ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Lhotecký** Jméno: **Vlastimil** Osobní číslo: **406989**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra měření**

Studijní program: **Inteligentní budovy**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Komunikační brána pro smart home

Název diplomové práce anglicky:

Smart home communication gateway

Pokyny pro vypracování:

Navrhněte a realizujte univerzální komunikační rozhraní pro systém řízení chytré domácnosti (smart home). Zařízení realizujte tak, aby umožňovalo připojení pomocí bezdrátových standardů Wi-Fi, Zigbee, Bluetooth, případně dalších. S ústřednou komunikujte pomocí web rozhraní, příp. mobilní aplikací. Monitorujte kvalitu ovzduší, intenzitu osvětlení a další klíčové parametry. Zajistěte kompatibilitu brány s některým z otevřených systémů smart home ústředen (např. Home Assistant, OpenHab). Otestujte funkčnost zařízení a dosažené prametry porovnejte s komerčně dostupnými alternativami.

Seznam doporučené literatury:

[1] Home Automation - A Smart Home Guide: The Beginner's Manual Including Google Home, Echo Dot and Amazon Alexa. Easy Instructions, Directions and Commands ... and Home Automation Guide Series Book 1), ASIN B01MY2B5R2
[2] Mach's einfach: Erste Schritte mit Smart-Home-Programmierung: Einstieg in die Hausautomation mit Node-RED, ISBN 978-3645606516

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Vladimír Janíček, Ph.D., katedra mikroelektroniky FEL

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

Datum zadání diplomové práce: **05.02.2020** Termín odevzdání diplomové práce: **22.05.2020**

Platnost zadání diplomové práce:
do konce letního semestru 2020/2021

Ing. Vladimír Janíček, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

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

.	
Datum převzetí zadání	Podpis studenta

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA ELEKTROTECHNICKÁ

INTELIGENTNÍ BUDOVY

Diplomová práce

Komunikační brána pro Smart Home

Bc. Vlastimil Lhotecký

Vedoucí práce: Ing. Vladimír Janíček, Ph.D.

5. ledna 2021

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce Ing. Vladimíru Janíčkovi, Ph.D. za trpělivost a cenné rady, které mi v průběhu zpracování diplomové práce poskytl. Dále také své rodině a přítelkyni za podporu a zajímavé připomínky k tématu.

Odkaz na tuto práci

Lhotecký, Vlastimil. *Komunikační brána pro Smart Home*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta elektrotechnická, 2021.

Prohlášení

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

V Praze dne 5. ledna 2021 .....

Vlastimil Lhotecký

Anotace

V této práci se zabývám návrhem a realizací univerzální komunikační brány, která by mohla sloužit jako ústředna chytré domácnosti umožňující integrovat co nejvíce běžně dostupných Smart Home produktů bez ohledu na výrobce či produktovou řadu. Brána bude založena na open-source projektech (GNU/Linux, Home Assistant, NodeRED aj.) a koncipována jako kompletní produkt připravený k nasazení. S koncovými zařízeními bude možné komunikovat pomocí technologií Bluetooth, Zigbee, Wi-Fi, RF 433MHz a GSM.

Kromě řady bezdrátových komunikačních rozhraní obsahuje brána i několik senzorů pro měření nejčastěji požadovaných parametrů prostředí. Pro komunikaci s domácí sítí a potažmo internetem je použito Ethernetové rozhraní, které zároveň zajišťuje napájení brány. Přenos dat je realizován protokolem MQTT a konfigurace pomocí webového rozhraní.

Během práce jsem se seznámil s konceptem Smart Home a technologiemi používanými v této oblasti, na základě poznatků navrhl, realizoval a otestoval zařízení po HW i SW stránce a porovnal výsledek s komerčními alternativami dostupnými na trhu.

Klíčová slova: chytrá domácnost, domovní automatizace, internet věcí, open-source, komunikační brána, ústředna, MQTT, Wi-Fi, Zigbee, Bluetooth, GSM, RF 433 MHz, GNU/Linux, Home Assistant, Node-RED, OpenHAB, Raspberry Pi

Abstract

The aim of this thesis is to design and create a universal communication gateway that could serve as a Smart Home hub, allowing integration of various commonly available Smart Home products regardless of their manufacturer or product line. The gateway will be based on open-source projects (GNU/Linux, Home Assistant, NodeRED etc.) and is intended to be a complete product ready for deployment. End devices will be able to connect to the gateway using Bluetooth, Zigbee, Wi-Fi, RF 433Mhz and GSM.

Besides the wireless communication interfaces, the gateway includes a few sensors for measuring the most commonly requested environmental parameters. An Ethernet interface is used for local network and internet connection as well as to provide the gateway with power. Data transfer is realized using the MQTT protocol, while a web interface is used for configuration.

Over the course of this thesis, I have familiarized myself with the concept of Smart Home and technologies currently used in this area. Based on my findings, I have designed and implement the gateway from both HW and SW standpoint and compared my end result with alternatives available on the market.

Keywords: smart home, home automation, internet of things, open-source, communication gateway, hub, MQTT, Wi-Fi, Zigbee, Bluetooth, GSM, RF 433 MHz, GNU/Linux, Home Assistant, Node-RED, OpenHAB, Raspberry Pi

Obsah

Seznam obrázků

Všechny obrázky použité v této práci jsou označeny jako dostupné k volnému užití.

Úvod

Současná situace na trhu Smart Home

Inteligentní domácnost v průběhu posledních let přestala být chápána jen jako komfort nad rámec základních funkcí budovy. Vzhledem k množství technologií, které obsahují moderní novostavby či zrekonstruované domy a byty má dobře navržené centrální řízení a automatizace potenciál šetřit každý den obyvatelům budovy čas i peníze. Reflektují to i výrobci spotřební elektroniky, kteří v návaznosti na popularizaci konceptů IoT (internet of Things) a rozvoj cloudových služeb uvedli na trh mnoho výrobků realizujících jejich vizi chytré (Smart Home) domácnosti.

Dosud měl zájemce o chytrou domácnost dvě hlavní možnosti: Řešení v podobě profesionálního systému na míru danému objektu, včetně značné finanční investice, která se k těmto systémům váže, nebo systém vytvořený svépomocí, který k instalaci a případnému odstraňování poruch naopak vyžaduje znalosti z oblasti elektroniky, potažmo IT. Nyní má možnost třetí – nakoupit výše zmíněné výrobky stejným způsobem jako jinou spotřební elektroniku a z nich bez nutnosti odborných znalostí poskládat systém podle svých požadavků.

Namísto zajištění cenově dostupného, standardizovaného propojení technologií v domácnosti došlo však spíše ke vzniku mnoha navzájem nekompatibilních „ekosystémů“. Každý výrobce má svou řadu Smart Home produktů a každý výrobce také samozřejmě vyniká v jiné oblasti (osvětlení, multimédia, řízení prostředí aj.).

Jedním z nabízených produktů zpravidla bývá brána, která umožňuje propojení systému s okolím. V případě souběžného provozu více ekosystémů pak každý z nich skrz příslušnou bránu komunikuje s ústřednou (hub). Ta mimo jiné zajišťuje jejich interakci, poskytuje uživateli komplexní přehled o domácnosti, a umožňuje nastavovat automatizační pravidla. Ústředna se s branami propojuje typicky pomocí IP sítě a může být realizována fyzickým zařízením (např. Amazon Alexa), nebo cloudovou službou (např. ifttt.com). Oba typy vyžadují pro některé funkce (u cloudových služeb pro všechny) připojení k internetu.

Existují iniciativy, které se snaží spolupráci Smart Home systémů sjednotit a také umožnit standardizovanou komunikaci s doposud používanými technologiemi budov (např. regulátory vytápění), příkladem může být Connected Home over IP. Jejich vývoj ale stále probíhá a zavedení není v dohledné době očekáváno. Dále existují alternativy v podobě open-source projektů, které se snaží o stejnou věc, ale naopak tím, že budou podporovat co nejvíce různých protokolů a výrobků dostupných v současnosti.

Účel brány

Cílem této práce je s využitím zmíněných open-source projektů navrhnout a vytvořit zařízení, kombinující ústřednu a univerzální bránu kompatibilní s co nejvíce Smart Home produkty bez ohledu na výrobce. Zařízení bude jednak těžit z výhod otevřených technologií (rychlý vývoj a implementace novinek, široká kompatibilita a v podstatě neomezená rozšiřitelnost) a zároveň reprezentovat produkt připravený k použití po stránce HW (vyřešená konstrukce) i SW (k oživení stačí instalační balíček).

Brána tak dává uživateli možnost kombinovat prvky různých Smart Home ekosystémů bez dalších mezičlánků a tím snižuje složitost a pořizovací cenu systému. Brána bude fungovat nezávisle na připojení k internetu, žádná data o domácnosti tedy neopustí lokální síť, pokud si to uživatel nepřeje.

Součástí brány je i několik senzorů nejběžnějších parametrů prostředí, konkrétně senzor teploty, vlhkosti vzduchu, koncentrace CO_2 a intenzity osvětlení. V místnosti, kde bude ústředna umístěna, tak již není třeba instalovat další koncová zařízení s těmito funkcemi.

Rešerše

Vznik a vývoj konceptu Smart Home

Historie

Smart Home je dnes velmi rozšířený a zároveň velmi vágní pojem. Pro pochopení, co vlastně pojem vyjadřuje a proč má smysl dělat domácnosti „chytré“ je dobré se podívat do historie na vývoj domovních technologií v několika kategoriích.

Klasická elektroinstalace

Elektroinstalace je definována jako soustava elektrotechnických zařízení k vedení a ovládání elektrického proudu v místě jeho výroby, přeměny nebo spotřeby [2]. Klasický přístup k návrhu elektroinstalace je dnes stále osvědčeným standardem a od počátku elektrifikace domácností se příliš nezměnil. V jeho pojetí je funkce a konfigurace zmíněné soustavy dána projektem při jejím vzniku a nelze ji měnit bez fyzického zásahu do zapojení, potažmo bez stavebních úprav.

Rozvody jsou hlavně silové a případné ovládací (tlačítka osvětlení schodiště, HDO aj.) bývají na stejné napěťové hladině. Složitější ovládací prvky se berou jako součást konkrétní technologie (kotel, domácí vodárna...), ne jako součást elektroinstalace, hranici tvoří zásuvka/napájecí kabel. Je třeba podotknout, že nezáleží na složitosti. Klasické instalace může sestávat jen ze světelného a zásuvkového okruhu ve starším bytě, ale stejně tak může čítat pár desítek okruhů moderního rodinného domu. Není jednoznačně určeno, kdy už lze elektroinstalaci nazvat „inteligentní“, většina pramenů ale uvádí jako hranici širší využití principů a prvků automatizace, případně jakýkoli přenos dat mezi prvky instalace.

Multimédia

Multimédia patří spíše do oblasti komfortu uživatele, ale díky integraci systémů s nimi souvisejících do Smart Home lze snížit počet redundantních prvků (např. používat v obýváku jako ovládací rozhraní televizor místo dedikovaného terminálu, využít reproduktory pro upozornění od systému, telefon, zvonek a další).

Pojem multimédia je chápán jako určitá kombinace textových, audio a video dat. Do domácností se dostala s prvními exempláři spotřební elektroniky – radiovými a televizními přijímači. Po nich následovaly přehrávače nejrůznějších analogových a posléze digitálních nosičů a v současnosti bývají uložena buď lokálně v počítači, případně streamována ze vzdálených zdrojů různých poskytovatelů (Netflix, Spotify aj.)[3]. Počet zařízení, na kterých lze tato média přehrávat

také podstatně vzrostl, k původnímu jednomu přijímači/přehrávači se přidal domácí PC a minimálně jedno mobilní zařízení na každého člena domácnosti. Pokud tedy chce uživatel mít svá data uložena lokálně, nastává problém s jejich redundancí a synchronizací, který Smart Home řeší pomocí lokálního streamingu a multi-room audia (viz níže).

PC a internet

Osobní počítače se do domácností dostaly původně jako pracovní nástroj, díky své univerzálnosti a modularitě se ale časem rozšířily do téměř všech oblastí života. PC v některých domácnostech také převzaly místo multimediálního centra. Běžná sestava v podobě CD/DVD přehrávače, rádiového přijímače, audio receiveru, případně dalších prvků jako herní konzole atd. byla nahrazena HTPC (Home Theater PC). Ten zastával všechny její funkce a přidával další (prohlížení webu, nahrávání...). Spolu s ním zůstal jen televizor využitý jako monitor. HTPC má všechny vlastnosti PC včetně operačního systému, ten je pouze rozšířen o ovládací rozhraní optimalizované pro obsluhu dálkovým ovladačem místo klávesnice a myši (např. Windows Media Center, Kodi aj.). Hardware je volen s ohledem na minimalizaci hluku (pasivní chlazení) a umístěn v HTPC šasi, které má vzhled a prvky připomínající zařízení z výše zmíněné běžné sestavy, viz obrázek. HTPC je v jistém ohledu předchůdcem Smart TV [4].

Obrázek 1 - HTPC

V současnosti se PC vrací do pozice pracovního nástroje, případně je využíván pro výkonově náročné aplikace, jako je produkce a úprava hudby a videa, grafické aplikace a hry. V jednodušších činnostech jako je prohlížení webu a komunikace je nahrazován mobilními zařízeními (smartphone, tablet), tyto činnosti zároveň reprezentují statisticky nejčastější využití internetu v domácnosti [5]. HTPC byl nahrazen za Smart TV, které se dají zjednodušeně označit za televizor s integrovaným HTPC a zajistí většinu jeho funkcí za výrazně nižší pořizovací cenu. PC tedy v domácnostech ubývá, naopak roste počet domácích serverů, které řeší problém synchronizace dat mezi všemi ostatními zařízeními.

Internet se postupně stal nejdříve alternativním, a po masivním rozšíření širokopásmové konektivity primárním kanálem pro distribuci multimédií, textovou a hlasovou komunikaci, zpravodajství, vzdělávání a jeho důležitost roste i v dalších oblastech (obchod, školství, týmová spolupráce...). Původní způsob připojení modemem přímo připojeným k PC byl nahrazen síťovými prvky, které

v jednom zařízení kombinují modem, router, switch a Wi-Fi přístupový bod a označují se jednoduše jako „router“ či „Wi-Fi rotuer“. Pomocí něj lze internet sdílet přes mnoho zařízení včetně Smart Home ústředny/brány. Internet také přinesl jeden z pilířů, na kterých Smart Home systémy stojí – sadu protokolů TCP/IP tvořící dnes nejrozšířenější komunikační standard pro datové sítě.

Technologie

V domácnostech se jedná nejčastěji o technologie úpravy prostředí a alternativní zdroje energie. Tradičně jsou brány jako samostatné funkční celky, které mají vlastní, dedikované ovládací prvky jako jedinou možnost ovládání a konfigurace. Tento přístup plně vyhovoval v době, kdy jediným technologickým celkem ve většině domácností byl kotel pro vytápění a ohřev TUV s pokojovým termostatem k nastavení požadované teploty pro celou budovu. Od té doby ale přibyla řada dalších, jako klimatizace, nucené větrání se zpětným získáváním tepla (často označované jako rekuperační jednotky), teplovodní solární kolektory, fotovoltaické systémy, tepelná čerpadla aj.

Při vhodné spolupráci těchto systémů mezi sebou je možné dosáhnout vyššího komfortu obyvatel budovy a zároveň významných úspor energie, tato spolupráce ale často bývá vyřešena špatně nebo vůbec, což u některých uživatelů vyvolává k technologiím odpor a budí dojem, že inovace přináší více problémů než užitku [6].

Integrace se Smart Home může kromě zmíněné spolupráce vyřešit I složitější řízení jednotlivých celků a zároveň umožnuje sběr provozních dat. Z nasbíraných dat lze tvořit statistiky a tyto statistiky pak dále použít pro doladění konfigurace systémů. V neposlední řadě se sníží počet ovládacích prvků na jeden univerzální.

Zabezpečení

EZS (elektronický zabezpečovací systém), EPS (elektronický protipožární systém), ACS (access control system) a CCTV (closed circuit television) jsou technologie původně instalované převážně do veřejných objektů (kanceláře, školy…), kvůli rostoucím požadavkům na bezpečnost osob a majetku a klesající ceně příslušných zařízení jsou ale dnes běžnou součástí domácností. Platí zde v podstatě to samé, co v předchozí kategorii – bývají navrhovány jako samostatné funkční celky bez interakce s ostatními technologiemi budovy.

Hlavním přínosem integrace se Smart Home je snížení redundance zařízení. Mnoho prvků (senzory, ovladače…) se dá současně využít pro EZS, ACS, řízení osvětlení nebo i komunikaci (např. video vrátný).

Problém nastává u některých dodavatelů EZS, kteří poskytují záruku nebo určité služby (např. dohled přes pult centrální ochrany) pouze pokud je systém monolitický a spravovaný jejich techniky. EPS by měl zůstat samostatný,

nebo alespoň nezávislý na funkci ostatních Smart Home subsystémů/komponent z důvodu spolehlivosti a dodržení norem, které pro tyto systémy v ČR platí [7].

Současnost

Jak je vidět z předchozí kapitoly, počet technologií v domácnostech v posledních letech výrazně vzrostl, a i když jednotlivé celky plní svůj účel, spolupráce mezi nimi často není vyřešena. Uživatel pak technologie, u kterých na první pohled není zjevný přínos nepoužívá, případně vypíná, protože jejich obsluhou či nastavením nechce trávit více času, než je nutné. Důsledkem toho pak nemusí být využit plný potenciál budovy, se kterým se počítalo při návrhu, a to vede k dalším problémům a nespokojenosti uživatele [6].

Smart Home systémy nabízí možnost, jak mezi sebou domácí technologie propojit, zjednodušit jejich obsluhu pomocí centrálního řízení a automatizace, a tím vytvořit v zásadě "system systémů". Zároveň samy přináší další funkce. Je však třeba zdůraznit, že nevhodně navržený Smart Home systém může výše popsanou situaci ještě zhoršit. V ideálním případě se návrh děje společně s návrhem zbytku budovy, pokud je system instalován dodatečně, může být stejně efektivní, ale je zpravidla složitější.

Connected Home

Dosud osvědčený výše popsaný „klasický" přístup k návrhu budov tedy ustupuje ve prospěch komplexnějšího, systémového přístupu. Je více možností, jak zmíněný „systém systémů" realizovat. Struktura master/slave nejčastěji používaná v průmyslové automatizaci, s centrálním řídícím prvkem, který zajišťuje kompletní kontrolu zde nelze efektivně použít, protože není jasně dána hierarchie prvků. Aplikuje se tedy koncept připomínající počítačovou síť, kdy jednotlivé uzly (technologické celky či samostatná zařízení) jsou do určité míry schopné pracovat autonomně. Jejich propojením pomocí ústředny pak nevznikne jen shluk technologií ovládaných z jednoho místa, ale přidaná hodnota, kterou je právě Smart Home funkcionalita popsaná v předchozí kapitole. Zároveň je zajištěno, že při výpadku jedné z komponent zbytek systému běží dál, je ale potřeba najít rovnováhu mezi nezávislostí subsystémů a redundancí prvků. V ideálním případě přijde uzel při výpadku konektivity pouze o funkce poskytované zbytkem systému, minimálně je nutné, aby při byl schopen bezpečně vypnout ovládanou technologii. Tento koncept je označován jako Connected Home (propojená domácnost).

Základní strukturu Connected Home lze zařadit do kategorie inteligentní elektroinstalace. Centrálním prvkem je již zmíněná ústředna, ke které jsou připojeny tzv. koncové prvky buď přímo, nebo prostřednictvím nějaké brány. Koncovým prvkem může být jednoduché zařízení (senzor, spínač, relé...) nebo technologický celek (VZT jednotka, domácí vodárna...), důležité je pouze, aby prvek měl

kompatibilní komunikační rozhraní. Pro vysokoúrovňovou komunikaci (řídící jednotky technolog. celků, ovládací terminály…) se téměř výhradně používá výše zmíněná IP síť provozovaná po Ethernetu nebo Wi-Fi. Nízkoúrovňová komunikace (typicky samostatné senzory a akční členy) pak probíhá po různých ovládacích sběrnicích (fieldbus), jako jsou Modbus, CAN nebo BACnet, případně bezdrátovými standardy jako Zigbee nebo Bluetooth. Systém je velice univerzální a lze jej rozšiřovat o další komunikační rozhraní, je tedy omezen zpravidla jen možnostmi integrace na straně koncových zařízení, pokud tato nemají žádné komunikační rozhraní, je nutné jim ho doplnit, případně je vyměnit.

Oproti klasické elektroinstalaci popsané v sekci Historie je většina konfigurace softwarově definována, koncové prvky lze libovolně mapovat a měnit jejich funkce (např ze stmívače osvětlení na ovládání hlasitosti) bez zásahů do zapojení. V závislosti na nastavení řídících jednotek technologických celků lze přistupovat k jejich jednotlivým částem a získávat data potřebná pro jiné komponenty (hladina vody v retenční nádrži, teplota vzdychy přiváděného do budovy…) Lze definovat pravidla automatizace na úrovní celé budovy, sbírat a vyhodnocovat provozní data (o systému samotném, spotřebě energií, parametrech prostředí…) za určitý čas atd.

Díky IP konektivitě je možné integrovat prvky domácí sítě, to lze využít např. pro řízení přístupu dětí k internetu, povolit návštěvám přístup k internetu, ale ne do domácí sítě apod. Přístup samotné ústředny k internetu je užitečný pro získávání dat z veřejných API (doprava, počasí…), vzdálenou správu a přehled o systému atd. Některé systémy (např. Google Home) využívají vzdálené servery ke zpracování vstupních dat (rozpoznávání hlasu, vyhledávání na webu, ukládání uživatelských dat…). Připojení k internetu s sebou ale vždy nese riziko a je nutno pamatovat na zabezpečení proti neoprávněnému přístupu.

Smart Home

Connected Home řeší centralizované ovládání domácnosti, spolupráci jednotlivých technologií, sběr dat, automatizaci atd. Jedná se ale o velmi komplexní systém a pro uživatele může být nastavení a obsluha pokročilejších funkcí příliš komplikované i pokud je systém správně navržený.

Smart Home v jeho originálním významu vychází z myšlenky, že by systém měl uživateli co nejvíce šetřit čas a práci a zároveň by mu měl být co nejméně na očích. Uživatel nepotřebuje vědět, jak systém funguje, bere ho jen jako nástroj, který mu ulehčuje život bez nutnosti větší interakce. Toho lze docílit využitím algoritmů umělé inteligence (AI) a strojového učení.

Systém z nasbíraných provozních dat učí vzory chování uživatele a podle těchto vzorů pak upravuje automatizační pravidla. Typický příklad je prediktivní spínání vytápění, kdy při odchodu uživatele z budovy systém přestane topit a před odhadovaným časem příchodu (na základě skutečného času příchodu za posledních několik dní) opět topit začne, aby v budově v momentě příchodu uživatele již byla požadovaná teplota. Systém může i simulovat přítomnost uživatele

v budově v době, kdy je dlouhodobě pryč. Kromě chování uživatele systém může reagovat na spotřebu energií, externí zdroje dat jako předpověď počasí a mnoho dalších vstupů.

Uživatel si pak může zvolit, jestli chce mít spíše více kontroly a možnost manuálního nastavení nebo ponechá řízení domácnosti na prediktivní automatice. Systém se většinou chová optimálně při co nejméně manuálních zásazích, pokud ale mají rozhodnutí AI úplnou prioritu, mohou mít někteří uživatelé pocit, že je systém „neposlouchá“ a brát to jako projev závady [17].

V současnosti se ale pojem Smart Home kvůli marketingu výrobců elektroniky používá kromě svého původního významu i pro označení domácností, ve kterých jsou využívány jakékoli prvky Connected Home (nemusí se jednat o kompletní systém), a obecně také pro jakákoli domácí zařízení se zabudovanou konektivitou (elektroinstalační prvky, spotřebiče…). V tomto významu je pojem používán i v této práci.

Struktura

Centrální prvek – ústředna (nejčastěji nazývaná anglickým výrazem hub) může být realizována dvěma způsoby. Prvním z nich je fyzické zařízení umístěné lokálně v objektu, kdy je ústředna implementována buď jako dedikovaný hardware nebo jako softwarová aplikace (běžící nejčastěji na domácím serveru nebo vlastním SBC), veškerá vnitřní komunikace probíhá po lokální síti. Druhým způsobem je softwarová aplikace běžící na vzdáleném serveru (v cloudu), kdy jsou brány či koncové prvky přímo připojeny k internetu a musí mít odpovídající zabezpečení. Většina ústředen a bran se konfiguruje pomocí webového rozhraní nebo mobilní aplikace.

Ostatní prvky systému jsou koncová zařízení. Ty spadají do následujících kategorií:

- Osvětlení – spínání a stmívání jednotlivých svítidel, spojování svítidel do skupin a vytváření světelných scén, různé algoritmy jako mood lighting (snaha navodit určitou náladu pomocí teploty a barvy světla), cirkadiánní osvětlení (změna teploty světla v průběhu dne tak, aby korespondovala s přirozeným venkovním prostředím), vizualizace hudby aj., ty ale bývají častěji součástí ústředny než koncových prvků. Senzory, které se pojí s osvětlením (pohyb, intenzita světla…). Osvětlení lze také použít pro signalizaci různých událostí, např. jako tichý zvonek.
- HVAC – vytápění, větrání a klimatizace, řízení zdroje tepla, případně spolupráce více zdrojů (teplené čerpadlo, kotel, solární kolektor…) a více spotřebičů (podlahové vytápění, ohřev TUV…), regulace teploty v jednotlivých místnostech nezávisle na sobě, větrání řízené senzorem CO2, detekce otevřených oken, řízení žaluzií a rolet (na pomezí s kategorií osvětlení), prediktivní spínání vytápění a klimatizace (zmíněno v předchozí

sekci) atd. Velké množství senzorů – čidla teploty a vlhkosti v místnostech, meteostanice, interní senzory technologických celků aj.

- Entertainment – hlavně multimediální systémy pro přehrávání a lokální streaming audia a videa. Někdy se sem zařazují I technologie bazénů a saun. Velmi časté je již zmíněné Multi-room audio – tedy zvukový systém s *N* výstupy (repro v místnostech) a *M* vstupy (lokální média, rádio, stream z mobilního zařízení...), který umožňuje libovolné *M*:*N* mapování, synchronizované přehrávání a další funkce. Návaznost na ostatní kategorie (např. vizualizace hudby, ztlumení světel při spuštění filmu...).
- Security – integrované prvky zabezpečovacích systémů (EZS) a systémů řízení přístupu (ACS), tedy alarm, zámky, pohony vrat a bran, vstup pomocí RFID klíče nebo kódu a podobně. Detekce přítomnosti konkrétních osob (uživatelů či hostů) v domácnosti, realizuje se např. detekcí připojení smartphone k lokální Wi-Fi síti nebo přes Bluetooth beacon. Návaznost na ostatní kategorie hlavně využitím senzorů (PIR pro osvětlení, okenní magnetické kontakty pro HVAC...) Patří sem i kamerové systémy, naopak sem nepatří EPS.
- Control – Ovládací prvky systému, se kterými uživatel fyzicky interaguje. Klasické elektroinstalační přístroje (tlačítka, spínače...), ovládací terminály s dotykovým displayem, hlasový asistent, smartphone, TV aj. Každý typ ovládacího prvku se hodí pro jinou aplikaci, např. pro jednoduché spínání světla je praktičtější klasické tlačítko na zdi než smartphone. Ovládací prvky jsou libovolně mapovatelné na jakékoli ovládané prvky. Nepatří sem senzory.
- Metering & Monitoring - Komponenty pro sběr provozních dat budovy, typicky měřiče spotřeby plynu, elektřiny a vody, a to nejen v rámci celé budovy ale i jednotlivých částí, např. se zvlášť měří spotřeba elektřiny pro vytápění, kuchyň, spotřeba vlastního Smart home systému aj. Detekce havarijních stavů (výpadky elektřiny, únik vody nebo plynu...) a prvky autodiagnostiky samotného Smart Home systému. Patří sem i EPS, ale jak již bylo zmíněno, měl by zůstat samostatným funkčním celkem a pouze poskytovat Smart Home systému údaje o svém stavu.
- Spotřebiče – propojení „chytrých“ elektrospotřebičů se systémem. Mohou to být klasické spotřebiče rozšířené o signalizaci stavu (ledničky, mikrovlnné trouby, pračky, rychlovarné konvice...) a potom spotřebiče, které integraci využijí více (robotická sekačka na trávu nebo robotický vysavač, který si může např. otevírat motorizované dveře mezi místnostmi). Spotřebiče mohou být také jen vypínatelné (pomocí relé), buď kvůli šetření energie, nebo jako doplňkový bezpečnostní prvek (vypnutí při odchodu z budovy).

- Ostatní – další zařízení podle typu objektu a specifických potřeb uživatele, příkladem mohou být zavlažovací systémy nebo monitoring akvária/terária/hospodářských zvířat. Různé brány a rozhraní pro spolupráci s ostatními systémy jako je automobil nebo domácí IT infrastruktura (řízení přístupu k internetu, domácí multimediální server etc.). Senzory, které se nedají zařadit do žádné z ostatních kategorií, např. senzor poštovní schránky. Někdy se sem řadí i virtuální zdroje, tedy různá internetová API (application programming interface) pro poskytování dat o počasí, dopravě, objednávky jídla a podobně.

Implementace

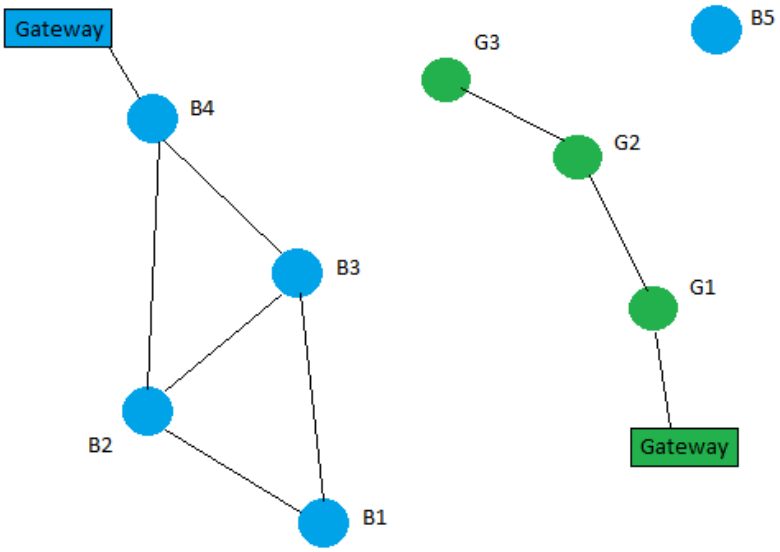
V úvodu je uvedeno, že zájemce o chytrou domácnost má dnes tři cesty, kterými se může vydat:

- Profesionální systém navržený na míru dané budově. Tyto systémy jsou dodávány specializovanými firmami, které zároveň zajišťují instalaci na místě, konfiguraci a servis. Systémy bývají vyvinuté přímo firmou, která je dodává, často jako nadstavba inteligentní elektroinstalace. Využívají tedy proprietární zařízení a komunikační protokoly, pouze komunikace mimo samotný systém (s technologiemi HVAC apod.) probíhá standardy jako Ethernet, KNX, modbus aj. Jedná se o uzavřené ekosystémy, ke kterým lze připojovat jen komponenty daného výrobce a jakékoli úpravy smí provádět pouze licencovaný technik, jinak většinou dochází ke ztrátě záruky. Z toho plynou hlavní výhody i nevýhody tohoto způsobu. Největší výhodou profesionálních systémů je velmi vysoká spolehlivost a záruka na funkci celku (nejen jednotlivých prvků), v rámci které vyřeší servisní technik případné závady v systémů do stanoveného časového limitu. Jako největší nevýhoda se často uvádí vysoká pořizovací cena, ta ale většinou odpovídá poskytovaným službám a přidané hodnotě. Dle mého názoru je největším problémem uzavřenost. Jakékoli nové technologie zařazené do systému musí výrobce nejdříve implementovat jako novou komponentu a tu následně řádně otestovat, což mimo jiné prostředky stojí dost času. Vzhledem k velmi rychlému vývoji v této oblasti mohou být profi systémy funkcionalitou značně pozadu za v současnosti populárními produkty tzv. Consumer Smart Home (viz níže), jako je Sonos, Nest, Amazon Alexa a další. Tam, kde vývoj vlastní komponenty nepřipadá v úvahu (např. zmíněná Alexa) výrobci umožní integraci produktu do svého systému. Příkladem profi systémů je v ČR populární Loxone, nebo méně známý Control4.
- Jít cestou DIY. Systém kompletně navržený, zkonstruovaný a nakonfigurovaný uživatelem. Díky dnešní dostupnosti komponent a nástrojů mohou DIY systémy dosahovat funkčnosti a kvality profesionálních produktů, a to za zlomek jejich ceny. Finanční investice je však vyvážena investicí ča-

sovou, navíc tento přístup samozřejmě vyžaduje of uživatele znalosti a dovednosti v oblasti elektroniky a IT a je tedy vhodný pouze pro ty, pro které je Smart Home koníčkem. Jakékoli poruchy nebo úpravy systému musí uživatel rovněž řešit sám. Naopak největší výhodou pak je, že takový systém je omezen doslova jen dostupným hardwarem, schopnostmi a představivostí uživatele. Pro konstukci prvků se nejčastěji využívají vývojové platformy pro mikrokontrolery (Arduino, ESP...), SBC (Single Board Computer – jednodeskový počítač) na platformě ARM (RaspberryPi a jeho klony), open-source software ať už pro návrh (KiCAD, Blender...) nebo provoz (GNU/Linux, Home Assistant...) a 3D tisk. Vzhledem k tomu, že každý takovýto systém je unikátní, nelze zde uvést konkrétní příklady.

- Poslední, stále oblíbenější možností, jsou produkty tzv. Consumer Smart Home (také nazývané Consumer IoT). Jedná se o elektronická zařízení, která se dosud v domácnostech běžně vyskytovala, ale nově jsou doplněna o nějakou formu konektivity (nejčastěji Wi-Fi nebo Zigbee). Dnes dostupná zařízení pokrývají všechny oblasti Smart Home uvedené v předchozí sekci, tedy osvětlení, elektroinstalační prvky (spínače, zásuvky...), elektrické spotřebiče, senzory, měřiče energií, multimediální prvky (reproduktory, streamovací zařízení jako např. Chromecast), případně různé moduly určené k vestavbě (např. modul se sadou relé). Tato zařízení reprezentují koncové prvky, k nim je dodávána brána (každý systém má svou) a jako ústředna se pak používá nejčastěji některý z populárních hlasových asistentů (Google Home, Amazon Alexa...) nebo cloudová služba (ifttt.com...). Tento přístup reprezentuje jakousi „stavebnici“, která umožňuje komukoli vytvořit systém ve stylu DIY bez větších znalostí v oboru. Většině lidí se dnes pod pojmem „chytrá domácnost“ vybaví právě tyto produkty a zájem o ně stále roste. Tento přístup vyniká pro rychlou realizaci jednoduchých požadavků, k vytvoření funkčního systému stačí nakoupit patřičné produkty a provést instalaci. Ta je rychlá a jednoduchá, jelikož koncové prvky jsou většinou napájeny z baterií a komunikují bezdrátově. Konfigurace je intuitivní a provádí se přes smartphone aplikaci. Pořizovací cena je díky masové produkci na úrovni spotřební elektroniky. Díky popularitě a konkurenci mezi výrobci se také tento segment rychle vyvíjí a jednotlivým produktům rychle přibývají nové funkce. Výrobci prezentují tyto systémy jako budoucnost Smart Home a je pravda, že i většině lidí se dnes pod pojmem chytré domácnosti vybaví právě tento přístup, složit z těchto prvků komplexnější systém ale nemusí být jednoduché. Produktové řady různých výrobců nejsou přímo kompatibilní mezi sebou kvůli zmíněné konkurenci. Každý výrobce vyniká v jiné oblasti (osvětlení, multimédia...) a pokud chce uživatel např, jen jeden koncový prvek z ekosystému daného výrobce, potřebuje i odpovídající bránu pro

připojení k ústředně. Při použití technologie Zigbee nebo Z-Wave pak zařízení různých výrobců netvoří stejnou mesh síť a při velkém počtu sítí se mohou i navzájem rušit, viz Obrázek 2. Problémy mohou být i se softwarovou kompatibilitou, jelikož API nejsou nijak standardizované a spoléhají jen na konvence výrobců. Výrobce většinou udává kompatibilitu s nejběžnějšími ústřednami – Google Home, Amazon Alexa a Apple HomeKit, pokud ale tato není uvedena, nebo pokud uživatel chce jinou ústřednu, zbývá jen metoda pokus-omyl. Stejně jako u DIY systému musí návrh systému a případné poruchy řešit uživatel sám, záruka je pouze na samostatné výrobky (prvky systému). Prvky napájené z baterií vyžadují jejich občasnou výměnu, většinou ale poskytují ústředně úroveň jejich nabití.



Obrázek 2 - dvě oddělené Zigbee mesh sítě, uzly „Gateway" reprezentují koordinátor. V případě, že by všechny uzly byly ve společné síti, uzel B5 vzdálený od zbytku modré části by se mohl připojit skrz G3, takto zůstane nepřipojen. Uzly v zelené části nemají záložní cestu, např. při výpadku G2 tak bude odpojen i uzel G3.

Brána vytvořená v rámci této práce je určena hlavně pro třetí popsaný přístup – Consumer Smart Home. Další text se tedy bude zabývat pouze jím.

Problémy

První z problémů už byl zmíněn v předchozí sekci, momentálně neexistuje jednotný standard pro propojení a integraci produktů různých výrobců, kompatibilitu mezi nimi je tak před realizací vždy třeba ověřit a případně doplnit systém o potřebná rozhraní, brány či moduly.

Druhým a asi největším problémem je soukromí a informační bezpečnost, hlavně při použití cloudové ústředny. Mnoho zařízení je osazeno mikrokontrolery, které sice umožňují konektivitu přímo do počítačové sítě (Wi-Fi či Ethernet rozhraní), ale zabezpečení přístupu k ovládání zařízení, nebo čtení dat z něj neřeší,

často kvůli tomu, že na něj nezbývá výpočetní výkon. I u zařízení, které proti neoprávněnému přístupu zabezpečena jsou, ale nastává stejný problém jako v ostatních oblastech IT – žádný software není bez chyby a nějaká bezpečnostní zranitelnost se dříve nebo později objeví. Je tedy nutné buď provádět pravidelné aktualizace software (firmware) na koncových prvcích, nebo odstranit cestu, kterou by mohla být napadena – umístit je na oddělenou síť bez přístupu k internetu. Je také možné, že výrobce dané zařízení přestane podporovat po vydání nového modelu a žádné nové aktualizace už dostupné nebudou, v tom případě je oddělená síť jediným řešením.

Zranitelný systém potenciálnímu útočníkovi může poskytnout nejrůznější data o budově (obzvlášť pokud je využita AI, kde se systém učí vzory chování uživatele - kdy spí, kdy není doma...) nebo např. záznamy z CCTV. V nejhorším případě má útočník přístup do kritických systémů a může si vypnout EZS, odemknout hlavní vchod a otevřít garážová vrata, případně způsobit škody skrz ovládání HVAC, vodovodních ventilů apod. Zatímco toto je pro běžnou domácnost jen hypotetický případ (pro zloděje je mnohem snazší překonat fyzické bariéry – např. rozbít okno – než se snažit napadnout budovu elektronickou cestou), úniky osobních dat a kamerových záznamů jsou bohužel časté. Obecně je Smart home oblastí, kde se fyzická bezpečnost setkává s bezpečností informační a při návrhu je potřeba myslet na obojí.

Nakonec ještě připomínka, že systém by měl sloužit uživateli, šetřit mu čas a práci a fungovat transparentně na pozadí, aby uživatel nad prováděnými procesy musel přemýšlet co nejméně. Při špatném návrhu to ale může být přesně naopak, systém je „neohrabaný“, náročný na obsluhu, je nutné ho často ručně nastavovat a uživateli spíše přidělává starosti. Tímto nejvíce trpí právě uživatelé, kteří se snaží z Consumer Smart Home produktů poskládat celek s pokročilejšími funkcemi a zároveň nechtějí věnovat mnoho času návrhu. Toto lze do určité míry vyřešit zjednodušením interakce jednotlivých komponent, ale obecně je nutné provést návrh pečlivěji, nebo ho svěřit někomu se zkušenostmi v této oblasti.

Možná budoucnost

Ideálním řešením by bylo vzít si z každého přístupu jen to, co funguje a navrhnout Smart home instalaci, která tyto výhody spojuje. Je třeba si uvědomit, že Smart home je komplexní záležitost, kde jednoduchá instalace a konfigurace je protichůdný požadavek ke spolehlivosti, bezpečnosti a transparentní funkci.
V tomto ohledu je podle mého názoru nejlepší svěřit návrh, instalaci a servis odborníkovi, s tím, že uživatel by si mohl volit konkrétní prvky, které v domácnosti chce. Tím by byla zajištěna určitá úroveň kvality, spolehlivosti a bezpečnosti a zároveň záruka pro uživatele na funkci Smart home jako celku. Místo vlastního vývoje celého systému lze využít Consumer Smart Home produkty, které při správném použití velmi dobře plní svůj účel a jejich ceny se blíží cenám prvků klasické elektroinstalace a jejich další vývoj řeší jednotliví výrobci, takže není

třeba se bát, že by systém zaostával za aktuálními trendy. Zbývající mezery jako je ústředna, ovládací rozhraní a integrace s technologickými celky pak lze po vzory DIY přístupu vyplnit univerzálními prvky (MCU/SBC) se softwarem přizpůsobeným danému účelu. Ten může být i volně dostupný či plně otevřený, což by umožnilo zkušenějším uživatelům mít správu systému kompletně ve vlastní režii. Servis pro ostatní uživatele by byl řešen buď jako jednorázová záruka na danou dobu po instalaci (jako u klasické elektroinstalace), nebo jako dohodnutá úroveň (reakce na závadu v časovém limitu) záruky po celou dobu provozu zařízení za paušální poplatek (jako ve správě IT nebo u některých EZS systémů).

Takové řešení samozřejmě vyžaduje poskytovatele, který ho bude zajišťovat. Jediný systém, který se tomuto popisu blíží nabízí britská firma Smartisant Ltd. (www.smartisant.com). Dva hlavní problémy zmíněné výše lze ale řešit do určité míry i jinak.

Kompatibilita

Kompatibilitu koncových prvků různých výrobců mezi sebou a s dalšími systémy by bylo možné zajistit nahrazením současné kombinace ústředny a několika komunikačních bran za ústřednu, která v sobě již integruje konektivitu pomocí nejpoužívanějších technologií. Žádný z výrobců takový produkt samozřejmě neposkytne, bude tedy potřeba využít některý z otevřených projektů. Původní nahrazená ústředna pak půjde k té nové připojit jako koncový prvek a může systému dále poskytovat své funkce (např. hlasový asistent).

Bezpečnost

Z pohledu informační bezpečnosti zvyšuje centralizace a propojenost technologií v budově výrazně zranitelnost celku. U IP části systému je možné aplikovat obecné bezpečnostní principy pro IP sítě. Smart Home systém je vhodné izolovat od zbytku domácí sítě umístěním do vlastní podsítě a zároveň povolit každému prvku pouze přístupy a činnosti, které potřebuje ke své činnosti (princip least privilege). K tomu je nutný router s firewallem (síťový prvek pro směrování a filtrování provozu), který omezuje komunikaci mezi jednotlivými podsítěmi a internetem, běžné kombinované síťové prvky (modem, router a Wi-Fi AP v jednom) tuto funkcionalitu většinou nemají. Konkrétně mezi Smart Home podsítí a internetem by měl být umožněn jen klíčový provoz, jako je např. stahování aktualizací. Mezi lokálními sítěmi také není vhodné povolovat více než je třeba, např. může být nutné umožnit Smart Home síti přístup ke sdíleným datům v LAN (lokální síť s počítači a mobilními zařízeními) kvůli přehrávání multimédií, ale naopak není třeba aby zařízení v LAN měla přímý přístup (ne skrz ústřednu) k prvkům Smart Home. V případě kompromitovaného zařízení v jedné podsíti tak nejsou ohroženy ostatní. Zakázat je nutné jakékoli „autokonfigurační“ služby jako UPnP nebo mDNS. Kritickým prvkem je ústředna, která přístup k internetu potřebuje pro získávání dat, komunikace by měla být povolena jen tehdy, pokud je

zahájena ústřednou, nikoli zvenčí. Jakýkoli vzdálený přístup k systému by měl být řešen pomocí VPN, nikoli přímým směrováním.

Dalšími potencionálními plochami k útoku, na které je třeba myslet při konfiguraci jsou jakékoli zvenčí dostupné ethernetové porty (typicky venkovní IP kamery) a samozřejmě Wi-Fi síť. Měla by být preferována drátová komunikační rozhraní z důvodu nemožného odposlechu (při vhodném umístění rozvodů), omezení vlivu zarušení radiových frekvencí ale i minimalizaci počtu prvků napájených z baterií. Pokud je nutné použít bezdrátový prvek, je potřeba zvolit komunikační protokol s podporou šifrování (kromě vzácných případů, kdy nevadí odposlech komunikace).

Princip Connected Home by se dal popsat i dalším, dnes velmi populárním pojmem – IoT (Internet of Things). Pojmy mají podobný význam, Connected Home je ale specifický pro oblast domácí automatizace.[8] Pojem záměrně není v tomto textu použit, mohl by být matoucí vzhledem ke snaze o co nevětší oddělení koncových prvků od internetu a uzavření komunikace do lokální sítě. Místo něj se může v realizační části objevovat zkratka NoT (Network of Things – síť věcí, pojem je převzatý od jednoho z vývojářů open source ústředny Home Assistant Francka Nijhofa), která má funkčně stejný význam, ale zdůrazňuje právě omezení na lokální síť.

Komunikační brána

Zařízení vytvořené v rámci této práce má za jeden ze svých cílů právě řešit problémy popsané v předchozí sekci. Díky použití otevřených technologií a integraci nejběžnějších komunikačních rozhraní nahradí toto zařízení jak ústřednu, tak brány pro jednotlivé Consumer Smart Home ekosystémy. Půjde tedy přímo připojit v podstatě jakýkoli koncový prvek bez ohledu na výrobce a zároveň se tím zjednoduší struktura systému i definice automatizačních pravidel. Zařízení se snaží poskytovat výhody DIY ústředny (otevřenost, univerzálnost, rozšiřitelnost), ale zároveň být kompletním centrálním prvkem připraveným k nasazení jako jakákoli jiná Consumer Smart Home ústředna.

Brána v sobě bude integrovat senzory pro nejčastěji požadované parametry prostředí – teplota a vlhkost vzduchu, intenzita osvětlení a obsah CO_2 ve vzduchu. V místnosti, kde bude instalována tak nebude třeba dalších koncových prvků s těmito funkcemi. Rozšiřitelnost bude omezena jen softwarovou podporou (kterou lze doplnit) a výkonem hardwarové platformy na které poběží.

Brána bude také zajišťovat některé z bezpečnostních funkcí popsaných v předchozí sekci. Připojení k síti bude realizováno pomocí ethernetového rozhraní, pro Wi-Fi konektivitu tedy brána bude moci sloužit jako AP a tím řešit oddělení Smart Home podsítě. Brána bude implementovaná jako aplikace nad nějakým operačním systémem, v tomto OS bude možné definovat filtrační pravidla pro lokální firewall a povolit jen nutnou komunikaci.

Základními bloky bude jádro (to má hardwarovou a softwarovou část), napájecí zdroj, komunikační periferie a v neposlední řadě senzory. Při výběru komponent bude bráno v potaz i to, že bránu by mělo být možné zkonstruovat v domácích podmínkách.

Jádro - Software

Ústředna má tři hlavní funkce:

- Integrace koncových prvků – propojení a spolupráce všech připojených zařízení bez ohledu na výrobce nebo technologii
- Centralizované ovládání – rozhraní, pomocí kterého může uživatel celý systém ovládat a získávat informace o jeho aktuálním stavu či historii
- Automatizační engine – komponenta, která umožňuje definovat automatizační pravidla, tedy chování systému v určitých podmínkách, reakce na události atd.

Pro zajištění této funkcionality bude použit open-source software. Tyto aplikace jsou vyvíjeny komunitou lidí, je k nim volně dostupná veškerá dokumentace a zdrojové kódy (odtud název open-source) a je možno je libovolně šířit a upravovat (detaily se mohou lišit podle konkrétní licence). Díky tomu lze tento software, za předpokladu potřebných znalostí, přizpůsobit konkrétnímu případu užití nebo rozšířit o nové funkce. Pokud jsou tyto úpravy užitečné i pro ostatní uživatele, je možné je po schválení příslušnou komunitou začlenit do původní aplikace. Výhodou otevřeného software je také vysoká rychlost vývoje a začleňování nových trendů a technologií.

Open-source projektů pro řízení Smart Home je celá řada a nachází se v různých stádiích vývoje, zde jsou uvedeny tři z nich, které jsou v současné době funkcemi nejpokročilejší, stabilní a aktivně vyvíjené.

Open HAB

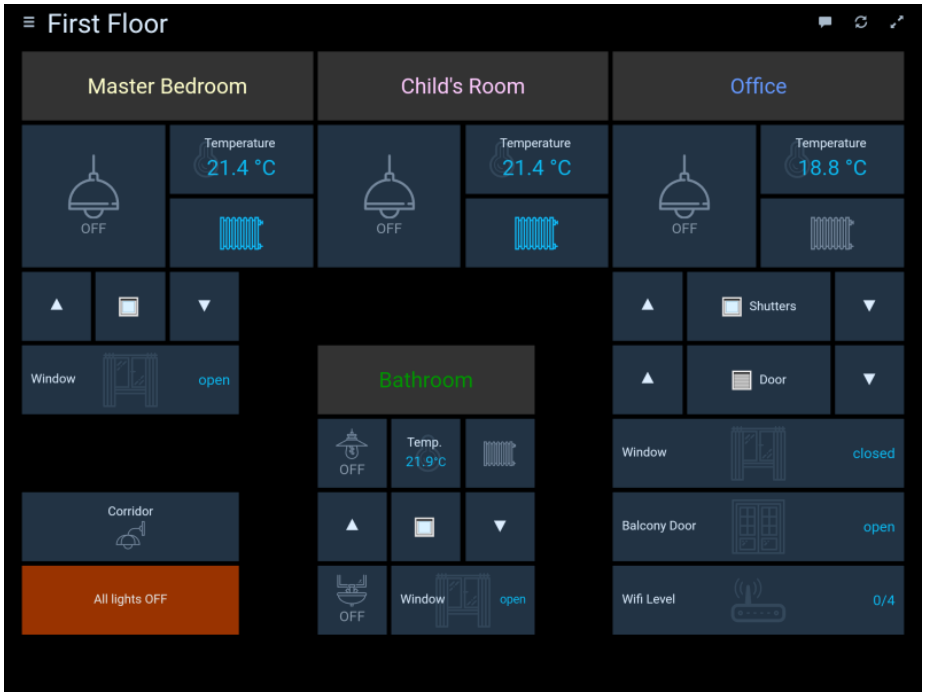
Open Home Automation Bus byl jedním z prvních projektů (vznikl v roce 2010) se záměrem vytvořit open-source systém pro řízení Smart Home a integraci DIY komponent s Consumer IoT výrobky. Autorem systému je Kai Kreuzer a projekt je aktuálně zastřešován organizací OpenHAB Foundation, která se kromě jeho správy snaží i informovat veřejnost o možnostech otevřených systémů v oblasti Smart Home.

OpenHAB je webová aplikace psaná v Javě, lze ho tedy provozovat na jakémkoli operačním systému s JVM (Java Virtual Machine), projekt nabízí i svůj klon linuxu - OpenHABian, což je hotový obraz disku, který stačí stáhnout a buď přimontovat jako systémový disk virtuálního stroje, anebo nakopírovat na SD kartu sloužící jako systémový disk u SBC platforem. Kromě něj je k dispozici instalační skript pro Linux i Windows a také Docker image. Po instalaci je frontend dostupný na TCP portu 8080, kde lze dokončit instalaci pomocí jedné z připravených šablon.

Systém má propracovanou interní strukturu, která může být ze začátku složitější na pochopení, ale zůstává přehledná i při složitější konfiguraci. Základním prvkem je item (položka), která reprezentuje nějaký vstup (např. venkovní teplota) a používá se při konstukci automatizačních pravidel a ovládacích prvků. Položka je nezávislá na fyzickém senzoru, při změně zdroje dat tak není nutné zmíněná pravidla a ovládací prvky upravovat. Integraci koncových prvků poskytují jednotlivé vazby (bindings), které jsou realizované jako zásuvné moduly umožňující systému komunikovat s konkrétní technologií (konkrétní ekosystémy jako např. Phillips Hue, nebo obecné protokoly jako např. MQTT). Každý binding mapuje specifické vlastnosti připojené technologie na univerzální reprezentaci zařízení uvnitř OpenHABu, tzv. thing (věc). Things mají funkce dané fyzickým zařízením (on/off, dimm, hodnota...), tyto funkce jsou označovány jako channel (kanál), které se mapují na již zmíněné položky pomocí propojek (link). Jako příklad lze uvést připojení Zigbee senzoru Xiaomi Aqara. Nejprve je třeba nainstalovat *binding*, který záleží na použité Zigbee bráně. Po nainstalování a připojení bude senzor detekován jako nová *věc* se všemi dostupnými *kanály* a umístěn do tzv. inboxu. Nakonec je potřeba vytvořit novou *položku* a namapovat na ni příslušné kanály.

Webový frontend poskytuje tři různá uživatelská rozhraní. Paper UI slouží hlavně jako konfigurační pro přidávání *položek* a *vazeb* a mapování *kanálů*. Nachází se zde zmíněná kategorie inbox, do které se umisťují zařízení nalezená autodetekcí. Slouží také k instalaci *rozšíření* (add-ons). Konfigurace pro *vazby* a *rozšíření* je ale řešena textovými soubory, přičemž Paper UI pro ně nabízí vestavěný editor. Formát konfiguračních souborů není standardizován a záleží na vývojáři daného rozšíření nebo vazby. Zbývající rozhraní Basic UI a HAB Panel jsou určena k ovládání systému. Basic UI je automaticky generované z dostupných *položek* a dynamicky mění velikost a rozložení podle obrazovky, je tedy vhodné pro mobilní zařízení. HAB Panel je naopak vysoce konfigurovatelný (přes vlastní GUI konfigurátor) a nabízí uživateli možnost přizpůsobit si ovládání systému do nejmenších detailů.

Automatizace se konfiguruje výhradně pomocí textových souborů. Základem jsou uživatelem definovaná pravidla (rules), ta jsou spouštěna nějakou událostí a volají nějaký skript, který pak může realizovat definovanou akci (zapnutí světla, odeslání dat do webové služby...). Syntaxe pravidel a skriptů je založená na výrazovém jazyce Xbase. Alternativně lze použít addon pro definici pravidel v grafickém jazyce Blockly, který ale dle mého názoru není tak expresivní a lepší alternativou je integrace s níže popsaným NodeRED [18].



Obrázek 3 - HABPanel - jedno z ovládacích rozhraní OpenHAB

Home Assistant

První verze Home Assistant vznikla v roce 2014, kdy ho jeho autor Paulus Schoutsen vytvořil jako jednoduchý skript pro automatizaci osvětlení. Od té doby se vyvinul do současné podoby nejoblíbenějšího řešení pro open-source Smart Home ústřednu. Oficiálně je software stále v beta stádiu vývoje, aktuální verze (říjen 2020) je 0.117. Vývoj je velice aktivní a nové verze vychází každé tři týdny, díky tomu systém velmi rychle adaptuje nové funkce, technologie a zařízení, zároveň mu ale tento přístup ubírá na stabilitě.

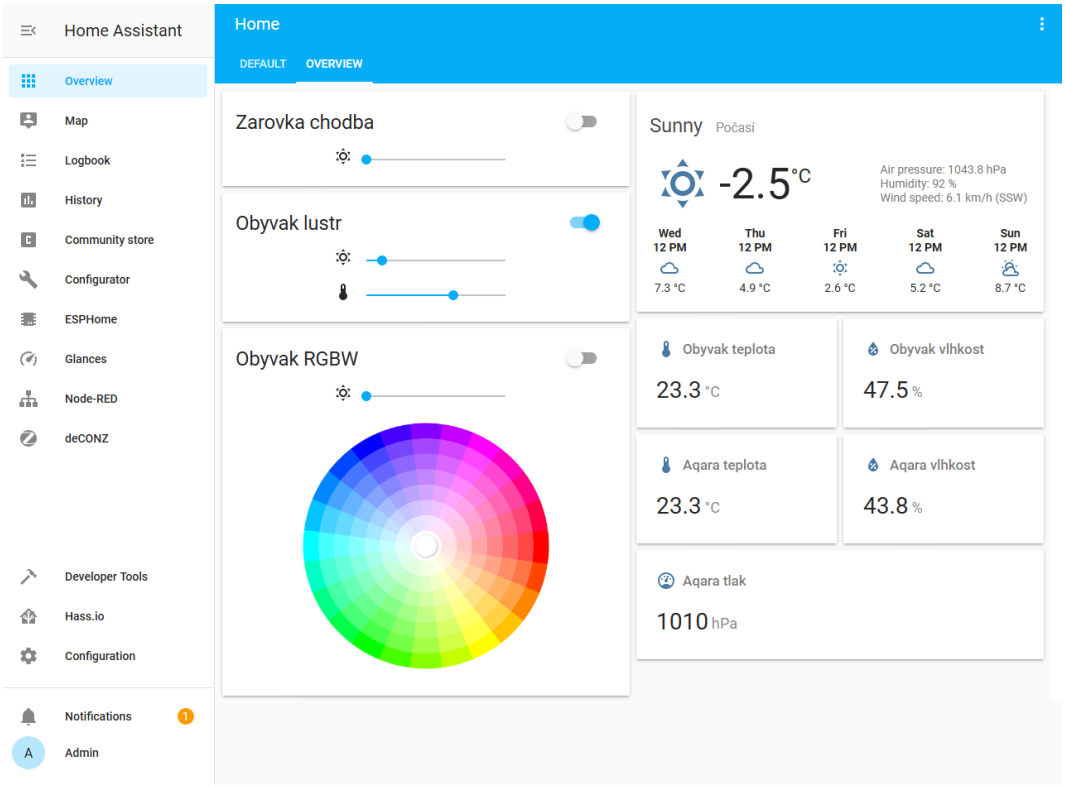
Opět se jedná o webovou aplikaci dostupnou pro OS Windows a Linux. Aplikace má dvě vrstvy, backend je psaný v jazyce Python, zatímco frontend je vytvořený v JavaScriptu a optimalizovaný primárně pro mobilní zařízení. Systém využívá pro svou správu kontejnerizační software Docker, jeden kontejner je vytvořen pro hlavní modul (core) a jeden pro každý instalovaný add-on (viz níže), poslední kontejner zvaný *supervisor* slouží pro správu všech ostatních kontejnerů. Díky tomu je možné provádět z webového rozhraní i restart nebo aktualizaci celé aplikace.

Základem vnitřní struktury jsou komponenty (`components`), které reprezentují určitý typ zařízení, např. *switch* - spínač, *light* – světlo aj. nebo součást aplikace, např. *addon, UI* (webové ovládací rozhraní zvané Lovelace). Na jednu stranu je toto označení matoucí, na druhou stranu to systému dává obrovskou přizpůsobitelnost, jedinou neměnnou součástí je framework, který komponenty spravuje. Pokud komponenta pro reprezentaci některého koncového prvku chybí, je možné ji doinstalovat. Komponenty mají parametr `platform`, který udává

druh připojeného zařízení (výrobce, ekosystém, technologie…), dvojice komponenty a platformy pak definuje tzv. `entity`, ty reprezentují funkce poskytované koncovým prvkem. Zde oproti OpenHABu chybí vrstva abstrakce a entitám je dobré dávat názvy, ze kterých lze jednoznačně určit kterou funkci kterého prvku reprezentují (např. *light.obyvak_gauc_jas*). Entity mohou mít stavy (`states`), které záleží na typu entity, např. entita typu *switch* má možné stavy *on* a *off*. Vstup od entity je reprezentován voláním události (`event`), např. stisknutí tlačítka, výstup je reprezentován tzv. službami (`services`), které entita systému poskytuje, např. zmíněný jas žárovky. Stavy, události a služby se dají dále mapovat na automatizační pravidla a ovládací prvky uživatelského rozhraní. Protože entity a komponenty mohou reprezentovat v podstatě cokoliv, přidává Home Assitant ještě kategorie `integrations` pro komponenty umožňující integraci nějaké technologie (analogie s vazbami v OpenHABu) a `devices` pro fyzické koncové prvky. Tyto kategorie slouží pouze pro větší přehlednost.

Kvůli svému organickému vývoji postrádá Home Assistant oproti OpenHABu některé vrstvy abstrakce, což občas ztěžuje rozlišení některých pojmů (např. entita vs služba) při nasazení v praxi to ale neubírá na funkcionalitě. Větší problém je automatizace, která je řešena dvěma způsoby, buď přes webové rozhraní, to je jednodušší, ale chybí zde některé pokročilejší funkce, nebo přes textový soubor ve formátu YAML, kde jsou sice dostupné veškeré dostupné funkce, ale pro použití je nutná znalost jazyka YAML a také některých vnitřních procesů systému. Tyto dva způsoby jsou na sobě úplně nezávislé a automatizace definovaná pomocí jednoho není vidět v tom druhém. Stejně jako u OpenHABu je možné integrovat NodeRED jako alternativní engine pro automatizaci.

Home Assistant má jen jedno ovládací rozhraní zvané Lovelace a dostupné jako záložka konfiguračního rozhraní. V implicitním nastavení je dynamicky generované, lze ho ale přepnout do ručního režimu a následně editovat jak grafickým editorem, tak pomocí konfiguračního YAML souboru, na rozdíl od automatizace se zde oba způsoby doplňují [18].



Obrázek 4 - Webové rozhraní aplikace Home Assistant

Node-RED

Node-RED na rozdíl od předchozích dvou aplikací není kompletní řešení ústředny typu „vše v jednom“, ale soustředí se na automatizační engine a v této oblasti nemá (volně dostupný) ekvivalent. Autory projektu, který vznikl v roce 2013 jsou Nick O'Leary and Dave Conway-Jones z IBM. Node-RED je webová aplikace napsaná v jazyce Node.js, takže ji lze provozovat na všech operačních systémech s Node.js interpreterem.

Node-RED využívá paradigma flow-based programování – program pracuje s tokem dat, který je postupně zpracováván řetězcem funkčních bloků, každý blok má vstup a výstup v jednotném formátu a jejich řetězec díky tomu lze libovolně měnit. Výhodou tohoto přístupu je možnost podle potřeby přidávat a ubírat bloky z řetězce bez nutnosti zásahu do zbytku programu, dále efektivní rozdělení složitějších pravidel na dílčí bloky a také snadná vizualizace, která usnadňuje orientaci v programu. Model toku dat, kdy je zpráva na vstupu zpracována podle nějakých pravidel a odeslána na výstup je pro automatizační pravidla Smart Home ideální.

Základním prvkem je `node` (uzel), který reprezentuje funkční blok a může být vstupní, výstupní nebo pracovní. Defaultní instalace zahrnuje nejpoužívanější nodes a další se dají doinstalovat buď samostatně, nebo jako balíky zvané kolekce, které typicky poskytují sadu nodes pro integraci s nějakou technologií (aplikace jako Home Assitant, komunikační sběrnice jako Modbus a mnoho dalších). Data přichází do aplikace skrz vstupní node, která je zformátuje do interní

reprezentace – JSON řetězce zvaného message (zpráva). Ta pak prochází řetězcem pracovních nodes až do výstupní, kde je převedena do formátu srozumitelného pro cíl. Řetězec nodes se nazývá sequence (příklad viz Obrázek 33) a jednotlivé sekvence se umisťují do skupin zvaných flow (ve webovém rozhraní reprezentovány jednotlivými záložkami). Kromě těla zprávy může program pracovat s proměnnými na úrovni sequence, flow nebo global (celá Node-RED instance). Jednotlivé nodes mají jasně dané funkce a konfigurují se graficky ve webovém rozhraní (přes „klikací" formuláře), výjimkou je node *function*, který reprezentuje jakýkoli blok kódu v NodeJS jazyce a může tak obsahovat velmi složitou logiku, která se pomocí ostatních nodes nedá poskládat. Může ale také kombinovat funkce několika ostatních nodes do jedné, což není doporučeno, protože to obchází princip flow-based programování a program tak ztrácí výše popsané výhody, které z něj plynou [14].

I když je Node-RED primárně určeno pro automatizaci, lze do něj pomocí kolekce Node-RED dashboard přidat uživatelské ovládací rozhraní, to ale není zdaleka tak propracované jako jeho alternativy HAB Panel a Lovelace. Nejlépe je tedy podle mého názoru použít Node-RED ve spojení s jedním ze dvou předchozích projektů.

Jádro - Hardware

Všechny tři projekty zmíněné v předchozí sekci jsou webové aplikace s podporou všech tří nejrozšířenějších OS – Microsoft Windows, GNU/Linux a Apple MacOS. Vzhledem k tomu, že aplikace nepotřebují, aby OS měl grafické rozhraní (je řešeno přes webový přístup), je nejvhodnější použít Linux bez GUI kvůli výrazně nižším nárokům na systémové prostředky. Jako HW platforma pro provoz aplikace se tedy nabízí některý z jednodeskových počítačů na platformě ARM s dostatečným výkonem požadovanou konektivitou (Ethernet, Wi-Fi, Bluetooth).

SBC (Single-Board Computer) je víceméně plnohodnotný počítač na jedné desce plošných spojů. Srdcem SBC je tzv. SoC (System on Chip), což je integrovaný obvod, ve kterém je spojeno CPU, GPU, paměť a řadiče vstupně-výstupních periferií. Procesory ARM jsou pro tuto aplikaci vhodné, protože jejich architektura je oproti těm používaným v klasických PC (x86/x64) jednodušší. Díky tomu jsou kompaktní, produkují méně odpadního tepla a mají menší spotřebu energie (což je klíčové pro napájení ústředny, viz níže), také ale poskytují nižší výpočetní výkon [19]. Pro středně velký Smart Home by však výkon měl být dostačující. SBC nemají ze své podstaty vyměnitelné komponenty, je tedy třeba vybrat vhodný model.

Asi nejznámějším SBC je Raspberry Pi, jehož první model vznikl v roce 2012 a byl navržen jako učební pomůcka pro výuku informačních technologií, hlavně v rozvojových zemích. Dnes má Raspberry Pi několik modelů pro širokou škálu použití, např. Pi 4 je vhodné pro aplikace náročné na výkon (4x1,5GHz procesor a až 8GB RAM), Pi Zero naopak vyniká nižší spotřebou energie, cenou a

velikostí. Dobrý poměr mezi výkonem a spotřebou nabízí model Pi 3B+, v úvahu přicházejí i jeho klony/alternativy od jiných výrobců:

- Raspberry 3B+
  - 64-bitový 4x 1,4 GHz procesor, 1GB RAM
  - 4x USB 2.0, Gigabit Ethernet (přes USB 2.0 – max. rychlost omezena na 300Mbps), HDMI
  - Wi-Fi 802.11b/g/n/ac, Bluetooth 4.2
  - systémový disk ve formě SD karty
  - OS Raspbian – odnož systému Debian GNU/Linux pro Raspberry Pi, ověřená plná podpora všech funkcí.
  - Spotřeba bez periferií – 2W (idle) – 5W (plná zátěž)
  - Cena cca 1000 Kč.
- Orange Pi 3
  - 64-bitový 4x1,8GHz procesor, 2GB RAM
  - 4x USB 3.0, 1x USB 2.0, Gigabit Ethernet, HDMI, Mini PCIe
  - Wi-Fi 802.11b/g/n/ac, Bluetooth 5.0, infračervený port
  - systémový disk ve formě eMMC modulu
  - OS Armbian GNU/Linux s nejnovějším mainline kernelem, PCIe port není podporován, občasné problémy s detekcí USB zařízení
  - Spotřeba bez periferií 2,5W (idle) – 10W (plná zátěž)
  - Cena cca 1200 Kč
- Banana Pi M64
  - 64-bitový 4x1,2GHz procesor, 2GB RAM
  - 2x USB 2.0, 100 Mbps Ethernet, HDMI
  - Wi-Fi 802.11b/g/n, Bluetooth 4.0
  - systémový disk ve formě SD karty nebo eMMC modulu
  - OS Debian GNU/Linux od výrobce je zastaralá verze, která po aktualizaci přestane fungovat. K dispozici je také neoficiální verze Armbianu, ověřené jsou ale jen některé funkce, Bluetooth je problematický.
  - Spotřeba bez periferií – 2W (idle) – 5W (plná zátěž)
  - Cena cca 1400 Kč
- Libre Computer La Frite (AML-S805X-AC)
  - 64-bitový 4x1,2 GHz procesor, 1GB RAM
  - 2x USB 2.0, 100 Mbps Ethernet, HDMI
  - Nemá bezdrátová rozhraní, pouze infračervený port
  - systémový disk ve formě eMMC modulu
  - V době vydání (červen 2019) v podstatě neexistující softwarová podpora, v současnosti je ale dostupný Armbian GNU/Linux s nejnovějším mainline kernelem a podporou většiny funkcí, problémy jsou pouze s grafickou kartou, která nebude v ústředně využita
  - Spotřeba bez periferií – <1W (idle) - 4W (max. zatížení)

- Cena cca 420 Kč

Komunikace

Consumer Smart Home produkty komunikují téměř výhradně bezdrátově. Následující komunikační standardy jsou v současnosti nejpoužívanější.

Wi-Fi

Standardizována jako rodina protokolů IEEE 802.11, Wi-Fi je dnes primárním (v domácnostech v podstatě jediným používaným) standardem pro lokální, bezdrátovou počítačovou síť. Typická aplikace v domácnosti je kombinovaný síťový prvek zajišťující kromě Wi-Fi AP (Access Point – přístupový bod) ještě funkci routeru, případně modemu pro připojení k internetu. Technologie je také často využívána poskytovateli internetu, při použití směrových antén s přímou viditelností mezi nimi lze vytvářet spoje na vzdálenosti v řádech kilometrů.

První specifikace vznikla v roce 1998 a od té doby prošel protokol řadou verzí - a, b, g, n a ac. Aktuálně se používají poslední dvě a pracuje se na zavedení nové verze ax. Standard 802.11n může pracovat v pásmech 2,4GHz nebo 5GHz, přičemž koncové prvky smart home většinou implementují jen tu jednodušší, 2,4GHz část specifikace. Maximální teoretická přenosová rychlost je 300Mbps s využitím technologie MIMO (multiple input multiple output). Standard 802.11ac využívá výhradně 5GHz pásmo a implementuje řadu technologií (MU-MIMO, beamforming...), které mu umožňují dosáhnout teoretické přenosové rychlosti až 1300Mbps při stejném pokrytí jako 802.11n v pásmu 2.4GHz (rádiové vlny s vyšší frekvencí mají menší dosah a hůře pronikají pevnými materiály než ty s nižší). Tento standard pro smart home koncové prvky není relevantní, protože by jednak jeho implementace zařízení komplikovala (a tím pádem zvyšovala jeho cenu a spotřebu energie), a navíc vyšší přenosovou rychlost koncové prvky nevyužijí. Standardy jsou určovány neziskovou společností Wi-Fi Alliance a jsou poměrně striktní, každé zařízení musí být certifikováno (certifikované výrobky jsou opatřeny logem Wi-Fi – v nadpisu sekce) a je tedy zaručena interoperabilita bez ohledu na výrobce.

Wi-Fi má hvězdicovou topologii, to znamená jedno centrální zařízení zvané AP (access point – přístupový bod), ke kterému jsou připojena ostatní zařízení označovaná jako Client. Každé koncové zařízení tedy musí být v dosahu AP. Pro pokrytí budovy domu stačí zpravidla jedno AP v každém podlaží, záleží ale na množství a konstrukci příček. AP má tzv. SSID (Service Set IDentifier), které slouží jako identifikátor sítě pro klienty. Jedno AP může mít více SSID (a poskytovat tak více sítí zároveň) a SSID může být viditelné nebo skryté. Jelikož původní určení Wi-Fi jsou počítačové sítě, kde je požadavek na velmi robustní komunikaci, Wi-Fi v praxi implementuje nejnižší vrstvu TCP/IP stacku. Detailní popis TCP/IP je nad rámec této práce, v principu se ale jedná o model komunikace ve čtyřech vrstvách, kdy nejnižší vrstva (network access) zajišťuje fyzické

spojení a řešení kolizí na přenosovém médiu (zde rádiové vlny), druhá (internet) řeší adresaci a směrování mezi koncovými uzly, třetí (transport) se stará o potvrzování, integritu dat, a adresaci v rámci jednoho uzlu. Nejvyšší vrstva (aplikační) je už tvořena vysokoúrovňovými protokoly, které počítají s tím, že nižší vrstvy plní svou funkci a mohou se tak starat jen o logiku komunikace. Dva nejpoužívanější aplikační protokoly v oblasti smart home jsou HTTP a MQTT, viz následující sekce.

Zabezpečení je řešeno standardem WPA2 (Wi-Fi protected access), u smart home zařízení konkrétně (opět kvůli jednodušší implementaci) WPA2 personal. Přenášená data jsou šifrována podle standardu AES symetrickou šifrou, kdy klíč se spočítá z SSID sítě a hesla označovaného jako PSK (pre-shared key), které, jak název napovídá, musí být předem známé oběma zařízením. Napadnout toto zabezpečení lze pouze hádáním PSK hrubou silou, případně přes WPS. WPS (Wi-Fi Protected Setup) je mechanismus, umožňující připojit nové zařízení k síti bez znalosti PSK. AP je vybaveno tlačítkem, které po stisknutí umožní připojení všem zařízením v dosahu. Tento proces stačí provést při prvním připojení, cílové zařízení si pak síť zapamatuje, stejně jako při běžném způsobu připojení pomocí zadání PSK. WPS bývá aktivní jen několik sekund až minut po stisknutí tlačítka, během této doby ale AP poskytuje všem zařízením svůj šifrovací klíč a pokud je tento odposlechnut útočníkem, lze ho pak použít k rozšifrování další odposlechnuté komunikace. Pro smart home zařízení nemá funkce WPS smysl a pro posílení zabezpečení je vhodné ji v nastavení AP vypnout. PSK může být maximálně 256bitů dlouhý (32 ASCII znaků) a platí pro něj obecná zásada pro hesla – čím delší a komplexnější, tím náročnější je prolomit ho hrubou silou.

Wi-Fi je ve smart home vhodná pro koncové prvky, které potřebují větší datové toky (ovládací terminály, audio…), naopak se nehodí pro zařízení napájená bateriemi, kvůli větší spotřebě zařízení způsobené vyššími nároky na výkon. Pro implementaci v bráně lze buď zvolit SBC s vestavěným Wi-Fi čipem, anebo použít USB adaptér, ten ale zvýší spotřebu energie.

Jak již bylo řečeno, Wi-Fi implementuje jen nejnižší vrstvu TCP/IP komunikace, v případě použití v bráně je tedy nutné zvolit ještě vhodný aplikační protokol. Většina koncových prvků Consumer Smart Home využívá jeden z následujících:

- HTTP – Hyper Text Transfer Protocol je základním kamenem webu. Využívá mechanismus požadavek/odpověď, kdy požadavek má dvě části - metodu (GET nebo POST), ta říká, co se má provést a URL (Uniform Resource Locator), ten udává cestu ke zdroji, na kterém se metoda zavolá. Odpovědí je dokument (zdroj) umístěný na požadované URL. Příklad klasického použití protokolu při prohlížení webu může být GET *http://example.com/test.txt*, což vrátí obsah souboru test.txt umístěného na serveru example.com. Komunikace vždy probíhá mezi dvěma stranami – serverem a klientem.

Existuje ale rozšíření tohoto protokolu zvané REST (REpresentational State Transfer), pomocí kterého lze HTTP efektivně používat pro manipulaci s abstraktními zdroji. Mechanismus, který to zajišťuje se pak nazývá REST API (Application Programming Interface). URL zde popisuje nějaký zdroj, tím může být cokoli (např. soubor na disku nebo data ze senzoru), REST popisuje pouze logiku komunikace, implementaci zdrojů je potřeba vyřešit nějakým programem na webovém serveru, který příslušné REST API poskytuje. Lze si představit např. RGB LED žárovku popsanou jako *http://ustredna.local/obyvak/zarovka*. REST mění význam dvou základních metod GET a POST a definuje další:

- POST – vytvoření nového zdroje
- GET – čtení ze zdroje, např. pokud chceme znát aktuální jas žárovky: GET *http://ustredna.local/obyvak/zarovka/brightness*
- PUT – úprava zdroje, např. pro vypnutí žárovky: PUT *http://ustredna.local/obyvak/zarovka/state/off*
- DELETE – smazání zdroje

- MQTT – Message Queuing Telemetry Transport využívá mechanismus publish/subscribe. Oproti http není komunikace ve formě požadavků a odpovědí mezi klientem a serverem, ale klienti (kterých může být mnoho) si mezi sebou pomocí serveru zvaného broker (ten je jen jeden) předávají zprávy (MQTT message). Zpráva může být jakýkoli textový řetězec, často se používají formáty pro serializaci dat, např. JSON.

  Broker poskytuje klientům kategorie zvané témata (topics), ty mohou být dále dělena na podtémata, téma je popsáno textovým řetězcem, podtémata jsou oddělena lomítkem. Klient může do tématu buď něco zapsat (publish), nebo může téma odebírat (subscribe). Pokud klient odebírá téma, vidí veškeré zprávy, které do něj ostatní klienti zapisují.

  Jako příklad si lze představit téma reprezentující teplotu v ložnici *prostredi/loznice/teplota.* Ventil podlahového vytápění bude téma odebírat a podle jeho aktuální hodnoty se otevírat/zavírat: SUBSCRIBE *prostredi/loznice/teplota* (opět se řeší jen komunikace, ne řízení samotného ventilu). Senzor teploty v ložnici bude naopak do tématu periodicky zapisovat: PUBLISH *prostredi/loznice/teplota* "23 C".

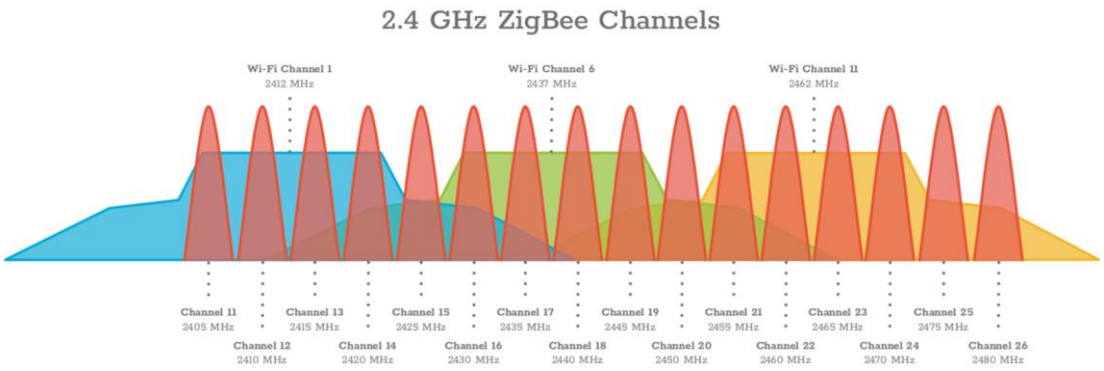
Publish/subscribe mechanismus se pro smart home aplikace dle mého názoru hodí více než požadavek/odpověď, implementace MQTT je také o trochu snažší než u HTTP REST API, protože zdroje není třeba předem definovat, vzniknou automaticky zápisem do tématu. Komponenty brány, u kterých je potřeba zvolit

komunikační protokol (např. senzory) budou tedy využívat MQTT, všechny výše zmíněné open-source ústředny ale podporují oba protokoly.

Zigbee

Otevřený standard, se kterým přišla v roce 2005 asociace Zigbee Alliance, je na rozdíl od Wi-Fi určen pro jednoduchá zařízení, která si vystačí s nižší datovou propustností přenosového kanálu, ale naopak vyžadují velmi nízkou spotřebu energie. Specifikace standardu uvádí minimální výdrž zařízení na baterii dva roky, bez splnění této podmínky nedostane dané zařízení certifikaci a výrobce v něm nemůže Zigbee technologii využít. Tyto vlastnosti dělají ze Zigbee ideální protokol právě pro smart home a automatizaci obecně, kde se přenáší hlavně řídící příkazy a jednoduché datové objekty (např. desetinné číslo – naměřená hodnota ze senzoru).

Zigbee pracuje stejně jako Wi-Fi na frekvenci 2.4Ghz, což může způsobovat problémy, jelikož toto pásmo je zejména ve městech značně zarušené. Jednotlivé kanály jsou ale oproti Wi-Fi mnohem užší a je snažší najít alespoň jeden volný. Kanály a překryv s Wi-Fi jsou znázorněny na Obrázek 5. S užšími kanály je spojena i maximální přenosová rychlost 250kbps, která odpovídá účelu protokolu.



Obrázek 5 - Překryv pásem Wi-Fi a Zigbee

Topologie je tzv. mesh (někdy volně překládáno jako mříž nebo pletivo), kde nezáleží na konkrétní fyzické struktuře a hierarchii sítě, pouze na tom, aby všechna zařízení byla propojena mezi sebou. Některé uzly tak mohou sloužit jako opakovače signálu pro další uzly, které by na sebe jinak navzájem nedosáhly, viz Obrázek 2. Díky tomu stačí při optimálním rozmístění zařízení v budově jen jeden řídící uzel (Coordinator – viz níže). Každé zařízení pracuje v jednom ze tří definovaných operačních módů:

- Coordinator (ZC) je uzel určený jako kořen sítě, zajišťuje řízení sítě, je zodpovědný za optimalizaci grafu meshe po připojení nových uzlů a umožňuje propojení Zigbee infrastruktury se zbytkem smart home. Coordinator

ukládá data o stavu sítě včetně bezpečnostních klíčů, je v síti vždy jediný a při jeho výpadku přestává celá mesh fungovat.

- End Device (ZED) je koncové zařízení které plní pouze svou primární funkci (jako spínač, senzor...). ZED je v grafu meshe vždy listem, komunikuje tedy pouze s rodičovským uzlem (ZR/ZC). Typicky se jedná o bateriově napájené sensory a ovladače, jejichž řídící prvek (mikrokontroler) a komunikační rozhraní jsou většinu času v režimu spánku kvůli šetření energie. Při aktivaci (stisknutí tlačítka), případně periodicky (měření veličiny) se na okamžik probudí (realizováno přerušením procesoru), odešlou data a opět se uspí.
- Router (ZR) plní svou primární funkci stejně jako ZED, ale zároveň funguje jako prostředník pro předávání dat mezi ostatními uzly. Kvůli této funkci musí být router vždy připraven na přenos dat a nemůže využívat režim spánku, většinou tedy bývají napájené přímo z elektrické sítě, např. svítidla. Router může pracovat také jen jako opakovač signálu bez primární funkce, pro vykrytí příliš velké mezery v meshi. Pokud mesh není optimální (nemá redundantní cesty mezi všemi uzly) a router je jediným spojovacím bodem mezi daným segmentem sítě a ZC, přestane při jeho výpadku tento segment fungovat.

Na rozdíl od Wi-Fi specifikuje Zigbee standard vlastní model komunikace se dvěma vrstvami. Síťová vrstva zajišťuje fyzické spojení, řešení kolizí, jednoduchou korekci chyb, adresaci, směrování a je zodpovědná za tvorbu meshe. Aplikační vrstva má dvě části, první z nich je ZDO (Zigbee Device Object), což je protokol zodpovědný za správu daného zařízení, přiřazení operačního módu uzlu, přístupová práva a šifrování. Druhou částí jsou AO (Application Objects), které jsou ekvivalentem aplikačních protokolů u TCP/IP (jako např. výše zmíněné HTTP.). Tady nastává hlavní problém s interoperabilitou koncových prvků Consumer Smart Home. Zatímco ZDO a jeho principy jsou jasně definovány standardem, chování AO definované není a je kompletně v rukou výrobců. Protože spolupráce mezi ekosystémy různých výrobců je z jejich pohledu nežádoucí (z důvodu konkurence), nejsou tyto většinou přímo kompatibilní (příkladem mohou být produktové řady pro osvětlení Phillips Hue a IKEA Tradfri). I když se systémy dají propojit nepřímo pomocí bran, jednotlivé prvky místo jedné společné meshe budou tvořit dvě samostatné, což není optimální a může způsobovat problémy s pokrytím objektu, viz Obrázek 2.

Pro zabezpečení komunikace se využívá opět AES šifrování s klíči o délce 128 bitů. Správu a vydávání klíčů novým uzlům sítě má na starosti coordinator. Zařízení si při prvním zapnutí v dosahu Zigbee meshe zažádá o vstup do sítě v rámci procesu párování. Ten je zpravidla aktivován tlačítkem na těle přístroje. Coordinator mu přidělí klíč podobně, jako při použití WPS u Wi-Fi, síť je tedy při párování zranitelná vůči odposlechnutí klíče. Další komunikace po spárování už je šifrovaná.

Zigbee je vhodný hlavně pro senzory, ovladače, spínací prvky a další jednodušší komponenty systému. Možnosti implementace v bráně jsou:

- Moduly XBee, případně jejich klony, které lze připojit pomocí UART rozhranní. Výhodou je možnost připojení k většině MCU/SBC a velmi malá spotřeba energie, nevýhodou je nutnost od základu implementovat mnoho funkcí nutných pro provoz brány. Pořizovací cena je cca 400 Kč.
- CC2531 – Původně určen jako debugger Zigbee sítí, pro tento modul od Texas Instruments je dostupný alternativní firmware, který z něj umožňuje vytvořit buď coordinator nebo router. Existuje pro něj i open source projekt Zigbee2MQTT, umožňující snadnou integraci s open-source ústřednami popsanými výše. Modul se připojuje pomocí USB rozhraní. Pořizovací cena je cca 200 Kč.
- Phoscon ConBee/RaspBee – modul coordinatoru firmy Phoscon (Dresden Elektronik) přímo určený jako univerzální hub pro smart home zařízení. Nabízí velmi propracované webové rozhraní včetně REST API a podporuje mnoho Consumer Smart Home produktů bez nutnosti nastavení. Software je ale proprietární, modul vyžaduje připojení buď přes USB (verze ConBee) nebo přes GPIO header na SBC (verze RaspBee). Ovladače jsou rovněž proprietární a jsou určeny jen pro Raspberry Pi, na jeho klonech fungují buď omezeně (Orange Pi) nebo vůbec (Banana Pi, La Frite). Pořizovací cena je cca 800 Kč.

Z-Wave

Z-Wave má se zigbee podobný nejen název, ale i většinu svých vlastností. Jedná se o síť s mesh topologií, nízkou datovou propustností a důrazem na nízkou spotřebu energie u zařízení. Z-Wave je proprietární standard firmy Sigma Designs s jednoznačně definovanou komunikací na všech vrstvách. Výroba čipů pro Z-Wave je omezena licencí, kterou aktuálně vlastní pouze Silicon Laboratories, Inc., mateřská společnost Sigma Designs. To na jednu stranu zaručuje maximální interoperabilitu Z-Wave zařízení, na stranu druhou to znamená, že každý výrobce si kromě Z-Wave hardware musí od Silicon Labs koupit i licenci pro použití tohoto protokolu. Z-Wave výrobky jsou kvůli tomu dražší než jejich Zigbee ekvivalenty. Kvůli této uzavřenosti je vývoj a budoucnost standardu výhradně v rukou Silicon Labs.

Podle své první specifikace využívala Z-Wave frekvenci 908.42 MHz, což je bezlicenční pásmo v Americe, kde byla specifikace vytvořena. V ostatních částech světa je ale pásmo licencované, takže pracovní frekvence závisí na konkrétním regionu nebo zemi. Na většině území Evropy včetně ČR se využívá pásmo 868,42 MHz, v různých dalších zemích pak různé frekvence z rozsahu 865-926 MHz (země se specifickými frekvencemi jsou např. Čína, Japonsko, Rusko, Indie,

Jižní Korea a další). Přenosová rychlost je 100kbps, ve srovnání se Zigbee méně než poloviční, přitom spotřeba elektrické energie je u Zigbee zařízení nižší.

Centrální uzel sítě se nazývá Controller, který funkcemi odpovídá Zigbee Coordinatoru s tím rozdílem, že síť může mít jeden primární a jeden sekundární. Ostatní uzly se označují jako slave a jsou rovnocenné. Ve srovnání se Zigbee má síť pevný limit 232 uzlů, Zigbee má teoretický limit 65000, v praxi ale saturuje své přenosové pásmo mnohem dříve, než tohoto počtu zařízení dosáhne. Z-Wave má vyšší maximální dosah, až 90m při přímé viditelnosti a lépe prochází překážkami (díky nižší frekvenci). Z-Wave využívá rovněž 128-bitové AES šifrování, má ale vylepšený proces rozšiřování sítě. Každé zařízení má unikátní PIN, který je nutno zadat na controlleru pro přidání zařízení do sítě. Tento PIN je pak použit jako klíč při prvním šifrovaném spojení. Po navázání tohoto spojení teprve controller předá koncovému prvku nový, přidělený klíč, který bude použit pro další komunikaci. Ten tedy není odposlechnutelný ani při párování.

Vzhledem k podobnostem se Zigbee má smysl v bráně implementovat pouze jeden z těchto protokolů. I když Z-Wave protokol ve srovnání se Zigbee několik nesporných výhod, dle mého názoru jeho nevýhody převažují a výrobků na trhu postupně ubývá (v době psaní této práce má největší český e-shop s elektronikou v nabídce 14 Z-Wave zařízení a přes 150 Zigbee zařízení. S produkty ze zahraničí je pak problém právě kvůli různým frekvenčním pásmům), brána tedy bude podporovat pouze Zigbee.

Bluetooth

Původní účel tohoto protokolu, jehož první specifikace vznikla již v roce 1998 (dříve než USB, viz níže), bylo nahradit sériový port RS-232 při připojení mobilního telefonu k PC, byl tedy určen pro komunikaci na velmi krátké vzdálenosti s relativně vysokými datovými toky a nízkou spotřebou energie. I když od té doby prošel Bluetooth mnoha změnami, tyto tři hlavní rysy si stále zachoval. Jedná se o otevřený standard, jeho vývoj a certifikace jsou zastřešovány organizací BT SIG (Bluetoooth Special Interest Group).

V současné době nejpoužívanější verze v oblasti smart home je Bluetooth 4, resp. jeho podkategorie BLE (Bluetooth Low Energy). Všechny verze Bluetooth operují v pásmu 2,4 GHz, BLE definuje 40 kanálů, každý o šířce 2MHz, maximální přenosová rychlost je 1Mbps, přičemž spotřeba energie je ještě menší než u Zigbee. BLE zařízení dosahuje asi dvou třetin spotřeby ekvivalentního Zigbee zařízení [10]. Stejně jako u Zigbee je zde využíván režim spánku, ve kterém se zařízení nachází po většinu provozní doby a buď periodicky nebo na základě vnější události se na krátkou dobu probudí, odešlou data a opět se uspí. Některá BLE zařízení aktivují režim spánku i v mezerách mezi odesíláním jednotlivých datových paketů. Teoretický dosah je specifikován na 100 m s přímou viditelností, reálně se ale pohybuje v rozmezí 10-30 m, je třeba také počítat s velkým zarušením pásma 2.4GHz, stejně jako u Wi-Fi a Zigbee.

Základní topologie je hvězdicová, kde jedno zařízení se nazývá *centrální* a ostatní k němu připojené jsou *periferní*. Detaily topologie se ale liší podle tzv. provozního profilu, který je definován případem užití. Standard prošel lety vývoje a každá nová verze se snaží být zpětně kompatibilní, specifikace je tedy velmi rozsáhlá a aktuálně definuje 36 provozních profilů. Jedním z nejstarších je SPP (Serial Port Profile) – profil pro výše zmíněnou emulaci sériového portu, z novějších je to pak např. profil pro streaming multimédií GAVDP (Generic Audio and Video Distribution Profile). Podkategorie BLE pak definuje dalších 19 profilů, od univerzálních jako je MESH profil pro vytvoření sítě s mesh topologií mezi BLE zařízeními, po velmi specifické jako je třeba BLP (Blood Pressure Profile) – profil pro zařízení měřící krevní tlak. Profilů je tedy mnoho a každý z nich v podstatě definuje nový komunikační protokol (ekvivalent aplikačního protokolu u TCP/IP modelu). Aby bylo Bluetooth zařízení kompatibilní s ostatními, musí v něm jeho výrobce implementovat určitou podmnožinu těchto profilů, která záleží na požadované funkci zařízení.

BLE není typický smart home protokol. Existují sice BLE senzory, např. dle mého názoru velmi povedený senzor teploty a vlhkosti Xiaomi Mija, ale jejich Zigbee ekvivalenty je ve všech ohledech kromě spotřeby energie předčí. Jedna z hlavních aplikací BLE je komunikace s tzv. wearables (volně přeloženo jako nositelná elektronika), která je určena hlavně jako příslušenství ke smartphone a patří mezi ně chytré hodinky, fitness náramky, zařízení pro monitoring spánku atd. Druhou nejčastější aplikací jsou pak tzv. BLE Beacons (majáky), což jsou jednoúčelová zařízení, jejichž jedinou funkcí je vysílat v pravidelných intervalech tzv. advertising packet, který obsahuje unikátní ID majáku a lokalizační informace v něm uložené (např. ID budovy, podlaží, předmětu etc.). Jejich aplikace ve smart home je označování předmětů, které se pak dají pomocí systému lokalizovat, případně spouštět různá automatizační pravidla na základě pohybu předmětů po domácnosti.

Z pohledu bezpečnosti má BLE dva režimy komunikace. Prvním z nich je režim *broadcast*, kdy zařízení vysílá data otevřeně pro všechna ostatní zařízení v dosahu, zde není zabezpečení žádné. Druhý režim je podobný jako u Zigbee, zařízení musí být před komunikací spárována a komunikace mezi nimi je šifrovaná. Implementace v protokolu 4.0 ale obsahuje i stejnou zranitelnost jako Zigbee, kdy během procesu párování lze jednoduše odposlechnout přenášené šifrovací klíče. Verze 4.2 ale párovací proces zdokonaluje a zavádí Diffie–Hellmanův algoritmus pro výměnu klíčů, pokud tedy zařízení podporují BLE 4.2 a jsou správně nastavena pro nový způsob párování, lze přenos dat považovat za bezpečný.

Možnosti pro implementaci protokolu v bráně jsou následující:

- I^2C modul, např. NRF51822 – tyto moduly jsou ale určeny pro funkci v režimu periferie, jako centrální uzel mohou fungovat nestabilně nebo dokonce vůbec.

- Mikrokotroler ESP32, který má nativní BLE 4.2, připojený k SBC přes UART rozhraní
- Nativní BLE rozhranní přímo v některém z SBC
- USB Bluetooth adaptér

433 MHz

Rádiová komunikace na frekvenci 433MHz je nejstarší technologií používanou v této oblasti. Nejedná se o standard ani protokol, pouze o bezlicenční pásmo dlouhodobě využívané pro nejrůznější bezdrátové ovládání, ať už s analogovou nebo digitální komunikací.

Pásmo je rozděleno do 69 kanálů po 25 kHz, a to od 433.075 MHz do 434.775 MHz. V Evropě smí vysílače v tomto pásmu mít pouze malou, integrovanou anténu a maximální výstupní výkon 10mW. Tato frekvence umožňuje komunikaci na velkou vzdálenost, ta je ale omezena právě malým povoleným vysílacím výkonem a pohybuje se kolem 100 m při přímé viditelnosti. Rychlost komunikace je zhruba 5 kbps, dostatečná pro přenos jednoduchých ovládacích příkazů, případně malých objemů dat (např. hodnota teploty a vlhkosti ze senzoru).

Samotný komunikační protokol zcela závisí na konkrétním připojeném zařízení a je definován jeho výrobcem, specifikována je jen holá fyzická vrstva komunikace a je nutné implementovat veškeré vyšší funkce, jako je detekce a oprava chyb při přenosu, potvrzování, obsluha více zařízení připojených zároveň, šifrování komunikace atd. Kromě toho musí protokol samozřejmě řešit i samotnou logiku předávání a zpracování dat/příkazů. Veškeré použité algoritmy a protokoly jsou tedy proprietární a pokud chce uživatel integrovat takové zařízení do svého smart home, je třeba protokol „rozluštit“ pomocí reverzního inženýrství. Pro mnoho oblíbených zařízení je na webu k dispozici (často neoficiální) dokumentace.

Nejčastější využití této technologie jsou dálkové ovladače k automobilům (u nich výrobci implementují složitější zabezpečení, od plovoucího kódu po challenge-response schéma, kdy ovladač pošle požadavek, automobil odpoví výzvou, na kterou ovladač použije nějaký definovaný algoritmus a jeho výsledek vrátí automobilu k vyhodnocení). Ve smart home jsou to pak ovladače svítidel, garážových vrat, bezdrátové zvonky a mnoho domácích meteostanic.

Tato technologie je postupně nahrazována za novější standardy, kvůli její jednoduchosti a nízké ceně ji však stále využívá mnoho nových zařízení. Pro implementaci v bráně je k dispozici velké množství jednoduchých RF modulů pro toto pásmo, všechny pracují téměř stejně a jejich komunikační rozhraní tvoří jediný GPIO pin. Pro obousměrnou komunikaci jsou potřeba dva moduly – vysílač a přijímač.

GSM

GSM (Global System for Mobile Communications) je dnes obecné označení pro skupinu rádiových pásem používaných pro přenos hlasu a dat v mobilních sítích.

Původní GSM specifikace reprezentuje druhou generaci mobilních sítí (2G) a byla vytvořena Evropským institutem telekomunikačních standardů (ETSI) jako globální standard pro digitální mobilní sítě, který by sjednotil dosavadní technologie. Od té doby se začleněním mnoha dalších technologií vyvinul do současné 4G/5G verze.

Rozdíl oproti ostatním technologiím v této sekci je, že GSM využívá infrastrukturu třetí strany – síť mobilního operátora, díky tomu má tato komunikace globální dosah (za předpokladu že uživatel i budova jsou v oblasti pokryté GSM signálem). Pro přístup do sítě slouží SIM karta (Subscriber Identity Module), na které je uloženo mimo jiné unikátní identifikační číslo. Po vložení SIM karty do mobilního zařízení a zadání PIN klíče lze toto číslo přečíst a použít pro registraci do sítě. Po registraci lze využívat služby poskytované operátorem (volání, SMS a datové přenosy). V rámci Smart Home systému jsou nejužitečnější SMS, např. pro zasílání upozornění nebo příkazů, pokud není k dispozici spojení do internetu.

Místo samostatného mobilního zařízení bude k bráně připojen modem, pro SMS stačí některý z jednodušších čipů, např. SIM800L výrobce SIMCom. Modemy se standartně připojují přes sériové rozhraní (USB nebo UART) a ovládají se pomocí AT příkazů (AT je zkratka pro „attention“). Tyto příkazy jsou standardizovaným způsobem ovládání pro všechny druhy modemů a zadávají se jako textový řetězec

AT<příkaz> <parametry> [CRLF].

CR (carriage return) a LF (line feed) jsou speciální znaky reprezentující odřádkování. Pokud příkaz začíná *AT+*, jde o rozšířený (extended) AT příkaz, který může být specifický pro konkrétní modem (na jiném nemusí fungovat) [9]. Seznam všech dostupných příkazů lze vždy najít v dokumentaci modemu (datasheetu čipu).

Ethernet

Ethernet je drátové rozhraní a nebude sloužit pro připojování koncových prvků, ale pro připojení brány do stávající domácí IP sítě a do internetu.

Jedná se o skupinu protokolů popsanou standardem IEEE 802.3 a určenou původně pro komunikaci v lokálních počítačových sítích (LAN). Standard vznikl už v roce 1983, nejstarší verze využívaly koaxiální kabel, časem se přešlo na kroucenou dvoulinku (4 páry vodičů) a optická vlákna.

Ethernet je v současnosti prakticky jediný používaný standard pro metalickou a optickou LAN. Různé verze mají různé přenosové rychlosti, nejrozšířenější jsou Fast Ethernet (100BASE-T) s rychlostí 100 Mbps a Gigabit Ethernet (1000BASE-T) s rychlostí 1 Gbps. Dosah je při použití specifikované kabeláže až 100 m. Ethernet má hvězdicovou topologii, jednotlivá koncová zařízení jsou připojena do centrálního propojovacího prvku zvaného switch (přepínač). Koncová zařízení jsou si rovnocenná a každý uzel může komunikovat s kterýmkoli jiným,

uzly jsou identifikovány pomocí unikátní 48-bitové MAC (Media Access Control) adresy.

Základní specifikace Ethernetu neposkytuje možnost připojené zařízení napájet, proto existuje několik rozšiřujících standardů souhrnně označovaných jako PoE (Power over Ethernet). U starších implementací pro 100BASE-T, který využívá k přenosu dat pouze dva páry pro bylo zařízení napájeno po zbývajících dvou volných párech. U 1000BASE-T a novějších standardů už ale musí data a napájení sdílet stejné vodiče. Ethernet na kroucené dvoulince pracuje s diferenciální signalizací, oba vodiče v páru nesou stejný potenciál, ale každý s opačnou polaritou, výsledný signál je dán rozdílem těchto potenciálů. Na rozdíl od běžnější nesymetrické signalizace nemusí tedy dvě zařízení propojená pomocí Ethernetu mít společný referenční potenciál (zem) a je vhodné je od sebe galvanicky izolovat. To je řešeno oddělovacími transformátory na obou stranách vedení viz Obrázek 7. Toto řešení navíc poskytuje určitou míru ochrany před závadami na vedení. U zařízení podporujících PoE jsou u primárních vinutí oddělovacích transformátorů (strana vedení) vyvedeny středy a k těm je připojeno napájení. Napájecí napětí je stejnosměrné, jeden pár tedy slouží jako plus a druhý jako minus (konkrétní páry závisí na použitém standardu). Stejnosměrná složka neprojde skrz oddělovací transformátor, do vnitřních obvodů zařízení se tedy dostane jen datový signál [20]. V samotném vedení se napájení (stejnosměrné) a signál (s frekvencemi v řádu stovek MHz) navzájem neovlivní, tento princip je označován jako fantómové napájení. Zdrojem napájecího napětí (také zvaný PSE – Power Sourcing Equipment) je buď přímo switch s podporou PoE, nebo PoE injektor napájený externě.

PoE je možné implementovat dvěma způsoby:

- Pasivní – nevyužívá žádné aktivní řízení, napájení je v injektoru/switchi jednoduše připojeno na stanovené páry vodičů. V koncovém zařízení jsou pak středy oddělovacích transformátorů opět přímo připojeny k napájecímu vstupu. Pasivní PoE se neřídí žádným standardem, parametry jako napájecí napětí a maximální přenášený proud jsou dány výrobcem koncového zařízení. Dvě různá zařízení s pasivním PoE tedy nemusí být kompatibilní a při nesprávném zapojení může dojít k poškození některého z nich.
- Aktivní - standardizované jako rozšíření Ethernetové specifikace IEEE 802.3af/at/bt využívá stejné základní principy, napájení je ale implicitně vypnuto. Po připojení koncového zařízení je zahájen proces, pomocí kterého si PoE switch nebo injektor ověří, zda koncové zařízení napájení potřebuje nebo ne a pokud ano, vyjedná správné parametry (PoE handshake). Tím se eliminuje možnost poškození koncových zařízení, která nejsou na PoE stavěna, na obou stranách je ale potřeba dodatečná řídící logika, která toto zajistí. Napětí je standardem stanoveno na 48 V [20], maximální přenášené výkony jsou vidět v tabulce:

Název	Standard	Max. výkon	Využité páry
PoE	IEEE 802.3af	15.4 W	2
PoE+	IEEE 802.3at	30 W	2
PoE++	IEEE 802.3bt (Type 3)	60 W	4
PoE++	IEEE 802.3bt (Type 4)	100 W	4

Senzory

Kromě rozhraní pro připojení koncových zařízení bude brána integrovat senzory pro měření různých parametrů prostředí. Díky tomu nebude nutné v místnosti, kde je brána umístěna instalovat další koncová zařízení s touto funkcí.

Senzory (také snímače nebo čidla) slouží jako vstupní prvky řídícího systému, a poskytují mu informace o jeho okolí. Senzor mění nějakou pozorovanou fyzikální veličinu na elektrický signál, který lze dále zpracovat. Senzory se dělí podle měřené veličiny, fyzikálního principu, způsobu připojení atd. Nejčastěji zkoumanými parametry prostředí v budovách jsou teplota a relativní vlhkost vzduchu, intenzita osvětlení a kvalita vzduchu (hodnocená na základě koncentrace určitých plynů, např. CO_2).

Teplota

Základní veličina, kterou je nutno znát pro řízení vytápění a chlazení. Jako senzor lze použít:

- Termistor – polovodič, který mění svůj odpor v závislosti na teplotě. Jako jednoduché, pasivní součástky jsou termistory velmi levné, ale pro měření teploty potřebují pomocný obvod, který změnu odporu převede na změnu napětí a AD převodník, který získané napětí převede na digitální hodnotu. Termistor je také nutné ručně zkalibrovat.
- Termočlánek – skládá se ze dvou elektrod, jejichž konce jsou spojeny a vyrobeny ze specifických kovů (u nejběžnějšího K termočlánku to jsou slitiny alumel a chromel), které společně vykazují termoelektrický jev (při zahřívání jejich spoje vzniká elektrické napětí úměrné teplotě). Pro měření je opět potřeba pomocný obvod (zde je to jen zesilovač) a ADC, a i zde je nutná ruční kalibrace.
- LM35 – Integrovaný analogový senzor od Texas Instruments, je postaven na K termočlánku, výstupem je ale napětí přímo mapovatelné na měřenou teplotu, kdy 10mV odpovídá 1°C. Senzor je z výroby zkalibrován, měří od -55 °C do 150 °C s přesností ±0.5 °C napájecí napětí může být v rozmezí of 4 V do 20 V. Jelikož výstupem je napěťová hodnota, je i zde potřeba ADC.

- DS18b20 – Integrovaný digitální měřicí modul firmy Dallas Semiconductor, opět postaven na K termočlánku. Senzor je z výroby zkalibrován, měří od -55 °C do 125 °C s přesností ±0.5 °C, napájecí napětí může být od 3 V do 5,5 V. Modul kromě všech pomocných obvodů obsahuje i 12-bitový ADC, s MCU/SBC komunikuje pomocí sběrnice 1-Wire. Cena modulu je kolem 50 Kč, stejně jako cena LM35.

Vlhkost

Senzory vlhkosti jsou buď kapacitivní (změna kapacity mezi dvěma prostorově vhodně uspořádanými elektrodami v závislosti na obsahu vodní páry ve vzduchu) nebo rezistivní (stejný princip, ale mění se odpor). Na rozdíl od termistorů a termočlánků potřebují složitější pomocný obvod (měří se pomocí střídavého napětí), běžně tedy nejsou dostupné samostatně, ale jako součást digitálních měřicích modulů. Tyto moduly většinou kromě vlhkosti měří zároveň i teplotu, senzory teploty v nich ale bývají méně přesné než ty dedikované.

- DHT11/DHT22 – Integrované digitální měřicí moduly firmy Aosong s obvodem AM2302. Verze DHT11 má menší rozměry, vyšší dotazovací frekvenci, ale nižší přesnost. Verze DHT22 je naopak větší, pomalejší (dotazovat hodnoty lze jen jednou za 2 sekundy, pro použití ve smart home je to dostačující), ale přesnější. U DHT22 je rozsah měřených teplot -40 °C až 80 °C s přesností ±0.5 °C. Měření vlhkosti je v rozsahu 0-100 %, ale jen v rozsahu 15-90 % je garantována přesnost ±2 %. Napájecí napětí může být od 3 V do 5,5 V, je třeba dát pozor na to, že horní hladina napětí datové sběrnice je stejná jako napájecí napětí. MCU/SBC komunikuje po jednovodičově digitální sběrnici, která je specifická pro tento senzor, díky jeho popularitě u DIY Smart Home systémů jsou ale dostupné knihovny pro nejrůznější platformy a jazyky včetně Pythonu, který se nejčastěji používá pro komunikaci s nízkoúrovňovými periferiemi pomocí GPIO pinů SBC.
- SHT71 – Modul firmy Sensirion velmi podobný DHT, měří teplotu od -40 do 90 °C s přesností ±0.4 °C a relativní vlhkost 0-100% s přesností ±3% na celém rozsahu. Napájecí napětí je od 2,4V do 5,5V, komunikační rozhraní je zde sběrnice I^2C.

Osvětlení

Úroveň osvětlení v místnosti je užitečné znát pro nejrůznější automatizace svítidel, ať už jednoduché, jako např. soumrakový spínač nebo pokročilejší, jako např. cirkadiánní osvětlení.

- Fotoresistor – Ekvivalent termistoru, tedy polovodič, který mění svůj odpor v závislosti na množství dopadajícího světla. Levný, jednoduchý, ale potřebuje pomocný obvod a ADC. Opět je nutné měřicí obvod po zapojení zkalibrovat.

- BH1750 – Integrovaný digitální modul firmy Rohm Semiconductor, který obsahuje fotodiodu, pomocný obvod, zesilovač a 16-bitový ADC. Poskytuje aktuální hodnotu intenzity osvětlení v luxech od 0 do 65535 (2^{16}), rozlišní je 16 bitů, nejmenší krok je tedy 1 lx. napájecí napětí je 3.3 V, modul komunikuje po sběrnici I^2C.

Kvalita vzduchu

Kvalita vzduchu podává informaci o tom, zda je třeba v budově vyvětrat, případně zda došlo k úniku nebezpečných plynů. Měří se na základně obsahu různých plynů ve vzduchu, v obytných prostorech je to zejména CO_2. Udává se jako hodnota ppm (parts per million), která vyjadřuje počet částic složky (CO_2) na jeden milion částic směsi (vzduchu). Tato koncentrace se měří pomocí elektrochemických nebo optických senzorů.

- MQ135 –levný a relativně jednoduchý elektrochemický senzor, obsahuje tenkou vrstvu SnO_2, jejíž odpor se mění s koncentrací měřeného plynu, zde je to kromě CO_2 ještě amoniak, benzen a propan-butan. Nevýhoda je, že nelze zvolit který z plynů chceme měřit a také to, že na CO_2 je senzor citlivý nejméně. Výhodou ale je, že lze senzor zároveň použit k detekci úniku propan-butanu (amoniak a benzen se v domácnostech běžně nevyskytují), právě kvůli rozdílu citlivostí. Senzor měří koncentrace cca od 10 do 5000 ppm s přesností ±20ppm. Napájecí napětí je 5 V, a jelikož senzor obsahuje topný prvek (měření je nejpřesnější při teplotě měřicí vrstvy kolem 50 °C), má ve srovnání s ostatními velký proudový odběr (150 mA při 5V). Senzor je analogový a potřebuje tedy navíc ADC. Napětí měřicího výstupu je přímo úměrné koncentraci plynu. Senzor je nutno při prvním použití nechat „zahořet“ (připojení topného prvku k napájení) po dobu 24 h a následně ručně zkalibrovat. Kalibrace je poměrně složitá, jelikož se jednotlivé kusy nepatrně liší v množství SnO_2 a je potřena nejdříve najít referenční hodnotu.
- Plantower DS-CO2-20 – Digitální modul s optickým senzorem na principu NDIR (non-dispersive infrared absorbtion) – senzor obsahuje komoru, kde na jedné straně je zdroj světla a optický filtr, skrz který projde jen úzké spektrum vlnových délek. Toto spektrum je zvoleno tak, aby ho plyn, jehož koncentraci chceme měřit co nejvíce pohlcoval. Fotodioda na druhé straně komory pak měří množství světla, které prošlo (nebylo pohlceno) a tím pádem nepřímo i koncentraci plynu. Tyto senzory jsou přesnější a reagují rychleji než elektrochemické, jejich cena je ale několikanásobně vyšší. Tento modul je napájen napětím 5V a komunikuje po UART rozhranní. Měří koncentrace 400-4000 ppm s přesností ±20ppm.

Návrh a realizace

Volba komponent a návrh brány se řídí následujícími podmínkami, z nichž některé jsou dané zadáním této práce, jiné se snaží adresovat problémy zmíněné v sekci Možná budoucnost. Důraz je také kladen na dustupnost brány, a to jak cenově, tak konstrukčně (zařízení by mělo být možné sestrojit v domácích podmínkách).

- Podpora komunikace přes Wi-Fi, Zigbee, Bluetooth, RF 433 a GSM
- Webové ovládací/konfigurační rozhraní
- Provoz v lokální síti – pro vlastní funkci Smart Home systému není nutné internetové připojení
- Softwarové komponenty na sobě budou co nejméně závislé z důvodu zjednodušení případných změn nebo rozšíření
- Z důvodu podpory a údržby softwarových komponent bude co možná nejvíce funkcí implementováno s použitím existujících open-source projektů
- Deployment script/package/image – jednoduché nasazení kompletního software brány pomocí skriptu, balíčku nebo obrazu disku
- Součástí návrhu brány bude ukázková realizace hardware

Raspberry Pi

Jako řídící prvek byl zvolen SBC Raspberry Pi 3B+ s operačním systémem Raspbian GNU/Linux (dále jen Linux). Hlavními důvody byla nativní konektivita Wi-Fi a Bluetooth, vhodný poměr mezi výkonem a spotřebou energie a ve srovnání s konkurencí nejlepší softwarová podpora. Raspberry Pi 3B+ má taky velmi dobře řešené PoE napájení (viz níže). GPIO header Raspberry Pi 3B+ poskytuje řadu nízkoúrovňových rozhraní včetně SPI, I2C, 1-Wire a UART.

Pro první spuštění je potřeba kromě samotného SBC ještě Micro SD karta s obrazem operačního systému a 5V napájecí adaptér. Pro napájení postačí adaptér schopný dodat proud 2A. SD karta se u Raspberry Pi využívá jako hlavní systémový disk a tím pádem je vytížena velkým množstvím náhodných přístupů, je tedy nutné použít kartu alespoň třídy A1 (Application performance class 1), která je pro tuto zátěž optimalizována a umožňuje až 1500 operací náhodného čtení, případně 500 náhodného zápisu za sekundu. Použití běžné karty, která je optimalizována pro sekvenční čtení/zápis, způsobí výrazné zpomalení systému z důvodu čekání na vstupně/výstupní operace a několikanásobně rychlejší opotřebení karty. Optimální kapacita je 32 GB, samotný OS zabere cca 8 GB.

Nejprve je třeba nainstalovat OS Raspbian. Po stažení oficiálního obrazu Raspbian Lite z webu *https://www.raspberrypi.org/downloads/raspberry-pi-*

os/ je třeba ho dostat na SD kartu. K tomu byl použit PC s OS Linux a nástroj pro blokovou kopii *dd*:

dd bs=4m if=./raspbian_10_lite.img of=/dev/sdb

parametr *if* udává cestu ke staženému obrazu a *of* cestu k SD kartě. Apple MacOS rovněž obsahuje příkaz *dd*, pod OS Microsoft Windows lze použít např. nástroj Balena Etcher. Prvotní nastavení lze provést dvěma způsoby, buď k SBC připojit klávesnici a monitor a nastavit lokálně, nebo nastavit po síti pomocí vzdáleného přístupu přes SSH. Druhý způsob je vhodnější, protože SSH přístup bude potřeba i při další konfiguraci, SSH je ale nutné nejdříve povolit editací konfiguračních souborů na připravené SD kartě. Karta je po nakopírování staženého obrazu naformátována se systémem souborů Ext4 určeným pro Linux, pod jinými OS je pro přístup k souborům nutné doinstalovat podporu Ext4. Pod OS linux je nutné se pro přístup přepnout na administrátorský uživatelský účet *root*. Prvním konfiguračním souborem je */etc/ssh/sshd_ config*, kde je nutné upravit následující (názvy direktiv dobře vysvětlují jejich funkci):

ChallengeResponseAuthentication yes
UsePAM yes
PermitRootLogin no

Dále bylo potřeba v souboru */etc/network/interfaces*, nastavit pevnou IP adresu, masku podsítě, výchozí bránu a DNS server. Raspbian označuje všechna ethernetová rozhraní jako *enoX*, kde X je pořadové číslo přidělené při startu systému. Vestavěná síťová karta má vždy přidělen identifikátor *eno1*, první direktiva zapíná autodetekci spojení, druhá určuje statickou definici adresy:

auto eno1
iface eno1 inet static
    address 192.168.10.210
    netmask 255.255.255.0
    gateway 192.168.10.200
    dns-nameservers 8.8.8.8

Posledním krokem je symbolický link (odkaz) pro zapnutí systemd služby (viz níže) pro ssh server

ln -s /lib/systemd/systém/ssh.service /etc/systemd/systém/multi-user-target.wants/ssh-service

Tím je SD karta připravena a je možné ji vložit do SBC. Po připojení k síti ethernetovým kabelem, zapnutí a startu systému bude na nastavené IP adrese k dispozici SSH služba. Defaultní uživatel je *pi* s heslem *raspberry*. Pod OS linux a MacOS lze použít příkaz *ssh*:

ssh pi@192.168.10.250

Pro OS windows je k dispozici open-source SSH klient Putty. Po úspěšném připojení je zobrazen příkazový řádek shellu *bash*. Je třeba se přepnout na účet *root* pomocí příkazu *sudo su* - a pokračovat v konfiguraci:

```
echo NoTGW > /etc/hostname
mv /home/pi /home/gwadmin
usermod -l gwadmin –d /home/gwadmin pi
apt-get update && apt-get upgrade
touch /etc/notgw.conf
```

První příkaz nastaví jméno systému na *NoTGW* (Network of Things Gateway), to lze provést i ručně editací souboru */etc/hostname*. Druhý a třetí změní jméno defaultního uživatele z *pi* na *gwadmin* včetně úpravy domovského adresáře. Dále je provedena aktualizace OS a vytvořen konfigurační soubor brány */etc/notgw.conf*, který bude sloužit pro centrální nastavení parametrů jednotlivých komponent.

Uživateli *gwadmin* je vhodné změnit defaultní heslo interaktivním příkazem *passwd gwadmin*. Jako bezpečnostní opatření je také vhodné vypnout protokol IPv6, který ve Smart Home síti nebude využíván. Na konec konfiguračního souboru */etc/sysctl.conf* je třeba připsat direktivy:

```
net.ipv6.conf.all.disable_ipv6=1
net.ipv6.conf.default.disable_ipv6=1
```

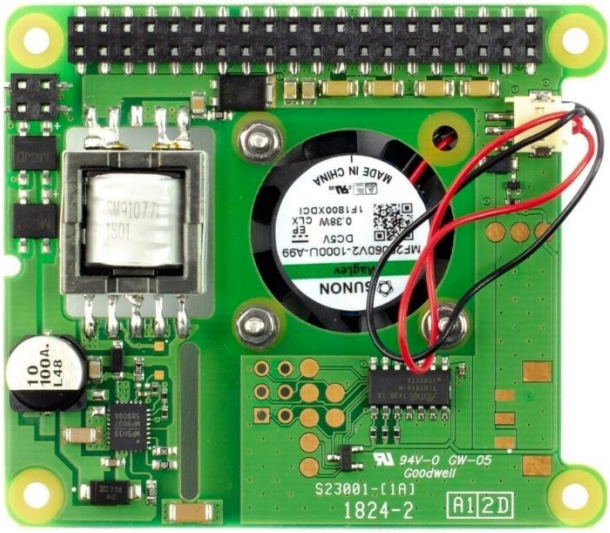
Posledním krokem je rozšířit systém souborů na celou SD kartu spuštěním nástroje *raspi-config* a volbou prvního příkazu „Expand Filesystem“.

Ethernet

Raspberry Pi 3B+ je vybaven čipem Microchip LAN7515, který poskytuje 1Gbps přenosovou rychlost, s procesorem je ale propojen přes USB2.0, a je tak omezen maximální rychlostí této sběrnice – 300Mbps. To je dostatečné pro všechny Smart Home aplikace včetně streamování multimédií.

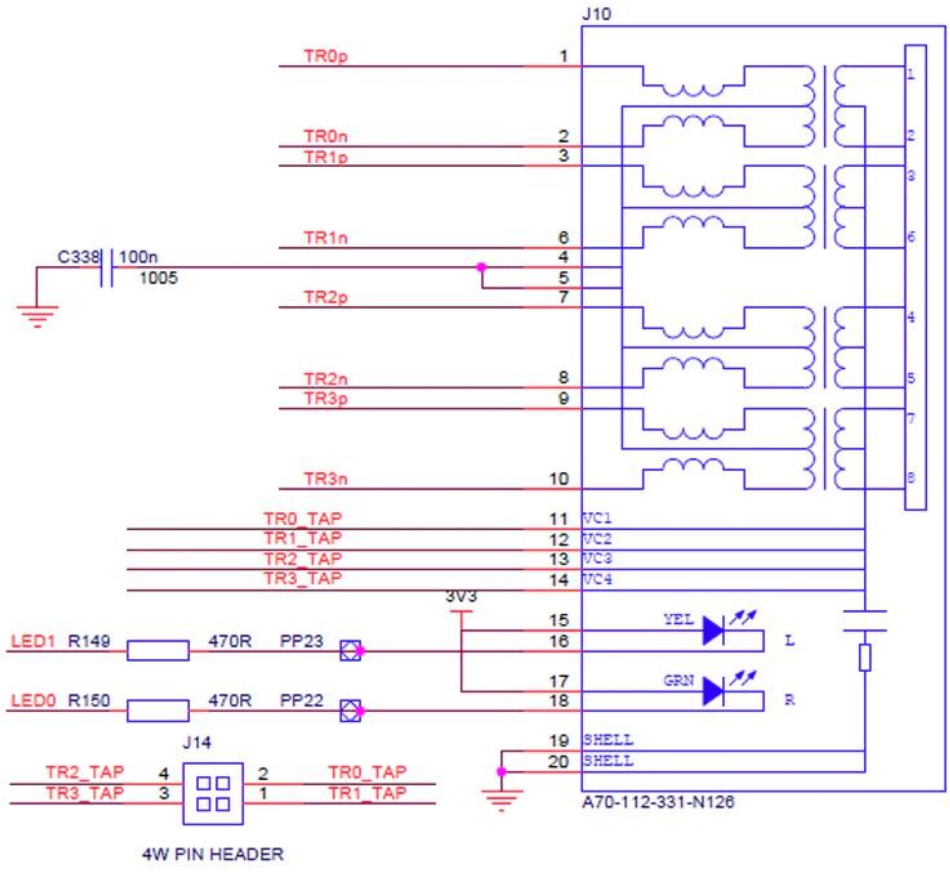
PoE

Model 3B+ je první Raspberry Pi deska, která integruje PoE napájení podle standardu IEEE 802.3af. Deska má vedle GPIO headeru vyvedeny 4 piny, které jsou připojeny na středy oddělovacích transformátorů Ethernetového rozhraní, schéma je na Obrázek 7. Samotný napájecí obvod se dodává zvlášť ve formě tzv. PoE HATu (HAT je označení pro rozšiřující hardwarové moduly k deskám Raspberry Pi, které jsou určeny k nasazení na GPIO header), ten obsahuje izolovaný DC/DC spínaný zdroj a logiku zajišťující PoE handshake, navíc obsahuje ventilátor pro chlazení procesoru.



Obrázek 6- Raspberry Pi PoE HAT

Kvůli několika problémům však tento modul nebyl použit. Jak je vidět z Obrázek 6, PoE HAT svými rozměry pokrývá celou plochu SBC včetně GPIO headeru, ten navíc na jeho desce nemá průchozí konektor a bez jeho úpravy není možné header použít. Více než polovinu desky PoE HATu navíc zabírá ventilátor nebo volné místo. Pořizovací cena je cca 600Kč, což je více než polovina ceny samotného Raspberry Pi 3B+.



Obrázek 7- schéma zapojení ethernetového rozhraní u Raspberry Pi 3B+

Jako alternativa bylo hlavně kvůli jednoduchosti a ceně zvoleno pasivní PoE s napětím 24V. Jako PSE (Power-Sourcing Equipment) je vhodný jakýkoli pasivní injektor (použit byl Mikrotik RBGPOE), na straně Raspberry Pi jsou pak příslušné piny PoE headeru připojeny na vstup DC-DC konvertoru (TR1_TAP – GND a TR3_TAP - VCC). Modul DC-DC konvertoru je založen na čipu XLSemi XL4015, který má rozsah vstupního napětí 4-38 V a max. výstupní proud 5A. Konvertor je na vstupu doplněn pojistkou o jmenovitém proudu 500mA, vzhledem k tomu, že obvod na desce SBC je navržen pro standard 802.3af, kde maximální přenášený výkon je 15,4W. To určuje maximální proudový odběr pro SBC včetně periferií 2,4A.Výstup konvertoru je připojen přímo na piny 4 (5V) a 6 (GND) GPIO headeru Raspberry Pi.

Nevýhoda tohoto řešení je, že SBC a napájecí zdroj nejsou galvanicky odděleny. To může způsobit problém ve specifickém případě – pokud je jako PSE použit uzemněný zdroj, zároveň je uzemněno Raspberry Pi (např. skrz stínění HDMI připojené k uzemněnému monitoru) a zároveň nejsou země zdroje a spotřebiče na stejném potenciálu. Většina zdrojů pro PoE injektory je ale plovoucí, navíc se předpokládá, že celé zařízení bude v provozu izolováno plastovou krabičkou a jediné zvenčí dostupné porty budou USB, u kterých je pravděpodobnost připojení uzemněného spotřebiče minimální.

Wi-Fi

Součástí Raspberry Pi 3B+ je Wi-Fi modul Cypress CYW43455, který podporuje standardy 802.11b/g/n/ac. V OS Linux je rozhraní označeno jako *wlan0* a implicitně se chová jako Client. V rámci brány ale bude sloužit jako AP (přístupový bod) pro připojení koncových zařízení. K tomuto účelu je nutné doinstalovat balíček *AP daemon*. Pro automatickou konfiguraci koncových zařízení je ještě potřeba DHCP/DNS server, zde software DNSMasq. Obojí lze nainstalovat z příkazové řádky takto:

```
apt-get install hostapd dnsmasq
```

AP musí mít statickou IP adresu, ta se přiřazuje stejným způsobem jako u Ethernetového rozhraní – pomocí konfiguračního souboru */etc/network/interfaces*, výše zmíněné rozhraní *wlan0* bude v síti 192.168.5.0/24:

```
auto wlan0
iface wlan0 inet static
        address 192.168.5.200
        netmask 255.255.255.0
```

Protože nyní má brána dvě síťová rozhraní a není žádoucí, aby mezi sebou sítě k nim připojené přímo komunikovaly (komunikace musí probíhat vždy prostřednictvím brány), je potřeba vypnout přeposílání paketů mezi sítěmi (směrování) editací příslušné direktivy v souboru */etc/sysctl.conf*.

Net.ipv4.ip_forward=0

Protokol DHCP (Dynamic Host Configuration Protocol) slouží k přidělování IP adres a dalších síťových konfiguračních údajů nově připojeným zařízením. V konfiguračním souboru DHCP serveru */etc/dnsmasq.conf* je nutné nastavit rozsah, ze kterého budou přidělovány IP adresy jednotlivým koncovým prvkům. První direktiva určuje rozhraní, na kterém server poběží, druhá udává rozsah ve formátu *<od>,<do>,<maska podsítě>,<doba platnosti přidělené adresy>*.

```
interface=wlan0
dhcp-range=192.168.5.1,192.168.5.199,255.255.255.0,24h
```

Další na řadě bylo nastavení Access Point daemona editací */etc/hostapd/hostapd.conf*. Důležité jsou direktivy *interface, ssid a wpa_passphrase*, jejichž název dostatečně vysvětluje funkci. Direktivou *channel* lze nastavit kanál, na kterém AP vysílá (viz Obrázek 5), např. kvůli překryvu se Zigbee nebo jinými Wi-Fi sítěmi.

```
interface=wlan0
driver=nl80211
ssid=NoTGW
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=S3cr3t
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

V soubru */etc/default/hostapd* je ještě nutné nastavit cestu ke konfiguraci:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Po nakonfigurování lze *hostapd* pomocí příkazu *systemctl* (správa služeb poskytovaných systémem – tzv. daemonů, viz níže) povolit (unmask), spustit (start) a zapnout spouštění při startu systému (enable). Poslední direktiva restartuje službu *dnsmasq* kvůli načtení nové konfigurace.

```
systemctl unmask hostapd
systemctl start hostapd
systemctl enable hostapd
systemctl restart dnsmasq
```

Nyní je k dispozici funkční Wi-Fi AP poskytující síť *NoTGW* s přístupovým heslem *S3cr3*.

Pokud by bylo třeba povolit provoz mezi sítěmi *192.168.5.0/24* (Wi-Fi) a *192.168.10.0/24* (Ethernet), např. kvůli sdílení multimédií z LAN sítě nebo staho-

vání aktualizací z internetu přimo do koncových prvků, lze to pomocí výše zmíněné direktivy *Net.ipv4.ip_forward* v souboru */etc/sysctl.conf*. Povolení veškerého provozu je ale bezpečnostní riziko, jádro OS linux ale obsahuje filtr IP packetů, který společně s konfigurační utilitou *iptables* může sloužit jako velmi kvalitní firewall. Pomocí něj by šlo nastavit pravidla pro komunikaci mezi zmíněnými dvěma sítěmi. Tato funkce nebyla implementována, protože v dedikované Smart Home síti by neměla být potřeba a nastavení pomocí textových souborů je relativně komplexní. Pro manipulaci s pravidly ale existují i grafická/webová rozhraní a integrace některého z nich by mohla být zajímavým rozšířením tohoto projektu.

MQTT

Primární způsob komunikace jak mezi komponentami brány, tak s koncovými prvky bude protokol MQTT. Brána tedy musí zajistit funkci MQTT brokeru. To lze mimo jiné realizovat doplňkem do aplikace Home Assistant (viz níže), kvůli požadavku nezávislosti SW komponent bude ale broker samostatná aplikace. Open-source broker Mosquitto napsaný v jazyce C je nenáročný na výkon a poskytuje všechny potřebné funkce. Lze zvolit verzi protokolu 3 nebo 5 a nastavit přístupová práva pro jednotlivá témata.

Veškeré posílané zprávy jsou ve formátu JSON (JavaScript Object Notation), což je standardizovaný formát pro serializaci dat, umožňující odesílat a přijímat složitější datové struktury ve formě textových řetězců (data se převedou z objektu na sérii znaků, odtud serializace). S tímto formátem nativně pracují některé použité komponenty (Node-RED, Zigbee2MQTT a OpenMQTTGateway) a bude použit i ve všech skriptech pro obsluhu senzorů. Příkladem JSON řetězce může být tato zpráva pro rozsvícení žárovky:

{"state": "ON","brightness": 255}

Mosquitto je v repozitářích raspbianu, instalace se provede příkazem:

apt-get install mosquitto mosquitto-clients

Broker se při instalaci sám nastaví jako služba spouštěná při startu systému. Po instalaci je třeba nastavit TCP port, na kterém broker naslouchá. Konfigurační soubor je */etc/mosquitto/mosquitto.conf*. Broker bude dostupný pouze na rozhraní *wlan0* (Wi-Fi) a *loopback* (lokální smyčka na adrese 127.0.0.1 – není dostupné ze sítě), to zajišťují první dvě direktivy *listener*. Druhé dvě direktivy zakazují anonymní uživatele a definují soubor pro ukládání přístupových údajů uživatelů.

```
listener 1883 192.168.5.200
listener 1883 127.0.0.1
allow_anonymous false
password_file /etc/mosquitto/pwfile
```

Po uložení konfigurace lze vytvořit hlavního uživatele *controller* a restartovat broker, aby se nová konfigurace načetla. Příkaz *mosquitto_ passwd* je interaktivní a po zavolání zobrazí výzvu k zadání hesla.

```
mosquitto_passwd -c /etc/mosquitto/pwfile controller
systemctl restart mosquitto
```

Funkčnost brokeru lze ověřit pomocí testovacích nástrojů *mosquitto_ sub* (subscribe do tématu) a *mosquitto_ pub* (publish do tématu), k tomu jsou potřeba dvě otevřené konzole (SSH spojení). Po zadání prvního příkazu budou na první z nich vidět všechny příchozí zprávy v tématu *broker* a všech podtématech, znak "#" znamená právě všechna podtémata rekurzivně (tedy i podtémata podtémat atd.), znak „*" by pak znamenal všechna podtémata, ale nerekurzivně. Po odeslání testovací zprávy druhým příkazem (řetězec za parametrem *-m*, zde „TEST") z druhé konzole by se i tato mělo zobrazit jako příchozí v první konzoli.

```
mosquitto_sub -d -h 127.0.0.1  -u controller -P -t broker/#
mosquitto_pub -d -h 127.0.0.1 -u controller -P -t broker/test -m TEST
```

Tím je funkčnost ověřena a je možno definovat ostatní uživatele a jejich práva (např. do témat pro senzory mohou zapisovat jen skripty obsluhující senzory atd.).

Při testování jsou v první konzoli také vidět zprávy *PINGREQ* (ping request) a *PINGRESP* (ping response). MQTT broker u všech připojených klientů periodicky ověřuje, zda jsou aktivní právě zasláním *PINGREQ*. Pokud klient neodpoví v nastaveném časovém limitu zprávou *PINGRESP,* broker ho označí za odpojený. Pokud odpojený klient předem definoval svou *last will* zprávu („poslední vůle"), bude tato odeslána všem klientům připojeným v režimu subscribe do daného tématu. Na to lze navazovat různé automatizace chování pro případ chyby.

Zigbee

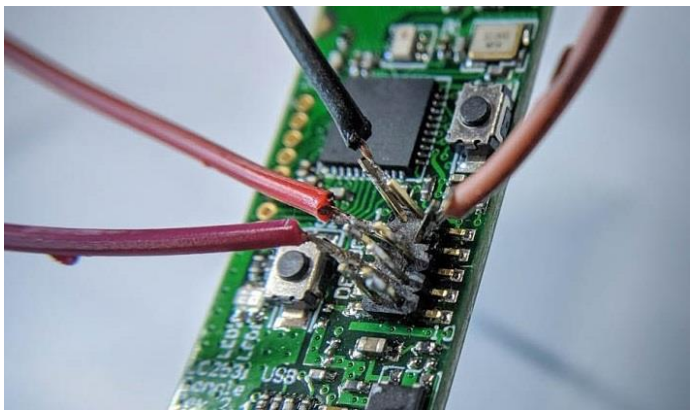
Z dostupných Zigbee periferií je nejvhodnější sniffer CC2531, vzhledem k velmi nízké ceně a jednoduché integraci pomocí open-source projektu Zigbee2MQTT, který řeší, jak název napovídá, právě komunikaci se Zigbee zařízeními prostřednictvím MQTT protokolu.

Firmware

Zařízení je z výroby určeno jako sniffer – přístroj, který dokáže odposlouchávat Zigbee komunikaci v okolí a tím umožňuje ladit např. aplikační protokol pozorovaných zařízení. Aby mohl fungovat jako Coordinator nebo Router, je nutné do něj nahrát alternativní firmware. To lze provést několika způsoby, z nichž jeden umožňuje použít samotné Raspberry Pi jako programátor. Sniffer je vybaven programovacím headerem, který je nutno připojit k Raspberry Pi podle této tabulky:

CC2531 pin	Raspberry Pi pin
1 (GND)	39 (GND)
7 (Reset)	35 (GPIO 24)
3 (DC)	36 (GPIO 27)
4 (DD)	38 (GPIO 28)
2 (3,3V)	1 nebo 17 (3,3V)

Piny headeru mají menší rozteč, než je standard, je třeba buď použít vhodnou redukci, nebo piny ohnout do stran jako je to na obrázku.



Obrázek 8 - sériové rozhraní snifferu pro ladění a upload firmware

Firmware společně s nástrojem pro jeho nahrání do snifferu lze nainstalovat z git repozitáře https://github.com/jmichault/flash_cc2531.git, pro svou funkci vyžaduje knihovnu *WiringPi*. První příkaz nainstaluje nástroj pro správu verzí git a knihovnu *WiringPi*. Pomocí *git clone* je pak vytvořena lokální kopie repozitáře flash_cc2531 s utilitami pro update firmware. Příkaz *./cc_chipid* slouží k ověření správného propojení SBC se snifferem.

```
apt-get install git wiringpi
git clone https://github.com/jmichault/flash_cc2531.git
cd flash_cc2531
./cc_chipid
```

V případě že hardware snifferu je v pořádku a spojení s ním je funkční, vrátí posední příkaz výsledek *ID = b524*. Po úspěšném ověření lze přistoupit k instalaci aktuálního Coordinator firmware do snifferu:

```
wget https://github.com/Koenkk/Z-Stack-firmware/raw/master/coordinator/Z-Stack_Home_1.2/bin/default/CC2531_DEFAULT_20190608.zip
unzip CC2531_DEFAULT_20190608.zip
./cc_erase
./cc_write CC2531ZNP-Prod.hex
```

Po smazání paměti příkazem *cc_ erase* začne blikat zelená LED na desce snifferu, po úspěšném zápise se trvale rozsvítí. Zápis trvá cca 3 minuty. Kromě Coordinatoru je dostupný i firmware pro Router, takže lze tyto moduly v případě potřeby využít i jako opakovače Zigbee signálu. Po nahrání firmware lze od snifferu odpojit programovací vodiče a připojit ho do jednoho z USB portů na Raspberry Pi 3B+.

Zigbee2MQTT

Dalším krokem je instalace aplikační části Zigbee2MQTT, nejprve prerekvizit (Node.js, node package manager a kompilátor jazyka C/C++) a poté samotné aplikace z jejího git repozitáře. U Node.js je třeba kvůli Node-RED (viz níže) nainstalovat nejnovější verzi, která není dostupná v repozitářích Raspbianu. Node.js má pro tento účel instalátor/správce verzí *nvm*. Aplikace bude nainstalována do adresáře */opt/zigbee2mqtt* a poběží pod uživatelem *gwadmin*, kterému je nutné přidat práva k adresáři příkazem *chown*.

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.35.3/install.sh
bash install.sh
nvm install 14.4.0
apt-get install make g++ gcc
git clone https://github.com/Koenkk/zigbee2mqtt.git /opt/zigbee2mqtt
chown -R gwadmin:gwadmin /opt/zigbee2mqtt
cd /opt/zigbee2mqtt
npm ci
```

Uživateli *gwadmin* bylo také nutné udělit práva pro přístup k Zigbee zařízení, které se u Raspberry Pi 3B+ mapuje na */dev/ttyACM0*. Zařízení je vedeno jako sériový port, uživatele tedy stačí přidat do skupiny dialout, která v Linuxu reprezentuje práva na obsluhu sériových portů:

```
usermod -a -G dialout gwadmin
```

Hlavní konfigurační soubor aplikace je */opt/zigbee2mqtt/data/configuration.yaml*, tam je třeba v sekci *mqtt* nastavit MQTT téma, adresu serveru (zde *localhost*) a přístupové údaje. V sekci *serial* pak port, ke kterému je připojen sniffer - */dev/ttyACM0* a rychlost 115200 baudů. Aplikaci lze spustit v interaktivním režimu příkazem *npm start* v instalačním adresáři aplikace */opt/zigbee2mqtt*. Párování nových zařízení je implicitně zapnuto (vypnutí viz níže). Zigbee2MQTT funguje podle očekávání, jak je vidět z výpisu událostí, který obsahuje úspěšné spárování zkušebního Zigbee koncového prvku – žárovky Ikea Tradfri:

```
Apr 13 00:16:09 IoTGW npm[644]: zigbee2mqtt:info  2020-04-13 00:16:09:
    Starting interview of '0xccccccfffe9763af'
Apr 13 00:16:09 IoTGW npm[644]: zigbee2mqtt:info  2020-04-13 00:16:09:
    MQTT publish: topic 'IoTGW/Zigbee/bridge/log', payload
    '{"type":"pairing","message":"interview_started","meta":{"frien
    dly_name":"0xccccccfffe9763af"}}'
```

Apr 13 00:16:27 IoTGW npm[644]: zigbee2mqtt:info 2020-04-13 00:16:27: MQTT publish: topic 'IoTGW/Zigbee/bridge/log', payload '{"type":"device_announced","message":"announce","meta":{"friendly_name":"0xcccccccfffe9763af"}}'

Apr 13 00:16:30 IoTGW npm[644]: zigbee2mqtt:info 2020-04-13 00:16:30: Successfully interviewed '0xcccccccfffe9763af', device has successfully been paired

Apr 13 00:16:30 IoTGW npm[644]: zigbee2mqtt:info 2020-04-13 00:16:30: Device '0xcccccccfffe9763af' is supported, identified as: IKEA TRADFRI LED bulb E14/E26/E27 600 lumen, dimmable, color, opal white (LED1624G9)

Apr 13 00:16:30 IoTGW npm[644]: zigbee2mqtt:info 2020-04-13 00:16:30: MQTT publish: topic 'IoTGW/Zigbee/bridge/log', payload '{"type":"pairing","message":"interview_successful","meta":{"friendly_name":"0xcccccccfffe9763af","model":"LED1624G9","vendor":"IKEA","description":"TRADFRI LED bulb E14/E26/E27 600 lumen, dimmable, color, opal white","supported":true}}'

Daemonizace

Aplikace nemůže stále běžet v testovacím režimu, po vypnutí konzole (SSH spojení) dojde i k ukončení aplikace. Je třeba, aby se aplikace spouštěla jako služba na pozadí při startu systému, běžela nepřetržitě a v případě výpadku se sama restartovala. Procesy, které se takto chovají, se v Linuxu označují jako *daemon* a jsou spravovány pomocí programu s*ystemd*, což je první proces (také zvaný init), který se spustí po startu OS a má na starosti správu všech ostatních procesů. Konfigurace daemonů se provádí pomocí souborů zvaných *unit files* (běžící daemon je označován jako *systemd unit*) a spravovaná aplikace běžící na pozadí se nazývá *služba* (service). U software instalovaného pomocí balíčků z repozitářů OS jsou unit files vytvářeny automaticky, zde je ale software nainstalován z externího zdroje a *systemd unit* je nutné vytvořit ručně. Systemd unit je (jako vše ostatní) reprezentován souborem, zde byl vytvořen */etc/systemd/system/zigbee2mqtt.service* s obsahem:

```
[Unit]
Description=zigbee2mqtt
After=network.target

[Service]
ExecStart=/usr/bin/npm start
WorkingDirectory=/opt/zigbee2mqtt
StandardOutput=inherit
StandardError=inherit
Restart=always
User=gwadmin

[Install]
WantedBy=multi-user.target
```

Syntaxe připomíná konfigurační soubory .ini na systémech Microsoft Windows. Důležité direktivy jsou *Description* (název/popis dané služby), ExecStart (cesta k programu, který se spustí jako daemon), WorkingDirectory (pracovní adresář

dané aplikace), Restart (automatický restart při pádu služby) a WantedBy (ta udává tzv. runlevel kterého je služba součástí, multi-user.target je standardní režim, ve kterém se nachází Linux systém bez GUI prostředí, pokud se všechny jeho součásti správně inicializovaly při startu. Dalším příkladem je např. nouzový režim – emergency.target).

Po uložení souboru je ještě nutné novu systemd unit načíst (1), zapnout automatický start služby po startu OS (2) a nakonec službu spustit (3). Tím je Zigbee komponenta připravena k použití:

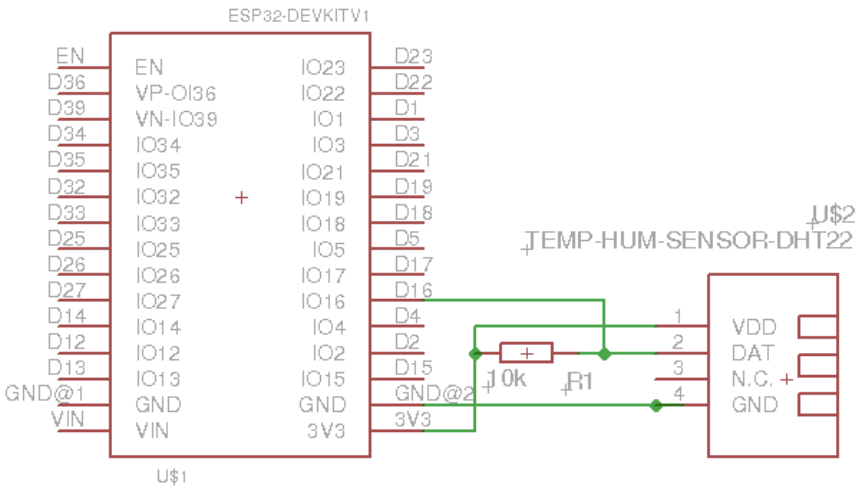
1) *systemctl daemon-reload*
2) *systemctl enable zigbee2mqtt.service*
3) *systemctl start zigbee2mqtt.service*

Bluetooth

Původní záměr byl využít vestavěnou Bluetooth kompatibilitu Raspberry Pi 3B+ a podobně jako u Zigbee některý z open source BLE-MQTT mostů (projektů s dostatečnou funkcionalitou je hned několik, např. ble2mqtt nebo Espruino). Ukázalo se ale, že Raspberry Pi SBC mají problém se souběžným provozem Wi-Fi AP a Bluetooth. Pokud běží oba zároveň, Wi-Fi je nestabilní a BLE má problém s párováním zařízení. Při provozu Wi-Fi rozhraní v režimu Client funguje vše v pořádku. Tato chyba není moc dobře popsaná, s největší pravděpodobností je problém přímo v hardware SoC (System on Chip) Broadcom, na kterém je Raspberry Pi postaveno, dokumentace k němu ale není veřejně dostupná. [12]

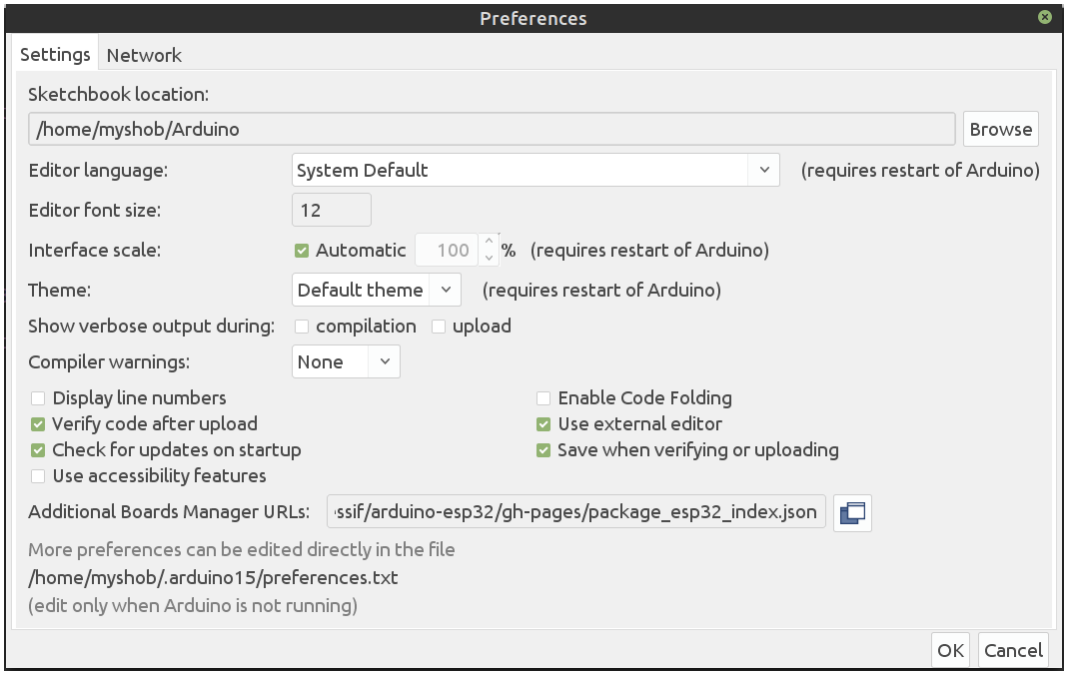
Použito tedy bylo alternativní řešení, kde Bluetooth konektivitu zajišťuje samostatný mikrokontroler ESP32 se softwarem OpenMQTTGateway. Ten s Raspberry Pi 3B+ komunikuje přes Wi-Fi (MQTT) a tvoří tak samostatný fyzický modul. Díky bezdrátové komunikaci přes Wi-Fi stačí tomuto modulu už jen napájení, s hlavní částí brány bude tedy spojen jen napájecím USB konektorem. V případě potřeby lze modul odpojit a provozovat např. ve vedlejší místnosti, kde může být blíže k BLE koncovým prvkům.

Pro ještě větší využití této modularity byl ESP32 doplněn o teplotní a vlhkostní senzor DHT22. Při umístění modulu v jiné místnosti, než je hlavní část brány budou k dispozici naměřené hodnoty z obou místností a díky tomu nebude potřeba další koncový prvek pro měření teploty a vlhkosti. Pokud je naopak modul a hlavní část na stejném místě, lze hodnoty ze senzorů průměrovat z důvodu vyšší přesnosti a kontrolovat odchylku z důvodu detekce vadného senzoru. Schéma propojení senzoru DHT22 a MCU ESP32 je na obrázku.



Obrázek 9 – schéma připojení senzoru DHT22 k ESP32

K naprogramování ESP32 je potřeba vývojové prostředí Arduino IDE, nejnovější verze je dostupná z *https://www.arduino.cc/en/Main/Software*. ESP32 nepatří pod rodinu mikrokontrolerů Arduino a do IDE je nutné doplnit definice desky. K tomu stačí otevřít Arduino IDE, v hlavní liště otevřít v menu *File* položku *Preferences*, a do kolonky *Additional Board Manager URLs* zadat cestu k definici: *https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json*.



Obrázek 10 - přidání definic pro nový mikrokontroler v Arduino IDE

Samotná OpenMQTTGateway aplikace je k dispozici na webu projektu *https://github.com/1technophile/OpenMQTTGateway/releases*, potřeba jsou

kompletní zdrojové kódy a knihovny na kterých závisí. Knihovny je potřeba nakopírovat do podadresáře *libraries* v instalačním adresáři Arduino IDE.

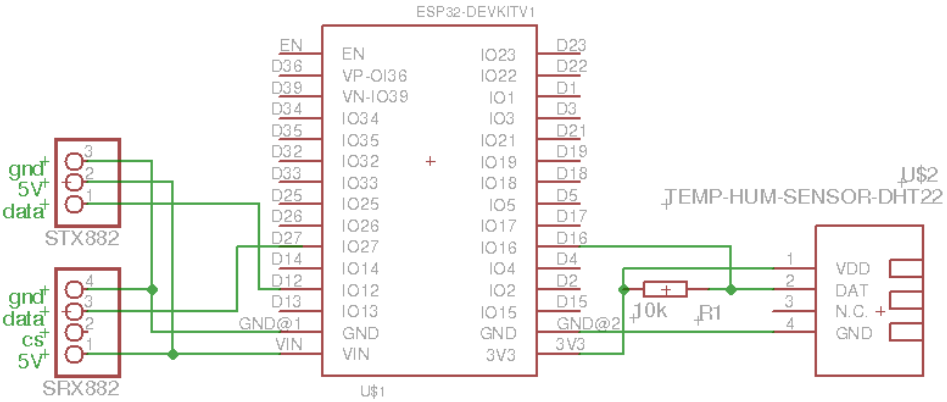
Před nahráním zdrojových kódů do MCU je vhodné tyto kódy ověřit pomocí funkce „Verify“, která je součástí Arduino IDE. Aby ověření fungovalo, je nutné vybrat správný typ desky (ESP Dev Module) v nabídce *tools -> board*. Po ověření je nutné provést konfiguraci pomocí hlavičkového souboru *User_config.h*. Konfigurační parametry jsou zadávány jako vnitřní konstrukce jazyka C++, jednotlivé direktivy ale mají intuitivní názvy, navíc je jejich funkce pečlivě okomentována. Aplikace je modulární, nepotřebné komponenty lze vypnout „zakomentováním“ příslušné *#include* direktivy (připsáním dvou lomítek “//” na začátek řádku, což je v C++ syntaxe pro komentář). Jediné dva moduly, které tento modul potřebuje jsou *ZgatewayBT* (Bluetooth) a *ZsensorDHT* (DHT senzor). Dále bylo třeba nastavit název zařízení, IP adresu, SSID a heslo pro Wi-Fi AP nastavené v sekci Wi-Fi. Důležité je také nastavení MQTT brokeru – IP adresa, uživatelské jméno, heslo a základní téma pod kterým si aplikace vytvoří své podtéma pro zápis zpráv. Jméno je stejné jako definovaný název zařízení, ESP tedy zprávy od přípojených Bluetooth zařízení bude zapisovat do tématu *NoTGW/OMG_BLE*.

Po konfiguraci je vhodné ještě jednou kódy verifikovat a poté nahrát program do mikrokontroleru. OpenMQTTGateway automaticky detekuje jakékoli podporované (viz web projektu https://docs.openmqttgateway.com) Bluetooth zařízení v dosahu, provede spárování a získaná data posílá do MQTT, koncové prvky tedy není vůbec potřeba přidávat manuálně. To může být i nevýhoda, protože nad připojováním zařízení není k dispozici žádná manuální kontrola, v případě problémů lze ale v User_config.h zapnout logovací modul, který vypisuje chyby a informace o provozu na sériový port (terminál je potřeba nastavit na rychlost 115200 baudů).

RF 433 MHz

Řešení Bluetooth komunikace pomocí samostatného modulu s ESP32 a OpenMQTTGateway (dále jen OMG) se po několika testech osvědčilo jako spolehlivé s velmi jednoduchým přidáváním nových koncových prvků. OMG projekt původně vznikl právě pro komunikaci přes RF 433 a podporuje obrovské množství koncových prvků, bude tedy použit i pro tuto komponentu.

Opět bude realizován samostatným modulem s MCU ESP32 a kromě vysílače a přijímače pro pásmo 433 Mhz k němu bude připojen rovněž senzor DHT22.

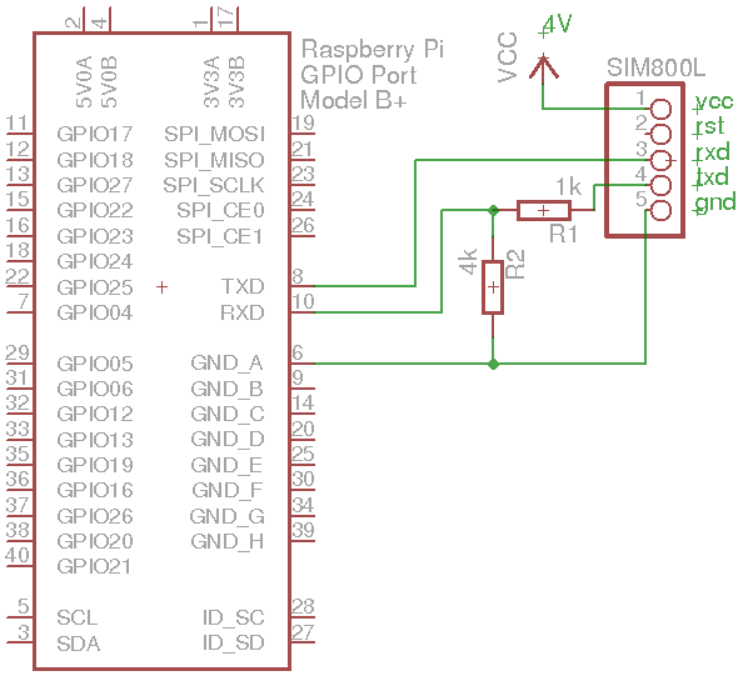


Obrázek 11 - schéma propojení ESP32 se senzorem DHT22, 433 MHz vysílačem STX882 a přijímačem SRX882

V konfiguračním souboru OMG *User_config.h* je potřeba povolit moduly *ZgatewayRF*, *ZgatewayRF2* a *ZsensorDHT*. Ostatní konfigurační parametry jsou stejné jako u Bluetooth verze, pouze použité MQTT téma (parametr *Base_Topic*) se změní na *NoTGW/OMG_RF*. Po naprogramování mikrokontroleru modul stejně jako u BLE verze detekuje jakékoli RF 433 zařízení v dosahu a data posílá do MQTT.

GSM

GSM protokol bude použit pouze pro posílání a přijímání SMS zpráv. Jako modem byl zvolen čip SIMCom SIM800L, který kromě SMS funkcionality využité v bráně umožňuje volání a GPRS datové přenosy. Raspberry Pi GPIO rozhraní disponuje jedním UART portem, ten bude použit pro připojení modemu místo USB. Modem je určen pro mobilní telefony a jeho provozní napětí je 3,4-4,4 V (kvůli napájení z Li-Po baterií, které mají jmenovité napětí 3,7 V), pro napájení tedy potřebuje svůj regulátor napětí a pro sériovou komunikaci dělič napětí na vysílacím pinu (GPIO rozhraní Raspberry Pi pracuje s napětím 3,3 V). Při testech se objevil problém s náhodnými restarty modemu při odesílání SMS, což bylo způsobeno špičkami v odběru proudu – běžný odběr je kolem 200 mA, při vysílání ale mohou nastat špičky až 2 A, to bylo vyřešeno přidáním vyrovnávacího kondenzátoru.



Obrázek 12 - schéma připojení modemu SIM800L k Raspberry Pi 3B+, jako zdroj napájecího napětí 4V bude použit lineární či spínaný regulátor

UART rozhraní je nejprve třeba aktivovat pomocí utility *raspi-config*, v nabídce *interfacing options* je položka *serial*, po otevření je nutné zvolit *disable* pro login shell a *enable* hardware sériového portu. Po aktivaci je UART v systému dostupný jako zařízení */dev/ttyS0*. Spojení s modemem lze otestovat příkazem *minicom*, po jeho spuštění je třeba otevřít nastavení sériového portu pomocí klávesové zkratky *ctrl + o* a v nabídce zvolit port */dev/ttyS0* s rychlostí komunikace 9600 baudů. V hlavním okně by pak po odeslání příkazu „AT“ měl modem odpovědět „OK“, lze otestovat různé AT příkazy, seznam všech podporovaných lze nalézt v datasheetu modemu.

Existuje několik různých open-source projektů pro komunikaci pomocí GSM modemu pod OS Linux (např. Kannel, Gammu aj.), jedná se ale o komplexní aplikace a většinu jejich funkcionality by brána nevyužila, obsluhu modemu bude tedy zajišťovat Python script vytvořený přímo pro tento účel, stejně jako je tomu u vestavěných senzorů. Veškeré přijaté SMS zprávy budou uloženy v souboru */var/local/GSM/SMS.messages* jako JSON řetězec ve formátu:

{"id":"<ID>","sender":"<odesilatel>","received":"<prijato>","content":"<obsah SMS>"}

Se zprávami půjde manipulovat pomocí MQTT tématu *NoTGW/GSM/SMS*, příkazy jsou označeny jako „action“ a odpovědi jako „info“. Výpis SMS ze souboru pomocí *{"action":"smslist"}* vrátí:

{"info":"smslist","list":"<seznam JSON řetězců výše uvedeného formátu>"}

Přijaté SMS zprávy jsou uloženy do souboru a zároveň se objeví v MQTT jako:

{"info":"smsrecv","id":"<ID>","sender":"<odesilatel>", "content":"<obsah SMS>"}

Odeslat sms lze pomocí:

{"action":"smssend","dest":"<cislo prijemce>","content":"<obsah SMS>"}

a při úspěšném odeslání je vráceno potvrzení

{"info":"SMS_send_succ","destination":"<cislo prijemce>","content":"<obsah SMS>"}

Při zápisu do souboru jsou přijaté SMS automaticky mazány z paměti SIM karty. Pro smazání ze souboru lze použít příkaz

{"action":"smsdel","id":"<ID>"}

který při úspěšném smazání vrátí potvrzení *{"info":"SMS_del_succ","id":<id>"}*. Jakákoli chyba, která nastane při práci s SMS je ohlášena zprávou v podobě:

{"info":"error","detail":"<podrobný popis chyby>"}

Vytvořený skript *gsmbridge.py* je umístěn v adresáři */usr/local/bin/GSM* a řeší pouze popsanou logiku práce s SMS zprávami. Pro samotné ovládání modemu je využita knihovna autora Jakhax dostupná z webu https://github.com/jakhax/raspberry-pi-sim800l-gsm-module, některé části kódu bylo ale třeba pozměnit, aby fungovaly, upravenou knihovnu lze najít v příloze této práce. Kromě ní byla použita také knihovna SimpleJSON, kterou lze doinstalovat z repozitářů systému:

```
apt-get install python3-simplejson
```

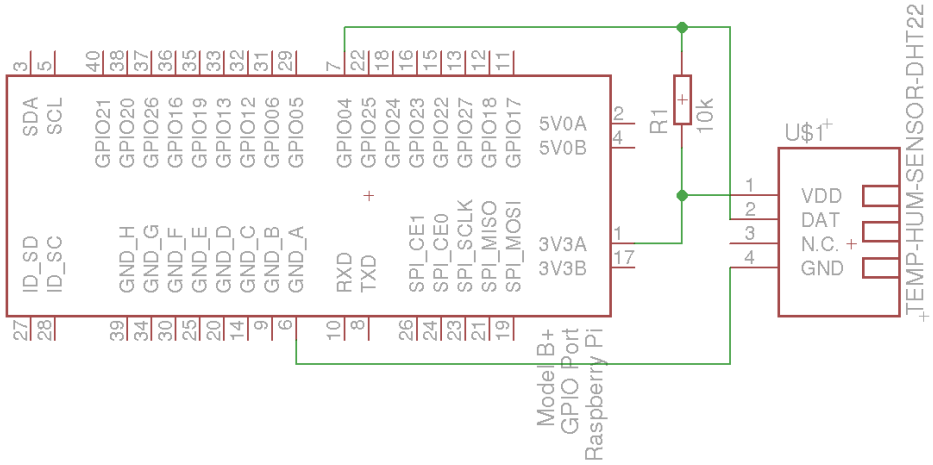
Skript pro práci s SMS je ze všech vytvořených nejsložitější, šel by ale použít jako generické rozhraní SMS – MQTT i v jiných aplikacích, než je tato brána. Stejně jako ostatní je řádně okomentován a společně s příslušným systemd unit file k nalezení v příloze práce.

Senzory

Jelikož Raspberry Pi neobsahuje žádný ADC, který by šlo využít pro připojení analogových senzorů, je nejvhodnější použít senzory digitální, které fungují samostatně a rovnou poskytují měřenou veličinu jako číslicovou hodnotu pomocí nějaké nízkoúrovňové sběrnice (I^2C, SPI aj.).

DHT22

Ze dvou modulů pro měření teploty i vlhkosti popsaných výše v sekci senzory je DHT22 levnější a rozdíl v přesnosti oproti SHT71 není tak velký, aby na něm záleželo při měření v obytném prostoru pro účely Smart Home. K Raspberry Pi 3B+ je senzor připojen podobně jako k ESP32, viz obrázek.



Obrázek 13 - schéma připojení senzoru DHT22 k Raspberry Pi 3B+

Z hlediska software stačí jednoduchý program, který bude periodicky číst data ze senzoru a výslednou hodnotu odesílat jako MQTT zprávu. To bude, stejně jako obsluha ostatních senzorů, realizováno skriptem v jazyce Python, pro který je dostupné velké množství knihoven určených přímo pro interakci s GPIO rozhraním u Raspberry Pi. Python je potřeba nainstalovat z repozitářů OS Raspbian včetně balíčkovacího manageru *pip* a knihovny pro MQTT komunikaci. Následně lze pomocí *pip* doplnit knihovnu *Adafruit_DHT* pro obsluhu DHT11/22 a knihovnu *configparser* pro jednodušší práci s konfiguračními soubory. Formát konfiguračních souborů, se kterými *configparser* pracuje odpovídá výše zmíněnému souboru */etc/notgw.conf*.

```
apt-get install python3 python3-pip python3-paho-mqtt
pip3 install Adafruit_DHT
pip3 install configparser
```

Hotový skript lze najít v příloze této práce, kód je jednoduchý a je řádně okomentován. Skript nejprve pomocí knihovny *configparser* načte nastavený interval dotazování senzoru z */etc/notgw.conf*, připojí se k MQTT brokeru jako *publish client* a dále periodicky čte pomocí knihovny *Adafruit_DHT* aktuální hodnoty ze senzoru, které v nastaveném intervalu odesílá do tématu *NoTGW/Sensors/DHT* jako JSON zpráva tvaru:

```
{"sensor":"DHT","temperature":“<teplota>“,"temp_unit":"C",
"humidity":“<vlhkost>“}
```

Odesílaná zpráva má zapnutý příznak *retain*, který zajišťuje, že broker si uloží poslední přijatou zprávu v daném tématu a jakýkoli nový odběratel, který se k tématu připojí ji okamžitě uvidí a nemusí tak čekat na aktualizaci hodnoty senzoru.

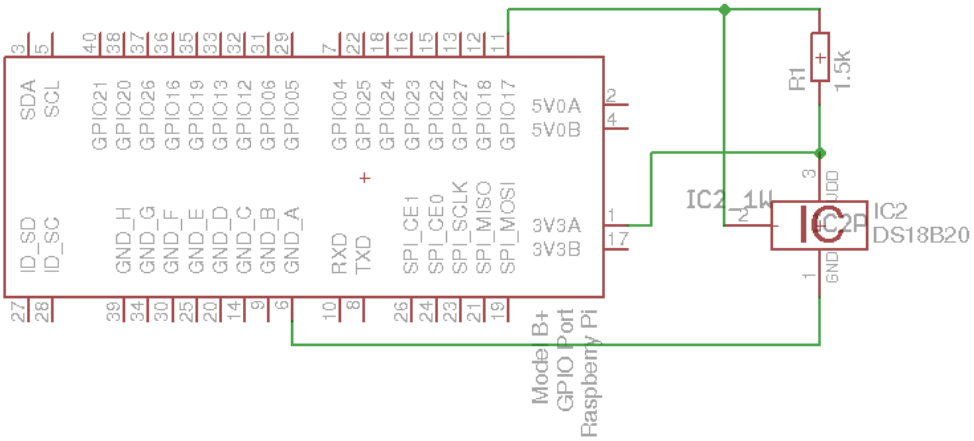
Skript je, stejně jako u ostatních senzorů, uložen v adresáři */usr/local/bin/*sensors a je ho třeba spouštět jako daemon na pozadí, postup nastavení je stejný jako u Zigbee2MQTT a systemd unit file lze rovněž najít v příloze

práce. Při vytváření systemd unit file je třeba dát pozor, aby byly skripty spouštěny správným interpreterem (z historických důvodů existují dvě verze – Python 2 a Python 3, které nejsou navzájem kompatibilní), direktiva *ExecStart* v sekci *Service* tedy musí vypadat takto:

ExecStart=/usr/bin/python3 /usr/kicak/bin/sensors/dht11.py

DS18B20

Zatímco DHT22 slouží pro měření uvnitř místnosti, tento senzor je myšlený pro měření venkovní teploty, případně nějaké pomocné teploty, pokud už je venkovní teplota měřena jinak. Senzor je v případě potřeby možné i úplně odpojit, připojen je k hlavní části brány přes konektor RJ-11. DS18B20 se dodává buď jako součástka v pouzdře TO92, nebo jako sonda v hermeticky uzavřeném kovovém pouzdře včetně přívodního kabelu. Senzor komunikuje po sběrnici 1-Wire, která kromě toho, že jí v tzv. parazitním módu stačí jeden vodič pro napájení a data zároveň (odtud název sběrnice) umožňuje komunikaci na relativně dlouhou vzdálenost. Konkrétně tento senzor by měl spolehlivě pracovat do délky kabelu 3 m. Pro připojení není použit parazitní mód (který snižuje maximální délku kabelu), ale separátní datový vodič.



Obrázek 14 - schéma připojení 1-Wire teploměru DS18B20 k Raspberry Pi 3B+

OS Raspbian obsahuje ovladače pro 1-Wire přímo v jádře, pro obsluhu senzoru tedy nejsou potřeba dodatečné knihovny. Podporované jsou až tři souběžně provozované sběrnice (přes GPIO piny 4, 17 a 27) a nastavení se provádí v souboru */boot/config.txt*. Sběrnice byla aktivována na vybraném pinu 17 přidáním záznamu:

dtoverlay=w1-gpio,gpiopin=17

Po restartu SBC je 1-Wire aktivní, v adresáři */sys/bus/w1/devices* přibyl podadresář s ID sběrnice a v něm lze najít všechna připojená zařízení. Následné přečtení souboru korespondujícího se senzorem vrátilo dva řádky, první z nich obsahuje kontrolní součet (CRC) a druhý naměřenou hodnotu:

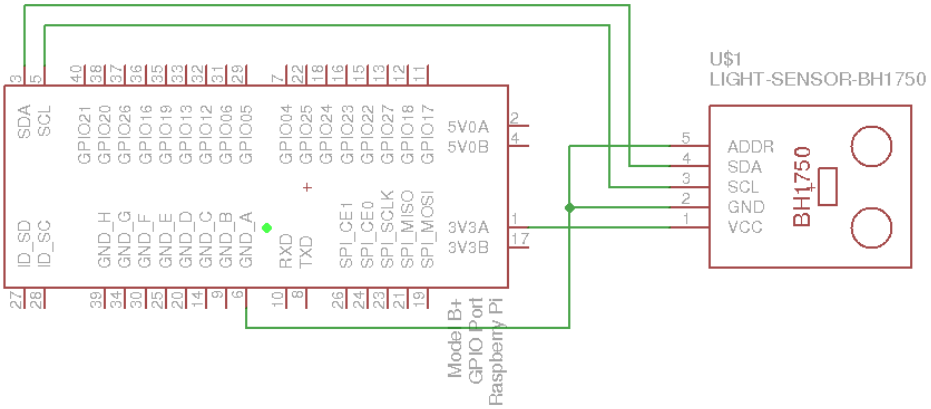
cat /sys/bus/w1/devices/28-01191c123c79/w1_slave

Po úspěšném testu je senzor připraven k použití, zbývá vytvořit obslužný skript a systemd unit file. Postup je stejný jako u DHT22, hodnota ze senzoru se získává pomocí systémových volání (knihovna os), není tedy třeba nic doinstalovávat. Hotový obslužný skript včetně systemd unit file je v příloze práce. Příslušné MQTT téma je zde *NoTGW/Sensors/Temp* a odesílaná zpráva vypadá takto:

{"sensor":"DS18B20","temperature":"<teplota>","unit":"C"}

BH1750

Digitální modul pro měření intenzity osvětlení, jehož nejčastějším případem aplikace je autoregulace podsvícení LCD displayů. S Raspberry Pi 3B+ komunikuje pomocí čtyřvodičové sběrnice I²C (inter-integrated circuit), pátý pin na modulu slouží k výběru jedné ze dvou možných I²C adres pomocí spojení tohoto pinu buď s GND nebo s VCC. Na jedné sběrnici lze tedy současně provozovat dva senzory. Adresní pin je spojen se zemí, čímž je zvolena adresa 0x23, celé schéma zapojení je na obrázku.



Obrázek 15 - schéma připojení senzoru intentizy osvětlení BH1750 k Raspberry Pi 3B+

I²C sběrnici je nutné zapnout pomocí interaktivního příkazu *raspi-config,* v nabídce *interfacing options* musí být položka *I²C* nastavena na *enabled*. Změna se projeví až po restartu systému. Pro obslužný Python skript je třeba doinstalovat knihovnu *pytnon-smbus*, pro test sběrnice pak nástroje *i2c-tools*:

apt-get install python-smbus i2c-tools

Pomocí jednoho z testovacích nástrojů je možné ověřit adresu připojeného zařízení:

i2cdetect 1

Vzhledem k tomu, že senzor je jediné připojené I²C zařízení, měla by být výstupem příkazu pouze jeho adresa odpovídající té nastavené, tedy 0x23. Obslužný Python skript a systemd unit file jsou v příloze této práce. Senzoru přísluší MQTT téma *NoTGW/Sensors/Light*, odesílaná zpráva vypadá takto:

{"sensor":"BH1750","light":"<úroveň osvětlení>","unit":"lx"}

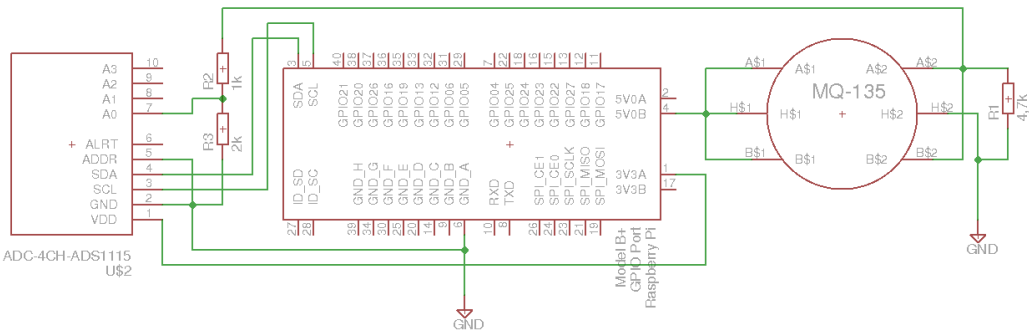
MQ135

Jediný použitý senzor, který není ve formě digitálního modulu, MQ135 byl zvolen z důvodu nízké ceny – cca 50 Kč. Optické, ale i jiné elektrochemické senzory určené k měření koncentrace CO_2 svou cenou často převyšují Raspberry Pi 3b+. MQ135 přitom má, za předpokladu správné kalibrace, dostatečnou přesnost pro potřeby Smart Home.

Senzor obsahuje topný rezistor, který ohřívá vrstvičku SnO_2 na optimální teplotu, při které je nejcitlivější na měřené plyny (nejvíce mění svůj odpor v závislosti na koncentraci plynu). Výsledky, které senzor poskytuje jsou tedy také závislé na teplotě prostředí, chyba však není tak velká, aby bylo nutné výsledky teplotně kompenzovat (zvlášť v oblasti 20–40 °C je senzor velmi stabilní). Před prvním použitím je senzor nutné nechat 48 hodin „zahořet“ (nechat ho nepřetržitě zapnutý), ideálně v místnosti, kde se koncentrace měřeného plynu během této doby nebude příliš měnit, poté je možné provést kalibraci.

Připojení

Na rozdíl od ostatních použitých senzorů pracuje MQ135 s napětím 5V. Vzhledem k tomu, že Pi Zero nemá analogové vstupní piny jako např. Arduino, je nutné použít externí ADC, konkrétně 16-bitový ADS1115, který komunikuje po sběrnici I²C. Tento převodník může pracovat na 3,3V i 5V, podle napájecího napětí se ale změní i logické úrovně komunikační sběrnice I²C. ADC tedy bude pracovat s napětím 3,3V a výstup senzoru bude z 5V na 3,3V převeden pomocí napěťového děliče. Připojení MQ135 k ADC je velmi jednoduché – výstup z napěťového děliče stačí zavést do jednoho ze čtyř vstupů ADC. Podobně jako u BH1750 byla I²C adresa nastavena pomocí ADDR pinu, adresa musí být jiná než u BH1750, protože jsou na stejné sběrnici, schéma adresace je ale rozdílné, připojením pinu k zemi (GND) se adresa nastaví na 0x48. V prototypu brány byl senzor MQ135 součástí modulu FC-22, konečná verze ale vuyžívá pouze samotný senzor, schéma zapojení je na obrázku.



Obrázek 16 - schéma zapojení měřicího obvodu pro senzor MQ-135 a jeho připojení k Raspberry Pi 3B+

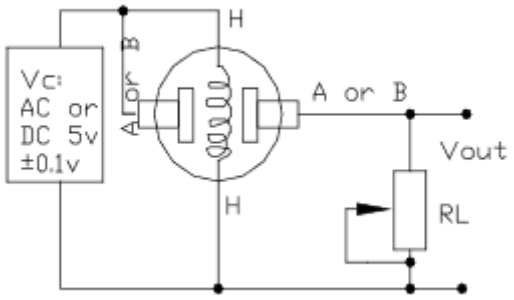
Po ověření připojení pomocí *i2cdetect 1* (stejně jako u BH1750) lze opět přistoupit k vytvoření obslužného skriptu. Pro práci s ADS1115 je dostupná python knihovna od firmy Adafruit, která vyrábí moduly právě s těmito ADC. Tu lze doinstalovat pomocí balíčkovacího manageru:

pip3 install adafruit-circuitpython-ads1x15

S použitím této knihovny byl nejprve vytvořen velmi jednoduchý skript pro výpis surových dat z převodníku do konzole. Následně bylo třeba senzor zkalibrovat a poté vytvořit finální obslužný skript jako u ostatních senzorů.

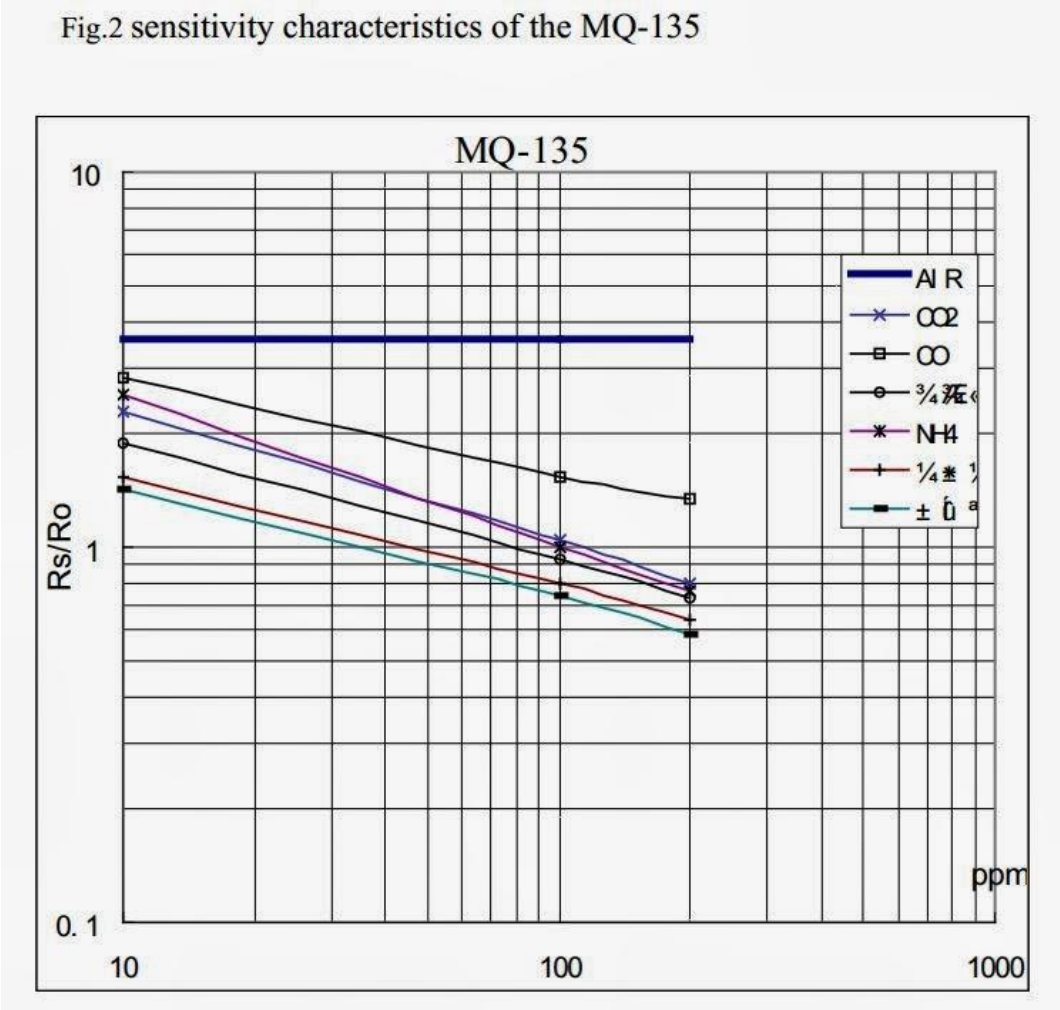
Kalibrace

V katalogovém listu MQ135 je základní schéma zapojení (viz Obrázek 17) a graf ukazující závislost naměřené hodnoty na poměru odporů R_S/R_0 (viz Obrázek 18).



Obrázek 17 - Schéma zapojení měřicího obvodu senzoru MQ135

R_S vyjadřuje odpor snímacího prvku při aktuálně měřené hodnotě ppm, R_0 pak odpor snímacího prvku při nějaké známé hodnotě ppm, R_0 je v rámci jednoho senzoru konstantní a používá se jako referenční hodnota, pro více senzorů se ale hodnota liší kus od kusu.



Obrázek 18- Závislost poměru Rs/R_0 na měřené hodnotě ppm pro různé plyny

Z grafu na obrázku 3 je mimo jiné vidět křivka závislosti poměru odporů na ppm pro CO_2, jedná se o mocninnou funkci (graf je v log-log souřadnicích) s obecným vzorcem:

$$y = a \cdot b^x$$

hodnotu ppm lze tedy zjistit takto:

$$ppm = a \cdot \left(\frac{R_S}{R_0}\right)^b$$

Jelikož hodnoty a b nejsou v katalogovém listu uvedeny, bylo nutné je získat metodou regresní analýzy. Výsledkem bylo:

$$a = 116.6020682$$
$$b = -2.769034857$$

Ze schématu na obrázku 2 je vidět, že snímač tvoří s přídavným rezistorem R_L dělič napětí. Na původně použitém modulu FC-22-I, byl tento rezistor již osazen a jeho hodnota byla 4,7 kΩ, tato hodnota byla zachována i v konečné konstrukci při použití samotného senzoru. ADS1115 poskytuje hodnotu napětí na svém vstupu U_{ADC}, výstup senzoru je k ADC ale připojen přes dělič napětí (s poměrem

2/3, kvůli výše zmíněnému rozdílu logických úrovní), takže hodnota výstupního napětí senzoru bude:

$$U_{OUT} = \frac{U_{ADC}}{2/3}$$

Teď už lze vypočítat okamžitou hodnotu R_S:

$$R_S = \frac{R_L \cdot (5 - U_{OUT})}{U_{OUT}}$$

A pomocí ní pak i referenční odpor R_0:

$$R_0 = R_S \cdot \left(\frac{ppm_{ref}}{a}\right)^{\frac{1}{b}}$$

Referenční hodnota koncentrace CO_2 ppm_{ref} naměřena optickým senzorem Plantower DS-CO2-20 byla 411 ppm (měřeno ve venkovním prostředí v Praze 10). Referenční odpor instalovaného senzoru (průměr z 50 naměřených hodnot) je

$$R_0 = 1836{,}8304365855317$$

Po získání této reference je možné výpočet ppm implementovat v obslužném skriptu a vytvořit příslušný systemd unit file (obojí opět k nalezení v příloze). MQTT téma pro senzor je *NoTGW/Sensors/Gas* a odesílaná zpráva vypadá takto:

{"sensor":"MQ-135","CO_2":"<ppm CO2 ve yzduchu>","unit":"ppm"}

Senzor byl testován porovnáváním naměřených hodnot s referenčním senzorem Plantower DS-CO2-20, který má udávanou chybovost ±50 ppm. Po zahřátí na provozní teplotu, které trvá cca 5 minut jsou naměřené hodnoty poměrně přesné, největší odchylka mezi senzory byla 73 ppm. V děliči napětí byly použity precizní rezistory s 0.1% tolerancí (i když se jejich vliv následnou kalibrací minimalizuje) a použité ADC má relativně vysoké rozlišení 16 bitů. Podle zkušeností ostatních uživatelů (zdrojem informací jsou různá témata na fóru arduino.cc) je na tom MQ135 s přesností hůře (s odchylkami až kolem 180ppm), je tedy možné, že dost záleží na konkrétním kusu.

Software ústředny

Dosud popsané komponenty zajišťují hlavně funkce brány pro komunikaci s koncovými prvky (včetně vestavěných senzorů), je třeba ale také zajistit funkce ústředny jako je uživatelské ovládací rozhraní, automatizace atd.

Home Assistant

Na základě zkušenosti s předchozími projekty Open-source Smart home [18] byla jako hlavní uživatelské rozhraní ústředny zvolena aplikace Home Assistant (dále jen HA), hlavními důvody jsou mimo jiné rychlý vývoj, největší počet dostupných integrací pro koncové prvky a propracovaná integrace s Node-RED.

HA nabízí jako primární způsob instalace vlastní obraz operačního systému Linux, který lze nainstalovat stejným způsobem jako výše zmíněný Raspbian. Tento OS je ale plně spravován přes komponentu HA Supervisor, což není vhodné kvůli požadavku na nezávislost komponent, navíc Supervisor má možnost upravovat i nastavení systému, která by narušila funkci ostatních komponent brány (síť, systemd aj.). Pro Debian Linux a jeho derivace (včetně použitého Raspbianu) je k dispozici alternativní instalace pomocí skriptu, který obsahuje jinou verzi HA Supervisoru schopnou spravovat pouze moduly a Docker kontejnery samotného HA. Tato metoda je aktuálně označena jako „Home Assistant Supervised" a je pro bránu nejvhodnější.

Nejprve je potřeba nainstalovat prerekvizity z repozitářů systému – balíčky, které HA potřebuje ke své funkci (1), následně nainstalovat Docker (2) z oficiálního zdroje get.docker.com (v repozitářích není dostupný) a vypnout službu ModemManager, která způsobuje problémy s detekcí nových koncových prvků v HA (3). Poté lze stáhnout a spustit zmíněný instalační skript (4).

(1) apt-get install software-properties-common apparmor-utils apt-transport-https ca-certificates curl dbus jq aahi-daemon
(2) systemctl disable modemmanager && systemctl stop modemmanager
(3) curl -fsSL get.docker.com | sh
(4) curl -sL "https://raw.githubusercontent.com/Kanga-Who/home-assistant/master/supervised-installer.sh" | bash -s

Po instalaci je webové rozhraní HA dostupné na 192.168.10.210:8123, při prvním přístupu proběhne inicializace, během které je zobrazena zpráva „preparing Home Assistant, this may take up to 20 minutes". Poté se spustí průvodce úvodní konfigurací, kterým lze nastavit hlavní uživatelský účet a základní údaje o instanci. Po dokončení je HA připraven k použití a je možné začít přidávat koncové prvky, seznam dostupných integrací je na webu projektu https://www.home-assistant.io/integrations.

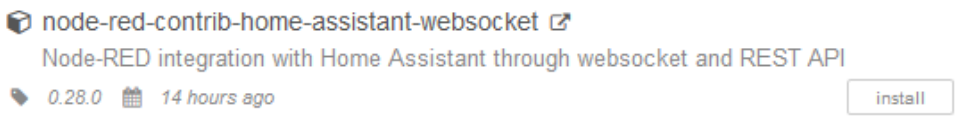
Node-RED

Node-RED (dále jen NR) bude doplňovat Home Assistant o automatizační engine a dodatečné integrace. Pro aplikaci je potřebný interpreter Node.js minimálně verze 14, který již byl nainstalován v rámci Zigbee2MQTT. NR lze nainstalovat pomocí balíčkovacího manageru *npm,* lepší je ale použít instalační skript určený pro Raspberry Pi, který kromě samotné aplikace přidá i systemd unit a několik nodes specifických pro Raspberry Pi (1). Poté je třeba spustit příslušného daemona (2).

(1) curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered | bash
(2) systemctl enable nodered && systemctl start nodered

Integrace Node-RED s HA je realizována kolekcí *node-red-contrib-home-assistant-websocket.* Kolekce se instalují přes webové rozhraní, které běží na

192.168.10.210:1880. V pravém horním rohu je hlavní hamburger menu a v něm volba „Manage palette“. Po jejím otevření se zobrazí seznam nainstalovaných kolekcí a nodes, nové lze vyhledat a instalovat po přepnutí na záložku „Install“, správná kolekce má nejkratší dobu od poslední úpravy (viz obrázek).



Obrázek 19 - Kolekce nodes pro integraci s HA

Před prvním použitím je potřeba nakonfigurovat komunikaci s HA instancí pomocí speciální node *server*.

Integrace komponent brány

Jednotlivá rozhraní a vestavěné koncové prvky je nutné integrovat se softwarem ústředny.

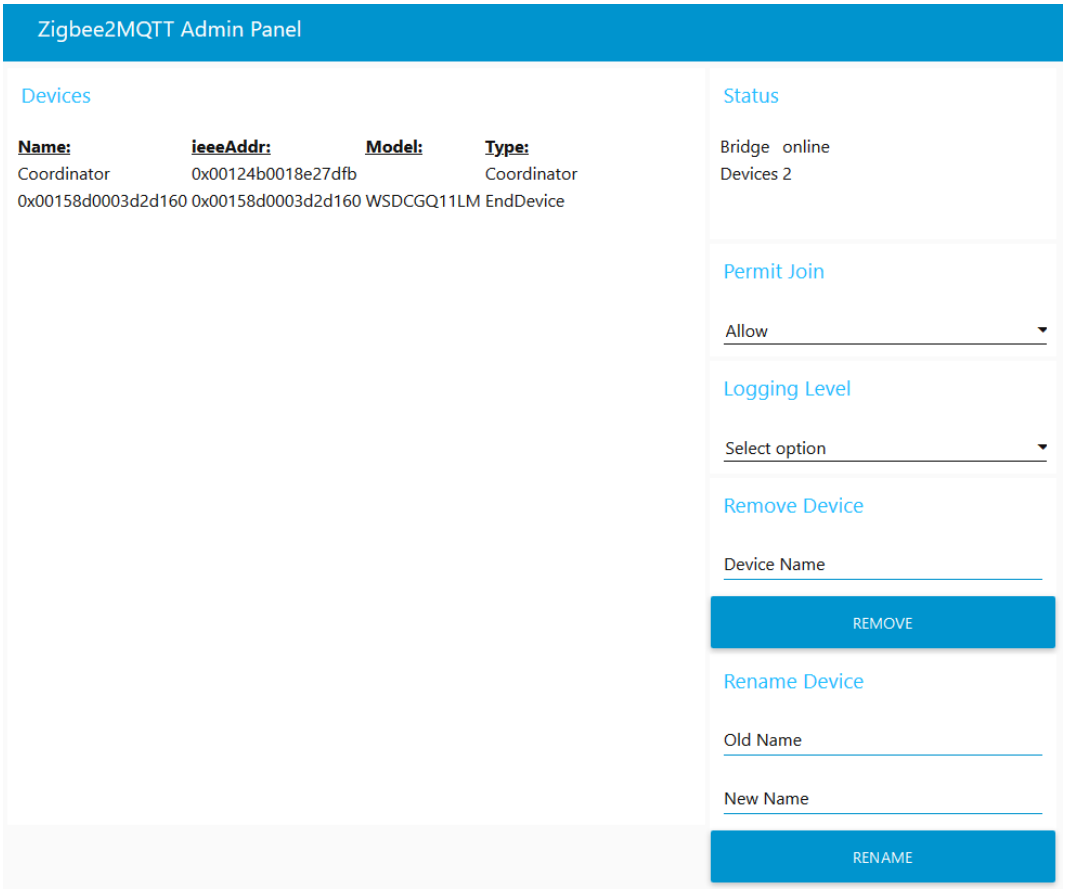
Zigbee

Zigbee2MQTT (dále jen Z2M) lze integrovat s HA pomocí MQTT Discovery.
V konfiguračním souboru Z2M */opt/zigbee2mqtt/data/configuration.yaml* je třeba nastavit direktivu *homeassistant: true*. V konfiguračním souboru HA je pak nutné nastavit:

```
mqtt:
  discovery: true
  broker: localhost
  client_id: 'home_assistant'
  username: 'controller'
  password: 'S3cr3t'
  birth_message:
    topic: 'hass/status'
    payload: 'online'
  will_message:
    topic: 'hass/status'
    payload: 'offline'
```

Konfigurační soubor uvnitř Docker kontejneru je mapován na /usr/share/hassio/homeassistant/configuration.yaml, lze ho ale editovat i pomocí webového rozhraní a add-onu *configurator,* ten lze doinstalovat rovněž přes webové rozhraní – kategorie *Add-on store* v záložce *Supervisor*. Pro aplikaci změn je nutné restartovat HA pomocí záložky *Server Controls* v menu Configuration. Nová zařízení přidaná do Zigbee2MQTT se nyní budou zobrazovat v seznamu zařízení a entit v HA. Defaultně jsou uspořádány podle své Zigbee adresy, pro lepší orientaci je vhodné entitám přidat název (menu Configuration, záložka Entities).

V Node-RED je možné pracovat přímo s MQTT zprávami pomocí node *mqtt in*, každá node se po přidání do nějakého flow nastavuje dvojklikem, ve zobrazeném formuláři stačí nastavit správné téma v poli *topic*. Při prvním použití node je také nutné nastavit parametry brokeru v poli *server*. V Node-RED je také dostupný užitečný doplněk pro správu zařízení připojených pomocí Z2M, dostupný z webu https://github.com/ben423423n32j14e/zigbee2mqttadminpanel. Pro přidání je nutné nainstalovat kolekci *node-red-dashboard* stejným postupem jako kolekci pro HA (viz výše), poté lze importovat příslušný flow. V tomto flow bylo potřeba upravit názvy témat a konfiguraci brokeru, upravená funkční verze je v příloze této práce. Po aktivaci (tlačítkem *Deploy*) je UI dostupné na *http://192.168.10.210:1880/ui/*, viz obrázek.



Obrázek 20 - Node-RED dashboard admin panel for Zigbee2MQTT

Bluetooth a RF 433

Obě komponenty jsou postaveny na OpenMQTTGateway (dále jen OMG), která stejně jako Zigbee2MQTT podporuje MQTT Discovery pro integraci s HA. Konfigurace na straně HA už je hotová (viz předchozí sekce), na straně OMG modulů je potřeba v souboru *User_config.h* povolit navíc modul *ZmqttDiscovery*. Nová zařízení detekovaná OMG se zobrazí v HA v seznamu zařízení a entit.

V Node-RED je opět možné pracovat pomocí node *mqtt in* přímo s MQTT zprávami v příslušných tématech (*NoTGW/OMG_BLE* pro Bluetooth a *NoTGW/OMG_RF* pro RF 433).

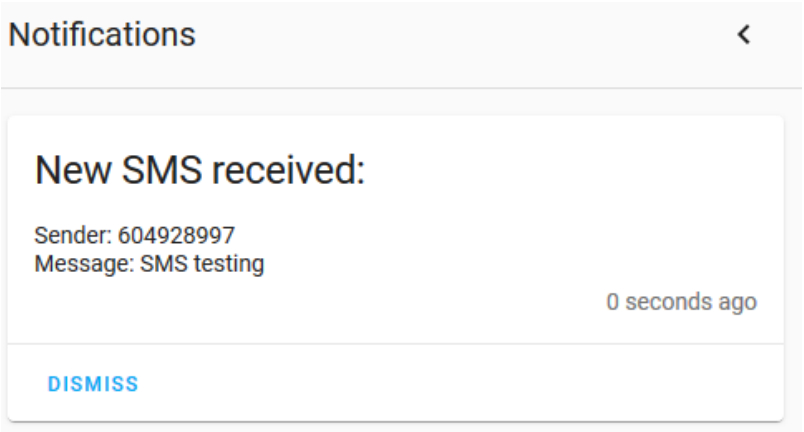
GSM

SMS posílané GSM modulem nebudou využívané jako komunikační prostředek pro uživatele, ale jako součást automatizačních pravidel, proto se s nimi bude pracovat hlavně přes Node-RED. Tam lze stejně jako u předchozích integrací pracovat přímo s MQTT zprávami využitím nodes *mqtt in* a *mqtt out*. Příkladem může být automatizace popsaná níže v sekci Testování komponent.

Pro lepší přehled bude ale i Home Assistant zobrazovat notifikace pro každou přijatou SMS zprávu. K tomu bude využita HA služba *persistent_notification*, ta ve webovém rozhraní HA zobrazí upozornění, jehož přečtení musí uživatel potvrdit odkliknutím

Notifikace se nastavují pomocí HA automatizací, které lze ručně definovat v souboru *configuration.yaml*. Následující záznam definuje vytvoření notifikace na Obrázek 21 pro každou přijatou MQTT zprávu ve formátu popsaném výše v sekci GSM.

```
automation sms_recv:
- alias: 'Display received SMS as notification'
  trigger:
    platform: mqtt
    topic: 'NoTGW/GSM/SMS'
  action:
  - service: persistent_notification.create
    data_template:
      message: "Sender: {{ value_json.sender | is_defined }}\n\
            Message: {{ value_json.content | is_defined }}"
      title: 'New SMS received:'
```



Obrázek 21 - Notifikace přijatých SMS zpráv

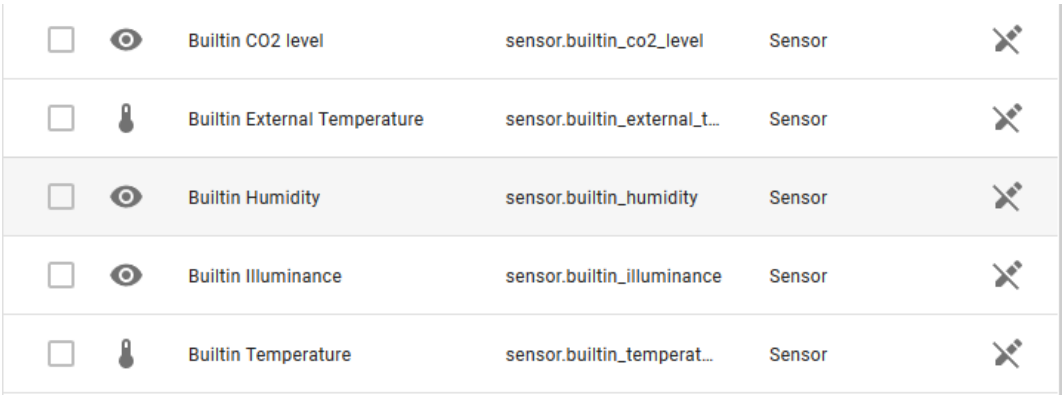
Senzory

V Node-RED lze samozřejmě i se senzory pracovat přímo pomocí node *mqtt in*. V HA je nutné manuálně vytvořit komponenty typu *sensor* platformy *mqtt* editací konfiguračního souboru *configuration.yaml*. Direktiva *value_template* určuje zobrazovanou hodnotu – z MQTT zpráv přijatých v definovaném tématu se načte

jakákoli JSON hodnota s daným klíčem. Funkce ostatních direktiv jsou dostatečně popsány jejich názvem. Syntaxe je YAML, je tedy důležité zachovat odsazení, konkrétní záznamy pro jednotlivé senzory jsou:

```
sensor:
- platform: mqtt
  state_topic: 'NoTGW/Sensors/DHT'
  name: "Builtin Temperature"
  unit_of_measurement: '°C'
  value_template: '{{ value_json.temperature | is_defined }}'
  expire_after: 21600 # 6 hours
- platform: mqtt
  state_topic: 'NoTGW/Sensors/DHT'
  name: "Builtin Humidity"
  unit_of_measurement: '%'
  value_template: '{{ value_json.humidity | is_defined }}'
  expire_after: 21600 # 6 hours
- platform: mqtt
  state_topic: 'NoTGW/Sensors/Temp'
  name: "Builtin External Temperature"
  unit_of_measurement: '°C'
  value_template: '{{ value_json.temperature | is_defined }}'
  expire_after: 21600 # 6 hours
- platform: mqtt
  state_topic: 'NoTGW/Sensors/Light'
  name: "Builtin Illuminance"
  unit_of_measurement: 'lx'
  value_template: '{{ value_json.light | is_defined }}'
  expire_after: 21600 # 6 hours
- platform: mqtt
  state_topic: 'NoTGW/Sensors/Gas'
  name: "Builtin CO2 level"
  unit_of_measurement: 'ppm'
  value_template: '{{ value_json.CO2 | is_defined }}'
  expire_after: 21600 # 6 hours
```

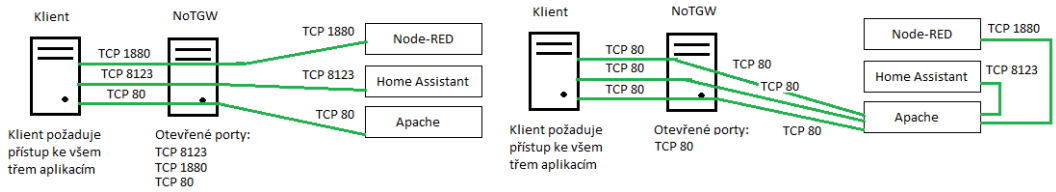
Po restartu HA (viz výše) jsou všechny senzory připojené k Raspberry Pi 3B+ dostupné jako samostatné entity, viz obrázek. DHT senzory v modulech ESP 32 jsou zařazeny pod daný modul v rámci integrace MQTT Discovery.



Obrázek 22 - Entity vestavěných senzorů

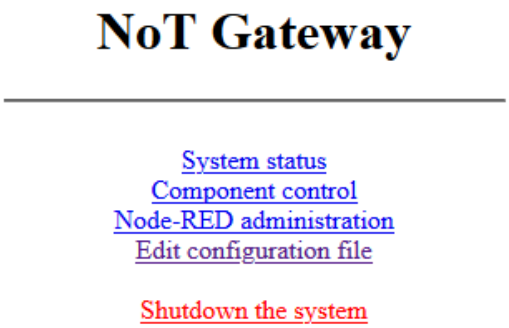
Webové konfigurační rozhraní

Přístup k IP adrese Ethernetového rozhraní vede na login formulář aplikace Home Assistant. Jeden z požadavků ale byl, že celá brána musí být konfigurovatelná pomocí webového rozhraní. Pro tento účel bylo vytvořeno jednoduché webové rozhraní propojující několik aplikací a běžící nad webserverem Apache2, který zároveň slouží jako reverzní proxy. Reverzní HTTP proxy server umožňuje aplikace, které standardně běží na vlastním TCP portu provozovat pouze lokálně na serveru, přistup zvenčí pak musí jít skrz tento proxy a je možno pro něj definovat filtrování, přesměrování, šifrování, zabezpečení heslem aj. Princip je vidět na obrázku.



Obrázek 23 - znázornění přístupu ke službám na serveru bez (vlevo) a s (vpravo) reverzní proxy Apache. Apache zároveň zajišťuje funkci webserveru.

Rozhraní je dostupné na *http://<IP brány>/admin* (tedy při výše uvedeném nastavení *http://192.168.10.210/admin*) a je tvořeno kombinací statických stránek a PHP skriptů. Ty jednak poskytují dvě základní funkce – vypnutí systému a editaci konfiguračního souboru */etc/notgw.conf*, jednak slouží jako rozcestník pro aplikace za reverzní proxy.



Obrázek 24 - Rozcestník webové administrace

První z těchto aplikací je správce daemonů realizovaný projektem *SysDWeb*, díky kterému lze zapínat a vypínat jednotlivé senzory a komponenty, případně je restartovat při problémech. Druhou částí je monitor systémových prostředků *Glances*, který v jednom okně prohlížeče přehledně shrnuje vytížení systému, zobrazuje v reálném čase aktualizované údaje o utilizaci CPU a RAM, přístupech k disku a síti, teplotě SoC, seznam běžících procesů a další. Třetí aplikací je webový frontend Node-RED.

Nejprve bylo nutné nainstalovat webový server *Apache* společně s dalšími potřebnými balíčky a zapnout moduly nutné pro funkci reverzního proxy.

```
apt-get install apache2 apache2-utils php python3-bottle glances
a2enmod proxy proxy_http rewrite
```

Následovala instalace *SysDWeb*. Zdrojové kódy projektu jsou k dispozici na githubu jeho vývojáře https://github.com/ogarcia/sysdweb. Virtuální prostředí (venv), které je při instalaci vytvořeno zajišťuje, že aplikace bude mít k dispozici správné verze Pythonu a všech pomocných nástrojů bez zásahu do verzí používaných ve zbytku OS.

```
apt-get install libsystemd-dev
git clone https://github.com/ogarcia/sysdweb.git
virtualenv3 ./sysdweb-venv
source ./sysdweb-venv/bin/activate
cd sysdweb
pip install -r requirements.txt
python setup.py install
```

V konfiguračním souboru */etc/sysdweb.conf* je nutné nastavit systemd units, které bude nástroj spravovat, síťové rozhraní, na kterém bude poslouchat (localhost kvůli reverzní proxy) a přístupová práva. Konfigurace služeb je velmi intuitivní, ukázka záznamu pro senzor DHT22:

```
[DHT]
title = DHT temperature and humidity sensor
unit = sensor-dht.service
```

Samotný *SysDWeb* běží jako daemon, systemd unit file se sám vytvoří při instalaci, po editaci konfiguračního souboru ho stačí spustit pomocí *systemctl start sysdweb*.

U komponenty Glances je potřeba upravit její systemd unit file */lib/systemd/system/glances.service*, aby se Glances spouštěl jako webová aplikace a poslouchal pouze na lokálním stroji. Direktivu *ExecStart* je nutné nastavit následovně:

```
ExecStart = /usr/bin/glances -w -B 127.0.0.1
```

Instalace a konfigurace Node-RED je popsána výše. U webserveru Apache2 byla celá konfigurace kvůli zjednodušení umístěna do souboru pro defaultní virtualhost (webserver může obsluhovat více domén na jedné IP adrese, každý virtualhost reprezentuje jednu doménu. Defaultní virtualhost obsluhuje doménu příslušející k IP adrese serveru a také všechny nedefinované. Apache je velmi komplexní software a detaily jeho konfigurace jsou nad rámec této práce, dokumentace je dostupná na webu *httpd.apache.org*).

Cesta k souboru je */etc/apache2/sites-enabled/000-default.conf*, připravený konfigurační soubor je dostupný v příloze této práce. V souboru je mimo jiné nastavena autentizace heslem, pro ni je nutno vytvořit seznam uživatelů. Následu-

jící příkaz přidá nového uživatele *gwadmin* (příkaz je interaktivní, na heslo se zeptá po spuštění). Je třeba zadat cestu k seznamu specifikovanou v direktivě *AuthUserFile* konfiguračního souboru. Pokud soubor v dané cestě neexistuje, příkaz ho vytvoří:

htpasswd -c /etc/apache2/htpasswd gwadmin

Po dokončení nastavení je možné do kořenu webu ve */var/www/html/admin* umístit příslušné soubory dostupné v příloze této práce.

- `index.html` – index webu, zde je to zmíněný rozcestník s odkazy.
- `notgw.conf` – symbolický link (odkaz) na konfigurační soubor /etc/notgw.conf
- `halt.php` – PHP skript pro korektní vypnutí systému
- `editconf.php` – PHP skript pro editaci notgw.conf

Aby všechny integrace fungovaly korektně, je třeba povolit uživateli, pod kterým běží Apache (www-data) zápis do konfiguračního souboru brány:

chgrp www-data /etc/iotgw.conf
chmod g+w /etc/iotgw.conf

Dále je nutné mu umožnit volat příkaz *halt* pro korektní vypnutí systému. Linux má k tomuto účelu program *sudo*, který umožňuje selektivně udělovat běžným uživatelům práva pro spouštění příkazů jako root. Jeho konfigurace se edituje příkazem *visudo*, na konec souboru, který příkaz otevře, je třeba přidat záznam:

www-data ALL = NOPASSWD: /sbin/halt

Tedy povolit uživateli *www-data* spouštět na jakémkoli stroji (zde irelevantní) příkaz *halt* (zde s absolutní cestou) bez nutnosti zadávat své heslo. Po restartu apache je webové rozhraní připraveno k použití:

systemctl restart apache2

Pro veškeré nastavení a práci se systémem je samozřejmě také k dispozici SSH. Při běžném provozu brány by ale přístup přes SSH neměl být potřeba, veškerou konfiguraci lze provést právě pomocí zde popsaného webového rozhraní, případně pomocí webového rozhraní HA.

Nasazení software

Ke konzistentnímu nasazení složitějšího software, kterým celek brány je, se obvykle přistupuje jedním ze tří způsobů

- instalační image – obraz disku s již nainstalovaným OS a veškerým dalším software, např. Raspbian nebo Home Assistant OS. Nejjednodušší způsob, image je však vždy určen pro konkrétní platformu (např. Raspberry Pi) a je nutné ho udržovat aktuální.

- Instalační skript – skript, který postupně projde všechny kroky manuální instalace (jako je popsána např. v této práci). Lze jednoduše parametrizovat, při spuštění je tedy možné zadat základní údaje (např. IP adresu systému, přístupové údaje etc.). Také je možné rozdělení instalačního procesu na vhodné kroky s reportováním případných chyb v každém z nich a skripty je možné psát tak, aby fungovaly na více platformách. Příkladem může být např. instalátor Node-RED.
- Instalace pomocí mechanismu OS – příkladem mohou být balíčkovací systémy různých distribucí OS Linux (RPM, APT...). Toto je vhodné spíše pro ucelenou aplikaci, ne pro systém složený z více různorodých aplikací.

Původní myšlenka byla připravit instalační skript v jazyce Bash po vzoru instalátoru Node-RED. V průběhu realizace se ale ukázalo několik problémů. Mnoho komponent brány není instalováno pomocí balíčků z repozitářů OS, ale právě pomocí vlastních instalačních skriptů. Ty jsou určeny pro interaktivní spouštění z příkazové řádky a při spouštění v rámci jiného skriptu mají některé z nich problém reportovat chyby (Node-RED, Glances), případně vůbec doběhnout (SysD-Web). Druhý problém byl, že skript je potřeba spustit v terminálu na připraveném Raspberry Pi. Přípravu SBC (viz sekce Raspberry Pi) je tedy potřeba provést ručně předem. V neposlední řadě pak některé procesy naskriptovat nelze, kromě přípravy SBC by tak bylo potřeba dodělat ručně ještě nahrání profilu pro Home Assistant, obnovení Flows pro Node-RED a naprogramování ESP32 modulů.

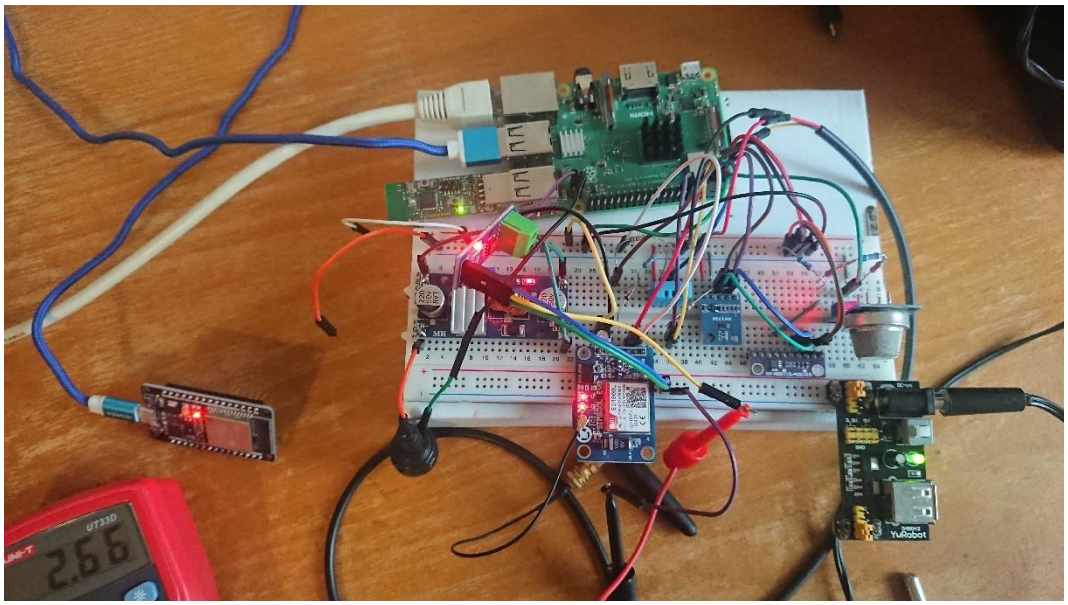
Po několika neuspokojivých iteracích byl místo instalačního skriptu vytvořen instalační image s kompletním software brány připravený k použití hned po nahrání na SD kartu Raspberry Pi 3B+. Jediná výraznější nevýhoda tohoto přístupu je, že nelze předem nastavit parametry zařízení, pro první připojení je tedy potřeba použít defaultní IP adresu a přístupové údaje (viz sekce Raspberry Pi).

Image je dostupný v příloze této práce a instaluje se stejným postupem, jako čistý Raspbian OS (opět viz sekce Raspberry Pi). Jako krok navíc ale stále zůstává nutnost manuálně naprogramovat moduly ESP32. Po dokončení obou kroků je systém připraven ve stavu, v jakém byl při testování (viz sekce Testování komponent). Defaultní údaje pro přístup jsou:

- IP adresa Ethernetového rozhraní: *192.168.10.210*
- Uživatel *gwadmin*
- Heslo *S3cr3t*

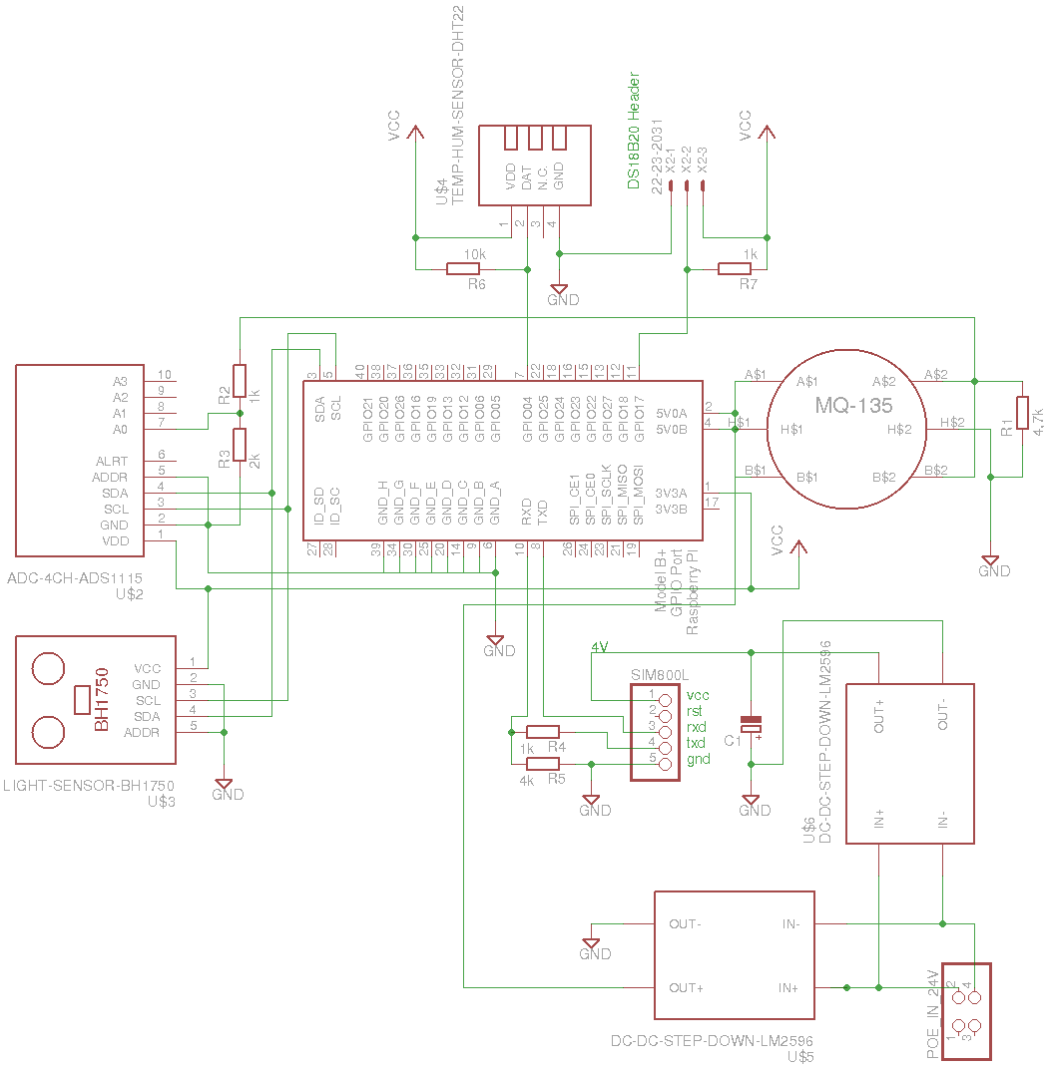
Konstrukce

V této fázi je z hlediska funkčnosti brána hotová a připravená k testování. Samotné fyzické zařízení je ale stále ve fázi prototypu, viz obrázek.



Obrázek 25 - Prototyp

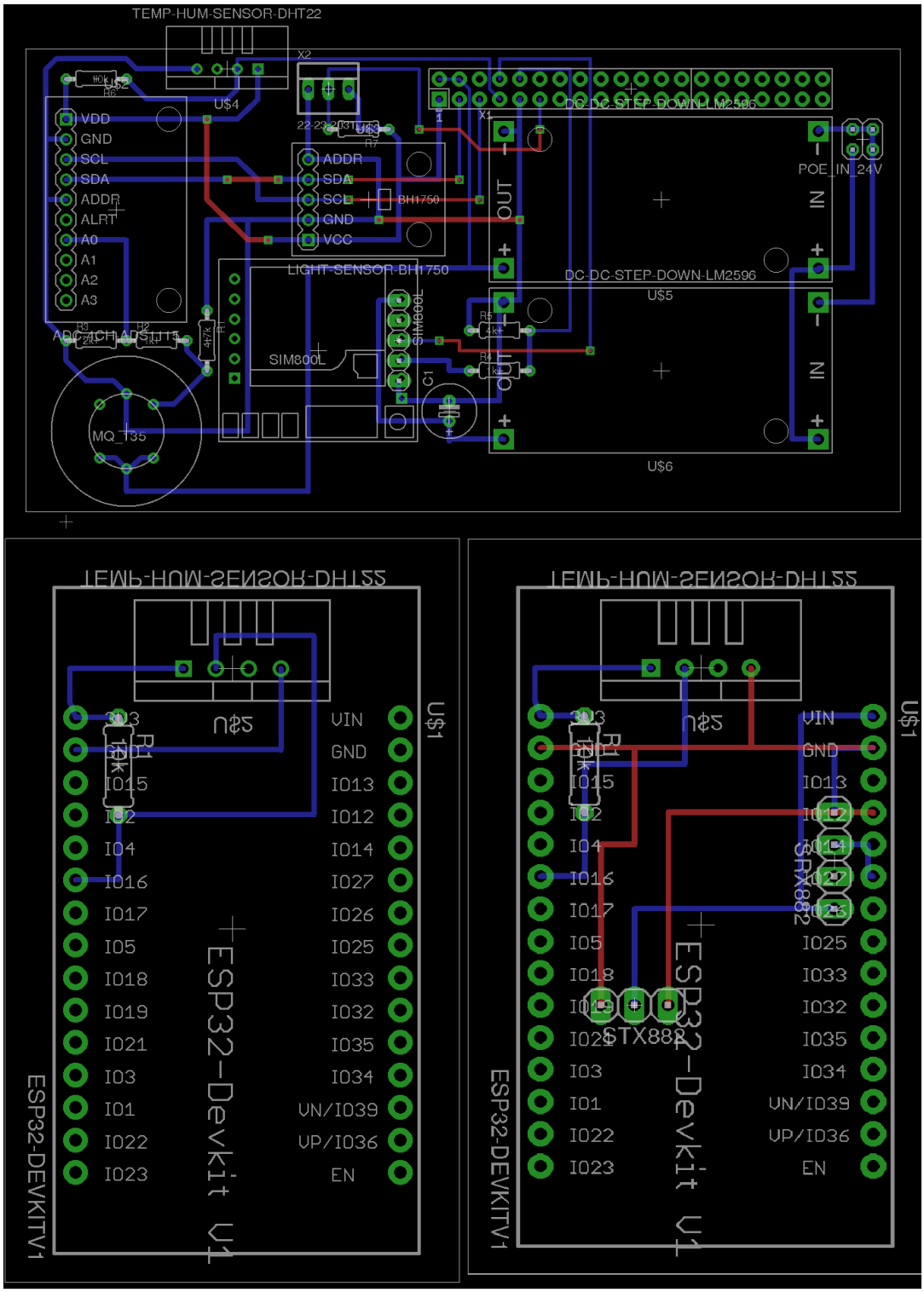
Pro bránu byly navrženy tři desky plošných spojů. První z nich je určena jako rozšiřující deska (HAT) pro Raspberry Pi 3b+ a obsahuje všechna komunikační rozhraní a vestavěné sensory. Druhé dvě jsou desky pro odpojitelné moduly s ESP32, které slouží jako komunikační most pro Bluetooth a RF 433 a zároveň jako externí senzory teploty a vlhkosti. V příloze práce jsou dostupné návrhové soubory a schémata pro nástroj Autodesk Eagle, který byl při návrhu použit. Na Obrázek 26 - Schéma centrálního modulu brány je vidět schéma centrální jednotky. Místo DC-DC převodníku XL4015 a navazujícího lineárního regulátoru pro modem SIM800L byly nakonec pro napájení použity dva převodníky LM2596, z nichž jeden poskytuje 5V pro senzory a Raspberry Pi, druhý pak 4V pro modul modemu.



Obrázek 26 - Schéma centrálního modulu brány

Původní záměr byl vejít se s rozměry do půdorysu Raspberry Pi, to znamená použít k návrhu samostatné integrované obvody a diskrétní součástky. To se ale ukázalo jako nedosažitelné hned z několika důvodů. Použité komponenty se prodávají jako hotové moduly (malé DPS opatřené header konektorem) určené k prototypování pomocí platforem jako je např. Arduino. Čipy použité v těchto obvodech se ale ve většině případů dají samostatně koupit jen ve velkoobchodním množství (řádově stovky kusů). Cenově pak vychází samostatný čip zhruba stejně jako celý modul. U jiných modulů (typicky DC-DC konvertory) je nákup jednotlivých součástek zase paradoxně několikanásobně dražší než hotový modul. Třetím důvodem jsou složitější čipy jako SIM800L (LGA pouzdro), nebo ESP32 (BGA pouzdro), které v domácích podmínkách bez specializovaných přístrojů nelze osadit. Jedinými komponentami, které se podařilo integrovat jako samostatné součástky jsou senzory MQ-135 a DHT22. Výsledná deska byla tedy navržena jako jednoduchý, jednovrstvý plošný spoj a slouží pro propojení jednotlivých

modulů. Rozložení jednotlivých desek je vidět na obrázku 27, bohužel není k dispozici ukázka osazeného zařízení, desky v době odevzdání této práce stále čekají na dodání výrobcem.



Obrázek 27 - Rozložení desek plošných spojů centrální jednotky (nahoře), ESP32 Bluetooth modulu (vlevo dole) a ESP32 RF modulu (vpravo dole)

Testy a srovnání

Testování komponent

Brána byla v provozu nepřetržitě po dobu více než dvou měsíců, po které byla využívána pro připojení alespoň jednoho koncového prvků pomocí každého z podporovaných komunikačních protokolů. Pro tuto jednoduchou aplikaci byl výkon Raspberry Pi 3B+ více než dostatečný, využití RAM bylo okolo 40 % a využití CPU kolem 10–20 %. Náročnost na výkon závisí hlavně na počtu aktivních automatizačních pravidel, dle mého odhadu je výkon dostačující pro byt či menší rodinný dům. Pro větší objekty může být nutné zmigrovat funkce ústředny na jiný stroj.

Odběr proudu zařízení včetně obou ESP32 modulů je cca 1A, špičky až 2A nastávají pouze během odesílání SMS (v datasheetu modemu SIM800L je uvedeno, že 2A může ve špičce odebírat samotný modem, toho se ale během testů nepodařilo dosáhnout). Spotřeba je tedy 5W v běžném stavu, 10W špičkově, PoE část Raspberry Pi 3B+ je dimenzována na 15W, z hlediska napájení má tedy brána značnou rezervu pro případné rozšíření.

Všechny integrace fungovaly v pořádku, problém byl pouze s použitím nevhodné SD karty (je potřeba aplikační třída, viz výše) a s náhodnými restarty SMS modemu (řešení popsáno v sekci GSM). Následují testy ověření funkce jednotlivých komponent

MQTT

Pro diagnostiku MQTT jsou na Linuxu k dispozici příkazy *mosquitto_sub* a *mosquitto_pub* pro čtení, respektive zápis libovolných témat. Takto lze otestovat interakci všech komponent brány. Testovací uživatel *controller*, má plný přístup do všech MQTT témat.

Senzory

Každý z vestavěných senzorů periodicky posílá data do MQTT v intervalech nastavených v souboru */etc/notgw.*conf dokud běží příslušný daemon. Pro test tedy stačí zobrazit všechny podtémata *NoTGW/Sensors* (wildcard #) a zkontrolovat, že žádná data nechybí a že JSON řetězce jsou v pořádku.

mosquitto_sub -d -h 192.168.10.210 -u controller -P S3cr3t -t NoTGW/Sensors/#

Vybrané výstupy pro všechny 4 senzory:

Client mosqsub/14365-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/Sensors/DHT', ... (67 bytes))
{"sensor":"DHT","temperature":30.0,"temp_unit":"C","humidity":27.0}

Client mosqsub/14491-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/Sensors/Temp', ... (52 bytes))
{"sensor":"DS18B20","temperature":29.312,"unit":"C"}
Client mosqsub/14491-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/Sensors/Gas', ... (56 bytes))
{"sensor":"MQ-135","CO2":616.0724452900344,"unit":"ppm"}
Client mosqsub/14491-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/Sensors/Light', ... (57 bytes))
{"sensor":"BH1750","light":459.1666666666667,"unit":"lx"}

Zigbee

U Zigbee2MQTT lze přes MQTT lze přistupovat k připojeným zařízením a zároveň komponentu nastavovat. Pro nastavení lze také použít grafické rozhraní v rámci Node-RED dashboard, viz výše. Stejně jako u senzorů je vhodné zobrazit všechna podtémata *NoTGW/Zigbee*.

mosquitto_sub -d -h 192.168.10.210 -u controller -P s3cr3t -t NoTGW/Zigbee/#

Hned po připojení se zobrazí aktuální konfigurace:

Client mosqsub/16266-MyshZEN received PUBLISH (d0, q0, r0, m0, 'IoTGW/Zigbee/bridge/config', ... (208 bytes))
{"version":"1.12.0","commit":"840b9d9","coordinator":{"type":"zStack12","meta":{"transportrev":2,"product":0,"majorrel":2,"minorrel":6,"maintrel":3,"revision":20190608}},"log_level":"info","permit_join":false}

Nastavení se provádí v tématu *NoTGW/Zigbee/config*, např. lze zapnout přidávání nových zařízení:

mosquitto_pub -d -h 192.168.10.210 -u controller -P S3cr3t -t NoTGW/Zigbee/bridge/config/permit_join -m "true"

Brána odpoví zprávou o novém stavu konfigurace:

Client mosqsub/16266-MyshZEN received PUBLISH (d0, q0, r1, m0, 'NoTGW/Zigbee/bridge/config', ... (208 bytes))
{"version":"1.12.0","commit":"840b9d9","coordinator":{"type":"zStack12","meta":{"transportrev":2,"product":0,"majorrel":2,"minorrel":6,"maintrel":3,"revision":20190608}},"log_level":"info","permit_join":true}

Nastavení není persistentní, pro trvalé úpravy je třeba editovat konfigurační soubor */opt/zigbee2mqtt/data/configuration.yaml*. Následující příkaz vypíše seznam připojených zařízení:

mosquitto_pub -d -h 192.168.10.210 -u controller -P S3cr3t -t NoTGW/Zigbee/bridge/config/devices/get -m ''

Dlouhá odpověď v JSONu není příliš čitelná, ale obsahuje požadované informace:

Client mosqsub/16286-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/Zigbee/bridge/config/devices', ... (1056 bytes))
[{"ieeeAddr":"0x00124b0018e27dfb","type":"Coordinator","networkAddress":0,"friendly_name":"Coordinator","softwareBuildID":"zStack12","dateCode":"20190608","lastSeen":1588666742116},{"ieeeAddr":"0xccccccfffe9763af","type":"Router","networkAddress":37658,"model":"LED1624G9","vendor":"IKEA","description":"TRADFRI LED

bulb E14/E26/E27 600 lumen, dimmable, color, opal white","friendly_name":"0xccccccfffe9763af","manufacturerID":4476,"manufacturerName":"IKEA of Sweden","powerSource":"Mains (single phase)","modelID":"TRADFRI bulb E14 CWS opal 600lm","hardwareVersion":1,"softwareBuildID":"1.3.002","dateCode":"20170315","lastSeen":1588662529676},{"ieeeAddr":"0x00158d0003d2d160","type":"EndDevice","networkAddress":36904,"model":"WSDCGQ11LM","vendor":"Xiaomi","description":"Aqara temperature, humidity and pressure sensor","friendly_name":"0x00158d0003d2d160","manufacturerID":4151,"manufacturerName":"LUMI","powerSource":"Battery","modelID":"lumi.weather","hardwareVersion":30,"softwareBuildID":"3000-0001","dateCode":"20161129","lastSeen":1588665288452}]

Mimo jiné tam lze najít ID připojené žárovky IKEA Tradfri, pomocí kterého je možné ji ovládat, následující příkaz žárovku rozsvítí modrou barvou:

mosquitto_pub -d -h 192.168.10.250 -u controller -P s3cr3t -t NoTGW/Zigbee/0xccccccfffe9763af/set -m '{"state": "ON","brightness":255,"color": { "rgb": "0,0,255" } }'

Bluetooth a RF 433 MHz

Projekt OpenMQTTGateway má velmi podobnou filozofii jako Zigbee2MQTT, má ale o dost méně možností nastavení. Detekce a přidávání nových zařízení je automatické a je stále zapnuto, je pouze možné přidat MAC adresy zařízení, která se nemají připojovat na blacklist, ty potom brána ignoruje. Po připojení do přiřazeného tématu lze vidět data, která do něj periodicky zapisují všechna dostupná zařízení:

mosquitto_sub -d -h 192.168.10.210 -u controller -P S3cr3t -t NoTGW/OMG_BLE/#

Zde je to senzor DHT22 připojený přímo k ESP32 a koncový prvek Xiaomi Mija, (což je také senzor teploty a vlhkosti) připojený přes BLE. U něj je navíc reportován stav baterie a parametry BLE komunikace. První dvě zprávy jsou informace o stavu zařízení a verzi OpenMQTTGateway:

Client mosqsub/20882-MyshZEN received PUBLISH (d0, q0, r1, m0, 'NoTGW/OMG_BLE/LWT', ... (6 bytes))
online
Client mosqsub/20882-MyshZEN received PUBLISH (d0, q0, r1, m0, 'NoTGW/OMG_BLE/version', ... (6 bytes))
v0.9.3
Client mosqsub/18156-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/OMG_BLE /DHTtoMQTT/dht1', ... (20 bytes))
{"hum":40,"temp":28}
Client mosqsub/18156-MyshZEN received PUBLISH (d0, q0, r0, m0, 'IoTGW/OMG_BLE/BTtoMQTT/2EBC1EB363BA', ... (80 bytes))
{"id":"2e:bc:1e:b3:63:ba","manufacturerdata":"L","rssi":-89,"distance":21.51847}
Client mosqsub/18156-MyshZEN received PUBLISH (d0, q0, r0, m0, 'IoTGW/OMG_BLE/BTtoMQTT/4C65A8D2791A', ... (23 bytes))
{"tem":24.4,"hum":39.5}
Client mosqsub/21147-MyshZEN received PUBLISH (d0, q0, r0, m0, 'IoTGW/OMG_BLE/BTtoMQTT/4C65A8D2791A', ... (11 bytes))
{"batt":87}

Analogicky lze zobrazit data zařízení připojených přes RF 433:

mosquitto_sub -d -h 192.168.10.210 -u controller -P S3cr3t -t NoTGW/OMG_RF/#

GSM

V případě GSM komponenty je třeba ověřit korektní formát JSON zpráv, správnou práci s úložištěm v souboru */var/local/GSM/SMS.messages* a samotné odesílání/příjem. Stejně jako u ostatních testů je nejprve potřeba zobrazit si všechny zprávy příslušného tématu:

mosquitto_sub -d -h 192.168.10.210 -u controller -P S3cr3t -t NoTGW/OMG_BLE/#

Přijatá testovací SMS se zobrazila jako:

Client mosqsub|21588-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (74 bytes))
{"info":"smsrecv","id":80,"sender":"+420604928997","content":"One more time"}

Vypsání seznamu SMS pomocí *{"action":"smslist"}* poté vrátilo:

Client mosqsub|21588-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (787 bytes))
{"info":"smslist","list": [{"id":70,"sender":"+420604928997","received":"01.12.20-02:30","content":"A"}, {"id":73,"sender":"+420604928997","received":"01.12.20-02:30","content":"Jshsh"}, {"id":74,"sender":"+420604928997","received":"01.12.20-21:55","content":"Lol"}, {"id":75,"sender":"+420602398309","received":"01.12.20-22:02","content":"=^^="}, {"id":76,"sender":"+420604928997","received":"01.12.20-22:30","content":"Oora"}, {"id":77,"sender":"+420604928997","received":"01.12.20-22:57","content":"Last"}, {"id":78,"sender":"+420604928997","received":"01.12.20-23:27","content":"It works"}, {"id":79,"sender":"+420604928997","received":"02.12.20-00:21","content":"restarted daemon"}, {"id":80,"sender":"+420604928997","received":"02.12.20-00:44","content":"One more time"}]}

Následoval test smazání dvou ze zpráv a opětovného vypsání seznamu:

Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (27 bytes))
{"action":"smsdel","id":79}
Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (32 bytes))
{"info":"SMS_del_succ","id":79"}
Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (27 bytes))
{"action":"smsdel","id":77}
Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (32 bytes))
{"info":"SMS_del_succ","id":77"}
Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (20 bytes))
{"action":"smslist"}
Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (608 bytes))
{"info":"smslist","list": [{"id":70,"sender":"+420604928997","received":"01.12.20-02:30","content":"A"}, {"id":73,"sender":"+420604928997","received":"01.12.20-02:30","content":"Jshsh"}, {"id":74,"sender":"+420604928997","received":"01.12.20-

21:55","content":"Lol"}, {"id":75,"sender":"+420602398309","received":"01.12.20-22:02","content":"=^^="}, {"id":76,"sender":"+420604928997","received":"01.12.20-22:30","content":"Oora"}, {"id":78,"sender":"+420604928997","received":"01.12.20-23:27","content":"It works"}, {"id":80,"sender":"+420604928997","received":"02.12.20-00:44","content":"One more time"}]}

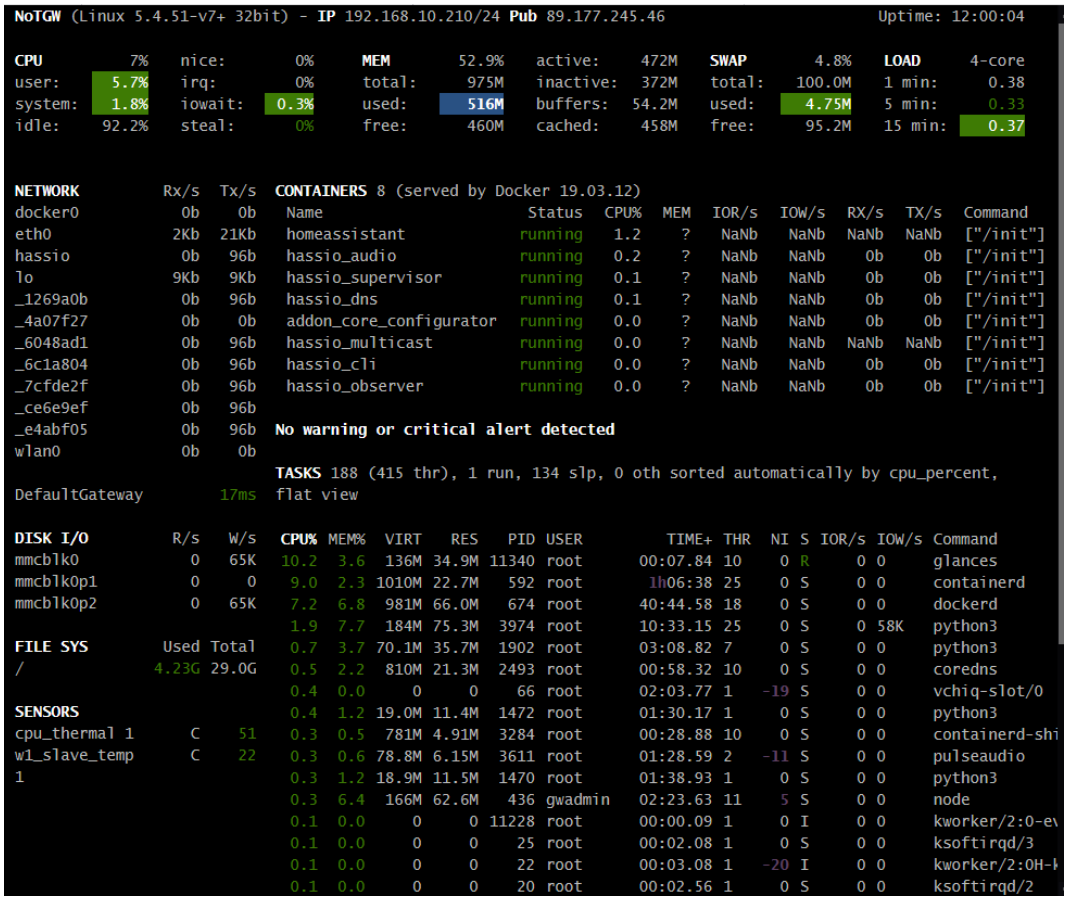
Všechny testy dopadly dle očekávání, odeslání SMS zprávy proběhlo rovněž v pořádku:

Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (64 bytes))
{"action":"smssend","dest":"604928997","content":"PoE powered!"}
Client mosqsub|21839-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/GSM/SMS', ... (75 bytes))
{"info":"SMS_send_succ","destination":"604928997","content":"PoE powered!"}

Webové konfigurační rozhraní

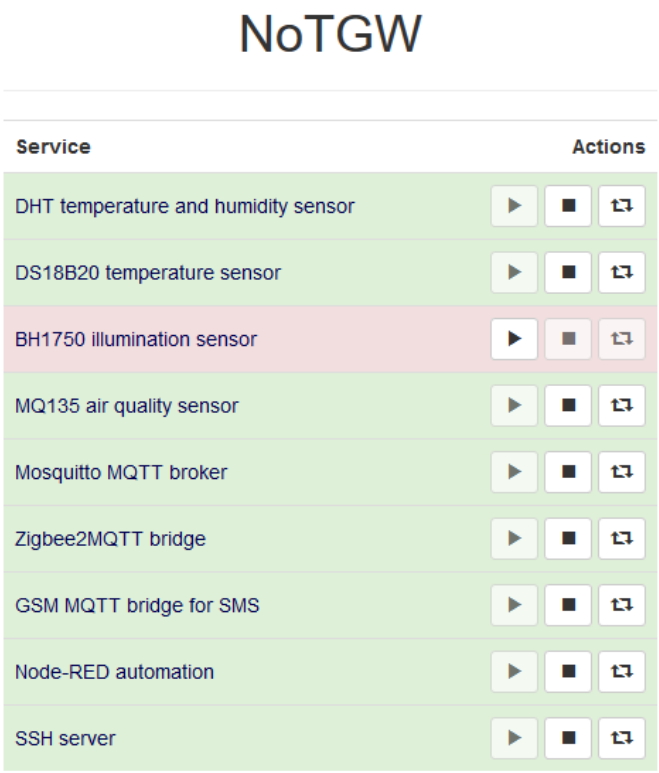
Rozhraní je dostupné na *http://192.168.10.210/admin* (IP samozřejmě záleží na konfiguraci ethernetového rozhraní) a po přihlášení se zobrazí rozcestník na Obrázek 24.

První odkaz otevře systémový monitoring Glances, viz Obrázek 28. Zde je vidět využití CPU, paměti, sítě a disku (SD karty) jednak celkové, jednak rozdělené podle běžících procesů. Také je vidět seznam aktivních Docker kontejnerů a senzory. 1-wire teploměr DS18B20 je díky podpoře sběrnice přímo v jádře OS také brán jako systémový senzor a je zde zobrazen. Zajímavý parametr je *iowait*, ten ukazuje procento času, které CPU stráví čekáním na vstupně/výstupní operace, zde hlavně čtení a zápis dat na pevný disk. Při použití SD karty určené pro sekvenční zápis tento parametr při zátěži dosahuje 50-80% a systém je znatelně zpomalen.



Obrázek 28 - Systémový monitoring Glances

Druhý odkaz v rozcestníku otevře nástroj *SysDWeb* pro správu daemonů obsluhujících komponenty brány. Každý daemon má tři operace – start, stop, restart – reprezentované třemi tlačítky na pravé straně. Ovládat lze všechny komponenty kromě ESP32 modulů a Home Assistanta, ten je spravován přes vlastní web UI. Lze zde také zakázat SSH přístup k bráně vypnutím příslušného daemona. Na Obrázek 29 je test vypnutí senzoru BH1750 pomocí tohoto nástroje.



Obrázek 29 - Správa komponent

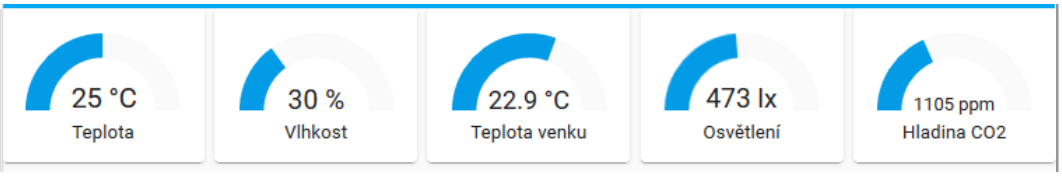
Třetí odkaz vede na webové rozhraní Node-RED. Čtvrtý pak otevře hlavní konfigurační soubor brány */etc/notgw.conf* v jednoduchém textovém editoru se dvěma tlačítky pro potvrzení nebo zrušení provedených změn. Soubor má syntaxi *.ini* konfiguračních souborů kvůli jednoduchému importu Python knihovnou *configparser*. Vzhledem k tomu, že komunikační komponenty se nastavují pomocí MQTT, jsou v souboru aktuálně pouze direktivy pro nastavení dotazovacích intervalů vestavěných senzorů. Při testování se objevil problém, kdy editor neměl práva pro zápis do souboru. Editor běží pod uživatelem webového serveru *www-data*, bylo tedy třeba práva přidat:

```
chgrp www-data /etc/notgw.conf
chmod g+rw /etc/notgw.conf
```

Poslední odkaz bránu vypne a vrátí informaci, zda je možné ji bezpečně odpojit od napájení. Na pozadí je volán příkaz *halt*, který potřebuje pro spuštění *root* práva, po prvním neúspěšném testu bylo tedy třeba nastavit spouštění přes *sudo*, viz výše.

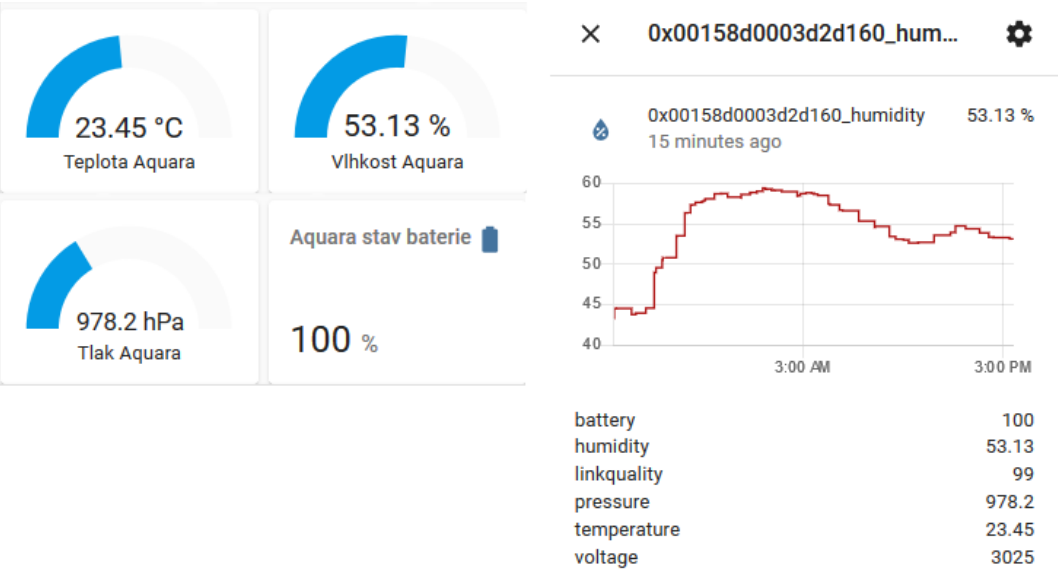
Home Assistant

Přidání vestavěných senzorů do HA bylo provedeno při jeho instalaci, na obrázku je vidět výsledek testu namapování těchto senzorů na karty typu *gauge* v ovládacím rozhraní Lovelace. Zobrazovaná data odpovídají skutečnosti.



Obrázek 30 - hodnoty z vestavěných senzorů v aplikaci Home Assistant

Komunikace Zigbee byla testována připojením kombinovaného senzoru teploty, vlhkosti a tlaku Xiaomi Aqara. Párování nových zařízení lze povolit pomocí administračního web UI Zigbee2MQTT, viz Obrázek 20 v sekci Software ústředny. Párování senzoru se aktivuje stisknutím tlačítka na jeho vrchní straně po dobu cca 5 s. Díky MQTT Discovery se entity příslušející senzoru automaticky přidaly do seznamu entit v HA. Na Obrázek 31 je opět vidět jejich namapování na *gauge* karty v Lovelace (vlevo) společně s detaily a historií jedné z entit (vpravo).

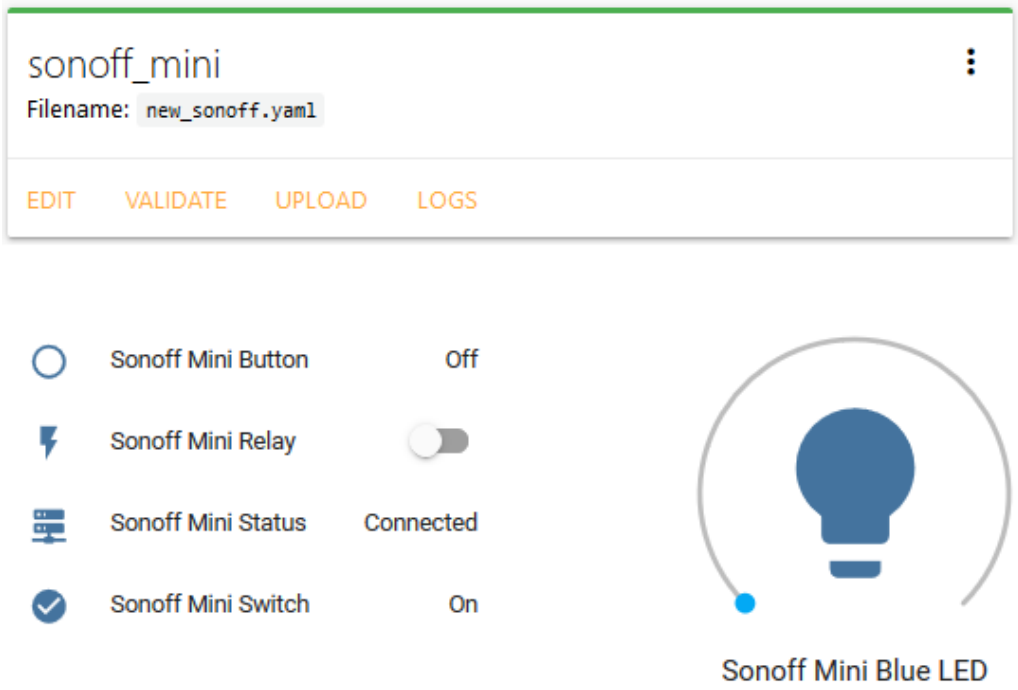


Obrázek 31 - data senzoru Xiaomi Aqara připojeného přes Zigbee

Obdobně proběhlo testování Bluetooth a RF zařízení připojených pomocí OpenMQTTGateway. Poslední test byl připojení koncového prvku přímo přes hostovanou Wi-Fi síť 192.168.5.0/24. Koncovým prvkem byl spínač Sonoff Mini DIY s již předem nahraným firmware ESPHome a konfiguračním souborem. Konfigurační soubor je k dispozici v příloze této práce. Pokud má Sonoff IP adresu mimo rozsah použité Wi-Fi sítě, připojení selže a Sonoff začne po chvíli vysílat vlastní Wi-Fi síť. K té je možné se připojit a pomocí zobrazeného webového rozhraní nakonfigurovat přístup k Wi-Fi poskytované bránou. V případě že Sonoff žádnou síť nevysílá, případně sek ní nelze připojit, je nutné konfiguraci nahrát přes USB. Postup je uveden na webu projektu *esphome.io* [18].

Projekt ESPHome nabízí i addon do HA, ten byl nainstalován standardně přes záložku *Add-on store* v nabídce *Supervisor* webového rozhraní HA. Po in-

stalaci je ESP Home zobrazen v záložkách webového rozhraní HA a nabízí podrobného průvodce pro přidání nových zařízení. Tento průvodce slouží pouze k vytváření konfiguračních souborů, vytvořený soubor je pak třeba nahrát do zařízení (viz výše) a teprve potom je zařízení dostupné v HA. Vzhledem k tomu, že konfigurační soubor už byl vytvořen a nahrán, stačí průvodce proklikat do konce. U nově přidaného zařízení je poté třeba zvolit Edit, do zobrazeného pole zkopírovat konfiguraci, uložit a zvolit Upload (viz obrázek). Nastavení je nyní synchronizováno v ESPHome i samotném zařízení a mělo by dojít k autodetekci jeho entit Home Assistantem. Výsledek testu je vidět na obrázku.

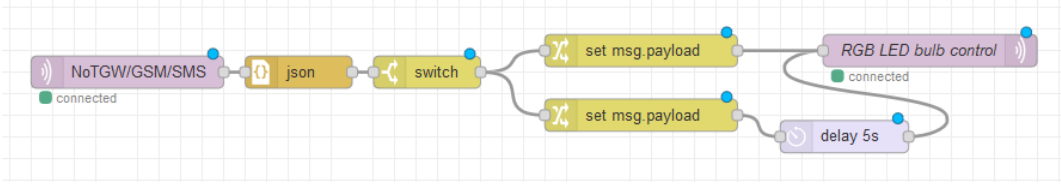


Obrázek 32- Integrace Sonoff Mini DIY připojeného k hostované Wi-Fi síti pomocí firmware ESPHome (nahoře) a příslušné Lovelace karty (dole)

Node-RED

Pro test Node-RED byla vytvořena jednoduchá automatizace, která po přijetí SMS zprávy "blue" rozsvítí na 5 sekund RGB LED žárovku Ikea Tradfri připojenou přes Zigbee modrou barvou.

Na prvním obrázku je vidět vytvořená sekvence nodes, na druhém pak detaily nastavení některých z nich.

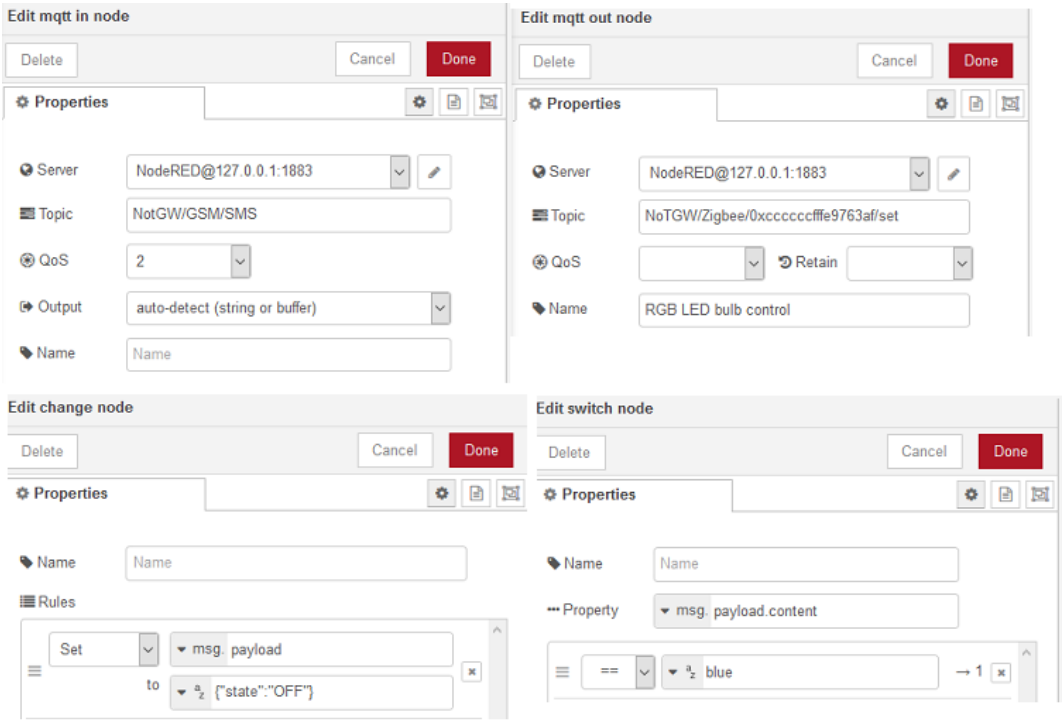


Obrázek 33 - Node-RED testovací sekvence

První node *mqtt in* přijímá zprávy z tématu *NoTGW/GSM/SMS*. V kolonce *Server* je třeba vyplnit nastavení MQTT brokeru, Node-RED si toto nastavení uloží jako šablonu pro opětovné použití. Node *json* provede konverzi JSON stringu vygenerovaného pro každou příchozí SMS (viz sekce GSM) na interní JavaScript object reprezentaci, díky tomu je následně možné v node *switch* přistupovat k jednotlivým prvkům JSON řetězce a porovnat klíč „content“ na hodnotu „blue“. Pokud hodnota souhlasí, zpráva je duplikována na dvě *change* nodes, vrchní z nich (není na druhém obrázku) zahodí dosavadní zprávu a dál pošle řetězec

{"state": "ON","brightness":255,"color": { "rgb": "0,0,255" } }

který je pomocí node *mqtt out* odeslán do tématu, které přísluší žárovce. Hodnota jasu 255 je maximální, hodnoty RGB udávají intenzitu jednotlivých barev v pořadí červená, zelená, modrá. Druhá *change* node (jak je vidět z obrázku) opět zahodí přijatou zprávu a dál pošle *{“state”:”OFF”}*, tedy příkaz pro vypnutí žárovky. Opět je odeslána do stejného MQTT tématu, pouze s pětisekundovým zpožděním, které způsobuje node *delay*. Test proběhl v pořádku, žárovka na přijaté SMS reaguje podle očekávání.



Obrázek 34 - Nastavení jednotlivých nodes pro testovací sekvenci

Porovnání s alternativami

Brána je primárně určena jako alternativa pro Consumer Smart Home ústředny, zde je porovnána s nejpoužívanějšími z nich po stránce funkcí, pořizovací ceny bezpečnosti a jednoduchosti použití. Hlavní kritéria jsou (zde pro vytvořenou ústřednu NoT Gateway):

- Ovládání a nastavení – ovládání pomocí webové/mobilní aplikace (Home Assistant Lovelace UI), konfigurace pomocí webového administrátorského rozhraní (viz sekce Webové konfigurační rozhraní)
- Automatizace – grafický programovací jazyk Node-RED, definovat lze v podstatě cokoli od jednoduchých podmínek po komplexní algoritmy
- Komunikační rozhraní – Ethernet pro připojení k lokální síti a internetu, Wi-Fi pro koncové prvky (oddělení IP sítě zajišťuje samotná ústředna), Zigbee, Bluetooth, RF 433 Mhz, GSM. Připojení k internetu je nutné jen pro využití externích zdrojů či vzdálenou správu.
- Instalace – složitější, v případě že uživatel má k dispozici hotové desky jednotlivých modulů, je nutné je propojit s Raspberry Pi 3B+, nahrát systémový image na SD kartu a naprogramovat moduly s ESP32
- Rozšiřitelnost – prakticky neomezená, vzhledem k tomu, že všechny komponenty jsou postaveny na otevřených technologiích a části software vytvořené autorem jsou rovněž open-source
- Cena – Raspberry Pi 3B+ 1000 Kč, 2x ESP32 300 Kč, ostatní hardware moduly celkem cca 700 Kč, konstrukce cca 300 Kč. Celkem cca 2300 Kč

Ústředny s hlasovým asistentem

Nejpoužívanější ústředny v oblasti Consumer Smart Home jsou produkty tří z největších technologických firem současnosti – Amazon Alexa, Google Home a Apple Home Kit. Právě tyto produkty tolik popularizovaly koncept chytré domácnosti v posledních letech.

Společným rysem těchto ústředen je princip funkce, kdy fyzické zařízení umístěné v domácnosti (nejčastěji reproduktor s mikrofonem) tvoří pouze uživatelské rozhraní ústředny, veškeré zpracování dat včetně rozpoznávání hlasu je prováděno na serverech příslušné společnosti, které zajišťují vysokou výpočetní kapacitu nezbytnou pro procesy prováděné na pozadí (rozpoznávání a syntéza hlasu, prohledávání internetu, analýza dat aj.). Servery samozřejmě pro optimální využití této kapacity obsluhují více klientů zároveň. Každý hlasový příkaz je tedy zaznamenán, odeslán na server, tam je zpracován a výsledek je odeslán zpět do ústředny. To bylo v minulosti zdrojem jisté kontroverze ohledně narušení soukromí uživatelů a obav z toho, kdy hlasový asistent „poslouchá“ a co vše zaznamenává [21].

Funkce Smart Home ústředny byla těmto zařízením přidána dodatečně, jejich původní účel byl poskytovat informace z veřejného webu, jako zprávy, počasí, hledání pojmů vyhledávačem atd., případně z aplikací, ke kterým uživatel povolil asistentovi přístup, jako např. plánovací kalendář, nákupní seznam aj. Vzhledem k realizaci fyzického zařízení jako reproduktoru s mikrofonem je druhou z původních funkcí přehrávání audia z různých zdrojů. Další funkce už záleží na konkrétním zařízení.

Amazon Alexa

Alexa je nejstarší (na trh uvedena roku 2014) a stále nejoblíbenější ze tří hlasových asistentů používaných jako Smart Home ústředny. Fyzické rozhraní má podobu „chytrých" reproduktorů Amazon Echo, dostupných je několik verzí, hlavní z nich jsou standardní Echo, zmenšený Echo Dot a verze s LCD displayem Echo Show.

- Ovládání a nastavení – hlavní způsob ovládání je hlasem, kromě něj je k dispozici i mobilní aplikace. Konfigurace přes tutéž mobilní aplikaci nebo přes webové rozhraní, které je dostupné odkudkoli z internetu protože nastavení je uloženo v Amazon účtu uživatele.
- Automatizace – řešeny pomocí tzv. rutin (*routines)*, tedy pravidel, u kterých je možno určit spouštěcí událost (*trigger*), podmínku/filtr a spouštěnou akci. Spouštěcí události i akce mohou být určeny časem, lokací, hlasovým příkazem, případně některou z integrací (*skills*, viz níže)
- Komunikační rozhraní – Wi-Fi, u některých modelů (např. Echo Link) také Ethernet, obě rozhraní slouží zároveň k přístupu do domácí sítě/internetu a zároveň k připojení koncových prvků. Modely určené přímo pro domácí automatizaci (např. Echo Plus) disponují také Zigbee. Připojení k internetu je nutné pro veškerou funkcionalitu, je ale možné nastavit několik hlasových příkazů, které budou fungovat i čistě lokálně při výpadku spojení.
- Instalace – velmi jednoduchá, uživateli stačí po zapnutí jednotku Echo registrovat do Amazon účtu a zařízení je připravené k použití
- Rozšiřitelnost – další funkcionalitu je možné doplnit pomocí tzv. dovedností (*skills*). Jejich seznam lze najít na webu výrobce *amazon.com* a jsou mezi nimi integrace pro mnoho Consumer Smart Home ekosystémů, v případě použití Echo Plus lze i prvky některých z nich připojit k ústředně přímo bez použití brány
- Cena – od 1000 Kč (Echo Dot) do cca 5000 Kč (Echo Show)

Google Home

Google Home byl poprvé prezentován v roce 2017 jako alternativa právě pro Amazon Alexa, pro kterou do té doby prakticky neexistovala konkurence. Nabízená řada fyzických zařízení je téměř identická s Amazon Alexa, k dispozici je standardní Google Home, zmenšený Google Nest Mini a verze s displayem Google Home Hub. Pojmy „Google Home" a „Google Nest" mají stejný význam a jsou v různých zdrojích často zaměňovány.

- Ovládání a nastavení – ovládání primárně hlasem, k dispozici je i mobilní aplikace. Konfigurace přes moblní aplikaci nebo webové rozhraní dostupné odkudkoli z internetu v rámci účtu Google.

- Automatizace – stejně jako Alexa má google automatizační pravidla zvaná *routines* fungující na stejném principu *trigger* - podmínka/filtr – akce.
- Komunikační rozhraní – Wi-Fi, která slouží zároveň k přístupu do domácí sítě/internetu a zároveň k připojení koncových prvků. Připojení k internetu je nutné pro veškerou funkcionalitu, při výpadku spojení je ale umožněno ovládání některých koncových prvků pomocí mobilní aplikace.
- Instalace – velmi jednoduchá, uživateli stačí po zapnutí zařízení Google Home/Nest registrovat do Google účtu a zařízení je připravené k použití
- Rozšiřitelnost – funkcionalita se opět přidává pomocí doplňků zvaných *skills* (někdy také *apps*). Počátkem roku 2019 uskutečnil Google akvizici společnosti Nest zabývající se právě produkty pro Consumer Smart Home. Díky tomu má aktuálně větší počet dostupných koncových prvků k integraci než Alexa
- Cena – od 900 Kč (Nest Mini) do cca 3000 Kč (Nest Hub)

Apple Home Kit

Apple šel trochu jinou cestou než Amazon a Google, jejich systém HomeKit nemá konkrétní modelovou řadu fyzických ústředen, ale definuje jednak softwarový framework, jednak komunikační protokol, které jsou součástí všech novějších Apple zařízení (počítač s MacOS, AppleTV...). Jakékoli z těchto zařízení tedy může sloužit jako ústředna, a navíc většinu svých funkcí poskytovat lokálně bez přístupu k internetu. HomeKit byl poprvé prezentován veřejnosti v roce 2014 a jako hlasového asistenta využívá Siri, která byla již dříve nasazena v mobilních telefonech Apple iPhone.

- Ovládání a nastavení – opět ovládání primárně hlasem, dále Apple nabízí mobilní i desktopovou aplikaci *Home* pro ovládání a konfiguraci, ta je ale určena pouze pro iOS, resp. MacOS.
- Automatizace – automatizace se nastavují také pomocí aplikace *Home*, k dispozici jsou podmínky, události, makra, scény a rozsáhlé možnosti mapování mezi nimi
- Komunikační rozhraní – Záleží na hardware, na kterém je ústředna provozována, nejčastěji se využívá AppleTV, která má Ethernetové a Wi-Fi rozhraní, obě slouží zároveň k přístupu do domácí sítě/internetu a k připojení koncových prvků. Připojení k internetu je nutné pouze pro hlasového asistenta Siri, zbytek systému funguje i offline.
- Instalace – velmi jednoduchá, stačí nainstalovat aplikaci Home a pomocí ní zkonfigurovat zařízení, které bude sloužit jako ústředna. Vzhledem k tomu, že systém funguje z velké míry lokálně, musí být zařízení určené jako ústředna nepřetržitě zapnuto (má roli serveru pro zbytek systému), je tedy vhodné určit např. AppleTV nebo reproduktor HomePod, nikoli např. notebook.

- Rozšiřitelnost – i když koncové prvky pro HomeKit nabízí mnoho výrobců, všechny musí mít certifikaci přímo od Apple. To sice omezuje jejich počet, na druhou stranu to ale zajišťuje vysokou míru bezpečnosti (všechny prvky mají z výroby přiděleny unikátní identifikační kód, komunikace probíhá pouze šifrovaně po Wi-Fi apod.). Zařízení v roli ústředny mohou ale být pouze výrobky Apple, navíc veškeré multimediální a aplikační integrace fungují jen s ostatními produkty Apple (iCloud, iTunes, Apple TV atd.), to je ale obecně platné u všech produktů firmy Apple.
- Cena – záleží na zařízení, např. cena AppleTV je kolem 5000 Kč

Samsung SmartThings

SmartThings ekosystém vznikl v roce 2013 a o rok později byl zakoupen společností Samsung. Dnes zahrnuje Consumer Smart Home hub a velké množství vlastních Zigbee koncových prvků, zároveň je kompatibilní s prvky ostatních výrobců. Jako řídící prvek může sloužit buď samostatný hub, nebo soustava zařízení *Samsung Mesh Wi-Fi router*, které kromě hubu plní i funkci domácího Wi-Fi routeru a umožňují systému určitou kontrolu nad IP sítí (např. povolení přístupu k internetu pro děti jen na určitou dobu). Je také možné oddělit LAN síť od Smart Home sítě. Do budoucna má Samsung v plánu přidat do SmartThings vlastního hlasového asistenta Bixby.

- Ovládání a nastavení – mobilní aplikace, zde jsou dvě – původní *Classic Smart Things* a nová *Samsung Connect*, některá nastavení se mezi nimi nepřenáší, je tedy vhodné vybrat si jen jednu. Dále je k dispozici webové rozhraní, které je nutné např. pro instalaci komunitních SmartApps (viz rozšiřitelnost)
- Automatizace – primárně pomocí *routines*, podobně jako ústředny zmíněné výše, lokální a složitější automatizace lze řešit pomocí SmartApps (viz rozšiřitelnost)
- Komunikační rozhraní – Wi-Fi (s možností oddělení sítí a omezení přístupu k internetu v případě použití síťových prvků Samsung), Zigbee, Z-Wave. Přestože některé automatizace lze nastavit lokálně pomocí SmartApps
- Instalace – jednoduchá – samotný hub je připravený k použití hned po zapnutí, stačí nainstalovat příslušnou mobilní aplikaci. Při instalaci Samsung Mesh Wi-Fi je potřeba znalost nastavení počítačové sítě.
- Rozšiřitelnost – funkce lze přidávat pomocí univerzálních pluginů zvaných SmartApps. Ty mohou zajišťovat integraci koncových prvků a jejich platforem (např. Phillips Hue…), automatizace a různé algoritmy (např. pokročilá automatizační platforma *webCoRE*), softwarová rozšíření a mnoho dalšího. Kromě oficiálních SmartApps jsou dostupné i další vyvíjené komunitou.

- Cena – cca 2200 Kč za hub, cca 2500 Kč za router systému Mesh Wi-Fi

Hubitat Elevation

V současné verzi dostupný od roku 2018, Hubitat Elevation vznikl jako alternativa pro SmartThings, která by fungovala kompletně lokálně, což přináší lepší bezpečnost a řádově nižší dobu odezvy automatizací. Systém nenabízí vlastní koncové prvky, k dispozici je pouze samotný hub, integrace Zigbee a Z-Wave prvků třetích stran ale probíhá stejně jako v Home Assistentu automaticky bez nutnosti samostatných bran. Ze zmíněných alternativ je Hubitat Elevation nejblíže bráně NoT Gateway vytvořené v rámci této práce.

- Ovládání a nastavení – Webové rozhraní pro ovládání i konfiguraci. Existuje i mobilní aplikace, ta ale slouží pouze jako wrapper pro webové rozhraní.
- Automatizace – Automatizace jsou řešeny pomocí různých *Apps*. Základní automatizaci na principu *trigger -> podmínka -> akce* poskytuje *Rule Machine*, pro pokročilejší lze stejně jako u SmartThings doinstalovat platformu *webCoRE*.
- Komunikační rozhraní – Ethernet (Wi-Fi koncové prvky je nutné připojit do stejné sítě jako ethernetové rozhraní hubu), Zigbee, Z-Wave. Připojení k internetu je nutné jen pro využití externích zdrojů či vzdálenou správu, jinak celý systém funguje lokálně.
- Instalace – velmi jednoduchá, po připojení k síti a zapnutí je hned dostupné webové rozhraní hubu. Pro ulehčení integrace koncových prvků je vhodné nastavit ústředně pevnou IP adresu.
- Rozšiřitelnost – Podporované koncové prvky Zigbee a Z-Wave lze připojit přímo bez nutnosti dalšího HW či SW. Pro prvky připojené pomocí IP sítě a další integrace jsou k dispozici plug-iny zvané jednoduše *Apps*. Podobně jako u SmartThings existují jak oficiální, tak komunitní *Apps*.
- Cena – cca 3200 Kč

Profesionální systémy

Zde už se nejedná o Consumer Smart Home, i tak ale porovnání s některým z profesionálních produktů může být zajímavé, hlavně z hlediska poměru cena/funkce.

Loxone

Aktuálně v ČR nejoblíbenější řešení Smart Home založené na profesionální inteligentní elektroinstalaci. Systém se soustředí hlavně na osvětlení, řízení prostředí a šetření energií. Nedávno byl uveden na trh také Loxone Audio Server, který doplňuje multimediální funkce. Systém je realizován jako řada elektroinstalačních pří-

strojů, hlavní řídící jednotka nazvaná *miniserver* je společně s rozšiřujícími moduly určena k montáži na DIN lištu do rozvaděče. Ostatní přístroje poté nahrazují prvky klasické elektroinstalace (spínače, senzory, svítidla aj.) a připojují se proprietárními sběrnicemi.

- Ovládání a nastavení – Ovládání je možné pomocí mobilní aplikace nebo webového rozhraní, které běží lokálně na miniserveru. Konfiguraci provádí instalační technik (viz instalace) pomocí specializované aplikace pro PC, podobně jako je tomu např. u systému KNX.
- Automatizace – Software obsahuje mnoho předpřipravených automatizačních algoritmů, které lze v nastavení jednoduše aplikovat na danou oblast domácnosti (např. řízení žaluzií podle kombinovaných požadavků na osvětlení a vytápění). Mimo to se systém učí vzory chování uživatele, kterým pak algoritmy přizpůsobuje. Lze také ručně definovat jednoduchá pravidla, makra a scény.
- Komunikační rozhraní – Ethernet pro připojení do lokální sítě. Proprietární sběrnice pro komunikaci miniserveru s rozšiřujícími moduly *Loxone Link* (drátová) a pro připojení elektroinstalačních přístrojů - *Loxone Tree* (drátová) a *Loxone Air* (bezdrátová). Celý systém funguje lokálně, vzdálená správa je možná pomocí VPN (Virtual Private Network) hostované na minserveru.
- Instalace – odborná, systém musí být instalován elektrikářem či firmou s licencí od výrobce (Loxone Partner). Záruční a pozáruční servis celé instalace je pak na smlouvě mezi Loxone Partnerem a zákazníkem, výrobce zajišťuje pouze záruku na jednotlivé komponenty.
- Rozšiřitelnost – velmi malá, jedná se o uzavřený ekosystém určený pouze pro Loxone prvky. Lze ale integrovat některé populární technologie, jako např. hlasového asistenta Amazon Alexa.
- Cena – Miniserver cca 14000 Kč, rozšiřující prvky podle funkce a složitosti od cca 1000 Kč (Loxone Smart Switch) do cca 14000 Kč (Loxone Audioserver).

Závěr

Výsledkem práce je funkční zařízení realizující komunikační bránu pro Smart Home s funkcemi ústředny a podporující technologie Wi-Fi, Zigbee, Bluetooth, RF 433 Mhz a GSM. Ovládání i konfigurace jsou umožněny pomocí webového rozhraní. Systém Home Assistant je přímo integrován, se systémem OpenHAB je zařízení kompatibilní. Vestavěny jsou senzory teploty, vlhkosti, osvětlení a koncentrace CO2, navíc lze monitorovat teplotu a vlhkost v dalších dvou místnostech pomocí odpojitelných modulů. Zadání bylo tedy podle mého názoru splněno.

Co se týče praktické aplikace, má vytvořené zařízení stále blíže k systémům kategorie DIY než k produktům Consumer Smart Home. Z porovnávaných alternativ je nejvíce podobné ústředně Hubitat Elevation, jeho instalace je ale náročnější a vyžaduje od uživatele základní znalosti v oblasti podobných systémů. Na druhou stranu si díky svému open-source základu zařízení zachovává obrovskou univerzalitu a rozšiřitelnost, která u Consumer Smart Home ústředen nemá obdoby.

Zařízení je využitelné jako základ pro menší Smart Home instalaci (např. byt), nebo i jako úvod do pokročilejších možností domácí automatizace pro uživatele, kterým přestává stačit některá z dosud používaných Consumer Smart Home ústředen. Pro instalace většího rozsahu by pravděpodobně výkon použitého SBC nebyl dostatečný.

Pro případný další vývoj zařízení se nabízí několik vhodných rozšíření, například integrace firewallu pro lepší kontrolu síťového provozu či integrace programátoru modulů ESP32, pro zjednodušení procesu instalace. Užitečné by bylo i nahradit senzor koncentrace CO_2 MQ-135, odpadla by tak nutnost kalibrace každého kusu, v dané cenové hladině ale neexistuje žádná alternativa.

Literatura

[1] Deschamps-Sonsino Alexandra: *Smarter Homes: How Technology Will Change Your Home Life (Design Thinking)*, 2018, ISBN 9781484233627

[2] Peter Bastian: *Praktická elektrotechnika*, EUROPA-SOBOTÁLES 2004, ISBN 80-86706-07-9

[3] *Historie spotřební elektroniky*, [online], 2011, dostupné z: https://www.tvfreak.cz/historie-spotrebni-elektroniky-1-dil-jaky-byl-vyvoj/4271

[4] Iftikhar Alam, Shah Khusro, Muhammad Naeem: *A review of smart TV: Past, present, and future*, IEEE 2017, ISBN 978-1-5386-1658-1

[5] *Internet Usage Worldwide,* [online], 2020, dostupné z https://www.statista.com/topics/1145/internet-usage-worldwide

[6] Aleš Brotánek; Klára Brotánková: *Jak se žije v nízkoenergetických a pasivních domech,* 2012, ISBN 978-80-247-3969-4

[7] Cech EPS České republiky, [online], 2020, dostupné z http://cecheps.cz

[8] W*hat is the Internet of Things,* [online], 2020, dostupné z: https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/

[9] *Serial Programming/Modems and AT Commands*, [online], 2020, dostupné z: https://en.wikibooks.org/wiki/Serial_Programming/Modems_and_AT_Commands

[10] *Comparison of BLE and Zigbee power consumption*, [online], 2013, dostupné z: https://www.microsoft.com/en-us/research/publication/power-consumption-analysis-of-bluetooth-low-energy-zigbee-and-ant-sensor-nodes-in-a-cyclic-sleep-scenario/

[11] Robert Chin, *A DIY Smart Home Guide: Tools for Automating Your Home Monitoring and Security Using Arduino, ESP8266, and Android*, 2020, ISBN 9781260456134

[12] *Raspberry Pi official forum, Bluetooth issues*, [online], 2020, dostupné z: https://www.raspberrypi.org/forums/viewtopic.php?t=243601

[13] *Raspberry Pi pinout and bus documentation*, [online], 2020, dostupné z: https://pinout.xyz

[14] Udo Brandes: *Mach's einfach: Erste Schritte mit Smart-Home-Programmierung: Einstieg in die Hausautomation mit Node-RED*, 2019, ISBN 978-3645606516

[15] – Turner Ryan: *Intermediate Guide to Learn Arduino Programming Step by Step*, 2019, ISBN 978-1647710194

[16] Evi Nemeth, Garth Snyder: *UNIX and Linux System Administration Handbook,* 2018, ISBN 978-0134277554

[17] John R. Patrick: *Home Attitude: Everything You Need to Know to Make Your Home Smart*, CreateSpace Independent Publishing Platform 2017, ISBN 978-1542710190
[18] Vlastimil Lhotecký: *Univerzální ústředna pro Smart Home,* 2019, semestrální práce v rámci předmětu Projekt 2 programu Inteligentní Budovy na FEL ČVUT, dostupné v příloze práce
[19] Kubátová, Hana: *Struktura a architektura počítačů s řešenými příklady*, skripta FIT ČVUT 2018, ISBN 978-80-01-06410-8
[20] Morty Eisen, *Introduction to PoE and the IEEE802.3af and 802.3at Standards*, [online], IEEE 2009, dostupné z https://www.ieee.li/pdf/viewgraphs/introduction_to_poe_802.3af_802.3at.pdf
[21] Geoffrey Fowler: *Alexa has been eavesdropping on you this whole time*, [online], Washington Post 2019, dostupné z https://www.washingtonpost.com/technology/2019/05/06/alexa-has-been-eavesdropping-you-this-whole-time/

Slovníček pojmů a zkratek

ACS – Access Control Systém, systém řízení přístupu do budovy

API – Application Programming Interface, definuje způsob, jakým daná softwarová aplikace může komunikovat s jinými.

ASCII – standard kódování znaků

CCTV – Closed Circuit TeleVision – kamerové zabezpečovací/dohledové systémy

CPU – Central Processing Unit, procesor

DAC – Digital to Analog Converter, digitálně-analogový převodník

ADC – Analog to Digital Converter, analogově-digitální převodník

DHCP – Dynamic Host Configuration Protocol, protokol pro automatickou konfiguraci síťových uzlů

DIY – Do It Yourself, „udělej si sám“, návrh a konstrukce zařízení svépomocí

Docker – software pro kontejnerizaci, což je forma virtualizace, kdy hostované aplikace (kontejnery) sdílí kromě hardware i jádro hostitelského operačního systému

EPS – Elektronický Protipožární Systém

EZS – Elektronický Zabezpečovací Systém

GPIO – General-Purpose Input/Output, univerzální rozhraní mikrokontrolerů

GPU – Graphics Processing Unit, grafický procesor

GUI – Graphical User Interface, grafické uživatelské rozhraní

HTPC – Home Theatre PC, počítač určen k funkci multimediálního centra

HVAC – Heating, Ventilation, Air Conditioning, technologie pro vytápění, větrání a klimatizaci

IDE – Integrated Development Environment – prostředí pro vývoj, ladění a spouštění programů v určitém programovacím jazyce

IO – Integrovaný obvod

IoT – Internet of Things, internet věcí – koncept, kdy elektronická zařízení komunikují mezi sebou prostřednictvím internetu (potažmo jakékoli datové sítě) bez interakce člověka

IT – Informační technologie

MCU – MicroController Unit, mikroprocesor s pamětí a periferiemi v jednom čipu

NAS – Network-Attached Storage, sdílené úložiště dat dostupné přes síť z více míst zároveň

OEM – Original Equipment Manufacturer

OS – Operační Systém

PCB – Printed Circuit Board, deska plošných spojů

PIR – Passive InfraRed, druh technologie senzorů pohybu

TCP/IP – Sada protokolů pro komunikaci v počítačových sítích

UART – Univerzální asynchronní sériové rozhraní

UI – User Interface, uživatelské rozhraní

UPnP – Universal Plug and Play – protokol, který umožňuje uzlům sítě měnit nastavení routeru (např. otevření některého TCP portu)

USB OTG – USB On the GO, zařízení se může chovat jako „host“ i jako „device“

VPN – Virtual Private Network – technologie pro vytvoření šifrovaného tunelu skrz internet, tunel se vzhledem k zařízením, která ho využívají chová jako lokální síť

Obsah přiloženého CD

Apache – konfigurační soubor pro Apache2 webserver a reverzní proxy server
Design – schémata a rozložení desek plošných spojů pro software Autodesk Eagle
GSM – skript a systemd unit file zajišťující SMS funkcionalitu, knihovna pro obsluhu modemu SIM800L upravená pro potřeby brány
Image – obraz disku pro nasazení software brány na Raspberry Pi 3B+
OMG – Konfigurační soubory *User_config.h* pro moduly s ESP32
P2 – Semestrální práce z předmětu Projekt 2 – pramen [18]
Sensors – skripty a systemd unit files pro obsluhu senzorů
SysDWeb – zdrojové kódy projektu SysDWeb upravené pro potřeby brány
Webroot – obsah kořene webového rozhraní
Z2M_Admin – definice Node-RED flow pro grafické ovládací rozhraní Zigbee2MQTT