály pro zpracování dat a aktivního členu, který ovládal LED osvětlení. Poté jsem provedl základní testy funkčních vzorků, změřil jsem jejich některé důležité parametry a zhodnotil jejich funkčnost. Na závěr jsem stručně zhodnotil náklady realizace a koncovou cenu v případě realizace tohoto systému pro potenciální obchodní uplatnění.

Celkově považuji tuto realizaci za úspěšnou a předpokládám, že i nadále budu pokračovat s jejím vývojem. Směr tohoto vývoje stručně naznačím v následující doplňující podkapitole.

8.2 Potenciální vylepšení

Pro budoucí vývoj se nabízí několik potenciálních úprav, které jsem buď objevil v závěru tohoto projektu nebo jsem již o nich věděl od počáteční analýzy, ale bylo mimo moje kapacity je plně implementovat. Mezi ně patří:

- Lepší algoritmus rozpoznávání gest - zde se nabízí možnost rozpoznání komplexnějších gest a větší konfigurovatelnost. Pro tyto účely zvažuji otestovat dostupné knihovny pro tvorbu neuronových sítí jako například TensorFlow.
- LoRa - použití rozhraní LoRa místo Wi-Fi by mohlo přinést několik výhod, mimo jiné nižší spotřebu možnost většího dosahu komunikace.
- Návrh krabic na míru - krabice na míru by byla velkým přínosem zejména u zařízení pro rozpoznávání gest, kde bylo třeba sériově vyráběnou krabici výrazně modifikovat kvůli odrazům v zorném poli senzoru.

- Optimalizace součástek - z analýzy nákladů vyplývá, že by bylo možné zařízení po konstrukční stránce zjednodušit a tím i snížit cenu za použité součástky.

Ověření těchto možností a provedení jejich realizace chystám v dohledné době.

Literatura

[1] HOROWITZ, Paul and HILL, Winfield. *The Art of Electronics*. 3rd edition. Massachusetts: Harvard University, 2015. ISBN 978-0-521-80926-9

[2] VOBECKÝ, Jan a ZÁHLAVA, Vít. *Elektronika - Součástky a obvody, principy a příklady*. třetí, rozšířené vydání. Praha: Grada Publishing, 2005. ISBN 978-80-247-1241-3

[3] ZÁHLAVA, Vít. *Návrh a konstrukce desek plošných spojů*. Praha: BEN - technická literatura, 2011. ISBN 978-80-7300-266-4

[4] VILÍM, Petr. *Senzorový systém pro řízení a regulaci*. Praha, 2018. Bakalářká práce. ČVUT v Praze, Fakulta elektrotechnická, Katedra mikroelektroniky.

[5] Rust [online]. Rust Team, 2019. [vid. 15.11.2019]. Dostupné z: `https://www.rust-lang.org/`

[6] Rust Platform Support [online]. [vid. 15.11.2019]. Dostupné z: `https://forge.rust-lang.org/release/platform-support.html`

[7] Global Opportunity Analysis and Industry Forecast, 2014-2022 [online]. [vid. 16.11.2019]. Dostupné z: `https://www.alliedmarketresearch.com/microcontrollers-market`

[8] Microcontroller Market Size, Share and Trends Analysis Report [online]. Grand View Research, 2019. [vid. 16.11.2019]. `https://www.grandviewresearch.com/industry-analysis/microcontroller-market`

[9] Microchip Technology [online]. *ATmega328P*. 2015. [vid. 21.11.2019]. Dostupné z: `http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf`

[10] Microchip Technology [online]. *ATtiny25/45/85*. 2013. [vid. 21.11.2019]. Dostupné z: `http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-2586-AVR-8-bit-Microcontroller-ATtiny25_45_85_Datasheet.pdf`

[11] Microchip Technology [online]. *ATtiny417/817*. 2019. [vid 27.12.2019]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/ATtiny417-817-DataSheet-DS40001901D.pdf

[12] Microchip Technology [online]. *PIC18F2420*. 2008. [vid. 27.12.2019]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/39631E.pdf

[13] STM8 [online]. STMicroelectronics, 2019. [vid. 29.12.2019]. Dostupné z: https://www.st.com/en/microcontrollers-microprocessors/stm8-8-bit-mcus.html

[14] MicroPython [online]. George Robotics Limited, 2019. [vid. 8.11.2019]. Dostupné z: https://micropython.org/

[15] MSP430 ultra-low-power sensing and measurement MCUs [online]. Texas Instruments, 2019. [vid 29.11.2019]. Dostupné z: https://www.ti.com/microcontrollers/msp430-ultra-low-power-mcus/overview.html

[16] STMicroelectronics [online]. *STM32L476xx. Product Specifications*. 2018. [vid. 10.2.2018]. Dostupné z: http://www.st.com/resource/en/datasheet/stm32l476je.pdf

[17] STMicroelectronics [online]. *STM32L031xx. Product Specifications*. 2019. [vid. 29.12.2019]. Dostupné z: https://www.st.com/resource/en/datasheet/stm32l031f6.pdf

[18] STMicroelectronics [online]. *STM32G031xx. Product Specifications*. 2019. [vid. 29.12.2019]. Dostupné z: https://www.st.com/resource/en/datasheet/stm32g031j6.pdf

[19] RF Solutions [online]. *Miniature Smart RF Transceivers. Datasheet*. 2018. [vid. 23.7.2019]. Dostupné z: https://www.rfsolutions.co.uk/downloads/1556008678DS-ZETAPLUS-6.pdf

[20] RF Solutions [online]. *RF LoRa. LongRange Transceiver*. 2018. [vid. 23.7.2019]. Dostupné z: https://datasheet.octopart.com/RF-LORA-868-SO-RF-Solutions-datasheet-62341029.pdf

[21] Bluetooth Mesh Models - A Technical Overview [online]. [vid. 30.12.2019] Dostupné z: https://www.bluetooth.com/bluetooth-resources/bluetooth-mesh-models/

[22] Why ZigBee? [online]. Zigbee Alliance, 2019. [vid. 30.12.2019]. Dostupné z: https://zigbeealliance.org/why-zigbee/

[23] Nordic Semiconductor [online]. *nRF52832 product brief*. 2019. [vid. 15.10.2019]. Dostupné z: https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52832

[24] STMicroelectronics [online]. *Dual-core, multi-protocol wireless STM32WB microcontrollers*. 2019. [vid. 15.10.2019]. Dostupné z: https://www.st.com/en/microcontrollers-microprocessors/stm32wb-series.html

[25] STMicroelectronics [online]. *Sub-GHz (868 or 915 MHz) low power programmable RF transceiver modules*. 2019. [vid. 15.10.2019]. Dostupné z: https://www.st.com/content/st_com/en/products/wireless-transceivers-mcus-and-modules/sub-1ghz-rf/spsgrf.html

[26] STMicroelectronics [online]. *Very low power application module for Bluetooth Smart v4.2. Product specifications*. 2019. [vid. 15.10.2019]. Dostupné z: https://www.st.com/content/st_com/en/products/wireless-transceivers-mcus-and-modules/bluetooth-bluetooth-low-energy/spbtle-1s.html

[27] Espressif Systems [online]. *ESP32-WROOM-32. Datasheet*. 2019. [vid. 16.12.2019]. Dostupné z: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

[28] Farnell.com [online]. [vid. 30.12.2019]. Dostupné z: https://www.farnell.com

[29] Testing the WiFi range of ESP32 [online]. [vid. 31.12.2019]. Dostupné z: https://blog.honzamrazek.cz/2017/08/testing-the-wifi-range-of-esp32/

[30] Extremetech.com [online]. [vid. 31.12.2019]. Dostupné z: https://www.extremetech.com/wp-content/uploads/2016/07/Raspberry-Pi-3-Model-B.jpg

[31] Beagle Bone AI [online]. [vid. 31.12.2019]. Dostupné z: https://beagleboard.org/ai

[32] APDS-9960 [online]. [vid. 1.9.2019]. Broadcom Limited. https://www.broadcom.com/products/optical-sensors/integrated-ambient-light-and-proximity-sensors/apds-9960

[33] Sparkfun APDS-9960 Arduino Library [online]. [vid. 1.9.2019]. Github. https://github.com/sparkfun/SparkFun_APDS-9960_Sensor_Arduino_Library/tree/V_1.4.2

[34] KiCad Developers Team. *KiCad EDA, ver. 5.1.6* [online]. Květen 2020. [přístup 25.5.2020]. Dostupné z: http://kicad-pcb.org/ [Požadavky na systém: procesor Pentium IV, Athlon nebo novější, grafická karta s OpenGL 2.1 nebo novější, operační systém Windows 7, 8, 10 nebo GNU/Linux, 5 GB volného místa na disku, 1024 MB operační paměť]

[35] TME.eu [online]. [vid. 15.4.2020]. https://www.tme.eu/cz/details/10007.5/univerzalni-krabicky/teko/

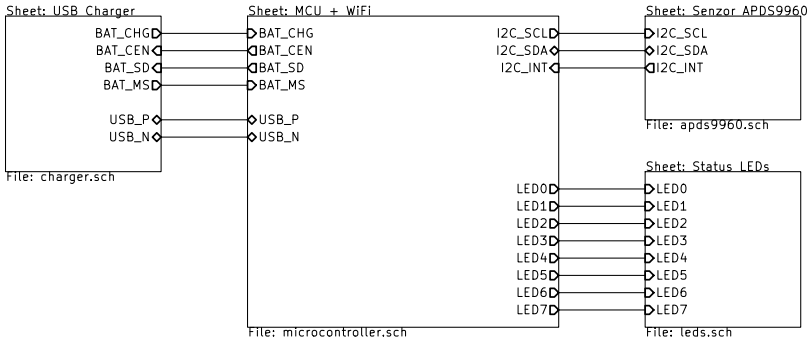
[36] TME.eu [online]. [vid. 15.4.2020]. https://www.tme.eu/cz/details/mini-25-l/montazni-krabice/spelsberg/31090801/

Seznam zkratek a symbolů

DPS	Deska Tištěných Spojů
ESD	Electrostatic Discharge (Elektrostatický výboj)
HAL	Hardware Abstraction Layer
I2C	Inter-Integrated Circuit (sériová komunikační sběrnice)
IoT	Internet of Things (Internet věcí)
LDO	Low Drop Out regulator (lineární stabilizátor s nízkým poklesem napětí)
LED	Light Emitting Diode (světlo vyzařující dioda)
MCU	Micro Controller Unit (Mikrokontrolér)
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
NB-IoT	Narrow Band IoT (Úzkopásmové IoT)
PCB	Printed Circuit Board (deska tištěných spojů)
PWM	Pulse Width Modulation (pulzně šířková modulace)
QFN	Quad Flat No-leads (typ pouzdra součástek)
REPL	Read, Eval, Print, Loop (Interaktivní konzole jazyka Python)
SMPS	Switched Mode Power Supply (spínaný zdroj)
SPI	Serial Peripheral Interface (sériové komunikační rozhraní)
SSH	Secure Shell (protokol pro vzdálený přístup)
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WB-IoT	Wide Band IoT (Širokopásmové IoT)

Přílohy

Příloha A1 - funkční schéma Sense

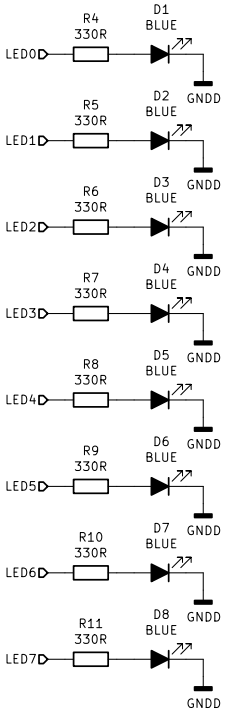


Petr Vilím

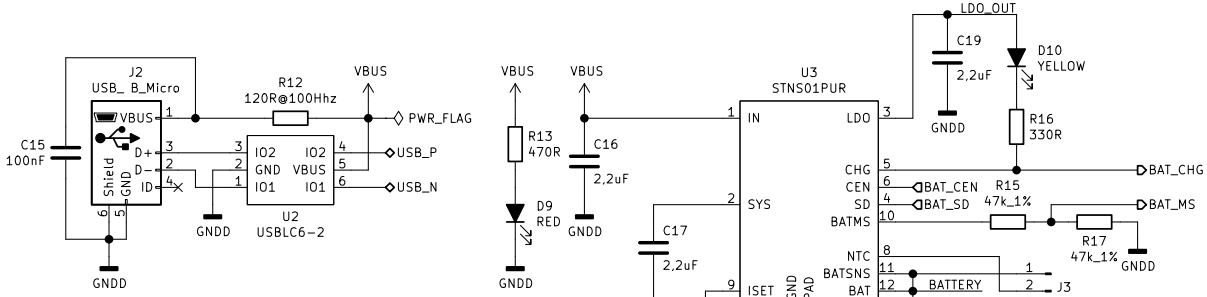
Sheet: /
File: gesture_sense.sch

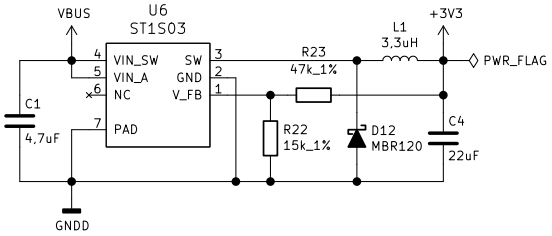
Title: Gesture Sense

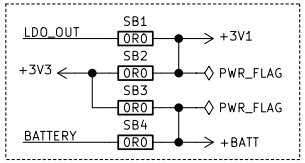
Size: A4	Date: 2020-04-28	**Rev: 0.4**
KiCad E.D.A.	kicad 5.1.6-c6e7f7d87ubuntu18.04.1	Id: 1/5



Petr Vilím		
Sheet: /Status LEDs/ File: leds.sch		
Title: Gesture Sense		
Size: A4	Date: 2020-04-28	**Rev: 0.4**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 2/5







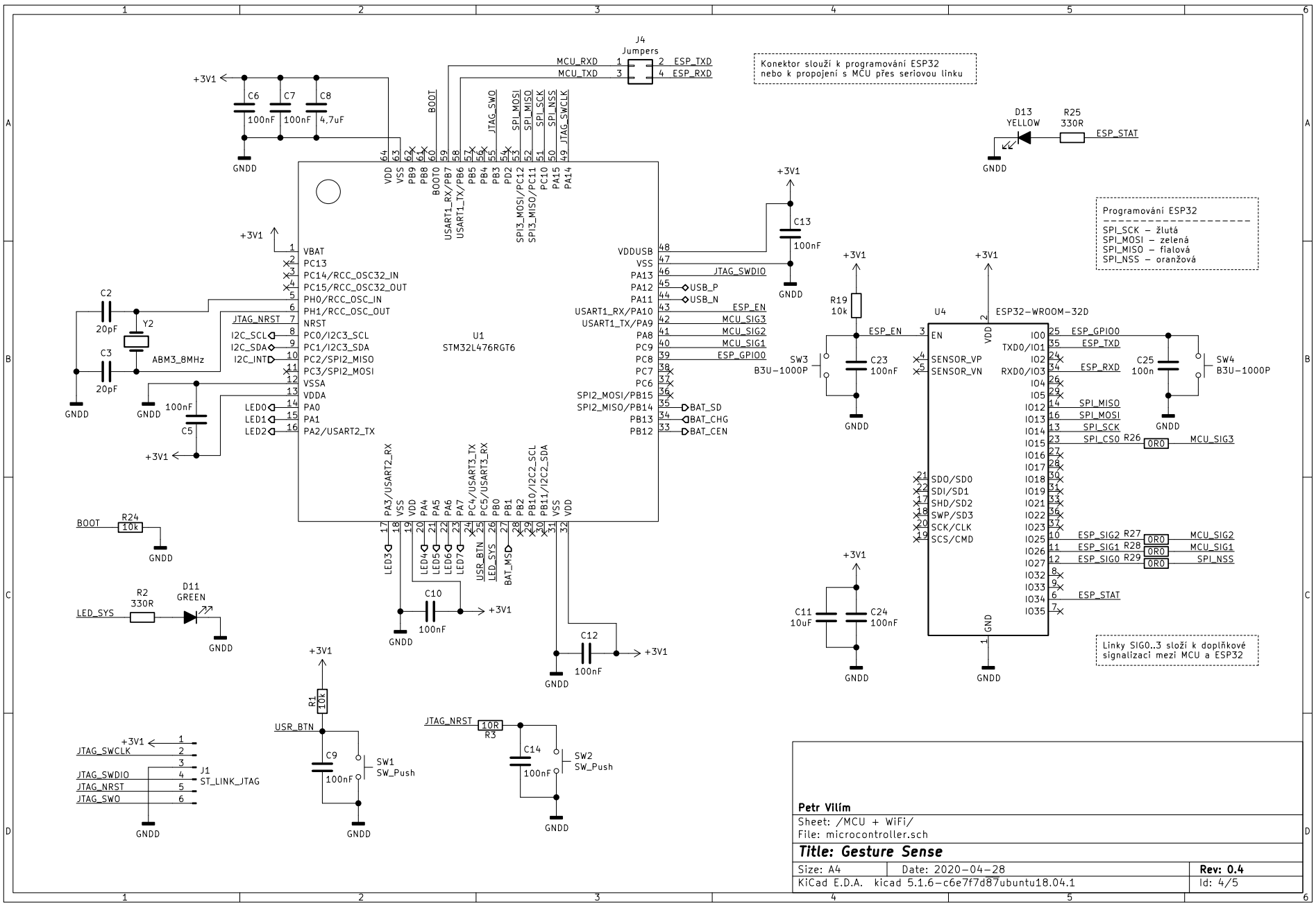
POWER SOURCE SWITCH

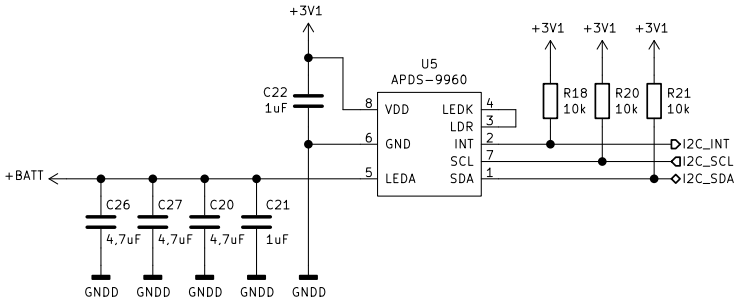
Place SB1 and SB4 for battery supply
Place SB2 and SB3 for direct USB supply

NOTE **!**
If CEN is low, charger is disabled, including the LDO

NOTE **!!**
It might be better to not use SD, CEN and CHG pins and leave them floting
(they include pull-up/down respective resistors)

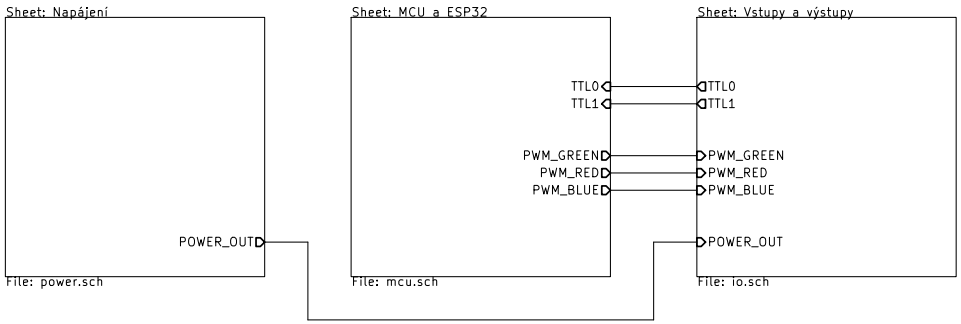
Petr Vilím		
Sheet: /USB Charger/ File: charger.sch		
Title: Gesture Sense		
Size: A4	Date: 2020-04-28	**Rev: 0.4**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 3/5


J4
Jumpers
MCU_RXD
MCU_TXD
ESP_TXD
ESP_RXD
Konektor slouží k programování ESP32
nebo k propojení s MCU přes seriovou linku
U1
STM32L476RGT6
U4
ESP32-WROOM-32D
D13
YELLOW
R25
330R
ESP_STAT
Programování ESP32
SPI_SCK – žlutá
SPI_MOSI – zelená
SPI_MISO – flalová
SPI_NSS – oranžová
Linky SIG0..3 složí k doplňkové
signalizaci mezi MCU a ESP32
ABM3_8MHz
ST_LINK_JTAG
SW_Push
B3U-1000P
Petr Vlím
Sheet: /MCU + WiFi/
File: microcontroller.sch
Title: Gesture Sense
Size: A4
Date: 2020-04-28
Rev: 0.4
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1
Id: 4/5

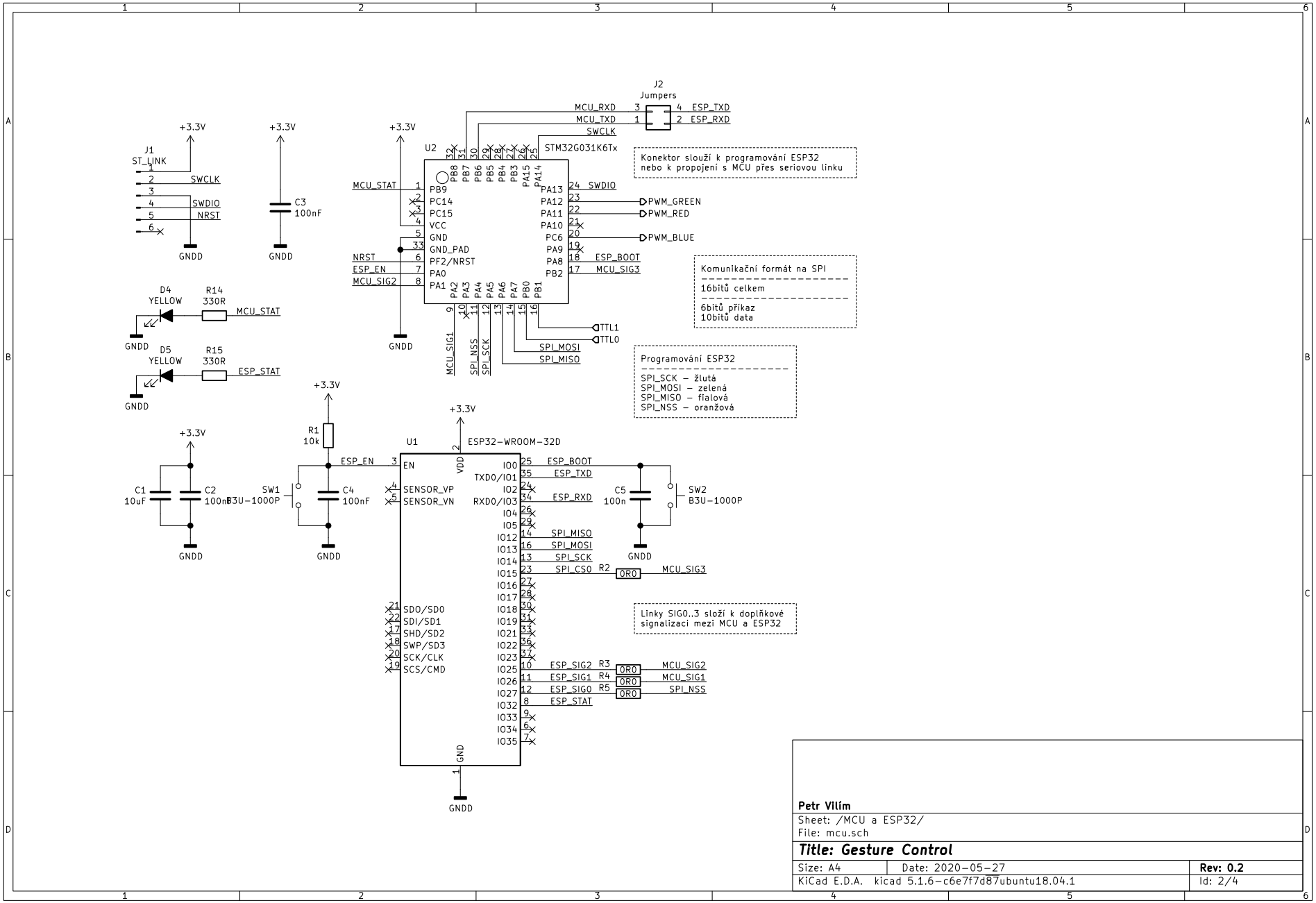


Petr Vilím		
Sheet: /Senzor APDS9960/ File: apds9960.sch		
Title: Gesture Sense		
Size: A4	Date: 2020-04-28	**Rev: 0.4**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 5/5

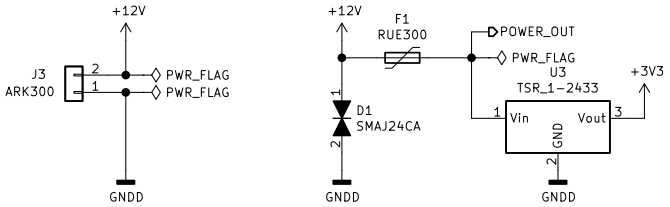
Příloha A2 - funkční schéma Control



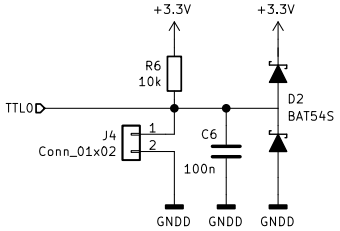
Petr Vilím		
Sheet: / File: gesture_control.sch		
Title: Gesture Control		
Size: A4	Date: 2020-05-27	**Rev: 0.2**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 1/4

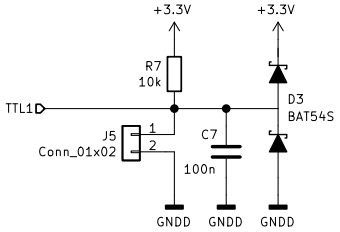


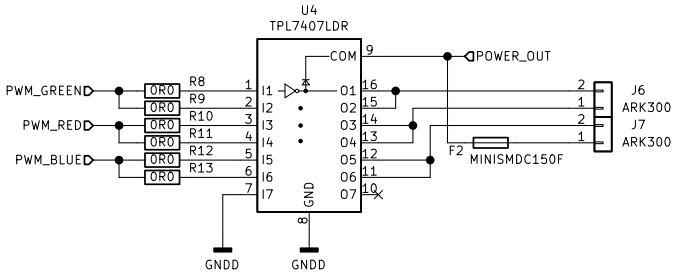
Petr Vilím		
Sheet: /MCU a ESP32/		
File: mcu.sch		
Title: Gesture Control		
Size: A4	Date: 2020-05-27	**Rev: 0.2**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 2/4



Petr Vilím		
Sheet: /Napájení/ File: power.sch		
Title: Gesture Control		
Size: A4	Date: 2020-05-27	**Rev: 0.2**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 3/4

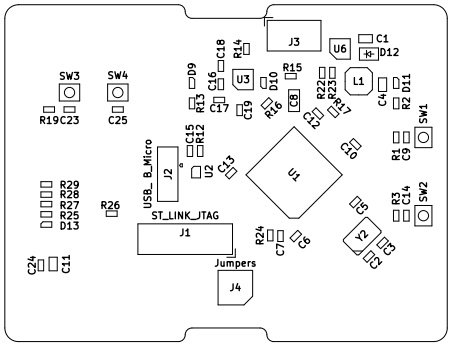






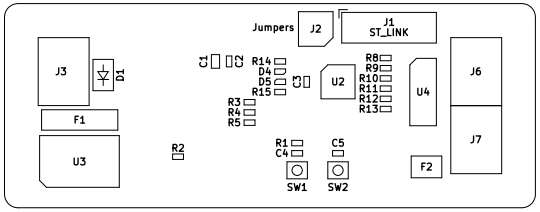
Petr Vilím		
Sheet: /Vstupy a výstupy/ File: io.sch		
Title: Gesture Control		
Size: A4	Date: 2020-05-27	**Rev: 0.2**
KiCad E.D.A. kicad 5.1.6-c6e7f7d87ubuntu18.04.1		Id: 4/4

Příloha B1 - rozložení DPS Sense



Petr Vilim		
Sheet: File: gesture_sense.kicad_pcb		
Title: Gesture Sense		
Size: A4	Date: 2020-04-20	Rev: 0.4
KiCad E.D.A. kicad 5.1.6		Id: 1/1

Příloha B1 - rozložení DPS Control



Petr Vilim		
Sheet: File: gesture_control.kicad_pcb		
Title: Gesture Control		
Size: A4	Date: 2020-04-12	Rev: 0.1
KiCad E.D.A. kicad 5.1.6		Id: 1/1

Příloha C - zdrojové kódy

Vzhledem k rozsahu zdrojového kódu jej zde neuvádím. Všechny zdrojové kódy jsou uloženy na přiloženém CD. Pro přehled zde uvádím popis jednotlivých zdrojových kódů:

- gesture_sense.py - zdrojový kód zařízení *Sense*
- gesture_control.py - zdrojový kód zařízení *Control*
- gesture_server.py - zdrojový kód pro *Server*
- config.txt - konfigurační soubor sdílený mezi zařízeními obsahující základní údaje pro připojení k síti a vzájemné komunikaci
- link_command.pickle - soubor obsahují vazby mezi vstupy a výstupy systému, používá jej ke své funkci *Server*
- link_command_generator.py - skript generující soubor link_command.pickle