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.

14. srpna 2020

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce Ing. Vladimíru Janíčkovi, Ph.D. trpělivost a cenné rady, které mi v průběhu zpracování diplomové práce poskytl.

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 14. srpna 2020 .....

Odkaz na tuto práci

Lhotecký, Vlastimil. *Komunikační brána pro Smart Home*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta elektrotechnická, 2020.

Anotace

V této práci se zabývám návrhem a realizací univerzální komunikační brány, určené primárně pro otevřené Smart Home ústředny jako je OpenHAB, Home Assistant nebo Node-RED, která umožňuje bezdrátově komunikovat s koncovými zařízeními využívajícími technologie Bluetooth, Zigbee nebo Wi-Fi.

Brána samotná obsahuje několik senzorů pro měření nejčastěji požadovaných parametrů prostředí.

Pro komunikaci s ústřednou je použito Ethernetové rozhraní, které zároveň zajišťuje napájení brány. Přenos dat je realizován protokolem MQTT a konfigurace pomocí webového rozhraní.

Během práce jsem zhodnotil nejvhodnější komponenty, navrhl, realizoval a otestoval zařízení po HW i SW stránce a porovnal výsledek s případnými alternativami dostupnými na trhu.

Klíčová slova: chytrá domácnost, domovní automatizace, internet věcí, otevřené technologie, komunikační brána, MQTT, Wi-Fi, Zigbee, Bluetooth, Home Assistant, Node-RED, OpenHAB, Raspberry Pi

Abstract

The aim of this thesis was to design and create a universal communication gateway, mainly intended for open Smart Home hubs like OpenHAB, Home Assistant or Node-Red, that would allow wireless communication with end devices using Bluetooth, Zigbee or Wi-Fi technology.

The gateway itself includes a few sensors for measuring the most commonly requested environmental parameters.

An Ethernet interface is used to communicate with a Smart Home hub 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.

I've gone through various available components and selected the most suitable ones for the project, then used them to design and implement the gateway from both a 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, MQTT, Wi-Fi, Zigbee, Bluetooth, Home Assistant, Node-RED, OpenHAB, Raspberry Pi

Obsah

Seznam obrázků

Všechny použité obrázky byly buď označeny jako dostupné k volnému užití nebo mnou vytvořené.

Úvod

Současná situace na trhu Smart Home

Koncept inteligentní domácnosti pomalu přestává být jen doménou milionářů a technologických nadšenců. Mnoho výrobců elektroniky během posledních let, hlavně v návaznosti na popularizaci konceptů IoT a rozvoj cloudových služeb, uvedlo na trh celé ekosystémy zařízení realizujících jejich vizi propojené (connected home), či chytré (smart home) domácnosti. Tato zařízení jsou mnohdy založena na univerzálních čipech (např. rodina mikrokontrolerů ESP, viz níže) a rozvněž univerzálních a běžně rozšířených, zpravidla bezdrátových komunikačních standardech (Wi-Fi, Zigbee, BLE...) a díky tomu se pohybují na mnohem nižších cenových hladinách než ucelené, profesionální systémy.

Dosud měl zájemce o chytrou domácnost v podstatě dvě možnosti: Řešení v podobě profesionálního systému na míru danému objektu, včetně značné finanční investice, která se k těmto systémům váže, nebo systémy vytvořené svépomocí, u kterých je zase problém neexistující záruka funkce a tím pádem vysoké nároky na uživatele, který v podstatě musí být buď profesionálem v oboru, nebo mít svůj chytrý příbytek jako velký koníček.

Díky výše zmíněným aktivitám výrobců elektroniky se teď naskýtá třetí možnost – smart home systém jako „off the shelf" produkt, který si může kdokoli koupit v obchodě s elektronikou, podobně jako např. novou televizi. Kromě relativně nízké ceny má system rovněž nízkou náročnost konfigurace. Ta je zhruba na úrovni zmíněné nové televize. I tato možnost má ale samozřejmě své problémy. Tyto jsou blíže rozebrané v sekci Koncept Smart Home, hlavně se ale jedná o spolupráci s okolím a systémy jiných výrobců, nutnost přístupu k internetu, automatizace a konfigurace komplexnějších úloh a nedostatečná bezpečnost.

I přes své nevýhody si myslím, že toto je ta správná cesta vpřed, protože profesionální systémy nezvládají držet krok s extrémně rychlým vývojem technologií v této oblasti a i v současnosti montované verze systémů jako např. Control 4 nebo Loxone jsou už v některých ohledech zastaralé [1]. Pokud by se výrobci těchto "off the shelf" zařízení více zaměřili na kvalitu jednotlivých výrobků, upustili od izolovanosti svých ekosystémů a místo toho začali využívat nějaký více standardizovaný způsob komunikace, mohl by tento způsob provedení chytrých domácností úspěšně konkurovat profesionálním instalacím na míru, nebo je dokonce, za předpokladu existence firem poskytujících servis pro tyto nové systémy, kompletně nahradit.

V současnosti je ale nutné pro integraci Smart Home zařízení různých výrobců využít buď produkty třetích stran (např. některý z hlasových asistentů

jako je Amazon Alexa nebo Google Home), anebo některý z otevřených projektů. Jedním z těchto projektů je i brána vyrvořená v rámci této práce.

Účel brány

Hlavním cílem brány je sloužit jako univerzální propojovací prvek mezi Smart Home ústřednou a koncovými prvky jako jsou nejrůznější senzory a akční členy, "chytré" domácí spotřebiče, multimediální zařízení atd. Důraz je kladen na kompatibilitu s co největším počtem zařízení bez ohledu na výrobce a na využití otevřených technologií. Brána tak ve spolupráci s ústřednou umožňuje jakkoli kombinovat a integrovat prvky různých Smart Home ekosystémů a uživatel si může z každého systému vybrat jen to, co mu vyhovuje.

Důležitá je také snadná instalace a nastavení brány, která by z pohledu uživatele měla být nejvýše tak náročná, jako u samotné ústředny na kterou bude brána napojena. To znamená návrh HW části jako modulu k existující platformě (použit SBC RaspberryPi Zero) a SW části jako obrazu disku, případně s možností instalačního skriptu nebo balíčku.

Samotná brána již integruje několik senzorů běžných pro tuto oblast, konkrétně senzory teploty, vlhkosti vzduchu, obsahu CO_2 a intenzity osvětlení. Spolu s ústřednou tak může tvořit jeden ze dvou základních kamenů chytré domácnosti, ke kterým už stačí jen doplnit koncová zařízení.

Brána slouží pouze jako univerzální komunikační rozhranní k již existující ústředně, ne jako ústředna samotná.

Rešerše

Vznik a vývoj konceptu Smart Home

Historie

Smart Home je dnes velmi rozšířený a zároveň velmi vágní pojem. Pro hlubší pochopení, co vlastně pojem vyjadřuje a jaký je vůbec důvod dělat domácnosti „chytré“ je třeba se podívat do historie na vývoj průměrného rodinného domu v České republice.

Klasická elektroinstalace

V době dokončení kompletní elektrifikace České republiky kolem roku 1950[2], využívaly domácnosti nově zavedenou energii hlavně na svícení. Dům, většinou s jednofázovou přípojkou měl typicky jeden světelný a jeden zásuvkový okruh, přičemž svítidla se ovládala jednoduchými mechanickými spínači. V případně třífázové přípojky mohl být navíc okruh pro třífázovou zásuvku na různé domácí stroje (např. okružní pila), a/nebo okruh pro domácí vodárnu pro čerpání vody ze studny.

Vytápění bylo realizováno jednoduchými kotly na tuhá paliva.

Audio a video

Koncem osmdesátých let [2] se už do většiny domácností rozšířila první podoba multimédií. Rádiové vysílání se ustálilo v pásmu VKV, které zůstává aktivně využívané i v současnosti. K němu se přidalo televizní vysílání ve standardu PAL, jehož velký počet barev a rozlišení 576i (576 řádků, ne pixelů na výšku, jelikož se jedná o analogovou technologii) bylo pro televizní přijímače v té době nejvyšší dostupné.

Rádiové a televizní přijímače byly prvními exempláři tzv. spotřební elektroniky. [3] Časem se k nim přidaly satelitní přijímače a přehrávače magnetických médií (audio média přešla od gramofonových desek, přes magnetofonové pásky až po známé audiokazety MC a u videa to byly, kromě několika proprietárních formátů pro domácí videokamery, kazety VHS).

Zařízení tohoto typu dodnes tvoří základ domácích multimediálních center.

Digitalizace

Digitální elektronika existovala již od sedmdesátých let, kdy se začaly objevovat první mikroprocesory. V oblasti spotřební elektroniky u nás ale významné změny

začaly až po přelomu století, stejně jako v dalších oblastech domácích technologií popsaných níže.

Audio a video formáty přešly na optická média s digitálním záznamem, pro zvukové záznamy to bylo Audio CD a pro obrazové záznamy DVD. Tyto zůstávají dodnes společně s Blu-ray disky standardem pro fyzickou distribuci multimédií. Později se objevují i různé komprimované formáty, z nichž nejznámější je dnes stále používaný audio formát MP3, který odstartoval vlnu populárních MP3 přehrávačů jako např. Apple iPod. Celá řada formátů vznikla samozřejmě i pro video. Komprimace umožňovala efektivnější využití datových nosičů a také šetření přenosové kapacity internetového připojení, která j byla v porovnání s velikostí multimediálních dat velmi malá.

Multimediální centrum už je pomalu v podobě, v jaké ho najdeme v současných domácnostech. LCD nebo plazmová televize, integrovaný hi-fi systém pro přehrávání CD, MC kazet a příjem FM rádia, DVD přehrávač, starší VHS přehrávač, případně i herní konzole. Začíná se objevovat problém s integrací zařízení a příliš mnoha (dálkovými) ovladači, je možné ho částečně eliminovat pomocí AV receiveru, který výstupy více zařízení propojuje s televizorem a reprosoustavou a automaticky je přepíná.

PC a internet

Přibližně ve stejné době se do domácností rozšiřují osobní počítače. Do internetu byla Česká republika připojena už v roce 1992, takže v této době (cca 2000-2005) už má jakákoli domácnost s telefonní přípojkou možnost vytáčeného připojení k internetu s rychlostmi až 56kbps. [4]

Osobní PC mnoha lidem nejen ulehčují práci, ale poskytují i nové možnosti pro zábavu a volný čas. Internet se stává alternativním distribučním kanálem pro audio a video média, alternativním zdrojem informací jako je např. zpravodajství a výukové materiály a alternativním způsobem komunikace, ať už v textové formě v podobě diskuzních fór a chatu, nebo v hlasové formě v podobě programů jako Skype nebo TeamSpeak. Začínají se objevovat i první online videohry. Jediným omezením, kvůli kterému zatím internet není primárním informačním kanálem jak ho známe dnes, je rychlost a odezva připojení, například stažení jednoho hudebního souboru (nahrávka dlouhá cca 4 minuty), který i po komprimaci v MP3 formátu má velikost kolem 5MB, trvá po vytáčeném připojení více než 10 minut.

Díky své univerzálnosti ale PC v některých domácnostech přebírají místo potřební elektroniky a v podobě tzv. HTPC nahrazují CD a DVD přehrávač, herní konzoli, rádio, receiver a další prvky multimediálního centra. Oproti předchozí sekci je tedy technologie koncentrována do dvou míst. Objevují se další problémy, např. jak tyto dvě centra propojit, pokud si chceme např. pustit film uložený v PC na TV v obýváku, případně nahrávat televizní vysílání na PC, v případě HTPC pak opět neintuitivní ovládání, které je realizováno různě, od

mapování univerzálních infračervených ovladačů, přes bezdrátové klávesnice s integrovaným touchpadem, po více či méně funkční dedikovaná zařízení, z nichž asi nejznámější je asi Logitech Harmony.

Automatizace

I když automatizace existovala v určité formě už dříve (např. mechanické sekvencery známé z prvních automatických praček nebo jukeboxů), až digitální elektronika a zejména mikroprocesory odstartovaly její masovou aplikaci.

Pro domácnosti to má jak nepřímé, tak i přímé důsledky. Těmi nepřímými je, že díky rostoucí automatizaci průmyslové výroby bylo možné výrazně snížit nároky na lidskou práci a přesunout výrobu do oblastí s levnější pracovní silou. To, společně s pokrokem v dalších oblastech, jako je vznik univerzálních integrovaných obvodů (např. audio zesilovače), mikrokontrolerů a FPGA výrazně urychlilo, a hlavně zlevnilo vývoj a produkci. Díky tomu se technologie, které byly dosud k vidění hlavně ve veřejných a průmyslových budovách (automatické kotle, klimatizace, tepelná čerpadla aj., viz níže) začínají rozšiřovat i do běžných rodinných domů nebo dokonce bytů [5].

Přímým důsledkem je automatizace samotné domácnosti. Příkladem může být složitější řízení vytápění kdy lze regulovat teplotu jednotlivých místností samostatně, přepínání zdrojů tepla podle aktuální potřeby a dostupnosti (např. mezi solárními kolektory, tepelným čerpadlem a kotlem), samočinné nastavení venkovních rolet a žaluzií na základě kýžených světelných podmínek v interiéru a v neposlední řadě první inteligentní elektroinstalace. Ty byly založeny na využití průmyslového PLC jako řídící jednotky a ovládacích sběrnic jako Modbus nebo DALI, pomocí kterých byly dosud statické prvky elektroinstalace (vypínače, svítidla...) připojeny k PLC jako senzory a akční členy, a tudíž je šlo dynamicky mapovat, vytvářet automatizační pravidla, scény atd. Pro změnu nastavení bylo potřeba přeprogramovat PLC, případně u pokročilejších systémů bylo k dispozici webové rozhraní. Tyto prvky přímé automatizace byly ale v domácnostech používány jen zřídka, kvůli složitosti a ceně.

HVAC

Jak už jsem zmínil v předchozí sekci, vývoj samozřejmě probíhal i u technologií pro úpravu prostředí. Místo jednoduchých kotlů na tuhá paliva máme automatické zplynovací kotle na uhlí, a různé podoby dřeva a kondenzační plynové kotle s vysokou účinností (přes 100 % díky využití kondenzačního tepla). Využívá se obnovitelných zdrojů tepla, pomocí tepelných čerpadel a solárních kolektorů.

Díky novým technologiím ve stavebnictví se zároveň čím dál tím více snižují tepelné ztráty budov v zimním období a pro novostavby a budovy po rekon-

strukci jsou potřeba výrazně menší tepelné výkony než dříve. Problémem ale začínají být příliš velké tepelné zisky v letním období, tím pádem se častěji kromě vytápění instalují i klimatizační jednotky. [6]

Objevují se také systémy vzduchotechniky, které zároveň zajišťují vytápění i chlazení, dále navíc větrání s rekuperací tepla a čištění vzduchu. Často také umožňují výše zmíněnou regulaci teploty v každé místnosti zvlášť nebo další pokročilé funkce jako vytápění jedné místnosti odpadním teplem z jiných atd.

Tyto systémy umožnují významné úspory energie, je u nich ale potřeba složitější řízení a nastavení, hodilo by se i sbírat data z jejich provozu a na základě jejich vyhodnocení toto nastavení upravovat. Většina lidí ale tato zařízení ponechá ve stavu, v jakém jsou po instalaci, maximálně mění nastavení teploty na hlavním termostatu. Tím se tito lidé připravují o další možné úspory energie. Toto bohužel platí u mnoha technických inovací v domácnostech, čím složitější systém, tím menší zájem uživatelů a porozumění jeho funkci.

Zabezpečení

Další z technologií, které se do domácností rozrostly z veřejných objektů (viz sekce automatizace) jsou různé zabezpečovací systémy. Jedná se o zabezpečení před zloději (EZS a kamerové systémy), zabezpečení proti požáru (EPS), výjimečně i se samočinným hasicím zařízením, případně elektronické řízení přístupu (ACS).

Tyto systémy jsou ve většině případů samostatné celky bez jakékoli interakce s okolím, vyjma jejich ovládacích prvků.

Současnost

Jak je vidět z předchozí kapitoly, během let (hlavně po roce 2007 se vývoj razantně zrychlil) se v průměrné domácnosti naakumulovalo množství technologií, které si po první instalaci víceméně „žijí vlastním životem“ a jsou, zpravidla s výjimkou přehrávání multimédií (viz níže), používány nezávisle na sobě.

Multimediální centrum zůstává dominantou obývacího pokoje a je postaveno kolem „Smart TV“, což je televizor s operačním systémem a internetovou konektivitou, který kromě klasického televizního a satelitního vysílání (které přešlo z analogového na digitální) zvládá přehrávat streamovaná média z internetu od nepřeberného množství poskytovatelů, případně uložená na lokálním pevném disku připojeném k TV přímo nebo přes místní síť LAN. Smart TV svými funkcemi prakticky nahrazuje kombinaci klasické TV a HTPC. Dalšími funkcemi bývá nahrávání a různé možnosti komunikace s mobilními zařízeními. TV může být doplněna reprosoustavu (samotná TV často umožňuje vícekanálový zvuk), doplňky pro komunikaci s okolím (typicky lokální streaming a ovládání pomocí smartphone) jako je Google Chromecast nebo Apple TV, případně o přehrávač

optických disků (které stále více ustupují, hlavním distribučním kanálem pro multimédia je internet).

Pevné, desktopové PC také ustupují a jsou využívány zpravidla jen jako herní stroje, případně pro práci náročnou na výkon (např. úprava videa). Nahrazovány jsou notebooky a dalšími mobilními zařízeními. Za tyto změny může kromě vývoje mobilních zařízení i vývoj připojení k internetu, které nyní využívá tři hlavní technologie – DSL, Wi-Fi a kabelové/optické připojení – s rychlostmi v desítkách až stovkách Mbps. Připojení je v domácnosti zakončeno typicky tzv. all-in-one routerem, který zároveň slouží jako modem, router, ethernetový switch a Wi-Fi přístupový bod (AP), ke kterému se připojují zmíněná mobilní zařízení. TV bývá, jako statické zařízení, připojena kabelem, zejména pokud poskytovatel internetu zároveň poskytuje digitální televizi.

Vývoj HW a SW pro embedded systémy se dále zjednodušuje, zrychluje a zlevňuje, ať už jde o metodiku, technologie nebo nové IO a programovací jazyky. Výrobci díky tomu začínají vybavovat různými „chytrými" funkcemi i domácí spotřebiče, což je více či méně užitečné, v závislosti na daném spotřebiči, např. u robotického vysavače nebo pračky se tyto funkce hodí, u mikrovlnné trouby nebo rychlovarné konvice už tolik ne. Objevují se i v úvodu zmíněné off-the-shelf výrobky pro chytrou elektroinstalaci. Tyto výrobky a spotřebiče využívají nové standardy bezdrátové komunikace jako je Zigbee, Z-Wave nebo BLE pro komunikaci mezi sebou, nebo se smartphonem uživatele.

Dalšími technologiemi jsou již zmíněné zabezpečovací a HVAC systémy a dale např. zavlažovací systémy, fotovoltaické elektrárny, bazénové a saunové technologie, rolety, žaluzie, okna, dveře, meteostanice, VoIP telefony a intercom, měření spotřeby energií, automobily a další.

Connected Home

I když mnoho investorů u současných novostaveb i rekonstrukcí stále používá klasický přístup, s rostoucím počtem technologií v domácnostech má smysl navrhovat budovy jako „systém systémů", kdy spolu jednotlivé subsystémy spolupracují, a jejich propojením získáme něco více než jen shluk technologií ovládaných z jednoho místa, získáme integrovaný systém, jehož komponenty se navzájem doplňují. To je filozofie tzv. Connected Home.

Samotný název Connected Home (volně přeložen jako „propojená domácnost") dobře vystihuje podstatu tohoto konceptu. Systém vychází z inteligentní elektroinstalace popsané v sekci *automatizace*, ale spíše než jako průmyslový řídící systém se chová jako počítačová síť. Centrální prvek se nazývá hub (ústředna) a s koncovými prvky komunikuje pomocí různých sběrnic (viz níže), samotné koncové prvky nejsou jen holé senzory či akční členy, ale většinou fungují autonomně, tedy při odpojení od ústředny dojde jen ke ztrátě komunikace s ostatními prvky a koncová zařízení dále samostatně plní svou funkci. Za koncové zařízení lze považovat v podstatě cokoli, co má kompatibilní komunikační

rozhraní, od jednoduché žárovky, přes smart TV až po celou jednotku vzduchotechniky a v závislosti na míře integrace např. zmíněnou VZT jednotku z ústředny ovládat buď jako celek, nebo i jako jednotlivé komponenty – čidla teploty, ventilátory atd. Systém je tedy velice univerzální a omezen právě jen možnostmi integrace jednotlivých zařízení.

Tento přístup umožňuje ovládání celé budovy z jednoho místa, což může být speciální terminál, hlasový asistent, běžné domácí PC, smartphone či jakákoli jejich kombinace. Dále je možné softwarově definovat mapování koncových prvků, např. není problém změnit funkci spínače/ovladače ze stmívače pro žárovku na ovládání hlasitosti, bez nutnosti zásahu do zapojení, také lze jednoduše upravovat některá automatizační pravidla bez znalosti programování. Propojenost zařízení do určité míry eliminuje redundantní prvky, příkladem může být PIR senzor na chodbě sloužící jako spínač osvětlení a zároveň jako senzor EZS v případě, že nikdo není doma.

Kromě toho, že tato centralizace dává uživateli výborný přehled a kontrolu nad všemi systémy v budově, umožňuje také sběr a vyhodnocování provozních dat, ať už o systému samotném, nebo např. spotřebě energií, parametrech prostředí etc. Někdy je jako součást Connected Home brána i domácí počítačová síť, pak lze např. kontrolovat přístup k internetu pro různá zařízení – omezit přístup k internetu pro děti, přístup k datům v lokální síti pro návštěvy a podobně.

Smart Home

Connected Home tedy poskytuje obrovské množství možností pro ovládání domácnosti, spolupráci jednotlivých systémů, sběr dat, atd. To ale nemusí být vždy výhoda, pro některé uživatele jsou obsluha a nastavení systému příliš komplikované a tak ho nechávají po celou dobu provozu ve stavu, v jakém byl po instalaci, případně po několika měsících zkoumání provozních dat a nastavování zjistí, že je to příliš časově náročný proces a přestanou některé funkce používat.

Smart Home v jeho originálním významu vychází z myšlenky, že by systém měl uživateli co nejvíce šetřit čas a práci, ne mu práci naopak přidělávat složitým nastavováním automatizace. Uživatel tak v podstatě nepotřebuje ani vědět jak systém funguje a může ho brát jen jako nástroj k ulehčení svého života. Zároveň si ale uživatel bude moci užívat všechny přednosti Connected Home. Toho je docíleno pokročilou automatizací s využitím strojového učení a umělé inteligence. Systém se z nasbíraných provozních dat učí vzory chování uživatele, např. kdy je doma, kdy větrá otevřenými okny, nastavení termostatu v průběhu roku atd. a podle těchto vzorů pak řídí např. topení a klimatizaci s cílem maximálního komfortu uživatele a úspory energie. Typický příklad je prediktivní spínání vytápění, kdy při odchodu uživatele z budovy systém přestane topit a před odhadovaným časem příchodu opět topit začne, aby v budově v momentě příchodu uživatele byla požadovaná teplota. Systém může i simulovat přítomnost uživatele v budově

v době, kdy je na dovolené. Kromě chování uživatele systém typicky reaguje i na naměřenou spotřebu energií, externí zdroje dat jako předpověď počasí a další, příkladů je opravu mnoho.

Někteří uživatelé ale naopak chtějí mít možnost kontroly, manuálního nastavení a vidět jak systém funguje „pod kapotou“, je tedy potřeba najít rovnováhu, některé systémy totiž vždy prioritizují rozhodnutí AI (např. při řízení prostředí v budově) před vstupem uživatele (např. z důvodu šetření energií), ten má pak dojem, že ho systém „neposlouchá“, což bere jako projev závady.

Systémy a produkty, kterými se v této práci zabývám, se nacházejí na spektru mezi Connected a Smart Home. V dalším textu ale budu tyto výrobky a technologie souhrnně označovat jako „Smart Home“, jelikož je tento pojem (hlavně díky marketingu výrobců) v široké veřejnosti již zažitý pro jakákoli zařízení tohoto typu, spolu s pojmem „IoT“, jehož použití se ale kvůli nejasné definici snažím pokud možno vyhnout (viz níže).

Struktura

Jak již bylo zmíněno, srdcem systému je ústředna (nejčastěji nazývaná anglickým výrazem hub). V praxi to může být dedikované elektronické zařízení nebo skupina zařízení postavených na nějakém mikrokontroleru a pracujících na stejném principu jako PLC, pouze s jiným způsobem programování a jinými periferiemi. Periferie a komunikační rozhranní velmi závisí na typu systému (viz níže) a výrobci.

Hub je doplněn o hlavní ovládací rozhraní. To je typicky jednoúčelový terminál s dotykovým displayem, webová aplikace, nebo aplikace pro smartphone. Pomocí něj je možné celý systém ovládat a většinou i konfigurovat (některé systémy vyžadují konfiguraci technikem).

Zbytek systému jsou koncová zařízení. Ty spadají do následujících kategorií:

- Osvětlení – spínání a stmívání jednotlivých svítidel, spojování svítidel do skupin a vytváření světelných scén, mood lighting (snaha navodit určitou náladu pomocí teploty a barvy světla), různé algoritmy jako např. cirkadiánní osvětlení (změna teploty bílého světla v průběhu dne tak, aby korespondovala s přirozeným venkovním prostředím a kopírovala lidský cirkadiánní rytmus), dále lze osvětlení použít pro signalizaci různých událostí (např. zvonek, příchod člověka), pro vizualizaci hudby aj.
- 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. Tato kategorie zastřešuje velké množství senzorů, od čidel teploty a vlhkosti v místnostech po meteostanice.

- Entertainment – hlavně multimediální systémy, přehrávání videa i audia z různých zdrojů (lokální úložiště, TV, rádio, stream z internetu, stream z mobilního zařízení...), multi-room audio (přehráváni hudby na jakémkoli zařízení v domě nebo i synchronizovaně na více z nich zároveň), hry (konzole, ovladače, VR) aj. Návaznost na ostatní kategorie (např. vypnutí hudby při příchozím telefonním hovoru, ztlumení světel při spuštění filmu...). Někdy se sem zařazují I technologie bazénů a saun.
- Security – integrované prvky zabezpečovacích systémů (EZS) a systémů řízení přístupu (ACS), tedy alarm, zámky, garážová vrata, vstup pomocí RFID klíče nebo kódu a podobně. Opět návaznost na ostatní kategorie (využití pohybových senzorů pro alarm i osvětlení, okenních senzorů pro alarm i HVAC, otevírání garáže z palubního počítače v autě aj.) Patří sem i kamerové systémy, naopak sem nepatří EPS.
- Spotřebiče – propojení „chytrých" elektrospotřebičů se systémem. Těchto spotřebičů na trhu přibývá, jsou to jednak 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). Řadí se sem i běžné spotřebiče doplněné o nějaký prvek komunikující se zbytkem systému, což může být i jen bezdrátově řízená spínací zásuvka.
- Control – Ovládací prvky systému, se kterými uživatel fyzicky interaguje od elektroinstalačních spínačů a tlačítek, přes terminály s dotykovým displayem až po smartphone, TV nebo počítač. Každý typ ovládacího prvku se hodí pro jinou aplikaci, např. pro zapnutí a vypnutí světla na WC je místo smarphone vhodnější klasické tlačítko na zdi. Ovládací prvky jsou zpravidla bez omezení mapovatelné na jakékoli ovládané prvky. Kategorie dále obsahuje senzory, které se nedají jednoznačně zařadit do některé z ostatních.
- 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. Dále sem patří detekce přítomnosti osob (myšleno jako přítomnost konkrétního uživatele v domácnosti, realizuje se např. detekcí připojení smartphone k lokální Wi-Fi síti nebo přes Bluetooth beacon) a detekce havarijních stavů (výpadky elektřiny, únik vody nebo plynu...). Elektronická detekce požáru (EPS) by měla zůstat jako samostatný jednoúčelový systém oddělený od systému smart home, jelikož musí splňovat určité speciální normy.[7]

- Ostatní – další zařízení podle typu objektu a specifických potřeb uživatele, např. pevný telefon (VoIP), zavlažovací systémy nebo monitoring akvária/terária/hospodářských zvířat. Dále také spolupráce s ostatními systémy jako je automobil, domácí IT infrastruktura (řízení přístupu k internetu, domácí multimediální server etc.) nebo různá veřejná API na internetu pro poskytování dat o počasí, dopravě, objednávky jídla a podobně, zde je třeba dbát na informační bezpečnost, (viz níže).

Implementace

Jak už jsem zmínil v úvodu, zájemce o chytrou domácnost má dnes tři hlavní možnosti:

- Profesionální systém navržený na míru dané budově. Tyto systémy jsou dodávany specializovanými firmami, které zároveň zajišťují instalaci na místě, konfiguraci a případný servis. Systémy se vyvinuly z prvních inteligentních elektroinstalací, pro montáž jsou zpravidla nutné stavební úpravy. Využívají se proprietární zařízení a komunikační protokoly, u některých komponent (ovládání zařízení mimo samotný systém – HVAC apod.) dlouhodobě ověřené 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. Výrobci těchto systémů si ale pomalu začínají uvědomovat, že takto uzavřená technologie nestíhá držet krok s rychlým celosvětovým vývojem, který v oblasti smart home v posledních letech probíhá, a začínají integrovat populární produkty ostatních firem, např. Sonos, Nest, Phillips Hue nebo Amazon Alexa. Výhodou profesionálních systémů je velmi vysoká spolehlivost a záruka na funkci celku, nevýhodou je zmíněná uzavřenost a samozřejmě relativně vysoká cena pořízení i servisu. Příkladem jsou v ČR populární systémy Loxone, nebo méně známý Control4.
- Jít cestou DIY. Systém kompletně navržený, zkonstruovaný a nakonfigurovaný uživatelem. Ve velkém se zde využívají univerzální mikrokontrolery (různé verze Arduino a ESP), SBC na platformě ARM (RaspberryPi a jeho klony) a open-source software (GNU/Linux, Home Assistant, Node-RED a mnoho dalších). Takový systém je omezen doslova jen dostupným hardwarem, schopnostmi a představivostí uživatele. Je třeba poznamenat, že „DIY“ už dnes neznamená „bastlení“ na koleně a zařízení, která jen tak tak fungují, pokud je tvůrce dostatečně schopný, může jeho systém, díky dnešní dostupnosti návrhových nástrojů a technologií (3D tisk, zakázková výroba a osazování PCB...), dosahovat funkčnosti a kvality profesionálních produktů a to za zlomek jejich ceny. A tím se dostáváme k největšímu problému – toto samozřejmě není řešení pro běžné uživatele, ale pro

„nadšence“, kteří se tematikou a technologiemi smart home do hloubky zabývají buď v práci nebo ve volném čase. Navíc ušetřené peníze v porovnání s pořízením profi systému jsou vyváženy množstvím času, které uživatel do tvorby systému vloží a také tím, že jakoukoli závadu na systému musí řešit sám. Vzhledem k tomu, že každý takovýto systém je unikátní, nelze zde uvést konkrétní příklady.

- Poslední možností, jsou tzv. off-the-shelf smart home produkty (také nazývané consumer IoT). Jedná se o víceméně univerzální elektroinstalační komponenty (žárovky, spínače, svítidla, zásuvky…) nebo přímo spotřebiče (ledničky, mikrovlnné trouby…), senzory (PIR, magnetické kontakty…), zámky dveří, měřiče energií aj., které jsou rozšířeny o komunikační rozhranní. To bývá nejčastěji bezdrátové (BLE, Zigbee, Wi-Fi) a umožňuje jednotlivé komponenty propojit a tím utvořit jakousi smart home stavebnici. Tento přístup v podstatě umožňuje komukoli vytvořit systém ve stylu DIY bez detailních znalostí technologií. Mnoha lidem se dnes pod pojmem „chytrá domácnost“ vybaví právě tyto produkty a zájem o ně stále roste. Hlavními výhodami jsou nízká cena a téměř nulová počáteční konfigurace, zařízení stačí zapojit a zapnout aby fungovalo a jakékoli další složitější nastavení se provádí většinou přes smartphone aplikaci. Pro složitější integrace bývají k dispozici brány, které umožňují připojení k počítačové síti, mnoho z těchto výrobků funguje bez centrální ústředny a místo toho využívá cloudové služby na serverech výrobce, případně třetí strany zajišťující podporu. Produktové řady různých výrobců nebývají přímo kompatibilní mezi sebou, typicky ale umožňují integraci s hlasovým asistentem Amazon Alexa nebo Google Home, ten pak slouží v podstatě jako ústředna. Připojení k internetu je výhodou z hlediska přístupu k datům ze zdrojů mimo budovu, ale nevýhodou z hlediska bezpečnosti, zejména koncová zařízení s Wi-Fi rozhraním (přímý přístup do sítě bez brány) často postrádají dostatečné zabezpečení komunikace a přístupu. Je třeba také pamatovat na to, že záruka výrobce je pouze na jednotlivá zařízení, ne na funkci celku, který z nich uživatel sestaví.

Problémy

Centralizovaný, propojený způsob návrhu má samozřejmě i své nevýhody. První z nich je z hlediska funkcionality, momentálně neexistuje jednotný standard pro propojení a integraci produktů různých výrobců, kompatibilitu mezi nimi je tak před realizací vždy třeba ověřit. To platí jak pro uzavřené profi systémy (kde lze většinou používat jen komponenty výrobce systému), tak pro off-the-shelf výrobky, u kterých je nutno kompatibilitu vždy před koupí nové komponenty pečlivě ověřit.

Druhým a asi největším problémem je soukromí a informační bezpečnost. Jelikož profi systémy mají typicky jen jednu komponentu (bránu) pro přístup do domácí počítačové sítě a ta bývá adekvátně zabezpečena díky tomu, že konfiguraci provádí odborný technik, týká se tento problém hlavně off-the-shelf produktů. Mnoho zařízení z této kategorie je osazeno mikrokontrolery nebo SBC, které umožňují konektivitu přímo do počítačové sítě (Wi-Fi či Ethernet rozhranní), ale zabezpečení přístupu k ovládání zařízení, nebo čtení dat z něj neřeší. To samozřejmě není pravda u všech zařízení, ale u těch, které proti neoprávněnému přístupu zabezpečena jsou nastává stejný problem jako např. u SOHO Wi-Fi routerů – žádný software není bez chyby a nějaká bezpečnostní zranitelnost se dříve nebo později objeví.

U běžných aplikací na PC či telefonech tohle řeší pravidelné aktualizace, běžný uživatel však mnohdy nemá tušení, že by měl aktualizace provádět i na zmíněném routeru a na zařízeních své chytré domácnosti. Je také možné, že výrobce dané zařízení přestane podporovat a žádné nové aktualizace už dostupné nebudou. Značné množství produktů ke své funkci využívá cloudové služby, takže je ani není možné provozovat izolovaně bez připojení k internetu. Zranitelný systém pak potenciálnímu útočníkovi může poskytnout nejrůznější data o budově (obvzlášť 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 odemknout hlavní vchod, otevřít garážová vrata, vypnout EZS nebo způsobit škody skrz ovládání HVAC, vodovodních ventilů apod. Zatímco toto je v podstatě jen hypotetický případ (pro případného zloděje je mnohem snažší překonat fyzické bariéry – např. rozbít okno – než se snažit napadnout budovu elektronickou cestou), úniky osobních dat a kamerových záznamů jsou bohužel časté. Tento problém je nejzávažnější u produktů, které využívají cloudové služby místo fyzické ústředny a pro jejich provoz je tedy nutnou podmínou přístup k internetu. Obecně je smart home oblastí, kde se fyzická bezpečnost setkává s bezpečností informační a při návrhu je potřeba myslet na obojí.

Nakonec ještě připomínka spíše než problém, a sice to, že systém by měl sloužit uživateli, co nejvíce mu šetřit čas a práci a fungovat transparentně na pozadí, aby uživatel nad prováděnými procesy nemusel vůbec přemýšlet. 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. Pokud tento uživatel chápe celý systém jen jako nástroj a nemá čas nebo chuť ho blíže zkoumat, časem ho nejspíš přestane používat a získá negativní náhled na celý smart home koncept. Tímto nejvíce trpí právě uživatelé, kteří se snaží z off-the-shelf produktů poskládat celek s pokročilejšími funkcemi.

Možná budoucnost

Jak tedy zmíněné problémy vyřešit? Každý z přístupů má své výhody i nevýhody, ideální by tedy 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. Zde bych tedy následoval vzor profi systémů, to znamená odborná firma zajišťující návrh, instalaci a servis. Tím by byla zajištěna určitá úroveň kvality, spolehlivosti a bezpečnosti a zároveň záruka pro uživatele na funkci smart home jako celku. Není však nutné vyvíjet svůj vlastní uzavřený systém, když jsou dnes na trhu dostupné off-the-shelf produkty, které při správném použití velmi dobře plní svůj účel a jejich ceny se blíží cenám prvků klasické elektroinstalace. Navíc u nich další vývoj a nové funkce řeší jejich jednotliví výrobci, takže není třeba se bát, že by systém zaostával za aktuálními trendy. Zbývající mezery jako je ústředna, ovládací rozhraní a integrace s ostatními technologiemi (HVAC...) bych po vzoru DIY řešení vyplnil univerzálními MCU/SBC a open source softwarem, s tím že jako jednodušší ovládací prvky a dodatečné sensory by šlo použít klasické elektroinstalační komponenty.

Takový systém mi přijde jako ideální kombinace dosavadních přístupů, která má zároveň nejnovější funkce, nízkou pořizovací cenu a profesionální podporu. Podobný přístup, kde uživatel platí za instalaci, podporu a servis a samotná využitá technologie je otevřená a běžně dostupná s úspěchem uplatňuje v IT světě, např. firma Red Hat. Podpora by mohla fungovat buď opět jako u správy IT, kdy za paušální poplatek má uživatel garantované uvedení systému do funkčního stavu v určitém časovém limitu od výskytu závady, nebo jako záruka na elektroinstalaci, kdy uživatel platí jen za provedené servisní úkony, ale nemá zmíněnou garanci opravy v daném čase. Vzhledem k tomu, že použitý HW a SW je otevřený a má tedy veřejně dostupnou dokumentaci, mohly by teoreticky instalaci a servis provádět různé firmy.

Bezpečnost

Z pohledu bezpečnosti (informační) zvyšuje centralizace a propojenost technologií v budově výrazně zranitelnost celku. I zde bych se inspiroval současnými postupy v IT pro zabezpečení serverů a sítí. Ústředna by běžela virtualizovaně na domácím serveru (který se čím dál tím častěji používá v domácnostech pro sdílení dat mezi více počítači a mobilními zařízeními, zálohování, streaming apod.), případně na vlastním SBC a pomocí kvalitního routeru s firewallem by byla, společně s připojenými zařízeními, izolována do vlastní podsítě. Další podsítě by byly vytvořeny pro běžnou LAN a pro kamerový systém, případně i jiné účely podle konkrétní domácnosti. Komunikace mezi touto smart home podsítí a ostatními, hlavně internetem, by byla omezena jen na nejnutnější provoz, jako je

stahování aktualizací, sdílení multimediálních dat pro přehrávání na TV a podobně. Žádné z koncových zařízení v této síti by tedy nemělo přímý přístup k internetu, takže případné chyby těchto zařízení neovlivní bezpečnost systému, jediným kritickým prvkem je samotná ústředna. 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. Ústředna může zároveň hostovat další software, např. pro dohled nad systémem a sítí (Icinga aj.). Všude, kde to jen trochu jde bych použil drátové komunikační rozhraní, kvůli dalšímu posílení bezpečnosti (hlavně u dveřních zámků a podobných zařízení), omezení vlivu zarušení radiových frekvencí a minimalizaci počtu zařízení napájených z baterií.

Od IoT k NoT

Snažil bych se i o to, aby všechna propojená zařízení byla schopna fungovat samostatně (tam kde to dává smysl, např. zařízení které je jen senzorem pro zbytek systému samostatně fungovat nemůže) a při ztrátě konektivity přišla jen o funkce, které danému zařízení konektivita poskytuje. Minimálně je nutné, aby při výpadku konektivity k ústředně byl koncový prvek schopen bezpečně vypnout ovládanou technologii. Koncové prvky postavené na otevřených technologiích budou multifunkční a bude jim možno časem další funkce přidávat, např. Raspberry Pi s dotykovým displayem umístěné na stěně předsíně může sloužit jako ovládací terminál, audio výstupní bod pro danou místnost, senzor různých veličin, obrazovka videovrátného atd. To zajišťuje tzv. future-proofing systému (tedy to, že v nejvyšší možné míře udrží krok s budoucím vývojem v oblasti smart home), což je v poslední době velmi diskutované téma, protože přístup používaný u spotřební elektroniky, kde uživatel jednou za pár let zařízení jednoduše vymění za novější model, je u smart home problematický [1]. Hlavní propojovací struktura by byl zmíněný oddělený segment domácí počítačové sítě využívající Ethernet a Wi-Fi, zařízení s jinými komunikačními technologiemi pak připojena pomocí bran. V rámci této práce se právě snažím vytvořit komunikační bránu, která společně s ústřednou a několika off-the-shelf koncovými prvky (např. Phillips Hue, Sonoff, UniPi Neuron…) může tvořit základ smart home v tomto duchu.

Tento princip by se dal označit i jako Internet of Things (internet věcí), což je pojem, který vznikl právě pro popis skupiny jednoduchých elektronických zařízení, které fungují samostatně, přičemž po připojení do společné sítě má každý uzel přístup k datům poskytovaným ostatními, případně může ostatní ovládat. Dnes se ale pojem IoT používá pro více různých konceptů, často i nesprávně a jeho význam se stává nejasným [8], navíc „Internet“ evokuje přímé připojení do veřejného internetu, které je zde nežádoucí. Nazvu tedy tento koncept jednoduše jako NoT – network of things (síť věcí), pojem je převzatý od jednoho z vývojářů open source ústředny Home Assistant Francka Nijhofa, který tak označuje právě infrastrukturu svého smart home.

Komunikační brána

Teď už se dostávám k bráně samotné a průzkumu dostupných komponent a technologií, které by šly použít pro její konstrukci a realizaci požadovaných funkcí. Základními bloky bude centrální řídící prvek (jádro), napájecí zdroj, periferie pro bezdrátovou komunikaci s koncovými prvky, periferie pro komunikaci s ústřednou a v neposlední řadě senzory. Při výběru komponent budu brát v potaz i to, že bránu by mělo být možné zkonstruovat i v domácích podmínkách.

Jádro

Jako řídící prvek brány bych zvolil nějaké univerzální zařízení, které lze jednoduše programovat/konfigurovat, a to i po tom, co je konstrukce brány hotová. Kromě toho, že návrh a realizace je daleko jednodušší než při použití běžného mikrokontroleru naprogramovaného např. v jazyce C, bude možné v budoucnu i jednoduše modifikovat funkce brány a přidávat nové. Při volbě je třeba myslet také na požadavky pro konektivitu (Ethernet, Wi-Fi, Bluetooth a Zigbee). Nabízí se tři hlavní možnosti:

Arduino

Vývojová platforma Arduino vznikla jako studentský projekt v Itálii v roce 2005 a jejím cílem bylo poskytnout rychlý a levný způsob návrhu prototypů vestavných systémů, který by zároveň nevyžadoval detailní znalosti mikrokontrolerů a jejich programování a mohli ho tak využívat i lidé bez specializace na elektroniku, např. architekti, návrháři oblečení, umělci a kutilové. Arduino se setkalo s obrovským úspěchem, pomohlo odstartovat vlnu „moderních bastlířů“ známých jako Makers (maker movement) a dodnes zůstává nejoblíbenější platformou tohoto typu.

Základem je softwarové vývojové prostředí Wiring, které poskytuje relativně vysokoúrovňový programovací jazyk velmi podobný C++ (syntaxe je identická a jazyk podporuje i mnoho C++ konstrukcí včetně objektového programování). Prostředí dale poskytuje překladač, který abstrahuje detaily architektury procesoru, takže z pohledu programátora téměř nezáleží s jakým mikrokontrolerem pracuje. Poslední komponentou je zavaděč nahraný v paměti mikrokontroleru. Označení “zavaděč” (angl. bootloader) zde není úplně přesné, tato komponenta slouží v podstatě jako vestavěný programátor a umožňuje nahrávat nové program přes sériové UART rozhraní.

Název Arduino oficiálně označuje jak celý projekt, tak vývojové desky navržené a distribuované firmou Arduino LLC, která byla založena vývojáři projektu a drží jeho ochranné známky. Neoficiálně se používá i pro vývojové prostředí Wiring a samotné mikrokontrolery s Arduino zavaděčem, což občas ztěžuje hledání informací. Arduino je otevřená technologie, hardware je licencován pod

Creative Commons Attribution and ShareAlike a software pod GNU GPL, obojí je tedy možno volně distribuovat a upravovat schémata i zdrojové kódy jsou veřejně dostupné.

Původně byla platforma postavena na mikrokontrolerech Atmel AVR, časem přibyly procesory ARM, Atheros a další. Přestože oroginální účel bylo prototypování, platforma se časem osvědčila jako spolehlivá a často se používá i jako finální řešení, např. právě v DIY smart home. V projektech není nutné používat celé vývojové desky, stačí samotný mikrokontroler s nahraným zavaděčem.

I když je teoreticky možné bránu na této platformě postavit, ve srovnání s alternativami by to bylo daleko komplikovanější, stejně jako případné budoucí úpravy a rozšíření. Většina z mikrokontrolerů, které jsou na výběr nemá dostatečný výkon, v úvahu připadá jedině model Arduino Zero s procesorem ARM ATSAMD21G18, kvůli konstrukci tohoto modelu by ale bylo nutné použít celou desku, tím bych přišel o hlavní výhodu této platformy. Po stránce software má Arduino k dispozici nespočet knihoven pro drtivou většinu běžně používaných periferií, zrovna knihovny pro komunikační periferie jako např. Wi-Fi ale poskytují jen nízkoúrovňové rozhraní a veškeré aplikační protokoly by bylo nutno implementovat ručně. I když je tedy tato platforma často využívána v DIY smart home, je vhodná hlavně pro koncové prvky jako jsou senzory a ovladače, ne pro komunikační uzly.

ESP

Čipy Čínské firmy Espressif Systems začaly jako velmi levné Wi-Fi periferie určené pro mikrokontrolery, které v sobě integrovaly kompletní TCP/IP stack a umožňovaly tak danému MCU komunikovat po IP síti bez výrazného dopadu na jeho výkon. První iterace byla distribuována jako modul ESP-01, který měl pouze osmipinové rozhraní a stručnou dokumentaci popisující základní příkazy pro Wi-Fi komunikaci.

Modul se rychle stal populárním právě v Maker komunitě (viz předchozí sekce), ta zjistila, že mikrokontroler ESP8266, na kterém je modul postaven výkonem předčí nejpoužívanější mikrokontroler Arduino platformy ATMega328. To bylo společně s nízkou cenou a integrovanou Wi-Fi konektivitou motivací k překladu kompletní čínské dokumentace a následné konstrukci vývojové desky NodeMCU, využívající plného potenciálu ESP8266, což je kromě Wi-Fi navíc 16 GPIO pinů, SPI, I2C UART a ADC. Pro NodeMCU vzniklo jednoduché vývojové prostředí využívající jazyk LUA.

Poté co Arduino projekt začal používat mikrokontrolery jiných výrobců než Atmel, byl do vývojového prostředí Wiring doplněn mechanismus pro jednoduché přepínání mezi alternativními překladači pro tyto nové mikrokontrolery pomocí pluginů. Krátce na to vznikl i komunitní plugin pro ESP8266 a Arduino se stalo dominantním vývojovým prostředím pro ESP.

Výrobce Espressif v důsledku tohoto úspěchu začal vyrábět vlastní moduly pro ESP8266 a v roce 2016 přišel s jeho nástupcem ESP32, který přinesl

vyšší výkon, více GPIO, piny pro kapacitní snímání, hardwarové šifrování, Bluetooth a další.

Velkou výhodou při použití ESP32 by byla integrovaná Wi-Fi i Bluetooth konektivita. Výkon procesoru by měl být dostačující. Při implementaci pomocí vývojového prostředí Arduino ale zůstávají všechny nevýhody zmíněné v předchozí sekci. Kromě Arduina jsou dostupná i další vývojová protředí, např. MicroPython nebo prostředí přímo od výrobce založené na operačním systému reálného času FreeRTOS.

ARM SBC

Poslední možností jsou jednodeskové počítače (SBC = Single-Board Computer) postavené na procesorech ARM. Jak název napovídá, jedná se o víceméně plnohodnotný počítač na jedné desce plošných spojů, na rozdíl od klasického PC, který má základní desku a do ní připojené další rozšiřující desky a komponenty. SBC jsou tedy kompaktní, jejich hlavní komponenty (procesor, paměť...) nejsou vyměnitelné/rozšiřitelné, bývají navrženy pro nízkou spotřebu energie a jednou z jejich hlavních aplikací je právě řízení vestavných systémů.

Srdcem SBC je tzv. SoC (Systém on Chip), což je sofistikovaný integrovaný obvod, ve kterém je spojeno CPU, GPU, paměť a řadiče vstupně-výstupních periferií. Procesory ARM jsou pro tuto aplikaci vhodné díky své nízké spotřebě energie a malému množství odpadního tepla které produkují.

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 různé aplikace, např. Pi 4 se svým čtyřjádrovým procesorem, až 8GB RAM a výkonným GPU je určeno pro aplikace kde je potřeba vyšší výkon, Pi Zero vyniká zase nižší spotřebou, nižší cenou a přibližně třetinovou velikostí Pi 4. Popularita Raspberry Pi dala vzniknout mnoha klonům od různých výrobců určeným pro různé případy použití, fakt že parametry SBC nejsou konfigurovatelné je tedy vyvážen velkým výběrem modelů.

Vzhledem k tomu, že SBC fungují jako plnohodnotný počítač s 32-bitovou architekturou (existují i 64-bitové), lze na nich provozovat i plnohodnotný operační systém, u většiny SBC je podporována některá odnož GNU/Linuxu, často jsou ale potřeba nestandartní ovladače dodané výrobcem (běžně jsou v Linuxu ovladače součástí jádra OS), bez kterých některé funkce SBC (např. GPU, síťové rozhraní, audio...) nejsou dostupné. Raspberry Pi s ovladači nemívá problémy, některé jeho klony jsou ale špatnou softwarovou podporou notoricky známé (viz níže). SBC také disponují stejnými rozhraními jako PC – USB, Ethernet, HDMI, Wi-Fi, Bluetooth aj, lze tedy připojit mnoho různých periferií a používat veškerý software dostupný pro daný OS.

Kvůli více než dostatečnému výkonu, množství komunikačních rozhraní a rozšiřitelnosti je použití SBC jako řídícího prvku brány velmi vhodné. Navíc lze

díky operačnímu systému využít již hotové softwarové komponenty, jako např. webserver, firewall nebo projekty typu Zigbee2MQTT, není tedy třeba vše implementovat ručně. Nevýhodou je, že SBC jsou v porovnání s mikrokontrolery daleko složitější zařízení, mají tím pádem vyšší spotřebu energie a musí se integrovat jako celek (deska). SBC také mívají ve srovnání s MCU méně nízkoúrovňových rozhranní a funkcí, typicky chybí ADC/DAC a PWM. Při výběru SBC bych se soustředil na co nejmenší rozměry, malou spotřebu energie a komunikační rozhraní. Vhodnými příklady jsou:

- Raspberry Pi Zero W
  - 1GHz procesor, 512MB RAM
  - rozhraní USB (přes USB OTG port), Wi-Fi 802.11n (viz níže), Bluetooth 4.1 (viz níže)
  - SD karta jako systémový disk
  - OS Raspbian – odnož systému Debian GNU/Linux pro Raspberry Pi, ověřená plná podpora všech funkcí.
  - Cena cca 400 Kč.
- Orange Pi i96
  - 1GHz procesor, 256MB RAM
  - rozhraní USB, Wi-Fi, Bluetooth
  - SD karta jako systémový disk
  - OS Debian GNU/Linux, Ubuntu Linux, Android – nejlépe funguje se systémem Android.
  - Cena cca 250 Kč
- Banana Pi BPI-P2 Maker
  - 1GHz čtyřjádrový procesor, 512MB RAM
  - rozhraní USB, Wi-Fi, Bluetooth, velkým plusem je Ethernet s možností PoE napájení (viz níže)
  - SD karta nebo eMMC modul jako systémový disk
  - OS Debian GNU/Linux – dostupný je ale jen jeden instalační image ze stránek výrobce, a SBC Banana Pi jsou velmi známé špatnou softwarovou podporou, kdy system nainstalovaný z poskytnutého image přestane fungovat například i po instalaci bezpečnostní aktualizace.
  - Cena cca 450 Kč
- Onion Omega2+
  - velmi zajímavý SBC s maximálně kompaktní deskou, na které nejsou umístěny standartní konektory, jako tomu je u většiny SBC. Místo toho jsou rozhraní vyvedena na dva headery na protilehlých okrajích desky, SBC lze tedy v projektu použit stejným způsobem jako integrovaný obvod, viz obrázek.
  - 580 MHz procesor, 128MB RAM
  - rozhraní USB, Wi-Fi, Ethernet

- podpora SD karty jako systémového disku
- OS je upravený image LEDE Linux, použitý procesor není ARM a standardní Linuxové jádro pro něj bohužel nemá ovladače, takže jiný OS použít nelze
- Cena cca 300 Kč

Komunikace s okolím

Brána je primárně určena pro připojení off-the-shelf smart home výrobků k ústředně. Tyto výrobky téměř výhradně komunikují bezdrátově, a to jedním z následujících standardů:

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 all-in-one zařízení obsahující router, Wi-Fi AP, případně modem, do kterého je zavedena internetová konektivita, která je dále sdílena právě přes lokální Wi-Fi. Technologie je také často využívána regionálními 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 další verze 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 a technologie MIMO (multiple input multiple output) umožňuje přístupovému bodu (AP) obsluhovat více zařízení současně, což je u smart home sítě velmi užitečné. 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 obecně menší dosah a hůře pronikají pevnými materiály než ty s nižší). Tento standard se používá pro datové sítě, pro smart home koncové prvky není relevantní, protože implementace v MCU je složitá (tím pádem by zvyšovala cenu zařízení a spotřebu energie, na které velmi záleží hlavně u prvků napájených bateriemi) a vysoké přenosové rychlosti nejsou potřeba. 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), tedy přístupový bod, ke kterému jsou připojena ostatní

zařízení označovaná jako Client. Každé koncové zařízení tedy musí být v dosahu AP. Pro pokrytí průměrného rodinného domu stačí většinou jedno kvalitní AP v každém podlaží. AP má své SSID, které slouží jako identifikátor sítě pro klienty. Jedno AP může mít více SSID (a poskytovat tak např. více podsítí 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 první, nejnižší z nich zajišťuje fyzické spojení a řešení kolizí na přenosovém médiu (zde rádiové vlny), prostřední dvě vrstvy se starají o potvrzování, integritu dat, korekci chyb, adresaci a směrování. 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 v podstatě jen 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 jednou, cílové zařízení si pak síť zapamatuje, stejně jako při běžném způsobu připojení pomocí hesla. WPS bývá aktivní jen několik sekund až minut po stisknutí tlačítka, během této doby ale AP poskytuje připojovaným zařízením svůj šifrovací klíč a pokud je tento odposlechnut útočníkem, lze ho pak použít k rozšifrování další odposlechnuté komunikace. Vzhledem k tomu, že pro smart home zařízení nemá funkce WPS smysl, je vhodné ji v nastavení AP vypnout. PSK může být až 256bitů dlouhý (32 ASCII znaků) a jako u všech hesel pro něj platí, že čím delší a komplexnější, tím náročnější je prolomit ho hrubou silou.

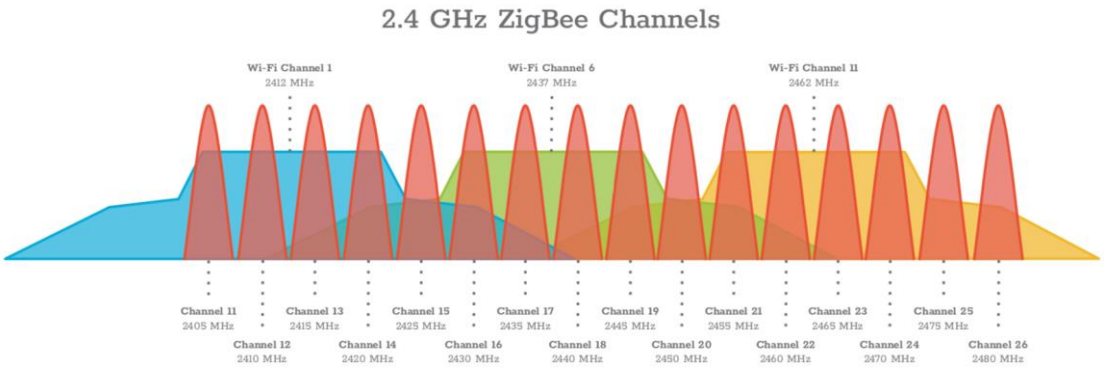
Wi-Fi je ve smart home vhodná pro koncové prvky, které potřebují větší datové toky (terminály, audio...), naopak se nehodí pro zařízení napájená bateriemi, kvůli větší spotřebě zařízení způsobené vyššími nároky na výkon. Pro implementaci v bráně se nabízí několik možností. ARM SBC mají Wi-Fi čip buď vestavěný, nebo u nich lze použít USB adaptéry určené pro běžné PC. Pro mikrokontrolery jsou dostupné různé moduly od ESP zmíněné v předchozí sekci.

Zigbee

Otevrený standard, se kterým přišla v roce 2005 asociace Zigbee Alliance, je na rozdíl od Wi-Fi určen specificky pro jednoduchá zařízení, která si vystačí s nižší datovou propustností přenosového kanálu, ale naopak od komunikačního rozhraní vyžadují nízkou spotřebu energie. Na to je kladen důraz i přímo ve specifikaci,

pokud nějaký výrobce chce, aby jeho zigbee zařízení (myšleno jako hotový prodkukt, např. bezdrátový vypínač) prošlo certifikací, musí mít výdrž na baterie minimálně dva roky. Tyto vlastnosti dělají ze Zigbee ideální protokol právě pro smart home a automatizaci budov obecně.

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 mnohem užší a je snažší najít jeden volný. Kanály a překryv s Wi-Fi je znázorněn na obrázku 1. S užšími kanály je spojena i maximální přenosová rychlost 250kbps, která odpovídá účelu protokolu.



Obrázek 1 - 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. Díky tomu stačí při optimálním rozmístění zařízení v budově jen jeden řídící uzel (Coordinator – viz níže). Definovány jsou tři operační módy zařízení.

- Coordinator (ZC) je uzel určený jako kořen sítě, zajišťuje řízení sítě, je zodpovědný za tvar meshe po připojení nových uzlů a umožňuje propojení Zibee 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.
- Router (ZR) je jedním z koncových zařízení, které plní svou určenou funkci, ale zároveň funguje jako prostředník pro předávání dat mezi ostatními uzly. Routery musí být vždy v pohotovosti a proto nemohou využívat režim spánku, jsou to tedy koncové prvky napájené přímo ze sítě, např. žárovky. Router může pracovat také jen jako opakovač signálu, pro vykrytí příliš velké mezery v meshi. Pokud mesh není optimální a router je jediným spojovacím bodem mezi nějakým segmentem sítě a coordinatorem, při jeho výpadku přestane tento segment fungovat.

- End Device (ZED) je koncové zařízení které pouze plní svou funkci a umí komunikovat s rodičovským uzlem (ZR/ZC). Typicky bateriově napájené sensory a ovladače, které jsou většinu času v režimu spánku kvůli šetření energie, při aktivaci se na okamžik probudí, odešlou data a opět se uspí.

Na rozdíl od Wi-Fi specifikuje Zigbee standard všechny vrstvy komunikace, ty jsou zde ale jen dvě. 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 ale nastává hlavní problém s interoperabilitou, 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 zpravidla přímo kompatibilní (příkladem mohou být produktové řady pro osvětlení Phillips Hue a IKEA Tradfri). I když se systémy dají propojit nepřímo pomocí HUBu, jednotlivé prvky místo jedné společné meshe budou tvořit dvě samostatné, což není optimální a může způsobovat problémy s pokrytím objektu.

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 připojení zažádá o vstup do sítě (zpravidla aktivováno „párovacím“ tlačítkem na těle přístroje) a dostane od coordinatora přidělen klíč. Při tomto procesu je síť zranitelná vůči odposlechnutí klíče. Další komunikace s uzlem 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, nevýhodou je nutnost od základu implementovat mnoho funkcí nutných pro provoz brány.
- 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, který umožňuje relativně snadnou integraci s ústřednou. Tato možost vychází také cenově nejlépe, vyžaduje ale připojení přes USB
- Phoscon ConBee/RaspBee – modul coordinatoru firmy Phoscon (Dresden Elektronik) specificky určený jako univerzální hub pro smart home zařízení. Nabízí velmi propracované webové rozhraní včetně REST API a

podporuje mnoho off-the-shelf smart home produktů bez nutnosti nastavení. Software je ale proprietární, modul vyžaduje připojení buď přes USB nebo přes GPIO header na RaspberryPi (ve verzi RaspBee) a jeho cena je vyšší než cena RaspberryPi 3B+.

Z-Wave

Z-Wave má se zigbee podobný nejen název, ale i většinu svých vlastností. Jedná se opět 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. Čipy pro Z-Wave navíc oficiálně smí vyrábět pouze její mateřská společnost Silicon Labs (Silicon Laboratories, Inc.). 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. Dále to znamená, že vývoj a budoucnost standardu je výhradně v rukou Silicon Labs.

Podle své první specifikace využívala Z-Wave frekvenci 908.42 MHz, což je bezlicenční pásmo v Americe, v ostatních částech světa ale už ne, takže pracovní frekvence závisí na konkrétním regionu nebo dokonce 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 další sekundární. Ostatní uzly se označují jako slave a jsou rovnocenné. Síť má 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.

I když má tento protokol své výhody, dle mého názoru jeho nevýhody převažují a na trhu zůstává jen proto, že jeho konkurent Zigbee vznikl až o tři roky později. Jako Z-Wave hardware pro bránu je dostupných několik USB modulů, ale vzhledem k jejich vysoké ceně, složité integraci a hlavně tomu, jak málo je v ČR Z-Wave rozšířený (pro představu – 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) nebudu tento protokol implementovat.

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. Název protokolu je inspirován starým skandinávským králem, který v desátém století propojil roztříštěné Dánské kmeny do jednoho království a po jeho vzoru měl Bluetooth propojit dosud různorodé komunikační protokoly. [9] 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, přičemž 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 dvoutřetinové 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 (senzory) nebo na základě vnější události (tlačítko) 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 100 m s přímou viditelností, reálně se ale pohybuje v rozmezí 10-30 m, je třeba také počítat s velkým provozem v pásmu 2.4GHz, stejně jako u Wi-Fi a Zigbee.

Základní topologie je hvězdice, kde jedno zařízení se nazývá *centrální* a ostatní k němu připojené jsou *periferní*. Detaily topologie se ale liší podle tzv. provozního profilu, který definuje daný případ 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 měření krevního tlaku. Profilů je mnoho a každý z nich v podstatě definuje nový komunikační protokol. 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 firmy Xiaomi, který

používám pro testování BLE konektivity brány, 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), které jsou původně určeny 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.

S bezpečností má Bluetooth dlouhodobé problémy a BLE není výjimkou. Zařízení mohou pracovat buď v broadcast režimu, tedy vysílat svá data otevřeně pro všechna zařízení v dosahu – zde není zabezpečení žádné – anebo v režimu, kdy jsou zařízení předem spárována a komunikace mezi nimi je šifrovaná. Implementace v protokolu 4.0 ale obsahuje zranitelnost, 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 dostatečně zabezpečený.

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.
- Vestavěné BLE rozhranní v mikrokotroleru ESP32
- Vestavěné BLE rozhranní v některém z ARM SBC
- Jakýkoli USB Bluetooth adaptér – je nutné mít USB rozhraní, to v podstatě znamená ARM SBC a u toho je vždy lepší zvolit model s integrovaným BLE.

433 MHz

Poslední z technologií je rádiová komunikace na frekvenci 433MHz. 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 10 mW. Tato frekvence teoreticky umožňuje komunikaci na velkou vzdálenost, v praxi je ale tato vzdálenost omezena právě malým vysílacím výkonem a pohybuje se kolem 100 m při přímé vidi-

telnosti. Rychlost komunikace je zhruba 5 kbps, dostatečná pro přenos jednoduchých ovládacích příkazů, případně malých objemů dat (např. hodnota teploty a vlhkosti ze senzoru).

Protokoly pro digitální komunikaci v tomto pásmu záleží kompletně na výrobci zařízení, k dispozici je jen holá fyzická vrstva komunikace a je nutné implementovat veškeré pokročilejší 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. To ale nebývá tak složité a pro mnoho oblíbených zařízení lze na internetu najít neoficiální dokumentaci.

Nejčastější využití této technologie jsou dálkové ovladače k automobilu (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.

Možnost implementace v bráně je jen jedna, a to RF moduly pro toto pásmo, vybavené jednoduchým rozhraním v podobě jednoho datového pinu. Pro obousměrnou komunikaci je třeba mít modul vysílače i modul přijímače. Jelikož je ale tato technologie postupně nahrazována novějšími standardy, a také kvůli komplikacím s proprietárními protokoly jsem se rozhodl tento typ komunikace v bráně neimplementovat.

Komunikace s ústřednou

Brána je z principu jen prostředníkem pro předávání informací, je potřeba zajistit propojení s ústřednou, která bude informace zpracovávat. Ústředna je software běžící na domácím serveru. Pro zjednodušení systému by bylo vhodné, aby komunikační rozhraní zároveň zajišťovalo napájení brány. Nabízí se tedy dvě hlavní možnosti:

USB

Standard Universal Serial Bus (univerzální sériová sběrnice) vznikl v roce 1996 jako výsledek snahy sedmi z největších IT společností té doby sjednotit mnoho různých komunikačních rozhraní a protokolů používaných pro připojování externích periferií k PC. Kromě definice fyzického rozhraní (konektorů) a způsobu přenosu dat (synchronní sériová linka) určuje USB mnoho dalších vlastností, z nichž ty klíčové jsou hot swap (připojení/odpojení za běhu), plug & play (periferie funguje hned po připojení bez nutnosti ruční konfigurace nebo instalace ovladačů) a napájení připojené periferie přímo z rozhraní. Cíl projektu byl víceméně splněn,

kolem roku 2010 už USB nahradilo všechna rozhraní kromě těch používaných pro připojení monitoru (DVI, HDMI) a síťovou komunikaci (Ethernet) a stalo se i standardem pro napájení a nabíjení baterií v mobilních telefonech a dalších přenosných zařízeních.

V současnosti existují dvě souběžně provozované verze USB. Starší USB 2.0 je jednodušší na implementaci, ale pomalejší, s maximální teoretickou přenosovou rychlostí 480 Mbps. Je vhodná pro periferie, které rychlejší datový přenos nevyužijí, např. klávesnice a myši, tiskárny apod. Novější a rychlejší USB 3.0 se s maximální teoretickou přenosovou rychlostí 5 Gbps a vyšší proudovou zatížitelností až 900 mA naopak hodí např. pro externí pevné disky. USB má architekturu master-slave s jedním hlavním uzlem (host) a až 127 podružnými zařízeními, ta mohou komunikovat pouze s hostem, ne přímo mezi sebou.

Pro aplikaci jako je vytvářená brána je rychlost 480 Mbps více než dostatečná, stejně jako maximální proudové zatížení 500 mA. Použil bych tedy USB 2.0. Hlavním omezením je maximální délka propojovacího kabelu, která by neměla překročit 5 m. Nejjednodušší realizací propojení pomocí USB by bylo na straně brány využít UART rozhraní vybraného MCU/SBC a převodník USB-UART. Maximální rychlost komunikace by se sice omezila na 115.2 kbps, to by mělo ale stále postačovat. Na straně ústředny (hosta) je USB nativní a k připojené bráně by se přistupovalo jako k emulovanému sériovému portu. Z hlediska software by bylo nutné buď navrhnout vlastní komunikační protokol, nebo použít nějaký univerzální, např. Firmata. Napájení je, jak jsem už zmínil, přímo součástí specifikace, napětí 5 V a maximální zatížení 500 mA by bráně poskytovalo 2.5 W, přičemž spotřeba samotného řídícího prvku (viz sekce *jádro*) se pohybuje mezi 0.3-0.7 W.

Ethernet

Ethernet je skupina protokolů popsaná standardem IEEE 802.3 a určená pro komunikaci v lokálních počítačových sítích (LAN). Standard vznikl už v roce 1983, nejstarší verze využívaly koaxiální kabel, časem se ale přešlo na kroucenou dvoulinku (4 páry vodičů) a optická vlákna.

Ethernet je v současnosti v podstatě jediný používaný standard pro metalickou a optickou LAN. Různé verze mají různé přenosové rychlosti, dnes nejpoužívanější jsou FastEthernet (100BASE-T) s rychlostí 100 Mbps a Gigabit Ethernet (1000BASE-T) s rychlostí 1 Gbps. Výhodou oproti USB je výrazně vyšší dosah – až 100 m, mnoho budov navíc už má zbudovanou strukturovanou kabeláž pro rozvod Ethernetu. Ethernet má hvězdicovou topologii, jednotlivá koncová zařízení jsou připojena do centrálního propojovacího prvku zvaného switch (přepínač). Na rozdíl od USB jsou si koncová zařízení rovnocenná a jakýkoli uzel může komunikovat s kterýmkoli jiným.

Nevýhodou je, že základní specifikace neposkytuje napájení, to lze ale vyřešit tzv. PoE (Power over Ethernet), což je souhrnné označení pro různé modifikace, které umožňují přenos dat a napájení po stejných vodičích. PoE může být pasivní nebo aktivní, u pasivního PoE je jedna strana jednoduše připojena ke zdroji napětí (buď switch s PoE funkcionalitou, nebo samostatný injektor). Starší 100 Mbps Ethernet potřebuje k přenosu dat jen dva ze čtyř párů kroucené dvojlinky, zbylé dva jsou tedy použity pro napájení. U gigabitového ethernetu, který potřebuje všechny páry pro data se využívá princip fantómového napájení (napájení a signál jsou vedeny po stejných vodičích, ale díky tomu, že napájení je stejnosměrné a signál je střídavý s frekvencemi v řádu stovek MHz, neovlivní se navzájem). Napájecí napětí a další parametry definuje výrobce konkrétního síťového prvku. Aktivní PoE, standardizované jako 802.3af/at využívá stejné základní principy, napětí je ale určeno – 48 V a napájení je implicitně vypnuto. Po připojení koncového zařízení je zahájen proces, pomocí kterého si PoE switch nebo injektor ověří, zda koncové zařízení napájení potřebuje nebo ne. Tím se eliminuje možnost poškození koncových zařízení, která nejsou na PoE stavěna.

Stejně jako Wi-Fi implementuje Ethernet jen nejnižší vrstvu TCP/IP komunikace, v případě použití v bráně je tedy nutné zvolit ještě vhodný aplikační protokol. Zde jsou dvě hlavní možnosti:

- 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 nějakými abstraktními zdroji. Mechanismus, který to zajišťuje se pak nazývá REST API (Application Programming Interface). URL zde popisuje nějaký zdroj, tím může být cokoli, REST popisuje pouze logiku komunikace, implementaci zdrojů je potřeba vyřešit nějakým programem na webovém serveru, který příslušné REST API poskytuje. Lze si 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 používá např. JSON (viz sekce návrh a realizace).

  Broker poskytuje klientům různá tzv. témata (topics), ty mohou být dále dělena na podtémata, téma je popsáno nějakým textovým řetězcem, podtémata jsou oddělena lomítkem. Klient může do tématu buď něco zapsat (publish), nebo může téma odebírat (subscribe). Pokud některý 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: SUB *prostredi/loznice/teplota* (opět se řeší jen logika komunikace, ne logika řízení ventilu). Senzor teploty v ložnici bude naopak do tématu periodicky zapisovat: PUB *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.

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 samostatná 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. Existuje mnoho druhů rozdělených 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 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 vykazují termoelektrický jev (při zahřívání jejich spoje vzniká elektrické napětí úměrné teplotě). Pro měření je opět potřeba pomocný obvod (zde je to 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 zkalibrovaný, 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 zkalibrovaný, 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 je menší, má vyšší dotazovací frekvenci, ale nižší přesnost. Verze DHT22 je naopak větší, pomalejší (dotazovat hodnoty lze jen jednou za 2 sekundy, to je ale pro použití ve smart home dostačující), ale přesnější, zvolil bych tedy spíše verzi DHT22. Roz-

sah měřených teplot je -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 opět 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 digitální sběrnici, která je specifická jen pro tento senzor, díky jeho popularitě jsou ale dostupné knihovny pro nejrůznější platformy včetně Arduina a ESP.

- 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. Podle několika různých nezávislých testů měří vlhkost přesněji než DHT, i když podle dokumentace je přesnost menší. Napájecí napětí 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 výsledné zapojení ručně 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, 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. 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 – Jednoduchý a levný 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 1500

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. 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 400-3000 ppm s přesností ±20ppm.

Návrh a realizace

Při volbě komponent a realizaci brány jsem se řídil následujícími podmínkami, z nichž některé jsou dané zadáním této práce, jiné jsem si určil podle své představy toho, jak by mohl systém, ve kterém bude brána použita vypadat (viz sekce Možná budoucnost).

- Podpora Wi-Fi, Zigbee a Bluetooth
- Webové ovládací/konfigurační rozhraní
- Provoz na uzavřeném segmentu lokální sítě bez nutnosti internetového připojení
- Brána bude autonomní, předpokládá se sice připojení k ústředně, ale jen pro poskytování dat, brána pro vlastní funkci nebude potřebovat/využívat žádné komponenty ústředny
- Použití otevřených technologií a co možná nejvíce funkcí implementovat (z důvodu budoucí podpory a údržby) pomocí existujících projektů
- Deploy script/image – jednoduchá instalace a nasazení pomocí skriptu a/nebo obrazu disku
- Možnost budoucího rozšíření o další funkcionalitu (po stránce HW i SW)
- Přijatelná cena – žádná z komponent/periferií nebude mít vyšší pořizovací cenu než řídící prvek (jádro)

Raspberry Pi

Jako řídící prvek jsem zvolil Raspberry Pi model Zero W. Hlavními důvody byla nativní konektivita Wi-Fi, Bluetooth a USB, dostatečný výkon a plnohodnotný OS, na kterém lze mimo jiné hostovat i MQTT broker a tím udělat bránu kompletně autonomní.

Pro první spuštění je potřeba kromě samotného SBC jen Micro SD karta s obrazem operačního systému a 5 V napájecí adaptér. Pro napájení postačí adaptér schopný dodat proud 1 A. I když software brány není náročný na přístup k disku, je vhodné použít rychlejší Micro SD kartu třídy 10, případně rovnou třídu A1, jejíž určení je právě k provozu aplikací (má rychlejší náhodný přístup a méně se opotřebovává častými zápisy), na rozdíl od běžných karet určených primárně k ukládání dat a optimalizovaných pro sekvenční přístup. Kapacita 8 GB stačí. Pi Zero má jen jeden USB 2.0 OTG port, pro připojení periferií je tedy potřeba USB hub a OTG adaptér. USB v použitém SoC není implementováno úplně, zejména chybí funkcionalita hot swap, takže je třeba před připojením/odpojením USB zařízení Pi Zero vypnout, jinak vzniká risk poškození portu. Tento nedostatek se dá obejít použitím vhodného USB hubu, který hot swap umožňuje. Jelikož jsem koupil Pi Zero bez GPIO headeru, bylo nutné ho osadit, header bude využit pro připojení senzorů.

Na začátek uvedu, že instalaci a nastavení provádím z OS Linux Mint. 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 jsem použil 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. Prvotní nastavení lze provést dvěma způsoby, buď připojit klávesnici a monitor a nastavit lokálně, nebo nastavit po síti přes SSH, já zvolil druhý způsob. SSH není implicitně zapnuté, je nutné editovat několik souborů na připravené SD kartě. To lze jen pod root účtem na linuxovém systému, pokud není k dispozici, nezbývá než pro první nastavení připojit k Pi Zero zmíněný monitor s klávesnicí a upravit nastavení lokálně. Prvním souborem je */etc/ssh/sshd_ config*, kde je nutné upravit následující:

ChallengeResponseAuthentication yes
UsePAM yes
PermitRootLogin no

Dále bylo potřeba nastavit pevnou IP adresu v souboru */etc/network/interfaces*, Raspbian označuje všechna ethernetová rozhraní jako *enoX*, po připojení první síťové karty tato dostane přidělen identifikátor *eno1*:

```
auto eno1
iface eno1 inet static
        address 192.168.10.250
        netmask 255.255.255.0
        gateway 192.168.10.200
        dns-nameservers 8.8.8.8
```

Posledním krokem je symbolický link 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, je možné ji vložit do Pi Zero a zapnout ho. Protože ale chci konfiguraci řešit vzdáleně po SSH, je nutné připravit si ještě síťové rozhraní, viz následující sekce Ethernet. Po přípravě síťového rozhraní jsem se připojil k Pi Zero pomocí SSH na zvolené IP adrese *10.0.0.150* jako defaultní uživatel *Pi* s heslem *raspberry*, přepnul se na uživatele root příkazem *sudo su* - a pokračoval v konfiguraci:

```
echo IoTGW > /etc/hostname
usermod -l admin pi
apt-get update && apt-get upgrade
touch /etc/iotgw.conf
```

tím se nastaví jméno systému na *IoTGW*, změní jméno defaultního uživatele z *pi* na *admin*, provede aktualizace OS a vytvoří se konfigurační soubor brány, který pak budu používat v rámci jednotlivých komponent.

Posledním krokem je rozšířit systém souborů na celou SD kartu, spustil jsem nástroj *raspi-config* a vybral první volbu „Expand Filesystem“.

Ethernet

Narozdíl od většiny ostatních modelů není Pi Zero osazen Ethernetovým rozhraním, je tedy nutné použít USB síťovou kartu. Vzhledem k omezené rychlosti USB 2.0 jsem zvolil 100 Mbps ethernet, na konkrétním modelu v podstatě nezáleží, já použil adaptér U2LAN od firmy i-tec, který se mi v minulosti osvědčil jako kvalitní a spolehlivý. Kartu stačí zapojit do jednoho z USB portů připojeného hubu, po startu systému je automaticky načtena, dostane přiřazen první volný identifikátor síťového rozhraní *eno1* a nastaví se podle příslušných záznamů v */etc/network/interfaces*.

Díky použití 100 Mbps ethernetu je také možné napájet bránu pasivním PoE pomocí jednoduchých OEM injektorů. Na vstupní straně jsem použil napájecí adaptér 24V/1A, výstupní injektor je připojen k DC-DC step-down měniči

s LM2596, který napětí mění a stabilizuje na 5V potřebných pro Pi Zero. Tento způsob je o poznání jednodušší a levnější než implementovat aktivní PoE, je ale třeba dát pozor pokud je místo brány na druhém konci připojeno nějaké jiné zařízení bez PoE podpory, mohlo by dojít k jeho poškození.

MQTT

Jako hlavní způsob komunikace jsem se rozhodl použít protokol MQTT. Díky výběru Raspberry Pi jako řídícího prvku je možné provozovat broker jako komponentu brány a používat MQTT částečně i pro vnitřní komunikaci mezi ostatními komponentami. Dále je protokol použit pro komunikaci s Wi-Fi koncovými prvky, mosty pro Bluetooth a Zigbee a samozřejmě s ústřednou.

Vybral jsem open-source broker Mosquitto napsaný v jazyce C, který je nenáročný na výkon a poskytuje všechny potřebné funkce, což je pro mou aplikaci v podstatě jen samotný protokol MQTT (lze zvolit verzi 3 nebo 5) a řízení přístupových práv 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 ústředny (např. Node-Red) a komponenty, které používám pro Zigbee a Bluetooth (viz níže). Příkladem JSON řetězce může být tato zpráva pro rozsvícení žárovky:

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

Nejdříve jsem z distribučních balíčků nainstaloval Mosquitto a nástroje pro jeho testování:

apt-get install mosquitto mosquitto-clients

Broker se při instalaci sám nastaví jako služba spouštěná při startu systému (daemon). Poté bylo třeba nastavit TCP port na kterém broker naslouchá na všech síťových rozhraních, zakázat anonymní uživatele a definovat soubor s uživatelskými přístupy, do konfiguračního souboru */etc/mosquitto/mosquitto.conf* jsem přidal následující direktivy:

allow_anonymous false
password_file /etc/mosquitto/pwfile
listener 1883

Poté už lze vytvořit hlavního uživatele *controller* a restartovat broker, aby se nová konfigurace načetla. Po zavolání prvního příkazu vyskočí výzva k zadání hesla:

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

Nakonec jsem otestoval funkčnost pomocí testovacího nástroje, k tomu jsou potřeba dvě otevřené konzole, na jedné zadám první příkaz a potvrdím heslem:

```
mosquitto_sub -d -u controller -P -t broker/test
mosquitto_pub -d -h 10.0.0.150 -u controller -P -t broker/test -m TEST
```

Když potom na druhé konzoli zapíšu do daného tématu nějakou zprávu (druhý příkaz, opět vyžaduje potvrzení heslem), uvidím ji v první konzoli. Tím je funkčnost ověřena a je možno případně definovat ostatní uživatele a jejich práva (např. do témat pro senzory mohou zapisovat jen skripty obsluhující senzory atd.).

Nakonec ještě zmíním, že MQTT broker u všech připojených zařízení periodicky ověřuje, zda jsou aktivní, posílá speciální zprávu PINGREQ a pokud se mu do určitého časového limitu nevrátí odpověď PINGRESP, označí zařízení jako odpojené. Na to se pak dají navázat různé automatizace, které ověřují funkčnost a integritu systému.

Wi-Fi

Pi Zero má vestavěný Wi-Fi modul, implicitně se ale chová jako Client. Aby se choval jako AP, je potřeba doinstalovat příslušné balíčky (AP daemon a DHCP/DNS server):

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

Pro správnou funkci potřebuje AP statickou IP adresu, ta se přiřazuje stejným způsobem jako u Ethernetového rozhraní – pomocí konfiguračního souboru */etc/network/interfaces*, do kterého jsem na konec přidal:

```
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 napřímo komunikovaly (komunikace probíhá kontrolovaně přes MQTT broker), je potřeba vypnout přeposílání paketů editací příslušné direktivy v souboru */etc/sysctl.conf*.

```
Net.ipv4.ip_forward=0
```

Pro Wi-Fi síť jsem nastavil automatické přidělování adres z DHCP pomocí direktiv v konfiguračním souboru DHCP serveru */etc/dnsmasq.conf*:

```
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*:

```
interface=wlan0
driver=nl80211
```

```
ssid=IoTGW
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
```

a */etc/default/hostapd*:

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

Po zapnutí/restartu příslušných služeb mám k dispozici funkční Wi-Fi AP poskytující síť IoTGW s přístupovým heslem s3cr3t:

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

Mezi sítěmi 10.0.0.0/24 (Wi-Fi) a 192.168.5.0/24 (Ethernet) by šlo selektivně povolovat a zakazovat provoz, pokud bych znovu zapnul IP forwarding v */etc/sysctl.conf* a patřičně nastavil firewall IPTables. Bylo by možné i navázat změny v pravidlech firewallu na automatizační pravidla ústředny, toho by šlo využít např. pro rodiče, kteří potřebují kontrolovat přístup svých dětí k internetu. protože se zde ale jedná o dedikovanou NoT síť, ke které by běžná zařízení neměla mít přístup, tuto funkcionalitu jsem neimplementoval.

Zigbee

Z dostupných Zigbee periferií jsem vybral sniffer CC2531, vzhledem k tomu, že má USB rozhraní přijatelnou, cenu a existuje pro něj open-source projekt nazvaný Zigbee2MQTT, který výborně řeší, jak název napovídá, právě komunikaci se Zigbee zařízeními prostřednictvím MQTT protokolu.

Firmware

Zařízení je z výroby určeno jako sniffer (dokáže odposlouchávat zigbee komunikaci v okolí a tím umožňuje ladit např. aplikační protokol daných zařízení), aby fungovalo jako Coordinator nebo Router, je nutné do něj nahrát jiný firmware. To lze provést několika způsoby, já se rozhodl použít samotné Raspberry Pi a programovací header na desce snifferu. Sniffer jsem připojil k Raspberry Pi podle

této tabulky (header má menší rozteč pinů než je standard, je třeba buď použít vhodnou redukci, nebo piny trochu ohnout do stran):

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)

Bylo třeba nainstalovat verzovací nástroj git, pomocí něj stáhnout a spustit program pro update firmware a tím otestovat spojení. Také bylo třeba nainstalovat knihovnu wiringpi, na které program závisí:

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

Pokud máme správý hardware a spojení je funkční, vrátí posední příkaz výsledek *ID = b524*. Po úspěšném ověření už stačilo stáhnout aktuální Coordinator firmware a nahrát ho do snifferu:

```
wget https://github.com/Koenkk/Z-Stack-firmware/raw/master/coordina
    tor/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 přikazem *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 už lze od snifferu odpojit programovací vodiče a připojit ho do jednoho z volných USB portů.

Zigbee2MQTT

Dalším krokem byla instalace aplikační části Zigbee2MQTT včetně prerekvizit, aplikace poběží pod uživatelem *admin*:

```
apt-get install -y nodejs git make g++ gcc npm
git clone https://github.com/Koenkk/zigbee2mqtt.git /opt/zigbee2mqtt
chown -R admin:admin /opt/zigbee2mqtt
cd /opt/zigbee2mqtt
npm ci
```

Uživateli *admin* bylo nutné udělit práva pro přístup k Zigbee zařízení, které se u Pi Zero mapuje na */dev/ttyACM0*. Zařízení je vedeno jako sériový port, uživatele tedy stačí přidat do skupiny dialup, které v Linuxu vlastní sériové porty:

usermod -a -G dialup admin

Konfigurační soubor aplikace je */opt/zigbee2mqtt/data/configuration.yaml*, tam jsem nastavil cestu ke zmíněnému Zigbee zařízení, Příkazem *npm start* v instalačním adresáři aplikace */opt/zigbee2mqtt* jsem komponentu Zigbee2MQTT spustil v testovacím provozu. Komponenta funguje, jak je vidět z výpisu událostí, který obsahuje úspěšné spárování 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":"de vice_announced","message":"announce","meta":{"frien dly_name":"0xccccccfffe9763af"}}'
Apr 13 00:16:30 IoTGW npm[644]: zigbee2mqtt:info 2020-04-13 00:16:30: Successfully interviewed '0xccccccfffe9763af', device has successfully been paired
Apr 13 00:16:30 IoTGW npm[644]: zigbee2mqtt:info 2020-04-13 00:16:30: Device '0xccccccfffe9763af' 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":{"frien dly_name":"0xccccccfffe9763af","model":"LED1624G9","ven dor":"IKEA","description":"TRADFRI LED bulb E14/E26/E27 600 lu men, dimmable, color, opal white","supported":true}}'

Daemonizace

Aplikace nemůže stále běžet v testovacím režimu. Je třeba, aby se spouštěla na pozadí při startu systému, běžela nepřetržitě a v případě výpadku se sama restartovala. Tyto procesy se v Linuxu označují jako *daemon* a jsou spravovány pomocí *Systemd*, 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 z balíčků jsou unit files vytvářeny automaticky, tady jsem to ale musel zajistit ručně. Vytvořil jsem soubor */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=admin

[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 vytvoření jsem následujícími příkazy načetl novu systemd unit, zapnul automatický start služby po startu OS a nakonec službu spustil, tím je Zigbee komponenta připravena k použití:

systemctl daemon-reload
systemctl enable zigbee2mqtt.service
systemctl start zigbee2mqtt.service

Bluetooth

Původní záměr byl využít vestavěnou Bluetooth kompatibilitu Pi Zero 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 Pi Zero má problém se souběžným provozem Wi-Fi AP a Bluetooth, kdy pokud běží oba zároveň, Wi-Fi je nestabilní a BLE má problém s párováním zařízení. Tato chyba není moc dobře popsaná, s největší pravděpodobností je problém přímo v hardware SoC Broadcom BCM2835, na kterém je Pi Zero postaveno, dokumentace k němu ale není veřejně dostupná. [12]

Rozhodl jsem se tedy pro Bluetooth konektivitu použít mikrokontroler ESP32 se softwarem OpenMQTTGateway, ten s Pi Zero komunikuje přes Wi-Fi a tvoří tak samostatný modul, který se zbytkem brány pevně spojuje jen napájecí USB konektor. Tento Bluetooth modul lze tedy podle potřeby odpojit a provozovat např. ve vedlejší místnosti, kde může být blíže k BLE koncovým prvkům. Pro ještě větší využití této modularity jsem k ESP32 připojil teplotní a vlhkostní senzor DHT22. Pokud je tedy modul v jiné místnosti než brána, budou k dispozici naměřené hodnoty z obou místností, pokud jsou naopak spojeny, lze hodnoty ze senzorů průměrovat z důvodu vyšší přesnosti a kontrolovat odchylku z důvodu detekce vadného senzoru. DHT22 je s ESP32 propojeno podle tabulky, mezi datový pin a 3,3 V je navíc nutné zapojit 10 kΩ pull-up resistor.

DHT22 pin	ESP32 pin
1 (VCC)	3,3 V
2 (Data)	GPIO 16
4 (GND)	GND

K naprogramování ESP32 je potřeba vývojové prostředí Arduino IDE, nejnovější verze je dostupná z *https://www.arduino.cc/en/Main/Software*. Po stažení a nainstalování jsem si přidal definice pro ESP32 ve správci desek. 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*. Dále jsem stáhnul samotnou OpenMQTTGateway ze stránky projektu *https://github.com/1technophile/OpenMQTTGateway/releases*, potřeba jsou kompletní zdrojové kódy a knihovny na kterých závisí. Knihovny je potřeba naimportovat přes IDE, nebo je stačí nakopírovat do adresáře *libraries* v kořenovém adresáři Arduina.

Zdrojové kódy jsem pak otevřel v Arduino IDE a funkcí verify zkontroloval, že jsou všechny závislosti v pořádku. Před nahráním zdrojových kódů do ESP32 bylo ale nejdřív potřeba aplikaci nakonfigurovat pomocí hlavičkového souboru User_config.h. Konfigurační parametry jsou zadávány jako různé konstrukce jazyka C++, jednotlivé direktivy ale mají intuitivní názvy a navíc je jejich funkce pečlivě okomentována. Aplikace je modulární, nepotřebné komponenty jsem vypnul zakomentováním (připsáním dvou lomítek "//" na začátek řádku, což je v C++ syntaxe pro komentář). Jediné dva moduly, které využívám jsou *ZgatewayBT* (Bluetooth) a *ZsensorDHT* (DHT senzor). Dále bylo třeba nastavit název zařízení, IP adresu, SSID a heslo Wi-Fi AP běžícího na Pi Zero a parametry 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, s mou konfigurací tedy ESP zapisuje do *NoTGW/ESP*.

Po konfiguraci stačilo už jen v IDE vybrat správný typ desky (ESP Dev Module) a 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. V případě problémů lze zapnout logovací modul, který vypisuje chyby a informace o provozu na sériový port (terminál je potřeba nastavit na rychlost 115200 baudů).

Senzory

Vzhledem k výběru řídícího prvku je nejvhodnější použít digitální senzory, 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 a vlhkosti popsaných výše v sekci senzory je DHT22 levnější a rozdíl v přesnosti oproti SHT71 není tak velký, aby na něm v této aplikaci záleželo. K Pi Zero jsem senzor připojil podle tabulky, stejně jako u DHT22 připojeného k ESP32 je nutné zapojit 10 kΩ pull-up resistor mezi datový pin a 3,3 V.

DHT22 pin	Raspberry Pi pin
1 (VCC)	3,3 V
2 (Data)	GPIO 4
4 (GND)	GND

Z hlediska software stačí jednoduchý program, který bude periodicky číst data ze senzoru a výslednou hodnotu odesílat jako MQTT zprávu. Samostatný program s touto funkcionalitou jsem nenašel (pouze jako součást rozsáhlejšího software), rozhodl jsem se tedy pro obsluhu senzoru (stejně jako ostatních třech, viz níže) vytvořit jednoduchý skript. Zvolil jsem jazyk Python, který má pro obsluhu DHT22 dostupnou knihovnu. Nejprve jsem provedl instalaci potřebných balíčků:

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

Celý skript lze najít v příloze práce, nejprve pomocí knihovny *configparser* načte nastavený interval dotazování senzoru z globálního konfiguračního souboru brány */etc/notgw.conf*. Dále používá knihovnu *paho mqtt client* pro připojení k MQTT brokeru, a nakonec knihovnu *Adafruit_DHT* pro čtení dat ze senzoru. Data jsou v nastaveném intervalu čtena a odesílána 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, systemd unit file lze také najít v příloze práce, důležité je v sekci *Service* spouštět skript správným interpretem:

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. DS18B20 se dodává buď jako součástka v pouzdře TO92, nebo jako sonda v hermeticky uzavřeném kovovém pouzdře včetně přívodního kabelu, a právě v tomto provedení ho lze použít stejně jako venkovní teplotní čidla klasických domácích meteostanic. 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 jsem nepoužil parazitní mód, ale separátní datový vodič, senzor jsem opět připojil dle tabulky a mezi datový vodič a 3,3 V vložil specifikací předepsaný 1,5 kΩ pull-up rezistor.

DS18B20 vývod	Raspberry Pi pin
červený (VCC)	3,3 V
žlutý (Data)	GPIO 17
černý (GND)	GND

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 (GPIO 4, 17 a 27) a nastavení se provádí v souboru */boot/config.txt*. Sběrnici na vybraném pinu 17 jsem aktivoval přidáním záznamu:

dtoverlay=w1-gpio,gpiopin=17

Po restartu Pi Zero je senzor připraven k použití, v adresáři /sys/bus/w1/devices přibyl podadresář s ID sběrnice a v něm 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 jsem, stejně jako u DHT22, vytvořil obslužný skript a systemd unit file, hodnota ze senzoru se získává pomocí systémových volání (knihovna os), celý obslužný skript včetně systemd unit file je opět k dispozici 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 Pi zero komunikuje pomocí čtyřvodičové sběrnice I²C (inter-integrated circuit), pátý pin na modulu slouží k výběru jedné ze čtyř adres pomocí spojení tohoto pinu s jedním ze čtyř ostatních. Celkem lze tedy na jedné sběrnici provozovat čtyři tyto senzory, Pi zero disponuje pouze jednou I²C sběrnicí. Adresní pin jsem tedy spojil se zemí a zbývající čtyři připojil podle tabulky:

BH1750 pin	Raspberry Pi pin
GND (1)	GND
SDA (3)	GPIO 2
SCL (4)	GPIO 3
VCC (5)	3,3 V

I²C sběrnici bylo potřeba zapnout pomocí příkazu *raspi-config*, kde jsem v nabídce *interfacing options* přepnul položku *I²C* na *enabled*. Změna se projeví až po restartu systému. Pro obslužný Python skript jsem potřeboval doinstalovat knihovny:

apt-get install python-smbus i2c-tools

Celý skript, který využívá knihovnu *smbus*, je také dostupný v příloze této práce, stejně jako systemd unit file pro tento senzor. 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 je nejlevnější způsob měření koncentrace CO_2. Senzor obsahuje topný rezistor, který ohřívá vrstvičku SnO_2 na optimální teplotu, při které je nejcitlivější na měřené plyny. 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.

Připojení

Na rozdíl od ostatních použitých senzorů je napájecí napětí MQ135 5 V. Vzhledem k tomu, že Pi Zero nemá analogové vstupní piny jako např. Arduino, byl jsem nucen použít externí ADC, konkrétně 16-bitový ADS1115. Tento převodník

může pracovat na 3,3 V i 5 V, podle napájecího napětí se ale změní i logické úrovně komunikační sběrnice I^2C. Abych se vyhnul převádění logických úrovní u této sběrnice, převedl jsem pomocí napěťového děliče 5 V na 3.3V už na výstupu z MQ135. Připojení MQ135 k ADC je velmi jednoduché – výstup z napěťového děliče stačí připojit do jednoho ze čtyř vstupů ADC. Dále jsem nastavil adresu, stejně jako u BH1750 připojením ADDR pinu (5) k zemi, adresa musí být jiná než u BH1750, protože jsou na stejné sběrnici. Propojení Pi Zero s ADC jsem pak realizoval podle tabulky:

ADS1115 pin	Raspberry Pi pin
VDD (1)	3,3 V
GND (2)	GND
SCL (3)	GPIO 3
SDA (4)	GPIO 2

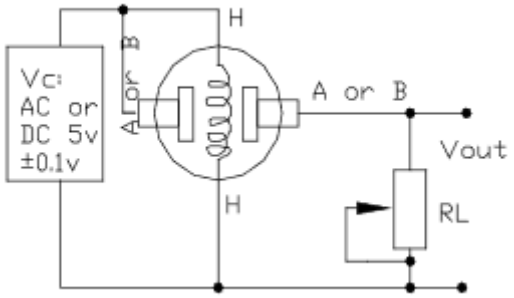
Příkazem *i2cdetect 1* jsem poté ověřil správnost připojení a začal tvořit obslužný skript. Pro ADS1115 je dostupná python knihovna od Adafruit, která velmi usnadňuje práci s tímto ADC:

pip3 install adafruit-circuitpython-ads1x15

S použitím této knihovny jsem vytvořil skript, který pouze vypisoval data z převodníku do konzole. Tím jsem měl jednodušší část za sebou, senzor bylo poté třeba zkalibrovat.

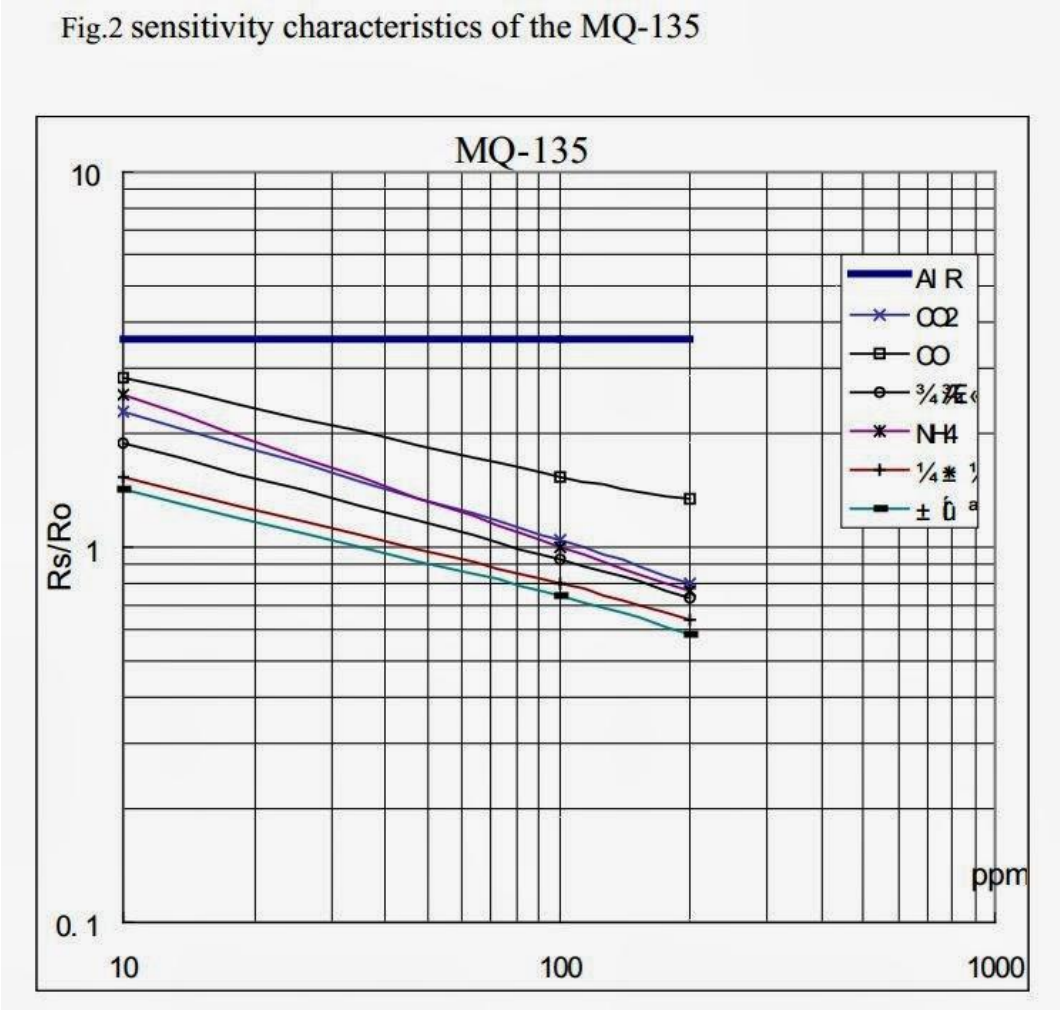
Kalibrace

V katalogovém listu MQ135 je základní schéma zapojení (obrázek 2) a graf ukazující závislost naměřené hodnoty na poměru odporů R_S/R_0 (obrázek 3).



Obrázek 2 - 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 konstantní, používá se jako referenční hodnota a je pro každý kus senzoru trochu jiný.



Obrázek 3- Závislost poměru Rs/R₀ 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 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. Jelikož nepoužívám samotný senzor, ale modul FC-22-I, je tento rezistor již osazen a jeho hodnota je 4,7 kΩ. ADS1115 poskytuje hodnotu napětí na svém vstupu U_{ADC}, výstup senzoru je k ADC ale připojen přes dělič napětí (s poměrem 3/5, kvůli výše zmíněnému rozdílu logických úrovní), takže hodnota výstupního napětí senzoru bude:

$$U_{OUT} = \frac{U_{ADC}}{3/5}$$

Teď už znám okamžitou hodnotu R_S:

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

A mohu spočítat 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 venku v Praze 10 Malešice). Referenční odpor mého senzoru (průměr z 50 naměřených hodnot) je tedy

$$R_0 = 1836{,}8304365855317$$

Výpočet ppm už pak stačilo jen zahrnou do obslužného 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 jsem testoval a měřené hodnoty porovnával s DS-CO2-20, který má udávanou chybovost ±50 ppm. Po zahřátí na provozní teplotu, které trvá cca 5 minut jsou naměřené hodnoty překvapivě přesné, největší odchylka mezi senzory byla 73 ppm. V děliči napětí jsem použil precizní rezistory s 0.1% tolerancí a ADC s relativně vysokým rozlišením 16 bitů. Podle zkušeností ostatních lidí je na tom MQ135 s přesností hůře (s odchylkami až kolem 150ppm), takže je možné, že velmi záleží na konkrétním kusu.

Webové rozhraní

Přestože veškerá výměna dat probíhá po MQTT, pro konfiguraci to není příliš vhodný protokol, i když používané projekty Zigbee2MQTT a OpenMQTTGateway ho tak využívají. Implementoval jsem tedy jednoduché webové rozhraní, které uživateli poskytne přehled o jednotlivých zařízeních a dá mu možnost nastavení.

Rozhraní se skládá ze tří hlavních komponent. První je správce daemonů realizovaný open-source projektem *SysdWeb*, díky kterému lze zapínat a vypínat jednotlivé senzory a komponenty, případně je restartovat při problémech. Druhou částí je monitor systémových prostředků, opět se jedná o otevřený software – nástroj *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. Zbytek je tvořen kombinací statických stránek a PHP skriptů, které jsem vytvořil pro

editaci konfiguračního souboru brány, vypnutí systému a jako rozcestník spojující vše dohromady.

Nejprve bylo nutné nainstalovat webový server *Apache* společně s dalšími potřebnými balíčky a zapnout moduly nutné pro funkci reverzního proxy (viz níže):

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

Následovala instalace SysdWeb. Zdrojové kódu projektu jsou k dispozici na githubu jeho vývojáře https://github.com/ogarcia/sysdweb, postup je podobný instalaci Zigbee2MQTT, virtuální prostředí (venv), které při instalaci vytvářím zajišťuje 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 a přístupová práva. Konfigurace služeb je velmi intuitivní, ukázka záznamu pro senzor DHT22:

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

Samotný SysdWeb běží jako daemon, nastaví se sám při instalaci, po editaci konfiguračního souboru ho stačí spustit pomocí *systemctl start sysdweb*. Komponentu Glances není třeba nastavovat, vyhovuje v defaultním stavu. Nyní tedy bylo možné přistoupit ke konfiguraci Apache. Jedná se o plnohodnotný webserver, který se běžně využívá pro hostování rozsáhlých webových prezentací, konfigurace je tedy trochu složitější a distribuovaná do více souborů. Pro zjednodušení jsem hlavní konfiguraci umístil do souboru pro defaultní virtualhost (webserver může obsluhovat více domén na jedné IP adrese, každý virtualhost reprezentuje jednu doménu. Defaultní virtualhost obsluhuje vše ostatní, co není definováno, včetně samotné IP adresy).

Cesta k souboru je */etc/apache2/sites-enabled/000-default.conf* tři nuly na začátku zajišťují, že soubor bude při parsování konfigurace první v pořadí. Konfigurační soubor je dostupný v příloze, direktiva *DocumentRoot* na začátku určuje kořenový adresář, kam je nutné uložit vše, co má být na webu dostupné. Zbytek konfigurace je rozdělen do sekcí, první pod hlavičkou *<Directory "/var/www/html">* nastavuje přístup k webu jen po přihlášení, další dvě sekce

s hlavičkami *<Location…* konfigurují tzv. reverzní proxy, to znamená, že jakýkoli příchozí HTTP požadavek je beze změny přeposlán na nastavený port. Díky tomu není nutné, aby porty pro Glances a SysdWeb byly viditelné zvenku a zároveň je umožněno schovat všechny komponenty za jednu společnou autentizaci. Pro nastavenou autentizaci je nutno vytvořit seznam uživatelů, tento příkaz přidá nového uživatele *admin* (po spuštění se zeptá na heslo). Je třeba zadat cestu k souboru, který je specifikován v *000-default.conf*, pokud zatím neexistuje, příkaz ho vytvoří:

htpasswd -c /etc/apache2/htpasswd admin

Po dokončení nastavení jsem do kořenu webu umístil potřebné soubory:

- `index.html` – index webu (soubor, který se zobrazí pokud je požadovaná url bez parametrů – např. www.example.com nebo IP adresa). Zde je to 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

Všechny soubory jsou dostupné v příloze. 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 koretkní vypnutí systému. Otevřel jsem tedy příkazem *visudo* editor konfigurace programu *sudo*, který umožňuje selektivně udělovat běžným uživatelům práva pro spouštění příkazů jako root. Na konec souboru jsem přidal záznam:

www-data ALL = NOPASSWD: /sbin/halt

Po restartu apache pak bylo webové rozhraní připraveno k použití:

systemctl restart apache2

Nakonec dodám, že jakékoli nastavení je možné provádět přes SSH, stejně jako instalaci, k tomu je ale třeba se orientovat na příkazové řádce. SSH nikdy není potřeba pro standartní činnosti jako např. přidání koncového prvku.

Testy a srovnání

Samostatné testy

Brána byla v provozu nepřetržitě po dobu více než dvou měsíců, po které byla využívána pro připojení senzoru Xiaomi Miija (BLE), RGB žárovky Ikea Tradfri (Zigbee) a dvou ESP8266 ovládajících relé (Wi-Fi) k Home Assistant ústředně. Výkon Pi Zero je dostatečný, využití RAM bylo okolo 40 % a využití CPU kolem 10–20 %. Komponenty neměly nejmenší problém se spoluprací. Jediný problém, na který jsem narazil byla vadná SD karta, která způsobovala extrémně pomalé zápisy. Ty se projevovaly „zatuhnutím" systému na několik sekund, zejména při operacích jako editace konfigurace pomocí web rozhraní. Vyřešil jsem to výměnou SD karty za novou (třídy A1).

MQTT

Pro diagnostiku MQTT jsou k dispozici příkazy *mosquitto_ sub* a *mosquitto_ pub* pro čtení, respektive zápis libovolných témat. Takto jde otestovat v podstatě veškerou funkcionalitu brány, jelikož všechny komponenty komunikují po MQTT. Pro test využívám uživatele *controller*, který má přístup do všech témat.

Senzory

MQTT témata umí wildcards, tedy speciální znaky, jejichž významem je „jakékoli téma" a to buď jednoúrovňově „*+*" nebo víceúrovňově „#". Pro čtení dat ze všech senzorů tak lze použít

```
mosquitto_sub -d -h 192.168.10.250 -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 otestování se nejřív přihlásím k odběru všech Zigbee2MQTT témat:

mosquitto_sub -d -h 192.168.10.250 -u controller -P s3cr3t -t NoTGW/Zigbee/#

a poté do nich budu zapisovat a pozorovat výsledky. 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 192.168.10.250 -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.* Lze si také vypsat seznam připojených zařízení:

mosquitto_pub -d -h 192.168.10.250 -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":"0xcccccfffe9763af","type":"Router","networkAddress":37658,"model":"LED1624G9","vendor":"IKEA","description":"TRADFRI LED bulb E14/E26/E27 600 lumen, dimmable, color, opal white","friendly_name":"0xcccccfffe9763af","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}]

A teď, když znám ID žárovky, mohu ji třeba rozsvítit 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

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 zařízení ignoruje. Po připojení do přiřazeného tématu lze vidět, jak se do něj periodicky zapisují data ze všech dostupných zařízení:

```
mosquitto_sub -d -h 192.168.10.250 -u controller -P s3cr3t -t NoTGW/Zigbee/#
```

Zde je to senzor DHT22 připojený přímo k bráně 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/ESP/LWT', ... (6 bytes))
online
Client mosqsub/20882-MyshZEN received PUBLISH (d0, q0, r1, m0,
'NoTGW/ESP/version', ... (6 bytes))
v0.9.3
Client mosqsub/18156-MyshZEN received PUBLISH (d0, q0, r0, m0, 'IoTGW/ESP/DHTtoMQTT/dht1', ... (20 bytes))
{"hum":40,"temp":28}
Client mosqsub/18156-MyshZEN received PUBLISH (d0, q0, r0, m0,
'IoTGW/ESP/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/ESP/BTtoMQTT/4C65A8D2791A', ... (23 bytes))
{"tem":24.4,"hum":39.5}
Client mosqsub/21147-MyshZEN received PUBLISH (d0, q0, r0, m0,
'IoTGW/ESP/BTtoMQTT/4C65A8D2791A', ... (11 bytes))
{"batt":87}
```

Wi-Fi

Zde záleží výhradně na daném koncovém zařízení a jeho nastavení. Komunikaci s brokerem a integraci s ústřednou je třeba řešit individuálně. Výhodou brokeru integrovaného v bráně je, že mezi Wi-Fi sítí obsahující koncové prvky a sítí, ve

které je ústředna se přímo nepřenáší žádná data, veškerá komunikace je prostřednictvím brány (brokeru). Uvedu zde příklad komunikace s připojenými ESP8266:

mosquitto_sub -d -h 192.168.10.250 -u controller -P s3cr3t -t NoTGW/wifi/#

Tyto MCU ovládají několik připojených relé, do MQTT zapisují pouze v reakci na události, např. zapnout jedno z relé lze takto:

mosquitto_pub -d -h 192.168.10.250 -u controller -P s3cr3t -t NoTGW/wifi/RelCtrl2 -m '{"state":"ON","RelayID":1}'

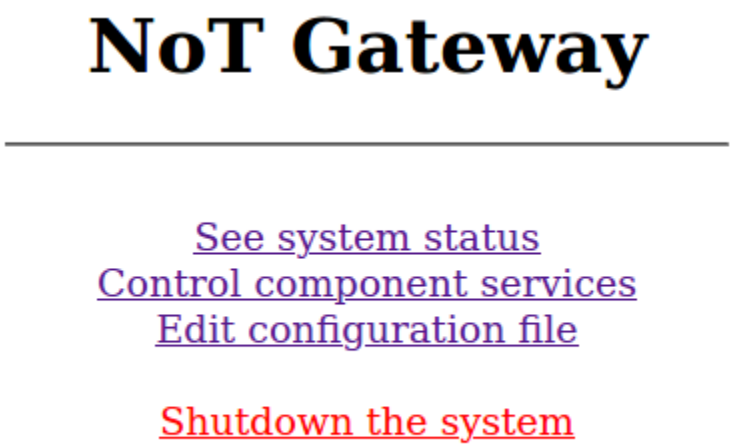
Odpovědí bude report o stavu všech relé:

Client mosqsub/13496-MyshZEN received PUBLISH (d0, q0, r0, m0, 'NoTGW/wifi/RelCtrl2', ... (78 bytes))
{"state":"ON","RelayID":1,"state":"OFF","RelayID":2,"state":"OFF","RelayID":3}

Report si lze rovněž vyžádat zápisem prázdné zprávy do tématu příslušného MCU.

Web

Webové rozhraní je velice jednoduché, na obrázku 4 je rozcestník, který se zobrazí po zadání IP adresy brány do adresního řádku webového prohlížeče a přihlášení se.



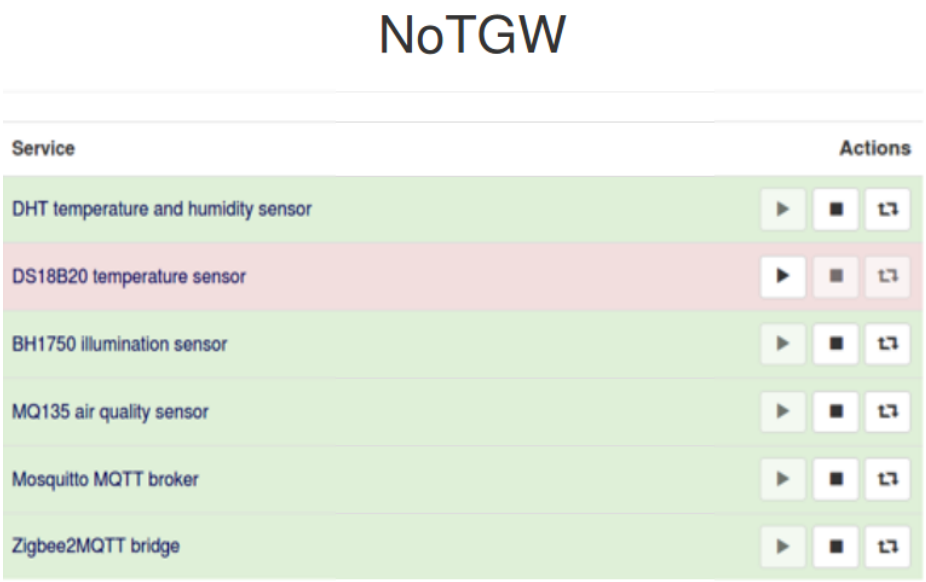
Obrázek 4 - Rozcestník webového rozhraní

První odkaz zobrazí systémový monitoring Glances, jeho UI je nejlépe popsáno na stránkách vývojářů https://nicolargo.github.io/glances/.

Druhý odkaz otevře hlavní konfigurační soubor brány */etc/NoTGW.conf* v jednoduchém textovém editoru se dvěma tlačítky pro potvrzení/zrušení provedených změn. Je použita syntaxe .ini konfiguračních souborů kvůli jednoduchému

importu Python knihovnou *configparser*. Vzhledem k tomu, že hlavní komponenty se nastavují pomocí MQTT, jsou v souboru aktuálně pouze direktivy pro nastavení dotazovacích intervalů senzorů.

Třetí odkaz otevře nástroj SysdWeb pro správu daemonů, obsluhujících komponenty brány. Na obrázku 5 je příklad vypnutí jedné z komponent



Obrázek 5 - Správa komponent

Poslední odkaz bránu vypne a vrátí informaci, zda je možné ji bezpečně odpojit od napájení.

Integrace

Brána je určena jako rozšíření open-source ústředen. Zde testuji její integraci se dvěma z nich, které jsou v současnosti nejpoužívanější. Vzhledem k použitému protokolu MQTT a logice komunikace je brána do jisté míry transparentní a integrace záleží na koncových prvcích a jaké funkce od nich budou požadovány. Integrace samotné brány a jejího MQTT brokeru je velice jednoduchá, jelikož MQTT je v této oblasti nejpoužívanějším komunikačním protokolem. Předpokládám, že brána je připojena pomocí Ethernetu do stejné sítě jako ústředna a není zapnuto žádné filtrování portů.

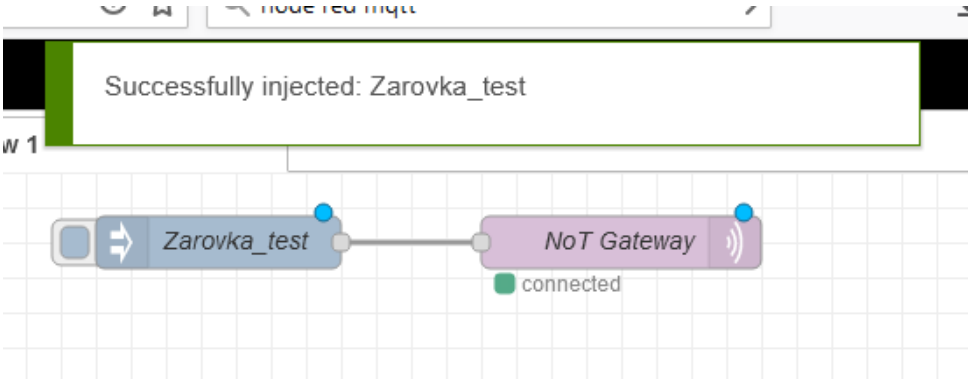
Node-RED

Nejprve je nutné připojit se k MQTT brokeru. Node-Red v základní instalaci obsahuje node MQTT in a MQTT out. Pro test použiji MQTT out a budu ovládat Zigbee žárovku (viz sekce samostatné testy).

V editaci node v kolonce „server“ zvolím „add new mqtt broker“. V záložce „connection“ zadám IP ethernetového rozhraní brány 192.168.10.250 a defaultní

port 1883. V záložce „secutity“ pak jméno testovacího uživatele *controller* a heslo. Po uložení nastavení a stisknutí tlačítka „Deploy“ by se měla node přepnout do stavu „connected“

Pomocí node Inject odešlu testovací zprávu pro rozsvícení žárovky modrou barvou *{"state": "ON","brightness":255,"color": { "rgb": "0,0,255" } }* do příslušného tématu *NoTGW/Zigbee/0xccccccfffe9763af/set*. V editaci inject node zvolím v kolonce „payload“ typ „string“ a do něj zadám zprávu. Do kolonky topic zadám téma a uložím. Opět provedu Deploy a ručně spustím inject. Jak je vidět na obrázku 6, vše funguje v pořádku.

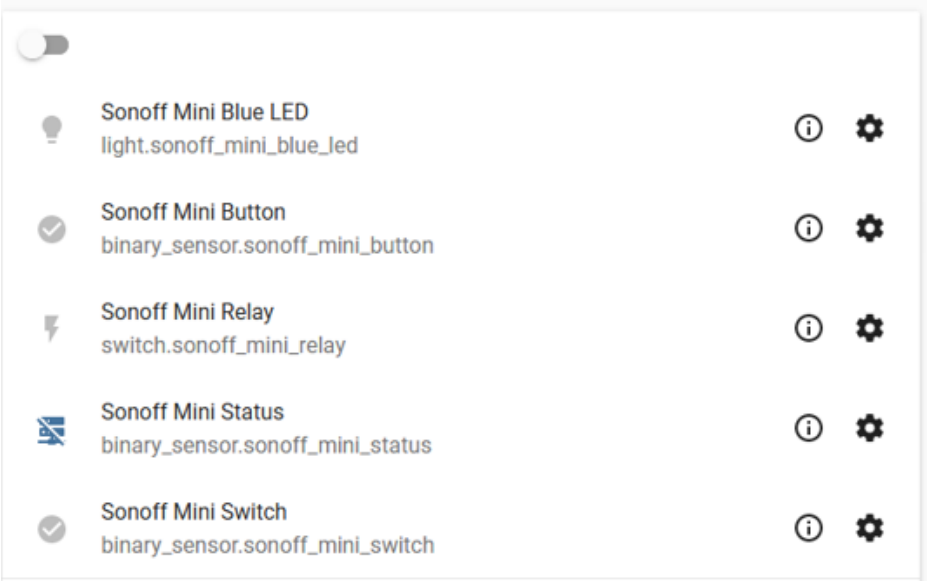


Obrázek 6 - Test Node-RED integrace

Home Assistant

Ve starších verzích bylo třeba MQTT broker přidat v textovém konfiguračním souboru, v současné verzi už je dostupná integrace pomocí webového rozhraní. Nejdříve je třeba přidat tuto integraci. V hlavním menu Home Assistant otevřu sekci „configuration“ a v ní menu „integrations“.

Po kliknutí na symbol „+“ v pravém dolním rohu najdeme v nabídce, která se objeví, integraci „mqtt“. V konfiguračním okně jsem opět zadal IP brokeru (ethernetového rozhraní brány), port 1883, a přihlašovací údaje uživatele *controller*. Pro automatickou integraci koncových zařízení je vhodné zaškrtnout „Enable discovery“, bohužel s žádným zařízením připojeným pomocí brány mi autodiscovery nefungovalo. Šlo by sice přidat zařízení manuálně, ale jednodušší bylo připojit k Wi-Fi brány zařízení Sonoff Mini s alternativním firmware Tasmota. Ten byl nalezen a integrován téměř okamžitě, na obrázku 7 jsou vidět entity, které poskytuje. Podle informací na fóru Home Assistant by podpora pro autodiscovery skrz Zigbee2MQTT by měla být přidána v nejbližší době.



Obrázek 7- Integrace Sonoff Mini do Home Assistant ústředny pomocí brány

Závěr

Svým určením jako univerzální rozhraní k open source ústředně je brána dost specifické zařízení a těžko se hledá, s čím ji porovnat. Komerční produkty se stejnou funkcionalitou většinou integrují i ústřednu samotnou (např. Amaxon Alexa), případně jsou u nich stejné funkce realizovány více samostatnými prvky (např. dedikovaná Zigbee brána s připojenými Zigbee senzory).

Dle mého názoru se tedy cíl práce podařilo splnit, vytvořená brána je relativně unikátní zařízení, které nejvíc vynikne za předpokladu, že v domácnosti již je nějaké další zařízení pro provoz ústředny (např. NAS server). V takovém případě se jedná o velmi levnou a efektivní volbu pro základ smart home systému, která dokáže funkcemi pokrýt více jednoúčelových zařízení.

Literatura

[1] Deschamps-Sonsino Alexandra: *Smarter Homes: How Technology Will Change Your Home Life (Design Thinking)*, 2018, ISBN 9781484233627
[2] Encyklopedie energetiky, [online], dostupné z: https://zdroje.elektrika.cz/book/cez-elektrina-encyklopedie-energetiky/
[3] *Historie spotřební elektroniky*, [online], 2011, dostupné z: https://www.tvfreak.cz/historie-spotrebni-elektroniky-1-dil-jaky-byl-vyvoj/4271
[4] Radim Chlad: *Historie Internetu v České republice*, [online], dostupné z: https://www.fi.muni.cz/usr/jkucera/pv109/2000/xchlad.htm
[5] *Industrial Revolution in Comparative Perspective*, 2000, 9780894649905

[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] *Milestones in the Bluetooth advance*, [online], 2004, dostupné z: https://web.archive.org/web/20040620150507/http://www.ericsson.com/bluetooth/companyove/history-bl/
[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 9781647710194
[16] Evi Nemeth, Garth Snyder: *UNIX and Linux System Administration Handbook,* 2018, ISBN 978-0134277554

Slovníček pojmů a zkratek

ACS – Access Control Systém, systém řízení přístupu do budovy
API – Application Programming Interface, využívané jinými aplikacemi
ASCII – standard kódování znaků
CPU – Central Processing Unit, procesor
DAC – Digital to Analog Converter, digitálně-analogový převodník
ADC – Analog to Digital Converter, analogově-digitální převodník
DHCP – protokol pro automatickou konfiguraci síťových uzlů
DIY – Do It Yourself, „udělej si sám“, návrh a konstrukce zařízení svépomocí
DSL – Protokol pro vysokorychlostní komunikaci po telefonních kabelech

EPS – Elektronický Protipožární Systém
EZS – Elektronický Zabezpečovací Systém
GPIO – General-Purpose Input/Output, univerzální rozhraní mikrokontrolerů
GPU – Graphics Processing Unit, grafický procesor
GUI – grafické uživatelské rozhraní
HTPS – Home Theatre PC, počítač určen pro nahrazení spotřební elektroniky
HVAC – Heating, Ventilation, Air Conditioning, vytápění, větrání a klimatizace
IO – Integrovaný obvod
IT – Informační technologie
MCU – MicroController Unit
NAS – Network-Attached Storage
OEM – Original Equipment Manufacturer
off-the-shelf – výrobky dostupné běžně v obchodě
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
USB OTG – USB On the GO, zařízení se může chovat jako host i jako slave
VoIP – Voice over IP, provoz telefonních služeb na počítačové sítí
VR – Virtuální Realita

Obsah přiloženého CD

Sensors – skripty a systemd unit files pro obsluhu senzorů
Webroot – obsah kořene webového rozhraní
Apache – konfigurační soubor pro Apache2