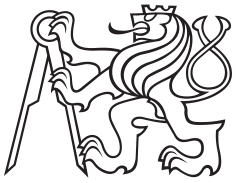


České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Experimentální senzorové moduly pro tvorbu hraček a domácího monitorování

David Juřík

Vedoucí práce: Ing. Petr Novák Ph.D.
Květen 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Juřík** Jméno: **David** Osobní číslo: **466013**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra kybernetiky**

Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Experimentální senzorové moduly pro tvorbu hraček a domácího monitorování

Název bakalářské práce anglicky:

Experimental Sensor Modules for Building Toys and Home Monitoring

Pokyny pro vypracování:

Nejen pro starší, samostatně žijící a postižené lidi jsou vhodné senzory pro monitorování / dohled nad jejich obecným stavem (pohyb po bytě, CO2, teplota, …), ale rovněž předměty / hračky kterými lze částečně měřit i jejich duševní / fyzický stav (třes ruky, srdeční aktivitu, pohyblivost, …). Pro rychlý vývoj takovýchto senzorových zařízení (zejména hraček) jsou velmi vhodné před-připravené experimentální moduly obsahující různé komunikace a možnost okamžitého připojení celkem dostupných typů senzorů.

1. Prostudujte podobné existující projekty / moduly a zhodnoťte jejich přínos pro tvorbu zmíněných experimentálních senzorových zařízení a tvorbu speciálních měřících hraček / předmětů.
2. Navrhněte několik experimentálních HW zařízení obsahující různé procesory (např. ARM, ESP32, …), využívající různé komunikace (např. BlueTooth, WiFi, …) pro přenos dat do PC a poskytující snadné připojení vybraných senzorů (detekce pohybu, teplota, CO2, …). Dbejte na jejich snadnou konfiguraci.
3. Vytvořte prototypy několika těchto HW zařízení. Dále SW aplikaci (PC/mobil) pro testování, ukládání a zákl. zobrazení poskytovaných dat a sloužící jako základ pro experimentování s těmito HW moduly.
4. Zvažte rovněž další možnosti jako např.: detekce spolehlivosti přenosu (výpadky) a dat (stálost hodnoty) ze senzorů, využití 3D tisku pro zapouzdření těchto experimentálních zařízení a další vlastní návrhy.

Seznam doporučené literatury:

[1] WWW stránky obdobných projektů / zařízení
[2] Carmine Noviello, Mastering STM32, Italy, 2018 (kapitoly dostupné na WWW, nebo podobné)
[3] Knížky / WWW stránky projektů pro Arduino, ESP32, případně dalších podle použití
[4] Mark Price, C# 7.1 and .NET Core 2.0, Packt, 2017 (nebo podobné pro C# .NET Core)
[5] PDF použitých komponent ARM / senzorů / atd.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Petr Novák, Ph.D., Analýza a interpretace biomedicínských dat FEL

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

Datum zadání bakalářské práce: **10.01.2020** Termín odevzdání bakalářské práce: **22.05.2020**

Platnost zadání bakalářské práce: **30.09.2021**

Ing. Petr Novák, Ph.D.	doc. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

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

. _____ _____

Datum převzetí zadání Podpis studenta

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Petru Novákovi Ph.D. za jeho pomoc a věcné připomínky při řešení problematiky spojené se závěrečnou prací.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 20. května 2020

podpis autora práce

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, May 20, 2020

Signature

Abstrakt

Obsahem práce je návrh a ověření možnosti rychlé tvorby experimentálních senzorových modulů určených převážně pro různé speciální hračky nebo dohled v domácím prostředí. Hlavní cíl je kladen na snadnou konfiguraci a modularitu celého systému. Při tvorbě tohoto řešení jsou využity cenově dostupné vývojové desky STM32-NUCLEO a moduly ESP8266 (ESP32), několik vzorových senzorů a rovněž různé druhy běžně používaných komunikačních kanálů jako jsou Bluetooth Low Energy, WiFi nebo USART.

Klíčová slova: STM32-L152re, ESP8266, ESP32, Bluetooth Low Energy, sběr dat ze senzorů, chytrá domácnost na míru

Vedoucí práce: Ing. Petr Novák Ph.D.

Abstract

This thesis describes design and testing of a fast development of experimental sensor modules mostly intended for diverse toys or home automation. The main goal is a simple configuration and modularity of the whole solution. Various inexpensive microcontrollers such as STM32-NUCLEO and ESP8266 (ESP32) as well as sample sensors are used and various types of communication channels such as Bluetooth Low Energy, WiFi of USART are implemented.

Keywords: STM32-L152re, ESP8266, ESP32, Bluetooth Low Energy, collecting data from sensors, tailor-made smart home

Title translation: Experimental sensor modules for building toys and home monitoring

Obsah

Obrázky

Kapitola 1

Úvod

V dnešní době rychlého technologického rozvoje a pronikání elektronických systémů do všech sfér lidského života dosahuje velké popularity Internet věcí (IoT - Internet Of Things). Jedná se o vzájemné propojení senzorů s řídicí jednotkou komunikující nejčastěji pomocí bezdrátových technologií (WiFi/BLE). Takto sestavené systémy se dají využít nejen k zalévání zahrady, vytápění domácnosti, zabezpečení majetku, ale rovněž k monitorování zdravotního stavu člověka atd. Komerčně dostupné produkty bývají často sice velmi precizní, ale současně drahé a ne vždy dostatečně univerzální, a již těžko použitelné pro vědecké nebo dokonce experimentální účely. Samozřejmě existují částečné možnosti tvorby systému i „na míru“. Výhodou těchto kitů je rychlé zapojení a snadné programování, avšak kvůli této „jednoduchosti“ bývá jejich schopnost a flexibilita dosti omezena. Tato práce se tedy bude zabývat návrhem a tvorbou vývojových komponent, které budou snadno konfigurovatelné a použitelné nejen pro chytré domácnosti a hračky, ale rovněž tak i pro experimentální využití v oblasti vědy, výzkumu a medicíny. Celková činnost vytvořeného návrhu bude demonstrována pomocí vybrané skupiny modulů a senzorů.

Kapitola 2

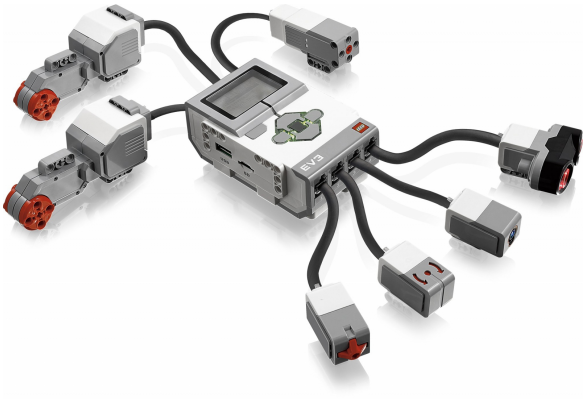
Existující senzorické projekty

Úvodem budou stručně představeny a zhodnoceny některé existující modulární (senzorické) systémy, zejména ty komerčně dostupné a bude zdůrazněno, proč jsou nebo nejsou vhodné pro použití ve vědecké a experimentální sféře.

2.1 Komerční stavebnice

2.1.1 LEGO MINDSTORMS EV3 s Bluetooth komunikací

Lego Mindstorms je populární stavebnice jak pro děti, tak i pro dospělé, nabízející množství součástek/komponent, díky kterým je možné sestavit nejen jednoduchou hračku, ale i složitější robotický systém. Součástí této stavebnice jsou senzory teploty, vzdálenosti, barev atd. Sada dále obsahuje programovatelnou řídicí jednotku do které je možno zapojit 4 vstupní senzory a 4 výstupní porty pro motory. Řídicí jednotka je vybavena procesorem ARM 9 na kterém je nainstalovaný operační systém založený na Linuxu. Dále obsahuje display, speaker a je programovatelná pomocí jazyků C nebo JAVA. Nechybí ani bezdrátová komunikace pomocí Bluetooth umožňující řídit systém vzdáleně.



Obrázek 2.1: Řídicí jednotka Lego Mindstroms s připojenými senzory a motory. [11]

Velkou nevýhodou je připojení pouze 4 senzorů současně. Dále lze připojit pouze senzory přímo určené pro stavebnici Lego Mindstroms, jenž nejsou nikterak přesné. Stavebnice je vhodná pro výrobu jednoduchých domácích systémů a hraček, ale pro využití při experimentování a ve vědě je však nedostačující.

2.1.2 LittleBits

LittleBits je modulární stavebnice určená rovněž pro děti i dospělé. Každý senzor, motor nebo jakákoli další komponenta je umístěna na obdélníkové podložce. Ta je vybavena magnety tak, aby každý díl jednoduše zapadl do jiného. Dále je k dispozici blok CloudBit umožňující připojit celé zařízení k internetu. Přes mobilní aplikaci nebo webový prohlížeč s využitím služby IFTTT (If This Then That) lze vytvořit jednoduché podmínky. Na ty reaguje buď přímo samotné zařízení, případně je uživatel informován pomocí SMS zprávy nebo E-mailem. Odesláním příkazů prostřednictvím SMS (e-mailu) může zařízení reagovat podle potřeby – komunikace je možná oběma směry.

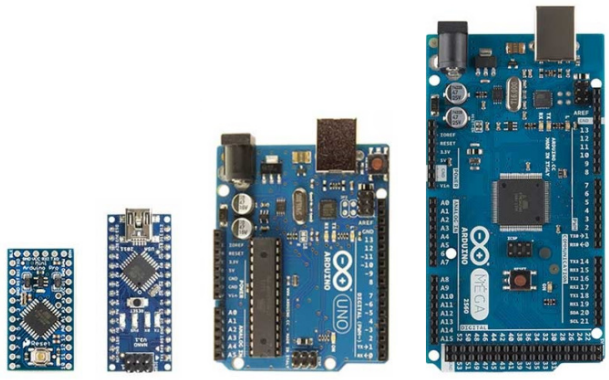
Výhodou tohoto systému je vcelku propracované a jednoduché připojení přes internet. Bohužel podmínky a větvení programu jsou velmi omezené. Dále je možné využít pouze senzorů speciálně vytvořených pro tuto stavebnici – lze tedy měřit jen některé veličiny. Z těchto důvodů není stavebnice nikterak vhodná pro vědecké a experimentální účely.

Obrázek 2.2: Stavebnice LittleBits. [12]

2.2 Výukové (vývojové/experimentální stavebnice)

2.2.1 ARDUINO projekty a klony

Arduino je open-source platforma vytvořená v Itálii. Jedná se o vývojovou desku programovatelnou pomocí jazyka Arduino, který je zcela založen na C/C++. Většina komerčně dostupných tzv. Arduino-Boardů využívá 8-bitové procesory ATmega od firmy Atmel. V posledních letech však přibývají i desky Arduino založené na 32-bitových procesorech typu ARM a dalších. V závislosti na typu procesoru má Arduino k dispozici vstupně-výstupní piny, díky kterým lze připojit velké množství senzorů a externích zařízení. Ty mohou být připojeny rovněž pomocí standardních sběrnic, jako SPI, I2C, OneWire a dalších.

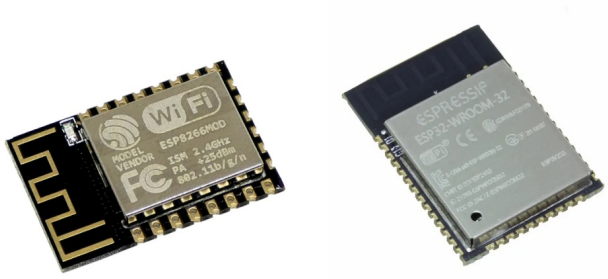


Obrázek 2.3: 8-bitová rodina Arduino – Arduino Mini, Mikro, Uno a Mega (zleva od jednoduššího procesoru). [13]

Asi největší nevýhodou projektu Arduina je chybějící debugger, tedy zcela chybí možnost ladit program za běhu na úrovni použitého procesoru, což může být pro složitější projekty velký problém. Většina originálních desek Arduino je založena na 8-bitových procesorech, které jsou v dnešní době již poněkud pomalé. Z výše uvedených důvodů je toto originální Arduino nedostačující pro komplexnější a profesionální projekty. Existují však tzv. klony, které tyto nedostatky do jisté míry odstraňující.

2.2.2 ESP8266 (WEMOS D1 MINI PRO)/ESP32 (NODEMCU)

Mezinárodní společnost Espressif Systems se sídlem v Šanghaji se zabývá produkcí a vývojem nízkoenergetických výkonných mikrokontrolérů s integrovanou Wi-Fi a Bluetooth technologií pro užití především ve sféře Internetu věcí. Jejich nejznámějšími produkty jsou populární čipy ESP8266 a ESP32, kde ESP32 je vylepšeným následníkem EP8266 (obsahuje více vstupně-výstupních pinů, dedikované jádro určené výhradně pro bezdrátovou komunikaci atd.). Architektura je založena na 32-bitových procesorech Xtensa od firmy Tensilica. Procesor je integrován přímo v modulu společně s velkou (až 16MB) pamětí FLASH a RAM. Dále obsahují integrovaný hardwarový JTAG interface umožňující ladění programu pomocí debuggeru. Ten však není součástí integrovaných obvodů a musí se k mikrokontroléru externě připojit. Vhodným debuggerem je ESP-prog vyvíjený a prodávaný přímo společností Espressif Systems.



Obrázek 2.4: Moduly ESP8266 (vlevo) a ESP32 (vpravo).[14]

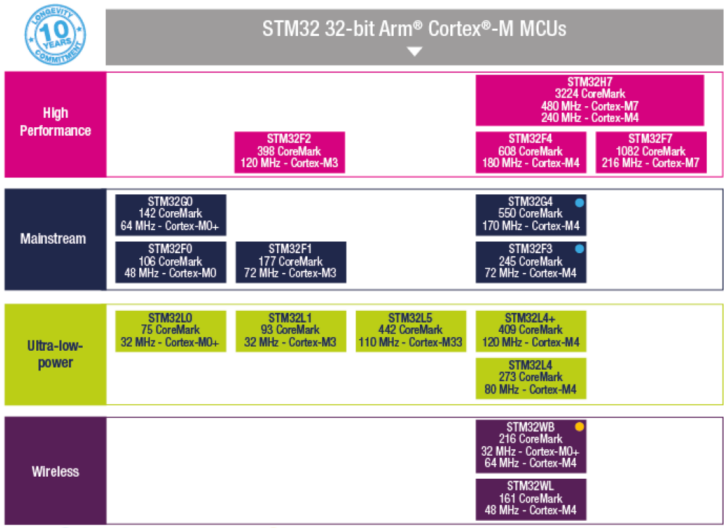
Vývojové desky NODEMCU a WEMOS D1 MINI PRO vybavené výše uvedenými čipy jsou velmi rozšířené mezi uživateli a jsou vhodnými kandidáty na použití i v této práci. Uvedené desky navíc obsahují regulátor napětí (oba čipy pracují na voltáži 3.3V), USB-UART převodník, tlačítko RESTART a další vhodné komponenty. Tyto desky lze rovněž pořídit velmi levně jako tzv. Arduino-Boardy.

2.3 Univerzální moduly s ARM procesory

ARM Holdings je britská společnost vyvíjející instrukční sadu a architekturu pro procesory nazvané ARM (Advanced Risc Machine). Zabývá se výhradně návrhy procesorů a ty pod licencí poskytuje mnoha výrobcům. V roce 2017 byly ARM procesory zastoupeny v 75 % všech světových mobilních zařízeních. Mnoho firem tedy na základě zakoupené licence vyrábí různě modifikované ARM procesory (standardní jádro + vlastní periferie). Ty jsou často zaměřeny na nějaké parametry jako například na spotřebu, výkon, preciznost AD převodníků atd. Architektura procesoru typu ARM je založena na množství (i speciálních) velmi často samostatně pracujících periferiích a jádře vykonávajícím instrukce zadaného programu pro obsluhu těchto periférií.

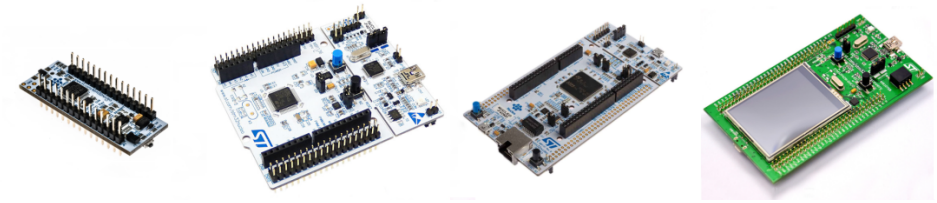
2.3.1 ST Microelectronics - STM32 NUCLEO/Discovery

Významným výrobcem a prodejcem produktů založených na ARM architektuře je společnost ST Microelectronics. Ta nabízí (mimo zcela vlastních 8-bitových STM8) 32-bitové miktokontroléry s označením STM32xxxxx. Ty jsou pro vývojáře a zejména experimenty velmi dostupné pomocí cenově příznivých vývojových desek nazvaných NUCLEO nebo Discovery, a to v různých verzích a provedeních, jak je patrné z obrázku 2.5.



Obrázek 2.5: Komerčně dostupné vývojové desky NUCLEO rozdělené podle jejich předností. [15]

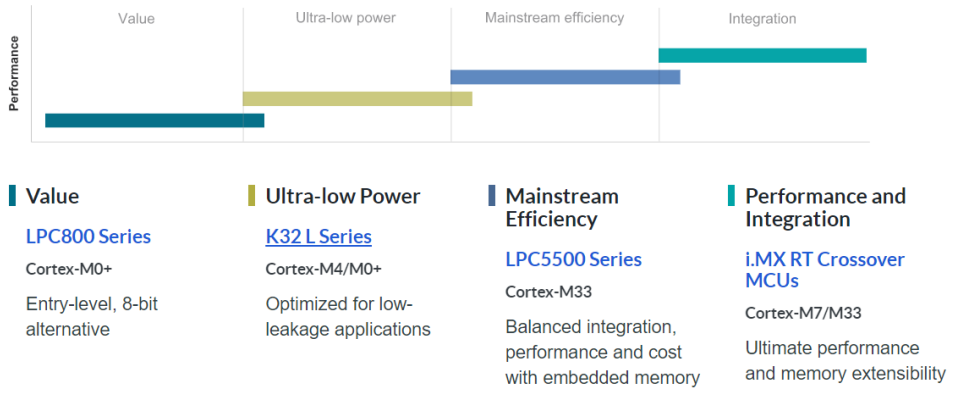
Velkou výhodou těchto miktokontorolérů je jejich velká rozmanitost a snadná dostupnost, například řada F – Fast, řada L – LowPower, řada G - GeneralPurpose. Dále tyto procesory ARM nabízejí přechod do různých šetřících módů, při kterých jsou jádro a periférie pouze zastaveny nebo dokonce zcela uspány, což se projeví v úspoře elektrické energie. Společnost ST Microelectronics pro některé své produkty garantuje 10 Years Longevity Commitment, tedy zaručuje prodej a podporu pro své vybrané produkty minimálně po dobu 10 let. Velkou výhodou již zmíněných desek NUCLEO a Discovery je možnost ladění programu pomocí debuggeru (ST-link nebo J-link) integrovaného přímo na vývojové desce. Po odladění aplikace lze část obsahující programátor snadno odlomit a využít tak pouze naprogramovaný procesor s periférie mi například přímo vložený do cílového experimentálního zařízení. Z výše uvedených důvodů se tedy desky s procesory od ST Microelectronics zdají být velmi vhodné pro experimenty a vědecké účely při vytváření systému, jenž je obsahem této práce. Navržení těchto desek a celkový ekosystém je dobrým základem pro využití nejen v profesionální sféře, ale i ve vědě, medicíně a průmyslu. Některé procesory od STM jsou rovněž ve formátu jako tzv. Arduino-Boardy.



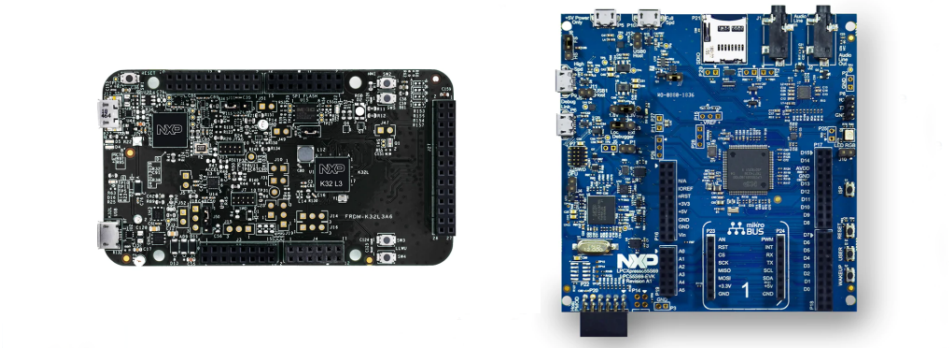
Obrázek 2.6: Vývojové desky STM32 - NUCLEO32, NUCLEO64, NUCLEO144 a Discovery (zleva od jednoduššího procesoru). [16]

2.3.2 NXP Semiconductor

Významným producentem ARM procesorů je americko-holandská společnost NXP Semiconductors N.V. s duálním sídlem v americkém Austinu (Texas) a holandském Eindhovenu. Zabývá se vývojem a produkcí široké sféry technologických zařízení a řešení – automobilový průmysl, mobilní technologie, chytré domovy, chytrá města, komunikační infrastruktura a jiné. Dále nabízí vlastní vývojové desky ve 4 různých sériích. Jádrem všech produktů jsou procesory typu ARM.



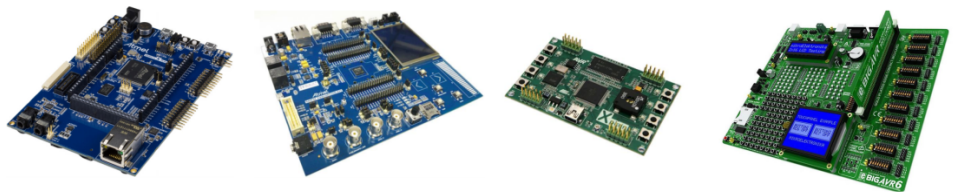
Obrázek 2.7: Čtyři vývojové série od NXP Semiconductors. [17]



Obrázek 2.8: Vývojové mikrokontroléry od společnosti NXP Semiconductors. [18]

2.3.3 Microchip Technology Inc. (Atmel)

Americká společnost Microchip Technology Inc. se sídlem v Chandleru (Arizona) se zabývá vývojem mikrokontrolérů a z nich tvořených vestavěných systémů a komunikačních zařízení. Navrhuje a vyrábí zcela vlastní 8-bitové a 32-bitové procesory pod označením PIC a AVR, ale je rovněž i velkým producentem ARM procesorů. (Firma Microchip v roce 2019 koupila firmu Atmel.)



Obrázek 2.9: Vývojové desky ATSAMV71-XULT (ARM CORTEX - M7), SAM4E-EK (ARM CORTEX – M4), Evaluation kit ATAVRXPLAIN (AVR - ATxmega128A1) a BIGAVR6 (AVR - ATmega128) (zleva doprava). [19]

Kapitola 3

Cíle práce

Bakalářská práce je zaměřena na návrh, vývoj a snadnou tvorbu univerzálních senzorických modulů s využitím různých typů komunikací pro experimentální a vědecké účely. Tyto moduly budou formou zdrojového kódu v jazyce C snadno konfigurovatelné a rozšiřitelné i pro další senzory. Vzájemně budou komunikovat pomocí různých technologií, jako jsou Bluetooth LE (Low Energy), WiFi nebo USART. Takto předpřipravené moduly umožní rychlé sestavení systému na míru s použitím potřebných senzorů a komunikací, čímž budou velmi vhodné zejména v oblasti vědy a medicíny. Hlavní cíle této práce jsou stanoveny následovně:

- Prostudujte podobné existující projekty/moduly a zhodnoťte jejich přínos pro tvorbu zmíněných experimentálních senzorových zařízení a tvorbu speciálních měřících hraček/předmětů.
  - V kapitole 2 byly podrobněji představeny některé existující projekty. Důraz byl kladen na vysvětlení, proč jsou, nebo nejsou tyto moduly vhodné pro užití ve vědecké a experimentální sféře.
- Navrhněte několik experimentálních HW zařízení obsahujících různé procesory (např. ARM, ESP32, ...), které využívají různé komunikace (např. BlueTooth, WiFi, USART, USB, ...) pro přenos dat do PC a poskytující snadné připojení vybraných senzorů (detekce pohybu, teplota, CO2, ...). Dbejte na jejich snadnou konfiguraci.
  - V kapitole 4 bude představen obecný návrh takovéhoto experimentálního zařízení. Obsaženo je jednoduché blokové schéma, bez konktrétních názvů miktokontrolérů nebo senzorů.
- Vytvořte prototypy několika těchto HW zařízení. Dále SW aplikaci (PC/mobil) pro testování, ukládání a základní zobrazení poskytovaných dat sloužící jako základ pro experimentování s těmito HW moduly.
  - V kapitole 5 budou uvedeny konkrétní hardwarové komponenty využité na tvorbu tohoto experimentálního zařízení. Kapitola 6 představí propojení komponent s mikrokontroléry. Důraz byl i zde kladen na snadnou konfiguraci senzorů.

- Zvažte rovněž další možnosti jako např.: detekce spolehlivosti přenosu (výpadky) a dat (stálost hodnoty) ze senzorů, využití 3D tisku pro zapouzdření těchto experimentálních zařízení a další vlastní návrhy.
    - V kapitole 7 bude představen modelovací software na 3D objekty a zmíněny kritické parametry pro správný 3D tisk. Dále zde budou zachyceny výsledné modely po dokončeném tisku.

Tato bakalářská práce se nevěnuje například těmto tématům:

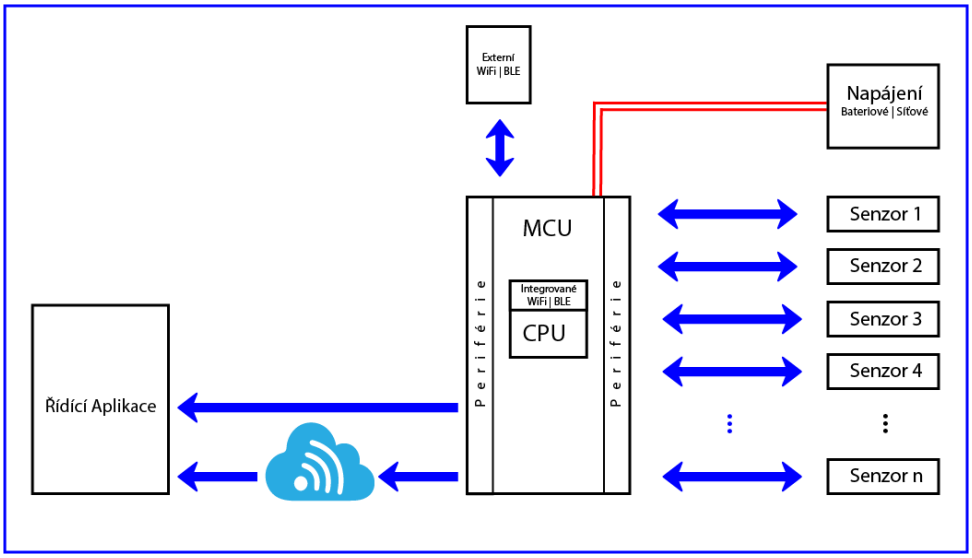
- Konečnému návrhu zařízení (dokumentace, popisy, ...).
- Zabezpečení BLE/WiFi komunikace.
- Výdrži baterie experimentálních modulů.

Kapitola 4

Obecný návrh

Tato kapitola představí obecný návrh zařízení bez konkrétní implementace nebo názvů komponent. Dále představí hlavní programovou smyčku ve formě vývojového diagramu.

Základem projektu je v podstatě jakýkoli mikrokontrolér (ARM-based, ESP, ...) s periferiemi (GPIO, ADC, ...), ke kterému je možné připojit senzory pomocí různých typů sběrnic (I2C, SPI, ...). Často je nutné mikrokontrolér ještě vybavit dodatečným modulem pro odeslání dat, tedy pokud není integrován přímo v použitém procesoru. Dále je nezbytné zajistit napájení daného systému – bateriové, nebo síťové. Mikrokontrolér je buď stále v aktivním stavu, nebo se podle potřeby pouze periodicky probouzí z důvodu odesílání dat vyčtených ze senzorů. Může také zareagovat na asynchronní událost, tedy probudit se dříve, než předem nastavený časový interval vyprší a odeslat tak data okamžitě. V době nečinnosti mohou být tedy periférie a jádro mikrokontroléru uspány zejména z důvodu šetření elektrické energie.



Obrázek 4.1: Blokové schéma experimentálního modulárního zařízení.

4.1 Technické požadavky

Pro další části této práce shrňme stručně základní HW (technické) požadavky na vytvářený modulární senzorový systém:

- Možnost použít v podstatě libovolný procesor. V uvedeném řešení budou jako příklady využity pouze snadno dostupné a velmi často využívané procesory, avšak princip bude možno aplikovat i na jiné typy.
- Zaměřit se zejména na dostupné vývojové desky (nikoli konkrétní součástky, obvody nebo procesory) ideálně obsahující možnost programového ladění (debuggeru). Tedy ty, která jsou pro experimentální vývoj nejvhodnější.
- Pro příklady využít nejběžněji používané senzory, ale samozřejmě umožnit doplnit i zcela vlastní.
- Využít nejčastěji používané sběrnice, nikoli ty zcela speciální/proprietární. Tedy hlavně ty obsažené ve většině běžně dostupných procesorů a senzorů.
- Možnost použití v podstatě libovolného komunikačního modulu pro odeslání dat.

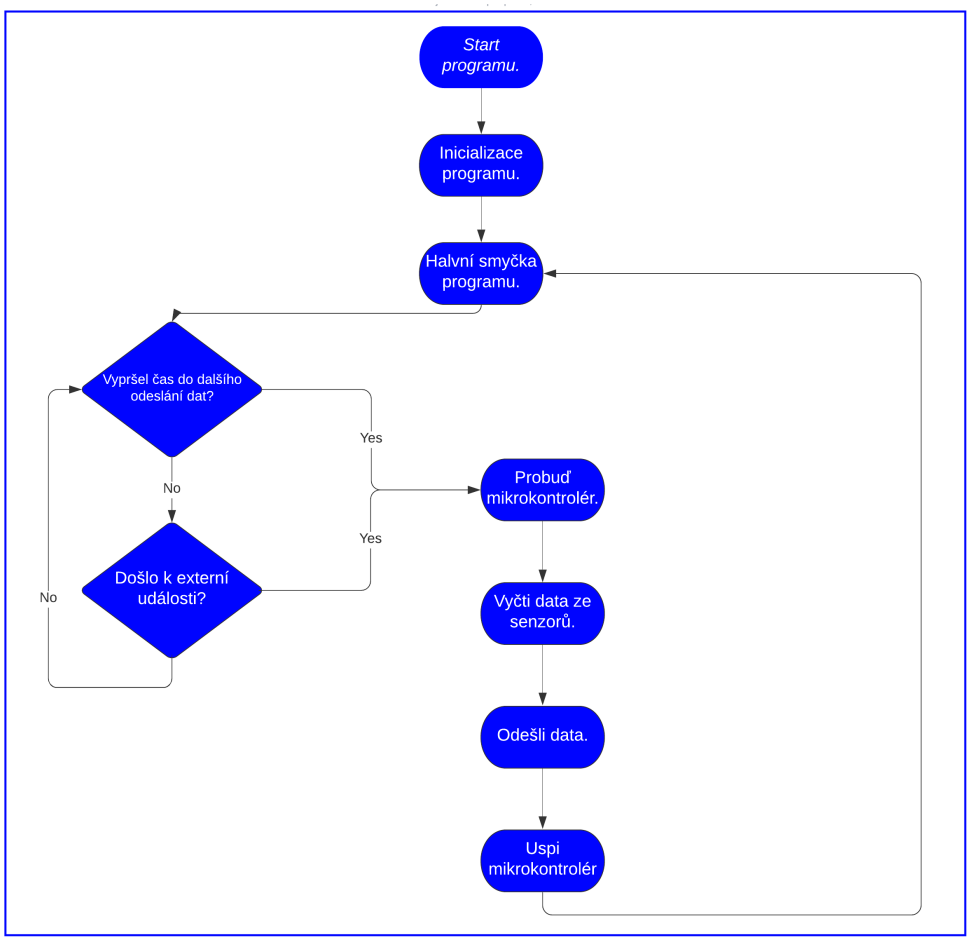
4.2 Funkční požadavky

Pro další části této práce shrňme stručně základní SW (funkční) požadavky na vytvářený modulární senzorový systém:

- Zaměřit se pouze na nejběžnější programovací jazyky jako je C/C++. Další jazyky jako Lua, Python využitelné i v oblasti procesorů nebudou zde uvažovány.
- Použít nejčastěji využívané protokoly pro komunikaci s připojenými senzory a moduly.
- Připojit senzory pomocí konfiguračních souborů, které budou snadno přenositelné na jiný typ procesoru - k dosažení snadného nastavení (inicializaci) senzorů a vyčtení dat.

Před dalším popisem je nutno stanovit co je pod některými pojmy skutečně myšleno:

- Senzor:
  - Součástka nebo malý modul měřící nějakou veličinu (elektrickou i neelektrickou) a na svém výstupu poskytující informaci (analogovou nebo číslicovou) o velikosti této měřené veličiny.
  - Výstupy senzorů mohou být různých typů (logický, analogový, SPI, I2C, USART, ...)
- Procesor:
  - Centrální výkonná jednotka (vykonávající instrukce programu).
  - Vyzvedává data ze senzorů, případně je částečně zpracovává a předávává do požadovaného komunikačního modulu.
- Komunikační modul:
  - Často (ne však vždy) samostatný modul připojitelný k libovolnému procesoru (například pomocí USART, SPI, I2C, atd.).
  - Umožňuje komunikaci s okolím (například s nějakou centrální jednotkou).
- Senzorový modul/zařízení:
  - Zařízení, často již nějak zapouzdřené, obsahující (nejčastěji pouze) jeden procesor, avšak jeden nebo více komunikačním modulů a rovněž jeden, nebo (nejčastěji) více senzorů. Senzory nemusí být nedílnou součástí zařízení, ale mohou být k němu drátově připojeny i z větší vzdálenosti (například venkovní teplota).
- Centrální/řídicí jednotka:
  - Vzdálené zařízení (počítač, tablet, telefon, jiný procesorový modul, ...) přijímající data ze (často i několika) senzorových modulů.
  - Příchozí data jsou zde uložena, zpracována a zobrazena.



Obrázek 4.2: Vývojový diagram běhu programu v procesoru.

Kapitola 5

Výběr vhodných komponent

Tato kapitola představí a podrobněji popíše všechny hardwarové komponenty využité při tvorbě prototypů zařízení. Ty budou dále v popisu rozděleny do skupin. První budou tvořit skutečné senzory – základní stavební kameny pro snímání nejen fyzikálních veličin. Druhou skupinou budou procesory – lokální modul zajišťující sběr dat a sestavení paketu pro odeslání. Poslední skupinou budou v podstatě komunikační moduly – přenosový kanál umožňující předání naměřených dat centrální aplikaci/systému ke zpracování.

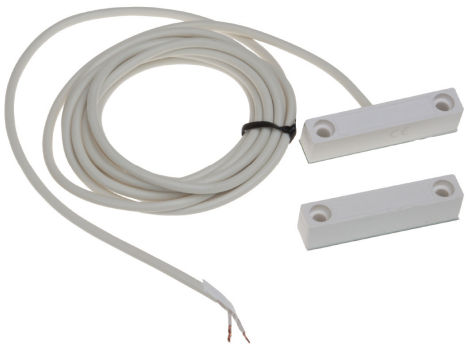
5.1 Použité senzory

Senzory jsou často nejjednodušší komponenty - většinou dosahují malých rozměrů a obsahují jen několik řídících signálů/registrů a někdy i dedikovaný obvod pro vyhodnocení měřené veličiny. Při vytváření projektu využiji zejména senzory, jejichž výstupním signálem je běžná analogová hodnota (veličina), nebo digitální logická úroveň. Dále rovněž senzory s komunikačním protokolem I2C nebo SPI.

V dalším textu jsou uvedeny pouze senzory použité při demonstraci činnosti systému a u každého z nich jeho základní parametry a zejména požadavky na komunikaci, neboli připojení na procesor. Rovněž jsou stručně uvedeny příklady použití těchto senzorů.

Magnetický dveřní kontakt

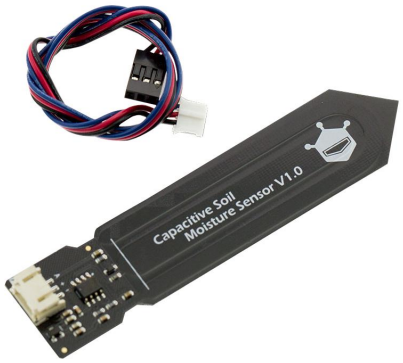
Dveřní kontakt je velmi jednoduché zařízení skládající se ze dvou částí. První část tvoří běžný magnet a druhou část na tento magnet reagující spínací kontakt. Jsou-li obě části v těsné blízkosti, tak je spínač v poloze sepnuto. V okamžiku oddálení (otevření dveří) dojde k rozepnutí spínače (některé typy pracují i opačně). Senzor je vhodné použít k detekci otevřených, nebo zavřených dveří či oken.



Obrázek 5.1: Magnetický dveřní kontakt.[20]

Senzor vlhkosti SEN0193

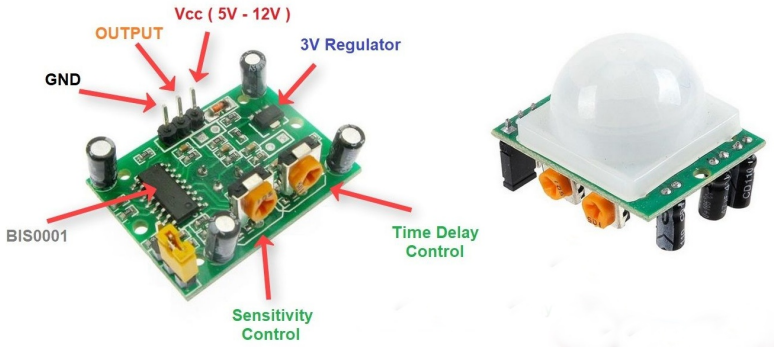
SEN0193 je kapacitní senzor vlhkosti, pracující na principu změny kapacity v závislosti na relativní permitivitě okolí, ve kterém je umístěn ($\epsilon_{vzduchu}$=1, ϵ_{vody}=80). Jeho výstupní veličinou na pinu AOUT je analogové napětí v rozmezí 0V – 3V odpovídající přímo měřené vlhkosti. Další dva piny VCC a GND jsou určeny pro napájení. Tento senzor je vhodný k měření vlhkosti půdy, například v květináčích nebo ve skleníku.



Obrázek 5.2: Kapacitní senzor SEN0193.[21]

Detektor pohybu HC-SR501

Senzor je založen na principu pyroelekrického jevu (PIR – Passive Infra-Red), generuje elektrický proud v závislosti na množství tepelného záření, které vydávají všechny objekty (člověk) v různých intenzitách. Obvod obsažený v senzoru je schopen detekovat změnu v dopadajícím tepelném záření. Projde-li tedy osoba (zvíře, věc) v okolí senzoru, je tato změna (okolní objekty typicky vyzařují nižší tepelné záření, než tělo člověka) v intenzitě tepelného záření vyhodnocena jako pohyb - senzor na pinu OUT vygeneruje výstupní logickou úroveň HIGH. Dále jsou k dispozici dva potenciometry pro nastavení citlivosti a délky výstupního impulzu. Piny VCC a GND jsou určeny pro napájení. Senzory pracující na tomto principu jsou běžně využívány k detekci pohybu, pro spínání kamerových systémů nebo rozsvěcení světel v přítomnosti člověka.



Obrázek 5.3: Detektor pohybu HC-SR501. [22]

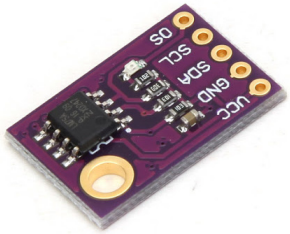
Senzor plynů MQ-5

Senzor pracuje na principu elektrochemické reakce, dokáže detekovat plyny LPG, H2, nebo CO. Jeho citlivé vlákno z materiálu Sn02 při kontaktu s plynem sníží svůj odpor, což vede ke zvýšení napětí na interním odporovém děliči. Velikost tohoto napětí tedy odpovídá koncentraci plynu. Aby tato reakce proběhla, je nutné snímací vlákno zahřát alespoň na teplotu 40°C. Modul má 4 piny, dva z nich jsou napájecí, dále A0 je analogový výstupní pin, jehož napětí odpovídá přímo koncentraci měřeného plynu. Dále pin D0 generující výstupní logickou úroveň HIGH při určité koncentraci plynu. Tato hodnota je snadno nastavitelná pomocí potenciometru umístěného na tištěném spoji senzoru. Je vhodné jej umístit například do místností s přívodem LPG a odhalit tak úniky zemního plynu nebo ke starým ohřívačům vody (tzv. karmám) a detekovat úniky oxidu uhelnatého, vznikajícího při nedokonalém hoření.

Obrázek 5.4: Senzor plynů LPG, H_2 nebo CO.[23]

Senzor teploty LM75A

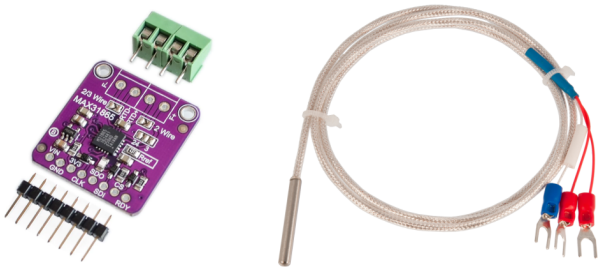
Senzor teploty obsahuje 5 výstupních pinů - 2 jsou napájecí (VCC, GND), 2 piny (SDA, CLK) zajišťují komunikaci s mikrokontrolérem pomocí protokolu I2C. Kromě samotného čtení teploty lze nastavit uvnitř čipu programovatelné registry - teplotní meze, při kterých je na pinu OS generován impulz, logická LOW, nebo HIGH. Senzor je určen k měření teploty v suchém prostředí – lze jej využít k měření pokojové teploty, nebo nejčastěji teploty různých elektrických komponent (procesory, grafické karty, atd) a při překročení povolených teplotních mezí například celý systém vypnout.



Obrázek 5.5: Senzor teploty LM75A.[24]

Převodník Max31865 pro senzor teploty pt100

Převodník teploty Max31865 obsahuje 6 výstupních pinů - 2 jsou napájecí (VCC, GND), 4 piny (CS, MISO, MOSI, CLK) zajišťují komunikaci s mikrokontrolérem pomocí protokolu SPI. Jeho činnost je založena na napěťovém děliči – přesné měření napětí na referenčním rezistoru. Dále nabízí možnost čtyř-svorkového, nebo tří-svorkového připojení senzoru k potlačení vlivu odporu přívodních vodičů. Senzor pt100 se vyznačuje přesným a stálým měřením teploty a jeho nerezový obal umožňuje umístění i do vlhkého prostředí.



Obrázek 5.6: Převodník Max31865 vlevo, senzor pt100 vpravo.[25]

5.2 Použité procesory

Pro demonstraci nemá význam vytvářet programový kód na mnoho typů procesorů. Důležité je vybrat ty nejvíce rozšířené, nebo ideálně ty, které zastupují nějakou (často používanou) ucelenou skupinu a jsou tedy vhodné jako výchozí příklady. Naprostá většina procesorů je programovatelná v jazyce C/C++ a zde vytvořené vzorové programy jsou tedy snadno přenositelné. Realizace navrženého řešení bude demonstrována na dvou příkladech procesorů zastupujících právě vždy jednu cílovou skupinu (zejména pro uživatele).

5.2.1 Procesory ARM od ST Microelectronic

První skupinu představují univerzální procesory typu ARM. Ty jsou stále více zastoupeny ve většině spotřební elektroniky. Pro snadnou činnost bude použit některý vhodný vývojový kit obsahující potřebný programátor pro snadné ladění vytvářeného programu. Jedná se tedy o vzorový příklad univerzálního procesoru typu ARM neobsahující žádný bezdrátový komunikační modul.

Při tvorbě projektu využiji vývojovou desku s názvem NUCLEO-L152RE od společnosti ST Microelectronics obsahující procesor typu ARM s označením STM32L152RE. Řada mikrokontrolérů s označením STM32Lxxx se vyznačuje velice nízkou spotřebou energie (ultra-low-power). Využívají se především v aplikacích s bateriovým napájením. V komerčně dostupných zařízeních se jedná o fitness hodinky, zdravotní přístroje, chytré měřící jednotky atp.

Obrázek 5.7: Komerční využití STM32L. [26]

5.2.2 Procesorový modul ESP8266 (ESP32)

Druhou skupinou jsou komplexnější procesorové moduly. Jejich příkladem může být velmi často využívaný procesorový modul označený ESP8266 (ESP32) obsahující již bezdrátovou komunikaci pomocí WiFi (případně Bluetooth). Ty začaly být v posledních letech velice populární zejména jako komponenty typu Arduino pro jejich velmi nízkou cenu a snadnou dostupnost.

Při tvorbě projektu využiji vývojovou desku/modul s názvem Wemos D1 MINI PRO obsahující procesor ESP8266. Desku lze snadno programovat pomocí vývojového prostředí Arduino IDE programovacím jazykem Arduino (založený na C/C++).

	ESP8266	ESP32
MCU	Xtensa 1-Core 32-bitů	Xtensa 2-Core 32-bitů
Wi-Fi (rychlost up+down)	802.11 b/g/n, HT20 (max 130Mbit)	802.11 b/g/n, HT40 (max 300Mbit)
Bluetooth	Ne	v4.2
Frekvence	80-120 MHz	80-240 MHz
SRAM	160 kBajtů	512 KBajtů
Flash	*1MB - 4MB*	*4MB - 16MB*
GPIO	17	36
SPI / I2C / I2S / UART	2 / 1 / 2 / 2	4 / 2 / 2 / 2
ADC (Analog Digital Converter)	10-bitů	12-bitů

Obrázek 5.8: Základní parametry modulů ESP8266 a ESP32.

5.3 Komunikace

Aby byl senzorový modul skutečně univerzální, měl by být schopen komunikovat pomocí několika různých kanálů, tedy umožňovat přenos dat několika způsoby. Mezi ty nejzákladnější komunikační kanály patří:

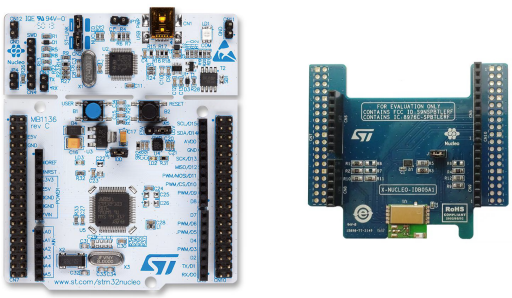
- USART (Universal Synchronous/Asynchronous Receiver and Transmitter):
  - Původní sériový přenos s omezenou rychlostí na kratší vzdálenosti (pomocí dodatečných budičů i sto metrů).
  - Zejména pro spojení více modulů, nebo přenosu dat do speciálních zařízení na kratší vzdálenosti.
- USB (Universal Serial Bus):
  - Moderní sériový a velmi rychlý přenos na krátké vzdálenosti.
  - Nejčastější typ přenosu dat do klasického počítače.
- WiFi:
  - Rychlý bezdrátový přenos pro větší objemy dat možný i na velké vzdálenosti (několik kilometrů).
  - Často převeden na nějaký typ LAN.
- Bluetooth (klasický):
  - Přenos dat celkem pomalou rychlostí na malé vzdálenosti.
  - Převážně pro mobilní zařízení na vzdálenost několika metrů.
- Bluetooth LE (Low Energy):
  - Přenos velmi malého množství dat (bloky zhruba 20 bytes) na velmi krátké vzdálenosti.
  - Využíván zejména pro jednoduchá zařízení při dosažení minimální spotřeby elektrické energie.

WiFi

WiFi je poněkud složitější komunikační kanál, a tudíž se velmi často využívá její integrace přímo v nějakém procesoru neboli modulu. Příkladem může být právě již zmíněný ESP8266 (ESP32). Ten na ploše 2x1 cm^2 obsahuje samotný procesor s dedikovaným jádrem pro WiFi a Bluetooth, 2.4 GHz PCB PCB (Printed Circuit Board) anténu a některé modely i U.FL konektor pro připojení externí antény.

Bluetooth Low Energy (BLE)

Pro komunikaci pomocí Bluetooth LE bude použita rozšiřující deska s označením X-NUCLEO-IDB05A1 BLE (Bluetooth Low Energy) určená pro kit NUCLEO-L152RE. BLE modul je sice umístěn na rozšiřujcí desce, je však možno jej využít jako zcela samostatnou komponentu například pod názvem SPBTLE-RFTR.



Obrázek 5.9: Základní deska NUCLEO-L152RE (vlevo), rozšiřující modul X-NUCLEO-IDB05A1 (vpravo).[27]

Mnoho zařízení používá samostatné BLE moduly připojené přes různé sběrnice (USART, I2C, SPI), proto výběr tohoto modulu je správným krokem. BLE moduly se používají zejména pro přímé spojení dvou zařízení, tedy tzv. peer-to-peer.

Od původního záměru použít v této práci pro demonstraci rovněž USB HID a klasický Bluetooth bylo nakonec (po dohodě s vedoucím práce) ustoupeno z důvodu velmi omezeného přístupu do budovy školy. Realizace HW částí zařízení vyžaduje časté konzultace a i několikahodinové práce na pracovišti s využitím někdy i mnoha měřících/podpůrných přístrojů.

Toto rozhodnutí však není pro výsledek práce nijak omezující, a to z následujících důvodů:

- USB HID je z hlediska přenosu vlastních dat velmi podobný Bluetooth LE. Data se rovněž přenáší pouze po blocích (zde do 64 bytes) akorát větší rychlostí.
- Klasický Bluetooth lze využít v několika režimech komunikace (disponuje mnoha přenosovými protokoly):
  - Je schopen emulovat klasický sériový port nazývaný „Virtual COM Port“. Z hlediska aplikace jej lze tedy využít zcela stejně jako klasický USART. Buď Bluetooth modul přímo převádí USART na Bluetooth komunikaci, nebo jej lze do tohoto typu přenosu přepnout programově.
  - Je schopen přenášet data pomocí v podstatě klasického TCP/IP protokolu, který je velmi podobný UDP.

Nevyužití USB HID a klasického Bluetooth nemá vliv na výsledky této práce. Z dalšího popisu bude zřejmá univerzálnost navrženého řešení, a tudíž pozdější rozšíření například o tyto dva typy komunikace bude v budoucnu velmi snadné. Současně lze konstatovat, že využití připojení formou USB HID (v podstatě pouze pro běžná PC) je spíše pouze pro prvotní testování, nikoli pro častou a dlouhodobou činnost. Rovněž klasický Bluetooth je stále více nahrazován pomocí Bluetooth LE.

Kapitola 6

Realizace návrhu

Tato kapitola bude popisovat konkrétní návrh experimentálního zařízení - pomocí vývojové desky NUCLEO-L152RE (obecný typ ARM) a modulu Wemos D1 MINI PRO (platforma Arduino). Dále bude vysvětlena obecná struktura společná pro obě zařízení pro připojení senzorů, vyčtení jejich dat a následného přenosu.

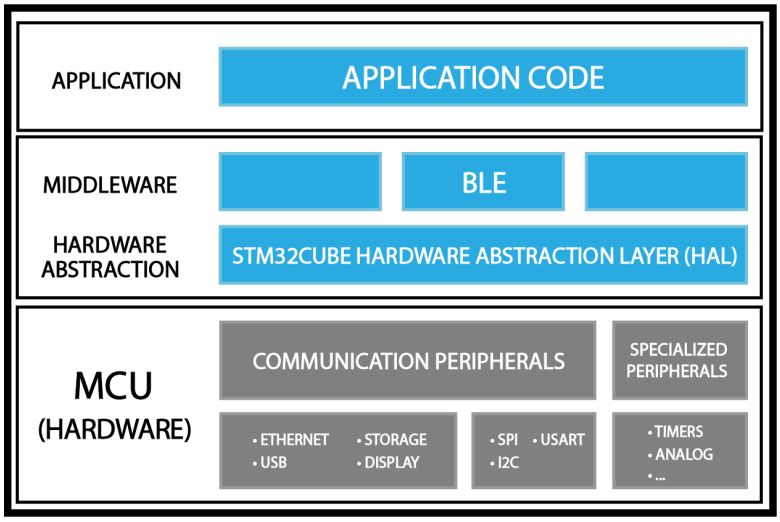
6.1 NUCLEO-L152RE

Vývojové prostředí STM32CubeIDE

K programování procesorů typu ARM, jako jsou STM32Lxxxx, využiji software s názvem STM32CubeIDE od společnosti ST Microelectronics. Jedná se o multifunkční a multiplatformní vývojové prostředí obsahující mnoho funkcí a možností nastavení. Celé toto prostředí je poskytováno zdarma, což je jeho velkou výhodou.

HAL (Hardware Abstraction Layer)

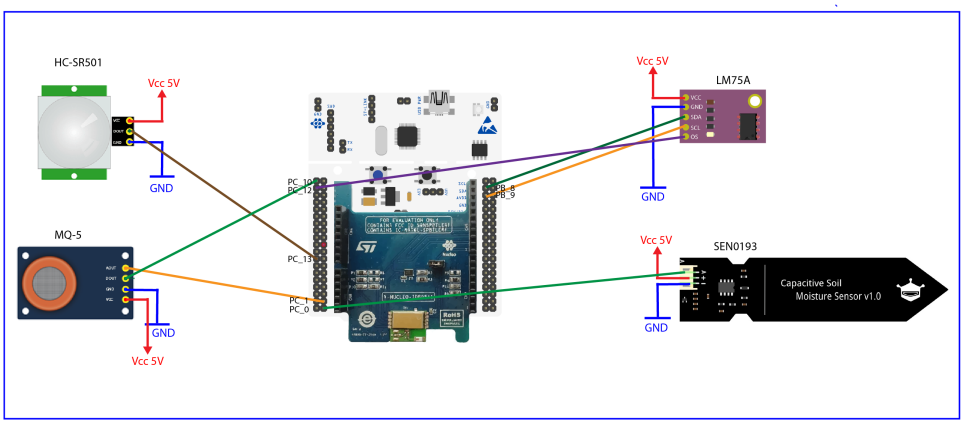
Celý projekt bude vytvořen v jazyce C. Budou využity knihovny HAL (Hardware Abstraction Layer) poskytující obecně použitelnou a jednoduchou sadu API (Applicatoin Programming Interface). Ta interaguje jak s vyšší (aplikační), tak s nižší (hardwarovou) vrstvou, jak je patrné z obrázku 6.1. HAL vhodně skrývá složitost programování celého procesoru a periferií koncovému uživateli. Programový kód vytvořený pomocí knihoven HAL je snadno přenositelný mezi naprostou většinou procesorů typu ARM vyráběných firmou ST Microelectronics.



Obrázek 6.1: HAL jako mezivrstva vyšší a nižší vrstvy aplikace.

Schéma zapojení

V následujícím schématu jsou připojeny senzory ke konkrétním pinům na perifériích mikrokontroléru – v této formě bude demonstrována činnost systému. Jiná konfigurace pinů je samozřejmě snadno nastavitelná pomocí dále popsaných konfigurací v aplikaci.



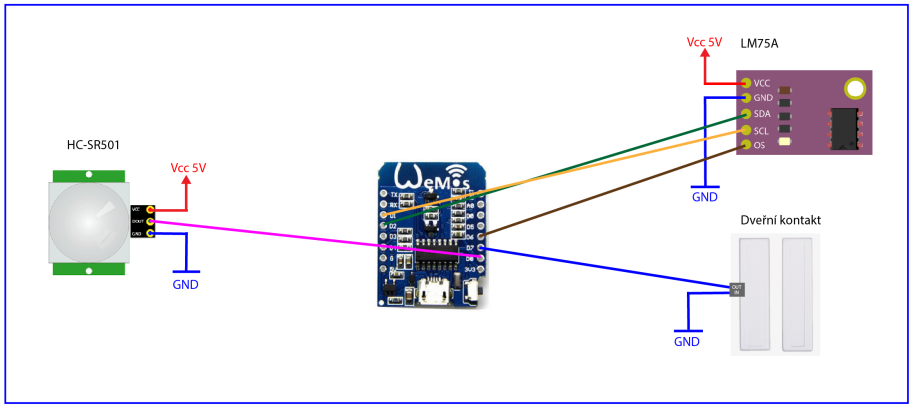
Obrázek 6.2: Schéma připojení vybraných senzorů k STM32 NUCLEO.

6.2 Wemos D1 MINI PRO (ESP8266)

Vývojové prostředí Arduino IDE

Arduino IDE je jednoduché vývojové prostředí podporující programovaní modulu ESP8266 (ESP32) prostřednictvím programovacího jazyka Arduino přímo založeném na C/C++.

Schéma zapojení



Obrázek 6.3: Schéma připojení vybraných senzorů k Wemos D1 MINI PRO.

V mnoha případech lze program vytvořený pro konkrétní desku typu Arduino přenést i na jinou desku typu Arduino, pokud jsou pro cílovou desku dostupné stejné knihovny. Ovšem ne vždy.

6.3 Přenositelnost na jiné procesory/moduly

Vybraný procesor ARM (STM32) je programovaný pomocí jazyka C/C++ s využitím již zmíněných HAL knihoven, modul ESP8266 prostřednictvím jazyka Arduino (přímo založeném na C/C++) a pomocí knihoven projektu Arduino. Syntaxe obou jazyků je sice v podstatě stejná, ale knihovní funkce jsou často velmi odlišné (inicializace periférií, hodin, komunikace, atd.). Konkrétní nízkoúrovňové činnosti musí být tedy uskutečněny na každém procesoru zcela odlišně. Avšak konfigurační struktury pro použité senzory a zařízení implementované (rovněž) v jazyce C mohou být společné pro oba typy zmíněných procesorů, čímž se dosáhne univerzálnosti a i v budoucnu přenositelnosti určité části kódu na zcela jiné procesory. Použitím konfiguračních struktur v programu a jejich správným vyplněním (typ komunikace, návratová hodnota, využití interruptu atp.) jsou data automaticky ze senzorů vyčtena, sestavena do paketu a odeslána. Jedná se tedy o vrstvu aplikace sloužící pro skrytí složitosti použitého procesu.

6.3.1 Připojení a správa senzorů

Zdrojové soubory pro senzory

Téměř každý senzor vyžaduje k činnosti zdrojové soubory (.h, .c). V těchto souborech jsou definované konstanty a implementovány funkce pro inicializaci a čtení dat z konkrétního senzoru. Typově podobné senzory mohou zdrojové soubory i sdílet.

Funkce systému pro senzory

Pro senzory připojené k procesoru pomocí sběrnic (I2C, SPI, ADC, atd.) jsou (na úrovni použitého procesoru) připraveny univerzální funkce pro snadnou inicializaci a vyčtení dat. Kvůli rozdílným knihovním funkcím procesoru ARM a modulu ESP budou těla těchto funkcí samozřejmě implementována pro každý procesor samostatně.

Příkladem prototypu takovýchto platformově nezávislých funkcí jsou:

```
1 CommStatusTypeDef Write_SPI(data_acquisition index_handle,cs_pin
      my_cs_pin,uint8_t *pData, uint16_t Size);
```

- Popis:
  - Funkce Write_SPI zajišťuje zápis (odeslání) uživatelem specifikovaných dat pomocí protokolu SPI do připojeného senzoru s využitím konkrétního pinu CS (Chip Select).
- Vstupní argumenty funkce jsou:
  - `data_acquisition index_handle:`
    - Index zvoleného komunikačního kanálu.
  - `cs_pin my_cs_pin:`
    - Aktivační pin senzoru, v systému definovaný jako konkrétní pin procesoru vlastním datovým typem cs_pin (CS_1, CS_2, atd.).
  - `uint8_t *pData:`
    - Data pro zápis (odeslání) do připojeného senzoru.
  - `uint16_t Size:`
    - Velikost dat v bajtech pro zápis do připojeného senzoru.

- Návratová hodnota:
  - Funkce má definovaný vlastní návratový typ CommStatusTypeDef (v současné době) nabývající 5 hodnot:
    - COMM_OK - Odeslání dat proběhlo v pořádku.
    - COMM_TIMEOUT_ERROR – Zařízení nereagovalo do určitého timeoutu.
    - COMM_BUSY_ERROR – Zařízení je zaneprázdněné.
    - COMM_TOO_FEW_DATA – Příliš málo dat pro odeslání.
    - COMM_UNSPECIFIED_ERROR – Nespecifikovaný error.

Následující 2 prototypy funkcí obsahují stejné návratové hodnoty a nebudou tedy znovu uváděny.

```
CommStatusTypeDef Write_I2c(data_acquisition index_handle,
    uint16_t DevAddress, uint8_t *pData, uint16_t Size);
```

- Popis:
  - Funkce Write_I2C zajišťuje zápis (odeslání) uživatelem specifikovaných dat pomocí protokolu I2C do připojeného senzoru na konkrétní adrese.
- Vstupní argumenty funkce jsou:
  - `data_acquisition index_handle:`
    - Index zvoleného komunikačního kanálu.
  - `uint16_t DevAddress:`
    - I2C adresa připojeného senzoru.
  - `uint8_t *pData:`
    - Data pro zápis (odeslání) do připojeného senzoru.
  - `uint16_t Size:`
    - Velikost dat v bajtech pro zápis do připojeného senzoru.

```
CommStatusTypeDef ReadADC(data_acquisition index_handle,
    uint32_t channel, uint16_t * adc_val);
```

- Popis:
  - Funkce ReadADC zajišťuje čtení analogové veličiny na zadaném pinu.

- Vstupní argumenty funkce jsou:
  - `data_acquisition index_handle:`
    - Index zvoleného komunikačního kanálu.
  - `uint32_t channel:`
    - Kanál (pin) pro čtení analogové veličiny.
  - `uint16_t * adc_val:`
    - Adresa proměnné pro uložení vyčtené analogové hodnoty.

Konfigurace (jednoho) senzoru

Nastavení jednoho konkrétního senzoru pomocí vhodné konfigurační struktury (definované v souboru sensors.h) zajistí snadnou inicializaci senzoru, vyčtení dat a vložení jeho hodnoty do datového paketu. Konfigurační struktura senzoru musí obsahovat následující položky:

- Typ komunikace:
  - Přenosový kanál pro komunikaci se senzorem a jeho index (procesor může obsahovat několik těchto kanálů).
    - I2C1, I2C2, ...
    - SPI1, SPI2, ...
    - ADC1, ADC2, ...
    - Interrupt.
- Návratová hodnota:
  - Jsou definovány vlastní datové typy pro přenos hodnoty ze senzoru do aplikace běžící na procesoru z důvodu odlišné interpretace dat na různých typech procesorů.
  - Nové datové typy musí být společné jak pro program v zařízení, tak i pro program obsluhy senzorů.
  - Příklady několika takovýchto typů:
    - BOOL_VAL - Hodnota typu bool (true/false).
    - UINT8 - Celočíselná kladná hodnota v jednom byte.
    - UINT16 - Celočíselná kladná hodnota v 2 bytes.
    - INT16p2 - Reálná hodnota v rozmezí –327.67 až +327.67
    - BYTESX - Variabilní počet bytes, kde první byte udává jejich počet.
    - ...

- Asociace interruptu:
  - Konkrétní pin procesoru v systému definovaný vlastním datovým typem interr_pin (INT_1, INT_2, atd.).
- Stav interruptu:
  - Zda došlo na asociovaném pinu k interruptu (defaultní hodnota je false).
- Inicializace senzoru:
  - Odkaz na funkci pro inicializaci daného senzoru:
    - Například nastavení řídicích registrů a dalších položek přímo uvnitř senzoru při jeho inicializaci (pomocí funkcí systému pro senzory, jako je například Write_I2C).
- Vyčtení dat ze senzoru:
  - Odkaz na funkci, pro vyčtení dat ze senzoru (využívají se stejné funkce systému, jako při inicializaci).

Programový kód výše uvedené konfigurační struktury (ve stejném pořadí) definované v souboru sensors.h:

```
typedef struct Sensors
{
// index kanálu, který je využitý pro získání dat ze senzoru
data_acquisition data_acq;
// návratová hodnota dat
data_types ret_type;
// který pin je využit jako interrupt
interr_pin interrupt_line_pin;
// proměnná, která udržuje stav, zda došlo k interruptu na
    daném pinu
// interrupt_line
bool triggered_interrupt;
// odkaz na funkci, která inicializuje daný senzor
void (*InitSensor)(void);
// odkaz na funkci, která čte data ze senzoru
void* (*ReadData)(void);
//pokud funkce senzoru nesplňují požadovaný formát, lze je
    obalit do jiné funkce, která daný formát splňuje
} Sensor;
```

Ve skutečném programu je tato struktura použita/vyplněna například následovně:

```
Sensor sensors[]=
{
  // senzor pohybu HC-SR501
  {EVENT,BOOL_VAL_INTERR,INT_1,false,NULL,NULL},
// senzor teploty LM75A
  {I2C_1,INT16p2, INT_2,false,&read_temperatureLM75A_w,&
    initLM75A_w},
```

```
// senzor plynu MQ5
  {ADC_1,PERCENTAGE, INT_3,false,&read_data_MQ5_wrapper,NULL}
// senzor vlhkosti sen193
//{ADC_1,PERCENTAGE,0,false,&read_data_SEN0193_wrapper,NULL}
};
```

Výhodou inicializace pomocí tohoto zápisu je přehlednost a snadné zakomentování/blokování nevyužitých senzorů. Rovněž posloupnost senzorů v této konfiguraci určuje přímo pozici dat ve vysílaném paketu a lze ji tedy přesunem řádků této konfigurace velmi snadno upravovat podle aktuální potřeby.

První vzorový inicializovaný senzor je detektor pohybu HC-SR501. Jedná se o příklad velice jednoduchého senzoru, jehož výstupní veličinou je pouze interrupt (hodnota true/false) na konkrétním pinu.

- EVENT:
  - Příjem dat pomocí interruptu.
- BOOL_VAL_INTERR:
  - Návratová hodnota je typu bool (byl zaznamenán pohyb – true, nebyl zaznamenán pohyb - false).
- INT_1:
  - Výstupní pin senzoru je připojen na pin sběrnice INT_1 (v systému definovaný na pinu 13).
- false:
  - Výchozí hodnota senzoru je false – nebyl zaznamenán pohyb.

Tento senzor neobsahuje žádné funkce pro inicializaci ani pro vyčtení hodnot, proto poslední dvě položky obsahují hodnotu NULL.

Druhým příkladem inicializace je senzor teploty LM75A. Jedná se o příklad sofistikovanějšího senzoru vyžadující vyplnění všech údajů struktury.

- I2C_1:
  - Příjem dat pomocí sběrnice I2C (první poskytované v zařízení).
- INT16p2:
  - Návratová hodnota je typu float (reálné číslo):
    - Reálná hodnota v rozmezí –327.67 až +327.67.
    - (znaménková Int16) / 100 (dvě desetinná místa).
- INT_2:
  - Výstupní pin senzoru je připojen na pin sběrnice INT_2.

- false:
  - Výchozí hodnota senzoru je false.
- &initLM75A_w:
  - Odkaz na funkci pro inicializaci daného senzoru.
    - Tato konkrétní funkce nastavuje rozlišení daného měření a mezní hodnoty teploty, při kterých dojde na pinu INT_2 k interruptu.
- &read_temperatureLM75A_w:
  - Odkaz na funkci pro vyčtení dat ze senzoru.

Obě funkce pro inicializaci a čtení dat ze senzoru jsou vždy implementovány ve zdrojových souborech příslušného senzoru pomocí volání systémových funkcí určených pro senzory. Příklad skutečné implementace funkce read_temperatureLM75A() ve zdrojovém souboru LM75A.c je následující:

```
float ReadTempLM75A(void)
{
// buffer pro data k odeslání
uint8_t sendData[1];
// přiřazení adresy registru, ze kterého hodláme číst
sendData[0] = TEMP_ADDRESS;
// buffer pro příjem dat
uint8_t recData[2];
// inicializace proměnné pro výstupní data
float temp;
// fuknce systému pro senzory, která vyčte data ze senzoru
// index_handle - jaké I2C se využívá ke komunikaci
// LM75A_ADDRESS - adresa senzoru
// sendData - buffer s daty k odeslání
// recData - buffer pro příchozí data
// 1 - počet odeslaných bytes
// 2 - počet příchozích bytes
Read_I2c(index_handle,LM75A_ADDRESS, sendData,recData,1,2);
// výpočet teploty podle informací uvedených v datasheetu pro
    LM75A
int16_t ret = (recData[0] << 8) | recData[1];
ret >>= 7;
temp = ret * 0.5;
// výpočet teploty podle informací uvedených v datasheetu pro
    LM75A
// návratová hodnota funkce je teplota typu float
return temp;
}
```

Hlavičkový soubor LM75A.h obsahuje tyto důležité položky:

- `#include "sensors.h"`
  - Obsahuje datové typy a definice konfigurační struktury senzoru.
- `#include "I2Cconnect.h"`
  - Obsahuje systémové fuknce pro vyčtení a odeslání dat ze senzoru připojeného pomocí I2C sběrnice.
- `#define LM75A_ADDRESS 0x90`
  - I2C adresa senzoru.
- `#define TEMP_ADDRESS 0x00`
  - Adresa registru pro vyčtení teploty.
- `float ReadTempLM75A(void);`
  - Definice funkce pro vyčtení teploty.
- `void InitLM75A(Sensor  sen);`
  - Definice funkce pro inicializaci senzoru (vstupním argumentem je daný senzor).

6.3.2 Senzorový modul/zařízení

Zařízení (ESP8266, NUCLEO-STM, atd.) může obsahovat několik senzorů a rovněž může data zasílat prostřednictvím jednoho nebo více typů komunikací (i současně). Pro zajištění univerzálnosti a správné činnosti musí být splněny některé náležitosti.

Konfigurace zařízení

Nastavení senzorového zařízení pomocí konfigurační struktury zajistí například snadné sestavení datového paketu a jeho odeslání. Zde pod pojmem zařízení rozumíme senzorový modul obsahující jeden procesor a libovolný počet připojených senzorů. Konfigurační struktura zařízení musí obsahovat následující položky:

- Počet připojených senzorů.
  - Tato položka je automaticky vyplněna pomocí struktury konfigurace senzorů.
- Sériové číslo zařízení:
  - Každé zařízení má uživatelem definované sériové číslo (16bitová hodnota) k snadnému odlišení příchozích dat v centrální aplikaci.
- Formát odeslaných dat:
  - Binární formát dat:
    - Odeslání dat v binárním formátu je nejúspornější metoda z pohledu množství odeslaných dat. Tento formát je vhodný pro komunikace s omezeným přenosem nebo rychlostí (například BLE).
  - AsciiHex formát dat:
    - Jsou-li data odeslána v AsciiHex formátu, je každý přenášený byte zakódován pomocí dvou ASCII znaků, tedy v podstatě do dvou bytes obsahující čitelné znaky. Výhoda tohoto kódování je snadná čitelnost dat lidským okem, je tedy vhodný pro experimentování a ladění programu. Nevýhoda tohoto přístupu je nutnost odeslat dvakrát více informace, než je potřeba. Tento formát je vhodný pro rychlé komunikace, jako například USART nebo UDP.
- Sleep mode:
  - Zda daný mikrokontrolér po odeslání dat přejde do sleep módu.
- Perioda vyčtení dat ze senzorů:
  - Jak často dochází k vyčtení dat ze senzorů a jejich odeslání.
- Odeslání dat:
  - Odkaz na funkci zajišťující sestavení a odeslaní (již vytvořeného) datového paketu.

Programový kód výše uvedené struktury (ve stejném pořadí) definované v souboru device.h:

```
typedef struct Devices
{
// počet připojených senzorů k zařízení
uint8_t connected_sensors;
// sérivé číslo zařízení, 16-bitová hodnota (první byte je vyšší
    , druhý nižší)
uint8_t serial_num[2];
// Ascii nebo binární formát dat
transmit_format trans;
```

```
9  // TRUE - zařízení přejde do sleep módu, FALSE - zařízení nepř
       ejde do sleep módu
10 bool sleep_mode;
11 // časový interval, po kterém jsou data ze senzorů vyčtena a
       odeslána, maximální interval je omezen 16-bitovou hodnoutou
       0xFFFF
12 uint16_t raport_period;
13 // odkaz na funkci, která odešle data
14 void (*SendData)(void);
15 //pokud funkce k odeslání dat nesplňuje požadovaný formát, lze
       ji obalit do jiné funkce, která daný formát splňuje
16 } Device;
```

Ve vzorovém programu je tato struktura vyplněna následovně:

```
1 Device mySTdevice = {sizeof(sensors)/sizeof(Sensor),{0x12,0x34},
      BINARY,false,60*60,&BLESendData};
```

- `sizeof(sensors)/sizeof(Sensor)`
  - Počet připojených senzorů.
- `{0x12/* first high */,0x34/* second low */}`
  - Sériové číslo zařízení.
- `BINARY`
  - Odeslání dat v binárním tvaru.
- `false`
  - Mikrokontrolér nepřejde do sleep módu.
- `60*60`
  - Data ze senzorů budou vyčtena a odeslána každých 60 minut.
    - V případě, že nějaký senzor generuje interrupt jsou data odeslána okamžitě, tedy ještě před uplynutím definované doby.
- `&BLESendData`
  - Odeslat data pomocí Bluetooth LE.

Funkce systému pro zařízení

- Začátek datového paketu:
  - Funkce vytvoří prázdný vysílaný datový paket a vyplní jeho úvodní část:
    - Např.: BLE vyžaduje před samotnými daty blok specifických dat (Manufacturer Specific Block).
- Naplnění datového paketu:
  - Funkce zajišťující automatické vyplnění paketu podle konfigurace senzorů a zařízení.

Formát paketu (s příslušnou velikostí v bajtech/bytes)

Paket (tedy jeho obsah) je vysílán v podstatě jako posloupnost bytes (nezávisle na skutečném přenosovém formátu). Tato posloupnost musí být pevná, a to následující:

- Délka dat (1B):
  - Počet následujících bytes.
- Typ paketu (1B):
  - Datový paket:
    - Obsahem paketu jsou data ze senzorů.
  - Textový paket:
    - Obsahem paketu jsou jména senzorů a k nim příslušné jednotky.
  - Stavový/chybový paket (celkového) zařízení:
    - Pokud zařízení není z nějakého důvodu schopno činnosti, tak vysílá paket informující o tomto stavu.
- Sériové číslo zařízení (2B):
  - První byte je vyšší a druhý nižší.
- Datový typ (1B).
- Příslušná data (1 - xB).
- . . . :
  - Zde se opakuje datový typ + příslušná data podle počtu připojených senzorů.
- Kontrolní součet.

Takto vytvořený paket může být podle aktuálně použitého přenosového kanálu (BLE, USART, UDP, ...) ještě obalen dalšími informacemi/bytes. Pro názornost je dále uvedeno, jaký je skutečný tvar odesílaného paketu pro některé druhy komunikace.

BLE Paket (přenášený binárním způsobem)

- Obsahuje tzv. „Adverisement část vysílání“ (před samotnými daty ze senzorů).
- Jeho velikost může být (v běžném případě) pouze 25 bytes.
- Jeho celkový formát je:
  - 0xFF 0xFF 0xFF ‘H’ ‘C’
    - 0xFF - „Manufacturer Specific Block“.
    - 0xFF 0xFF - Nespecifikované ID výrobce BLE zařízení.
    - ‘H’ ‘C’ - Identifikace zařízení pro náš projekt.
  - Data ze senzorů (dříve popsaný sestavený paket pro odeslání).

USART/UDP Paket (přenášený AsciiHex způsobem)

- Obsahuje kontrolní znaky (zde hranaté závorky) z důvodu snadné detekce začátku a konce paketu.
- Jeho celkový formát je:
  - ‘[’ - Otevírací hranatá závorka značí začátek paketu.
  - Data ze senzorů v AsciiHex formátu.
  - ‘]’ - Uzavírací hranatá závorka značí konec paketu.

Skutečné odeslání dat

Odeslání dat se skládá ze dvou částí, a to:

- Nezávislé na systému/procesoru:
  - Sestavení paketu podle konfigurace senzorů (přepokládáme-li samozřejmě použití C/C++). Tedy v podstatě vytvoření pole bytes.
- Závislé na systému:
  - Skutečné odeslání paketu jako dříve vytvořené posloupnosti bytes pomocí konkrétního komunikačního protokolu.

Funkce pro odeslání dat nezávislé na systému/procesoru se nacházejí v souboru „device.c“. Příklad funkce vytvářející datový paket podle předané konfigurace pro zařízení a vkládající data ze senzorů je následující:

```
int CreateDataPacket(Device * device, Sensor * sensors, uint8_t
    * data)
{
// Vytvoření paketu jako pole binárních hodnot
// Device device - struktura zařízení
// Sensor * sensors - struktura nezorů
// uint8_t * data - datový buffer
// začátek na pozici 1, 0-tá pozice je určena pro délku dat,
    kterou vypočteme na konci funkce
int pos = 1;
// o jaký typ paketu se jedná
data[pos++] = DATA;
//sériové číslo zařízení
data[pos++] = device.serial_num[0];
//sériové číslo zařízení
data[pos++] = device.serial_num[1];
// vyčti data ze všech senzorů
for (int i = 0; i < device.connected_sensors; i++ )
{
// pomocná fuknce, která vyplní datový buffer typem dat a přísl
    šnými daty z konkrétního senzoru
pos = FillPacketDataSensors(sensors[i], data, pos);
// nastavit defaultní hodnoty interruptu
sensors[i].triggered_interrupt=false;
// pokud je pozice rovna 0 došlo k chybě
if(pos == 0)
  return 0;
}
// délka dat zapsaná do první pozice v bufferu
data[0] = pos;
// kontrolní součet
data[pos++] = CreateCRC(data, pos);
// návratová hodnota je délka celého paketu
return pos;

}
```

Pomocná funkce FillPacketDataSensors pro vložení dat z jednoho seznoru do vytvářeného paketu má následující implementaci (uvedena pouze část funkce):

```
int FillPacketDataSensors(Sensor sensor, uint8_t *data, uint8_t
    pos)
{
// Naplnění paketu daty z jednoho senzoru
// před samotná data uvést datový typ
data[pos++] = sensor.ret_type;
// každý datový typ je zpracován individuálně
    switch(sensor.ret_type)
    {
       case INT8: ;
        // inicializace pointru na int8_t
        int8_t *ptr2 = NULL;
```

```
12          // přiřazení adresy na vyčtená data
13          ptr2 = (int8_t*)sensor.ReadData();
14          // uložení dat do pozice v paketu
15          // a následná iterace pozice
16          data[pos++] = *ptr2;
17          break;
18
19        case UINT16:;
20          // inicializace pomocné proměnné
21          uint16_t data3 = 0;
22          // inicializace pointru na int16_t
23          uint16_t *ptr3 = NULL;
24          // přiřazení adresy na vyčtená data
25          ptr3 = (uint16_t*)sensor.ReadData();
26          // uložení dat do proměnné
27          data3 = *ptr3;
28          // rozdělení dat do 2 bytes
29          data[pos++] = (uint8_t)((data3 >> 8) & 0xFF);
30          data[pos++] = (uint8_t)(data3 & 0xFF);
31            break;
32        default:
33            // neexistující datový typ
34            return 0;
35      }
36    return pos;
37  }
```

Funkce pro skutečné odeslání paketu podle typu použitého procesoru/zařízení jsou uloženy v souborech s názvy „SendPaketSTM32.c“ nebo „SendPaketESP8266.cpp“ atd., tedy podle konkrétního zařízení.

Příklad implementace konkrétní funkce odesílající data pomocí protokolu UDP vytvořené přímo pro modul ESP8266 využívající výše uvedenou funkci „CreateDataPacket“:

```
1  void UDPSendData()
2  {
3  // vytvoření datového paketu
4  CreateDataPacket(device, sensors, text_to_send_glob);
5  // device - struktura zařízení
6  // sensors - struktura senzorů
7  // text_to_send_glob globální pole typu uint8_t určené pro vytvo
       ření paketu (uložení dat do pole podle patřičného pořadí)
8  // adresát paketu
9  int done1 = Udp.beginPacket(ip, MyComm.port_remote);
10 // připravit data k odeslání
11 Udp.write(text_to_send_glob.c_str());
12 delay(2);
13 // odeslat data
14 int done2 = Udp.endPacket();
15 // kontrola odeslaných dat v serial monitoru
16 Serial.println("UDP Send ... ( Status create: " + String(done1)
       + " / Status send: " + String(done2) + " )");
17
18 }
```

Nabízí-li zařízení více způsobů komunikace, může si uživatel snadno navolit požadovaný typ přidáním odkazu na funkci pro odeslání dat v konfigurační struktuře zařízení.

6.4 Informace o senzorech

V naprosté většině případů stačí ze zařízení vysílat pouze hodnoty jednotlivých senzorů (jejich data). Centrální jednotka (již většinou) obsahuje informace, jaké hodnoty jsou z jakého senzoru (teplota, dveře, ...) přenášeny. Pro testovací účely může být často velmi vhodné, aby zařízení (senzorový modul) nějaké informace o svých senzorech v případě potřeby poskytlo i samo na požádání. Zde se nemusí jednat o zcela stejné (komplexní) informace, jaké jsou uloženy v centrální jednotce (venkovní teplota na severní straně), ale může jít pouze o základní informace pro servisního technika (venkovní teplota).

Jelikož pakety vysílané ze zařízení (senzorového modulu) nejsou potvrzovány, lze je tedy velmi snadno zachytit libovolným (dalším) testovacím zařízením (tablet/telefon/...) a tím je v podstatě číst/zobrazit například pro kontrolu. Půjde však pouze o číselné hodnoty bez jakéhokoli popisu. Právě z tohoto důvodu je vhodné do zařízení vložit nějaké základní informace o připojených senzorech a ty na požádání (v nejjednodušším případě pouze na stisk tlačítka) vyslat do (přijímacího) testovacího zařízení pro lepší porozumění, jaká data jsou vlastně přenášena. Z tohoto důvodu je potřeba do zařízení vložit jednoduchý textový paket obsahující základní informace o připojených senzorech.

Tvar textového paketu

- Délka dat (1B):
  - Počet následujících bytes.
- Typ paketu (1B):
  - Textový.
- Sérivé číslo zařízení (2B):
  - První byte je vyšší a druhý nižší.
- Jméno senzoru (1 - xB):
  - Textový řetězec variabilní délky zakončený 0x01.

- Jednotky daného senzoru (1B):
  - Předem nachystané konstanty reprezentující dané jednotky zakončené 0x00:
    - 0x00 – žádné jednotky.
    - 0x01 – °C.
    - 0x02 – %.
    - ....
- ...:
  - Zde se opakuje jméno senzoru + jednotky daného senzoru podle počtu připojených senzorů.
- Kontrolní součet.

Část textového paketu vyplněného daty (jméno senzoru + jednotky daného senzoru) je následující (uvážíme-li, že zařízení obsahuje 3 senzory):

- „teplota“ 0x01 0x01 0x00 „plyn CO2“ 0x01 0x02 0x00 „pohyb“ 0x01 0x00 0x00

Mezi názvem pro senznor a jeho jednotkami je oddělovací hodnota 0x01 a bloky informací pro jednotlivé senzory jsou odděleny/ukončeny hodnotou 0x00. Uvozovky u názvu senzorů jsou uvedeny pouze pro přehlednost.

6.5 Spolehlivost dat a jejich přenos

Nedílnou součástí tvorby systému je zajištění jeho správné činnosti. Tato práce se nezabývá přímo spolehlivostí, nebo dokonce zabezpečením přenosu dat jelikož jde velmi často o komplexní a složitý proces. Přesto však lze vytvořit jednoduché mechanismy pro detekci a signalizaci nežádoucích stavů v systému. Ty jsou zde rozděleny na dvě části:

1. Signalizace nesprávné činnosti vlastního senzorového modulu. Senzorový modul sám detekuje chybu (nějakou nesprávnou činnost) a je schopen o tomto stavu informovat.
2. Signalizace nesprávné činnosti (celého) systému. Centrální jednotka detekuje nesprávné chování senzorového modulu nebo celého systému a je potřeba tento stav nahlásit.

6.5.1 Signalizace senzorového modulu

Senzorový modul je v mnoha případech schopen detekovat vlastní částečné selhání, tedy přesněji řečeno selhání nějaké své části. Zde jsou uvedeny pouze ty základní:

- Selhání senzoru:
  - Při komunikaci se senzorem (inicializace/čtení dat) může nastat chyba (Timeout, nesprávná hodnota, ...). Tuto skutečnost lze velmi snadno v datovém paketu signalizovat například přidáním dalšího typu hodnoty „Err“. V paketu na místě skutečného typu dat pro chybný senzor bude hodnota „Err“ a za ní následuje jednobytový kód chyby tohoto senzoru (nikoli skutečná hodnota ze senzoru).
- Chybná data senzoru:
  - V mnoha případech je sám senzorový modul schopen detekovat, zda senzor poskytuje správná data, například v rozsahu zadaných mezí. Pokud data čtená ze senzoru tyto meze překročí, lze v datovém paketu místo hodnoty ze senzoru využít již dříve zmíněnou položku „Err“.
- Selhání zařízení:
  - Při startu zařízení inicializuje potřebné periférie a i zde může dojít k selhání. Tento stav lze hlásit pomocí stavového paketu a tím se dožadovat pozornosti ze strany centrální jednotky nebo servisní obsluhy. V tomto typu paketu je rovněž uveden kód detekované chyby.
- Omezení činnosti zařízení:
  - V případě skutečně hodně vybitého akumulátoru (nízký stav baterie byl zařízením dlouhodobě hlášen) již není zařízení schopno své plné činnosti. Zařízení tedy vypne všechny životně nepotřebné části (zejména senzory) a vysílá stavový paket o své nouzové činnosti („existuji, ale nejsem schopno plné činnosti“).

První dva body souvisí s problémem konkrétního senzoru. Data jsou odeslána ve standardním datovém paketu, ale místo hodnoty senzoru je uložen chybový kód detekovaného selhání:

- „Typ dat“ je nahrazen chybovým typem „Err“.
- „Data ze senzoru“ jsou nahrazeny konkrétní chybou.
  - Err_comm:
    - Při komunikaci se senzorem došlo k chybě.
  - Err_init:
    - Při inicializaci senzoru došlo k chybě.
  - Err_timeout:
    - Senzor nereagoval do nastaveného timeoutu.

- Err_out_of_bounds:
  - Hodnota ze senzoru je mimo stanovený rozsah.
- ....

Poslední dva body (chyby v zařízení) souvisí s problémem celkového zařízení. Data jsou odeslána ve stavovém paketu a příčinou může být jedna z těchto chyb:

- Err_peripheral_failure:
  - Došlo k problému s periférií:
    - Konfigurace pinu.
    - Nastavení AD převodníku.
    - ....
- Err_main_power_failure:
  - Hlavní síťový zdroj energie byl odpojen.
- Err_battery_power_failure:
  - Napětí baterií kleslo pod kritickou hranici.
- Err_clock_failure:
  - Došlo k problému s nastavením hodin procesoru.
    - Například po probuzení ze sleep módu, kdy je potřeba opět nastavit hlavní hodiny systému.
- ....

6.5.2 Signalizace systému

Pokud je selhání senzorového modulu „fatální“ (zcela došel akumulátor, velké mechanické poškození, ...), tak není schopen sám o sobě toto selhání dostatečně nahlásit. V tomto případě je nutno detekci vytvořit na vzdálené přijímací straně, tedy v centrální jednotce. Toto lze uskutečnit na několika úrovních:

- Odmlčení senzorového zařízení:
  - Postačuje v podstatě pouze detekovat interval mezi příchozími daty ze senzorového zařízení, a pokud je tento interval delší, než zadaná povolená mez, tak tento chybový stav signalizovat. Z tohoto důvodu je vhodné do popis/konfigurace senzorového zařízení (v centrální jednotce) přidat položku TimeOut pro vysílací zařízení.

- Chybná data:
  - U složitých senzorů již nemusí být snadné detekovat chybná data ze senzoru přímo v senzorovém modulu (například koncentrace různých plynů, představující jejich hraniční hodnoty). To může být způsobeno například nestabilitou napětí pro některý senzor, nebo jeho postupně se projevující/začínající poruchou.
- Chybné chování (nejen senzorového modulu):
  - Tento typ detekce nesprávného stavu již zcela jistě přesahuje rozsah této práce, ale je potřeba jej alespoň zmínit. Někdy nelze chybu konkrétního senzoru odhalit jinak, než porovnáním jeho výstupních hodnot s hodnotami jiných senzorů.
  - Příklad chybného chování může být následující:
    - Venkovní senzor teploty hlásí -20°C, teplota v místnosti hlásí +20°C a spínací kontakt hlásí již několik hodin otevřené okno. Tato kombinace údajů je vysoce nepravděpodobná a tedy minimálně jeden ze senzorů musí uvádět chybnou hodnotu, nebo je špatně umístěn.

Tedy i pomocí těchto velmi jednoduchých opatření rozdělených mezi senzorový modul a centrální jednotku lze ve velké míře zajistit spolehlivost celého systému.

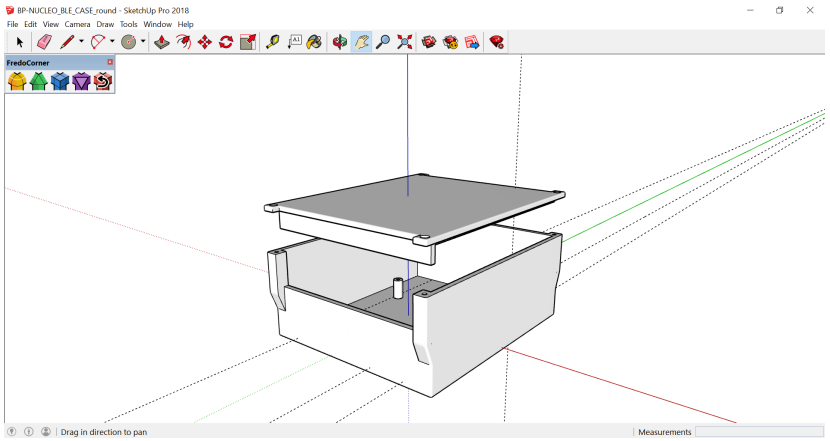
Kapitola 7

Návrh a realizace zapouzdření

Tato část práce bude věnována návrhu a realizaci zapouzdření jak pro celkové zařízení (senzorový modul), tak i případně pro jednotlivé senzory (nejsou-li přímou součástí jednoho zařízení) s nekrytými kontakty a konektory na plošném spoji.

7.1 Modelovací program SketchUp

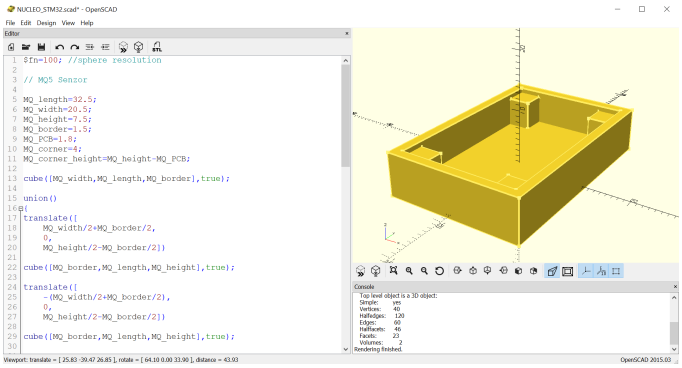
K tvorbě ochranného obalu (pouzdra) pro NUCLEO s BLE shieldem využiji 3D modelovací program SketchUp od společnosti Trimble INC. Jedná se o populární 3D software určený k tvorbě prototypů jak jednoduchých, tak i složitých struktur. Objekty jsou vytvářeny v interaktivním prostředí - tahem kurzoru. Následné úpravy do požadovaných rozměrů lze realizovat pomocí nástrojů v liště aplikace (změna velikosti, posun, rotace, ...). Program je dostupný ve třech verzích, k tvorbě modelu bude využita trial verze (zkušební doba na 30 dní s omezenými funkcemi programu) SketchUp Pro. Dále bude využit plug-in FredoCorner (na vyhlazení hran) dostupný zdarma pomocí SketchUp Extension Warehouse.



Obrázek 7.1: Spodní a vrchní část krytu na NUCLEO a BLE shild.

Modelovací program OpenSCAD

K tvorbě 3D objektů pouzder pro senzory využiji open-source software s názvem OpenSCAD. K modelování je využit skriptovací jazyk speciálně navržený pro tento program – model je vytvořen na základě kompilovaných příkazů. K dosažení komplexních modelů je zapotřebí prolnout nebo sloučit základní primitivní objekty jako jsou válce, koule, kvádry atd.

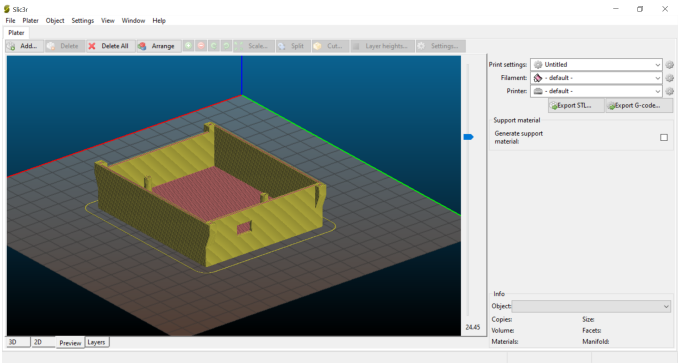


Obrázek 7.2: Modelování krytu na senzor plynu MQ5.

Po dokončení všech modelů je třeba všechny objekty vhodně vyexportovat. U 3D tisku je jedním z populárních formátů tzv. "stl", který využiji.

Příprava 3D tisku – program Slic3r

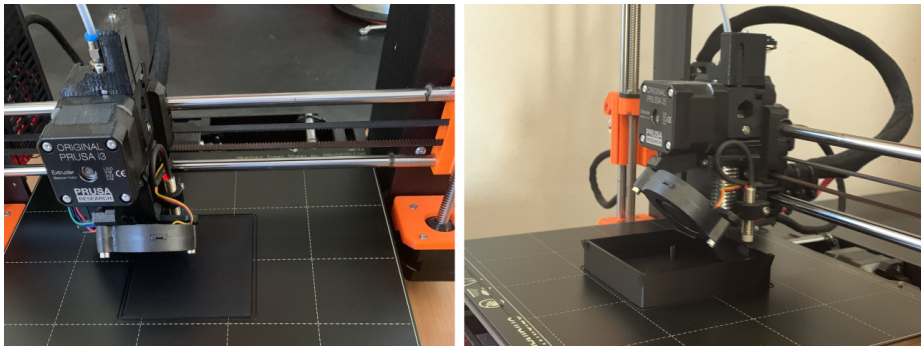
V poslední fázi tvorby zapouzdření využiji open-source software Slic3r, jehož vstupem bude vyexportovaný model v .stl formátu. Tento program rozdělí 3D objekt do horizontálních vrstev a vygeneruje G-code, což jsou instrukce pro koordinovaný pohyb motorů a extruderu. V záložce Print Settings lze nastavit důležité parametry 3D tisku jako: výška vrstvy, počet perimetrů, hustota výplně atd.



Obrázek 7.3: Náhled spodní části krytu na NACLEu v programu Slic3r.

3D tisk

K samotnému tisku využiji českou tiskárnu Original Prusa i3 MK3. Jedná se o jednu z nejlepších kartézských tiskáren na světě. K tisku byl zvolen filament z materiálu PLA (polylatic acid), což je termoplast vytvořený z obnovitelných zdrojů.



Obrázek 7.4: První vrstva obalu na NUCLEO (vlevo) a závěrečná fáze tisku (vpravo).

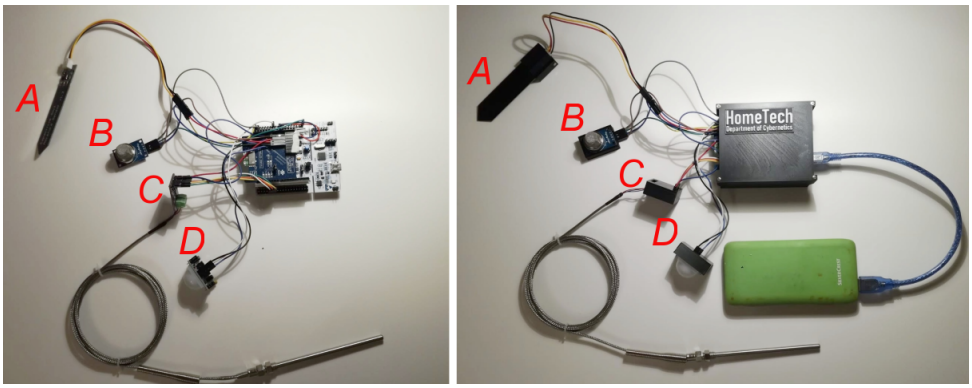
Ze zde uvedeného je zřejmé, jak snadno lze za pomocí dostupných SW nástrojů a s celkem běžnou 3D tiskárnou vytvořit i dostatečně kvalitní zapouzdření (obal) pro navrhovaný experimentální senzorový systém.

Kapitola 8

Výsledná podoba praktické části

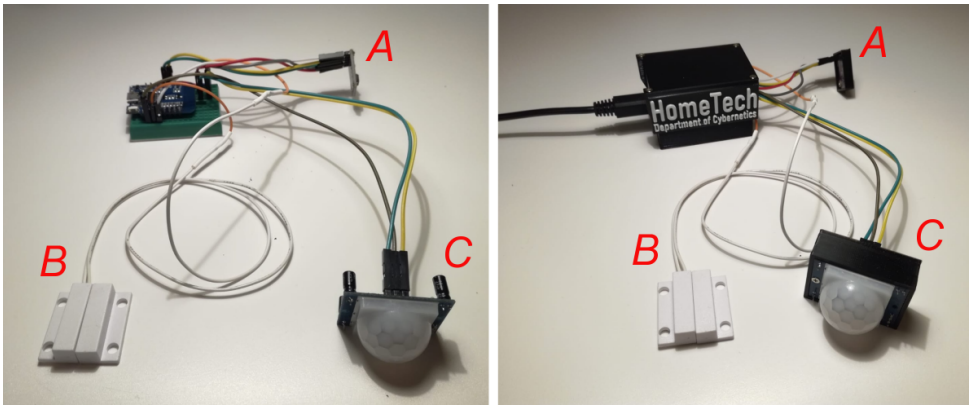
Uložením naprogramovaných vývojových desek s připojenými senzory do vytištěného zapouzdření získáváme finální podobu praktické části bakalářské práce. Následně bude otestována činnost celého systému.

První zařízení (napájené z powerbanky) je tvořeno vývojovou deskou NUCLEO L152RE se sériovým číslem 1234 komunikující s řídicí aplikací pomocí Bluetooth LE. Obsahuje 4 připojené senzory (A - senzor vlhkosti SEN10, B - senzor plynů MQ-5, C - senzor teploty MAX31865, D - senzor pohybu HC-SR50103), z nichž první jmenovaný je v programu zařízení zakomentován (blokován) – řídicí aplikace z tohoto senzoru data neobdrží.



Obrázek 8.1: Vytvořené demonstrační zařízení s NUCLEO L152RE (vlevo) a jeho výsledná zapouzdřená podoba (vpravo).

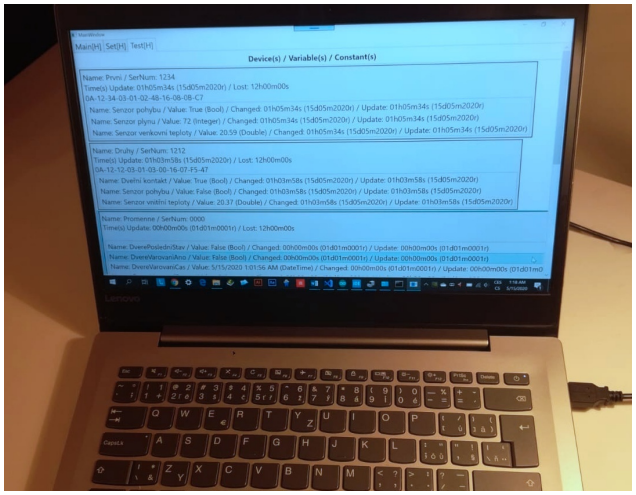
Druhé zařízení (napájené z USB portu notebooku) je tvořeno vývojovou deskou ESP8266 se sériovým číslem 1212 a komunikující s řídicí aplikací pomocí (virtuální) sériové linky (USART, na desce je použít převodník USB-USART, přes který je deska rovněž napájena). Obsahuje 3 připojené senzory (A - senzor teploty LM75A, B - dveřní kontakt, C - senzor pohybu HC-SR501).



Obrázek 8.2: Vytvořené demonstrační zařízení s modulem ESP8266 (vlevo) a jeho výsledná zapouzdřená podoba (vpravo).

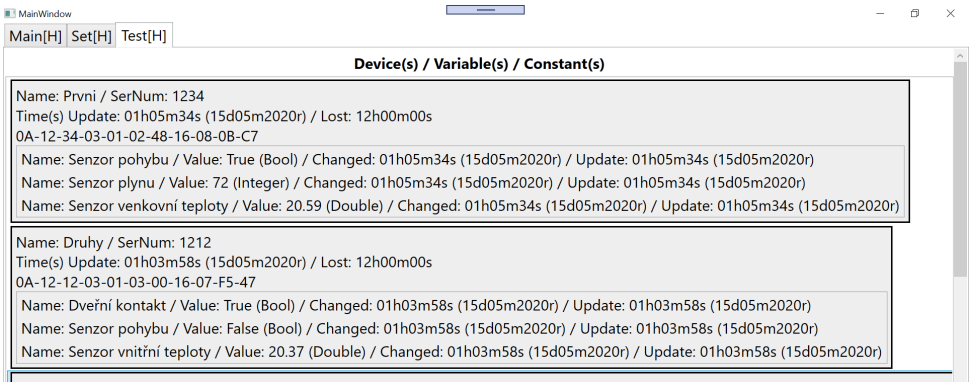
Obrázek 8.3: Detail na zapouzdření senzoru pohybu HC-SR501.

Řídicí aplikace je spuštěná pro demonstrační účely na notebooku. Lze sledovat příchozí data i na jiných mobilních zařízeních (tablet, smartphone atd.).



Obrázek 8.4: Řídicí aplikace na počítači.

Z výpisu aplikace jsou patrná data přicházející z obou senzorových zařízení. U prvního senzorového modulu (se sériovým číslem 1234) byl zaznamenán pohyb (hodnota True), procentuální hodnota senzoru plynu je 72% (na senzor byl vpuštěn plyn určený do zapalovačů) a venkovní teplota (pro demonstrační účely měřená uvnitř) má hodnotu 20.59°C. U druhého senzorového modulu (se sériovým číslem 1212) došlo k rozpojení dveřního kontaktu (hodnota True), senzor pohybu žádný pohyb nezaznamenal (hodnota False) a vnitřní teplota má hodnotu 20.37°C.



Obrázek 8.5: Detail výpisu demonstrační/testovací aplikace.

Ukázková aplikace byla vytvořena na základě programových částí/bloků poskytnutých vedoucím práce (například obsluha některých typů komunikací nemusí být nikterak snadná). Obsahovala tedy základní třídy pro komunikace (USART, BLE, UDP) a základ pro zobrazení. Pro vytvořené demonstrační senzorové moduly byla aplikace doplněna o potřebnou konfiguraci a další třídy zajištující vyzvedávání přijatých paketů/dat z jednotlivých typů komunikací, jejich vkládání do interní fronty aplikace, zpracování a nakonec zobrazení přijatých dat. Původní vytvořená a následně mnou upravená aplikace slouží pouze pro testování a demonstraci činnosti senzorových modulů nikoli jako plnohodnotná centrální aplikace pro zpracování přijatých dat. Z tohoto důvodu je v zobrazení dat přítomno množství dodatečných informací jako například čas posledního příjmu dat z příslušného senzoru, čas poslední změny veličiny ze senzoru a další, což mohou být velmi důležité informace při vývoji vlastního senzorického systému.

Demonstrační aplikace pro MS Windows a Google/Android je vytvořena v prostředí Microsoft .NET. a skládá se ze tří částí:

1. Část specifická pro MS Windows:
   - Zejména GUI a nízkoúrovňová obsluha některých typů komunikací.
   - Vytvořeno v .NET Framework/WPF (verze prostředí .NET pro MS Windows).
2. Část specifická pro Google/Android:
   - Zejména GUI a nízkoúrovňová obsluha některých komunikací.
   - Vytvořeno v Xamarin (verze prostředí .NET pro Google/Android).
3. Část platformově nezávislá:
   - Vysokoúrovňová obsluha komunikací a programový kód nezávislý na cílové platformě.

I když se v obou případech jedná o prostředí .NET a jazyk C# , některé části nelze vytvořit zcela platformově nezávislé. Příkladem je obsluha BLE komunikace na nejnižší úrovni při vyzvednávání tzv. „Adevertisement paketů“ zachycených z okolních zařízení. Každý systém k této skutečně nízkoúrovňové komunikaci přistupuje s poněkud jinou filozofií. Tyto činnosti jsou vytvořeny pro každou platformu samostatně, mají tedy odlišné zdrojové soubory. Již poněkud lepší situace je v oblasti GUI. V obou případech .NET (MS Windows a Google/Android) je pro popis GUI použit tzv. XAML a programový kód na pozadí. Syntaxe XAMLu a metody pro obsluhu GUI prvků nejsou sice zcela identické, ale zde již lze obluhu GUI rozdělit na platformově nezávislou a platformově společnou část. Společné soubory jsou vytvořeny mimo oba projekty a do každého projektu pouze tzv. „přilinkovány“.

Zcela ideální je stav ve zbytku aplikace. Nejen jazyk C#, ale i (většina) API prostředí .NET jsou pro obě platfomy zcela stejné (a stále více se přibližují). Tudíž ostatní soubory pro zpracování binárních nebo AsciiHex paketů, vytvoření jejich fronty, vyzvednutí v nich uložených dat a předání do zobrazení nebo detekce nežádoucích stavů jsou vytvořeny pouze jednou a do obou projektů opět pouze „přilinkovány“.

Stejným stylem lze v prostředí .NET vytvořit aplikaci i pro Apple/iOS (není však součástí této práce). Použitím (programového) prostředí .NET je rovněž demonstrována možnost snadné tvorby univerzálních, nebo vlastních testovacích a ladících nástrojů/aplikací vhodných při vývoji senzorického systému.

Kapitola 9

Závěr a diskuze

V této bakalářské práci byly zmíněny některé existující projekty a moduly a rovněž zhodnocen jejich přínos pro tvorbu experimentálních senzorových zařízení pro zejména vědecké účely. Na základě vykonané studie byl navržen a pomocí vybraných vhodných vývojových desek prezentován vlastní návrh pro vytvoření experimentálního zařízení pro sběr dat ze senzorů připojených přes různé datové sběrnice (I2C, SPI, ADC, atd.). Kromě samotného zdrojového kódu pro procesor ARM a modul ESP byly implementovány společné konfigurační soubory a funkce pro práci a připojení senzorů (drobnou úpravou zdrojového kódu lze snadno a rychle přidat nové senzory). Dále byly implementovány jak bezdrátové (WiFi, BLE), tak kabelové komunikační kanály (USART) umožňující přenos dat ze zařízení do řídicí/centrální aplikace. Pro miktokontroléry a demonstrační senzory bylo navrženo a pomocí 3D tisku vytištěno zapouzdření zakrývající odhalené elektronické kontakty. Součástí práce je i jednoduchá demonstrační aplikace usnadňující vývoj vlastního senzorického systému. Práci se tedy podařilo vyhotovit v úplném rozsahu.

Výstupem práce není konečné (kompletní) zařízení určené pro výrobu/prodej, ale komplexní návrh s podrobným popisem pro tvorbu vlastních experimentálních senzorových zařízení. Zde navržený postup tvorby senzorového systému může být tedy vhodným základem nejen pro prvotní experimentování s libovolnými senzory, ale rovněž jako začátek pro tvorbu i mnohem složitějších zařízení.

Další budoucí modifikace a rozšíření navrhovaného senzorického systému mohou zahrnovat například zabezpečení (šifrování) vysílaných dat, optimalizaci programu z hlediska odběru elektrické energie, umožňující napájet zařízení pomocí baterií a v neposlední řadě lze zařízení rozšířit o možnost připojení senzorů pomocí průmyslových datových sběrnic CAN, RS-232 atd.

Příloha A

Literatura

[1] Noviello, Carmine, *Mastering STM32 [PDF]*, Listopad 2016, Itálie, Leanpub

[2] Owsiak, Tom, *Beginning C# 7 Hands-On – The Core Language [PDF]*, Srpen 2017, Packt, ISBN-13: 978-1788294263

[3] Liberty, Jesse, *Windows 10 Development with XAML and C# 7*, Prosinec 2017, Apress, ISBN-13: 978-1484229330

[4] *HAL User Manual*, `https://www.st.com/content/ccc/resource/technical/document/user_manual/97/4d/5f/9a/ed/e4/4e/66/DM00132099.pdf/files/DM00132099.pdf/jcr:content/translations/en.DM00132099.pdf`, [Online, navštíveno 17. 5. 2020]

[5] *L152RE User Manual*, `https://www.st.com/resource/en/user_manual/dm00105823-stm32-nucleo-64-boards-mb1136-stmicroelectronics.pdf`, [Online, navštíveno 17. 5. 2020]

[6] *L152RE Reference Manual*, `https://www.st.com/content/ccc/resource/technical/document/reference_manual/cc/f9/93/b2/f0/82/42/57/CD00240193.pdf/files/CD00240193.pdf/jcr:content/translations/en.CD00240193.pdf`, [Online, navštíveno 17. 5. 2020]

[7] *St Company*, `https://www.st.com/content/st_com/en/about/st_company_information/who-we-are.html`, [Online, navštíveno 17. 5. 2020]

[8] *Espressif Company*, `3.https://www.espressif.com/en/company/about-us/who-we-are`, [Online, navštíveno 17. 5. 2020]

[9] *STM32CubeIDE*, `https://www.st.com/en/development-tools/stm32cubeide.html#overview`, [Online, navštíveno 17. 5. 2020]

[10] *MQ-5 gas sensor*, `https://wisense.wordpress.com/2015/07/06/gas-leakage-module-using-the-mq-5-gas-sensor/`, [Online, navštíveno 17. 5. 2020]

[11] *Obrázek 2.1*, https://www.pomoceszkolne24.pl/lynxmotion-al5b-robotic-arm-kit/9614/, [Online, navštíveno 17. 5. 2020]

[12] *Obrázek 2.2*, https://arduino.cz/stante-se-vynalezcem-s-littlebits/, [Online, navštíveno 17. 5. 2020]

[13] *Obrázek 2.3*, https://blog.techiehunter.org/the-use-of-arduino-in-the-iot-wearable-and-other-handy-devices-has-been-inc [Online, navštíveno 17. 5. 2020]

[14] *Obrázek 2.4*, https://www.digikey.com/product-detail/en/espressif-systems/ESP32-WROOM-32/1904-1010-1-ND/8544305, [Online, navštíveno 17. 5. 2020]

[15] *Obrázek 2.5*, https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html, [Online, navštíveno 17. 5. 2020]

[16] *Obrázek 2.6*, https://grobotronics.com/stm32-nucleo-development-board-for-stm32-f1-series-with-stm32.html?currency=EUR&sl=en, https://au.rs-online.com/web/p/processor-microcontroller-development-kits/1438574/, https://ph.rs-online.com/web/p/processor-microcontroller-development-kits/9173775/, https://www.aliexpress.com/item/32754485004.html, [Online, navštíveno 17. 5. 2020]

[17] *Obrázek 2.7*, https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus:GENERAL-PURPOSE-MCUS, [Online, navštíveno 17. 5. 2020]

[18] *Obrázek 2.8*, https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/lpc5500-cortex-m33/lpcxpresso55s69-development-board:LPC55S69-EVK,https://www.nxp.com/design/development-boards/freedom-development-boards/mcu-boards/nxp-freedom-development-platform-for-k32-l3-mcus:FRDM-K32L3A6, [Online, navštíveno 17. 5. 2020]

[19] *Obrázek 2.9*, https://www.arrow.com/en/reference-designs/atsamv71-xult-sam-v71-xplained-ultra-evaluation-kit-used-to-evaluate-the-ats d30be1d8d2c2e45aeb196cf872c9d054, http://ww1.microchip.com/downloads/en/devicedoc/atmel-42067-sam4e-ek_userguide.pdf, https://www.thedebugstore.com/bigavr6-development-board-mikroe-456.html, https://www.radiolocman.com/op/device.html?di=105600&/ATAVRXPLAIN, [Online, navštíveno 17. 5. 2020]

[20] *Obrázek 5.1*, https://obchod.hw.cz/eshop/magneticky-dverni-kontakt-mk4/, [Online, navštíveno 17. 5. 2020]

[21] *Obrázek 5.2*, https://cz.farnell.com/dfrobot/sen0193/analog-capacitive-soil-moisture/dp/2946124, [Online, navštíveno 17. 5. 2020]

[22] *Obrázek 5.3*, https://www.instructables.com/id/How-to-Use-PIR-Motion-SensorHC-SR501with-an-Arduin/, [Online, navštíveno 17. 5. 2020]

[23] *Obrázek 5.4*, https://elty.eu/cs_CZ/p/MQ-7-Senzor-oxidu-uhelnateho/1449, [Online, navštíveno 17. 5. 2020]

[24] *Obrázek 5.5*, http://robotstore.cz/obchod/senzory/lm75a-i2c-arduino-modul-pro-mereni-teploty/, [Online, navštíveno 17. 5. 2020]

[25] *Obrázek 5.6*, https://www.aliexpress.com/i/32804522819.html, [Online, navštíveno 17. 5. 2020]

[26] *Obrázek 5.7*, https://www.st.com/content/ccc/resource/sales_and_marketing/promotional_material/brochure/6c/48/c0/f1/bb/35/4a/b4/brstm32ulp.pdf/files/brstm32ulp.pdf/jcr:content/translations/en.brstm32ulp.pdf, [Online, navštíveno 17. 5. 2020]

[27] *Obrázek 5.9*, https://www.st.com/en/ecosystems/x-nucleo-idb05a1.html, [Online, navštíveno 17. 5. 2020]

Příloha B

Použité zkratky

- ADC - Analog to Digital Converter
- ARM - Advanced Risc Machine
- BLE - Bluetooth Low Energy
- CS - Chip Select
- GND - Ground
- GPIO - General Purpose Input Output
- HAL - Hardware Abstraction Layer
- HW - Hardware
- I2C - Inter Integrated Circuit
- IoT - Internet of Things
- LAN - Local Area Network
- SPI - Serial Peripheral Interface
- SW - Software
- UDP - User Datagram Protocol
- USART - Universal Synchronous/Asynchronous Receiver and Transmitter
- VCC - Voltage supply
- WiFi - Wide Fidelity

Příloha C

Obsah CD

- Řídicí/centrální aplikace.
- Zdrojové soubory pro procesor ARM.
- Zdrojové soubory pro modul ESP8266.
- Zdrojové soubory pro 3D tisk zapouzdření vývojových desek.