VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

IOT SYSTÉM PRO DOMÁCNOST

IOT HOME SYSTEM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. VIKTOR KOVAŘÍK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. TOMÁŠ GOLDMANN**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2019/2020

Zadání diplomové práce

Student: **Kovařík Viktor, Bc.**
Program: Informační technologie Obor: Počítačové a vestavěné systémy
Název: **IoT systém pro domácnost**
IoT Home System
Kategorie: Vestavěné systémy
Zadání:

1. Nastudujte a sumarizujte základní informace o IoT systémech. Zjistěte, jaké protokoly se v těchto systémech používají. Prostudujte a analyzujte 3-4 IoT systémy používané v inteligentních domácnostech. Zaměřte se podrobněji na technologii Google Home.
2. Sumarizujte informace o mikrokontroléru ESP32.
3. Navrhněte řešení pro řízení inteligentní domácnosti s využitím technologie Google Home. Systém bude na základě dat z meteostanice, realizované v rámci bakalářské práce, ovládat žaluzie a upravovat intenzitu venkovního osvětlení. Dále navrhněte vlastní řídící modul s platformou ESP32 pro ovládání branky a garážových vrat.
4. Implementujte potřebný software pro realizaci systému inteligentní domácnosti a implementujte uživatelské rozhraní pro ovládání systému. Dále pak realizujte navržené hardwarové řešení k řízení aktuátorů.
5. Systém nasaďte v reálné domácnosti a otestujte.

Literatura:

- PFISTER, Cuno. Getting Started with the Internet of Things: Connecting Sensors and Microcontrollers to the Cloud. " O'Reilly Media, Inc.", 2011.
- CHOU, Timothy. Precision-Principles, Practices and Solutions for the Internet of Things. McGraw-Hill Education, 2017.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Goldmann Tomáš, Ing.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 3. června 2020
Datum schválení: 31. října 2019

Abstrakt

Cílem této diplomové práce bylo seznámit se a sumarizovat základní informace o IoT systémech, jaké protokoly se používají a zaměřit se na technologii Google Home. V první části práce jsou popsány jednotlivé části systému — mikrokontroléry, senzory, světelné prvky a možné systémy pro backend. V rámci implementační části práce bylo navrhnuto řešení pro řízení inteligentní domácnosti s využitím technologie Google Home. Systém na základě dat z meteostanice ovládá a upravuje intenzitu venkovního osvětlení a ovládá žaluzie. Dále je implementován řídicí modul pro ovládání branky a garážových vrat. Systém se také stará o plánování vysávání v domě s využitím vysavačů iRobot Roomba, do kterých je přidán modul s Wi-Fi. Závěrem jsou shrnuty dosažené výsledky.

Abstract

The aim of this thesis was to learn and summarize basic information about IoT systems, which protocols are used and introduction of Google Home system. The first part of the thesis describes the individual parts of the system — microcontrollers, sensors, light elements and possible systems for backend. In the implementation part of the thesis was designed a solution for smart home controling using Google Home technology. Based on data from the weather station, the system controls and adjusts the intensity of outdoor lighting and controls the blinds. Furthermore, a control module for gate and garage door control is implemented. The system also takes care of vacuum cleaning in the house using iRobot Roomba vacuum cleaners with custom Wi-Fi module. The final part of the thesis summarizes the achieved results.

Klíčová slova

autonomní žaluzie, Somfy, automatizace, ESP32, ESP8266, Python, meteostanice, senzory, Tuya, Sonoff, chytré žárovky, inteligetní zvonek, ovládání branky, ovládání garážových vrat, Google Home, Home Assistant, ESPHome, iRobot Roomba

Keywords

autonomous blinds, Somfy, automation, ESP32, ESP8266, Python, weather station, sensors, Tuya, Sonoff, smart bulbs, intelligent doorbell, entrance gate control, garage door control, Google Home, Home Assistant, ESPHome, iRobot Roomba

Citace

KOVAŘÍK, Viktor. *IoT systém pro domácnost.* Brno, 2020. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Tomáš Goldmann

IoT systém pro domácnost

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Tomáše Goldmanna. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Viktor Kovařík
10. června 2020

Poděkování

Rád bych poděkoval mému vedoucímu práce panu Ing. Tomáši Goldmannovi za odbornou konzultaci při realizaci této diplomové práce. Dále bych poděkoval slečně Bc. Martině Grzybowské za pomoc při korektuře gramatiky i stylistiky. Velké poděkování též patří mojí rodině na obrovskou podporu, rodinné zázemí a ochotu využívat nové technologie. Zvláštní poděkování patří též mému brantanci Mgr. Janu Knotkovi za rady s programováním v jazyce Dart a s využíváním platformy Flutter se kterou jsem se do té doby nesetkal. Děkuji taky mým kamarádum Bc. Juraji Kubišovi, Bc. Jendovi Bartoňovi a Adéle Bartoňové-Slováčkové, Bc. Patriku Velemu, Bc. Filipu Jaškovi a Bc. Václavu Hodulíkovi za dobrou pohodu a příjemné diskuze.

Obsah

Kapitola 1

Úvod

Svět IoT systémů je v současné době velmi různorodý. Je mnoho druhů zařízení, která využívají různých technologií přenosu dat a způsobů ovládání. Máme mnoho výrobců, kteří vyrábí svá zařízení pro použití pouze s jejich vlastními (nebo vybranými) systémy. Lze ale najít i systémy, které je možné napojit alespoň na nějakou významnější cloudovou či lokální službu (či bránu) pro propojení mimo uzavřený ekosystém. Existují i systémy otevřené, které jsou někdy i pečlivě zdokumentovány, a možné je proto využít v určitých případech i v systémech třetích stran.

Co se týče IoT služeb či platforem, tak zde máme taky mnoho voleb. Můžeme využít hlasových asistentů Siri, Alexa nebo třeba Google Assistant a jejich řešení pro chytrou domácnost (Apple HomeKit, Google Home...). Každý ze systémů je různě modifikovatelný, různě otevřený a umožnit nějakému zařízení pracovat se všemi je poměrně složité. Jsou také otevřená řešení jako ESPEasy, Home Assistant a OpenHAB.

Chytrá zařízení lze zakoupit nebo dokonce je možné si je vyrobit a naprogramovat. U průmyslových řešení se spíše využívá komerčních jednoúčelových mikrokontrolérů, které do určité míry jdou uzpůsobit pro potřebu firmy. Běžný uživatel, který vyhledává jednoduché řešení, si koupí již hotový hardware (senzor, aktuátor), který lze využít s jeho stávajícím systémem (případně tím, který plánuje pořídit). Třetí možností je využít některých zdokumentovaných senzorů, mikrokontrolérů typu ESP32 nebo třeba linuxové SBC (třeba Raspberry Pi) a takové chytré zařízení realizovat po svém.

Cílem této práce je tedy sumarizovat základní informace o IoT systémech, analyzovat několik systémů používaných v inteligentních domácnostech a také se zaměřit na technologii Google Home. Dále je představen mikrokontrolér ESP32 a jeho možnosti využití v inteligentních domácnostech. Rovněž je realizováno ovládání venkovního osvětlení na základě dat z nové meteostanice 2.0, která je vybavena oproti meteostanici z bakalářské práce novým vlastnoručně navrženým HW a kompletně předělaným SW. Nově je do inteligentního systému připojeno ovládání branky a detekce zvonění včetně notifikace pomocí audiovizuálních signálů v místech, kde není zvonek slyšet. Systémem je možné ovládat i garážovou bránu a starší modely vysavače iRobot Roomba, které nedisponují takovýmto ovládáním.

Práce sestává z několika částí. Úvodem je představen systém Google Home a další existující IoT systémy. Na to navazuje mikrokontrolér ESP32 a jeho různé varianty, router Turris, jednodeskový počítač Odroid HC1 a Raspberry Pi. Následuje představení komponent, ze kterých sestává autonomní systém a jejich možnosti ovládání. V druhé části textu práce je vysvětlena implementace jednotlivých komponent systému a propojení akčních členů mezi sebou. Je také představeno uživatelské rozhraní pro ovládání systému. Závěrem práce je popsán průběh nasazení a otestování systému v reálné domácnosti.

Kapitola 2

IoT systémy

Internet of Things (IoT) lze chápat jako síťové propojení zařízení, senzorů a výpočetních jednotek, které spolu navzájem komunikují. IoT systém se skládá nejčastěji ze senzorů, aktuátorů, služeb, sítě a dat. Následující kapitola pojednává o těchto systémech a IoT platformách.

2.1 Prvky IoT systémů

Typický IoT systém se skládá z koncových zařízení (senzorické jednotky, zobrazovací jednotky a aktuátory), zdroje napětí, brány a služeb. Vstupní koncová zařízení jsou senzorické moduly, kterými lze zaznamenávat fyzikální i nefyzikální veličiny. Skládá se obvykle ze senzoru, mikrokontroléru a komunikační jednotky. Senzor s mikrokontrolérem komunikuje obvykle pomocí GPIO (pulzy, pravdivostní hodnoty), pomocí pinu s A/D převodníkem (například voltmetr) nebo přes sběrnice (USB, I^2C, SPI, UART) [27]. Výstupní koncové zařízení je aktuátor nebo zobrazovací jednotka. Typickým aktuátorem je například topení či motor.

Pro napájení IoT zařízení se obvykle využívají baterie nebo napájení ze sítě. Mimo to ale IoT systémy mohou svoji energii čerpat z jiných zdrojů (tzv. energy harvesting) a to například v místech, kde není zajištěna stálá dodávka elektrické energie. Zařízení být napájeno ze solárních panelů, mechanickými pohyby (dynamo), radiofrekvenční energií nebo termálních energií.

Dalším prvkem IoT systému je brána (gateway), která je hardwarové zařízení nebo program, který propojuje senzorické moduly s internetem. Může na ní běžet software, který zpracovává data, případně i umožňuje jejich vizualizaci. Umožňuje komunikaci se staršími zařízeními či zařízeními bez konektivity k internetu, zpracovává a agreguje data a umožňuje konfiguraci zařízení. Komunikace probíhá často jako M2M (*many-to-many*) a brána často zajišťuje určitý stupeň zabezpečení systému.

Služba, též nazývaná jako cloud, je soubor výpočetních prostředků, které slouží k shromažďování dat, jejich zpracování a uchovávání. Cloud nám poskytuje možnost vzdáleného spravování zařízení a senzorů, zpracovávat data odkudkoliv v reálném čase a velkou možnost škálování (výkonu i velikosti úložiště). Cloudové systémy obsahují plnohotnotné výpočetní server a díky tomu je možné využívat i vyšší programovací jazyky [27].

2.2 IoT platformy

Existuje mnoho IoT platforem, které propojují jak senzorické jednotky, tak cloud, akční členy a zobrazovadla do jednoho celku. Firmy jako Apple, Google či Amazon umožňují s IoT systémem pracovat i hlasovými příkazy. V následující sekci jsou představeny IoT platformy od výše zmíněných firem, hardware, který vyrábí či podporují. Je také představeno otevřené řešení pomocí platformy Home Assistant.

2.2.1 Google Home

Google Home je IoT platforma, která v sobě integruje hlasového asistenta Google Assistant, hardware, jako chytré reproduktory, displeje a mobilní aplikaci, která slouží k ovládání všech připojených zařízení.

Google Assistant

Google Assistant je hlasový pomocník, který se podobá na další systémy jako Amazon Alexa nebo Siri od Apple. Je často součástí telefonů se systémem Android a dalších vyrobcích od Google [25]. Dále existuje označení „Google Assistan built-in“, které sdružuje kompatibilní zařízení jako chytré reproduktory [6] a smart TV založené na Android TV platformě [7].

Hovor s asistentem se zpravidla uvozuje hláškou „Ok Google“, případně „Hey Google“. Pomocníka lze taky aktivovat na mobilních telefonech kliknutím na logo mikrofonu. S Google Assistantem lze komunitovat také formou textu [47].

Chytré reproduktory a displeje

Google má své porfolio chytrých reproduktorů a displejů, které obsahují hlasového asistenta (sekce 2.2.1) a zároveň jej doplňují o zajímavé funkce.

Reproduktory:

Původní Google Home Originální Google Home byl vydán 4. listopadu 2016 a je stále v prodeji. Obsahuje reproduktor, mikrofon a dotykovou plochu na nastavení hlasitosti. Také obsahuje tlačítko pro vypnutí mikrofonu v případě, že si uživatel nepřeje být slyšen [37].

Google Home Mini Je menší verze původního Google Home. na rozdíl od původního reproduktoru má vypínač mikrofonu, který mikrofon odpojí fyzicky (originální pouze softwarově zakáže snímání zvuku). Jeho cena se nyní pohybuje okolo 25 USD, což z něj dělá nejlevnější zařízení, které lze pořídit s Google Assitentem.

Google Home Max Větší a dražší alternativa Google Home Mini.

Google Nest Mini Nová generace, která navazuje na původní Google Home Mini. na rozdíl od původního má tento kulatý napájecí konektor, AI koprocessor na rychlejší zpracování příkazů a možnost pověšení na zeď. Jeho cena odpovídá ceně původní generace při vydání a předpokládá se, že tento výrobek nahradí ten původní, jakmile se vyčerpají skladové zásoby.

Displeje:

Google Nest Hub Smart displej Google Nest Hub kombinuje reproduktor na přehrávání hudby, hlasového asistenta a displej pro zobrazování důležitých informací pro své uživatele. Podporuje ovládání zařízení jako jsou chytrá světla, vypínače, zásuvky nebo pračka přímo ze zařízení, takže není potřeba říkat žádný hlasový příkaz ani není nutné použít mobilní aplikaci Google Home [26].

Google Nest Hub Max Je větší verze Google Nest Hub, která obsahuje výkonnější reproduktory a větší displej.

Značka Nest Jak je možné vidět, tak nověji vydaná zařízení (a některá starší) byla přejmenována z Google Home na Google Nest. Toto přejmenování vzniklo jako snaha Google více integrovat ekosystém od firmy Nest, kterou koupil do svého portfolia. Očekává se, že již všechna nová Google zařízení pro chytrou domácnost budou vycházet pod značkou Google Nest [24].

Made for Google

Made for Google je značka, která signalizuje, že produkt, který ji má byl navržen pro fungování s Google ekosystémem. Některé výrobky opatřené touto značkou jsou například oficiální obaly a příslušenství na řadu Google Pixel [12], ale často toto označení společně s Google Assistant built-in (sekce 2.2.1) najdeme na kompatibilních reproduktorech třetích stran. Mezi ně patří například chytrý budík Lenovo Smart Clock, smart displej Lenovo Smart Display, některé reproduktory firem Sonos, Harman, LG nebo Sony.

Kromě reproduktorů jsou značkou Made for Google označeny výrobky firmy Nest (nyní součást Googlu), některé Bluetooth žárovky od firmy Philips (není potřeba brána a fungují přímo) nebo například žárovky C by GE.

Podpora pro zařízení třetí strany

Google Home platforma samozřejmě mimo to podporuje zařízení třetích stran. Jejích podpora je poměrně rozsáhlá, systém podporuje bez jakýchkoliv úprav více než sto různých výrobců chytrých zařízení.

Mezi aktuálně podporovaná zařízení dle dokumentace mimo jiné patří [15]:

- klimatizace, osvěžovače a čističe vzduchu,
- markýzy, žaluzie, rolety, stínítka, okna, dveře, skříně,
- vany, mixéry, postele, kávovary, sporáky, trouby, pračky, sušičky, myčky, fritézy, výrobníky jogurtů,
- vytápění, boilery, termostaty,
- světla, vypínače, žárovky, měřiče spotřeby,
- vysavače, mopy, sekačky na trávu.

Všechna tato zařízení mají své akce, které u nich jdou využít a mapovat je na vstupy a výstupy vlastního cloudového řešení pomocí API. Taky existují služby[1], které například

[1] https://gbridge.io/

umí pro některé tyto komponenty vytvořit bridge např do MQTT a tím pádem jednodušeji propojit vlastnoruční řešení s ekosystémem Google Home. Samozřejmě si lze toto napojení vyrobit i vlastní cestou. Pokud se vývojář rozhodne vyrobit si vlastní napojení, je potřeba si vytvořit testovací projekt v Google Actions Console[2]. Poté se zvolí typ projektu, v tomto případě tedy „Smart Home“. Pak je nutné zvolit způsob autentizace a adresu serveru, kde běží backend. Google poté komunikuje pomocí zdokumentovaných webhooks se serverem, který poté požadavek přináší jednotlivým zařízením [4].

Aplikace Google Home

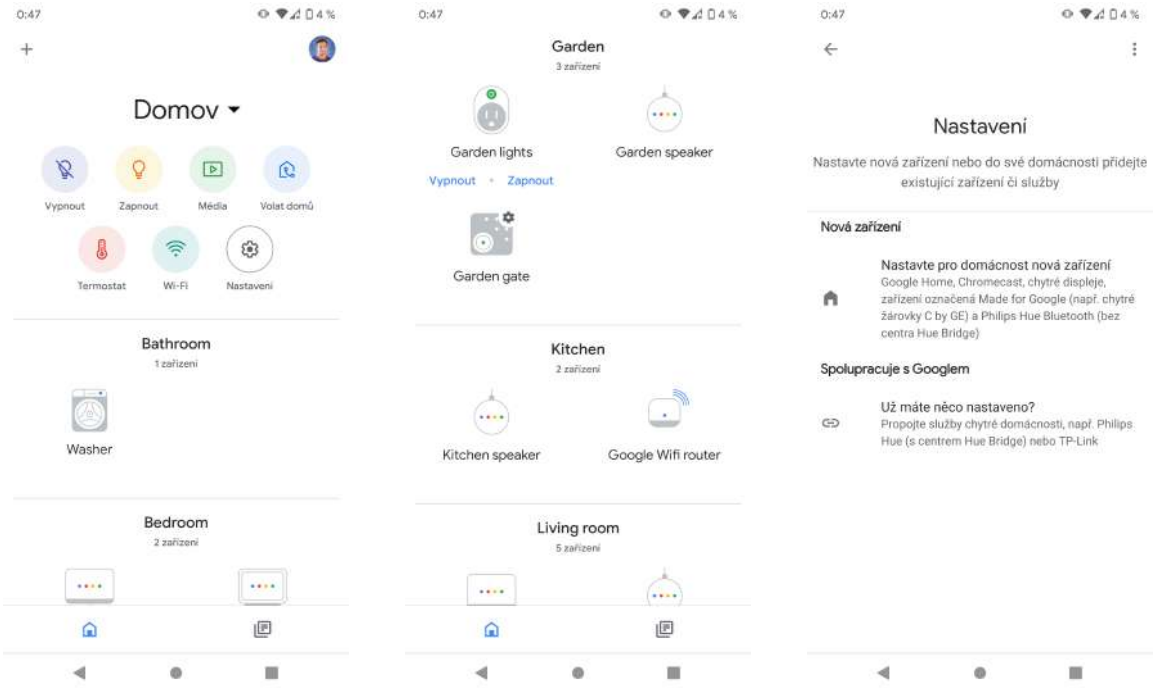
Pro propojení zařízení v domácnosti, jejich nastavování a celkový přehled existuje aplikace Google Home. Je nutné ji mít, jinak nepůjdou zařízení spárovat s Google účtem a komunikovat s asistentem.

Na hlavní obrazovce aplikace se nachází seznam zařízení ve zvolené domácnosti rozdělený podle pokojů (obrázek 2.1). Domácnosti lze libovolně přidávat a přepínat, lze pozvat do své domácnosti svoji rodinu, která má potom stejná práva na každé zařízení. Hlasoví asistenti v domě dokážou rozeznat až 6 různých hlasových profilů [11], a tím pádem každému navrhnout personalizované výsledky typu události v kalendáři nebo doba cesty do práce při aktuální dopravní situaci.

Dále aplikace obsahuje sekci pro přidání dalších zařízení, tam lze buď vybrat, zda chceme přidat zařízení, které je certifikováno jako Made for Google (sekce 2.2.1), nebo zda chceme Google Home propojit se službou třetí strany (sekce 2.2.1).

Aplikace dále umožňuje upravovat hlasitosti jednotlivých reproduktorů, spojovat je do více celků a tím pádem mít možnost poslouchat například hudbu ve více místnostech [3]. Dále je možné některá zařízení ovládat přímo z aplikace bez použití hlasového příkazu (například žárovky, vypínače nebo termostaty). Některé poměrně nově přidaná zařízení nelze ještě z aplikace ovládat přímo, je to možné pouze hlasovým příkazem. Tyto zařízení se zobrazují v aplikaci s logem ozubeného kola a po kliknutí na jejich ikonku aplikace místo obvyklé obrazovky pro ovládání zobrazí základní nastavení daného zařízení[3].

[2] https://console.actions.google.com/
[3] https://support.google.com/googlenest/thread/570067



Obrázek 2.1: Aplikace Google Home.

2.2.2 Amazon Alexa

Alexa je hlasový asistent od firmy Amazon. Jako první se Alexa objevila v roce 2014 na zařízeních Amazon Echo a Echo Dot, která jsou obdobou Google Home a Google Home Mini [19]. V době svého vydání byl tento systém jeden z prvních svého druhu (byl dříve než Google Home i Apple HomeKit, ti měli v tu dobu pouze hlasové asistenty pro mobilní telefony) [40].

Amazon Alexa má také svoje displeje a reproduktory. U reproduktorů má 3 základní modelové řady, Amazon Echo, který je základní smart speaker od firmy Amazon, který je modelovou řadou srovnatelný s výrobkem Google Home. Dále Amazon Echo Dot, které velmi podobné Google Home Mini, avšak v některých provedeních obsahuje také ukazatel času[4] na rozdíl od varianty od Google. Posledním zástupcem je Amazon Echo Plus, prémiový chytrý reproduktor, který je větší a obsahuje kromě samotných reproduktorů také vestavěný senzor teploty. Navíc obsahuje také ZigBee modul, a tudíž slouží jako Wi-Fi bridge pro různé žárovky, senzory, vypínače či zámky [39].

Firma Amazon dále vydala zařízení Amazon Echo Input[5], který je pouze chytrým mikrofonem a jeho výstup se připojuje na stávající domácí audiosystém. Výhodou poté je, že ze starých, „hloupých“ reproduktorů lze udělat zařízení typu Amazon Echo, které ještě v závislosti na kvalitě zvolených reproduktorů může dosahovat i kvalitnějšího zvukového podání než oficiální Echo reproduktory.

Kromě reproduktorů má Amazon i své smart displeje. Jsou označeny, jakou Amazon Echo Show. Aktuálně jsou na trhu 3 varianty, Amazon Echo Show (druhé generace), což je chytrý displej s úhlopříčkou 10.1"a ZigBee technologií. Amazon Echo Show 8 je menší a levnější varianta předchozího, má pouze 8" displej a procesor Intel Atom zde byl vyměněn

[4] https://www.amazon.com/gp/product/B07N8RPRF7/
[5] https://www.tomsguide.com/us/amazon-echo-input,review-5762.html

za Mediatek[6]. 5 palcový Amazon Echo Show 5 je jak svojí velikostí, tak i tvarem velmi podobný[7] budíku Lenovo Smart Clock, jež má certifikaci Made for Google (sekce 2.2.1). Amazon Echo Spot je nejmenší smart displej od Amazonu, má kulatý displej a je vhodný pro použití jako asistent na noční stolek [45].

Amazon své výrobky Echo pravidelně aktualizuje a vydává jejich nové generace. Výrobky se liší zejména po vzhledové stránce, avšak některé z nich obsahují i zásadnější rozdíly ve funkcionalitě [43]. Podobně jako Google Home má hlasového asistenta Alexa i několik výrobků třetích stran [34]. Mezi ně patří například zařízení od výrobců Sonos, Riva, Anker, Bose a další. Jedná se však pouze o reproduktory.

2.2.3 Apple HomeKit

Systém pro chytrou domácnost Apple HomeKit sestává z chytrých senzorů, kamer, spínačů, zásuvek a světel, které překvapivě Apple vůbec nevyrábí [8], ale nechává to na výrobcích třetích stran. Dále systém také umožňuje použití chytrých reproduktorů Apple HomePod a samozřejmě Siri asistenta, který je také jeho součástí [9]. Poslední součástí tohoto ekosystému je aplikace Apple Home, která je pro iOS telefony, tablety [10] a dokonce i pro počítače se systémem MacOS (Alexa ani Google Home neobsahuje desktopovou aplikaci) [13].

2.2.4 Home-Assistant a ESPhome

Home assistant je otevřená platforma pro chytrou domácnost. Mezi její hlavní výhody patří obrovská podpora pro různé druhy senzorů, dále pak možnost přidat si jakýkoliv senzor od jakéhokoliv výrobce, jehož API je zdokumentováno a také podpora MQTT, která se u komerčních IoT systémů nevyskytuje [28].

Mimo potřeby doprogramovat jednotlivé senzory a jejich rozhraní lze některé senzory, služby nebo konkurenční platformy přidat za pomoci takzvaných integrací. Namátkově Home Assistant podporuje integraci například Google Cast, která umožňuje posílat textové zprávy přímo Google asistentům, pouštět hudbu na reproduktorech a obsah v Android TV zařízeních. Dále je pak podporován například HomeKit, IFTTT, Z-WAVE protokol, některé brány ZigBee, platforma Smartthings, TP-Link Smart Home nebo například OpenTherm [20].

ESPHome

Za zajímavou integraci lze určitě považovat platformu ESPHome. Tato platforma umožňuje nakonfigurovat mikrokontroléry ESP32 a ESP8266 pomocí jednoduchých konfigurací [16] a tím logicky oddělit knihovny a jejich konfiguraci od sebe. Taky usnadňuje asynchronní přístup k jednotlivým komponentám mikrokontroléru [2], zotavení z chyb a také se například na ní velmi dobře získávají a filtrují data ze senzorů [14].

Při zprovoznění integrace v Home Assistantu se dané zařízení a všechny jeho veřejná rozhraní objeví v dashboardu webové aplikace (díky tzv. NativeAPI) a systém je okamžitě připravený k provozu. Jelikož obecné senzory (například garážová vrata) kopírují strukturu Home Assistantu, tak je integrace do tohoto systému přímá včetně všech funkcionalit a není potřeba komponentu dál upravovat. V případě, že se kód v ESPHome jakkoliv změní, upraví se i rozhraní Home Assistenta a tím pádem i případná logická propojení [5].

[6] https://www.androidcentral.com/echo-show-8-vs-echo-show-2nd-gen
[7] https://www.techradar.com/news/amazon-echo-show-5-vs-lenovo-smart-clock

Konfigurace senzorů je opravdu jednoduchá, přímočará a přehledná. Konfigurace senzoru teploty BME280 v ESPHome platformě se realizuje pouze několika řádky kódu[8]. Zkonfigurovat tento senzor ve Wiring a ArduinoIDE by bylo mnohem komplikovanější, méně přehledné a bylo by potřeba řešit, jakým způsobem poté data poslat na server [41].

2.3 Komunikace

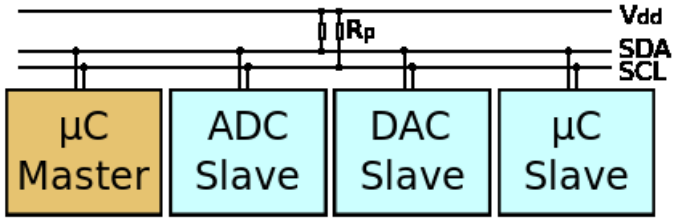
Aby IoT systém vůbec fungoval, je potřeba, aby nějakým způsobem komunikoval. Tato kapitola se tedy zaměřuje na komunikační sběrnice na mikrokontrolérech, dále pak na často využívané bezdrátové technologie v IoT a taky na to, jakým komunikačním protokolem tato data putují.

2.3.1 Sběrnice

Existuje mnoho rozhraní, přes která mohou senzory a zařízení komunikovat a některá jsou vhodná i pro IoT. V následující podsekci jsou popsána známá komunikační rozhraní, která jsou využívána v širokém spektru zařízení.

I²C

I²C (*Inter-Integrated Circuit*) je seriová sběrnice původně vyvinutá firmou Philips, která slouží k nizkorychlostnímu připojování periférií. Je využívána pro připojení periferií k vestavěným systémům a jiným zařízením (využívá se například pro vyčítání hodin RTC, přístup k různým senzorům, ovládání motorů a podobně). Pracuje buď v režimu master nebo slave, kde master iniciuje přenosy a generuje hodinový signál SCL. Sběrnici je možné používat v režimu SingleMaster, která je jednodušší na implementaci a slouží k propojení jednoho master uzlu s několika slave uzly, dále pak je ale možné použít sběrnici v módu MultiMaster, kdy se zavádí proces arbitráže a bus busy detekce. Sběrnice je adresovaná, takže pouze s pomocí dvou vodičů (hodiny a datový vodič) a napájení dokáže propojit až 128 různých zařízení [38].



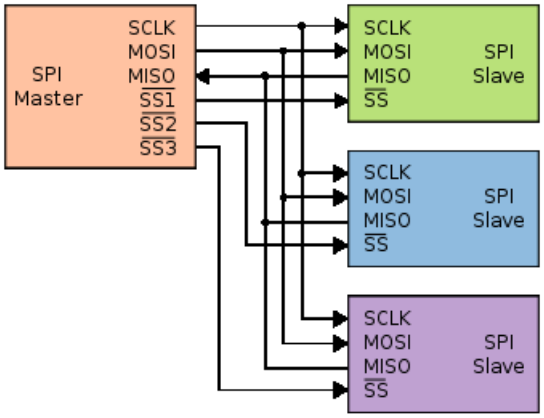
Obrázek 2.2: I²C rozhraní s více zařízeními[9].

SPI

SPI (*Serial Peripheral Interface*) je komunikační sériové rozhraní, které se využívá také k nízkorychlostní komunikaci mezi zařízeními u vestavěných systémů. Komunikace probíhá

[8] https://esphome.io/components/sensor/bme280.html
[9] Převzato z https://commons.wikimedia.org/wiki/File:I2C.svg

přes dva vodiče, jeden je MISO (*master in slave out*) a MOSI (*master out slave in*), každý tedy obstarává jeden směr komunikace a je to tedy obousměrné spojení (*full-duplex*). Zařízení, které v této komunikaci je iniciátor spojení (*master*), generuje synchronizační signál (SCL). Na rozdíl od I^2C u tohoto způsobu komunikace probíhá adresování pomocí takzvaných select pinů (označují se jako CS — *chip select* nebo SS — *slave select*). Implicitně je vodič CS na logické 1 a zařízení, se kterým se má komunikovat se tento pin nastaví na 0 [38]. Nevýhodou je větší množství potřebných vodičů na rozdíl od I^2C.



Obrázek 2.3: SPI rozhraní s více zařízeními[10].

UART

UART (*Universal Asynchronous Receiver/Transmitter*) není na rozdíl od SPI nebo I^2C komunikační protokol, ale hardwarový obvod, který se stará o komunikaci a jeho hlavní účel je přijímat a odesílat sériová data. Jako výhodu lze považovat to, že zařízení mezi sebou komunikují pouze pomocí dvou vodičů RX a TX a nepotřebují synchronizační signál, jelikož si je vodiče generují samy. Komunikace probíhá pouze mezi dvěma zařízeními a tím pádem není možné v rámci jednoho vodiče jednoduše přidávat více zařízení. UART je implementováno různými sériovými rozhraními, jako například známým rozhraním RS-232 nebo RS-485 a je také často využíváno v různých zařízeních jako jsou GPS moduly, Bluetooth moduly a podobně [21].

Přenos dat přes UART probíhá tak, že se propojí TX vodič prvního s RX vodičem druhého (tudíž z pohledu zařízení se propojuje vodič pro odesílání dat s vodičem pro příjem dat na druhém zařízení), analogicky poté je propojen TX vodič ze druhého zařízení s vodičem RX na zařízení prvním (vodiče jsou prohozeny). Data jsou organizována do paketů a každý paket je uvozen takzvaným start bitem, po němž následuje 5-9 bitů datového rámce a volitelně také parity bit. Paket je ukončen jedním nebo dvěma stop bity. Přenosová linka je běžně držena na vodiči jako logická 1 a příjem dat probíhá tedy na sestupné hraně. Časování přenosu probíhá tak, že každá strana musí mít nastavený stejný takzvaný baud rate, jinak si strany mezi sebou nerozumí [50].

[10]Převzato z https://commons.wikimedia.org/wiki/File:SPI_three_slaves.svg

2.3.2 Bezdrátové technologie

Senzorické jednotky a zařízení mezi sebou mohou komunikovat i bezdrátově, což umožňuje je umístit i na nedostupná místa. V následující sekci jsou popsány často využívané bezdrátové technologie v IoT systémech, jako je například Wi-Fi či ZigBee.

Wi-Fi

Technologie Wi-Fi je v současné době jedna z nejpoužívanějších a nejdostupnějších bezdrátových komunikačních technologií. Je vhodná pro větší přenosy dat na kratší vzdálenosti (v rámci blízkého okolo obydlí). První Wi-Fi zařízení se objevila na trhu někdy kolem roku 2000 a nyní téměř každé chytré zařízení je vybaveno Wi-Fi.

Existuje aktuálně 6 generací Wi-Fi. Dříve byla Wi-Fi zařízení dělena podle názvu standardu IEEE 802.11, ale nyní se nově přechází na nové značení, které je číslováno podle generace. U starších zařízení se nejčastěji vyskytuje rozhraní Wi-Fi 3, které se značí také jako standard IEEE 802.11g. Standardem v dnešní době je pak Wi-Fi 5, které má vyšší datovou rychlost a vysílá v 5 GHz frekvenčním pásmu (Wi-Fi 4 - IEEE 802.11n má též 5 GHz režim, ale s nižší rychlostí). Většina zařízení napříč různými generacemi by měla být do jisté míry mezi sebou kompatibilní (pokud jsou obě schopna pracovat ve stejném frekvenčním pásmu a je zvolen vhodný kanál) [35].

Bluetooth

Bluetooth je bezdrátová komunikační technologie, která slouží pro přenos menšího množství dat na kratší vzdálenosti. Bluetooth byl vytvořen v roce 1994 Švédskou firmou Ericsson. Název Bluetooth (česky přeloženo jako „modrý zub“) byl zvolen podle krále Harald I., Modrozuba, který s využitím diplomatických zkušeností sjednotil skandinávské kmeny. Bluetooth měla za cíl sjednotit komunikaci mezi počítači a telefony [35].

Dříve se tato technologie zaměřovala zejména na propojení mezi počítači a telefony (či PDA) nebo komunikaci telefon-telefon. V současné době se ale Bluetooth používá například i pro přenos audia (bezdrátová sluchátka, handsfree sady), ovládání různých zařízení v domě (chytré žárovky a vypínače, hodinky a trackery), nebo například i pro získávání dat ze senzorů (čidla, venkovní teploměry). Původní Bluetooth (dnes nazvané též jako Bluetooth Classic) bylo však nehospodárné na spotřebu a vyžadovalo párování a obnovování spojení. V rámci nové verze protokolu 4.0 vznikla tedy odnož Bluetooth LE (Low Energy), která je uzpůsobena pro nízkou spotřebu, a poměrně překvapivě, také na větší dosah [23].

ZigBee

ZigBee je další ze známějších bezdrátových komunikačních technologií. Je zaměřené na univerzalitu, nízkou spotřebu vzájemně operujících zařízení a do jisté míry také na nezávislost na výrobci. Ve mnoha případech je možné spojit různá zařízení různých výrobců přes ZigBee bránu a umožnit zařízením spolupráci v jedné síti. ZigBee je taky zaměřeno na vyšší pokrytí a navíc v síti může být teoreticky připojeno desítky tisíc zařízení [35].

ZigBee ve světě pracuje na frekvencích 2.4 GHz, a zároveň některá zařízení komunikují v pásmech 915 MHz v USA a 868 MHz v Evropě (v dalších regionech se frekvence taky různě liší). Toto způsobuje, že při nákupu zařízení z USA je potřeba si ověřit, zda dané zařízení využívá mezinárodní frekvenci nebo lokální. Ve většině případů však ZigBee běží na univerzální 2.4 GHz.

Komunikace využívající nelicencované pásmo 433 MHz a 868 MHz

Kromě 2.4 GHz a 5 GHz je Evropě taky velmi často využíváno pásmo buď 868 MHz nebo 433 MHz. Všechna tato frekvenční pásma jsou nelicencovaná a tudíž je může využít kdokoliv, pokud dodrží pravidla jejich užívání (například se při vysílání nesmí překračovat předepsaný maximální vysílací výkon) [51].

Tato pásma jsou vhodná pro zařízení, která nepotřebují velký datový přenos, třeba bezdrátové ovladače od aut či garáží, domácí meteostanice a podobně. Nižší frekvence se také lépe šíří do dálky [36], takže je možné při volbě vhodné antény využít většího dosahu nebo například dosahu srovnatelného s Wi-Fi sítěmi, avšak s několikanásobně nižší spotřebou.

2.3.3 Komunikační protokoly

Získaná data ze senzorů je potřeba přepravit k dalšímu zpracování. Jako nosič se dá využít například některá z bezdrátových technologií nebo Ethernet, nicméně data je potřeba na přenos připravit. K tomuto účelu se využívají komunikační protokoly.

HTTP REST API

REST (*Representational State Transfer*) je architektura rozhraní navržená pro distribuované prostředí. V zásadě využívá HTTP GET, POST, PUT a DELETE volání pro práci s obsahem a tím pádem se jedná o velmi jednoduché a univerzální řešení pro přenos dat.

Tento komunikační protokol se často používá pro přenos textových řetězců. Příjem i odeslání je implementačně jednoduché a přímočaré. Proto se často například datové struktury z mikrokontrolérů přenáší serializované pomocí různých způsobů reprezentace, například v dnešní době se používá XML nebo JSON [42].

MQTT

Pro komunikaci mezi IoT zařízeními je možné využít i protokol MQTT. Vyznačuje se snadnou implementací, nízkými nároky na hardware a velkou rozšířeností.

MQTT (*Message Queuing Telemetry Transport*), je protokol pro přenos dat mezi zařízeními. Využívá se prostředníka, který se nazývá *broker* a slouží k tomu, že shromažďuje zprávy jdoucí po síti a zařizuje, aby je dostal ten, který má. Zprávy, které jdou sítí, jsou rozdělovány do témat (topic) a zařízení připojená k brokeru data buď posílají do určitého tématu (v MQTT terminologii je toto označeno jako *publish*) nebo je klient připojen k odběru určitého tématu (*subscribe*) a broker se poté postará o to, aby zprávy z odebíraného tématu byly odeslány do zařízení. Jakýkoliv klient může být subscriber i publisher zároveň a může publikovat i odebírat do/z více témat najednou [29].

Přenáší binární data, ale i například JSON či text. Maximum, které jde přenést v jedné zprávě přes MQTT je 256 MB, ale tato velikost je více než dostačující pro běžné použití, a dokonce většina jednodušších zařízení nebo knihoven, jako například knihovna PubSubClient pro ESP8266/ESP32 tak velké zprávy nedokáže přijmout [44].

Témata v MQTT mají podobu adresářové struktury, takže se téma například může nazývat `kuchyň/teplota`. Tento tvar zprávy je poté dobrý například pro to, že si jednoduše můžeme vyžádat zprávy ze všech senzorů ve stejné místnosti (téma by v tom případě bylo `kuchyň/#`, kdy # znamená, že se odebírá libovolné podtéma). MQTT podporuje názvu témat i zpráv UTF-8, tudíž není problém s diakritikou.

MQTT podporuje i QoS, pro zajištění doručení dat a to tak, že QoS 0 je `publish once at most`, kdy zpráva je publikována pouze jednou a není zaručenou doručení. QoS 1 funguje tak, že po přijetí dat brokerem je zpráva poslána i všem zařízením v tématu, ale přijetí potvrdí broker (zajišťuje tedy jen to, že broker dostal zprávu a publikoval ji). Pokud u QoS 1 broker nepotvrdí přijetí, tak zařízení posílá zprávy brokeru znovu, dokud broker nepotvrdí přijetí. U QoS 2 potvrzuje přijetí samotný příjemce, tudíž každou zprávu musí potvrdit příjemce, odesílatel musí potvrdit odpověď na zprávu o přijetí a pak ještě musí jednou odpovědět, že si data ukládá. Tento způsob přenosu zprávy je dobrý pro zamezení výpadku jak brokeru, tak klienta[11].

2.4 Koncová zařízení pro IoT systémy

Nedílnou součástí IoT systémů jsou zařízení, senzory a akční členy, které může IoT systém využívat. IoT uzel se skládá z akčního členu nebo senzoru a mikrokontroléru.

2.4.1 Mikrokontroléry

Aby bylo možné zaznamenávat jednoduše údaje, je potřeba mít zařízení, které má nízkou spotřebu, je dobře zdokumentované a dobře se programuje. V této podsekci jsou představeny mikrokontroléry ESP8266/ESP32 pro svoji popularitu, jejich možnosti a funkce.

ESP8266

Informace v této podsekci jsou čerpány mimo jiné z mé bakalářské práce [33].

ESP8266 je poměrně známý, velmi levný a dobře dostupných Wi-Fi mikrokontrolér navržený a vyráběný firmou Espressif Systems. Přesněji se jedná o Wi-Fi čip, který v sobě obsahuje také procesor založený na RISC architektuře a je možné do něj nahrát vlastní software. Programuje se často například v LUA, MicroPythonu či C/C++ (nebo v jeho modifikované verzi, která se používá pro programování Arduina zvaná *Wiring*). Nespornou výhodou tohoto mikrokontroléru pro vývojáře je dostupnost různých programovacích jazyků a softwaru.

Podstatnou možností je taky kompatibilita se mnoha původními Arduino knihovnami a podpora jakéhokoliv kódu C/C++ ve Wiring prostředí pro Arduino MCU bez nutnosti větších úprav kódu pro použití s ESP8266. Navíc lze využít i Arduino IDE, kde se poté programuje úplně stejně jak na Arduino, jen se zvolí správný kontrolér v IDE a pro dostupnost Wi-Fi konektivity je potřeba zahrnout také knihovnu `ESP8266WiFi.h`.

ESP8266 je stavěn na napájecí napětí o velikosti 3,3 V, obsahuje jednojádrový RISC procesor, který má výchozí frekvenci 80 MHz a lze jej přetaktovat na 160 MHz. Procesor je doplněný o 32KB instrukční paměti RAM a 80KB uživatelské RAM. Má 17 GPIO konektorů, ze kterých jeden slouží jako ADC převodník a obsahuje také 1 UART (na vývojářských verzích je většinou využit USB-TTL převodníkem), jeden hardwarový SPI (flash paměť). Využití SPI nebo I^2C i I^2S je na ESP8266 možné pouze přes SW implementaci, jelikož ESP8266 neobsahuje patřičný hardwarový radič [31].

Samotný čip se nachází obvykle na SMD modulu, který obsahuje ještě další komponenty a někdy také anténu. Většina provedení SMD modulů jsou v zásadě velmi podobná. Deska obsahuje uzemněnou stínící krytku, pod ní se nachází několik komponent jako například

[11] https://www.emqx.io/blog/introduction-to-mqtt5-protocol-qos

pomocné diody, rezistory, ale také SPI flash paměť. Deska také obvykle obsahuje buď keramickou nebo tištěnou anténu na části DPS a pokud anténu neobsahuje, bývá zde umístěn IPX konektor[12]. Mezi jeden z nejznámnějších SMD modulů patří ESP-12, který obsahuje anténu ve formě cesty na tištěném spoji, ochrannou stínící krytku a LED diodu, která indikuje spuštění. Má vyvedených 9 GPIO portů + ADC, reset PIN, napájení a uzemnění.

Tento modul je pak často osazován na vývojové desky, které jsou vzhledem často podobné Arduinu a mají vyvedeno rozhraní GPIO, napájení, obsahují přídavné LED diody a podobně. Přes GPIO poté je pak možné jednoduše připojit například externí LED diody, motory, LCD displeje, senzory a jiné a některé vývojové kity jsou tak populární, že vznikají na jejich GPIO přímo tzv. shieldy, které ulehčují instalaci, protože se daná komponenta může jen vložit do konektoru jako rozšiřující modul.

Zařízení využívající ESP8266:

NodeMCU se poprvé objevil na trhu v roce 2014 a byl tím pádem jako jeden z prvních k dispozici pro komunitu. Byl velmi populární jak mezi uživateli, tak mezi zákazníky a tím pádem sloužil pro výrobce jako inspirace a referenční model pro výrobu svého vlastního návrhu nebo dokonce pro klonování[13].

Obrázek 2.4: NodeMCU v1.0 devkit[14].

Wemos je jeden z nejvíce známých a nejvíce kopírovaných[13] výrobců vývojářských kitů založených na ESP8266. Mezi jejich nejpopulárnější kity patří Wemos D1, D1 Mini nebo LOLIN D32 (ten je ale již založený na ESP32 mikrokontroléru, sekce 2.4.1)

- Wemos D1[15] má 11 digitálních GPIO pinů, které dovedou například pracovat s I^2C, SPI, signály přerušení, či například PWM nebo OneWire, jeden pin na GPIO je analogový vstup (0–3.2 V) realizovaný 10bitovým AD převodníkem. Napájení je řešeno kulatým (barrel) konektorem, a D1 akceptuje libovolné napětí v rozmezí 9–24 V.

[12] https://www.sparkfun.com/pages/RF_Conn_Guide

[13] https://hackaday.com/2017/05/15/attack-on-the-clones-a-review-of-two-common-esp8266-mini-d1-boards/

[14] Převzato z https://commons.wikimedia.org/wiki/File:NodeMCU_DEVKIT_1.0.jpg

[15] https://wiki.wemos.cc/products:d1:d1

- Jak již název napovídá, tak D1 Mini[16] je zmenšený původní Wemos D1, má stejné funkce, ale je napájen přes microUSB a mám mnohem menší rozměry, pouze 34.2 mm x 25.6 mm.

ESP32

ESP32 je nová verze Wi-Fi mikrokontroléru od firmy Espressif Systems, která po svém předchůdci (sekce 2.4.1) mnoho věcí vylepšila a obsahuje velké množství nových funkcionalit, lepší výkon i nabídku portů a technologií[17].

Zařízení kromě rychlejšího Wi-Fi modulu nabízí i podporu Bluetooth ve verzi 4.2 (a to v obou verzích, jak verze Classic, tak úsporná LE specifikace a mohou běžet najednou [18]). ESP32 je stavěno na napětí 3,3 V, obsahuje dvoujádrový RISC procesor o frekvenci 160 MHz doplněných o 520KB RAM. Má 34 GPIO konektorů, z toho 7 umí i analogový vstup (má 12bitový ADC převodník). Procesor dokáže adresovat 3 SPI rozhraní (jedno bývá obsazeno flash pamětí, která je externí, ostatní jsou volné a každé z nich může mít připojeno velké množství dalších zařízení), 2 I^2C a 2 I^2S. Dále pak umožňuje mít až 3 UART rozhraní (jedno z nich je zpravidla obsazenou u vývojových verzí USB-TTL převodníkem). Zajímavostí je také jistě to, že téměř každý GPIO pin zároveň může být použit jako dotykový (`touch`) pin, daný GPIO vstup tedy může reagovat na dotyk rukou, a podobně. [32].

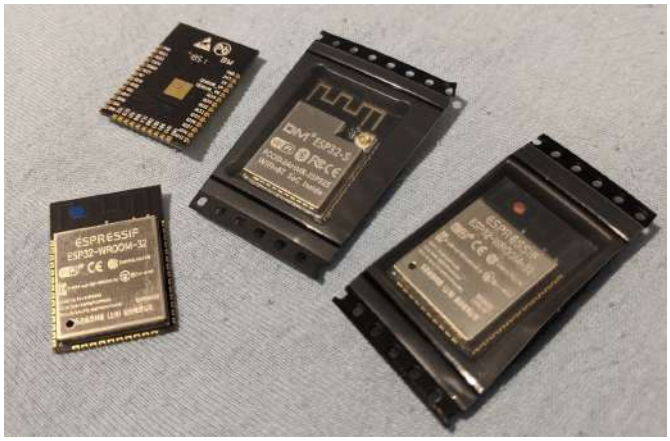
Podobně jako u ESP8266 je tento Wi-Fi mikrokontrolér osazený na pomocné desce, která obsahuje zpravidla krycí stínění, flash paměť, anténu (na tištěném spoji, keramickou, či formou slotu) a samozřejmě GPIO vodiče. Mezi nejznámější moduly pro ESP32 patří například modul WROOM-32, který obsahuje anténu formou tištěného spoje, 4 MB flash paměť a ochranné stínění. Tento modul má navíc vyvedeny téměř všechny piny, které ESP32 má (celkem 32 je jich k dispozici na modulu). Je ovšem potřeba vynechat 6 pinů (GPIO6–GPIO11), ke kterým je připojena flash paměť. Další modul, který se často vyskytuje, je ESP32-S, který má úplně stejné rozložení pinů a velikost, ale na rozdíl od WROOM-32 má přepínatelnou anténu, standardní na tištěném spoji, ale zároveň má i U.FL konektor na externí anténu. Volba antény poté probíhá pomocí spojů, které je poté potřeba propojit, aby anténa byla spojena fyzicky se zařízením[18].

Zařízení využívající ESP32:

ESP32-DevKitC je referenční vývojový kit vydaný přímo firmou Espressif. Byl uvolněn do prodeje společně s ESP32 a je osazený modulem WROOM-32. Po stranách má vyvedeny všechny GPIO, jak na modulu, ale mimo to má USB na TTL převodník, programovatelnou LED diodu a dvě tlačítka, která slouží pro přehrání firmware a restart zařízení.

LOLIN D32 Pro je navrženo firmou Wemos a má na rozdíl od referenčního vývojového kitu speciální konektory, například konektor na přídavnou baterii, slot na microSD karty, I^2C konektor (GH 4-pin) a TFT port (konektor pro displej).

[16] https://wiki.wemos.cc/products:d1:d1_mini
[17] https://makeradvisor.com/esp32-vs-esp8266/
[18] https://www.14core.com/esp32-s-cam-in-face-detection-and-recognition-with-esp-who-library/



Obrázek 2.5: Moduly ESP32-WROOM-32 a ESP32-S.

2.4.2 Komerční mikrokontrolérové platformy

Zařízení značky Sonoff

Jedno ze zajímavých zařízení z hlediska ceny, dobré možnosti integrace a relativně velké míry přizpůsobení jsou zařízení značky Sonoff od firmy ITEAD. Jedná se o například inteligentní relé, chytré vypínače, zařízení na ovládání dveří či zámků, bezdrátové senzory teploty či pohybu [30], přičemž jejich cena se pohybuje kolem 100 korun.

Tato zařízení jsou založena z větší části na mikrokontrolérech ESP8266 se speciálním firmwarem, který spolupracuje s eWeLink aplikací, která je dostupná jak pro mobily se systémem Android, tak pro iOS. Aplikace dovede komunikovat se zařízením přes internet a podporuje také podmíněné akce a časovače a historii stavu senzorů či vypínače. Aplikace také umožňuje integraci do Apple HomeKit, Google Home i do Amazon Alexa.

Firmware však jde velmi jednoduše u téměř každého zařízení přehrát za vlastní. Na DPS každého zařízení jsou přichystány UART vodiče pro připojení vlastního programátoru (například USB/TTL adaptér nebo jiné zařízení s UART) a je poté možné nahrát svůj vlastní kód. Některá zařízení v určitých verzích firmwaru lze flashovat i OTA, stačí k tomu pouze nástroj (buď oficiální od ITEAD, ten ale umí pouze pár druhů zařízení[19] nebo komunitní cestou[20]). S instalací vlastního firmware napojení na aplikaci eWeLink přestane fungovat.

U zařízení Sonoff je ve velké míře oblíbený například Tasmota firmware. Je to alternativní firmware, který umí poskytnout mnohem více funkcí než standardní firmware s aplikací eWeLink. Lze například přidat do zařízení různé senzory, tlačítka, monitorovat více různých dat anebo je přenášet několika různými způsoby[21]. Sonoff je také velmi dobře podporován v ESPHome (sekce 2.2.4), konkrétně je obsažena nativní podpora pro všechny hlavní druhy zařízení (Sonoff S20, Sonoff Basic/Dual, Sonoff Touch, Sonoff 4CH), na které je veškerá dokumentace, včetně návodu na rozebrání a nahrání softwaru a také ukázky konfigurace. Dalších asi 20 zařízení je velmi podobných, takže k nim ESPHome poskytuje pouze seznam GPIO a jejich činnost (včetně odkazu na repozitář Tasmota, kde jsou nejnovější zařízení zdokumentována) a odkazuje na existující konfigurace u podobných zařízení.

[19] https://github.com/itead/Sonoff_Devices_DIY_Tools
[20] https://github.com/mirko/SonOTA
[21] https://www.iotwithus.com/what-is-tasmota/

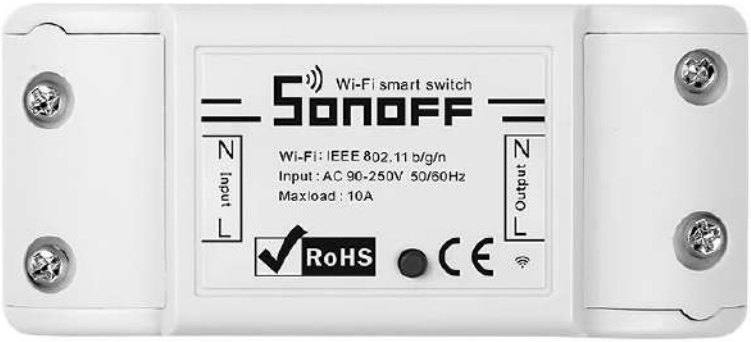
Sonoff Basic je nejlevnější ze všech prodávaných zařízení, jeho cena se pohybuje kolem 4 USD a lze jej využít na celou řadu věcí. Toto zařízení totiž je, zjednodušeně řečeno, Wi-Fi relé, které jde ovládat z mobilního telefonu na dálku (i přes internet). Jako vstupní napětí akceptuje 90-250 V AC, které z části slouží k napájení modulu a z části je pouštěno dál zařízením přes relé směrem k cílovému zařízení. Je tedy možné si vyrobit vlastní napájecí kabel k určitému zařízení pouhým rozpůlením stávajícího kabelu a přidáním tohoto modulu. Na zařízení je i vypínač, který slouží k uvedení zařízení do režimu párování, ale také jako manuální vypínač.

Sonoff S20 jedna z více variant zásuvek, je funkčně podobná Sonoff Basic, ale má standardní AC slot a tím pádem funguje okamžitě, bez potřeby připojovat kabeláž. Cena je ale vyšší než u předchozího modelu.

Sonoff Touch je dotykový vypínač, který se instaluje do zdi na místo standardního vypínače, jsou varianty jednokanálové i vícekanálové. V případě, že není zařízení spárováno nebo není připojeno k síti, lze světlo ovládat i dotykovými tlačítky na vypínači. Toto zařízení vyžaduje, jako všechna ostatní i neutral (N) vodič, který ve starších elektroinstalacích není doveden do vypínačů.

Sonoff Mini je zařízení, které se nainstaluje za stávající vypínač a pouze se připojí vodiče ze současného vypínače na vstupy zařízení. Tím pádem nenaruší vzhled původního vypínače.

Sonoff 4CH varianta na Sonoff Basic, avšak obsahuje 4 relé misto jednoho.



Obrázek 2.6: Sonoff Basic[22].

Zařízení od výrobce Tuya

Tuya je společnost, která má vlastní cloudové IoT řešení, včetně mnoha různých druhů zařízení a nabízí tyto IoT služby a výrobky jiným firmám. Společnost se například chválí

[22]Převzato z https://cdn-media.itead.cc/media/catalog/product/1/_/1_7.jpg

spoluprací s více než 180000 klienty napříč celým světem, kteří nyní údajně nabízejí přes 90000 variant jejich produktů[23]. Pyšní se i s poměrně známými firmami, jako je například Lidl, TCL, Hama, Siemens, Lenovo a Asus. Vyrábí vlastní Wi-Fi žárovky a světla, zásuvky, různé vypínače a časovače, topidla a vytápěcí hlavice, senzory kouře a podobně. Platforma využívá buď vlastní cloudové aplikace Tuya, která je dostupná jak pro Android, tak pro systém iOS, nebo pod různými aplikacemi s požadavky podle přání klienta (známá je například aplikace SmartLife). Co tyto aplikace spojuje, je vzájemná kompatibilita a až nápadně podobný vzhled. Jejich aplikace lze napojit také na jiné systémy inteligentní domácnosti, jako je Google Home a Amazon Alexa. Některá jejich zařízení tedy jsou podobná zařízením Sonoff (sekce 2.4.2), ale nabízí na rozdíl od firmy ITEAD mnoho druhů Wi-Fi žárovek včetně těch s regulací svícení a RGB.

Zajímavé na této společnosti je rovněž to, že často ve svých smart home produktech založených na technologii Wi-Fi používá mikrokontroléry od firmy Espressif. Zařízení (např. žárovka) se tedy dá relativně jednoduše přeprogramovat na vlastní firmware. Jelikož se ale žárovky docela špatně rozebírají, aniž by došlo k poškození minimálně jejího krytu, vyvstává myšlenka, jak takovou žárovku naprogramovat bez zásahu. Tuto otázku si pokládal i Michael Steigerwald, zakladatel německé IT security firmy VTRUST. Jelikož firma Tuya tvrdí, že její řešení má „Military-grade AES and HTTPS WAN/LAN encryption“, rozhodl se, že se zaměří na její zabezpečení. Zjistil, že přes několik málo triků, jako například podvržení DNS a vynucení OTA update z vlastního serveru, lze do libovolného zařízení od firmy Tuya nahrát vlastní software. Společně s ct magazínem[24] proto sestavili vlastní nástroj, který umožňuje komukoliv přehrát firmware v Tuya zařízení a osvobodit tak svoji žárovku od nutnosti používat jejich cloud a upravit si ji tím pádem podle svých představ. Společnost se snaží bránit proti tomuto vydáváním nových verzí firmware i hardwarových revizí, které neobsahují již ESP moduly, ale například Realtek[25] (které nejsou populární v komunitě). Kompatibilních zařízení je ale celá řada a hodně z nich je i dobře zdokumentovaných[26]. Dané zařízení se pak dá jednoduše zprovoznit v ESPHome nebo Tasmotě a napojit na vlastní cloud.

Jelikož zařízení od této společnosti je plný trh a jsou velmi levné (například RGBW žárovky s regulací jasu se pod značkou FCMILA někdy pouze za 5 USD za kus). Přitom srovnatelná žárovka s tolika možnostmi (napojení na Google Home, ovládání přes internet a mobil, RGB osvětlení, regulace jasu, ...) stojí alespoň trojnásobek.

2.4.3 Vybrané prvky pro IoT home systém

Pro realizaci IoT systému doma lze využít mnoha prvků, které lze dobře integrovat. V následující sekci byly vybrány takové prvky, které se v domácnosti buď často vyskytují (žaluzie, domovní zvonek, vysavač), nebo jsou cenově dostupné a mají zajímavé funkce.

Meteostanice

Součástí systému je i meteostanice, kterou využívá venkovní osvětlení pro svoji činnost. Kit meteostanice SEN-08942 sestává z anemometru (senzor rychlosti a směru větru) a senzoru srážek. Je na sloupovku, který má na sobě přidělány jednotlivé komponenty a instaluje se na střechu domu. Jednotlivé senzory obsahují konektor RJ45 (6P4C) a připojují se ním

[23] https://en.tuya.com/company
[24] https://www.ct.nl/magazine/
[25] https://github.com/arendst/Tasmota/issues/6434
[26] https://github.com/ct-Open-Source/tuya-convert/wiki

k zařízení, které signály ze senzorů umí číst. Kit v mém případě obsahoval ještě krytku, kterou lze zakrýt případnou elektroniku před povětrnostními podmínkami, dále pak držák na tuto krytku, ale již ne základnu, do které se elektronika může upevnit.

Senzor Xiaomi LYWSD03MMC

K monitorování vnitřní teploty a vlhkosti se využívá senzor Xiaomi LYWSD03MMC. Je vybaven LCD displejem, který zobrazuje teplotu, vlhkost a grafický ukazatel, který znázorňuje, zda tato teplota a vlhkost jsou v přijatelné hodnotě. Senzor je dále vybaven technologií Bluetooth Low Energy (sekce 2.3.2), která slouží k předávání dat mobilnímu telefonu, případně IoT bráně.

Zvonek Alcad KAS-41021

Zvonek od výrobce Alcad obsahuje domovní telefon, samotný zvonek a elektromagnetický zámek na branku. Spojení mezi zvonkem a telefonem je realizováno pomocí analogového 4+n zapojení (4 vodiče jsou společné každému zvonku a další samostatný vodič pro každé zvonkové tlačítko). Přenáší se zvuk z i do telefonu, signál pro elektrický zámek, jeden vodič je vyhrazen jako společná svorka (zem) a poslední je určen pro zvonění. Zvonění je přenášeno přímo zvukem a zařízení nesmí zaznamenat jiný odběr elektrického proudu, než je stanoveno ve specifikaci (nelze tedy například detekovat změny napětí při zvonění a je nutné přímo detekovat stisk tlačítka a ten dále případně spínat pomocí relé). Venkovní část zvonku sestává z tlačítka pro zazvonění a hlasové jednotky[27], která obsahuje vstupy a výstupy z domácího telefonu, 12 V AC vstup pro napájení, který rovněž napájí i domácí telefon a také výstup pro podsvícení zvonku. Dále jednotka také obsahuje 12 V AC výstup, který se sepne při stisknutí tlačítka pro otevření dveří na telefonu, toto napětí odblokuje elektromagnetický zámek, který lze poté již otevřít.

Vysavač iRobot Roomba 580

Vysavač iRobot Roomba 580 je plně automatický robotický vysavač, který umožňuje vysávat jak hladké plochy, tak i koberce. Je vybaven velkým množstvím senzorů (senzory vzdálenosti, nárazu, přiblížení, optického senzoru ujeté vzdálenosti a další), které umožňují vysavači se volně pohybovat po domě či bytě, aniž by například sjel ze stolu nebo se někde zasekl. Lze jej nabíjet i z nabíjecí stanice, ze které umí sám vyjet a při dokončení úklidu (či vybití baterie) ji zase vyhledat a zadokovat se.

Vysavači lze vymezit prostor pomocí takzvaných virtuálních zdí, tyto zdi jsou moduly, které se umístí do místnosti a nasměrují se výstupem tím směrem, kudy má tato virtuální zeď vést. Vysavač poté dostane informaci, že se jedná o skutečnou zeď, a tím pádem hranici stanovenou tímto modulem nepřekročí. Tento modul lze taky přepnout do takzvaného beacon módu (majáku), kdy při umístění ke dveřím do jiných místností lze vysavači říct, jak má postupovat při vysávání. Zařízení poté vysaje první místnost, poté se vydá do další a v ní zase pokračuje ve své činnosti. Toto umožní omezit chaotické přejíždění vysavače mezi místnostmi. Vysavač lze také ovládat bezdrátovým ovladačem, který slouží jak k ručnímu přímému ovládání, tak ke vzdálenému iniciování vysávání či příkazu k zadokování se. Ovladačem lze taky nastavit naplánování vysávání.

Vysavač obsahuje konektor typu MiniDIN, který slouží k připojení k počítači nebo k mikrokontroléru. Komunikační protokol je vlastní navrhnutý firmou Roomba a jmenuje

[27] http://www.altronik.cz/documents/kas41021/cs/n%C3%A1vod%20k%20obsluze.pdf

se OI (Open Interface). Tento protokol je založený na sériové UART komunikaci využívající 5 voltovou logiku. MiniDIN konektor tedy obsahuje RX a TX piny a také neregulovaný výstup z baterie (12–15 V), kterým lze napájet například připojený mikrokontrolér. Pomocí OI se dají číst data ze všech senzorů, které vysavač obsahuje, dále pak lze monitorovat stav baterie, všechny chybové stavy vysavače a také vysavač plně ovládat. To znamená, že lze ovládat vysavač jak ručně (všechny motory, které obsahuje), tak spustit jeden určený režim vysávání, který nabízí (Clean nebo Spot), nebo spustit režim Dock, který vyhledá dokovací stanici a zaparkuje do ní pro nabití baterie.

Open Interface je komunikační protokol založený na posílání číselných kódů přes sériové rozhraní[28]. Před zahájením jakékoliv komunikace je potřeba zaslat zprávu `Start`, která je v rozhraní označena hodnotou operandu 128. Po této zprávě lze vysavači zaslat již libovolný příkaz. Vysavač může běžet ve třech módech, `Passive`, `Safe` a `Full`. V pasivním režimu vysavač naslouchá pouze příkazům, které se dají zadat i přímo na zařízení (start a pozastavení vysávání, návrat do docku, ...) a vyčítat všechny senzory vysavače a stav baterie. Tento stav je výchozí ihned po zahájení komunikace, ale lze na něj přepnou použitím libovolného z `Cleaning` příkazů. V nouzovém režimu vysavače lze přistupovat ke všemu, k čemu v pasivním režimu a k tomu navíc ke všem aktuátorům vysavače. Lze například ovládat všechny motory, vysávání, LED diody nebo třeba přehrávat tóny. Motory taky lze ovládat pomocí PWM, tudíž, lze regulovat jejich rychlost. V nouzovém režimu však zařízení zůstávají v provozu některé základní systémy, jako detekce schodů nebo například přepadnutí kola přes přepážku, a tím pádem je zabráněno pádu vysavače z výšky. Na tento režim lze přepnou zasláním operačního kódu 131. Plný režim (`Full mode`) má všechny funkce předešlého režimu, avšak bez bezpečnostních prvků, takže zařízení ignoruje veškeré překážky a hlášení.

Mezi další důležité příkaze patří příkaz pro čtení senzorů. Pro přečtení určitého senzoru je potřeba prvně odeslat operační kód 142 následovaný tzv. `paket ID`, které označuje senzor, který se má vyčítat (například kód 22 označuje senzor, který monitoruje napětí baterie). Příkaz pro dotazování se stavu senzorů vrací poté hodnotu dotázaného senzoru a to tak, že posílá hodnoty po bajtech. Některé hodnoty se vejdou do jednoho bajtu (například náraz do překážky a jiné bool hodnoty) a jiné jsou posílány po více bajtech za sebou a to tak, že první dorazí nejlevější bajt. OI umí i dávkové dotazování senzorů (operační kód 149), kdy lze s jednou skupinou příkazů dotázat na více senzorů najednou. Zařízení poté vrátí tyto hodnoty v datovém toku po bajtech a je potřeba si jednotlivé bajty uspořádat a přiřadit k jednotlivým senzorům. Obdobně lze také využít streamovacího příkazu (kód 149), který funguje úplně stejně jak dávkové dotazování, ale na rozdíl od něj funguje neustále v reálném čase a dotázané senzory zasílají data každých 15 ms).

[28] https://cdn-shop.adafruit.com/datasheets/create_2_Open_Interface_Spec.pdf

paket ID	Název senzoru	Počet bajtů
8	Zeď	1 bajt
13	Virtuální zeď	1 bajt
14	Nadproud	1 bajt
15	Detekce nadměrného prachu	1 bajt
18	Tlačítka	1 bajt
19	Vzdálenost od překážky	2 bajty
20	Úhel	2 bajty
21	Stav nabíjení	1 bajt
22	Napětí baterie	2 bajty
23	Proudový odběr	2 bajty
24	Teplota	1 bajt
25	Nabití baterie	2 bajty
26	Kapacita baterie	2 bajty
39	Rychlost	2 bajty

Předokenní rolety Somfy RTS

Předokenní rolety Somfy RTS jsou bezdrátové žaluzie, které svým provedením dokáží nejen odstínit dům od světla, ale také ochránit okna, zamezit únikům tepla nebo například v létě zamezit přehřívání domu. Tyto žaluzie obsahují netriviální komunikační protokol a pro jeho ovládání je potřeba mít přidružený speciální ovladač, kterým se poté žaluzie ovládají.

Komunikační protokol Somfy RTS je zabezpečený, jednosměrný komunikační protokol, jehož implementace je podobná jako v technice bezdrátového odemykání automobilů. Protokol využívá 433,42 MHz, přičemž je využito ASK/OOK modulace a kódování Manchester, kde náběžná hraje je logická 1 a sestupná logická 0. Komunikace sestává ze synchronizačních a probouzecích sekvencí, které umožňují žaluziím rozpoznat, že se jedná o zprávu pro žaluzie a je následována daty. Délka datového rámce je 56 bitů a jeden bit je přenášen v periodě 1208 ns. Na začátku zprávy se nachází probouzecí sekvence, která je po dobu 9415 μs na náběžné hraně a po dobu 89565 μs na hraně sestupné. Tato sekvence však není povinná a žaluzie fungují i bez ní.

Samotná 56bitová datová zpráva sestává z klíče, kontrolního součtu, rolling kódu a adresy ovladače. Na prvních 8 bitech je identifikační klíč protokolu, který vždy začíná hodnotou 0xA na prvních 4 bitech a je doprovozena 4 bitama libovolné hodnoty. V následujících 4 bitech je kód tlačítka ovladače (tabulka 2.4.3), další 4 bity obsahují kontrolní součet. Na 16 bitech je uložen takzvaný rolling kód (neznamenkový big-endian) a na posledních 24 bitech je uložena adresa ovladače (little-endian). Rolling kód se inkrementuje každým stiskem ovladače a zároveň po přijetí signálu žaluzií se tato hodnota porovnává s hodnotou v paměti žaluzie. Pokud hodnota rolling kódu je stejná nebo nižší, než je v paměti pohonné jednotky, tak žaluzie rozkaz nevykoná a čeká dále na korektní zprávu.

key	ctrl | cks	Rolling Code	Address(A0|A1|A3)

Obrázek 2.7: Struktura dat u protokolu Somfy RTS [46].

K žaluziím lze libovolně přidávat a odebírat ovladače, kdy jedna žaluzie může mít přiřazených více různých ovladačů a jeden ovladač může být spárován s více žaluziemi. Přiřadit k žaluzii nový ovladač lze buď resetem zařízení do továrního nastavení (odpojení elektřiny

na 3s, obnovení přívodu elektřiny na 8s a odpojení elektřiny od zdroje na 3s) nebo pomocí již spárovaného ovladače. Na ovladači, který je již spárován se stiskne tlačítko `PROG`. Pokud žaluzie zaznamená tento příkaz, posune se nahoru a dolů o pár centimetrů. Na novém ovladači se taky stiskne tlačítko `PROG` a pokud se vše povedlo, žaluzie se zase posune nahoru a dolů. Poté je možné již žaluzii ovládat. Odebrání ovladače se buď realizuje resetem do továrního nastavení (popsáno výše) nebo obdobně jak párování. Na ovladači, který si přejeme ponechat, stiskneme tlačítko `PROG` a poté na ovladači, který má být odebrán, uděláme to samé. Žaluzie se zase pohnou nahoru a dolů a druhý ovladač přestane žaluzie ovládat.

Kód	Tlačítko	Popis
0x1	My	Zastaví žaluzii nebo ji posune do předvolené pozice
0x2	Nahoru	Pohyb žaluzie nahoru
0x3	My + Nahoru	Nastavení maximální horní pozice motoru (zvednutí)
0x4	Dolů	Pohyb žaluzie dolů
0x5	My + Dolů	Nastavení maximální spodní pozice motoru
0x6	Nahoru + Dolů	Přejde do nastavení limitů motorů (prvotní nastavení)
0x7	—	—
0x8	Prog	Přepíná žaluzie do režimu (od)registrování ovladačů
0x9	Sun + Flag	Povoluje sluneční detektor (jen ovladač Telis Soliris RC)
0xA	Flag	Zakazuje sluneční detektor

2.5 Brána

Brána v IoT systému je zařízení, slouží k propojení senzorických modulů s internetem (sekce 2.1). Takové systémy se neobejdou bez podpůrného hardwaru a softwaru. V následující sekci je tedy popsán hardware využitelný pro IoT brány a software.

2.5.1 Odroid HC1

Odroid HC1 (home cloud one) je SBC (*single board computer* – jednodeskový počítač), který je vhodný pro využití jako domácí NAS. Obsahuje 8-jádrový procesor Samsung Exynos5422, 2 GB RAM, SATA-3 konektor pro připojení pevného disku, gigabitový ethernet port a slot na microSD kartu[29]. Celý počítač je umístěn v kovovém rámečku pro 2.5" disky, který slouží zároveň jako velký pasivní chladič pro procesor a také jako držák, který umožňuje skládat na sebe více těchto zařízení. Zařízení podporuje linuxové distribuce Debian, Ubuntu, Arch, Kali Linux nebo například NASový OpenMediaVault (sekce 2.5.4). Jelikož tento model neobsahuje video výstup ani GPU, takže všechny distribuce jsou pouze v lite/headless sestaveních. Mimo tuto verzi existuje jestě verze HC2, která má téměř stejné parametry, ale má rámeček pro 3.5" disk a je napájena 12 V místo 5 V, které využívá HC1.

2.5.2 Raspberry Pi

Raspberry Pi je další jednodeskový počítač založený na ARM platformě. Je velmi populární mezi komunitou pro svoji cenu a vzájemnou HW i SW kompatibilitu napříč generacemi. Základem je (podle verze) 1–4 jádrový procesor, který obsahuje 256–1024 MB RAM. První až třetí generace obsahuje GPU VideoCore IV, ke kterému je poskytnuna plná dokumentace, je open source a podporuje OpenGL ES 2.0, 1080p akceleraci H.264 a MPEG-4 videa [48].

[29] https://www.hardkernel.com/shop/odroid-hc1-home-cloud-one/

U aktuální čtvrté generace Raspberry Pi se nachází novější GPU VideoCore VI, který oproti svému předchůdci přidává podporu OpenGL ES 3.0, dále 4K rozlišení a to jak u HW akcelerace přehrávání a kódování 4K 60fps videjí, tak i 4K rozlišení displeje. Nově je taky podporováno připojení dvou displejů současně. GPU podporuje jak HW akceleraci videa z předchozí generace (H.264 a MPEG-4), tak i nový formát H.265 HEVC, který se využívá například u nového DVB-T2 vysílaní [49].

Podobně jako je u ESP32 nebo Arduina, tak Raspberry Pi obsahuje taktéž GPIO rozhraní. Mimo to toto zařízení disponuje i DSI (display serial interface) rozhraním, které slouží k připojení displeje a CSI (camera serial interface) pro připojení Raspberry Pi kamery. Kromě toho RPi obsahuje HDMI 1.3 (první až třetí generace, v edici zero miniHDMI 1.3 a ve čtvrté generaci dvě microHDMI 2.0). Všechny generace obsahují Ethernet port a to o rychlosti 100 Mbit/s (1–3) a 1000 Mbit/s (RPi 4). Obdobně je na tom USB, kdy zase starší generace obsahují obvykle 4x USB 2.0 a čtvrtá generace 2x USB 3.0 a 2x USB 2.0. Raspberry Pi také obsahují výstup na 3.5 mm jack (audio i kompozitní výstup) a k napájení slouží microUSB (USB-C u RPi 4) konektor. Systém může běžet jak s SD karty, tak i z USB (1–2 generace musí mít připojenou SD kartu se zavaděčem, ale u nových generací může systém běžet celý přímo z USB).

Existuje několik různých provedení Raspberry Pi. Dříve byly modely A a B tohoto počítače. Model A byla levnější verze, s menším počkem portů a případně s chybějícím ethernetem. Tyto modely se již nevyrábí a Raspberry od druhé generace je nyní pouze v provedení B. Často před uvedením nové generace je vydána revize (B+), která obsahuje často něco z novější generace (vyšší velikost operační paměti, či například rychlejší ethernet port), avšak HW současti jako CPU a GPU jsou vždy stejné jako u modelu B.

Dále existuje Raspberry Pi v provedení Zero a Zero W. Toto provedení je mnohem menší než standardní modely RPi a je také velmi levné. Raspberry Pi Zero stojí kolem 5 USD a obsahuje jednojádrový CPU na frekvenci 1 GHz (taktovaný procesor z první generace Raspberry Pi), microUSB OTG, 512 MB RAM a microSDHC slot. Základní Zero neobsahuje Wi-Fi modul, avšak varianta Zero W, která stojí kolem 10 USD tento modul již obsahuje.

Další verzí RPi je takzvaný `Compute Module`. Toto provedení je určeno pro vývojáře, kteří si chtějí navrhnout vlastní DPS pro své účely. Je v provedení SODIMM DDR2 modulu, avšak není s tímto modulem v žádném případě elektricky ani HW kompatibilní. Tyto moduly sledují aktuální generaci Raspberry Pi a vždy, nějaký čas po vydání nové generace se dočká i tento modul svého nástupce. Velkou výhodou je, že se dá novější modul vyměnit za starý a periferie zůstanou kompatibilní bez potřeby měnit návrh DPS. Tyto moduly na rozdíl od standardního Raspberry Pi neobsahují microSDHC slot, ale eMMC paměť.

Generace

Raspberry Pi aktuálně má 4 generace, mezi každou generací následovala několik let pauza a každá generace přináší nějaké znatelné vylepšení. Raspberry Pi 1 obsahuje 700 MHz jednojádrový BCM2835 (ARM1176JZF-S, ARM Cortex-A6) procesor doplněný o 256 MB operační paměti. Obsahuje dvě USB (model B, 1x USB model A) a Ethernet port (model A jej nemá). Druhá verze tohoto počítače (RPi 2) přináší vylepšení výkonu. Obsahuje totiž 900 MHz čtyřjádrový procesor BCM2836 (ARM Cortex-A7) a 1 GB RAM. Obsahuje čtyři USB 2.0 porty i 100 Mbit/s Ethernet. Ve třetí verzi se poprvé objevuje 64bitový procesor, konkrétně ARM Cortex-A53, který je taktován na 1,2 GHz. Stejně jako u druhé generace toto zařízení obsahuje 1 GB RAM a má uplně identické rozvržení portů, jako

má i druhá generace. Lze tedy bez problémů využívat obaly a krabičky z druhé generace. Dalším podstatným vylepšením je přítomnost Wi-Fi a Bluetooth modulu přímo na zařízení. Aktuální generace Raspberry Pi má 64bitový procesor BCM2711, ke kterému je dodáváno buď 1 GB, 2 GB, 4 GB nebo i 8 GB RAM. Nové verze se dočkal také grafický čip, VideoCore VI, který nahrazuje čip VideoCore IV, co se nachází v první až čtvrté generaci. Jak již bylo zmíněno úvodem, toto nové GPU obsahuje podporu nově i pro kodek HEVC a dva 4K monitory. Nově je přídáno taky USB 3.0 a nabíjení je řešeno pomocí USB-C. Nově je taky zvýšena rychlost Ethernetu z 100 Mbit/s na 1 Gbit/s.

Rozšiřující moduly

Raspberry Pi je možné, podobně jako u Arduina nebo ESP32 rozšířit různými rozšiřujícími moduly (shieldy). Tyto shieldy například přidávají možnost pro podporu baterie, lepší audio čip, nebo například různá relé nebo senzory. Tato rozšíření možnosti vyplývají z toho, že RPi obsahuje univerzální GPIO rozhraní, které podporuje různé komunikační protokoly (UART, I^2C, SPI, PWM nebo I^2S) a tudíž lze použít i různé moduly z jiných zařízení.

Počítač obsahuje také rozhraní CSI a DSI, které mají svůj konektor na desce. CSI kamera ve verzi v2 má Sony IMX219 senzor a rozlišení 8 megapixelů (v1 má 5 megapixelů). DSI není příliš rozšířené, avšak slouží k připojení různých displejů pomocí speciálního sériového rozhraní.

2.5.3 Router Turris

Jako další možnou bránu lze použít například router Turris. Tento router je vyráběn společností CZ.NIC a je vybaven na router relativně rychlým hardwarem, otevřeným, pravidelně udržovaným a dobře zabezpečeným Turris OS, který je založen na OpenWRT. Routery z tohoto projektu mohou sloužit k anonymnímu sběru dat (v případě, že si tuto možnost uživatel povolí), kdy se tato data poté využívají na detekci hrozeb na českém internetu [22]. Jelikož tyto routery obsahují OS založený na Linuxu, tak lze na nich provozovat velké množství aplikací a služeb, které by normálně vyžadovaly serverový hardware. Na Turrisu se například dají nativně provozovat LXC kontejnery (sekce 2.5.5), které lze vytvářet, mazat a spravovat přes webové rozhraní.

Existuje několik verzí tohoto zařízení. Úplně první verzí byl původní modrý router Turris 1.0, který vybraní šťastlivci dostávali za jednu korunu výměnou za anonymní monitoring ze strany CZ.NICu v rámci `Projektu Turris`. Tato verze obsahuje dvoujádrový 1200 MHz procesor Freescale P2020, který je založen na architektuře PowerPC. Obsahuje 2 GB RAM, 16 MB NOR flash a 256 MB NAND flash. Router obsahuje několik USB 2.0 konektorů, slot na MicroSD kartu a dva miniPCIe sloty. Samozřejmostí je 5 gigabitových LAN konektorů a jeden gigabitový WAN konektor. V základu byl obsažen Wi-Fi modul 802.11a/b/g/n s 3x3 MIMO anténami. Vylepšená verze Turris 1.1 přidává oproti původní verzi pouze USB 3.0 konektory.

Nástupcem modrého routeru je Turris Omnia, tato verze na rozdíl od předchůdce obsahuje dvoujádrový ARMv7 procesor o frekvenci 1.6 GHz, který je doplněn v aktuální HW revizi 2 GB RAM a 8 GB flash paměti. V routeru se nachází 3 miniPCIe sloty (slot nejvíce vpravo lze využít i pro mSATA disky), z nichž dva jsou obsazeny Wi-Fi moduly. Jeden z nich je 3x3 MIMO 802.11ac, který slouží zejména pro 5 GHz rychlou Wi-Fi a druhý, 2x2 MIMO 802.11bgn, slouží pro 2.4 GHz pásmo. Router obsahuje SIM slot pro volitený LTE modul, USB 3.0, 5 gigabitových LAN konektorů, jeden gigabitový WAN konektor a

jeden SFP konektor pro optické připojení. Díky využití ARM procesoru je dostupné větší množství programů a aplikací než u předchůdce a celá řada programů a ovladačů má své balíčky přímo v Linuxu bez nutnosti si je kompilovat.

Nejnovějším příruskem do rodiny těchto českých routerů je Turris Mox. Jedná se o modulární router s možnostmi různorodého rozšíření. Uživatel si může pořídit například pouze modul `MOX A`, který obsahuje samotný CPU (Marvell Armada 3720), RAM, USB 3.0, GPIO, microSD slot a gigabitový WAN/LAN konektor. Mox v tomto případě tedy slouží jako linuxový server bez funkcionalit navíc. Pokud chceme Turrisu přidat další gigabitové LAN porty, lze využít modulu `MOX C`, který obsahuje 4 kusy těchto konektorů. Moduly lze navíc skládat za sebe a tudíž není problém například připojit modul `MOX B`, který obsahuje miniPCIe slot a slot pro vložení SIM karty. Do tohoto modulu lze poté připojit například Wi-Fi modul a z Moxu lze rázem udělat plnohodnotný Wi-Fi router.

2.5.4 OpenMediaVault

OpenMediaVault je NAS řešení založené na distribuci Debian Linux. Obsahuje služby jako SSD, FTP, SMB/CIFS, DAAP a podobně. Má modulární design a je možné do něj doinstalovávat různé pluginy (OpenVPN, Docker kontejnery, apod.). Tato služba je primárně zaměřená na domácí použití, případně v malých firmách a umožňuje velice jednoduchou konfiguraci v uživatelském rozhraní. Je vhodná jako OS do domácích NAS boxů a serverů [1].

2.5.5 Linuxové konteinery

Linuxový kontejner je soubor jednoho nebo více procesů, které jsou izolovány od zbytku systému. Všechny potřebné soubory k běhu se nachází přímo v kontejneru, díky čemuž jsou linuxové kontejnery přenositelné. Díky tomu, že narozdíl od plnohodnotné virtualizace neobsahuje kontejner celý OS, tak tyto kontejnery nezabírají tolik místa. Slouží k uzavřenému spouštění procesů bez přístupu k hostujícímu OS. V systému tedy může běžet i několik instancí různých linuxových systémů a všechny využívájí linuxového jádra hostujícího OS, nejedná se tedy uplně o virtualizaci [17].

Mezi často využívané platformy pro linuxové kontejnery patří LXC a Docker. LXC je zkratkou pro Linux Containers a umožňuje provozovat vlastní virtuální prostředí s linuxovou distribucí, avšak nejedná se o plnohodnotnou virtualizaci. Tím pádem je režie nižší než v případě použití virtuálních strojů. Docker byl dříve nadstavba nad LXC (nyní již jede na vlastní mezivrstvě `libcontainer`) a je srovnatelný s LXC. Jako rozdíl s LXC lze považovat hlavně to, že u LXC se jednotlivé kontejnery konfigurují často pomocí linuxových příkazů v shellu a konfigurují se jednotlivé komponenty, avšak u Dockeru se využívá hotových obrazů (Docker images) a skriptů, které umožňují jednodužší konfiguraci.

Kapitola 3

Návrh řešení

V následující kapitole je popsán návrh vzájemného propojení zařízení a jakým způsobem je vše automatizováno. K propojení různých senzorů a akčních členů lze využít zejména mikrokontroléru ESP32 a ZigBee mikrokontroleru E18-MS1-IPX. Tato zařízení jsou poté připojena k centrální řídicí jednotce Odroid HC1 (sekce 2.5.1), na kterém běží OpenMediaVault (sekce 2.5.4). V něm se nachází několik Docker kontejnerů, z nichž na jednom běží Home Assistant, na dalším MQTT broker a na jednom se nachází Assistant Relay (sekce 3.4.3).

3.1 Koncová zařízení

IoT systém sestává z různých částí, které jsou mezi sebou propojeny. V této sekci je tedy uvedeno, jaké části to jsou a co je k čemu připojeno.

3.1.1 ESP32 modul

Pro všechny jednotky v chytré domácnosti mimo světel je využito mikrokontroléru ESP32. Mimo to je využito v případě DPS, které slouží jako multifunkční brána, i mikrokontroléru E18-MS1-IPX, který obsahuje ZigBee. Všechny tyto mikrokontroléry jsou vybrány tak, aby měly SMD package a tím pádem nezabířaly příliš místa. U ESP32 je využito dvou druhů SMD variant mikrokontroléru a to ESP-WROOM-32 a ESP32-S (sekce 2.4.1). Svým rozložením pinů jsou totožné, avšak ESP32-S obsahuje slot pro připojení externí IPX antény (kvůli lepšímu dosahu BLE).

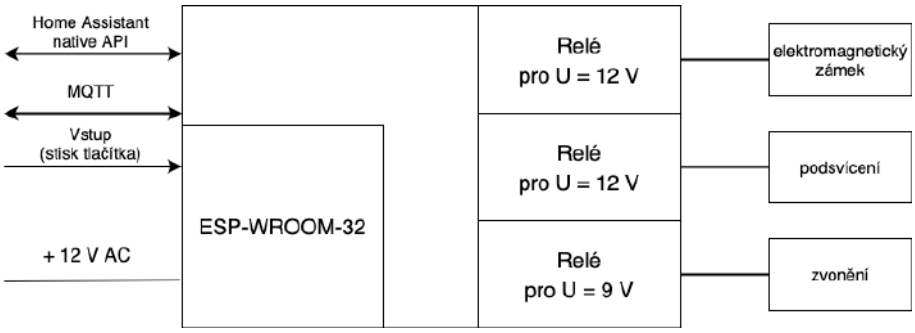
Komunikační protokol bude MQTT a nativní Home Assistant/ESPHome API. V určitých případech bude využito ještě REST API. V případě MQTT a REST API komunikace bude formát zprávy JSON. ESP32 na meteostanici se bude periodicky uspávat a probouzet a posílat data pouze jednou za čas. Podobně budou uspávány i ESP32 na vysavačích Roomba a případné nedoručení zpráv a příkazů v době spánku bude vyřešeno přes MQTT QoS (sekce 2.3.3). U vysavačů dále bude potřeba vyřešit převod logických úrovní z 3,3 V, které využívá ESP32 na 5 V logiku, kterou využívá UART na vysavačích Roomba.

Software ESP32 a propojení

Součástí návrhu je využití platformy ESPHome (sekce 2.2.4), která umožňuje jednoduchou konfiguraci zařízení, OTA updaty a podporuje celou řadu přídavných modulů. Tato platforma je poté připojena k Home Assistantu, který je lokálně hostován v Docker konteineru

v SBC Odroid HC1. Mimo to ještě běží separátně kontejner MQTT brokeru, který se stará o směrování správ mezi dalšími podobnými zařízeními.

3.1.2 Chytrý zvonek



Obrázek 3.1: Návrh vstupů a výstupů chytrého zvonku.

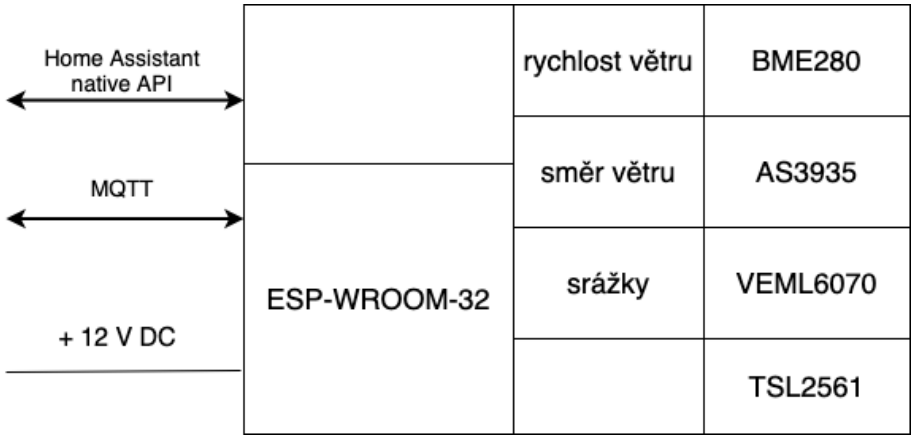
Zvonek detekuje stisk tlačítka pro zazvonění a poté předává tuto informaci Home Assistantu přes API, MQTT brokeru jako zprávu do příslušného tématu (topicu) a také jako REST API požadavek do Assistant Relay. Zároveň zvonek sepne patřičné relé, které se spíná identicky s tlačítkem zvonku.

Home Assistant tuto informaci zobrazí ve svém UI, zaznamená do historie stavů a u zařízení, které mají aplikaci Home Assistant nainstalovanou a mají povoleny notifikace, zobrazí upozornění. K MQTT brokeru jsou připojeny všechny RGBW žárovky na zahradě a pokud obdrží do patřičného tématu zprávu, tak na ni zareagují podle svého nastavení. V tomto případě žárovky zeleně zablikají a poté se vrátí ke svému předchozímu stavu. U Assistant Relay REST API volaní způsobí spuštění příkazu `broadcast` (sekce 2.2.1) s patřičnou zprávou („there is someone ringing the doorbell“) a tato zpráva je poté předána všem zařízením Google Home v domácnosti, které ji poté vyhlásí.

Jelikož domovní zvonky nebývají standardizované, měla by cílová platforma sloužit jako prostředník mezi původním systémem a tlačítkem zvonku. Tudíž je potřeba zvolit například široké spektrum napájecích napětí (AC i DC napětí), mít několik tlačítkových vstupů, několik výstupů (jeden pro emulaci původního tlačítka zvonku, další například pro otevření branky a podobně) a celé toto řešení by mělo byt připojitelné ke stávajícímu systému.

Pro systém jsem se tedy rozhodl využít tří relé, kde první relé slouží k emulaci stistu tlačítka zvonku, druhé pro spuštění elektromagnetického zámku na brance a třetí pro rozsvícení podsvícení u jmenovky). Dále jsou k dispozici dva vstupy, kdy jeden je připojen k původnímu tlačítku a druhý je volný a připraven pro případné další využití. Napájení je vyřešeno diodovým můstkem a 3,3 V lineárním regulátorem, a tudíž lze systém napájet napětími zhruba v rozmězí 4–16 V DC nebo kolem 6–18 V AC. Pro připojení ke stávajícímu zvonku jsou zvoleny svorkovnice, aby se s modulem dobře manipulovalo. Toto zařízení tím pádem půjde použít i se zvonkem Alcad (sekce 2.4.3).

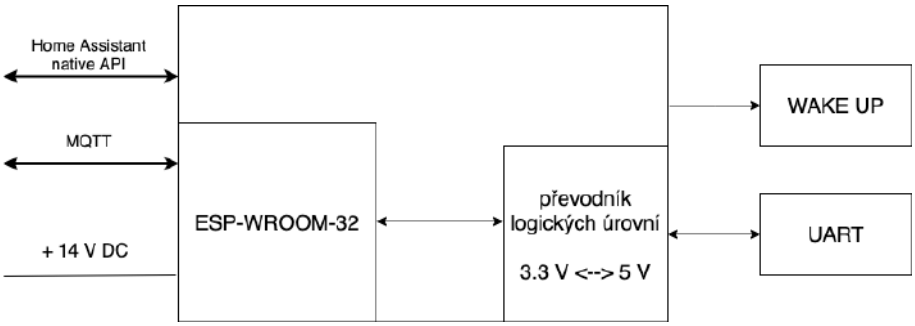
3.1.3 Meteostanice 2.0



Obrázek 3.2: Návrh vstupů a výstupů meteostanice.

Meteostanice (sekce 2.4.3) je opatřena konektory RJ45 (6P4C), které je potřeba vhodně připojit k ESP32. Dále jsou na meteostanici mnou přidané senzory intenzity světla a UV záření, které se připojují přes rozhraní I^2C a dvěma 4 PINovými konektory. Vzhledem k tomu, že jednotka by měla být jednoduše vyměnitelná za jinou, rozhodl jsem se, že budou zachovány stávající RJ45 i 4 PINové konektory a jejich protikusy budou vhodně na DPS k meteostanici umístěny. Dále bude z původní DPS přestěhován senzor AS3935 (senzor blesků) a DPS bude doplněno o senzor BME280, který snímá teplotu, tlak a vlhkost.

3.1.4 Řidicí jednotka pro iRobot Roomba

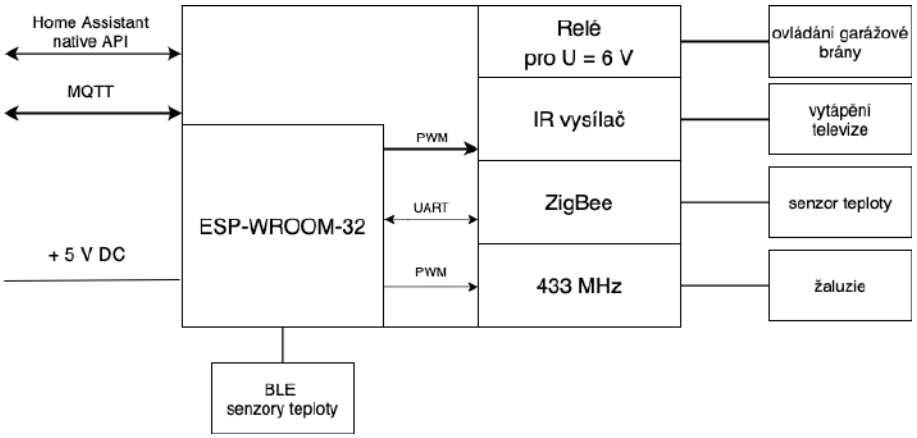


Obrázek 3.3: Návrh vstupů a výstupů modulu chytrého vysavače.

Detekce osob v domě probíhá za pomoci iTag BLE senzoru (sekce 3.2) a také pomocí mobilní aplikace. Pokud přece jen je vysavač spuštěn během přítomnosti někoho doma, tak jej lze odvolat pomocí mobilní aplikace či příkazem pro Google Home. Rovněž v případě manuálního spuštění, mobilem nebo automatickým spustěním v danou dobu, vysavač oznámí pomocí REST API volání službě Assistant Relay, aby příslušnou informaci oznámila.

U tohoto modulu je potřeba vyřešit nejen napájení (MiniDIN konektor na vysavači poskytuje napětí 12–15 V, více v sekci 2.4.3), ale i rozdílnou napěťovou úroveň v OI (5 V logika) a v ESP32 (3,3 V logika). Tím pádem je potřeba vyřešit napájení lineárním regulátorem (jako u všech ostatních PCB), kdy se využije lineární regulátor TS1117BCW33, který toto napětí dokáze regulovat. Druhý lineární regulátor LE50CD se poté stará o zajištění 5 V pro UART vysavače. V neposlední řadě je potřeba využít převodníku jednotlivých napěťových úrovní a to pomocí TXS0102 převodníku logických úrovní.

3.1.5 ESP32 bridge



Obrázek 3.4: Návrh vstupů a výstupů ESP32 bridge.

Mimo ovládání zvonku je součástí návrhu taky DPS ESP32 brány, která se stará o převod různých bezdrátových technologií na Wi-Fi. Konkrétně se stará o BLE, ZigBee, 433 MHz a IR. Dále je v návrhu DPS, které se stará o ovládání vysavače iRobot Roomba pomocí UART a které z vysavače dělá bezdrátové a plně autonomní zařízení. Poslední DPS v návrhu je DPS k meteostanici z BP, které nově obsahuje ESP32 a je celé na tištěném obvodu (původní verze byla na prototypovací THT desce).

Součástí návrhu bridge je bezdrátový vysílač a příjimač na frekvenci 433 MHz, kdy vysílač slouží k ovládání žaluzií a bezdrátového domovního zvonku (venkovní zvonek na brance a vnitřní zvonek byly nezávislé na sobě) a příjimač bude sloužit na vyčítání 433 MHz senzoru teploty, senzoru otevírání garážových vrat a pro detekci zvonění na zvonek na dveřích.

Dále je součástí návrhu ZigBee modul E18-MS1-IPX, který je formou mikrokontroléru a komunikuje s ESP32 pomocí UART komunikace. Ten se stará o vyčítání senzoru teploty WSDCGQ01LM a lze jej využít například i pro ovládání různých žárovek či jiných ZigBee zařízení.

V neposlední řadě obsahuje návrh infračervenou LED diodu a přijímač, které budou sloužit k ovládání televize poblíž jednotky která tento vysílač obsahuje. IR přijimač naopak bude sloužit k čtení dat z dálkových ovladačů, které lze poté namapovat na určité akce (třeba venkovní světla ovládaná ovladačem v domě).

Jelikož je systém automatizace ovládání žaluzií neměnný s tím, jak tomu bylo dosud, tak jsou následující řádky částečně převzaty z mojí bakalařské práce [33]. Pro autonomní řízení žaluzií byla navrhnuta pravidla, která zajištují alespoň částečnou ochranu okna před zničením z důvodu špatných povětrnostních vlivů, ochranu květin před horkem či silným slunečním svitem a automatické ovládání žaluzií na základě denní doby.

Následující pravidla jsou sesazena od nejvyšší priority po nejnižší. Pravidla s nejnižší prioritou platí téměř vždy a v případě, že přijde nějaká událost s vyšší prioritou, převezme kontrolu toto pravidlo a po jeho zneplatnění se žaluzie nastaví do hodnot dle nižších priorit.

`NearStormPreventionRule`

Pravidlo vyhodnotí, zda je bouřka v blíž než 5 kilometrů od domu a pokud ano, zavře všechna okna po dobu uloženou v konstantě `LIGHTNING_COOLDOWN` (výchozí hodnota 10 minut). Po vypršení doby pravidlo přestává platit.

`FarStormPreventionRule`

Pravidlo pro vzdálenější bouřky, u blesků blíž než 15 kilometrů preventivně zavře žaluzie do poloviny oken a v případě, že nějakou dobu nedošlo k dalším bouřkouvým událostem, se toto pravidlo zneplatní a je předáno řízení pravidlu s nižší prioritou.

`VeryFastWindPrevention`

V případě prudkého (velmi silného) větru (cca 50 km/h[1]) dojde k uzavření všech oken, dokud se rychlost větru nesníží. Nepočítá se rychlost okamžitá, ale průměrná rychlost větru vycházející z počtu naměřených hodnot podle konstanty `NUMBER_OF_MEASUREMENTS` (v případě výchozí hodnoty se jedná o průměrnou rychlost větru za posledních zhruba 5 minut).

[1] https://www.tpocasi.cz/meteorologicke-pojmy/beaufortova-stupnice-sily-vetru/

`WindPrevention`

Pravidlo pro silný vítr (39 km/h a více) do půlky zavře pouze žaluzie, na které vítr vane. V tomto případě se tedy využívá převážného směru větru za poslední dobu a toto pravidlo slouží zejména k tomu, aby otevřená okna zasažená větrem nebyla poškozena.

`StrongRainPrevention`

Pokud dojde k velmi silné intenzitě srážek (srážkový úhrn vyšší, než je 40 milimetrů za hodinu[2]), dochází k úplnému uzavření oken všech do doby, než přestane pršet nebo se déšť zmírní. Toto pravidlo slouží zejména k tomu, že za velkého deště může dojít k namočení podlah v domě nebo případných elektrických zařízení u oken, pokud není okno řádně uzavřeno.

`RainPrevention`

Toto pravidlo se aktivuje v případě silného deště (cca 8 milimetrů za hodinu), aby došlo k ochránění domu před vnikem vody otevřenými okny. Dochází k uzavření všech oken do půlky. Rovněž toto pravidlo následuje předchozí pravidlo, tím, že pokud velmi silná intenzita přechází na silnou intenzitu, dojde k otevření všech žaluzií alespoň do poloviny.

`HeatPreventionRule`

V horkých letních dnech je potřeba ochránit žaluziemi květiny v domě před poškozením od slunce a také udržovat dům v rozumných teplotách. K tomu slouží toto pravidlo, které měří průměrnou teplotu v domě i venku, dále potom intenzitu světla a UV záření. Pokud je opravdu horký den, s vysokým stupněm UV záření a v domě teplota také překračuje povolené meze, dojde k uzavření žaluzií do poloviny. Toto má za následek ochranu před přímým slunečním světlem většiny rostlin a také před nadměrným růstem teplot.

`NightCloseRule`

Pravidlo zajišťující uzavření žaluzií večer, když poklesne sluneční svit pod patřičnou úroveň. Toto pravidlo počítá s průměrnou intenzitou světla za posledních několik minut a je aktivováno pouze ve večerních hodinách. Tím pádem se přechází například uzavření žaluzí při zastínění senzoru.

`DayOpenRule`

Zde dochází k opačné události než u předchozího pravidla, ráno se žaluzie pomocí tohoto pravidla samy otevřou.

`NoLumiSensorRule`

V případě, že nedošly žádné údaje ze senzoru intenzity světla (například kvůli poruše), dojde ke zjištění času východu/západu slunce z internetu a poté k provedení patřičné akce.

[2] https://cs.wikipedia.org/wiki/Sr%C3%A1%C5%BEky

`Default`

Výchozí pravidlo pokrývá dobu, kdy například jsou již večerní hodiny, avšak slunce ještě nezapadlo. Pravidlo si pamatuje poslední událost z předchozích tří pravidel a pokud dojde třeba k bouřce mezitím, je tímto pravidlem vrácen předchozí stav žaluzií před bouřkou (například při krátké bouřce v 17 hodin by se jinak již žaluzie neotevřely, protože z hlediska času již není den a blíží se doba uzavírání).

3.1.6 Světla

Pro automatické osvětlení zahrady a notifikaci o různých stavech IoT systému byla do návrhu zařazena implementace a nahrání vlastního firmwaru pro žárovky FCMILA, které jsou založeny na ESP8266. Tyto žárovky jsou pro patici E27, obsahují RGB a čistě bílé osvětlení s možností regulace jasu.

Světla na zahradě reagují podobně jako žaluzie na data z meteostanice, ale také na zvonění zvonku nebo, pokud je potřeba předat různé notifikace. Pokud někdo zazvoní, světla zeleně zablikají a poté se vrátí k původnímu stavu (popsáno v sekci 3.1.2). Dále, podle denní doby (střídání dne a noci s využitím dat ze senzorů intenzity světla z meteostanice) se rozsvítí či zhasnou. Pokud meteostanice detekuje bouřku, tak je toto oznámeno zablikáním světel s tím, že barva nebo čas probliknutí korespondují se vzdáleností bouřky od domu). Dále světla upozorňují na určité události v domě (někdo dorazil domů, vysavač začal s vysáváním, vysavač se zasekl o překážku) různými barvami či barevnými efekty.

3.2 Senzory

Jako senzory se využijí již stávající z meteostanice (sekce 2.4.3), takže senzory intenzity světla a UV, senzory teploty, tlaku rychlosti a směru větru a v neposlední řadě senzor blesků AS3935. Nově se na stávající meteostanici přidává i senzor vlhkosti. Mezi nové senzory uvnitř patří BLE senzory od Xiaomi (sekce 2.4.3), ZigBee senzor teploty a vlhkosti Xiaomi WSDCGQ01LM[3] a také iTag BLE senzor, který bude slouží k vyhledávání klíčů (je v plánu vyvolat zvonění přes Google Home) a také jako senzor přítomnosti (pokud je detekováno, že se někdo vrátí domů v době, kdy vysavače iRobot pracují – sekce 3.1.4, tak dojde k vyvolání příkazu k návratu do nabíjecí stanice – sekce 2.4.3).

3.3 Aktuátory

Ovládán bude vysavač, žaluzie, vjezdová brána, domovní branka, venkovní osvětlení, domovní zvonek a zařízení Google Home, které budou notifikovat o různých stavech. Žaluzie mají proprietární komunikační protokol Somfy RTS běžící na frekvenci 433 MHz. Zpráva sestává z kodu protokolu, příkazu, checksumu, rolling kódu a jedinečný identifikátor ovladače, který příkaz odeslal (více v sekci 2.4.3). Branka se otevírá spínáním pomocí relé, kdy po přivedení 12 V AC dojde k dočasnému odblokování elekromagnetického zámku a poté ji lze otevřít. Žárovky obsahují mikrokontrolér, který ovládá RGBW LED diody. Světla poté mohou reagovat na různé stavy IoT systému a notifikovat uživatele příslušnou barvou světla a efektem. Vysavače mohou komunikovat s uživatelem buď pomocí fyzického stisku kláves na vysavači, s použitím dálkového ovladače nebo dokonce jej lze připojit k počítači či mikrokontroléru za pomoci UART rozhraní. Při UART komunikaci je využito komunikačního

[3] https://www.zigbee2mqtt.io/devices/WSDCGQ01LM.html

protokolu Open Interface (sekce 2.4.3), kterým lze ovládat veškeré činnosti vysavače. Pro správnou funkci UART rozhraní se na vysavači nachází ještě takzvaný BRC pin (baud rate change), který k vynucení nižší rychlosti UART komunikace, protože ihned po zahájení komunikace vysavač běží na 115200 baudů a pro rychlost 19200 baudů je potřeba tímto pinem vynunit tuto rychlost (baudy lze libovolně měnit během UART komunikace, tato metoda je pro stav nouze, kdy druhé zařízení umí pouze nižší rychlosti UART komunikace). Tento pin však slouží hlavně jako `WAKE UP` signál pro vysavač, protože vysavač se periodicky uspává, pokud neprovádí nějakou činnost, aby šetřil baterii. Při režimu spánku se přeruší i UART komunikace, takže vysavač nereaguje na výzvy.

3.4 Brána a její komponenty

3.4.1 Docker kontejnery

IoT systém bude obsahovat 4 různé docker kontejnery (sekce 2.5.5). Jeden pro Home Assistant, druhý pro MQTT broker, třetí pro Assistant Relay a poslední bude se starat o automatizaci žaluzií.

3.4.2 MQTT broker

MQTT broker má na starost vystupovat jako centrální bod pro MQTT protokol (sekce 2.3.3) a má za úkol směrovat zprávy, které se pomocí tohoto protokolu přenáší. Pro využití v práci byl vybrán broker Eclipse Mosquitto, protože je relativně populární a má k dispozici oficiální Docker image na Docker Hubu[4].

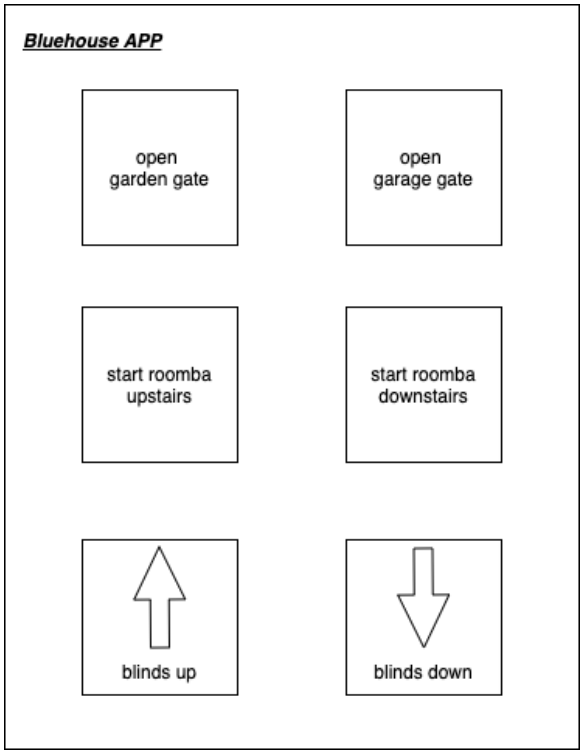
3.4.3 NodeJS Assistant Relay

Možností napojení IoT systému na Google Home existuje mnoho. Systém lze napojit na Home Assistant a poté využít jejich cloudové služby, která podporuje Google Home. Dale lze využít služby gbridge.io[5], která slouží jako převodník mezi API Google Home a MQTT. Ani jedna z výše zmíněných platforem však neumí zajistit předání příkazu přímo jednotlivým Google Home zařízením pomocí API. Tento problém řeší Assistant Relay[6], který funguje jako virtuální Google Home zařízení, do kterého lze předávat libovolní příkaz, který lze říct Google Assistantovi hlasem. Lze jej tedy využít k tomu, aby Google Home zařízení notifikovali v domě obyvatele, že někdo zvoní nebo kdo právě dorazil domů, či že ve sklepě je voda. Komunikace s uživatelem probíhá pomocí jednoduchého REST API (sekce 2.3.3) volání.

3.5 Mobilní aplikace

Součástí návrhu je i jednoduchá mobilní aplikace, kterou se bude dát otevřít branka, vjezdová brána, řídit vysavač a ovládat žaluzie. Aplikace bude komunikovat se systémem přes MQTT (případně REST API) a bude členěná do dlaždic, kdy každá dlaždice bude právě jedna akce pro IoT systém (zapnout vysavač, vysunout žaluzie).

[4] https://hub.docker.com/_/eclipse-mosquitto
[5] https://gbridge.io/
[6] https://github.com/greghesp/assistant-relay

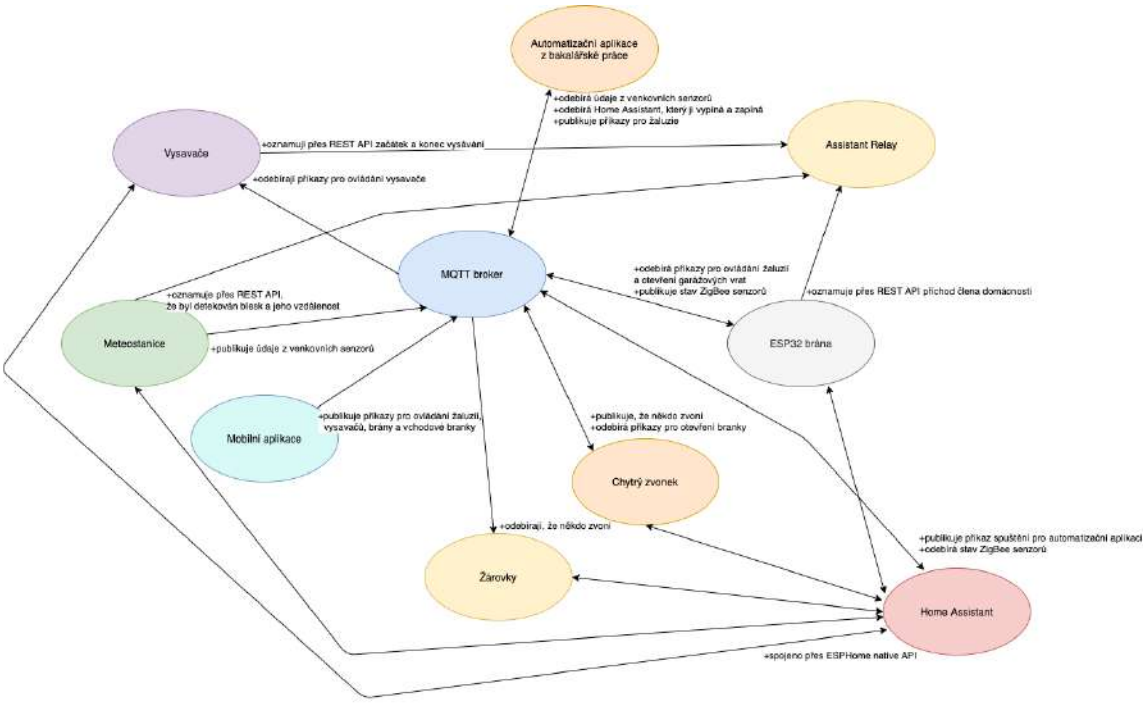


Obrázek 3.5: Návrh mobilní aplikace.

3.6 Vzájemná komunikace

Zařízení spolu komunikují několika bezdrátovými technologiemi a v případě domácí sítě přes několik komunikačních protokolů a API. Bezdrátová komunikace probíhá na technologiích Wi-Fi, BLE, ZigBee, 433 MHz, 868 MHz (ovladač vjezdové brány). Dále je zařízení připraveno pro IR přenos.

Co se týče komunikačních protokolů a API, tak komunikace probíhá po místní síti. ESP32 s ESPHome využívají jak komunikaci přes MQTT broker (například zvonek a žárovky), tak i pomocí nativního API pro Home Assistant. V případě použití nativního API probíhá detekce zařízení automaticky a není potřeba nic nastavovat. Chytrý zvonek komunikuje s Google Home pomocí Assistant Relay (sekce 3.4.3), v tomto případě je využito REST API. Automatizační aplikace z bakalářské práce komunikuje se žaluziemi pomocí MQTT brokeru a nově vyvinutá mobilní aplikace rovněž tak. Senzory jsou připojeny přes BLE, ZigBee, či 433 MHz, ty poté zpracuje ESP32 brána a pošle do sítě přes MQTT a nativní Home Assistant API. Vzájemné propojení a komunikace je zobrazena na obrázku 3.6.



Obrázek 3.6: Diagram vzájemného propojení komponent a komunikace.

Kapitola 4

Implementace

V této kapitole je popsán postup implementace jednotlivých částí systému, jako je realizace hardwarových částí systému, popis implementace softwaru a další detaily vývoje. Jelikož je popis komponent relativně rozsáhlý, je tato kapitola členěna do sekcí podle typu zařízení a to je následně popsáno jak po HW, tak ji po SW stránce.

4.1 Implementace koncových zařízení

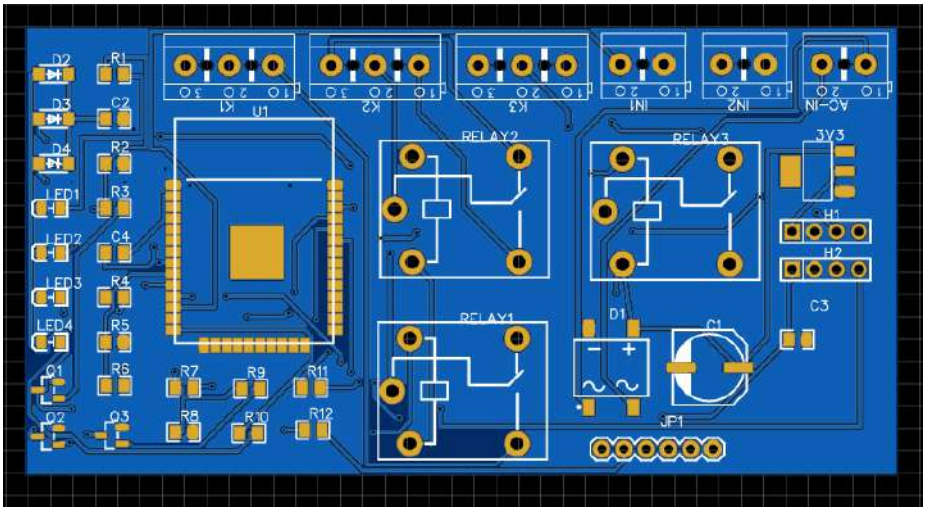
Byly realizovány 4 typy DPS. Každé z nich obsahuje mikrokontrolér ESP32 (v jednom případě doplněn i o ZigBee mikrokontrolér E18-MS1-IPX), své vstupy/výstupy (relátka, vstupní terminály ve svorkovnicovém provedení) i vyřešené napájení (regulátory napětí, v případě zvonku i AC-DC převodník).

4.1.1 Chytrý zvonek

Implementace chytrého zvonku sestavá jak z implementace HW části (DPS, osazení komponent), tak z SW části.

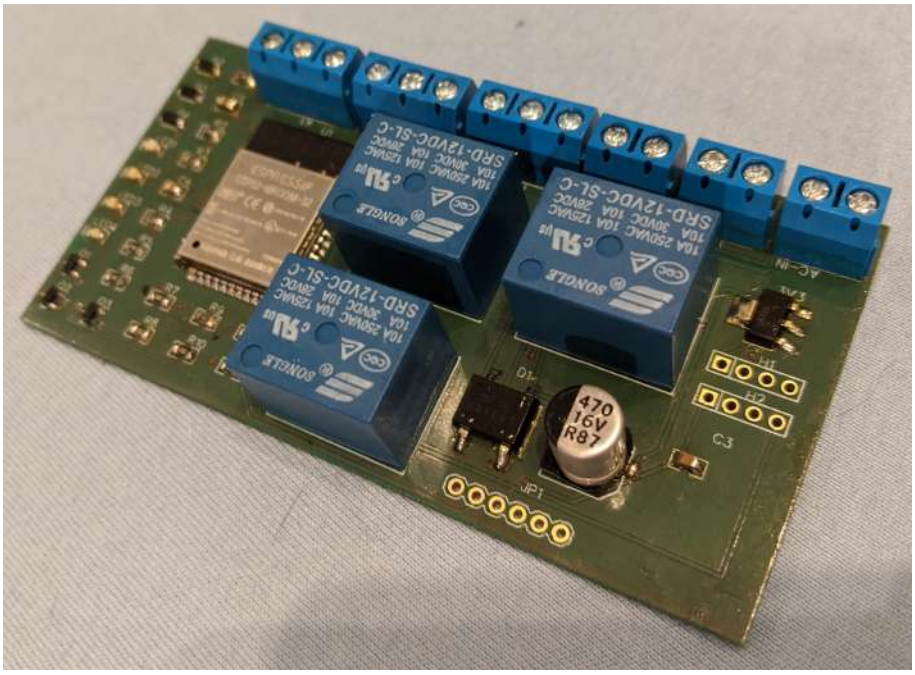
Realizace HW řešení

DPS k chytrému zvonku (obvodové zapojení B.1) obsahuje AC-DC převodník sestávající se z diodového můstku B250C1000SMD, vhodného elektrolytického kondenzátoru a lineárního regulátoru TS1117BCW33, který zajišťuje napájení ESP32 napětím 3,3 V. Dále DPS obsahuje tři 12 V relé (PCB počítá s napětím kolem 12 V AC), 3 modré LED diody svitící při aktivaci relé a jednu zelenou status LED diodu. Vývody z relé jdou na svorkovnice, stejně jako dva vstupy určené pro tlačítka a vstup pro napájení. Dále DPS obsahuje vyvedené I^2C rozhraní formou dvou pin headerů pro možné připojení libovolného senzoru. Mimo to DPS obsahuje ještě FTDI rozhraní pro snadné nahrání firmwaru.



Obrázek 4.1: Návrh DPS chytrého zvonku.

DPS zvonku byla osazena jako první, byly nainstalovány 3 relé, svorkovnice, ESP32 mikrokontrolér a další komponenty (obrázek 4.2). Přes USB-UART převodník byl nahrán FW a bylo ostestováno nahrávání nového firmware přes Wi-Fi.



Obrázek 4.2: Finální osazené DPS chytrého zvonku.

Realizace software

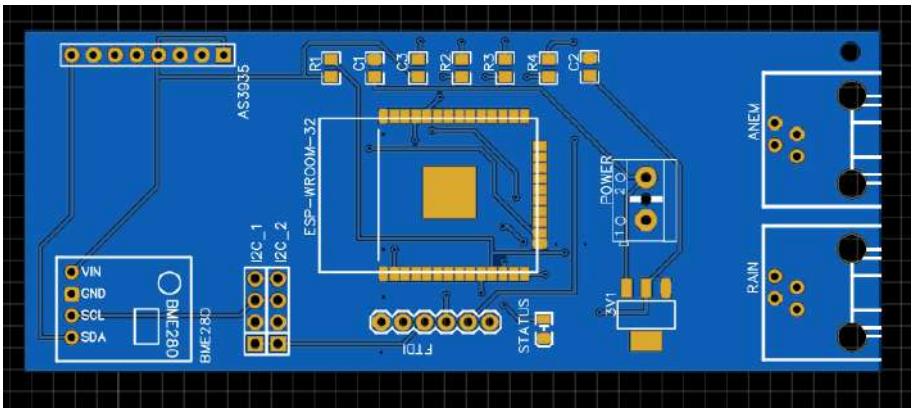
Byly zadefinovány tři přepínače, jeden na otevírání branky, druhý na zvonek a třetí na podsvícení zvonku. Pokud se přepínač ovládá přes Home Assistant, tak relé (komponenta switch) pro zvonění a branku se sepnou po určitý čas (zazvonění trvá 2 sekundy a elektromagnetické otvírání u branky je aktivní po dobu 3 sekund). Relé na osvětlění se nevypíná po určitém čase a lze jej zapnout trvale. Dále jsou implementovány dva vstupy, z nichž jeden je využit pro tlačítko zvonku a druhý je neobsazen. V případě stisku tlačítka zvonku, je tento vstup přímo napojen na relé, které spíná zvonění a je sepnuto do doby, dokud je stisknuto tlačítko zvonku. Zároveň je tento stisk oznámen přes REST API do Assistant Relay platformy a přes MQTT žárovkám. Stav systému je oznamován zelenou status LED, která když bliká, tak není zařízení přípojeno a pokud svítí bez přestávky, tak je zařízení připojeno jak k Wi-Fi, tak k MQTT brokeru a Home Assistantu.

Jednotlivá relé jsou v kódu řešena jako komponenta switch na platformě `gpio`. Na určitém pinu je připojeno relátko, které touto platformou jde ovládat. Pro potřeby ovládání z Home Assistantu byla tato platforma integrována do `template` platformy. Ta slouží jako abstraktní vrstva pro přepínač tak, aby u branky se relé sepnulo pouze na určitý počet vteřin (a tím umožnilo otevřít banku) a poté se zase vypnulo. Vstupní tlačítka byla implementována jako vstup (komponenta `binary_sensor`) na platformě `gpio` a bylo potřeba vstup nastavit jako `INPUT_PULLUP` (vstup je po stisku propojen s `GND` a tím dojde k logické nule) s invertovanou hodnotou. Aby tento vstup fungoval správně, tak byl implementován tzv. `debouncing`, kterým lze odfiltrovat velmi krátké stisky tlačítka, nedokonalý kontakt obvodu, případně poklesy napětí na pinu, které by mohly být nesprávně interpretovány (ukázka konfigurace v příloze C).

4.1.2 Meteostanice 2.0

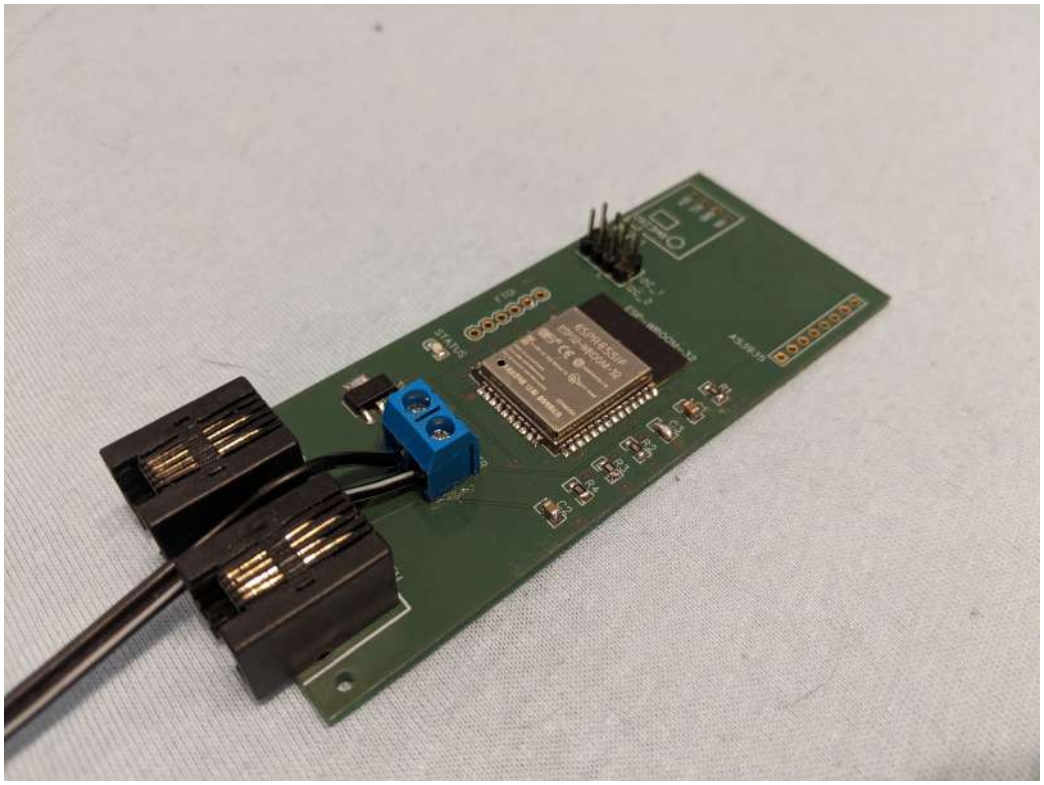
Realizace HW řešení

PCB pro meteostanici (obvodové zapojení B.2) obsahuje již zmiňovaný 3,3 V regulátor, dále pak konektor pro senzor blesků AS3935 a senzor teploty, tlaku a vlhkosti BME280. Mimo to obsahuje další 2 I^2C konektory pro senzor UV záření a intenzity světla (VEML6070 a TSL2561). DPS také obsahuje dva 6P4C RJ-45 konektory, ke kterým se připojuje anemometr (senzor rychlosti a směru větru) a senzor srážek z kitu SEN-08942.



Obrázek 4.3: Návrh DPS meteostatnice.

Deska poté byla osazena komponenty (obrázek 4.4), otestována a připravena na umístnění na střechu. Jelikož se senzory AS3935 a BME280, stejně jako TSL2521 a VEML6070 nachází již na původní meteostanici, tak budou přidány až při nasazení systému.



Obrázek 4.4: Osazené DPS meteostanice 2.0.

Realizace software

Meteostanice měla senzory TSL2561, AS3935 a BME280 již součástí platformy ESPHome.io, tudíž je stačilo pouze správně iniciovat[1] a integrovat do Home Assistantu. U senzorů směru větru a intenzity srážek však bylo potřeba použít platformu `pulse_counter`, která na sestupné hraně signálu počítá počet pulzů, a ty potom pronásobí konstantami, které po výpočtu v platformě `integration` vrací rychlost větru (m/s) a srážkový úhrn (mm/h).

U senzoru VEML6070 bylo potřeba si napsat vlastní komponentu[2], která s využitím Adafruit knihovny pro tento senzor umožňuje vyčítat data z tohoto senzoru.

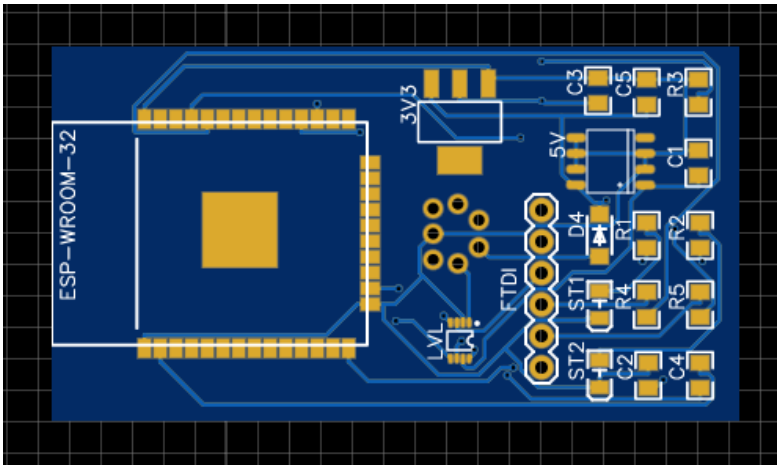
4.1.3 Řidicí jednotka pro iRobot Roomba

Realizace HW řešení

Poslední z DPS (obvodové zapojení B.3) slouží k ovládání vysavače iRobot Roomba pomocí Wi-Fi. Obsahuje lineární regulátory TS1117BCW33 pro 3,3 V a LE50CD pro 5 V. Jelikož iRobot obsahuje rozhlaní OI (Open Interface), které je založeno na UART komunikaci s 5 V logikou, bylo potřeba vyřešit převod 3,3 V logiky, kterou disponuje ESP32 a 5 V logiky,

[1] https://esphome.io/components/sensor/as3935.html
[2] https://esphome.io/components/sensor/custom.html

kterou podporuje OI. Toto bylo dosaženo použitím TXS0102, který se stará o převod mezi 5 V a 3,3 V logikou. Dále DPS obsahuje dvě status LED diody a pin header ve tvaru DIN konektoru pro připojení k vysavači.



Obrázek 4.5: Návrh DPS Wi-Fi modulu k iRobot Roomba.

Komponenty byly připájeny na PCB (obrázek 4.6), u miniaturního TXS0102 bylo využito pájecí pasty a horkovzdušné pájecí stanice. Konektor MiniDIN byl osazen piny, aby šel připojit do slotu na vysavači. Do mikrokontroléru byl nahrán SW přes USB-UART převodník.

Realizace software

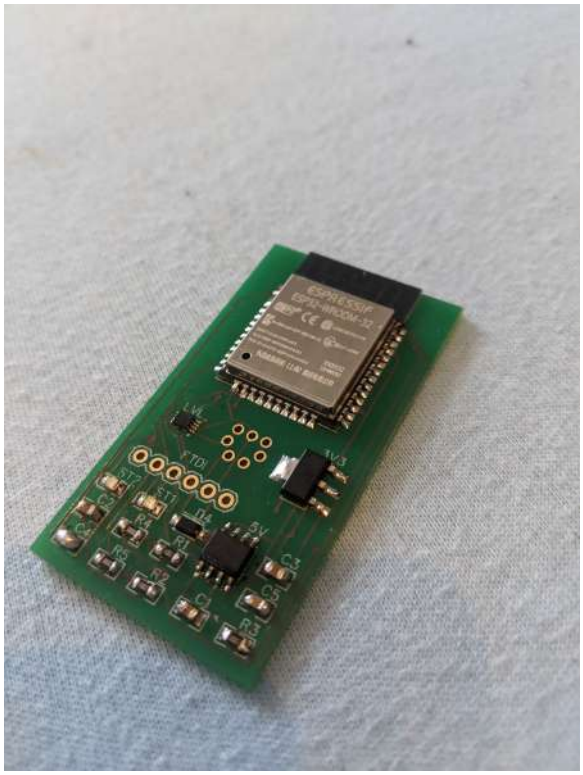
U vysavače bylo potřeba napsat si vlastní rozhraní, které propojuje Arduino knihovnu Roomba s ESPHome. Knihovna se stará o komunikaci přes UART a překládá zprávy tak, aby jim vysavač rozumněl, a aby ESP32 dokázalo z datového toku dostat správné informace. Před každým přenosem zprávy je potřeba vysavač probudit na pinu `BRC` a poté poslat správný UART příkaz, na který vysavač zareaguje odpovídajícím způsobem. Lze vyčítat stav baterie, status vysavače či informace z libovolného senzoru.

Vlastní kompomenta k vysavači sestává z Arduino knihovny Roomba 1.3[3], která slouží k usnadnění zpracování dat z vysavače a detekci stavů. Komponenta, kterou jsem implementoval, sestává z komponent binárního senzoru (`BinarySensor`, číselného senzoru (`Sensor` - vrací číselnou hodnotu) a přepínače (`Switch`). Senzorové komponenty jsou iniciované jako tzv. `PollingComponent`, kdy čtení určitého senzoru neprobíhá každý takt, ale pouze jednou za určitý čas. V tomto případě jednou za 5 vteřin. V rámci inicializace komponenty (metoda `void setup()`) je zahájena sériová komunikace, iniciován probouzecí pin a spojení s vysavačem.

Další metotou je `void write_state(bool state)`, která je metodou komponenty `Switch`. Obsluhuje tedy přepínače, které slouží k zaslání příkazu vysavači k vysávání či pozastavení činnosti, k návratu do nabíjecí stanice nebo k čištění určitého místa. Samotné odeslání příkazu vysavači je uvozeno nejprve probouzecí sekvencí na pinu `BRC` následované patřičným příkazem. V závěru je publikování potvrzení o úspěšné změně stavu.

Poslední metoda `void update()` se stará o vyčítání dat ze senzorů vysavače. Lze vyčítat data jako, stav nabití, vzdálenost od překážky, spotřebu energie či kapacitu bate-

[3] https://github.com/Apocrathia/Roomba



Obrázek 4.6: Osazené DPS řidicí jednoty iRobot Roomba.

rie a stav nabíjení. Jednotlivé senzory mají v UART komunikaci určitý číselný kód (například kód 25 vrací stav nabití baterie v mAh) a s jejich použitím při volání metody `Roomba::getSensors(sensorID, uint_8t * out, sizeof(out))` lze zjistit stav senzoru a ten následně publikovat. Knihovna však vrací surová data nakopírovaná po bajtech do pole bez patřičného kontextu, a tudíž je potřeba data, která se přijala zpracovat a převést na správné hodnoty. Některá data jsou v jednom bajtu (například stav nabíjení), data ze senzoru vzdálenosti jsou ve dvou bajtech a některé hodnoty jsou znaménkové (odběr proudu je záporná hodnota při vybíjení a kladná při nabíjení). Touto metodou jsou obsluhovány jak číselné, tak binární senzory a výstupní data jsou publikována metodou příslušné třídy.

```
if (_command == "distance"){
    int32_t state = 0;
    this->roomba.getSensors(19, sensor_values, sizeof(sensor_values));
    int16_t distance = sensor_values[0] * 256 + sensor_values[1];
    state += distance;
    Sensor::publish_state(state);
}
...
else if (_command == "charging_status"){
    this->roomba.getSensors(21, sensor_values, sizeof(sensor_values));
    uint8_t charging_code = sensor_values[0];
    bool state = (chrg_code == 1|| chrg_code == 2 || chrg_code == 3);
    BinarySensor::publish_state(state);
```

```
}
```

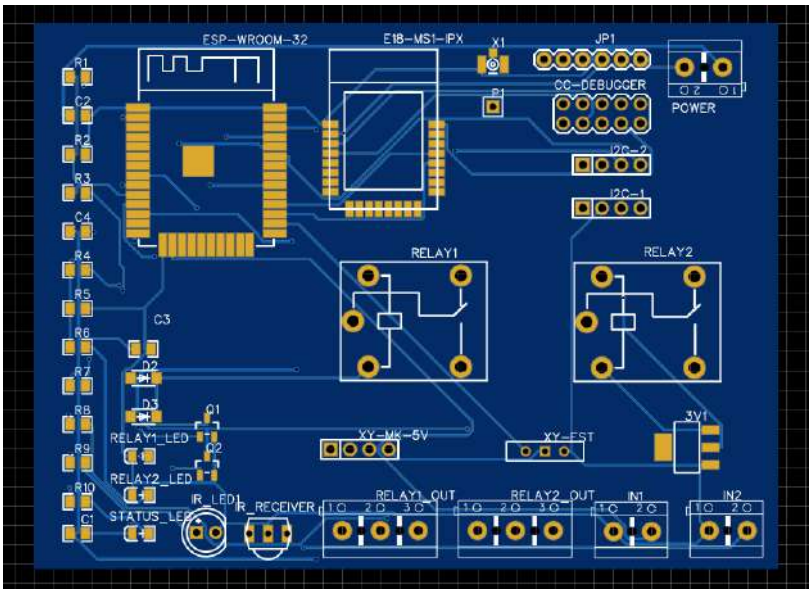
Výpis 4.1: Dotazování senzorů vysavače pomocí Open Interface (sekce 2.4.3)

Vysavač je pravidelně při nečinnosti delší než 5 minut uspává (pokud nevysává nebo neprobíhá interakce s uživatelem). Při režimu spánku dochází k přerušení i UART komunikace. Je tedy vhodné vysavač probudit v případě potřeby. Zabránit spánku lze tím, že během 5minutového okna, kdy vysavač nespí se posílá probouzecí puls (BRC pin obdrží každých několik minut puls s logickou nulou). V mezičase ESP32 je uspané a šetří energii, pokud v době spánku obdržel mikrokontrolér příkaz přes MQTT, tak QoS (sekce 2.3.3) se postará o doručení zprávy po probuzení.

4.1.4 ESP32 bridge

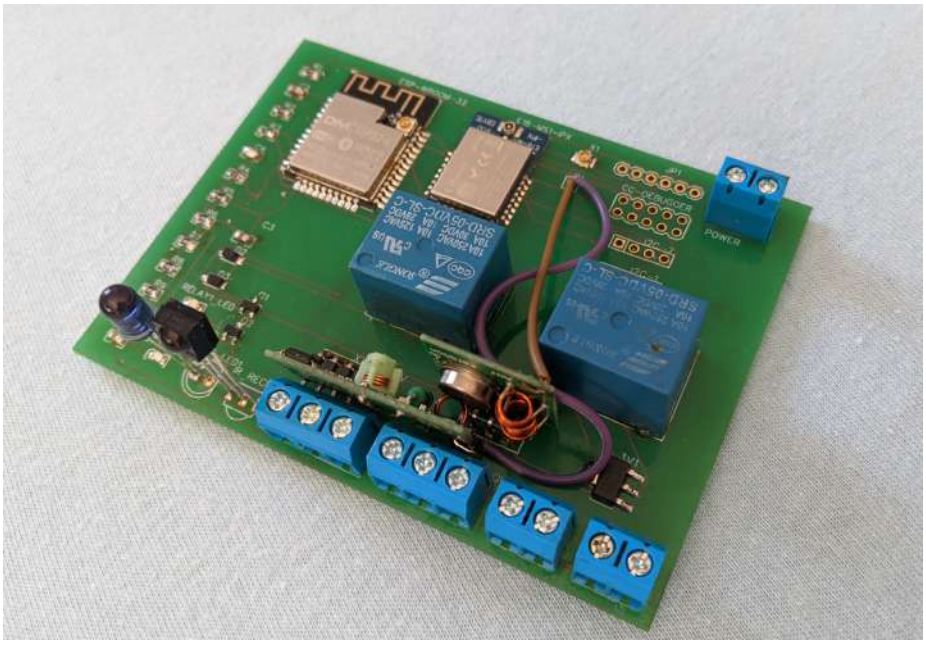
Realizace HW řešení

Toto DPS (obvodové zapojení B.4) obsahuje lineární regulátor TS1117BCW33, avšak už ne AC-DC převodník (předpokládá se použití DC 5 vstupního napětí). Dále pak dvě relé a svorkovnice s výstupy, svorkovnici pro vstup a dvě svorkovnice pro tlačítka. Co se týče výbavy, je toto DPS zajímavé použitím jak ESP32, tak i ZigBee mikrokontrolérem E18-MS1-IPX. DPS dále obsahuje 433 MHz vysílač i přijímač a také IR přijímač a IR LED diodu pro vysílání. Samozřejmostí je použití LED pro relé i status, a také I^2C konektory.



Obrázek 4.7: Návrh DPS ESP32 bridge.

DPS bylo osazeno (obrázek 4.8) dvěma 5 V relé, 433 MHz přijímačem i vysílačem, jejichž antény byl nasměrovány do `U.FL` konektoru pro použití s anténou. Dále byly přidán IR přijímač a vysílač i ZigBee modul. Do ZigBee modulu i do ESP32 byl nahrán firmware přes USB-UART převodník.



Obrázek 4.8: Finální osazené DPS ESP32 bridge.

Realizace software

ESP32 brána obsahuje dva mikrokontroléry, v E18-MS1-IPX se nachází ZigBee mikrokontrolér CC2530, ve kterém je nahrán kód[4], který povoluje UART komunikaci na PINech P02 a P03. UART komunikaci poté čte ESP32 a její obsah převání na datový tok. Tento tok poté zpracovává řídicí jednotka a přistupuje k němu jako k fyzicky připojenému sériovému zařízení. Toto zařízení je poté napojeno na Home Assistanta přes patřičný plugin (zigbee2mqtt[5]), který se stará o detekci, párování a správu ZigBee zařízení.

Dále je v kódu brány implementováno ovládání pro dvě relé. Jedno se stará o ovládání ovladače brány (868 MHz ovladač, který se špatně implementuje v SW) a druhé je určeno pro budoucí použití. Při aktivace relé v Home Assistantu dojde k jeho aktivaci na 400 ms (odpovídá krátkému stisku tlačítka) a brána se otevře. Dále zařízení obsahuje 433 MHz vysílač a přijímač, s vysílačem pracuje mnou implementovaná knihovna pro ovládání žaluzií, která byla přepsána pro integraci do ESPHome. Přijímač pouze vyčítá určité typy zpráv, zpráv a pokud narazí na známou zprávu (například z magnetického senzoru otevření dveří), tak oznámí stav v Home Assistantovi. IR přijímač je připraven na čtení dálkových ovladačů a IR LED má implementováno zapínání a vypínání TV. Přes I^2C je připojen senzor BMP280, který se stará o monitorování teploty a tlaku.

Samotný kód sestává ze souboru `esp_gateway.yaml`, který obsahuje konfigurace Wi-Fi, MQTT a home-assistantu. Ovladač žaluzií je implementován jako `cover` komponenta založená na `template` platformě[6]. Tato platforma slouží v tomto případě pro sdružení a mapování více přepínačů (komponenta `switch`) pod jednu komponentu, která v systému vystupuje jako žaluzie. Jednotky v komponentě `switch` jsou instance třídy mojí knihovny pro ovládání žaluzií Somfy, kdy každá instance obsahuje jednu akci určitého ovladače (pohyb

[4] https://www.zigbee2mqtt.io/information/connecting_cc2530.html
[5] https://www.zigbee2mqtt.io/integration/home_assistant.html
[6] https://esphome.io/components/cover/template.html

žaluzií nahoru a dolů, zastavení žaluzií, otevření žaluzií do půlky, zastavení pohybu žaluzií a párování). Simulovat lze takto libovolné množství ovladačů a každý ovladač lze spárovat s více žaluziemi). Jednotlivá tlačítka se instancují v komponentě `switch` na platformě `custom` (`custom component`) pomocí položky `lambda`, která umožňuje volat kód v C++ přímo z konfigurace níže.

Třída SomfyBlind je do konfigurace přidána přes `include` a její implementace se nachází v souboru `blinds.h`. V tomto souboru se nachází jak samotná definice vlastní (`custom`) komponenty, tak i mnou implementovaný Somfy protokol. Tento protokol sestává z tzv. `wake up` zprávy, která vyšle data v určitém časovém sledu, aby probudila žaluzie. Tato zpráva ale není povinná v rámci tohoto protokolu, ale slouží k tomu, aby žaluzie zaznamenaly opravdu každou zprávu. Může se stát, že by na některá stiknutí nemusely reagovat, protože nezaznamenaly kompletní zprávu (sekce 2.4.3). Probouzecí data jsou následovány hardwarovou a softwarovou synchronizací, což je posloupnost bitů, které jsou v určitém časovém rozmezí posílány žaluzii a ta poté má informaci, že zpráva je určena pro ni. V poslední část zprávy je datový rámec, který obsahuje šifrovací klíč, kód tlačítka ovladače, kontrolní součet, dále tzv. `rolling code`, který se inkrementuje s každým stiskem ovladače a v neposlední řadě adresa ovladače. Pokud je ovladač spárovaný se žaluzií (sekce 2.7 a je dodrženo inkrementování `rolling code` a správné časování, tak je zpráva kompletní a žaluzie vykoná patřičný příkaz.

```
void prepare_frame(uint8_t button) {
  uint8_t checksum = 0;

  _frame[0] = 0xA7;      //# Protocol key
  _frame[1] = button << 4;     //# Button - 4 bits, 4 bits checksum
  _frame[2] = _rolling_code >> 8;    //# Rolling code (big endian)
  _frame[3] = (_rolling_code & 0xFF);  //# Rolling code
  _frame[4] = _RemoteAddress >> 16;   //# remote addr
  _frame[5] = ((_RemoteAddress >> 8) & 0xFF); //# remote addr
  _frame[6] = (_RemoteAddress & 0xFF);   //# remote addr

  for(unsigned int i = 0; i < 7; i++) {
      checksum = checksum ^ _frame[i] ^ (_frame[i] >> 4);
  }
  checksum &= 0b1111;     //# we need only last 4 bits

  _frame[1] |= checksum;    //# replacing zeros in checksum with new

  for(unsigned int i = 1; i < 7; i++) {
      _frame[i] ^= _frame[i-1];   //# FINALIZING FRAME
  }
}
```

Výpis 4.2: Tvorba datového rámce pro ovládání žaluzií (struktura popsána na obrázku 2.7)

Další vlastní komponenta je most mezi UART sběrnicí ke které je připojen ZigBee mikrokontrolér a `Telnet`. Zařízení iniciuje vlastní server pomocí knihovny `WifiServer`[7], ke

[7] https://github.com/espressif/arduino-esp32/blob/master/libraries/WiFi/src/WiFiServer.h

kterému se poté na určitém portu připojuje klient. Pokud je přijata zpráva mikrokontrolérem přes Telnet, ESP provede zprávu zpracuje a pošleji sériovou komunikací do ZigBee mikrokontroléru. Naopak, pokud ESP zaznamená zprávu doručenou přes UART ze ZigBee modulu, tak ji po zpracování odešle přes webový server klientovi.

4.1.5 Chytré žárovky

Systém chytrého zvonku je napojen na žárovky FCMILA, které původně obsahovaly SW platformy TUYA, avšak byly nahrazeny[8] vlastním implementovaným SW taky založeným na platformě ESPHome. Tyto žárovky se nachází po celé zahradě a jejich SW byl implementován tak, aby se v průběhu dne chovaly, jako běžná chytrá žárovka (včetně automatického rozsvěcování po západu slunce), ale zároveň sloužily jako světelná notifikace, když někdo zvoní. Po zazvonění tedy žárovka upozorní, na zvonek, tak, že po dobu 5 sekund zasvítí zeleně a poté přejde do původního stavu. Původní zvonek totiž byl slyšet jen v domě.

U žárovek bylo potřeba zjistit, které piny jsou přiřazeny jednotlivým barevným složkám světla a bílému světlu. Bylo zjištěno, že na `GPIO14` se nachází červená barevná složka, na pinu `GPIO12` se nachází zelená a na pinu `GPIO13` je připojena modrá barva LED diody. Bílé LED diody jsou zvlášť a nachází se na pinu `GPIO4`, jedná se tedy o tzv. RGBW žárovku. Tyto piny byly poté přiřazeny komponentě `output`, která na platformě `esp8266_pwm` (tato žárovka je založena na mikrokontroléru ESP8266) umožňuje PWM regulaci jednotlivých pinů. Lze tudíž ovládat jak jas jednotlivých barevných složek, tak i nastavovat různé barvené kombinace. Barevné složky z komponenty `output` byly poté přiřazeny ke komponentě `light` a její platformě `rgbw`, která umožňuje, aby se žárovka správně hlásila v Home Assistantu, a přistupovalo se k jejím barvám přes abstraktní vrstvu, a ne přímo přes PWM.

4.2 Mobilní aplikace

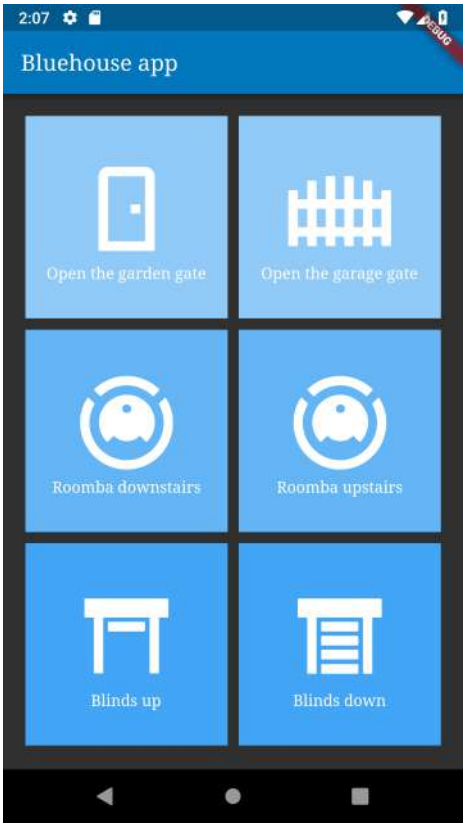
Pro implementování mobilní aplikace bylo využito platformy Flutter[9]. Byla zvolena z důvodu, že implementace jednoduchých aplikací je v této platformě velmi efektivní. Flutter totiž umožňuje takzvaný `Hot Reload`, který dovoluje spustit kód bez potřeby jej znovu kompilovat a složitě spouštět. Zařízení si rovnou načte potřebná data a lze tedy vyvíjet v reálném čase za běhu aplikace.

Aplikace využívá upravený Material Design vzhled, a `MDI` ikony, které využívá i Home Assistant. Plocha byla rozdělena do $2x3$ dlaždic, kde každá dlaždice obsahuje jedno akční tlačítko, které po kliku odešle předem definovanou MQTT zprávu do brokeru.

Dále byla implementována akce na klik, která publikuje do určeného tématu na MQTT brokeru svá data. Bylo využito knihovny `mqtt_client`[10], kde bylo potřeba implementovat vlastní rozhraní pro připojování k brokeru, odebírání témat a publikování dat do nich. Klient byl nakonfigurován tak, aby se použil k připojení se k brokeru autentizaci a SSL zabezpečení (vygenerovaný vlastní certifikát). Zabezpečení je realizováno pomocí komponenty `SecurityContext`, která je součástí `dart:io`[11]. Akce `onPressed` daného tlačítka vyvolává příkaz `broker.publish(topic, value)`, který publikuje JSON pro akci v MQTT brokeru (pro otevření žaluzií se v tématu `action/blind` publikuje JSON obsahující adresu ovladače

[8] https://github.com/ct-Open-Source/tuya-convert
[9] https://flutter.dev/
[10] https://pub.dev/packages/mqtt_client
[11] https://api.dart.dev/stable/2.8.2/dart-io/SecurityContext-class.html

a příkaz `0x2`, který koresponduje s stiskem tlačítka nahoru – sekce 2.4.3). Toto téma odebírá ESP32 bridge, který dekóduje JSON a předá příkaz příslušné žaluzii.



Obrázek 4.9: Finální aplikace pro ovládání systému.

4.3 Implementace řídicí jednotky

Do SBC Odroid HC1 (sekce 2.5.1) byla nainstalována a nakonfigurována platforma OpenMediaVault (sekce 2.5.4) a do ní byl nainstalován plugin pro Docker. Instalace sestávala ze zápisu obrazu zavaděče na SD kartu a obrazu systému na pevný disk. Následně byly nainstalovány pluginy pro Docker a OpenVPN. Byly zprovozněny a zkonfigurovány Docker konteinery pro Home Assistant, MQTT broker a Assistant Relay (příloha C). V Home Assistantu byly ESPHome zařízení detekována díky NativeAPI (sekce 2.2.4) a přidána do správné místnosti v domě. V Assistant Relay průvodci byl systém začleněn do Google Home díky konfiguraci v Google Actions Console (sekce 2.2.1). Lze tedy do Google Home zasílat libovolné příkazy přes text a pomocí REST API. V MQTT brokeru byla nakonfigurována autentizace a šifrované spojení[12].

[12] https://openest.io/en/2020/01/03/mqtts-how-to-use-mqtt-with-tls/

Kapitola 5

Nasazení systému v domácnosti a testování

Nasazení systému v domácnosti spočívalo zejména v instalaci komponent v domě a jejich vzájemné propojení.

5.1 Prostředí

Dům je samostatně stojící se zahradou. Na domě se nachází tři venkovní světla, která slouží k nasvícení vchodu do domu a chodníku kolem něj. Na zahradě se dále nachází šest venkovních světel, které nasvěcují okolí pergoly a vířivku. Pro tato světla bylo nutné přivést elektřinu a vybetonovat pro ně základnu v rámci přípravy na finální nasazení. Na pergole se nachází osvětlení, které ovládáno pomocí Sonoff Basic (obrázek 2.6). Před domem je domovní zvonek s audio telefonem a elektromagnetickým zámkem pro branku.

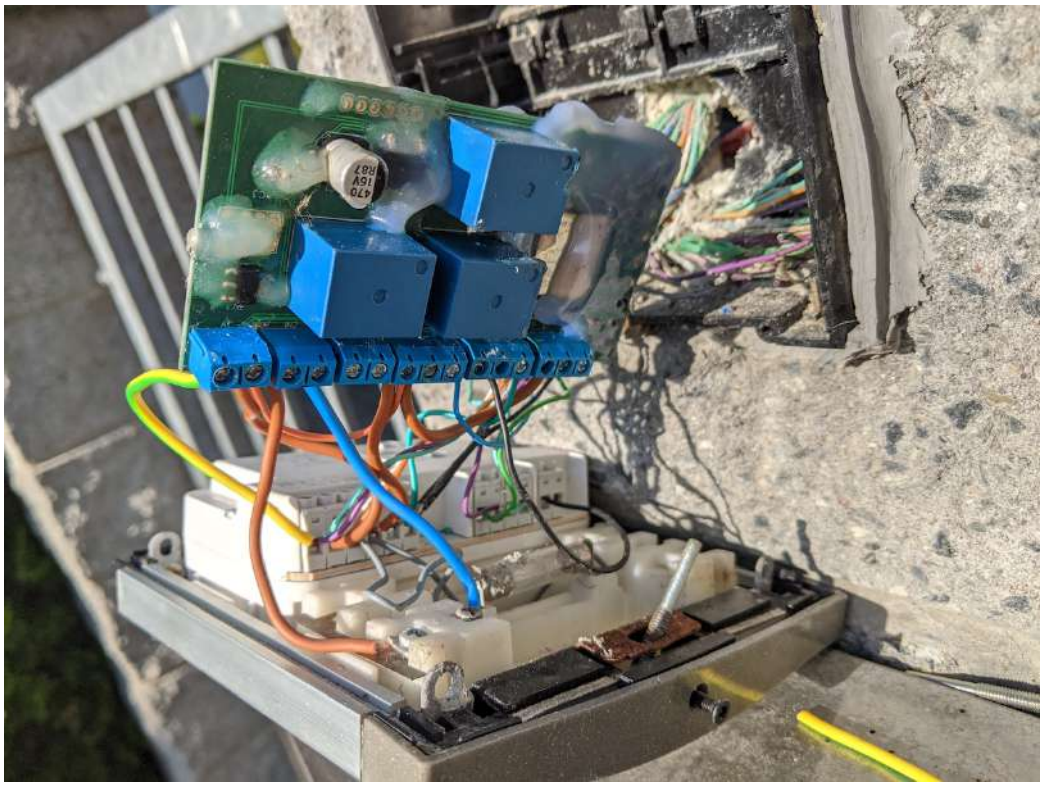
Rodinný dům je orientován třemi velkými francouzskými okny na západ, jedním oknem na jih a dvěma okny na sever. Všechna tato okna v přízemí obsahují podomítkové žaluzie s motorem a od firmy Somfy a jejich ovládání je realizováno dvojicí vícekanálových ovladačů (jedním ovladačem lze ovládat více žaluzií). Každý ovladač má k sobě přiřazeny tři žaluzie (každá na zvláštní kanál), čtvrtý kanál je neobsazen a pátý kanál (svítí všechny 4 indikační LED diody, čímž naznačuje, že jsou vybrány všechny žaluzie) je spárovaný se žaluziemi z kanálů 1–3. Mimo to jsou dané žaluzie spárovány k ESP8266 jednotce s 433 MHz anténou, kterou jsem realizoval v bakalářské práci [33]. Na střeše domu je umístěna meteostanice 1.0, která byla rovněž realizována v rámci BP. Obě zařízení dostávají vlastní vyrobené DPS místo stávající desky vlastní výroby, ESP32 mikrokontrolér a funkce navíc.

V domě se dále nachází dva vysavače iRobot Roomba 580, jeden v přízemí a jeden v patře. Každý z nich má vlastní nabíjecí základnu, do které se po dokončení vysávání vrací. Jejich ovládání probíhá standardním způsobem (stiskem tlačítka Clean), ale bylo rozšířeno o Wi-Fi funkcionalitu.

5.2 Umístění DPS do zvonku

Původní funkce zvonku, včetně audio telefonu není nijak ovlivněna. Pří zvonění dojde k detekci stisku tlačítka a tato akce je předána původnímu zvonku. Systém zvonku se postará o zvonění a zároveň ESP32 detekovaný stisk tlačítka publikuje přes MQTT tento stav do patřičného tématu, kde poté zareagují na toto akční členy (Google Cast API, RGB žárovky

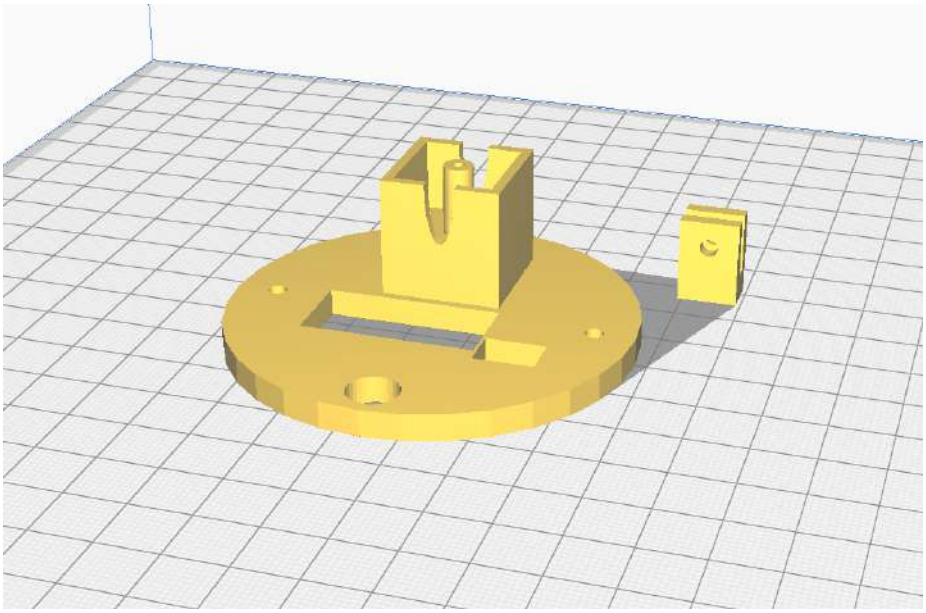
na zahradě, 433 MHz vysílač, který emuluje vnitřní zvonek). V případě, kdy na domovní jednotce někdo stiskne tlačítko pro vpuštění dovnitř, je toto provedeno původním systémem bez vlivu nového. Zároveň však se druhé relé stará o to, že přivede stejným způsobem signál do elektromagnetického zámku, jako v případě původního zvonku. Třetí rozsvěcuje podsvícení zvonku, které jde využít pro notifikaci.



Obrázek 5.1: Zaizolované DPS připojené ke starému zvonku.

5.3 Instalace meteostanice 2.0

Jelikož nové DPS k meteostanici má mírně odlišné rozmístění komponent (například mezi RJ45 konektory je větší mezera z důvodu použití širšího kabelu), tak bylo potřeba upravit 3D model původního držáku DPS z bakalářské práce a vytisknout novou verzi tohoto držáku. Byl zvolen materiál PLA, a to vzhledem k tomu, že se s ním velmi dobře s ním tiskne a také proto, že výsledný držák není vystaven povětrnostním vlivům jako dešti nebo přímému slunci, tak vydrží minimálně několik let bez problémů (původní držák byl vytištěn taky z PLA a po dvou letech působení okolí se nezdá vůbec degradován).



Obrázek 5.2: 3D model držáku DPS pro meteostanici.

Pro umístění nové DPS na meteostanici bylo potřeba vylézt na střechu a demontovat původní DPS. Bylo potřeba také vyměnit kabeláž z původního microUSB kabelu, který byl připojený na prodlužovací kabel na dvoulinkový 7 metrový kabel, který byl protažen husím krkem až k zásuvce a připojen k 12 V zdroji napětí.

Nové DPS bylo vloženo do nově vytištěného držáku (obrázek 5.2) a poté umístěno a přišroubováno do krytky. Jednotka byla poté připevněna k ramenu meteostanice (obrázek 5.3).

Obrázek 5.3: Meteostanice 2.0 na střeše.

5.4 Úprava vysavače iRobot Roomba

Pro instalaci DPS do vysavače stačí pouze sundat vrchní kryt vysavače, pod kterým se ukrývá MiniDIN konektor a do něj DPS zasunout (obrázek 5.4). Po připojení DPS začne být ESP32 napájeno z vysavače a komunikuje s ním. Bohužel, pokud by se měl vracet zpět kryt na původní místo, je potřeba DPS odpojit nebo v krytu udělat díru v místě konektoru. DPS lze ale i vložit do prostoru uvnitř vysavače a propojit desku a konektor vodiči, tím pádem lze kryt bez problémů nasadit.

Obrázek 5.4: iRobot Roomba s nainstalovanou DPS.

5.5 Umístění ESP32 bridge v domě

Deska brány byla umístěna do boxu, do kterého byly umístěny dvě antény, jedna pro 2.4 GHz frekvence a druhá pro 433 MHz. Dále byla připojena LED na boxu do svorkovnice na DPS a napájecí souosý konektor (5,5 mm/2,5 mm) byl připojen k napájecí svorkovnici. Do svorkovnice prvního relé byly připojeny vodiče ovladače garážových vrat a to tak, že mezi tlačítkem, který bránu aktivuje byly přidány vodiče. Jelikož celý ovladač funguje na dvou 3 V bateriích zapojených sériově (to znamená napětí 6 V), tak je tento ovladač napájen přímo z 5 V zdroje bridge. Box s komponentami umístěn do kuchyně poblíž Google Wi-Fi routeru.

Obrázek 5.5: Finální provedení ESP32 bridge.

5.6 Zprovoznění systému jako celek

Zařízením byla po připojení do Wi-Fi nastavena v routeru DHCP rezervace, aby se IP zařízení neměnila. Následně po připojení k Home Assistantu proběhla automatická detekce všech ESPHome zařízení a ty byly přidány do místností. Automatizace venkovního osvětlení byla nastavena v Home Assistantu na základě údajů ze senzoru osvětlení a to tak, že pokud klesne osvětlení pod určitou úroveň, tak se provede rozsvícení světel. Pokud senzor údaje nedodává, tak se světlo ovládá podle času západu slunce z internetu. BLE klíčenky byly dány členům domácnosti, a na základě jejich přítomnosti jsou ovládány vysavače, které v případě, že není nikdo, doma začnou vysávat ve stanovený čas.

Spustění vysávání v domě je podmíněno denní dobou a přítomností lidí v domě. Pokud je všední den dopoledne mezi 10–12 hodinou a zároveň nikdo není doma, tak se vysavače spustí a provedou vysávání a poté se vrátí k nabíjecí stanici. Pokud se někdo domů vrátí v průběhu vysávání, tak vysavače zajedou do dokovací stanice. Nevysává se každý den, ale pouze v úterý, čtvrtek a sobotu. Jelikož v sobotu se v domě většinou někdo nachází, tak vysavače ignorují přitomnost obyvatel a vysávají v dobu, kdy se obvykle uklízí v tento den.

Pro zprovoznění Google Assistant Relay (sekce 3.4.3) bylo potřeba vygenerovat si přihlašovací údaje (OAuth client ID) a tyto údaje poskytnout platformě během počáteční konfigurace. Dále byly v konfiguraci nastaveny tzv. `Quiet Hours`, kdy nedochází k hlasitému vysílání zpráv typu, že právě někdo zvoní nebo že vysavač ukončil úklid.

Aplikace na automatické ovládání žaluzií z bakalářské práce byla spuštěna ve vlastním kontejneru a byla do ní přidána funkce napojení na Home Assistant. Funguje tak, že pokud je v Home Assistantu spuštěný přepínač automatizace, tak aplikace vykonává ovládání

žaluzií. Pokud je přepínač vypnutý, tak aplikace čeká na povolující zprávu a údaje ze senzorů neodebírá od brokeru na daných tématech.

Mobilní aplikace byla nainstalována na telefony členů domácnosti a předvedena funkčnost. Aktuálně funguje z bezpečnostních důvodů pouze v interní síti. Na zkoušku však broker otevřen do sítě internet pod jiným portem a se zapnutou autentizací.

5.7 Testování jednotlivých komponent

V této sekci je popsána reakce jednotlivých komponent systému na nestandardní situace. Testovat lze reakci na poškozené pakety, nevalidní REST API či MQTT zprávy, nebo syntakticky špatný JSON.

Poškozené pakety

Pro otestování reakce na zvláštní a poškozené pakety bylo využito nástroje `sendip`[1], který umožňuje manipulovat různými způsoby s paketem, který se odesílá. Tímto způsobem byly různě pozměněné UDP, TCP a ICMP pakety směrovány jak na ESP32 zařízení, tak na řidicí jednotku a další členy systému. Byly vybrány náhodné pakety určené pro určitý ESP32 modul, avšak zpráva byla doručena jinému zařízení. ESP32 špatné pakety ignorovalo a nebylo zaznamenáno, že by tyto pakety reálně něco způsobily. Obdobně se chovaly pakety určené jiným zařízením.

Nevalidní REST API volání

Pro otestování funkčnosti REST API volání jsem testoval HTTP požadavky na Assistant Relay podle doporučených testovacích strategií[2]. Správny HTTP požadavek je potom zpracován na příkazy pro Google Home. Assistant Relay podporuje pouze `POST` HTTP požadavky, které ve svém těle obsahují validní JSON, kterému Assistant Relay rozumí.

Pro otestování HTTP požadavků jsem si napsal několik testů za pomocí nástroje `pyhttptest`[3]. Zkoušel jsem zaměnit `POST` za `GET` či `PUT` nebo překlepy v HTTP požadavku (POS, PST, POSTE, ...). V případě GET vracelo API webovou stránku odpovídající URL v požadavku (při `GET` na stejnou URL, jakou má `POST` požadavek je vrácena webová stránka). V ostatních případech vracel server chybu `HTTP/1.1 400 Bad Request`, což je předpokládané chování. Dále jsem zkoušel zaměnit Content-Type u `POST` zprávy z `application/json` na `text/plain`, toto bohužel neuměl nástroj uskutečnit, tak bylo potřebovat toto otestovat nástrojem `reqbin`[4]. U tohoto testu se systém choval tak, že pokud Content-Type nebyl `application/json`, tak server vracel znovu `HTTP/1.1 400 Bad Request`.

Nevalidní JSON

ESP32 modulům a Assistant Relay serveru byly odeslány přes MQTT (a REST API v případě Assitant Relay) nevalidní JSON zprávy. Tyto zprávy byly nevalidní tím, že byly nedostatečně nebo příliš uzávorkované, chyběly uvozovky, nebo třeba jednotlivé atributy nebyly

[1] http://www.earth.li/projectpurple/progs/sendip.html
[2] https://www.sisense.com/blog/rest-api-testing-strategy-what-exactly-should-you-test/
[3] https://hackernoon.com/testing-rest-apis-easily-in-python-with-pyhttptest-1d2x328d
[4] https://reqbin.com/

oddělený čárkami. Samotná zpráva jako taková projde systémem úplně bez problémů, jelikož MQTT broker neprovádí kontrolu JSON syntaxe (data jsou doručena jako plain text). Pokud syntaxe není validní, tak je na koncovém zařízení, aby zprávu zpracovalo nebo zahodilo. ESP32 zprávu, která není validní, nedekóduje a následně zahodí. Obdobně se chová i Assistant Relay, který taky ignoruje nevalidní JSON zprávy. Pokud dojde syntakticky validní JSON zpráva, ale není k ní přiřazena žádná akce, tak s ní systém nepočítá a zahozena podobně, jako v případě syntaktické chyby JSONu.

Odpojený nebo nefunční senzor

Při odpojení I^2C senzoru z ESPHome platformy za chodu přestane ESPHome zobrazovat data z tohoto senzoru a v Home Assistantu se přestane zobrazovat hodnota. U senzorů I^2C, které byly implementovány vlastními silami (například komponenta pro senzor VEML6070), knihovna příslušného senzoru vrací často vlastní chybový kód nebo hodnotu, která není reálná. V případě senzoru UV záření to byla hodnota 65535 a jelikož by tato hodnota narušovala statistiku (senzor dokáže naměřit až desítky tisíc lx), tak bylo toto ošetřeno tím, že při této hodnotě senzor nepublikuje data (viz kód v sekci 4.1.2).

5.8 Testování celkového systému

Systém je modulární, takže výpadek určité komponenty nezpůsobí celkový výpadek systému. Pokud dojde k výpadku Wi-Fi připojení, tak chytrý zvonek bude fungovat stále alespoň v off-line režimu, data ze senzorů jsou vyčítána a po obnovení připojení je jejích stav doručen Home Assistantu. V případě výpadku senzoru nejsou odesílány nesmyslné hodnoty, ale pouze informace, že stav není dostupný, to umožňuje zabránit chybným akcím automatizace.

Všechny komponenty systému zároveň logují svůj stav do debuggovacích témat v MQTT brokeru, odkud tato data odebírá logger v backendu, který je třídí a zapisuje do souborů v reálném čase. Lze mít tedy přehled o všech komponentách v daný čas. Podobným způsobem loguje stav i Home Assistant v přehledném GUI, avšak méně dopodrobna.

Kapitola 6

Závěr

V práci byly nastudovány a sumarizovány základní informace o IoT systémech. Byly popsány IoT systémy používané v inteligentních domácnostech i komunikační protokoly, které takové systémy využívají. V psáci rovněž byly sumarizovány informace o mikrokontroléru ESP8266, ESP32 a komerčních koncových zařízení využívajících tyto mikrokontroléry. Také byla popsána IoT brána a uvedeny příklady použitelného hardwarového řešení pro takovou bránu.

V návrhu byl popsán IoT systém, který lze zrealizovat doma. Koncová zařízení, automatizační pravidla, senzory i aktuátory. Byl též popsán návrh komponent IoT brány. Závěrem návrhu byla popsána mobilní aplikace včetně návrhu uživatelského rozhraní.

V rámci implementace byly vyrobeny DPS, osazeny součástkami, implementován jejich firmware a nahrán do zařízení. Dále byla implementována mobilní aplikace ve Flutteru, která slouží k ovládání akčních členů IoT systému.

Závěřem práce byla jednotlivá koncová zařízení nasazena v reálném prostředí a celý systém byl otestován. Testovány byly jak jednotlivé kompomenty systému, tak i celý systém. V rámci testování byla použita celá řada testovacích nástrojů a vyzkoušeny reakce na různé stavy.

Literatura

[1] *About / openmediavault* [https://www.openmediavault.org/about.html]. [Online; navštíveno 2.06.2020].

[2] *Automations and Templates*. [Online; navštíveno 12.11.2019]. Dostupné z: https://esphome.io/guides/automations.html.

[3] *Create and manage speaker groups*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://support.google.com/googlenest/answer/7174267.

[4] *Fulfillment and authentication | Actions on Google Smart Home*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://developers.google.com/assistant/smarthome/concepts/fulfillment-authentication.

[5] *Getting Started with ESPHome through Hass.io*. [Online; navštíveno 12.11.2019]. Dostupné z: https://esphome.io/guides/getting_started_hassio.html.

[6] *Google Assistant is ready and built-in to specific speakers*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://assistant.google.com/platforms/speakers/.

[7] *Google Assistant on Android TV*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://assistant.google.com/platforms/tv/.

[8] *HomeKit - Všechny doplňky*. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.apple.com/cz/shop/accessories/all-accessories/homekit.

[9] *HomePod*. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.apple.com/homepod/.

[10] *IOS - Home*. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.apple.com/ios/home/.

[11] *Link your voice to your Google Assistant device with Voice Match - Android*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://support.google.com/assistant/answer/9071681.

[12] *Made For Google*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://get.google.com/madefor/.

[13] *Nastavení a používání aplikace Domácnost*. [Online; navštíveno 12.11.2019]. Dostupné z: https://support.apple.com/cs-cz/HT204893.

[14] *Sensor Component Filters*. [Online; navštíveno 12.11.2019]. Dostupné z: https://esphome.io/components/sensor/index.html#sensor-filters.

[15] *Smart Home Device Types | Actions on Google Smart Home*. Google. [Online; navštíveno 12.11.2019]. Dostupné z: https://developers.google.com/assistant/smarthome/guides.

[16] *What is ESPHome*. [Online; navštíveno 12.11.2019]. Dostupné z: https://esphome.io/.

[17] *What's a Linux container?* [Online; navštíveno 14.05.2020]. Dostupné z: https://www.redhat.com/en/topics/containers/whats-a-linux-container.

[18] Espressif Implements CEVA Bluetooth in ESP32 IoT Chip. *Wireless News*. Oct 28 2016. Copyright - Copyright 2016 Close-Up Media, Inc. All Rights Reserved; Last updated - 2016-10-28. Dostupné z: https://search.proquest.com/docview/1833037620?accountid=17115.

[19] *Všechno o Amazon Echo a Alexa*. Sep 2019. [Online; navštíveno 12.11.2019]. Dostupné z: https://365tipu.cz/vsechno-o-amazon-echo-a-alexa/.

[20] Assistant, H. *Integrations*. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.home-assistant.io/integrations/.

[21] Basics, C. *Basics of UART Communication*. [Online; navštíveno 10.01.2020]. Dostupné z: http://www.circuitbasics.com/basics-uart-communication/.

[22] CZ.NIC. *Projekt Turris*. [Online; navštíveno 13.05.2020]. Dostupné z: https://project.turris.cz/cs/.

[23] Dočekal, D. *TIP#883: Co je to Bluetooth LE, BLE, Bluetooth Low Energy, Bluetooth Smart, Bluetooth 5)? – @365tipu*. Září 2017. [Online; navštíveno 13.01.2020]. Dostupné z: https://365tipu.cz/2017/09/06/tip883-co-je-to-bluetooth-le-ble-bluetooth-low-energy-bluetooth-smart-bluetooth-5/.

[24] Gartenberg, C. *Google just renamed its smart home brand to Google Nest*. The Verge, May 2019. [Online; navštíveno 13.05.2020]. Dostupné z: https://www.theverge.com/2019/5/7/18531532/google-nest-rename-smart-home-rebrand-io-2019.

[25] Gebhart, A. *What is Google Assistant?* CNET, Jun 2019. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.cnet.com/news/what-is-google-assistant/.

[26] Gebhart, A. *Google Home Hub review: Google's smart display is still the one to beat - CNET*. Únor 2020. [Online; navštíveno 13.05.2020]. Dostupné z: https://www.cnet.com/reviews/google-nest-home-hub-with-google-assistant-the-best-smart-display-review/.

[27] Goldmann, I. T. *Prvky IoT systému*. [Online; navštíveno 2.06.2020]. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/cwk.php.cs?title=Main_Page&src=Prednaska_2_toi_goldmann_r1a.pdf&ns=TOI&action=download&csid=719826&id=13614.

[28] Hildenbrand, J. *Home Assistant makes your smart devices work together the way you imagined*. Android Central, Apr 2018. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.androidcentral.com/whats-home-assistant-and-why-should-home-automation-enthusiasts-consider-it.

[29] Hillar, G. C. *MQTT Essentials-A Lightweight IoT Protocol*. Packt Publishing Ltd, 2017.

[30] ITEAD. *Sonoff - ITEAD documentation*. [Online; navštíveno 17.01.2020]. Dostupné z: https://www.itead.cc/wiki/Sonoff.

[31] Kolban, N. *Kolban's Book on ESP8266*. 2015. 1–467 s.

[32] Kolban, N. *Kolban's Book on ESP32*. USA: Leanpub, 2017.

[33] Kovařík, V. *Autonomní řízení žaluzií*. Brno, CZ, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Dostupné z: https://www.fit.vut.cz/study/thesis/21079/.

[34] Kozuch, K. *20 Alexa speakers, ranked from best to worst*. Tom's Guide, Dec 2019. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.tomsguide.com/us/pictures-story/1595-best-alexa-speakers.html.

[35] Labiod, H. *Wi-Fi, Bluetooth, ZigBee and WiMAX*. Dordrecht: Springer, 2007. ISBN 978-1-4020-5396-2.

[36] Lairdtech. *3 Factors that Limit Range in RF Applications - Laird Technologies Wireless Connectivity Blog*. [Online; navštíveno 10.01.2020]. Dostupné z: http://www.summitdata.com/blog/3-factors-limit-range-rf-applications/.

[37] Lardinois, F. *Google Home will go on sale today for $129, shipping November 4*. TechCrunch, Oct 2016. [Online; navštíveno 12.11.2019]. Dostupné z: https://techcrunch.com/2016/10/04/say-hello-to-google-home/.

[38] Leens, F. An introduction to I^2C and SPI protocols. *IEEE Instrumentation & Measurement Magazine*. IEEE. 2009, roč. 12, č. 1, s. 8–13.

[39] Leger, H. S. *Amazon Echo Plus review*. TechRadar, Nov 2019. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.techradar.com/reviews/amazon-echo-plus-review.

[40] Lorenzetti, L. *Amazon launches new voice-activated speaker named Alexa*. Fortune, Nov 2014. [Online; navštíveno 12.11.2019]. Dostupné z: https://fortune.com/2014/11/06/forget-siri-amazon-now-brings-you-alexa/.

[41] Lubo. *Senzor BME280 - měření teploty, relativní vlhkosti a barometrického tlaku*. Arduino návody, Jan 2017. [Online; navštíveno 10.01.2020]. Dostupné z: https://navody.arduino-shop.cz/navody-k-produktum/senzor-bme280-mereni-teploty-relativni-vlhkosti-a-barometrickeho-tlaku.html.

[42] Malý, M. *REST: architektura pro webové API - Zdroják*. [Online; navštíveno 12.01.2020]. Dostupné z: https://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/.

[43] Natale, N. *Which Amazon Echo speaker should you buy? We find out*. Jul 2018. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.techadvisor.co.uk/review/digital-home/amazon-echo-vs-echo-2-comparison-3664841/.

[44] Patel, K. K., Patoliya, J. a Patel, H. Low cost home automation with ESP8266 and lightweight protocol MQTT. *Transactions on Engineering and Sciences*. 2015, roč. 3, č. 6, s. 2347–1875.

[45] Prospero, M. *Amazon Echo Spot Review: Alexa Just Killed Your Alarm Clock*. Tom's Guide, Jun 2019. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.tomsguide.com/us/amazon-echo-spot,review-4984.html.

[46] Pushstack. *Somfy RTS Protocol*. [Online; navštíveno 10.05.2020]. Dostupné z: http://pushstack.wordpress.com/somfy-rts-protocol/.

[47] Ravenscraft, E. *How to Change Google Assistant to Typing Instead of Voice By Default*. How To Geek, Jun 2017. [Online; navštíveno 12.11.2019]. Dostupné z: https://www.howtogeek.com/309779/how-to-change-google-assistant-to-typing-instead-of-voice-by-default/.

[48] Richardson, M. a Wallace, S. *Getting started with Raspberry Pi*. "O'Reilly Media, Inc.", 2012.

[49] Smith, C. *Brand new Raspberry Pi 4 supports 4GB of RAM and 4K displays – BGR*. Červen 2019. [Online; navštíveno 13.05.2020]. Dostupné z: https://bgr.com/2019/06/24/raspberry-pi-4-release-date-price-and-specs-announced/.

[50] Sparkfun. *Serial Communication - learn.sparkfun.com*. [Online; navštíveno 10.01.2020]. Dostupné z: https://learn.sparkfun.com/tutorials/serial-communication/.

[51] ČTÚ. *Využívání vymezených rádiových kmitočtů | Český telekomunikační úřad*. [Online; navštíveno 12.01.2020]. Dostupné z: https://www.ctu.cz/vyuzivani-vymezenych-radiovych-kmitoctu.

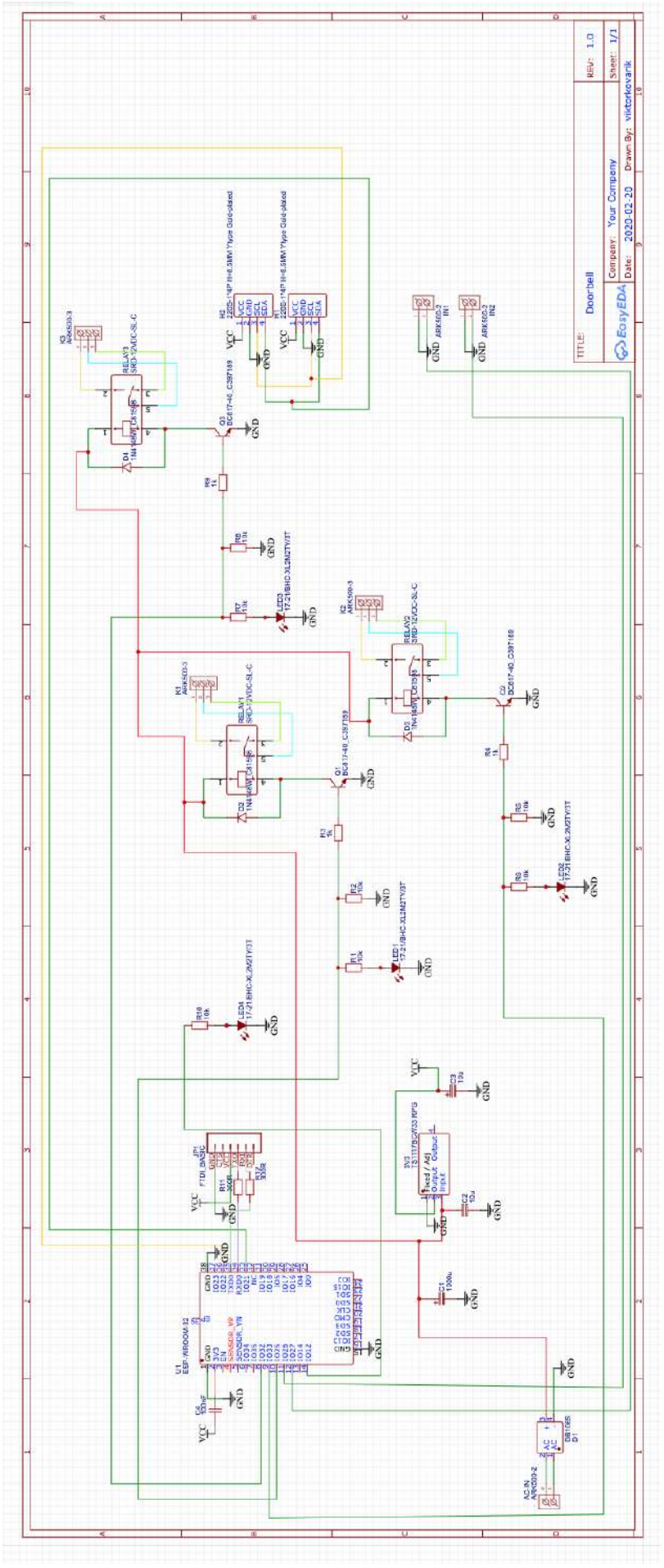
Příloha A

Obsah přiloženého CD

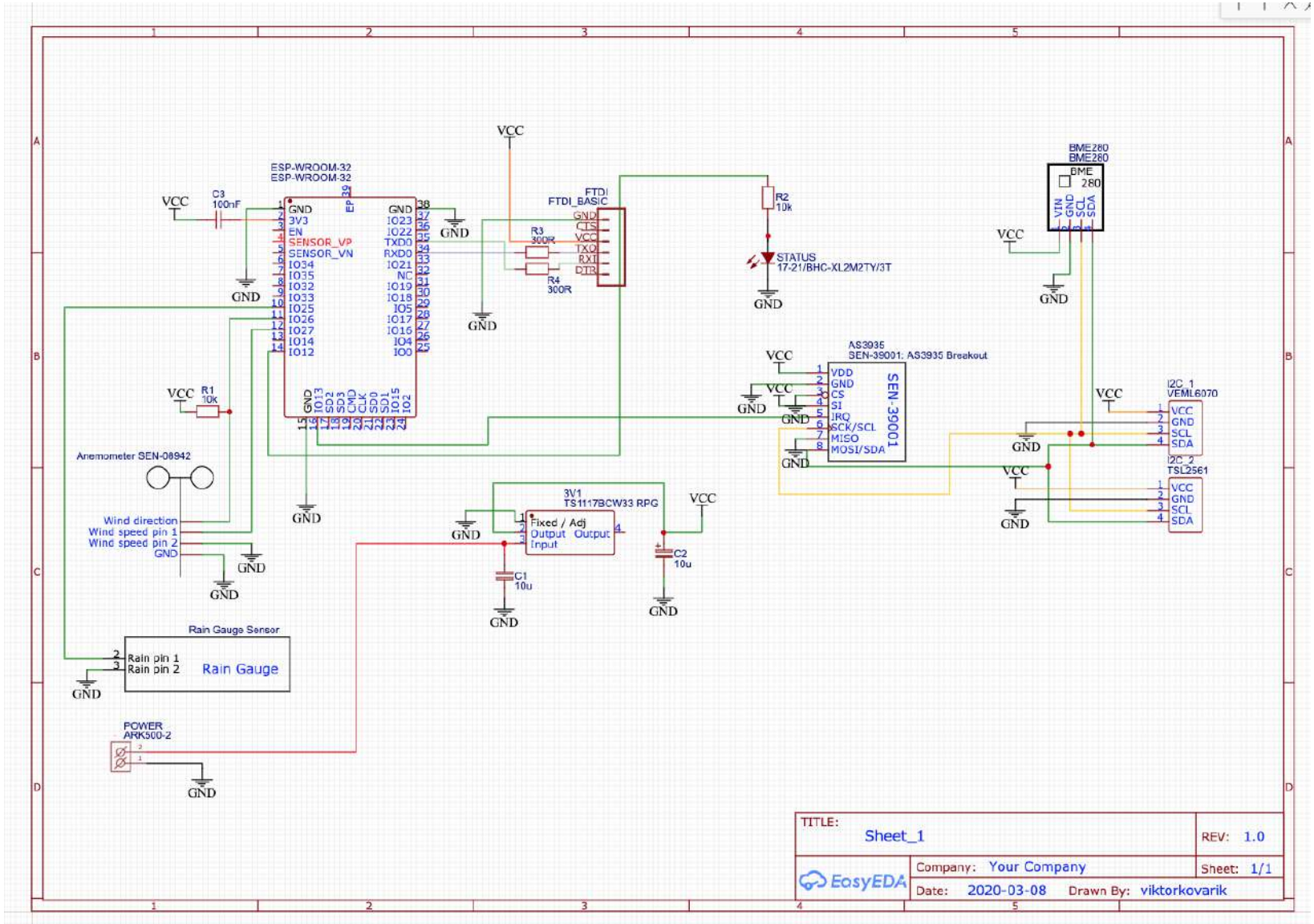
```
|---- README.TXT          - podrobnější informace o souborech a instalaci
|---- docs
|   |---- other
|   |---- pcb_layouts     - schémata zapojení DPS
|   |---- diagrams        - diagramy3
|   +---- thesis          - práce a její zdrojové soubory
|       +---- src
|---- photos
|   |---- PCBs            - fotografie DPS využitých v práci
|   +---- final           - fotografie finálních zařízení
|---- videos              - demonstrační videa
+---- src
    |---- mcu             - zdrojové kódy pro mikrokontroléry
    |   |---- smart_doorbell        - firmware pro chytrý zvonek
    |   |---- weatherstation_2.0    - firmware pro meteostanice 2.0
    |   |---- roomba                - firmware pro řídicí jednotku iRobot Roomba
    |   |---- esp32_bridge          - firmware pro ESP32 bridge
    |   +---- light_bulbs           - firmware pro chytré žárovky FCMILA
    |---- mobile_app      - zdrojové kódy automatizačního programu
    |---- 3D_tisk         - 3D modely využité při realizaci práce
    +---- other
        |---- diagrams            - zdrojové soubory k diagramům
        |---- pcb_layouts         - zdrojové soubory k DPS schématům
        +---- label               - etiketa odevzdaného disku
```

Příloha B

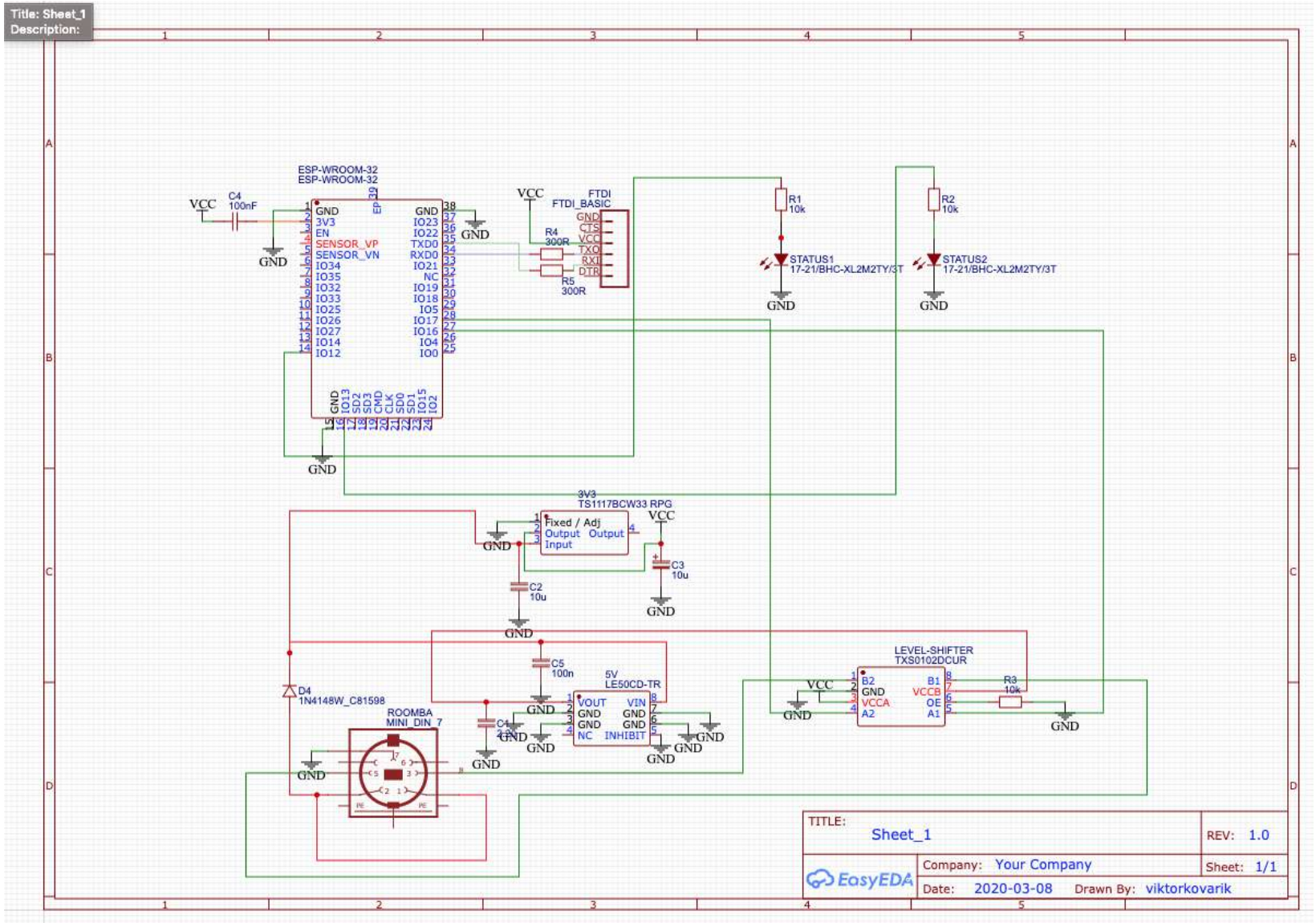
Schémata obvodového zapojení



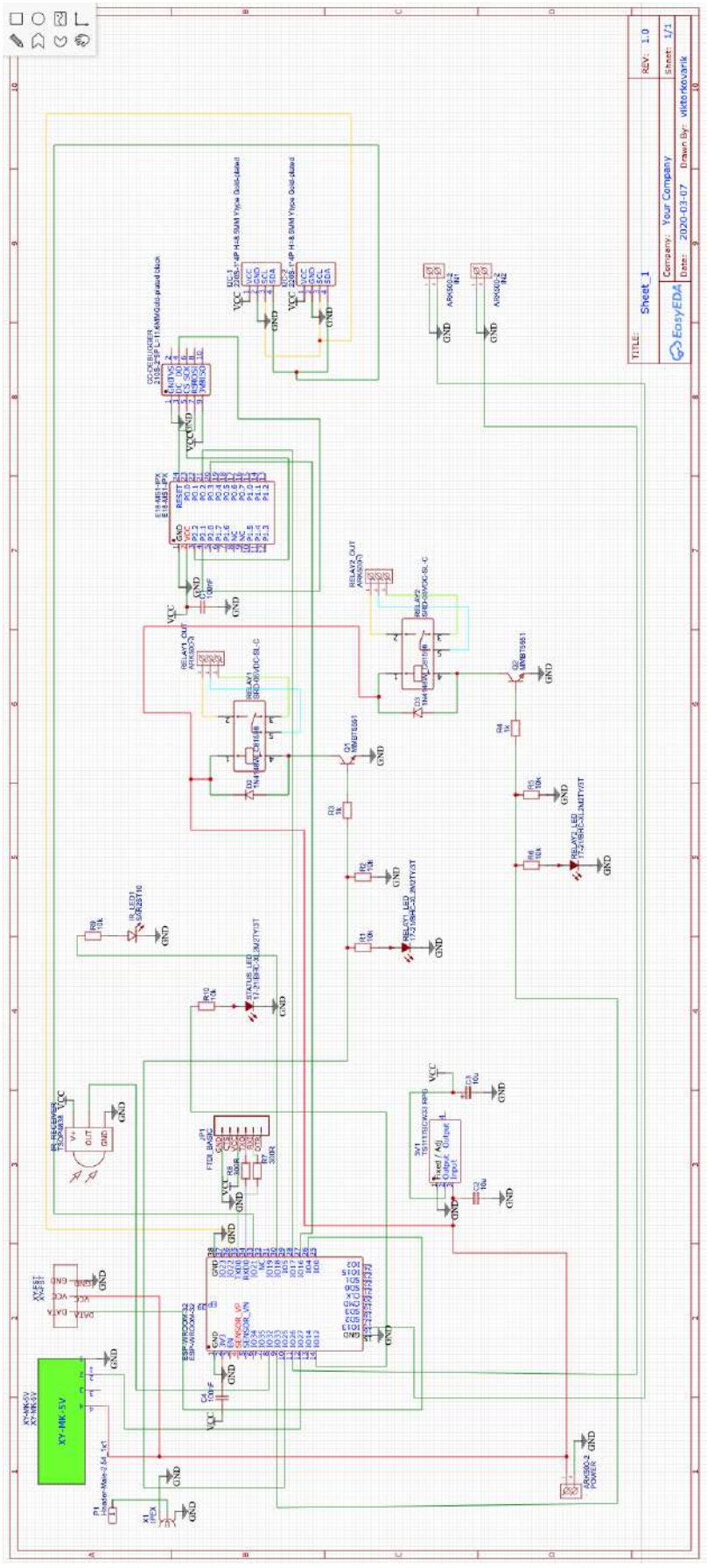
Obrázek B.1: Obvodové zapojení chytrého zvoku



Obrázek B.2: Obvodové zapojení meteostatnice



Obrázek B.3: Obvodové zapojení Wi-Fi modulu k iRobot Roomba



Obrázek B.4: Obvodové zapojení ESP32 bridge

Příloha C

Ukázky konfigurace

Ukázka konfigurace chytrého zvonku

```
switch:
  - platform: gpio
    pin: 25
    id: fence_relay
    restore_mode: ALWAYS_OFF
  - platform: template
    name: "Fence door"
    icon: "mdi:gate"
    turn_on_action:
    - switch.turn_on: fence_relay
    - delay: 3000ms
    - switch.turn_off: fence_relay
    turn_off_action:
    - switch.turn_off: fence_relay
binary_sensor:
  - platform: gpio
    pin:
      number: 26
      mode: INPUT_PULLUP
      inverted: True
    id: doorbell
    name: "Doorbell"
    filters:
      - delayed_on: 10ms
      - delayed_off: 100ms
    on_press:
      then:
        - switch.turn_on: bell_relay
    on_release:
      then:
        - switch.turn_off: bell_relay
```

Konfigurační příkazy Docker kontejnerů

Home Assistant

```
docker run --init -d --name="home-assistant" \
-e "TZ=Europe/Prague" -v /root/docker/home-assistant:/config \
--net=host \
homeassistant/home-assistant:stable
```

Mosquitto broker

```
docker run -it -p 1883:1883 -p 9001:9001 \
-v /root/docker/mosquitto/mosquitto.conf:/mosquitto/config/mosquitto.conf \
-v /root/docker/mosquitto/data:/mosquitto/data \
-v /root/docker/mosquitto/log:/mosquitto/log eclipse-mosquitto
```

Assistant Relay

```
docker run -d --name assistant_relay \
--network host \
-v /root/docker/assistant-relay/config.json:/data/config.json:rw \
-v /root/docker/assistant-relay/audio-responses:/data/audio-responses:rw \
apipa169/armv7-hassio-assistant_relay:latest
```

ESPHome

```
docker run --rm -v /root/docker/esphome:/config \
-it esphome/esphome weatherstation.yaml wizard
```