ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta elektrotechnická
Katedra výkonové elektroniky a strojů

DIPLOMOVÁ PRÁCE

Vývoj HW komponent „smart Home automatizace“

Autor práce: **Tomáš Tunka**
Vedoucí práce: **Ing. Jan Molnár, Ph.D.**

2023

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Tomáš TUNKA**
Osobní číslo:	**E20N0045P**
Studijní program:	**N0713A060013 Výkonové systémy a elektroenergetika**
Specializace:	**Výkonové elektronické technologie a pohony**
Téma práce:	**Vývoj HW komponent „smart Home automatizace“**
Zadávající katedra:	**Katedra výkonové elektroniky a strojů**

Zásady pro vypracování

1. Proveďte obecný rozbor problematiky domácí automatizace.
2. Vytvořte přehled použitelných SW a HW prostředků.
3. Zvolte vhodnou SW a HW platformu pro realizaci demo stanoviště a realizaci vybrané komponenty „smart home“.
4. Formou „bastlení“ otestujte připojení různých typů senzorů a dalších periférií (snímače teploty, vlhkosti, tlaku, potenciometr, inkrementální kodér, display, expandéry IO, atd.)
5. Vytvořte technickou specifikaci zvoleného typu zřízení. Realizujte kompletní HW (krabička, DPS, firmware, dokumentace, atd.)
6. Vytvořte demo stanoviště a integrujte vytvořené zařízení do systému řízení.

Rozsah diplomové práce: **40 – 60**
Rozsah grafických prací: **dle doporučení vedoucího**
Forma zpracování diplomové práce: **elektronická**

Seznam doporučené literatury:

Student si vhodnou literaturu vyhledá v dostupných pramenech podle doporučení vedoucího práce.

Vedoucí diplomové práce: **Ing. Jan Molnár, Ph.D.**
Research and Innovation Centre for Electrical Engineering

Oponent diplomové práce: **Ing. Martin Sirový, Ph.D.**
Research and Innovation Centre for Electrical Engineering

Datum zadání diplomové práce: **7. října 2022**
Termín odevzdání diplomové práce: **26. května 2023**

Západočeská univerzita v Plzni * Fakulta elektrotechnická *

L.S.

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

Prof. Ing. Václav Kůs, CSc.
vedoucí katedry

V Plzni dne 7. října 2022

Abstrakt

Teoretická část obsahuje stručný popis smart technologií. Jsou zde popsány jejich přínosy, hrozby a možnosti využívání v různých odvětví. Důraz se pak klade především na využívání chytrých technologií v domácnostech. V souvislosti s chytrou domácností je zde představen i Open-Source systém Home Assistant a jeho doplněk ESP Home, který umožňuje do tohoto systému integrovat uživatelem vytvořené chytré komponenty.

Praktická část se zabývá testováním připojitelnosti různých typů periférií (senzorů, displejů, sběrnic, IO vstupů a výstupů atd.) a následným vývojem vlastních chytrých komponent. Tyto chytré komponenty jsou pak demonstrovány na vytvořeném testovacím stanovišti se systémem Home Assistant.

Klíčová slova

Chytrá domácnost, Programování v jazyce C++, Home Assistant, ESP32, Arduino, Sonoff

Abstract

The diploma thesis deals with the development of smart devices designed for the Home Assistant system.

The theoretical part contains a brief description of smart technologies. Their benefits, threats and possibilities of use in various industries are described here. The emphasis is then mainly on the use of smart technologies in households. In connection with the smart home, the Open-Source system Home Assistant and its addon ESP Home, which allows the integration of user-created smart components into this system, are also presented here.

The practical part deals with testing the connectivity of various types of peripherals (sensors, displays, buses, IO inputs and outputs, etc.) and the subsequent development of own smart components. These smart components are then demonstrated on the created testing station with the Home Assistant system.

Key Words

Smart Home, Programming in C++ language, Home Assistant, ESP32, Arduino, Sonoff

Poděkování

Chtěl bych tímto poděkovat Ing. Bedřichu Bednářovi Ph.D. a komunitě MakerSpace FEL za výpomoc při tvorbě dílů na stanoviště.

Obsah

1. Úvod

Chytré technologie jsou v současné době využívány stále častěji, a to v mnoha odvětvích včetně domácností (tzv. chytré domácnosti). Zařízení do chytrých domácností ale nejsou vždy cenově dostupná zvlášť pokud se jedná o vytvoření rozsáhlého smart systému od profesionální firmy. Tato práce má tedy ukázat možnosti vytvoření vlastní chytré domácnosti a jejích komponentů za nižší pořizovací náklady, než jaké by byly vynaloženy na koupi hotového smart systému.

V úvodu se práce zabývá zejména představením open-source systému Home Assistant, jeho přínosů a funkcí. Důraz se klade i na jeho doplněk ESP Home umožňující vytvářet vlastní chytré komponenty a integrovat je do Home Assistantu.

Další součástí práce bylo vyzkoušet různé typy periférií (senzorů, displejů, sběrnic, IO vstupů a výstupů atd.) a otestovat jejich propojitelnost se systémem Home Asisstant pomocí doplňku ESP Home. Z nabytých poznatků jsem pak navrhl a vytvořil dvě vybrané smart komponenty, které jsem integroval do Home Assistantu poskytující softwarovou podporu celé chytré domácnosti.

Vytvořená chytrá zařízení společně se systémem Home Assistant (nainstalovaném na mikropočítači Raspberry Pi 4) jsou demonstrovány na vytvořeném testovacím stanovišti společně s několika dalšími chytrými zařízeními od výrobců Sonoff a Denver.

2. Co je to smart systém

Smart systémy jsou technologicky vyspělé systémy, které vnímají a reagují na svět kolem sebe. Jsou vybaveny komponenty a funkcemi pro snímání a řízení, které umožňují shromažďovat informace z okolí a autonomně se rozhodovat. Právě autonomní řízení je znakem smart systémů a zařízení. Moderní smart systémy mají i umělou inteligenci, která jim dává schopnost učit se a upravovat procesy podle chování uživatele. Dalším důležitým kritériem je komunikace mezi jednotlivými chytrými zařízeními (obvykle pomocí internetu). Zároveň by však měla být schopná pracovat i samostatně, a proto každé zařízení vyžaduje vlastní procesor, který jim umožňuje jejich autonomní provoz. Společně tak vytvářejí celistvý inteligentní systém, který může obsahovat různé komponenty velice odlišných technologií a účelů.

Připojení všech zařízení k internetu má i výhodu možnosti nepřetržitého sledování a nastavování prováděných procesů jednotlivých zařízení i systému jako celku. Je tak možné i hlídat technický stav strojů a lépe tak naplánovat čas údržby. Právě ke sledování v reálném čase pomocí smartphonů se dnes využívá velká část chytrých zařízení ať už se jedná o sledování teploty v místnosti nebo bezpečnostních systémů [1,2,11].

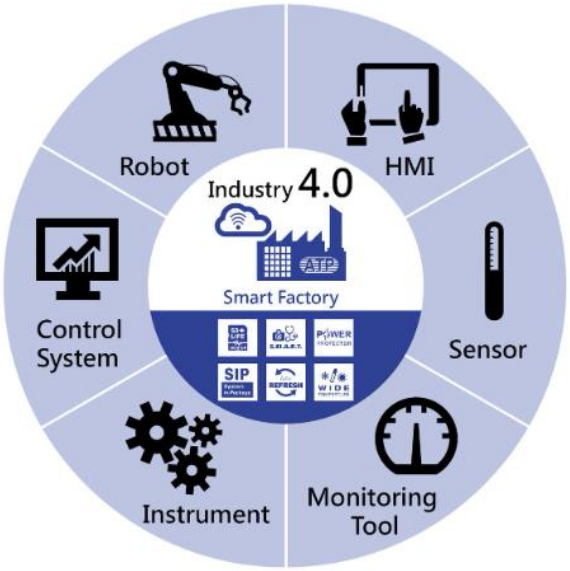
3. Smart systémy v průmyslu a průmysl 4.0

V současné době čím dál tím více průmyslových podniků instalují ve svých závodech smart systémy za účelem zvýšit svou efektivitu a spolehlivost. Ty dokážou například sami řídit a optimalizovat spotřebu energie, řídit proces výroby nebo předvídat poruchy různých zařízení a naplánovat podle nich údržbu. Příkladem můžou být senzory a měřící zařízení které si sami nastaví vhodný rozsah, senzory určené pro diagnostiku schopné naplánovat údržbu, nebo chytré vytápěcí systémy s rekuperací tepelné energie [1,2].

V souvislosti s využitím těchto smart systémů v průmyslu se pak často mluví o takzvaném průmyslu 4.0 a chytrých továrnách. Jedná se o kompletní digitalizaci a robotizaci výrobního procesu. Jednotlivá pracovní stanoviště jsou integrována do jedné sítě, kde mezi sebou jednotlivé systémy komunikují a řídí výrobní procesy bez zásahu člověka. Tato stanoviště musí být tedy obsahovat smart komponenty, které sledují a řídí jednotlivé úkony a dokážou výrobní proces přizpůsobit vnějším vlivům či poruchám. To může vést až k chytrým továrnám, které si podle požadavků zákazníka sami naplánují výrobní proces. To poskytuje vysokou flexibilitu, neboť není nutné výrobní linku s každou zakázkou přestavovat či jinak upravovat, ale místo toho je továrna tvořena samostatnými buňkami, které mezi sebou komunikují a továrna si sama vytvoří trasy mezi výrobními stanovišti pomocí dopravníků, nebo upraví úkony jednotlivých strojů. Toto uspořádání poměrně nahrává využívání CNC strojů a technologií 3D tisku. Tento přístup i umožňuje odebírat lidské dělníky a dává větší prostor kvalifikované obsluze. V současné době však stále existují procesy, které nelze automatizovat a lidský faktor je tedy nenahraditelný. Začlenění robota a člověka do jednoho výrobního systému ale představuje bezpečnostní riziko a robot tedy musí být opatřen zábranami či senzorikou, aby robot nemohl svými pohyby způsobit zranění nebo dokonce smrt [1,2].

Obr 1. Příklad uspořádání výrobních stanovišť v chytrých továrnách [26]

Průmysl 4.0 však nezahrnuje pouze fyzická zařízení ale i software a umělou inteligenci bez které se tento systém neobejde. Tyto systémy tedy vyžadují i vysoký výpočetní výkon, aby bylo možné všechny procesy zpracovávat v reálném čase. Mezi běžné softwarové vybavení patří i cloudy a takzvaný internet věcí, který tvoří základ Průmyslu 4.0 [1].



Obr 2. Koncept průmyslu 4.0 [3]

3.1 Internet věcí

Internet věcí tvoří stěžejní prvek průmyslu 4.0 ale i většiny smart domácností a podobných systémů využívající chytrá zařízení. Jedná se o označení pro síť fyzických zařízení, která jsou vybavena elektronikou, softwarem, senzory, pohyblivými částmi a síťovou konektivitou, která umožňuje těmto zařízením se propojit a vyměňovat si data. Každé z těchto zařízení je v síti jasně identifikovatelné a spolupracuje s ostatními prvky, zároveň je ale schopno pracovat samostatně. Díky tomu je možné všechna takto propojená zařízení obsluhovat na dálku například pomocí smartphonu. V průmyslu se používá zejména pro sbírání dat ze všech připojených objektů napříč lidskými, materiálovým, strojními a systémovými zdroji, za účelem zabezpečení automatizace a optimalizace příslušných procesů [3,4].

Obr 3. Monitorování strojů pomocí smartphonu [27]

Internet věcí v průmyslu 4.0 využívá čtyři základní principy:

Interoperabilita

Představuje propojitelnost jednotlivých zařízení a systémů. Nejedná se přitom pouze o periferní připojení jednoho zařízení na druhé nebo centralizované spojení stroje s řídícím systémem, ale o komplexní síť. Právě tato komunikace je klíčová pro vytvoření komplexních inteligentních systémů, neboť umožňuje například spolupráci různorodých systémů na jednom procesu. V podnikovém prostředí to znamená propojení strojů, lidí, materiálů, produktů, informačních a komunikačních technologií a systémů a v neposlední řadě i dat ve formě dokumentů. Všechny tyto prvky musí být samozřejmě kompatibilní a vzniklá síť by měla mít schopnost se dále rozšiřovat a umožňovat připojení dalších zdrojů [4].

Decentralizace

Centralizované systémy na principu server–klient disponovaly jednou centrální databází, což mělo za následek udržení konzistence dat, jejich jednoduchou správu a vysokou úroveň bezpečnosti. Takovéto uspořádání však bylo omezeno kapacitou serveru a představovalo zvýšené riziko při výpadcích systému. Decentralizované systémy zastávají myšlenku, že nejlepší je rozhodnutí udělat co nejblíže k místu, kde je zaměřený jeho dopad, protože na tomto místě jsou dostupné potřebné informace i rychlá zpětná vazba. Jednotlivé podsystémy by tedy měli umět rozhodovat sami za sebe a neřídit se pouze nadřazenými příkazy. Tato struktura umožňuje komunikaci jednotlivých komponentů sítě mezi sebou z pozic rovnocenných rolí. Na druhou stranu, komponenty připojené do sítě generují rozsáhlé množství heterogenních dat, které je třeba zpracovat. Přínosem je pak rozšiřitelnost sítě, zvýšená odolnost vůči výpadkům samotné sítě, jednotlivých připojených podsystémů a jejich komponentů [4].

Inteligence

Chytré řízení výroby a logistiky se neobejdou bez nástrojů a technologií umělé inteligence. Smart Industry systémy využívají inteligentní algoritmy na monitorování, kontrolu, řízení a plánování komplexních procesů operací napříč celým výrobním procesem a dodavatelským řetězcem. Postupem času se tyto technologie budou implementovat do více a více systémů za účelem zvyšování autonomie jednotlivých komponentů sítě. Dosáhnutí tohoto cíle je možné nejenom cestou zlepšování softwarového vybavení a navyšování výpočtového výkonu daných zařízení, nýbrž i virtualizací těchto prvků a vytvářením jejich digitálních dvojčat. Vznikají tak kyberneticko-fyzikální systémy propojující fyzické objekty s jejich virtuálními protějšky [4].

Rekonfigurabilita

Je charakterizována jako nezávislá adaptabilnost na vnitřní a externí podněty. Jejím projevem je auto-optimalizace procesů, jež představuje další stupeň inteligence. Nevyhnutelným předpokladem pro dosažení této schopnosti je sběr a přenos různorodých dat napříč podnikovým ekosystémem a jejich bezodkladná analýza. To umožňuje předvídat a pružněji reagovat na případné změny, nečekané situace nebo výjimečné stavy [4].

3.2 Bezpečnostní hrozby spojené se smart systémy

Jelikož je možné se k zařízením připojit v podstatě ze všech míst na světě prostřednictvím internetu vyvstává potíž kybernetického zabezpečení. Žádný software nezajišťuje stoprocentní jistotu bezpečí a čím více je systém komplexnější tím spíše v něm útočník najde nějakou slabinu. Dosavadní způsoby zabezpečení již tedy pro internet věcí nemusí být dostatečné a je zde riziko rostoucího počtu kyberútoků. Mnoho podniků se navíc potýká s nedostatkem kvalifikovaného personálu, který by dokázal bezpečnostní systémy obsluhovat a spravovat [5].

Problémy s bezpečností však nemají pouze podniky, ale měli by jí brát vážně i obyvatelé chytrých domácností. Například takový chytrý termostat na první pohled nemůže způsobit velké škody, ale pokud k němu útočník získá přístup může podle aktuálního vytápění určit, jestli je někdo doma nebo je dům zrovna prázdný. Největší problém ale tvoří zabezpečovací systémy jako třeba ovládání zámků a bezpečnostních kamer, které mohou být opět ovládány na dálku pomocí internetu. Jelikož je ale internet věcí stále na začátku a zatím zde chybějí standardy, které by specifikovaly, jakým způsobem zabezpečovat jednotlivé zařízení, vznikají zde i problémy s poskytnutím náležitého zabezpečení [5,6].

Další hrozbu pro chytré systémy mohou být takzvané Botnet sítě. Botnet síť je tvořena velkým množstvím počítačů nebo chytrých zařízení, které jsou napadeny virem. Z této sítě pak mohou útočníci vysílat velké množství komunikačních požadavků, které způsobí pád cíleného serveru [6].

3.3 Další využití smart systémů

Využití smart systémů v automobilech

Mnoho automobilek se v současné době snaží integrovat smart systémy do svých modelů, za účelem usnadnění řidičům ovládání vozidla a zvýšení bezpečnosti jízdy. Patří sem různé komunikační a navigační systémy jako systémy pro rozpoznávání a zobrazování dopravních značek, nebo varovné systémy upozorňující na dopravní nebezpečí. Používají se zde ale i systémy, které mohou do řízení aktivně zasáhnout. Řadí se sem například nouzová automatická brzda či systém prevence vyjetí z jízdního pruhu. Za zmínku stojí i systém

detekující únavu řidiče, který podle změn chování řidiče dokáže vyhodnotit stav jeho bdělosti, nebo systém schopný rozpoznat protijedoucí auta a odstínit od nich dálková světla [7,8].

Obr 4. Systém pro odstínění dálkových světel [28]

Využití smart systémů ve zdravotnictví

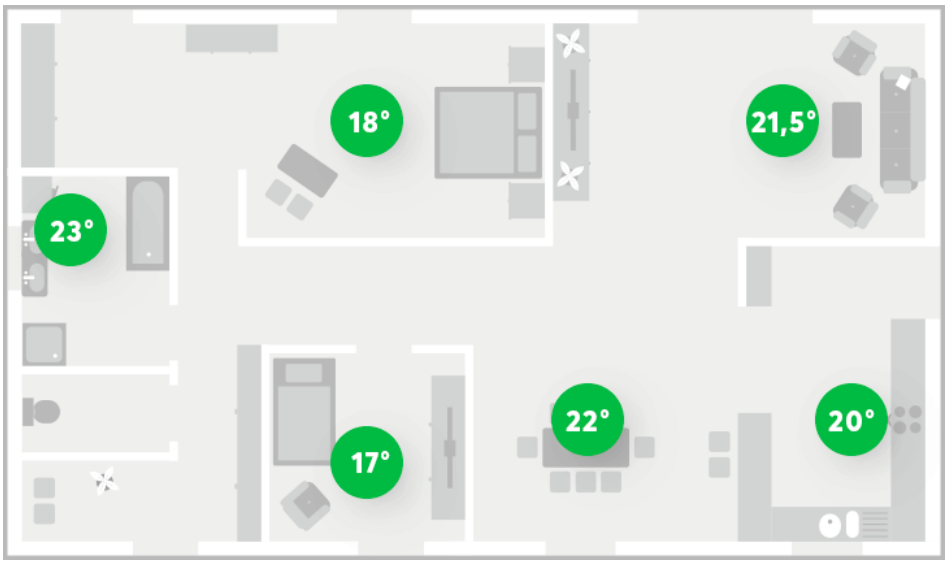
V oblasti medicíny se v současnost rozvíjí zejména umělá inteligence, kterou smart systémy poskytují. Ta se zde trénuje například na analýzu rentgenových snímků, podle nichž dokáže následně posoudit průběh léčby pacienta. Má zároveň uplatnění i v sanitkách, kde pomocí technologie rozpoznávání obličejů detekuje příznaky krvácení a pošle upozornění nemocnici ještě před příjezdem. Rozpoznávání obličejů se však využívá i v samotných nemocnicích k identifikaci totožnosti. Nezřídka se stává, že personál chybně identifikuje pacienta, což může vést až k jeho úmrtí. Všechny tyto systémy jsou ale stále spíše asistenční a konečné rozhodnutí musí vždy učinit člověk [9,10].

4. Smart systémy v domácnostech

Jak už bylo naznačeno chytrá zařízení a systémy se dnes ve velké míře instalují i do domácností. V takovémto případě se mluví o takzvané chytré domácnosti neboli Smart Home. Tyto systémy mají zde za úkol optimalizovat vnitřní prostředí pro zvýšení komfortu a úspory energie. Běžně jsou zde využívány systémy na ovládání světel, vnitřní teploty či zabezpečovací systémy. Smart systém však může zahrnovat téměř jakýkoliv spotřebič schopný připojit se k internetu (či jiného způsobu komunikace) [12,13].

4.1 Přínosy chytré domácnosti

Jeden z významných přínosů smart systému je úspora energií. K řízení teploty totiž nevyužívá pouze elektřinu či plyn ale dokáže i zohlednit a využít venkovní teplotu. Namísto standartních radiátorů se tedy může k vyhřívání (nebo naopak k ochlazování) místnosti přenést teplo z venku. Jedná se o takzvanou rekuperaci. Obdobným způsobem lze pomoci klimatizaci vhodnou stínící technikou (například žaluzie), která v létě bude bránit průchodu slunečních paprskům do budovy. Tyto funkce obvykle spolupracují s meteostanicí, která dodává údaje o počasí. Úsporu energie tvoří i možnost odlišného nastavení teplot v jednotlivých místnostech. Pro spánek je například vhodnější chladnější teplota, zatímco v koupelně je příjemnější spíše vyšší. Umělá inteligence pak dokáže vyhodnotit denní režim uživatelů a vypínat vytápění v místnostech které nejsou aktuálně využívané, nebo naopak zapínat vytápění tam kam si myslí že brzy vejdete [12,14].



Obr 5. Příklad vyhřívání jednotlivých místností [29]

Dalším častým využitím chytrých domácností je ovládání světel. Ta můžou opět vnímat přítomnost uživatele a řídit osvětlení v místnostech bez jeho zásahu. Zároveň dokážou reagovat na intenzitu slunečního osvitu, nebo měnit intenzitu osvětlení podle denní doby [14].

Smart systémy najdou uplatnění i v zabezpečovacích systémech. Ty dokážou chránit jak nedovolenému vniknutí, tak i před požárem či záplavou. Všechny systémy lze samozřejmě sledovat například pomocí smartphonu což je užitečné zejména u kamer. Zároveň i informuje o osobách uvnitř domu či otevřených dveřích a oknech. Některé systémy poskytují i funkci

simulace nepřítomnosti, který samovolně zapíná a vypíná světla, ovládá stínící techniku apod. Tento systém slouží k odrazování potenciálních zlodějů, kteří tak nepoznají že je dům ve skutečnosti prázdný [14].

Největším přínosem chytré domácnosti je ale možnost ovládat všechna zařízení pomocí smartphonu či podobných zařízení, a to odkudkoliv. Právě možnost ovládat a sledovat všechny smart prvky z jednoho místa by měla obsahovat každá chytrá domácnost ať už se jedná o ovládání světel, termostatu nebo garážových vrat. Mnoho smart systémů dokáže navíc pomocí telefonu sledovat polohu uživatele, což využívají například bezpečnostní systémy, které se automaticky zapínají, když uživatel opouští dům [12,14].

Toto byly ty nejběžnější použití. Jak už bylo ale řečeno, možnosti využití smart zařízení a smart systémů jsou dnes velice rozsáhlé. Může se jednat třeba o chytrou lednici, která pozoruje stav potravin a může vytvořit i nákupní seznam. Systémy na údržbu bazénů, která se na základě senzorů teploty, hladiny či úrovně dezinfekce sami spouštějí. A samozřejmě televize a ostatní multimédia schopná ozvučit celý dům. Vymyslet se dá opravdu leccos [13].

Obr 6. Příklad chytrých systémů a zařízení ve smart domácnosti [30]

4.2 Hlasoví asistenti

V chytrých domácnostech se často využívají hlasoví asistenti. Tito asistenti umožňují ovládat smart systémy nikoliv pomocí obrazovky smartphonu či tabletu ale pomocí hlasových příkazů. Obvykle se jedná o malá zařízení s mikrofonem a reproduktorem, ale může se nacházet i v samotných smartphonech či tabletech (Siri, Cortana). Jejich hlavní funkcí je tedy

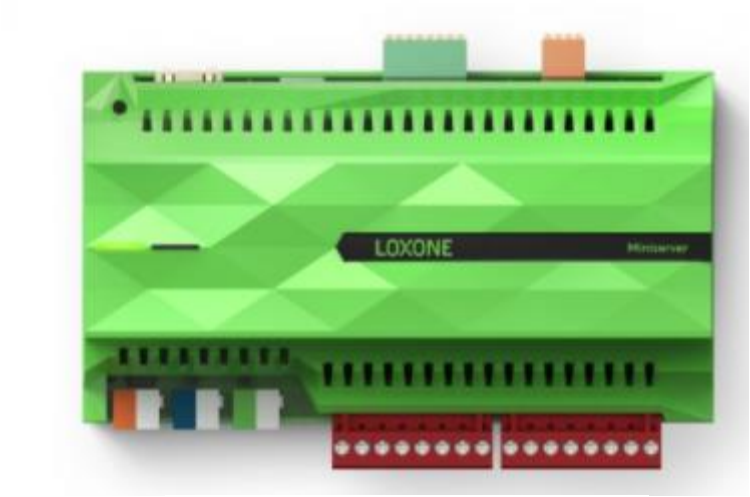
rozpoznávat a vykonávat hlasové povely. Ty se však nemusí týkat jen chytré domácnosti, ale dokážou i hlídat e-mailovou schránku a kalendář, rezervovat místo v restauraci, objednat nákup či vyhledat informace na internetu. Komunikace by měla být i obousměrná. Hlasový asistent by tedy měl být také schopný odpovídat na otázky nebo upozorňovat na problémy, a to mluvenými slovy pomocí reproduktoru. To vše samozřejmě již vyžaduje umělou inteligenci na vyšší úrovní. V současné době jsou na trhu čtyři významní výrobci těchto asistentů. Alexa od Amazonu, Siri od Applu, GoogleHome a Cortana od Microsoftu. Mezi těmito značkami je však dominantní právě Amazon se svou Alexou. Některé automobilky ji dokonce využívají ve svých modelech pro ovládání rádia, ohlašování úrovně paliva a kapalin či hledání vozidla na parkovišti. Nedostatkem těchto asistentů je pak absence české lokalizace. Se všemi zmíněnými asistenty se musí komunikovat anglicky, což může být pro některé uživatele nepohodlné [15].

Obr 7. Zařízení pro komunikaci s hlasovými asistenty [16]

4.3 Cloudová a free-cloudová varianta

Stejně jako v případě průmyslu 4.0 i v chytrých domácností všechna zařízení mezi sebou obvykle komunikují skrze internet věcí. Ten však obvykle vyžaduje nějaký hardware, který zprostředkovává komunikaci mezi jednotlivými zařízeními, poskytuje úložný prostor pro data, které smart systémy generují a obsahuje softwarovou základnu s umělou inteligencí. Jedna možnost je vyžít cloudy, které některé společnosti pro potřeby chytrých domácností poskytují. Tento způsob však má pár negativ. Ten hlavní je závislost na připojení k serverům těchto společností. Cloud v tomto případě zprostředkovává i veškerou komunikaci mezi zařízeními a případný výpadek připojení k internetu, nebo porucha samotného cloudu může způsobit narušení činnosti některých smart systémů nebo můžou přestat fungovat úplně. V tomto případě

by byl výpadek pouze krátkodobý, může se ale stát, že se společnost rozhodne službu cloudu trvale ukončit, a to by již znamenalo poměrně závažnější problém. Druhým rizikem cloudů je ochrana soukromí. Nejednou již velké společnosti čelili obviněním, že odposlouchávali své klienty. Alternativou ke cloudu je pak takzvané free-cloudové řešení. Jedná se o řešení, kdy se hardware (miniserver) s uložištěm a softwarovou základnou nachází přímo v obydlí uživatele. Tento způsob pracuje pouze v rámci lokální sítě, a tak nevyžaduje připojení k internetu, když si uživatel potřebuje třeba jen rozsvítit. I když připojení k internetu je stále zapotřebí pro některé dílčí funkce jako třeba vzdálený přístup. Negativní stránkou tohoto řešení může být nedostatečná softwarová podpora [17].



Obr 8. Miniserver firmy Loxone [31]

4.4 Možnosti realizace chytré domácnosti

Jsou dva základní způsoby, jak do domácnosti nainstalovat smart systémy. První je při stavbě nového domu. Tento případ je výhodnější, neboť nová stavba může být lépe projektována a přizpůsobena na využití smart systémů. Navíc se instalací smart systémů do novostaveb dá i ušetřit za elektroinstalaci. Jelikož všechna zařízení (světla, termostat, žaluzie, …) lze ovládat přes internet z jednoho místa, můžou se tak ušetřit peníze za různé ovládací prvky (vypínače termostaty, …), kabely apod. Cena chytré domácnosti tak může být i srovnatelná s tradičním ovládáním. Chytrý dům tedy může být dostupný i pro běžného člověka, a nejen pro několik vyvolených.

Druhý způsob je pak dodatečná instalace smart systémů, do již postavených budov. V tomto případě však můžeme narazit na různé limity a komplikace. Některé objekty nemusí být pro chytré systémy vhodné a pro některá moderní zařízení by byla nutná alespoň částečná přestavba. Zejména u starších budov může být instalace nákladná a bez významných pozitiv, které by měl smart home přinášet. V současné době se však na trhu objevuje více a více zařízení, které se mohou dát bez větších problémů kamkoliv. Typickými příklady jsou chytré žárovky, zásuvky či televize. Ve většině případů však tyto zařízení postrádají umělou inteligenci a schopnou učit se [18].

4.5 Chytrá města

V souvislosti se smart systémy vznikla i myšlenka chytrých měst (smart city). Chytré systémy zde slouží k efektivnějšímu využití své infrastruktury, snížení spotřeby energií a obecně zvýšení kvality života. To může zahrnovat nové obnovitelné zdroje, optimalizaci dopravy, nebo taky sdílení dat a sítí pro veřejné účely. Smart city by tak mělo být řešením pro velké metropole, které se potýkají s vysokým růstem obyvatel a tím i s rostoucími nároky na dopravu a energii, větší spotřebu vody či produkcí odpadu [19,20].

Obr 9. Koncept chytrého města [32]

Zde jsou ideje, které chytré město obsahuje nebo do budoucna obsahovat může.

Doprava

V chytrých městech mají fungovat systémy sledování provozu, řízení provozu pomocí chytré dopravní signalizace a systémy chytrého parkování, které navádí řidiče k volnému parkovacímu místu. Veřejná doprava by měla využívat solární energii či bioplyn a řídit ji bude počítač nikoliv řidič či strojvedoucí. Zastávky budou obsahovat informační cedule s časem do příjezdu dalšího spoje a nabídnou bezplatnou wifi či nabití telefonu. Po městě by pak měly být rozmístěny rychlonabíjecí stanice na elektromobily a chytré stojany na sdílené bicykly [19,20].

Obr 10. Chytré stojany na sdílené bicykly [33]

Energetika

Chytrá města se soustředí na využívání obnovitelných zdrojů. Budovy tak mají na střechách solární panely, které je činí alespoň částečně energeticky nezávislé. Budovy zároveň opět využívají principy chytrých domácností pro snížení nákladů na energii (například rekuperaci). Modernizací projde i elektrická síť, která bude efektivněji řídit výrobu a spotřebu elektrické energie a dokáže rychle reagovat na poruchy. Obdobně budou fungovat i vodovody, které budou vybaveny senzory pro únik vody a kontaminaci. A nakonec pouliční osvětlení bude měnit intenzitu svého osvětlení podle slunečního svitu a bude reagovat na přítomnost obyvatel [19].

Životní prostředí

V chytrých městech by měly být prosazovány technologie zlepšující ekologii a životní prostředí. Po městě se tedy bude měřit kvalita ovzduší a podle potřeby se bude pěstovat více stromů a zeleně, a to ať už na střechách v parcích nebo na náměstí. Občané budou motivováni k šetrnost, recyklaci a snižování emisí například budováním více cyklostezek. Budou instalovány podzemní kontejnery a chytré odpadkové koše, které si sami zavolají odvoz, když se naplní. Bezplatná wifi a zásuvky pro nabití telefonu budou k dispozici prakticky všude, kde je to možné (na zastávkách, v lavičkách apod.) [19].

Obr 11. Chytrá lavička [34]

Informační systém

Posledním výrazným aspektem je digitalizace informačního systému a sociálních služeb. Může se jednat o digitální vývěsky, nahrazení rozhlasu mailingem či automatickými SMS zprávami, nebo třeba lepší možnost vzdálené komunikace a zařizování nejrůznějších záležitostí, pro které není nezbytně nutná přítomnost občana v dané instituci. Občané by tak měli být schopni pomocí smartphonu sledovat kulturní akce či dopravní situaci. Zároveň by občané měli mít možnost se na tomto informačním systému sami aktivně podílet [20].

5. Home Assistant

Již jsem se zmínil o nedostatcích cloudového řešení a proč je výhodnější mít chytrou domácnost opatřenou vlastním miniserverem. I v případě miniserveru však lze narazit na jistá úskalí, a to v podobě komptability zařízení od různých výrobců. Pokud se rozhodnete pro vybudování smart domácnosti využít odbornou firmu, bude se nejspíše snažit celou domácnost vybavit svým vlastním hardwarem včetně miniserveru. Když ale budete chtít do smart systému připojit zařízení od jiného výrobce může se stát, že miniserver nedokáže zařízení a jeho funkce rozpoznat a není tak schopen ho řídit. V takovém případě je pro ovládání zařízení potřeba odlišná mobilní aplikace. To může vést až k situaci, kdy se každé zařízení v domácnosti ovládá jinou aplikací a chytrá domácnost tak spíše život komplikuje, nežli usnadňuje [23,24].

Řešení může poskytnout platforma Home Assistant. Jedná se o bezplatný software určený primárně pro vytvoření chytré domácnosti. Je to velice populární řešení, které klade důraz na lokální ovládání a soukromí. Jeho obrovskou výhodou je schopnost propojit a ovládat prakticky všechny prvky chytré domácnosti jedinou aplikací. V současné době dokáže integrovat přes 1800 zařízení od stovek výrobců čímž se eliminuje problém více mobilních aplikací, které se tímto zúžili v jednu [21].

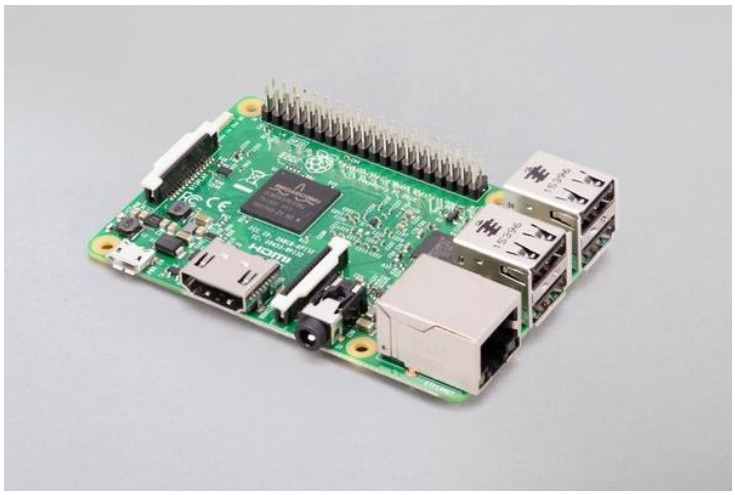
Samotný software se ale samozřejmě neobejde bez fyzického hardwaru. Ten může být zakoupen přímo od výrobce Home Assistanta, může se ale použít i odlišný. Jelikož je celý software napsán v Pythonu, nabízí se tu možnost využít například populární mikropočítač Raspberry Pi, které taky většina uživatelů pro tyto účely používá. Do toho stačí už jen nainstalovat firmware Home Assistanta, volně dostupný na internetových stránkách. Tímto by byl připravený centrální miniserver schopný zajistit komunikaci mezi chytrými zařízeními, HMI, nadřazené řízení apod. [22]

Home Assistant však není pro každého. Zacházení vyžaduje alespoň základní dovednost programování, a zvláště s některými pokročilejšími funkcemi si již běžný člověk nebude vědět rady. Home Assistant je však dnes velice rozšířený a vyvinula se kolem něj silná komunita ochotná pomoci [22,24].

Dalším záporem je časová náročnost. Vybavení a naprogramování chytré domácnosti může zabrat i desítky hodin. Pokud tedy člověk nemá vůli a trpělivost, vyplatí se mu spíše zaplatit profesionální firmě [24].

5.1 Instalace Home Assistant na Raspberry Pi

Již jsem se zmínil o možnosti instalace Home Assistant na mikropočítač Rasspberry Pi. Lze samozřejmě využít i jiné platformy jako třeba stolní PC s operačním systémem (Windows, Mac), Raspberry ale mezi nimi vyniká skvělým poměrem cena/výkon a je i doporučováno vývojáři Home assistanta. Co se týče konkrétního modelu, tak by mělo být použito Rasspberry Pi 3 nebo novější.



Obr 12. Raspberry Pi 3 Model B [35]

Instalace pak začíná stažením obrazu Home Assistant OS z webových stránek. Tento obraz se následně pomocí vhodného adaptéru a softwaru (Balena Etcher) nahraje na micro SD kartu aplikační třídy 2 (doporučená velikost je 32 GB nebo větší). Tato SD karta bude sloužit jako harddisk a bude na ni nahrán operační systém Home Assistanta. Pro připojení Raspberry k routeru/switche je možné použít ethernetový kabel nebo wifi, přičemž je spíše doporučován kabel pro jeho spolehlivost a ušetření práce s instalací konfiguračních souborů. Po vložení SD karty, připojení ethernetového kabelu a přiložení napájení se následně operační systém sám nainstaluje.

Po dokončení instalace operačního systému je třeba ještě provést konfiguraci. Ta se obvykle provádí pomocí stolního PC a prohlížeče tak, že se v prohlížeči otevře

http://homeassistant:8123 (nebo se namísto aliasu napíše přímo IP adresa). Na této stránce se provede vytvoření uživatelského účtu a základní nastavení jako název domácnosti, poloha, základní fyzikální jednotky apod. Home Assistant se následně pokusí vyhledat všechna dostupná zařízení v síti. Tímto by byla instalace dokončena a Raspberry Pi s Home Assistantem je připraveno k použití [21,22].

5.2 Přidání zařízení

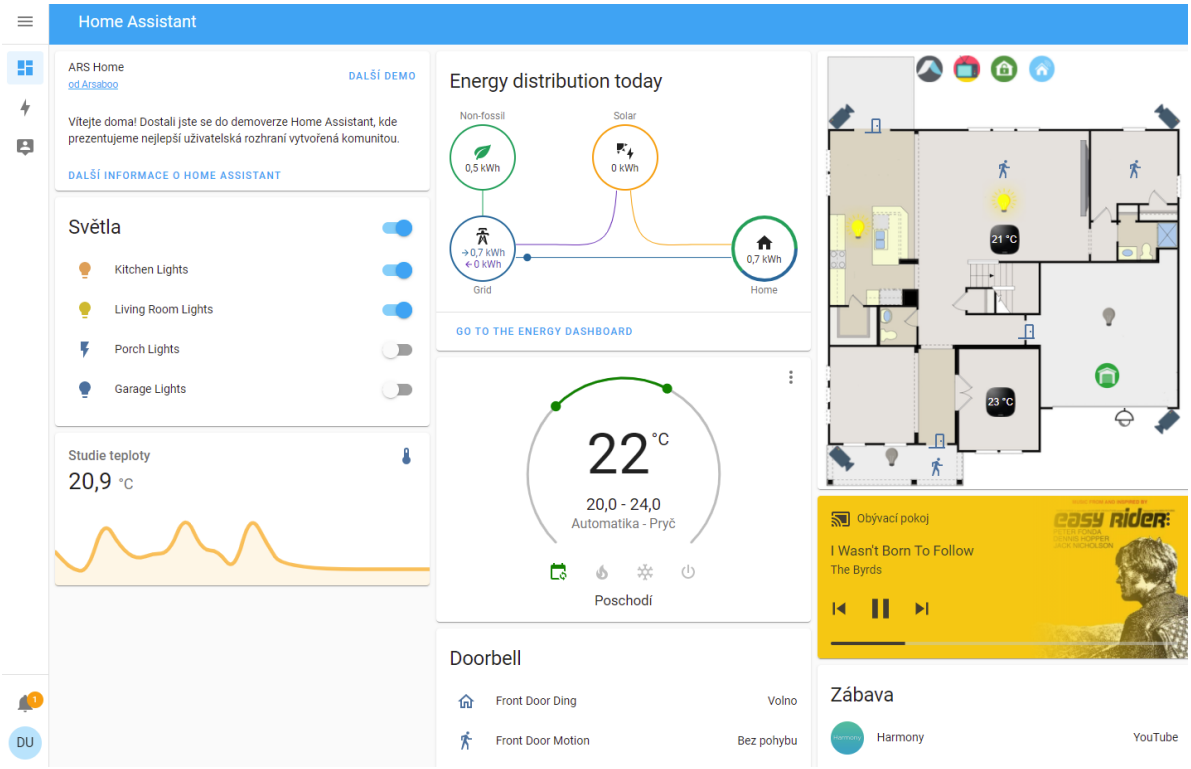
Může se stát, že některá zařízení Home Assistant nedokáže vyhledat automaticky a je třeba konfigurovat je manuálně pomocí textového editoru. To se provádí opět pomocí webového prohlížeče otevřením *http://homeassistant:8123*. Nyní se však namísto úvodního konfiguračního nastavení otevře rozhraní, které stejně jako mobilní aplikace slouží k ovládání smart domácnosti. Nová zařízení se přidávají v záložce Nastavení/Zařízení a služby. Návod, jak konkrétní zařízení nastavit je k dispozici na webových stránkách Home Assistanta (https://www.home-assistant.io/integrations), kde jsou vypsány i příklady. I když se často nestává, že by Home Assistant nedokázal zařízení vyhledat, tak tento způsob konfigurace je často používaný i pro možnost přizpůsobit si funkce zařízení na míru. Může se takto upravit například výchozí intenzita osvětlení žárovky, různá časová zpoždění, režimy apod.

Všechna připojená zařízení a prvky lze najít v záložce Vývojářské nástroje. Zde jsou popsány jak veškeré specifikace, tak i aktuální nastavení a stavy jednotlivých zařízení a prvků chytré domácnosti. V tomto okně lze i všechna zařízení ovládat. Při více zařízení však začíná být toto okno nepřehledné a je nepohodlné v něm něco stále hledat. Často používané prvky se proto vyplatí vložit do uživatelského rozhraní [22,23].

5.3 Uživatelské rozhraní

Ovládání chytré domácnosti probíhá na kartě *Přehled* ve webovém prohlížeči nebo aplikaci Home Assistantu. Na této kartě se nachází takzvaný *Lovelace* což je přehled zařízení a funkcí, která si daný uživatel přeje ovládat nebo využívat. Toto rozhraní může mít každý uživatel odlišné podle preferencí nebo přístupových práv (běžný uživatel by například neměl mít přístup k GPS poloze ostatních uživatelů). Pomocí funkce *uživatelé* se tedy dají nastavovat i funkce přístrojů, kdy na jednom tabletu se zobrazují informace o počasí a na druhém tabletu se ovládá teplota v jednotlivých místnostech (stejným způsobem je i vhodné vytvořit zvláštní panel pro mobilní zařízení kvůli velikosti obrazovky).

Další způsob, jak využít funkci *uživatelé* je sledování vlastní polohy, resp. přítomnosti. Tato funkce může pracovat s několika různými technologiemi (GPS, Bluetooth, …) a může buď sledovat pouze zdali se uživatel nachází v určitém prostoru (využívá se například pro zapnutí vytápění, když se uživatel blíží k domovu) nebo celkový pohyb (využití v krokoměrech). Podmínkou správného fungování je tedy nutnost neustále u sebe nosit nějaké chytré zařízení s nainstalovanou aplikací Home Assistantu (chytrý telefon nebo hodinky), což může být v některých případech dosti omezující.



Obr 13. Ukázka uživatelského rozhraní [36]

Mnoho lidí si rádi Lovelace upravují i po grafické stránce (vzhled karet, pozadí, …). Jedním z estetických vylepšení pro ovládání domácnosti je tedy i vytvoření obrázku jejího půdorysu. Do tohoto obrázku se pak vkládají interaktivní ikony svítidel, termostatů a dalších chytrých zařízení připojených na Home Assistant. Toto uskupení pak poskytuje dobrý přehled o rozmístění zařízení a jejich stavu na rozdíl od prostého “sloupcového výpisu“. Samotný půdorys se nejsnadněji vytváří pomocí rozšířené reality (například aplikací *magicplan*). Půdorys je ještě vhodné doplnit o nábytek, okna apod.

Obr.14: Ukázka půdorysu s interaktivními prvky [3]

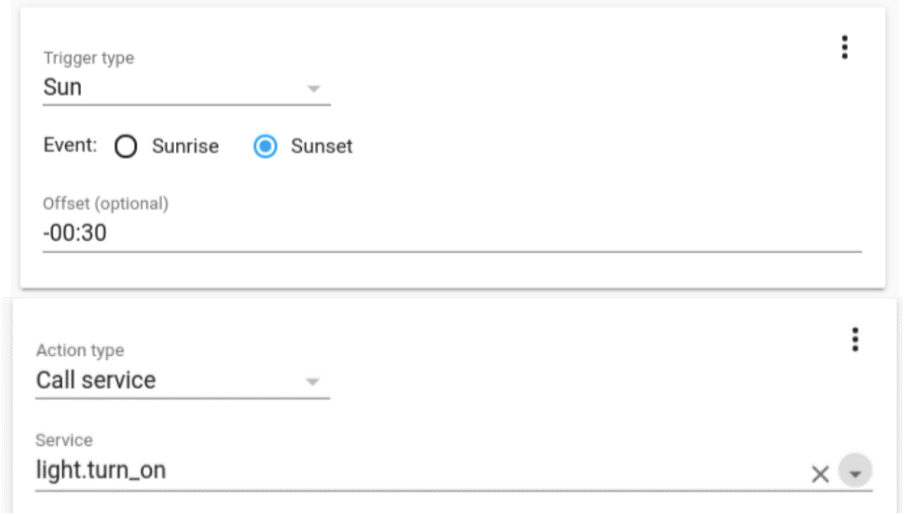
5.4 Automatizace

Dosud se stále jedná o prosté řízení, vykonávané uživatelem. Přednost chytrých domácností spočívá ale hlavně ve vzájemné spolupráci všech prvků a jejich automatizaci, která se vykonává bez zásahu uživatele. V Home Assistantu se tato automatizace nastavuje v záložce Nastavení/Automatizace.

Jako první se zde nastaví Spouštěč, což je událost, která nastartuje daný proces. Může se jednat o detekci pohybu v místnosti, východ slunce, opuštění domu apod. Těchto podmínek je možné nastavit několik. Například v případě, kdy by byl jeden Spouštěč senzor, který

zaregistroval pohyb a zároveň by druhý Spouštěč hlídal západ slunce (do místnosti ve které je tma vešla osoba => zapnout světlo). V tomto případě by musely být splněny obě podmínky najednou (log. funkce AND), Spouštěče ale lze samozřejmě nastavit i tak že stačí splnit pouze jednu z nich (log. funkce OR).

Poté se nastaví Akce, což je úkon, který se provede po aktivaci Spouštěče. Například rozsvícení světla, otevření okna, spuštění zabezpečovacího systému apod.



Obr 15. Ukázka nastavení automatizace [20]

Pro náročnější uživatele je možné programovat automatizaci v jazyce C nebo Python. Ve většině případů je ale tento jednoduchý způsob postačující. Na internetu jsou k dispozici všemožné projekty od komunity, které dokazují, že s trochou fantazie se dá tímto způsobem automatizovat téměř cokoliv. Například kávovar, který se spustí při vstávání z postele. Chytrý vysavač, který začne uklízet, když je dům zrovna neobydlený. Nebo již zmíněné zapínání/vypínání světel reagující na přítomnost uživatele v místnosti. Samozřejmě lze pro automatizaci využít i hlasového asistenta nebo smartphone například pro zasílání různých upozornění [22,23].

5.5 Integrace

Do této kategorie spadají především veškerá chytrá zařízení, která jsou připojena a ovládána pomocí Home Assistantu.

Home Assistant však není schopen integrovat pouze fyzická zařízení ale i služby. V tomto případě se nejedná pouze o doplněk ale o systémy, které můžou fungovat nezávisle na Home Assistantu (Home Assistant s nimi pouze komunikuje). Zde je několik užitečných služeb, které Home Assistant podporuje.

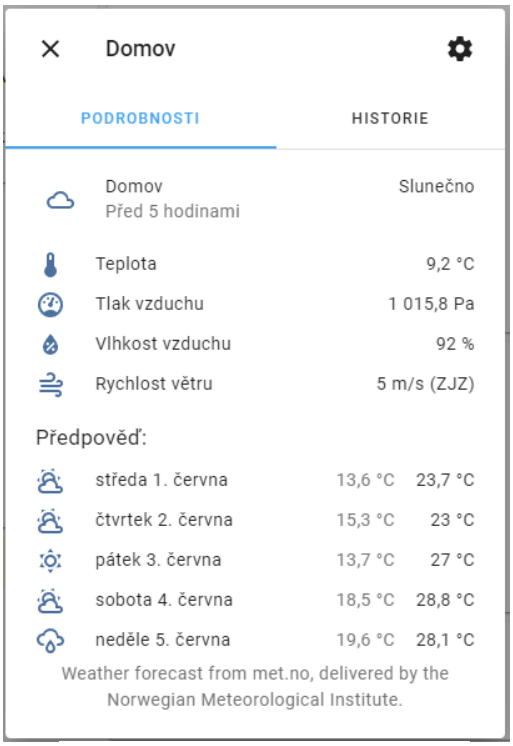
IFTTT

V podstatě se jedná o jednoduchý systém pro chytrou domácnost. Na rozdíl od Home Assistantu, který vykonává většinu funkcí lokálně, IFTTT funguje na principu cloudu. Na tomto cloudu se zpracovávají jednoduché podmínky ve smyslu “pokud nastane tento stav, vykonej tento úkon“. To může obnášet prosté ovládání (po stisknutí vypínače se rozsvítí světlo), nebo nastavenou automatizaci (při poklesu teploty se zapne vytápění). Využíváním IFTTT však Home Assistant přichází o výhody bez-cloudového systému. Tou hlavní nevýhodou je doba odezvy, která se pohybu v řádu sekund. Navíc při výpadku cloudu jsou všechna smart zařízení prakticky nepoužitelná (to samé platí při problému s komunikaci s cloudem, která je o poznání náročnější). IFTTT však obsahuje některé funkce, které Home Assistant neumí nebo je složitější je nastavit (například připojení na sociální sítě). Největší výhodou je pak jednodušší ovládání chytré domácnosti i když se uživatel aktuálně nenachází doma, nebo ovládání chytrých zařízení, které jsou mimo dosah lokální sítě.

Home Assistant dokáže spolupracovat i s dalšími systémy chytrých domácností na principu cloudu jako Google Assistant nebo Amazon Alexa. V těchto případech ale již nepřináší velké benefity a často se využívají jen kvůli hlasovým asistentům.

Počasí (Weather)

Tato a podobné integrace umožňují vyhledání nejbližší meteostanice a sledování podrobných údajů o počasí (teplota, vlhkost vzduchu, tlak, rychlost větru) včetně předpovědi. Tato funkce je dnes velice často využívaná a bývá tedy i automaticky nainstalovaná při instalaci Home Assistantu.



Obr.16: Údaje z meteostanice

Vlastní integrace

I když Home Assistant podporuje integraci mnoha zařízení od mnoha výrobců, existují stále i přední výrobci kteří oficiální podporu nemají. Home Assistant naštěstí umožňuje i instalaci vlastních (neoficiálních) integrací a doplňků. Vzhledem k rozsáhlé komunitě tedy začaly vznikat “komunitní“ integrace, kterými lze tyto nedostatky vyřešit. Ty je možné instalovat v Obchodě s doplňky kolonkou Repositáře.

5.6 Zálohování

Může se stát, že se Home Assistant přestane fungovat. Pokud Home Assistant běží na Raspberry Pi (nejčastější případ), může se stát, že zejména při výpadku napájení dojde k poruše dat na SD kartě, a tedy i kritickému poškození Home Assistantu. Další častou příčinou je špatná obsluha, která má následek způsobenou chybu v konfiguraci. Tento případ obvykle nebývá tak závažný a je jednodušší ho opravit. [11]

Pro takové případy je vhodné systém často zálohovat. Záloha se vytváří v Nastavení/Doplňky/Zálohy, kde se dá vytvořit buď úplná nebo částečná záloha (záloha jen vybraných částí). Úplná záloha může mít velikost i několik stovek MB ale vyplatí se jí dělat,

neboť některé části konfigurace se mohou nalézat jinde, než by se očekávalo (konfigurace doplňků se například nenalézá v souborech s doplňky ale v primárních souborech Home Assistantu). Vytvořená záloha se pak běžně ukládá na SD kartu což je riskantní v případě, že se SD karta poškodí. Je proto vhodnější si zálohy ukládat i na jiná paměťová média (nejčastěji PC nebo cloud). Zálohování se využívá i pro migraci Home Assistantu na nový hardware, kdy se všechna uživatelská data a nastavení jednoduše “zkopírují“ a není tak potřeba Home Assistant konfigurovat od nuly.

Případná porucha může mít tři následky. [11]

1. Chyba v konfiguraci způsobí výpadek pouze části Home Assistant jako třeba jedné skupiny zařízení nebo doplňku. V takovém případě obvykle stačí vrátit se do nastavení zálohování a vybrat poslední vytvořenou zálohu.

2. Chyba znemožní přihlášení do Home Assistantu, ale Home Assistant stále běží. Tady je již potřeba zálohu obnovit nejlépe pomocí SSH serveru, pomocí kterého se dá k zařízení s Home Assistantem připojit a zálohu obnovit skrze příkazový řádek (vyžadováno doplněk *SSH server).* K tomuto účelu se dá použít například program PuTTy.

3. Pokud se připojení pomocí SSH serveru nepodaří, pravděpodobně to znamená již závažnou chybu (například již zmíněnou chybu SD karty) a Home Assistant je třeba přeinstalovat. Poté nahrát zálohu z PC či cloudu (programem Filezilla nebo doplňkem SambaShare) a opakovat postup s SSH serverem.

6. ESPHome

Home Assistant poskytuje i možnost instalace dalších doplňků (add-on), které dále rozšiřují možnosti integrace různých zařízení, jejich automatizace apod. Jeden z velice užitečných doplňků je ESPHome umožňující vytvářet vlastní firmware. Jeden ze způsobů využití je vytvoření vlastního firmwaru do zakoupených výrobků, kterým chcete vytvořit nové funkce nebo je přepojit z cloudu na svoje Raspberry s Home Assistantem. Největším přínosem je ale možnost tvorby vlastního hardwaru a smart prvků, které pak lze připojit k chytré domácnosti. Může se jednat o různé senzory, dekorační osvětlení či obyčejná spínací relé. Lze

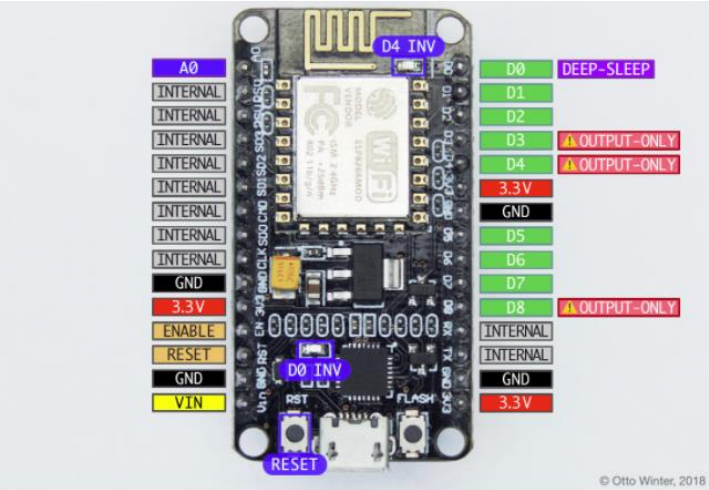
tímto způsobem i vytvořit z klasického domácího spotřebiče smart zařízení a obyčejnou žárovku tak ovládat pomocí smartphonu. Výhodou těchto podomácku vyrobených zařízení je hlavně cena. Cena materiálu na obyčejný senzor pohybu se pohodlně vejde do 200 Kč, zatímco kupované senzory se běžně pohybují od nižších stovek až do tisíců korun. Další výhodou je možnost vytvoření designu sobě na míru. Lze tak tedy vyrobit zařízení všech tvarů a funkcí (zejména v kombinaci s 3D tiskárnou), která se v obchodech neprodávají.

ESP Home se do Home Assistanta instaluje v záložce ADD-ON STORE. Nový firmware se poté vytvoří v záložce ESP Home. Programování standartně probíhá pomocí speciální skriptovacího jazyka, složitější funkce je ale vhodnější programovat v jazyce C nebo Python (možné i naprogramovat jako Automatizaci). Pokud si nejste jistí, jak dané zařízení správně naprogramovat, tak na webových stránkách ESP Home jsou k dispozici návody a příklady pro běžně užívaná zařízení a funkce. Nahrání hotového firmwaru je nejjednodušší pomocí Raspberry Pi s Home Assistantem (přes USB port).

```
binary_sensor:
  - platform: gpio
    pin:
      number: GPIO0
      mode: INPUT_PULLUP
      inverted: true
    name: "Sonoff Basic Button"
    on_press:
      - switch.toggle: relay
```

Obr 17. Ukázka skriptovacího jazyka [37]

Jako hardware, do kterého se firmware nahrává ESP Home doporučuje vývojovou desku s modulem ESP8266 nebo novější ESP32. Základní komponenty této desky standartně tvoří mikrokontroler, vstupně/výstupní piny a WIFI modul. Právě WIFI modul je velice podstatnou součástí, neboť po nahrání firmwaru umožňuje bezdrátovou komunikaci s Home Assistantem. Lze ho tedy tak řídit na dálku jako běžné smart zařízení a zároveň tak snadno upravovat firmware již bez připojení kabelu. K mikrokontroleru se pak pomocí pinů připojují zařízení typu senzor, LED páska, relé apod. čímž se finální výtvor stává plnohodnotným smart zařízením, které je součástí Home Assistatntu [21,24,25].



Obr 18. Vývojová deska NodeMCU ESP8266 [38]

6.1. Způsoby programování ESPHome

Programování pomocí jazyka YAML

Jelikož objektově orientované jazyky mohou být vzhledem k jejich struktuře náročné pro ukládání a zpracovávání paměťmi, je vhodné jejich strukturu před zápisem do paměti nejdříve upravit.

Jeden ze způsobů této úpravy je právě jazyk YAML, který dokáže objektově orientovaný program “serializovat“ a znovu zrekonstruovat do původní podoby. Tento jazyk je velice oblíbený u Python programátorů ale podporuje prakticky všechny objektově orientované jazyky (C++, Java, …). [40]

V ESPHome se jazyk YAML používá jako hlavní způsob programování. V podstatě jde pouze o vyplňování parametrů jednotlivých vstupů, výstupů, proměnných a vazeb mezi nimi. Vzniká tak velice přehledná struktura vhodná zejména pro jednodušší programy.

```
sensor:
  - platform: apds9960
    type: BLUE
    name: "APDS9960 Blue"
    id: blue_color
```

Obr.19: Ukázka YAML

Lambda

Pro složitější programy již není základní YAML struktura dostačující a je třeba jí rozšířit jinými programovacími jazyky. K tomu ESPHome využívá takzvanou Lambdu, která umožňuje jazyk YAML doplňovat C++ kódem. Tímto způsobem se dá vytvořit pokročilejší automatizace na kterou je již jazyk YAML krátký.

```
switch:
  - platform: template
    name: Living Room Light
    lambda: !lambda |-
      if (id(living_room_switch).state) {
        return True;
      } else {
        return False;
      }
```

Obr.20: Ukázka YAML kódu s použitou

Programování pomocí jazyka C++/Python

I když ESPHome upřednostňuje programování pomocí YAML kódu, lze program psát i v jazycích C++ nebo Python. Je však stále doporučováno rozdělit program na menší podprogramy pro jednotlivé komponenty (konfigurace vstupů, automatizace, …) a ty poté importovat do YAML souboru. Značně se tak usnadní kompilace a nahrání finálního programu na desku.

```cpp
#include "esphome.h"

class MyComponent : public Component {
 public:
  void setup() override {
    pinMode(5, INPUT);
    pinMode(6, OUTPUT);
  }

  void loop() override {
    if (digitalRead(5)) {
      digitalWrite(6, HIGH);
    }
  }
};
```

Obr.21: Ukázka kódu v jazyce C++

```yaml
esphome:
  includes:
    - my_component.h

custom_component:
- lambda: |-
    auto my_custom = new MyComponent();
    return {my_custom};
```

Obr.22: Importování C++ kódu do YAML struktury

6.2. Vytvoření a nahrání firmwaru pro ESPHome

Vytvoření nového zařízení v Home Assistantu je poměrně jednoduché. Stačí otevřít kartu ESPHome kliknout na *New device* a poté se řídit průvodcem, kde se nastaví jméno zařízení, vybere se použitá ESP deska (například ESP32) a zadají se přihlašovací údaje k WIFI síti. Home Assistant poté vytvoří firmware se základní konfigurací pro ESP desku. První nahrání firmwaru na desku se nejsnáze provádí pomocí USB připojení desky k PC (v menu vytvořeného zařízení). Poté se deska připojí k nastavené WIFI síti a další úpravy kódu se mohou provádět bezdrátově. Ačkoliv Home Assistant dokáže po většinu času pracovat bez připojení k internetu, tak vytvoření nového zařízení vyžaduje stahování určitých knihoven a je to tedy jeden z okamžiků kdy je připojení nezbytné (stejně jako stahování nových doplňků).

Konfigurační firmware je vhodné ještě doplnit o nastavení statické IP adresy. V opačném případě může docházet k problémům s připojením (například při výměně routeru).

6.3. Další užitečné doplňky (add-ons)

Samba Share

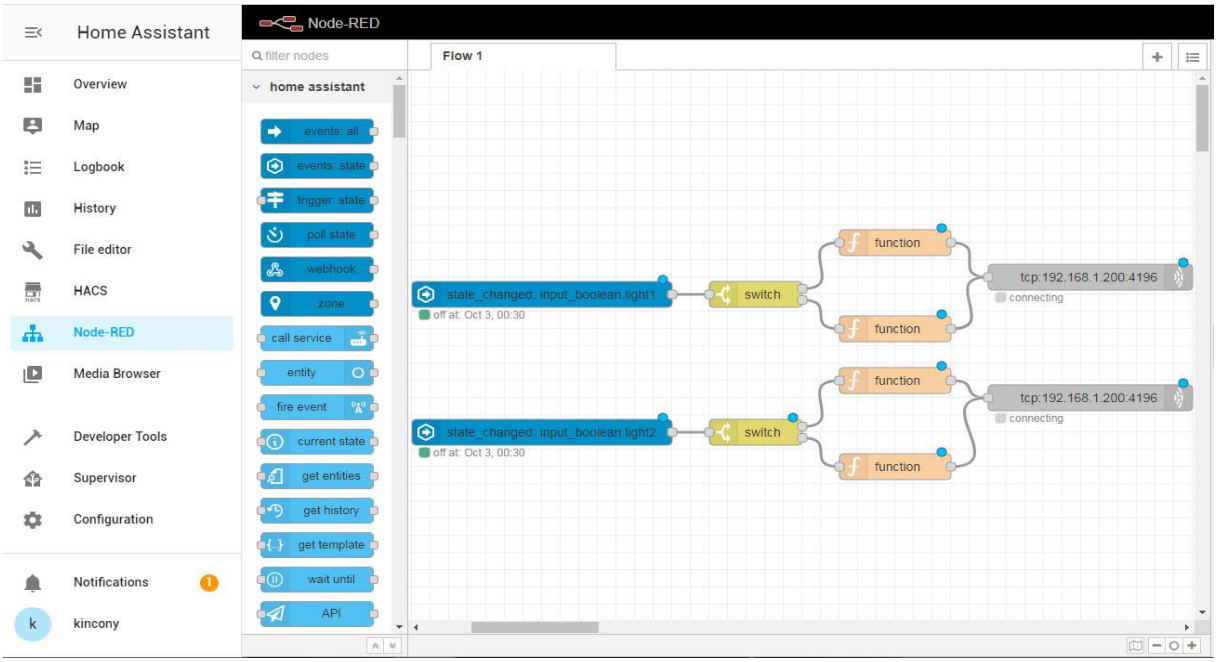
Tento doplněk je určený pro sdílení dat mezi zařízeními s různými operačními systémy. V Home Assistantu se často využívá k zálohování nebo práci se soubory v operačním systému Windows. [39]

File Editor

Poskytuje přehled a možnost editace všech souborů obsažené v Home Assistantu. Lze tak ručně upravovat automatizované procesy, konfiguraci apod. Je to nezbytný prvek pro využívání pokročilých funkcí Home Assistantu.

Node-Red

Jedná se o programovací nástroj, který dokáže dále rozšířit možnosti automatizace v Home Assistantu. Využívá grafické rozhraní, do kterého se vkládají takzvané uzly plnící určitou funkci. Ty jsou následně mezi sebou podle potřeby propojovány, čímž vzniká poměrně přehledné a intuitivní schéma automatizace.



Obr.23: Doplněk Node-Red [25]

7. Postup při návrhu vlastních chytrých zařízení

Návrhu chytrých komponent předcházelo vyzkoušet připojit a naprogramovat různé periférie tak, aby dokázaly spolupracovat se systémem Home Assistant a vyzkoušet tak jejich funkce a možnosti. V první řadě bylo třeba zvolit vhodný programovací jazyk.

I když doplněk ESP Home upřednostňuje programovaní v jazyce YAML, kvůli jeho “jednoduchosti“ je zde obtížné naprogramovat komplexnější řízení. Nabízí se tedy využít jazyk C++. Ten je vhodný i pro běžné “bastlíře“, kteří jsou v něm jsou díky popularitě platformy Arduino zvyklí pracovat. Velká část výrobců elektronických komponent dokonce ke svým výrobkům vydává knihovny napsané právě pro tuto platformu. A právě existence těchto knihoven je další důvod pro využití jazyka C++, neboť většina knihoven určené pro Arduino je možné použít i pro moduly ESP8266 a ESP32.

Největší překážkou je zkombinovat C++ kód s entitami, které využívá Home Assistant a ESP Home. K tomu se využívají způsoby, které většina “bastlířů“ není zvyklá používat nebo je ani nezná a může tak být pro ně náročnější do programování ESP Home proniknout. Přesto jsem usoudil, že programovat v tomto jazyce je pro mě nejlepší volba.

Důležitým aspektem při návrhu bylo dbát na to, aby zařízení dokázalo pracovat i autonomně bez připojení k Home Assisstantu. I když Home Assistant zprostředkovává komunikaci mezi chytrými zařízeními, což je jedna z hlavních předností chytré domácnosti, tak v případě odpojení od WIFI sítě nebo poruchy Home Assistantu by jinak byla zařízení nepoužitelná, což by mohlo být velice problematické například v situacích, kdyby chytré zařízení ovládalo osvětlení nebo vytápění.

7.1. Testované komponenty

Níže jsou vypsané komponenty a hlavní funkce Home Assistantu, které jsem během práce použil nebo vyzkoušel.

Entita Přepínač (Switch)

V podstatě se jedná o dvoustavový IO výstup ovládaný z uživatelského rozhraní Home Assistantu. Je to tedy asi nejzákladnější prvek, neboť zde spadá například ovládání světel (rozsvítit, zhasnout), nebo zařízení ovládaná stykačem či relé. Obtížnější může být v tomto

případě zajistit ovládání takovéhoto zařízení z více zdrojů. Pokud by byl například k zařízení připojen zároveň i fyzický vypínač, bylo by vhodné, aby se při vypnutí zařízení tímto vypínačem projevila změna stavu i na entitě v rozhraní Home Assistantu (entita se sama přepnula do polohy Vypnuto). To platí i o dalších používaných entitách.

Obr.24: Entita Switch

Entita Binární senzor (Binary sensor)

Z názvu je již poznat, že se jedná samozřejmě o digitální IO vstup. V chytrých domácnostech nahradily především nástěnné přepínače jednak kvůli jednoduššímu zapojení ale především díky pohodlnějšímu zacházení (zvláště při schodišťovém zapojení). Snadněji se na něm totiž používá například funkce vícenásobného stisknutí, která by se na standartním přepínači provozovala poměrně nepohodlně. Do této skupiny lze zařadit jakýkoliv senzor s dvojstavovým výstupem (koncové spínače, PIR senzor, …).

Pozn.: Mezi binární senzory spadají i například výstupy RFID čtečky nebo bezdrátové přijímače, ty je ale obvykle třeba nejdříve zpracovat příslušným komunikačním protokolem (viz. níže).

Obr.25: Entita Binary sensor

Entita Čísla (Number)

Podobně jako entita switch i toto je výstup ovládaný z uživatelského rozhraní Home Assistantu, tentokrát ale analogový. Pro pohodlné používání se na této entitě dá nastavit rozsah, krok, se kterým se hodnota může měnit, a i podoba zobrazení v rozhraní Home Assistantu (posuvník nebo text).

Obr.26: Entita Number

Entita Výběru (Select)

Někdy může být vhodnější namísto zadávání číselné hodnoty vybírat z textových možností. V takovém případě tedy lze použít entitu Select, která se v rozhraní Home Assistanu zobrazuje jako výběr z možností.

Entita Textového senzoru (Text sensor)

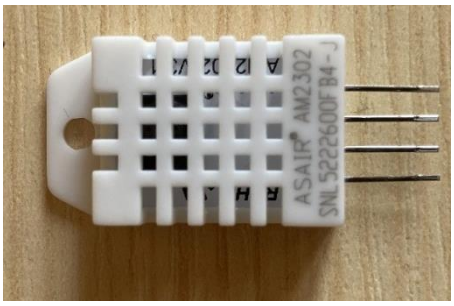
Stejně jako v předchozím případě tak i v případě senzorů (vstupů) může být někdy přívětivější zobrazovat textové zprávy namísto číselné hodnoty. Proto tedy existuje entita Text sensor, která na základě požadavku zobrazuje přednastavenou textovou zprávu.

Obr.27: Entita Text sensor

Senzor teploty a vlhkosti (DHT)

Běžně se používá na měření teploty ve vytápěných prostorech pro účely regulace. I když lze v takových případech použít i jednodušší čidla, tak dodatečné měření vlhkosti může být užitečné zejména ve vlhkých prostorech. Tento senzor se pak společně s dalšími senzory často používá i v meteostanicích.

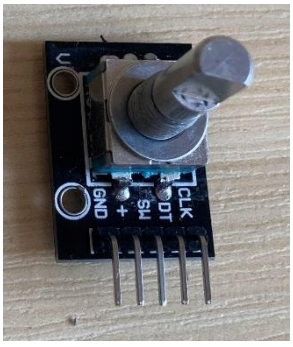
Pro zobrazení v Home Assisatntu se využívá entity Sensor, která pracuje podobně jako Binary sensor, jen namísto dvoustavové hodnoty zobrazuje floatové číslo.



Obr.28: DHT sensor

Rotační enkodér

Využití má především jako nástěnné ovladače pro regulaci teploty nebo osvětlení, kdy se otáčením enkodérem inkrementuje či dekrementuje hodnota proměnné (požadovaná hodnota), podle níž se vykonává akční zásah (zapnutí či vypnutí vytápění). Lze ho také použít jako senzor polohy.



Obr.29: Rotační enkodér

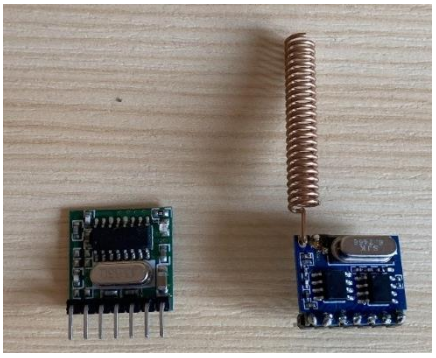
I²C/SPI sběrnice

Některým zařízením již nestačí obyčejný binární ani analogový vstup nebo výstup a vyžadují připojení přes sběrnici s určitým protokolem. Mezi ty rozšířenější sběrnice vyžívaných na ESP modulech patří například sběrnice I^2C a SPI. Tyto a podobné sběrnice mají výhodu v efektivitě komunikace a jelikož jsou všechna zařízení připojena ke stejným vodičům, dokáží i uspořit použité piny na desce.

Bezdrátový přijímač/vysílač

V některých případech není nutné, aby zařízení mezi sebou komunikovala pomocí WIFI sítě, ale stačí i jednodušší způsoby bezdrátové komunikace které mají navíc nižší spotřebu energie a jsou tedy vhodnější pro napájení z baterií. Mezi nejběžnější způsoby patří IR

(infrared) nebo 433 Mhz vysílače a přijímače. Hlavní nevýhodou těchto vysílačů oproti WIFI síti je jejich kratší dosah.



Obr.30: 433 Mhz přijímač a vysílač

Servo a krokové pohony

Chytré domácnosti nejsou jen o ovládání teploty a osvětlení, ale často je potřeba i pohyblivých částí. V aplikacích, kdy se klade důraz na přesnost při řízení polohy je vhodné použít právě servopohon (řízená pomocí PWM) nebo krokový motor. Jejich předností je zejména možnost přímého řízení polohy bez použití senzoru.

Čtečka otisku prstu a RFID čtečka (čtečka čipových karet)

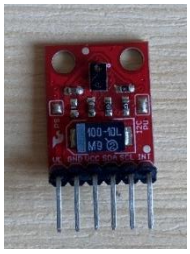
Tyto zařízení jsem zařadil do jedné kapitoly kvůli jejich podobnostem ve využití. V obou případech se jedná se zařízení schopná identifikovat uživatele podle jedinečného ID, a to v podobě otisku prstu nebo čipu ve formě karty nebo přívěsku. Obvykle se využívají v zabezpečovacích systémech. Výhodou těchto systémů je zejména pohodlnost používání ve srovnání s klasickým klíčem. Jelikož je ale zároveň mechanický zámek nahrazen elektronickým, můžou zde vzniknout obavy z hackerských útoků.

Obr.31: RFID čtečka

Obr.32: Čtečka otisku prstu

APDS Senzor (Senzor gest a RGB)

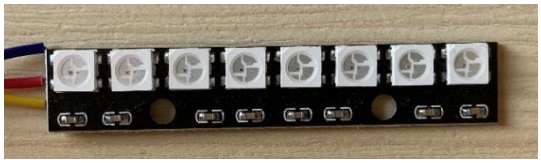
Jedná se o senzor schopný snímat úroveň světla a jeho barvy (úroveň červené, zelené a modré). Druhá funkce tohoto senzoru je schopnost snímat pohyb, a dokonce i směr pohybu (nahoru, doleva, …). Krátký dosah (jen několik centimetrů) však jeho využití v chytré domácnosti poměrně omezuje. Dá se však využít k pohodlnému ovládání některých zařízení pomocí pohybu ruky.



Obr.33: APDS senzor

LED pásky

LED pásky a podobná LED osvětlení dnes patří mezi velice populární estetické doplňky. Proto jim také sám ESP Home poskytuje vysokou podporu ve formě různých režimů blikání, nastavení intenzity a barvy osvětlení, a to i pro RGB osvětlení. Těchto přeprogramovaných funkcí se využívá poměrně často, neboť programování vlastní funkce může být poněkud zdlouhavé.

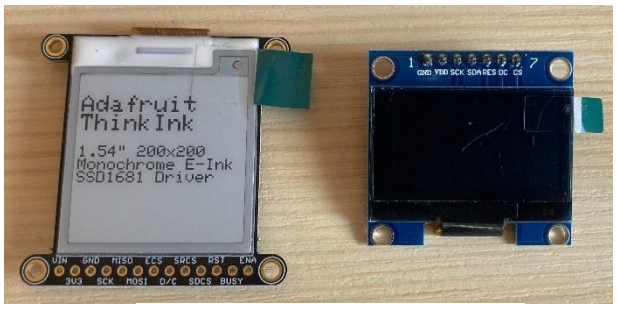


Obr.34: LED pásek

TFT a ePaper Displeje

V neposlední řadě jsem se věnoval modernějším typům displeje jako TFT nebo ePaper. Ty nejen že zobrazují na pohled příjemně čitelný text. Zároveň jsou ale schopné i vykreslovat

obrázky z vložené SD karty, a to vše v barvách. Vzhledem k jejich velikosti mají využití například jako nástěnné termostaty nebo jako ukazatele údajů z meteostanic.



Obr.35: E-paper a TFT displej

7.2. Navržení funkce vlastních chytrých zařízení

Další část práce se týkala využití nových poznatků na vytvoření vlastních chytrých zařízení. Nabízela se tedy otázka, jaká využití a funkce by měla tato zařízení mít. Hlavním cílem práce bylo formou “bastlení“ vytvořit chytré zařízení za nižší cenu než zařízení srovnatelných parametrů, která jsou běžně k dostání v obchodech. Byl jsem tedy v tomto ohledu limitován prostředky.

Nakonec jsem se rozhodl vytvořit chytrý termostat a jakési univerzální ovládání schopné řídit garážová vrata či rolety.

7.3. Návrh termostatu

Použité komponenty a základní funkce

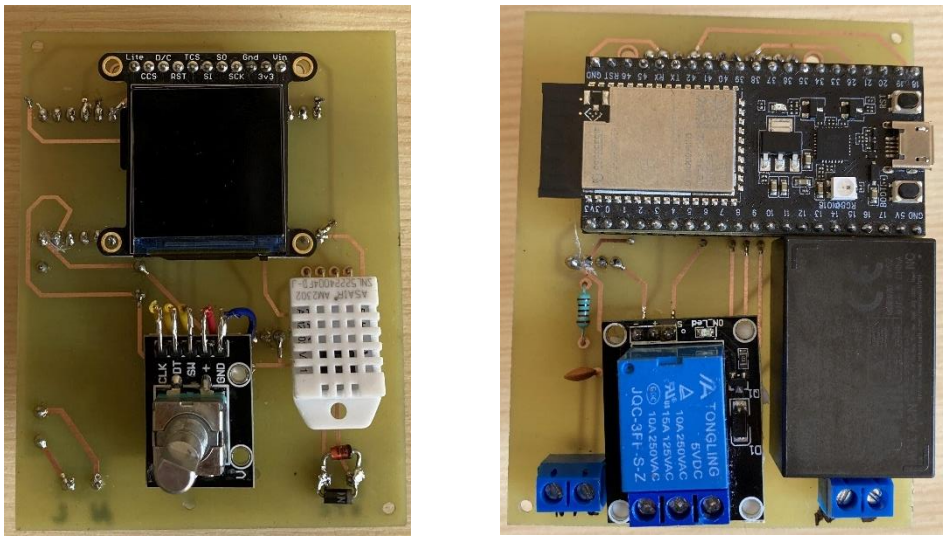
Hlavní funkcí termostatu je regulace teploty. V tomto případě se jedná nástěnný termostat určený k regulaci teploty vnitřních obytných prostor. Je tedy zřejmé že by zařízení mělo obsahovat teplotní čidlo. Moderní termostaty ale zároveň dokážou měřit i vlhkost, proto jsem použil senzor DHT22 schopný měřit obě veličiny zároveň. Termostat dále obsahuje TFT displej (ePaper jsem zamítl kvůli absenci podsvícení), na kterém je zobrazena nastavená teplota, naměřená teplota v místnosti, vlhkost v místnosti a stav termostatu (topí netopí, termostat vypnut). Nastavování teploty probíhá pomocí rotačního enkodéru s tlačítkem, které jsem použil na vypínání termostatu a “probouzení“ displeje, který po určité době nečinnosti zhasne. Pokud se jedná o prostor s podlahovým vytápěním je vhodné mít v podlaze umístěné další teplotní čidlo kvůli zajištění, aby topné těleso nepřesáhlo maximální povolenou teplotu. Proto také termostat obsahuje svorky pro připojení 10k NTC čidla. Topné těleso je pak pro jednoduchost

spínáno relé modulem. Všechny části jsou umístěny na oboustranném plošném spoji, který samozřejmě obsahuje i vývojovou desku ESP32, která zajišťuje řízení celého termostatu a umožňuje komunikaci se systémem Home Assisatnt pomocí Wifi. V tomto případě byla použita deska s energeticky úpornějším modulem ESP32-S2.

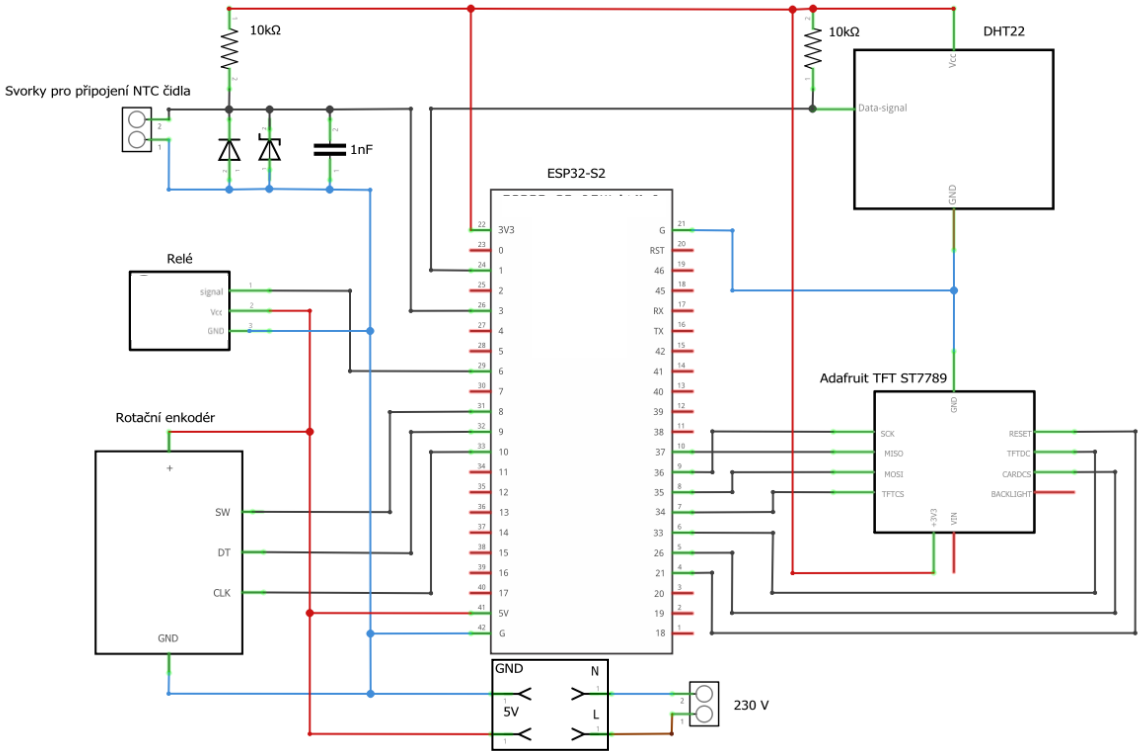
Obr.36: Vlastní chytrý termostat

Parametry komponent použitých na plošném spoji

Spínaný zdroj Input: 230VAC 0,7A Output: 5VDC 0,6A
ESP32-S2 Development board 2,4 GHz Wifi + Bluetooth, Napájení: 5VDC, 240mA
Relé modul 1-kanálový (250VAC/30VDC 10A) Napájení: 5VDC, 60mA
Rotační enkodér s tlačítkem, 20 pulzů na otáčku, Napájení: 5VDC
DHT22 senzor, teplotní rozsah: -40 - 80°C, vlhkostní rozsah: 5 – 99%, přesnost teploty: 0,5°C, přesnost vlhkosti: 2%, Napájení: 3,3/5VDC 1,5mA
NTC čidlo 10k (při 25°C), přesnost: 0,5°C, Beta: 3950, Rozsah: -20 - 105°C
TFT displej Adafruit ST7789 1,3“, slot pro SD paměťovou kartu, komunikace SPI sběrnicí, Napájení: 5VDC, 25mA



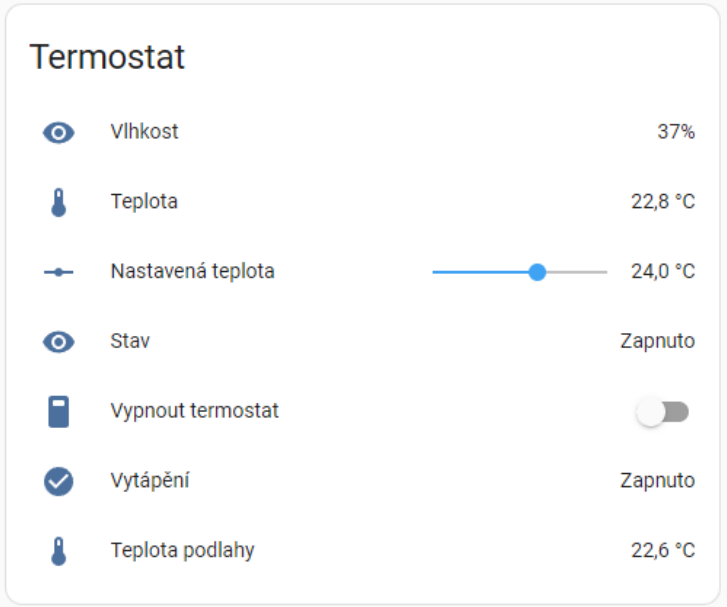
Obr.37: Osazené plošné spoje vlastního chytrého termostatu



Obr.38: Schéma vlastního chytrého termostatu

Ovládání z rozhraní Home Assistantu

V rozhraní Home Assistantu se zobrazují všechny hodnoty, které se zobrazují i na displeji (nastavená teplota, naměřená teplota v místnosti, naměřená vlhkost v místnosti a stav). Zároveň je zde možné i sledovat naměřenou teplotu podlahy, tato možnost je ale určená spíše jen pro servisní účely. V Home Assistant je možné sledovat i časový průběh všech zmíněných hodnot.



Obr.39: Rozhraní v Home Assistantu pro řízení vlastního chytrého termostatu

V sekci automatizace se pak nachází vzor pro nastavení harmonogramu, kdy se nastavená teplota mění v závislosti na denní době. Například v době, kdy se v obydlí pravidelně nikdo nenachází, je vhodné teplotu snížit a šetřit tak energie.

Funkční popis

Regulace teploty probíhá pomocí dvoustavového regulátoru s hysterezí 0,1 °C. K sepnutí relé (spuštění vytápění) tedy dojde v případě, že naměřená teplota klesne o 0,1 °C pod teplotu nastavenou. Relé se rozepne v momentě, kdy naměřená teplota přesáhne nastavenou opět o 0,1 °C. Celý tento proces je zároveň podmíněn teplotou podlahy, která je opět vybavena hysterezí (tentokrát 0,5 °C). Tj. Pokud teplota podlahy stoupne nad maximální (29 °C) navýšenou o 0,5 °C (tj. 29 + 0,5), vytápění se zakáže. Naopak pokud teplota podlahy klesne pod maximální poníženou o 0,1 °C (tj. 29 - 0,5) vytápění se opět povolí. Relé se tedy sepne pouze v případě, že je vytápění povoleno a teplota v místnosti je menší než nastavená teplota snížená o hysterezi.

Regulační smyčka probíhá s periodou 1 sekunda, kdy v každé smyčce se provede: změření všech vstupních veličin (vlhkost, teplota místnosti, teplota podlahy), vyhodnotí a provede se akční zásah, odešlou se vybrané hodnoty do rozhraní Home Assistant a aktualizují se hodnoty na displeji.

Nastavená teplota se mění pomocí otáčení rotačního enkodéru (defaultně o 0,1°C/Pulz), nebo virtuálním posuvníkem v rozhraní Home Assistant. Pokud je nastavená teplota upravena enkodérem, vyšle se zpráva Home Assisatntu, která aktualizuje tuto hodnotu v rozhraní (podobný úkon se vykoná i při vypnutí termostatu tlačítkem na enkodéru viz níže).

Po stisknutí tlačítka, nebo po otočení enkodérem dojde k rozsvícení displeje a zobrazení vlhkosti, naměřené teploty nastavené teploty a stavu termostatu. Pokud není enkodérem vykonána žádná akce po dobu 30 sekund displej se opět vypne. Pokud se tlačítko drží stisknuté po dobu 3 sekund termostat se přepne do stavu vypnuto, ve kterém přestane vytápět. Po opětovném držení tlačítka po dobu 3 sekund se termostat opět zapne. Tento stav lze zároveň přepínat virtuálním přepínačem v rozhraní Home Assistant.

Popis algoritmu

Regulační smyčka

Podmínka – čas od posledního akčního zásahu regulátoru je větší nebo rovna 1 sekundě:

Načtení hodnot ze senzorů teploty, vlhkosti a teploty podlahy a provedení jejich korekce (snížení naměřené teploty o 3°C kvůli teplu z procesoru a odfiltrování NAN hodnot z DHT22 senzoru)

Podmínka – teplota podlahy se zvýšila nad teplotu povolenou navýšenou o hysterézi (29°C + 0,5°C) a proměnná HeatAllow = true:

Proměnná HeatAllow = false (vytápění zakázáno)

Podmínka – teplota podlahy se snížila pod teplotu povolenou sníženou o hysterézi (29°C – 0,5°C) a proměnná HeatAllow = false:

Proměnná HeatAllow = true (vytápění povoleno)

Podmínka – HeatAllow = true (vytápění povoleno):

Podmínka – naměřená teplota je menší než nastavená teplota snížená o hysterézi (naměřená teplota < nastavená teplota – 0,1)

Proměnná HeatState = true (relé spínající topné těleso může být sepnuto)

Podmínka – naměřená teplota je větší než nastavená teplota zvýšená o hysterézi (naměřená teplota > nastavená teplota + 0,1)

Proměnná HeatState = false (relé spínající topné těleso musí být rozepnuto)

Podmínka – termostat je zapnut:

Proměnná HeatState se zapíše na výstup relé (HeatState = true -> relé sepnuto, HeatState = false -> relé rozepnuto)

Podmínka – termostat je vypnut:

Relé je rozepnuto

Zobrazení teploty, vlhkosti, teploty podlahy, proměnné HeatState a zprávy, zda je termostat vypnut/zapnut v rozhraní Home Assistantu

Reakce na otáčení rotačním enkodérem

Událost – rotační enkodér pootočen ve směru hodinových ručiček:

Nastavená teplota zvýšena o 0,1°C

Zobrazení nové nastavené teploty v rozhraní Home Assistantu

Obrazovka displeje se rozsvítí

Časovač zhasínaní displeje se vynuluje

Událost – rotační enkodér pootočen proti směru hodinových ručiček:

Nastavená teplota snížena o 0,1°C

Zobrazení nové nastavené teploty v rozhraní Home Assistantu

Obrazovka displeje se rozsvítí

Časovač zhasínaní displeje se vynuluje

Reakce na stisknutí tlačítka na enkodéru

Událost – tlačítko na enkodéru stisknuto (při náběžné hraně):

Obrazovka displeje se rozsvítí

Časovač zhasínaní displeje se vynuluje

Událost – tlačítko stisknuto po dobu 3 sekund:

Podmínka – termostat je vypnut:

Termostat je zapnut

Zobrazení zprávy o zapnutí termostatu v rozhraní Home Assisantu

Přepnutí virtuálního přepínače “Vypnout Termostat“ do polohy OFF

Podmínka – termostat je zapnut:

Termostat je vypnut

Zobrazení zprávy o vypnutí termostatu v rozhraní Home Assisantu

Přepnutí virtuálního přepínače “Vypnout Termostat“ do polohy ON

Displej

Událost – inicializace displeje při přivedení napájení:

Zobrazení uvítacího obrázku na displeji po dobu 2 sekund

Nastavení bílého pozadí displeje

Výpis naměřené vlhkosti (v levé horní části, velikost písma: 3, barva: modrá)

Výpis naměřené teploty (ve středu displeje, velikost písma: 5, barva: černá)

Výpis nastavené teploty (ve středu dolní části, velikost písma: 4, barva: černá)

Událost – tlačítko na enkodéru stisknuto (při náběžné hraně) nebo rotační enkodér pootočen:

Obrazovka displeje se rozsvítí

Časovač zhasínaní displeje se vynuluje

Podmínka – Obrazovka displeje je rozsvícena a časovač zhasínání displeje přesáhl dobu 30 sekund:

Obrazovka displeje zhasne

Podmínka – naměřená teplota se změnila:

Aktualizace naměřené teploty na displeji

Podmínka – naměřená vlhkost se změnila:

Aktualizace naměřené vlhkosti na displeji

Podmínka – nastavená teplota se změnila:

Aktualizace naměřené teploty na displeji

Podmínka – termostat byl zapnut/vypnut nebo proměnná HeatState se změnila:

Podmínka – termostat je vypnut:

Výpis “OFF“ (v horní pravé části, velikost písma: 3, barva: červená)

Podmínka – termostat je zapnut a proměnná HeatState = true (relé sepnuto):

Výpis značky indikující že vytápění je aktivní (v horní pravé části, velikost písma: 3, barva: oranžová)

Podmínka – termostat je zapnut a proměnná HeatState = false (relé rozepnuto):

Smazání značky indikující vytápění a výpisu “OFF“

Rozhraní Home Assistantu

Událost – nastavená teplota změněna virtuálním posuvníkem:

Nastavená teplota změněna

Událost – virtuální přepínač “Vypínač termostatu“ přepnut do polohy ON:

Termostat vypnut

Událost – virtuální přepínač “Vypínač termostatu“ přepnut do polohy OFF:

Termostat zapnut

7.4 Návrh ovládání garážový vrat a rolet

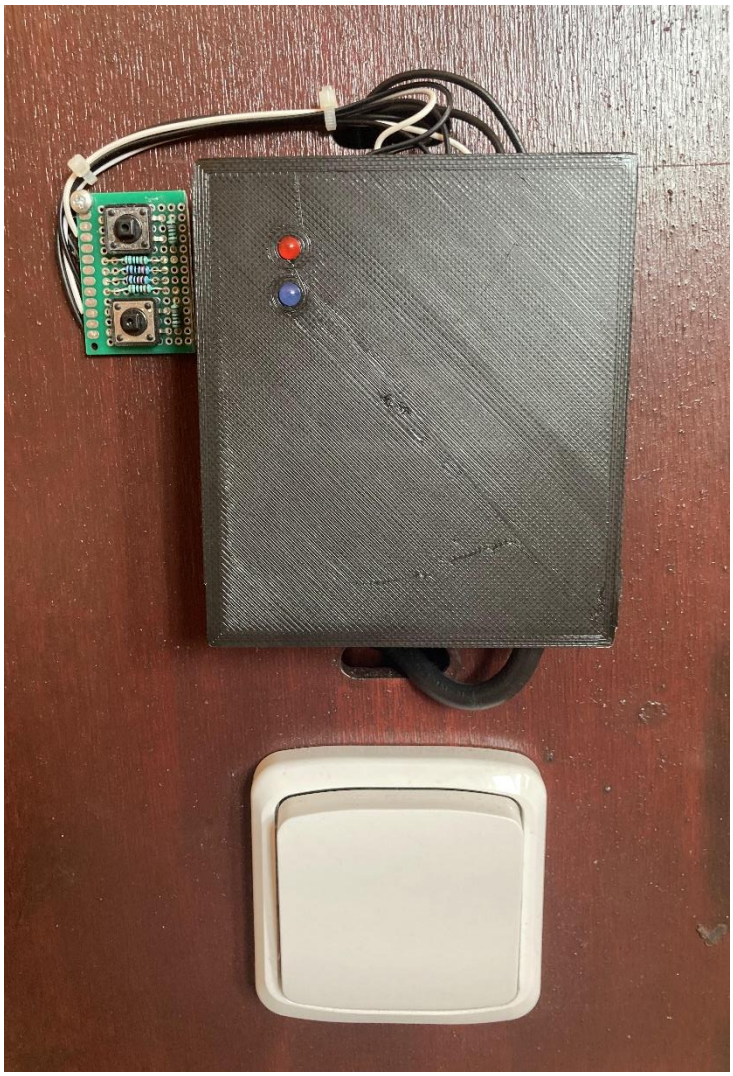
Použité komponenty a základní funkce

Toto zařízení má za úkol rozšířit možnosti ovládání garážových vrat či okenních rolet, a to především propojením s Home Assistantem. Lze tak pohodlně ovládat rolety pomocí smartphonu, aniž by k nim uživatel musel přistupovat. V případě ovládání garážových vrat se mimo vzdáleného řízení jedná i o jakousi kontrolu, zda jsou vrata otevřená či zavřená a moci je v případě nutnosti zavřít, aniž by musel být u vrat přítomen. Vzdálené řízení se pak dá využít i ke zajištění větrání pomocí režimu ventilace.

I když je vzdálené ovládání pomocí smartphonu výhodné a pohodlné, vrata i rolety by mělo být možné ovládat i pomocí nástěnných tlačítek či přepínačů. Plošný spoj má proto svorky na připojení externích tlačítek. U garážových vrat je dnes běžnou praxí používání koncových

spínačů či senzorů, které zastaví vrata při dosažení jedné z krajních poloh. Pro tento účel má tedy zařízení i svorky pro připojení těchto spínačů. Zařízení samozřejmě obsahuje i dva relé moduly pro spínání pohonu a v neposlední řadě i 433Mhz přijímač. Ten je velice praktický zejména pro ovládání garáže, když se uživatel vrací k obydlí (nebo při jeho opouštění). I když se dá vzdálené otevírání/zavírání nyní provést pomocí smartphonu, použití 433Mhz dálkového ovládání bývá při krátké vzdálenosti od vrat rychlejší a praktičtější. Jako pohon pro ovládání vrat se často používá pohon s funkcí step-by-step. Jeho řízení probíhá tedy pomocí pulzů, kdy při každém pulzu pohon změní svůj pohyb (otevírání – stop – zavírání – stop – otevírání - …). Tím se odlišuje od rolet, které mění směr pohybu v závislosti na sepnutém relé. Plošný spoj tedy obsahuje i kontakty s jumperem, který určuje, zda je zařízení využíváno pro ovládání garážových vrat, rolet či zda je chytré ovládání vyřazeno. Na závěr ještě zařízení obsahuje dvě LED diody, které indikují napájení zařízení a zda jsou vrata/rolety aktuálně v pohybu.

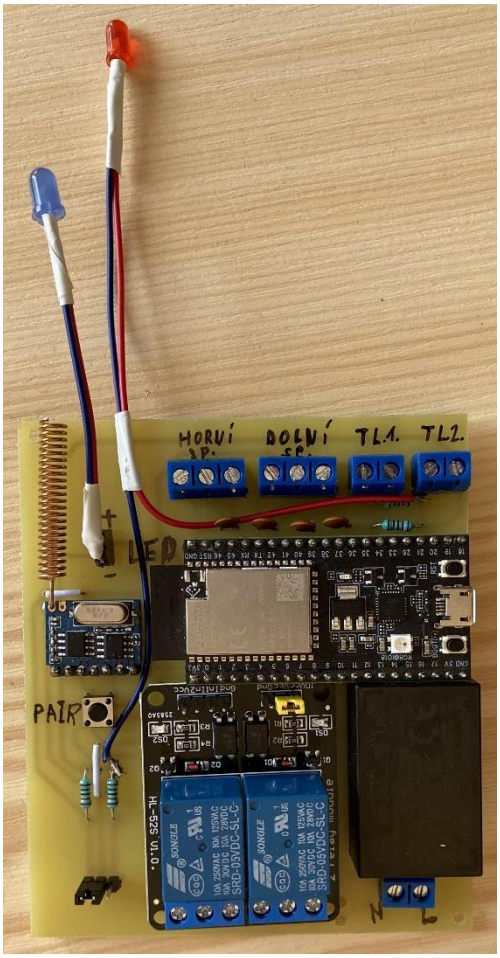
Aby bylo možné zařízení řádně otestovat, nachází se vedle něj plošný spoj s dvěma tlačítky. Tento obvod nahrazuje koncové spínače pro účely prezentace.



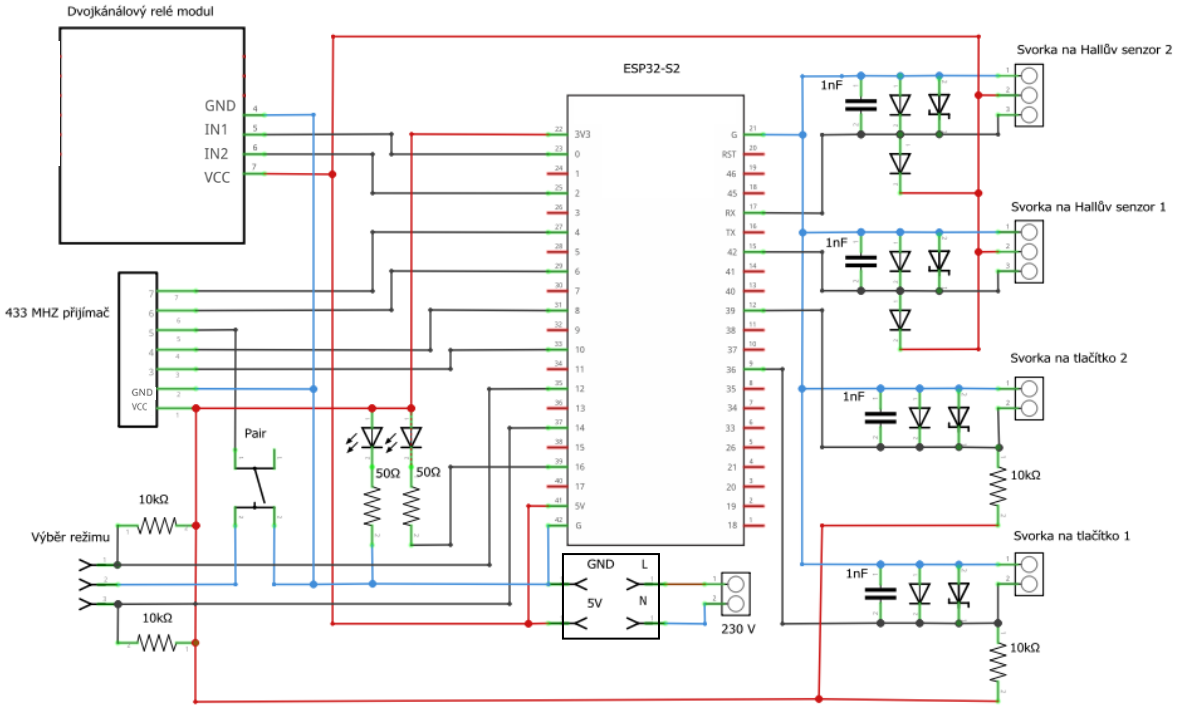
Obr.40: Vlastní chytré ovládání garážových vrat

Parametry komponent použitých na plošném spoji

Spínaný zdroj Input: 230VAC 0,7A Output: 5VDC 0,6A
ESP32-S2 Development board 2,4 GHz Wifi + Bluetooth, Napájení: 5VDC, 240mA
Relé modul 2-kanálový (250VAC/30VDC 10A) Napájení: 5VDC, 60mA
2x Vstup s pull-up rezistorem (pro připojení tlačítek)
2x Vstup pro připojení koncových spínačů
433 MHz přijímač SRX885, Napájení: 3-5VDC, 4,5mA
433 MHz dálkový ovladač 4-kanálový, Napájení: A27 baterie (12VDC)
2x LED dioda difúzní**,** 5mm, Napájení: 2VDC, 20mA



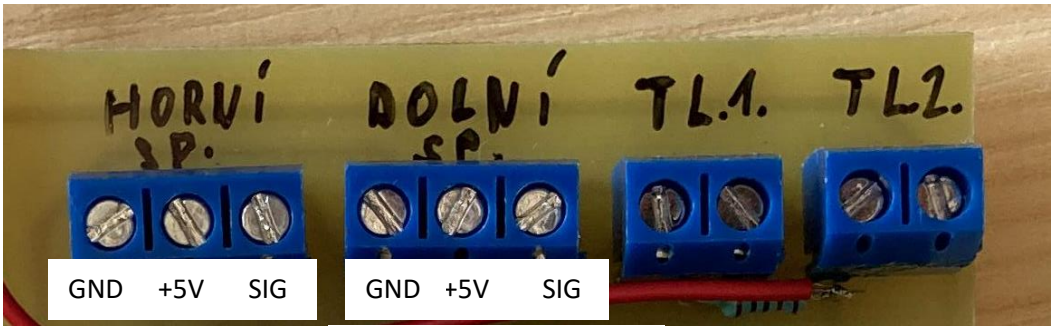
Obr.41: Osazené plošné spoje vlastního chytrého ovládání garážových



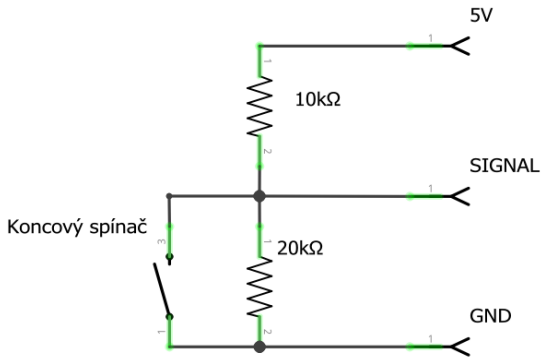
Obr.42: Schéma vlastního chytrého ovládání garážových vrat/rolet

Zapojení

Na rozdíl od termostatu, kde stačí připojit pouze napájení a podlahové čidlo zde je zapojení složitější kvůli většímu počtu vstupů a výstupů. V první řadě rozhoduje, v jakém režimu je ovládání provozováno. V režimu garážových vrat je ovládací tlačítko připojeno na pozici do druhé řady svorek zprava (TL1). Pohon je pak připojen na rozpínací svorky levého relé. Koncový spínač/senzor umístěn na spodní straně vrat se připojuje do druhé řady svorek zleva. Koncový spínač/senzor na horní straně pak do první řady svorek zleva. Svorky pro připojení koncových spínačů jsou opařena 5 V napájením pro možnost připojení hallových senzorů namísto spínačů. V obou případech je však důležité, aby zapojení umožňovalo dvouhodnotovou logiku 0-3,3V aktivní při 0V. Pro obě řady svorek platí že do levé svorky se zapojuje GND, do prostřední 5V a do pravé SIGNAL.



Obr.43: Označení svorek

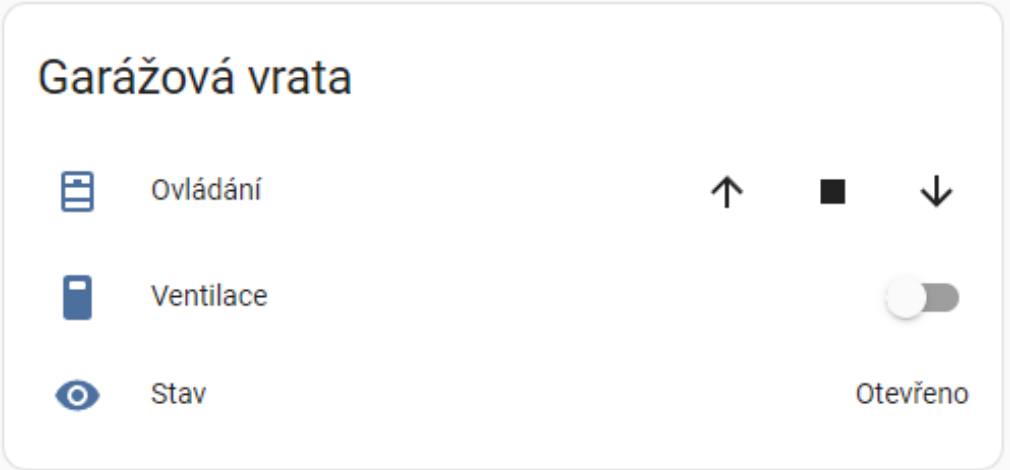


Obr.44: Ukázka správného zapojení koncového spínače

Při režimu ovládání rolet je tlačítko ovládající pohyb nahoru zapojeno do druhé řady svorek zprava (TL1) a tlačítko ovládající pohyb dolů je zapojeno do první řady svorek zprava (TL2). Rozpínací kontakty relé nalevo jsou pak určeny pro spínání pohony při pohybu nahoru a rozpínací kontakty relé napravo spínají pohon při pohybu dolů. Koncové spínače se v tomto režimu nezapojují.

Ovládání z rozhraní Home Assistantu

K ovládání rolet/vrat jsem použil entitu Cover, která je určená speciálně na ovládání garážových vrat. Obsahuje virtuální tlačítka pro pohyb nahoru dolů a zastavení pohybu. Zároveň je zde možné při ovládání garážových vrat spustit režim ventilace kdy se vrata plně zavřou na koncovou polohu a poté lehce pootevřou. Jako poslední funkce je zde pomocí entity textový senzor zobrazován stav vrat/rolet. Ten má především informovat, zda jsou vrata aktuálně otevřená či zavřená (tyto stavy jsou zjišťovány pomocí koncových spínačů, a proto se nepoužívají při ovládání rolet, kde se koncová čidla neuplatňují), zároveň informují i o pohybu vrat/rolet (otevírání, zavírání) což má sloužit jako zpětná vazba, že byl příkaz z rozhraní přijmut a zpracován.



Obr.45: Rozhraní v Home Assistantu pro řízení vlastního chytrého ovládání garážových vrat/rolet

Funkční popis

Při zapnutí či restartu zařízení se provede načtení vstupů přivedených z pinů pro jumper a vyhodnotí se tak požadovaný režim. Pokud se jumper nachází na levém a prostředním pinu, ovládání se uvede do režimu garážových vrat. Pokud se jumper nachází na pravém a prostředním pinu, ovládání se uvede do režimu rolet. Pokud je jumper zcela odejmut, ovládání se uvede do servisního režimu, kdy je chytré řízení vyřazeno a vstupy z tlačítek se propisují přímo na výstupy s relé. Tento režim má usnadnit programování speciálních funkcí garážových pohonů čemuž by chytré ovládání mohlo jinak bránit (např. nastavení detekce překážky, které se obvykle nastavuje delším držení tlačítka).

Režim garážových vrat:

Při stisknutí nástěnného tlačítka nebo tlačítka A na 433 Mhz ovladači (náběžných hranách) se sepne relé 1 po dobu 500 ms (následuje 2 sec. prodleva před dalším možným pulzem), čímž se vrata uvedou do pohybu. Vrata se zastaví při dalším stisknutí tlačítka, nebo pokud je zaznamenáno sepnutí koncového spínače (v takové případě se uvažuje že vrata zastaví sám garážový pohon). Znovustisknutím tlačítka se vrata dají do pohybu ale tentokrát v opačném směru (podmínky zastavení jsou obdobně stejné jako předtím). Při každém stisknutí tlačítka (fyzického nebo virtuálního) nebo sepnutí koncového spínače se odešle zpráva o aktualizaci stavu v rozhraní Home Assistantu (otevřeno, zavřeno, …). Pokud se vrata pohybují po dobu 2 minut, odešle se upozornění na chybu vrat. Upozornění se odešle i v případě, že se vrata namísto

zavření otevřely. To může nastat v případě, že je mezi vraty překážka a garážový pohon z bezpečnostních důvodů vrata otevře.

Dvěma stisknutím tlačítka během jedné sekundy (dvojstisknutím) se spustí režim ventilace, kdy se pohonu vysílají pulzy (s prodlevami 2,5 sekund) do doby, než se vrata začnou zavírat (vzhledem k step-by-step řízení je někdy potřeba vyslat více pulzů, než se vrata začnou zavírat). Směr je vyhodnocován podle paměti, která počítá vykonané pulzy a vyhodnocuje z nich předpokládaný pohyb a směr vrat (tato paměť se obnoví pokaždé když vrata sepnou jeden z koncových senzorů). Po sepnutí spodního koncového spínače a zastavení vrat se vyšle pulz k jejich otevření. Po 3 sekundách se opět vyšle pulz k zastavení, čímž vrata zůstanou lehce pootevřená. Zároveň se odešle zpráva Home Assistantu, která vypíše stav “Ventilace“ v rozhraní. Kterákoliv z těchto akcí se dá kdykoliv přerušit jakýmkoliv úkonem způsobující pohyb či zastavení vrat. K přerušení dojde i v případě neočekávané události (Sepnutí horního koncového spínače, nepřiměřeně dlouhému pohybu, apod). Režim ventilace se dá zároveň spustit z rozhraní Home Assistantu virtuálním přepínačem. Ten se vrátí do původní polohy, pokud byly akce vedoucí k režimu ventilace dokončeny nebo přerušeny.

Při ovládání vrat z rozhraní (entitou cover) Home Assisiatnu se opět odpočítá počet pulzů, které uvedou vrata do požadovaného pohybu. Pohyb není vykonán, pokud mu brání příslušný koncový spínač (např. zavírání vrat není možné, pokud je sepnut spodní koncový spínač).

Režim rolet:

Při stisknutí nástěnného tlačítka nahoru, nebo tlačítka A na 433 MHz ovladači (náběžné hraně) se sepne relé 1, čímž se rolety začnou zvedat. K zastavení pohybu dojde při znovustisknutí tlačítka nahoru (nástěnného nebo na 433 ovladači). Stejná situace platí i pro tlačítko dolů a tlačítko B ,které ovládají zavírání rolet pomocí relé 2. Pokud je stisknuto tlačítko nahoru během zavírání rolet, pohyb rolet se nejdříve zastaví (rozepne se relé 2) a po prodlevě 500 ms se rolety začnou zvedat (sepne se relé 1). To samé platí opět i v opačném případě. Při každém stisknutí některého z tlačítek se opět odešle zpráva do Home Assistantu o aktualizaci stavu (kvůli absenci koncových spínačů však hraje menší roli). Pokud jsou rolety v pohybu po dobu 60 sekund, jejich pohyb se automaticky zastaví rozepnutím obou relé.

Při ovládání z rozhraní Home Assistantu se rolety uvedou do pohybu při stisknutí příslušného virtuálního tlačítka (Nahoru, Dolů). Pohyb se zastaví stisknutím virtuálního tlačítka Stop (nebo opět po 60 sekundách). Virtuální přepínač Ventilace v režimu ovládání rolet neplní žádnou funkci a okamžitě se vrátí do původní polohy.

Popis algoritmu

Inicializace chytrého ovládání

Načtení vstupů přivedených z pinů jumperu

Podmínka – Pin 1 = false (levý a prostřední pin překlenuty jumperem):

Ovládání přechází do režimu garážových vrat

Zobrazení zprávy “Otevřeno“ v rozhraní Home Assistantu (výchozí stav)

Podmínka – Pin 2 = false (pravý a prostřední pin překlenuty jumperem):

Ovládání přechází do režimu rolet

Zobrazení zprávy “Nečinné“ v rozhraní Home Assistantu (výchozí stav)

Podmínka – Pin 1 = true a Pin 2 = true (jumper odebrán):

Ovládání přechází do servisního režimu

Zobrazení zprávy “Servisní režim“ v rozhraní Home Assistantu

Režim garážových vrat:

Pozn.: k předvídání pohybu vrat se používá proměnná StepByStepCount, jejíž stav se odvíjí od počtu signálů vyslaných garážovému pohonu a od spínání koncových spínačů. Popis stavů: 0 = vrata zastaveny při pohybu nahoru, 1 = pohyb dolů, 2 = vrata zastaveny při pohybu dolů, 3 = pohyb nahoru.

Událost – horní koncový spínač sepnut (náběžná hrana):

Podmínka – předpokládalo se zavírání vrat (Proměnná StepByStepCount = 1):

Zobrazení zprávy “Pozor! Vrata nezavřena“ v rozhraní Home Assistantu

Odeslání notifikace do smartphonu “Pozor! Vrata se nepodařilo zavřít. Mezi vraty je nejspíše překážka“

Přerušení všech úkonů spojených s režimem ventilace (Požadavek na režim ventilace je přerušen)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Podmínka – zavírání se nepředpokládalo (vrata se měla otevírat nebo se nepohybovat):

Zobrazení zprávy “Otevřeno“ v rozhraní Home Assistantu

Proměnná StepByStepCount = 0 (vrata zastaveny při pohybu nahoru)

Událost – spodní koncový spínač sepnut (náběžná hrana):

Zobrazení zprávy “Zavřeno“ v rozhraní Home Assistantu

Proměnná StepByStepCount = 2 (vrata zastaveny při pohybu dolů)

Podmínka – Požadavek na režim ventilace je aktivní:

Vydán požadavek na otevírání vrat (vrata jsou zavřená a nyní se pootevřou)

Událost – spodní koncový spínač rozepnut (i při použití jiného způsobu otevření než chytré ovládání):

Zobrazení zprávy “Otevřeno“ v rozhraní Home Assistantu

Událost – stisknuto nástěnné tlačítko nebo tlačítko “A“ na 433Mhz ovladači (náběžná hrana):

Přerušení režimu ventilace (a všech úkonů s ním spojených)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Podmínka – od posledního stisknutí tlačítka uběhla méně než 1 sekunda (dvojstisk):

Požadavek na režim ventilace je nyní aktivní

Podmínka – od posledního stisknutí tlačítka uběhla více než 1 sekunda:

Proměnná StepByStepCount = StepByStepCount +1

Podmínka – StepByStepCount = 4:

StepByStepCount = 0 (Proměnná má pouze stavy 0 – 3, po překročení jde opět od nuly)

Vydán signál garážovému pohonu (relé se sepne po dobu 0,5 sekundy, poté následují 2 sekundy vyčkávání, než se bude moci vyslat další signál)

Podmínka – Proměnná StepByStepCount = 1 (vrata se pohybují dolu):

Zobrazení zprávy “Zavírání“ v rozhraní Home Assistantu

Vynulování časovače měřící čas pohybu vrat

Podmínka – Proměnná StepByStepCount = 3 (vrata se pohybují nahoru):

Zobrazení zprávy “Otevírání“ v rozhraní Home Assistantu

Vynulování časovače měřící čas pohybu vrat

Podmínka – Proměnná StepByStepCount = 0 nebo 2 (vrata se nepohybují):

Zobrazení zprávy “Otevřeno“ v rozhraní Home Assistantu (vrata nebyla zastavena spodním koncovým spínačem a jsou tedy stále otevřená)

Událost – stisknuto virtuální tlačítko “Nahoru“ v prostředí Home Assistantu:

Podmínka – horní koncový spínač rozepnut

Vynulování časovače měřící čas pohybu vrat

Zobrazena zpráva “Otevírání“ v rozhraní Home Assistantu

Přerušení všech úkonů spojených s režimem ventilace (Požadavek na režim ventilace je přerušen)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Smyčka – ukončení smyčky při StepByStepCount = 3 (vrata se pohybují nahoru):

Podmínka – horní koncový spínač sepnut nebo bylo stisknuto jakékoliv tlačítko:

Ukončení smyčky (pohyb je přerušen)

Podmínka – spodní koncový spínač sepnut (náběžná hrana):

StepByStepCount = 2

StepByStepCount = StepByStepCount + 1

Podmínka – StepByStepCount = 4:

StepByStepCount = 0

Vydán signál garážovému pohonu (relé se sepne po dobu 0,5 sekundy, poté následují 2 sekundy vyčkávání, než se bude moci vyslat další signál)

Událost – stisknuto virtuální tlačítko “Dolů“ v prostředí Home Assistantu:

Podmínka – spodní koncový spínač rozepnut

Vynulování časovače měřící čas pohybu vrat

Zobrazena zpráva “Zavírání“ v rozhraní Home Assistantu

Přerušení všech úkonů spojených s režimem ventilace (Požadavek na režim ventilace je přerušen)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Smyčka – ukončení smyčky při StepByStepCount = 1 (vrata se pohybují dolů):

Podmínka – spodní koncový spínač sepnut nebo bylo stisknuto jakékoliv tlačítko:

Ukončení smyčky

Podmínka – horní koncový spínač sepnut (náběžná hrana):

StepByStepCount = 0

StepByStepCount = StepByStepCount + 1

Podmínka – StepByStepCount = 4:

StepByStepCount = 0

Vydán signál garážovému pohonu (relé se sepne po dobu 0,5 sekundy, poté následují 2 sekundy vyčkávání, než se bude moci vyslat další signál)

Událost – stisknuto virtuální tlačítko “Stop“ v prostředí Home Assistantu:

Přerušení všech úkonů spojených s režimem ventilace (Požadavek na režim ventilace je přerušen)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Zobrazení zprávy “Otevřeno“ v rozhraní Home Assistantu (vrata nebyla zastavena spodním koncovým spínačem a jsou tedy stále otevřená)

Podmínka – vrata jsou v pohybu (StepByStepCount = 1 nebo 3):

Vydán signál garážovému pohonu (relé se sepne po dobu 0,5 sekundy, poté následují 2 sekundy vyčkávání, než se bude moci vyslat další signál)

Událost – virtuální přepínač “Ventilace“ v prostředí Home Assistantu přepnut do polohy ON:

Požadavek na režim ventilace je nyní aktivní

Událost – virtuální přepínač “Ventilace“ v prostředí Home Assistantu přepnut do polohy OFF:

Požadavek na režim ventilace je přerušen

Podmínka – Požadavek na režim ventilace je aktivní (podmínka se opakuje, dokud není požadavek ukončen nebo přerušen)

Podmínka – spodní koncový spínač je v sepnutém stavu (vrata jsou zavřená):

StepByStepCount = 3

Vydán signál garážovému pohonu (vrata se nyní otevírají)

Vynulování časovače měřící čas pohybu vrat

Vyčkání 3 sekund (pokud během této doby není režim přerušen)

StepByStepCount = 0

Vydán signál garážovému pohonu (vrata nejsou v pohybu)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Zobrazení zprávy “Ventilace“ v rozhraní Home Assistantu

Požadavek na režim ventilace je ukončen

Smyčka – ukončení smyčky při StepByStepCount = 1 (vrata se pohybují dolů):

Podmínka – bylo stisknuto jakékoliv tlačítko:

Ukončení smyčky a všech úkonů režimu ventilace (Požadavek na režim ventilace je přerušen)

Podmínka – spodní koncový spínač není v sepnutém stavu:

StepByStepCount = StepByStepCount + 1

Podmínka – StepByStepCount = 4:

StepByStepCount = 0

Vydán signál garážovému pohonu

Vynulování časovače měřící čas pohybu vrat

Podmínka – vrata jsou v pohybu po dobu 2 minut (StepByStepCount = 1 nebo 3 a časovač měřící čas pohybu vrat přesáhl tuto hodnotu):

Zobrazení zprávy “Pozor! Chyba vrat“ v rozhraní Home Assistantu

Odeslání notifikace do smartphonu “Pozor! Během pohybu vrat došlo k chybě“

Přerušení všech úkonů spojených s režimem ventilace (Požadavek na režim ventilace je přerušen)

Přepnutí virtuálního přepínače “Ventilace“ v Home Assistantu do polohy OFF

Režim rolet

Událost – Stisknutí nástěnného tlačítka “Nahoru“ nebo tlačítka “A“ na 433Mhz ovladači (při náběžných hranách) nebo při stisknutí virtuálního tlačítka “Nahoru“ v rozhraní Home Assistantu:

Podmínka – pokud se roleta pohybuje směrem dolů (relé 2 je seplé):

Zastavení pohybu rolety (rozepnutí relé 2)

Vyčkání 0,5 sekundy

Podmínka – roleta se pohybuje směrem nahoru (relé 1 je seplé):

Zastavení pohybu rolety (rozepnutí relé 1)

Zobrazení zprávy “Nečinné“ v rozhraní Home Assistantu

Vyčkání 0,5 sekundy

Podmínka – roleta se nepohybuje nahoru (je nehybná nebo se před první podmínkou pohybovala směrem dolů):

Vynulování časovače měřící čas pohybu rolety

Uvedení rolety do pohybu nahoru (sepnutí relé 1)

Zobrazení zprávy “Otevírání“ v rozhraní Home Assistantu

Vyčkání 0,5 sekundy

Událost – Stisknutí nástěnného tlačítka “Dolů“ nebo tlačítka “B“ na 433Mhz ovladači (při náběžných hranách) nebo při stisknutí virtuálního tlačítka “Dolů“ v rozhraní Home Assistantu:

(analogicky stejné podmínky a akce jako v předchozím případě)

Podmínka – roleta je v pohybu (relé 1 nebo relé 2 je sepnuto) déle než 60 sekund (časovač měřící čas pohybu rolety překročil tuto hodnotu):

Zastavení pohybu rolety (rozepnutí relé 1 a relé 2)

Událost – Stisknutí virtuálního tlačítka “Stop“ v rozhraní Home Assistantu:

Zastavení pohybu rolety (rozepnutí relé 1 a relé 2)

Zobrazení zprávy “Nečinné“ v rozhraní Home Assistantu

Vyčkání 0,5 sekundy

Událost – přepnutí virtuálního přepínače “Ventilace“ do polohy ON:

Přepnutí virtuálního přepínače “Ventilace“ do polohy OFF (v režimu rolet nemá žádnou funkci)

<u>Servisní režim</u>

Událost – Stisknutí nástěnného tlačítka (náběžná hrana):

Sepnutí relé 1

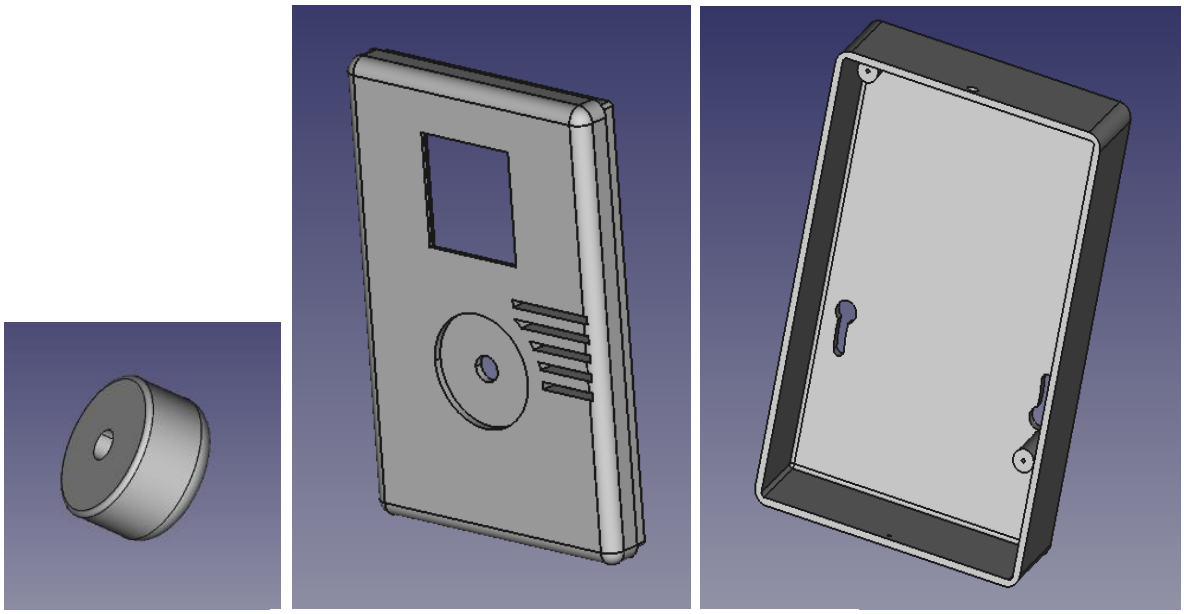
Událost – Uvolnění nástěnného tlačítka (sestupná hrana):

Rozepnutí relé 1

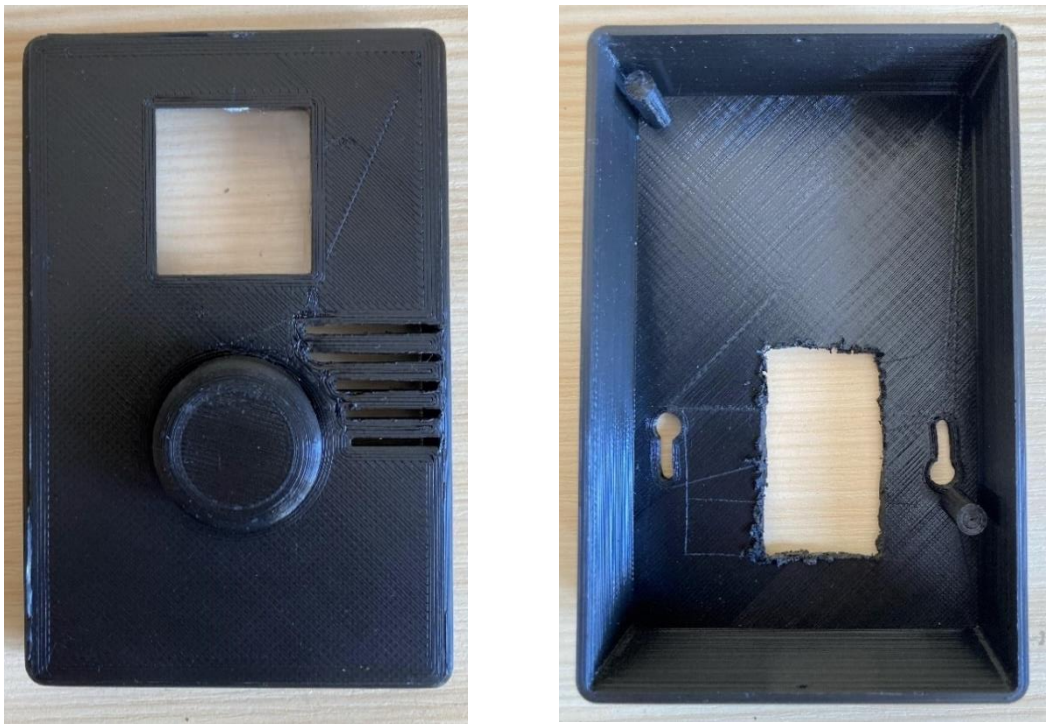
7.5 Návrh a zhotovení krytů

Kvůli ochraně plošných spojů a vzhledu celkového zařízení je vhodné plošné spoje umístit do ochranných krytů. Vzhledem k náplni práce by tyto kryty měly být vyrobeny co možná nejlevněji, zároveň by ale způsob výroby měl být co nejdostupnější pro většinu uživatel. Rozhodl jsem se tedy vytvořit tyto kryty technologií 3D tisku. Ten je v současné době velice populární a dají se pomocí něj vytvořit vzhledné výrobky, a to i složitějších tvarů.

Kryt na termostat se skládá ze tří částí. Spodní část slouží zejména k upevnění plošného spoje ke krytu. Termostat je určen k instalaci na povrch elektroinstalační krabice, spodní část tedy zároveň obsahuje otvory uzpůsobené na uchycení tímto způsobem. Termostat nemá v krytu otvory pro přivedení kabeláže, ty je potřeba opatrně vyřezat opět do spodní části dle potřeby. Vrchní část krytu je opatřena výřezem, pomocí kterého zapadá do spodní části. Tyto dvě části se následně vůči sobě zajistí dvěma šroubky v otvorech nacházející se na protilehlých stranách krytu. Vrchní část pak obsahuje otvor na hřídelku rotačního enkodéru a otvor pro displej chráněný plexisklem. Třetí část krytu tvoří ergonomický nástavec na hřídelku enkodéru.

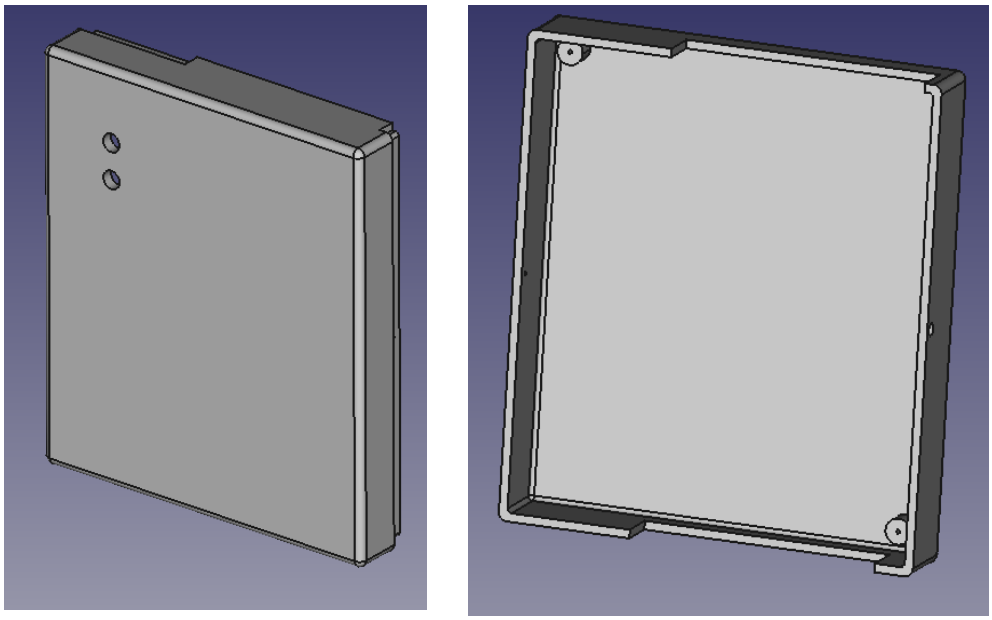


Obr.46: 3D model krytu pro chytrý termostat

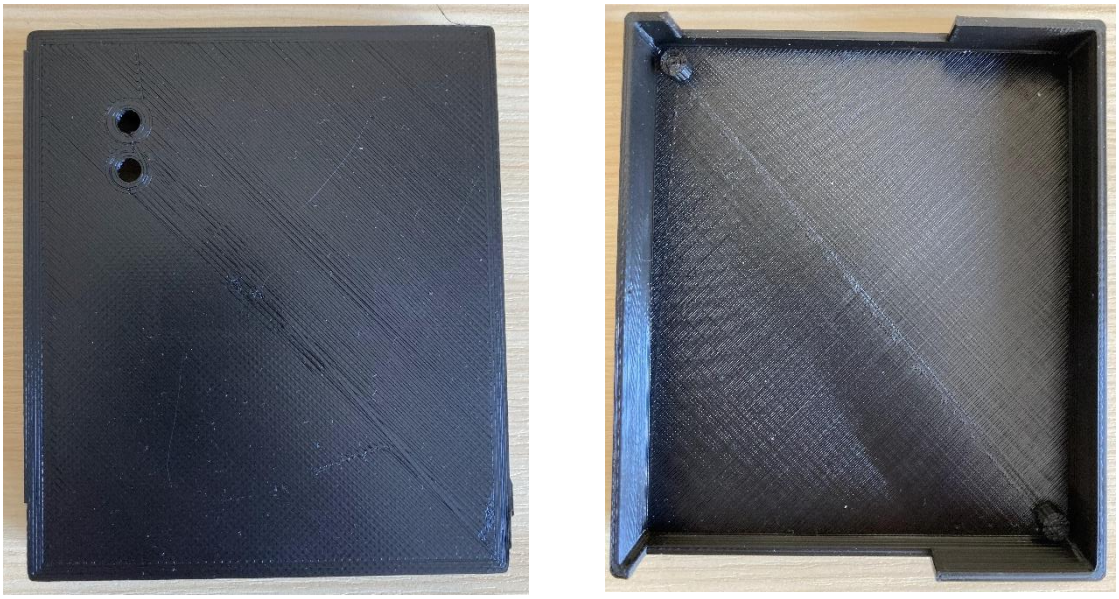


Obr.47: Kryt pro chytrý termostat (s vyřezaným otvorem pro kabeláž)

Kryt ovládání garážových vrat/rolet se skládá ze dvou částí. Spodní část opět slouží k upevnění plošného spoje ke krytu a umístění zařízení na stěnu. Kvůli svým rozměrům a pozici kabeláže se na rozdíl od termostatu nehodí na uchycení k elektroinstalační krabici a ovládání je spíše určeno na uchycení pomocí šroubů do zdi. Vrchní část krytu je opět opatřena výřezem a otvory na šroubky, aby do sebe obě části zapadaly a mohly být vůči sobě zajištěny. Vrchní část ještě obsahuje dva malé otvory na LED diody signalizující napájení zařízení a pohyb vrat/rolet.



Obr.48: 3D model krytu pro chytré ovládání garážových vrat/rolet



Obr.49: Kryt pro chytré ovládání garážových vrat/rolet

8. Návrh a realizace testovacího stanoviště

Poslední část práce se týkala demonstrace vytvořených zařízení na testovacím stanovišti. Stanoviště má za úkol představit mnou vytvořená zařízení a základní prvky chytré domácnosti prostřednictvím systému Home Assistant.

Stanoviště tedy musí v první řadě obsahovat fyzický hardware do kterého se Home Assistant nainstaluje. Jak již bylo zmíněno jedním z takovýchto nejběžnějších hardwarů je mikropočítač Raspberry Pi 4. To je zde umístěno v pouzdře Argon ONE M2, které zajišťuje mikropočítači dostatečnou mechanickou ochranu a chlazení. Zároveň poskytuje i podporu pamětem SATA SSD (nahrazujícím dosavadní SD karty) pro vyšší přenosovou rychlost a spolehlivost.

Obr.50: Raspebbry PI 4 (zapouzdřené) s nainstalovaným Home Assistantem

Komunikaci mezi jednotlivými zařízení zajišťuje router typu Ubiquiti AirRouter s interní anténou.

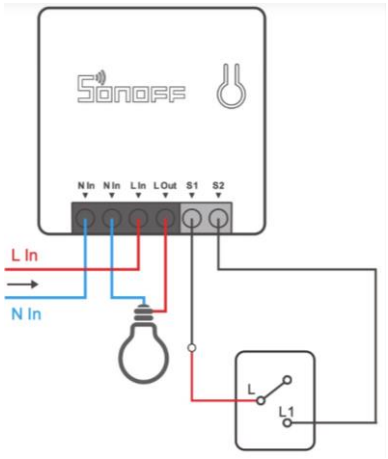
Obr.51: Router zprostředkující komunikaci mezi zařízeními

Napájení routeru a Raspberry Pi 4 je přivedeno skrze dvojitou zásuvku umístěnou na stanovišti. Stanoviště obsahuje i druhou dvojitou zásuvku, která poskytuje napájení případným externím zařízením (notebooku, tabletu, …). Obě zásuvky jsou připojeny do čtyř-modulového rozvaděče vybaveného chráničem a B6 jističem od výrobce Schneider Electric. Pomocí tohoto rozvaděče jsou zároveň napájena všechna zařízení na testovacím stanovišti.

Obr.52: Rozvaděč napájející stanoviště

Na stanovišti jsou mimo mnou vytvořených zařízení také některá typická představena chytrá zařízení, která jsou běžně k dostání v obchodech. Jedná se o chytrý vypínač Sonoff MINI, chytrou zásuvka Sonoff TH16 a chytrou žárovku Denver SHL-350. Principem těchto zařízení je možnost vzdáleného ovládání pomocí mobilní aplikace a případně i automatizování jejich provozu. Zároveň by ale ovládané prvky neměly ztratit nic z jejich dosavadní funkce. Například osvětlení by mělo být stále možné řídit pomocí nástěnných přepínačů. Vypínač Sonoff MINI má tedy i svorky na připojení tlačítek/přepínačů. Svými rozměry je dokonce vhodný k umístění do elektroinstalační krabice za tyto přepínače. Zásuvka Sonnof TH16 však takto schovat nejde a je tedy designovaná spíše na povrchové použití o čemž vypovídá i ovládací tlačítko na zařízení (což by byl naopak problém pro Sonoff vypínač, kvůli nedostatečnému IP krytí). Zásuvka má navíc možnost připojení senzoru teploty a vlhkosti, čímž

může plnit zároveň funkci termostatu. Výhodou chytré žárovky je zejména možnost řídit intenzitu, barvu, a i teplotu vydávaného světla.



Obr.53: Schéma zapojení vypínače Sonoff MINI [42]

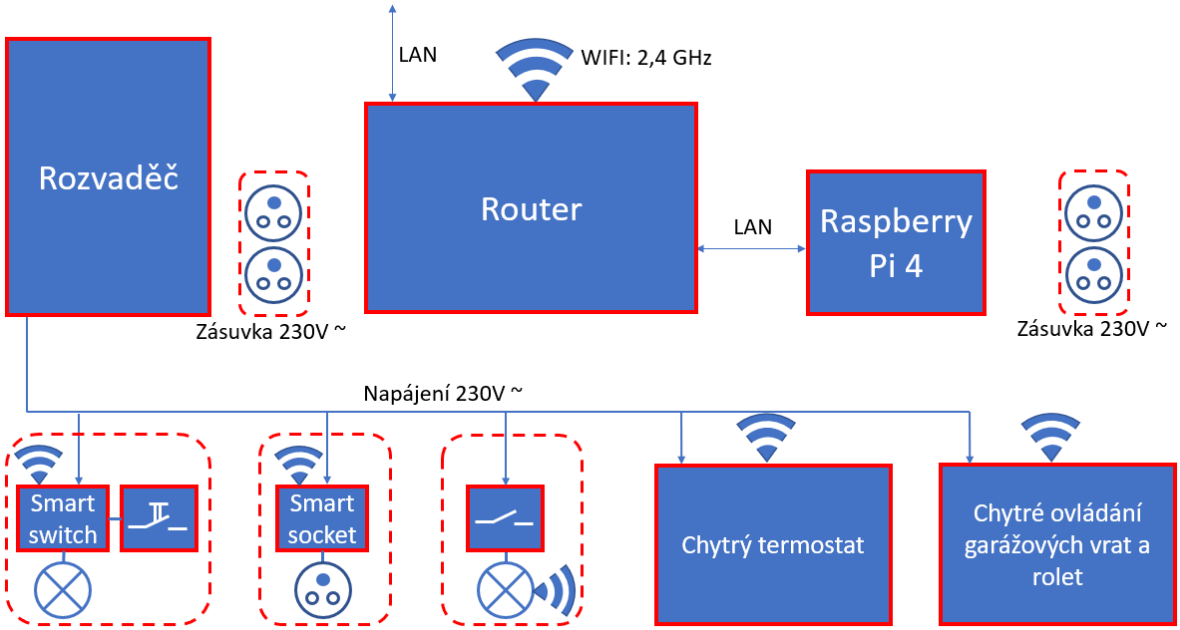
Obr.54: Sestava Sonoff MINI na testovacím stanovišti

Obr.55: Schéma zapojení chytré zásuvky Sonoff TH16 [41]

Obr.56: Sestava Sonoff TH16 na testovacím stanovišti

Obr.57: Chytrá žárovka Denever SHL-350

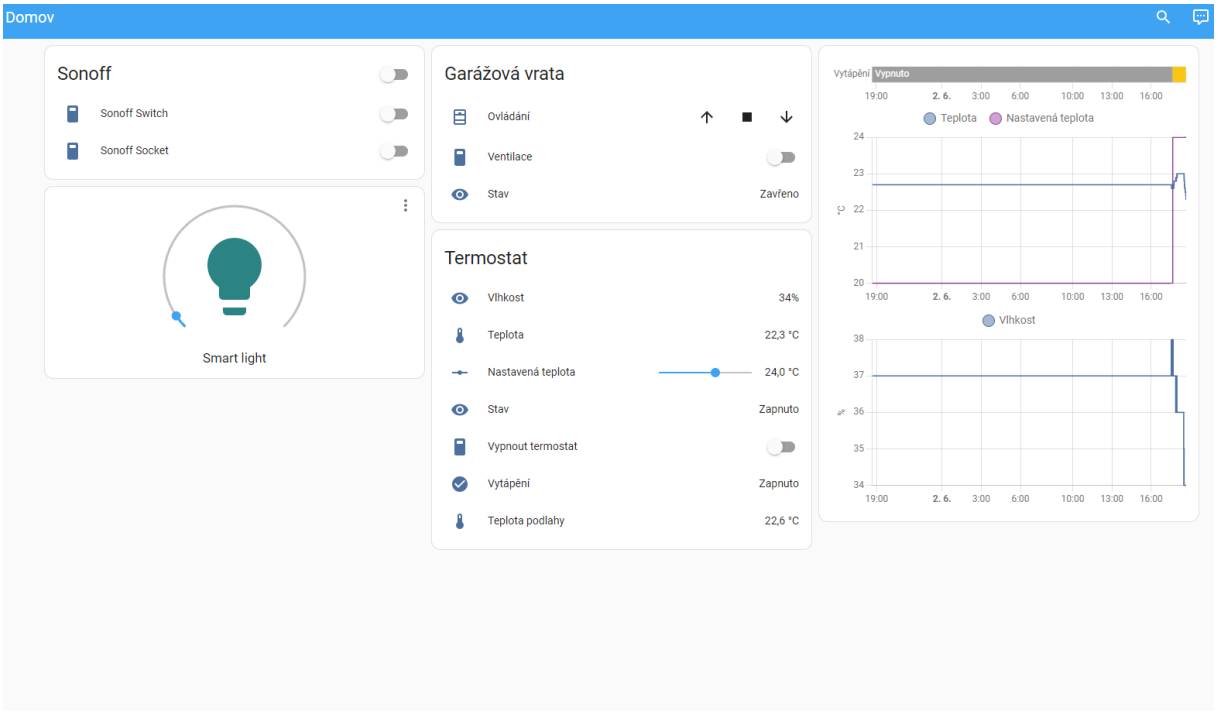
Samotné stanoviště se skládá z několika dřevěných překližek o tloušťce 8 mm, natřených olejovou lazurou pro ochranu dřeva a příjemnější vzhled. Tyto překližky dohromady tvoří dutou stěnu s vyřezanými otvory pro upevnění a přístupnost všech prezentovaných součástí testovacího stanoviště. Výhodou duté stěny je především úhledné zakrytí kabeláže bez použití instalačních lišt a možnost použít zapuštěné elektroinstalační krabice na uchycení elektroinstalačních zařízení. Většina stěn je k sobě pevně připevněna pomocí dřevěných zámků. Výjimkou je zadní stěna, která je odnímatelná kvůli případné údržbě. Na zadní stěně se také nachází konektor pro připojení napájení 230 V a Ethernet konektor pro připojení PC k Home Assisatntu (prostřednictvím LAN). O stabilitu stanoviště se stará TV stojan od výrobce My Wall.



Obr.58: Blokové schéma testovacího stanoviště

Obr.59: Přední pohled na testovací stanoviště

Obr.60: Zadní pohled na testovací stanoviště



Obr.61: Rozhraní Home Assistentu pro ovádání stanoviště

9. Závěr

V práci jsem se zabýval vývojem vlastních chytrých zařízení, které je možné integrovat do open-source systému Home Assisatnt poskytující software pro chytrou domácnost. Tato zařízení spolu celistvým systémem chytré domácnosti měla ukázat možnosti chytré domácnosti, zvláště pak možnost vyrobit chytrá zařízení na míru pro aplikace, u kterých by zakupování hotových zařízení mohlo být problematické. Použitý materiál a způsob výroby měl být zároveň dostupný co nejvíce lidem, kteří by o podobný systém měli zájem. Část práce se proto také zabývala popisem systému Home Assistant a jeho funkcí.

Mnou vytvořená a další chytrá zařízení od výrobců Sonoff a Denever jsem vestavěl do mnou vyrobeného testovacího stanoviště, jehož účel je demonstrovat výhody a možnosti chytré domácnosti. Dutá stěna zakrývající kabeláž působí velice esteticky a hnědá barva dřeva vytváří příjemný kontrast s bílými komponenty. Systém chytré domácnosti se na stanovišti dá dobře demonstrovat a může být využito i k výukovým účelům.

Použitá technologie 3D tisku se nevyrovná průmyslovým technologiím používaných při výrobě moderních zařízení, přesto si myslím že v rámci možností vypadají velice dobře. Zařízení svou funkci navíc plní dle požadavků i když je zde stále prostor pro zlepšení.

Po získaných zkušenostech bych navrhování a vývoj vlastních chytrých zařízení zhodnotil následovně. Hlavní nevýhodou, která od výroby vlastních zařízení může odrazovat je časová náročnost. Vytvořit plně funkční zařízení vyžaduje trpělivost, a to jak při návrhu, tak i během programování jeho funkcí. Zvláště pro nezkušené programátory může být druhý bod velice odrazující, neboť studování YAML struktury ESPHome bylo hlavně v počátcích dosti zdlouhavé. I při návrhu jsem několikrát narazil na problém, kvůli kterému jsem část návrhu předělat, což se opět projevilo na časové náročnosti.

Hlavním pozitivem je již zmíněná možnost vytvoření zařízení sobě na míru za poměrně nízkou cenu. Celkové náklady na mnou vyrobený termostat vychází přibližně na 900 – 1000 Kč, zatímco termostaty srovnatelných parametrů stojí běžně 1200 – 2000 Kč (nebo i více). Podobnou cenu má i mé ovládání garážových vrat/rolet.

10. Použitá literatura

[1] Ing. et Ing. OREL Daniel, Ph.D. *Smart Factory co nám přinese?* [online]. [cit. 17.8.2021] Dostupné z: https://plasticportal.cz/cs/smart-factory-%E2%80%93-co-nam-prinese.html/c/3176/

[2] DOSTÁL Dalibor. *Rychleji, levněji, úsporněji. Průmysl 4.0 slibuje větší zisky a šetrnější provoz* [online]. [cit. 17.8.2021] Dostupné z: https://businessinfo.cz/clanky/rychleji-levneji-usporneji-prumysl-40-slibuje-vetsi-zisky-a-setrnejsi-provoz/

[3] Ing. MARCOŇ Petr, Ph.D. *Průmysl 4.0* [online]. [cit. 17.8.2021] Dostupné z: https://docplayer.cz/40631959-Prumysl-4-0-industry-4-0.html

[4] BÍLIK Peter. *Čtyři klíčové principy inteligentního průmyslu* [online]. [cit. 17.8.2021] Dostupné z: https://systemonline.cz/rizeni-vyroby/4-klicove-principy-prumysloveho-internetu-veci.htm

[5] VIJAYAN Jaikumar. *Bezpečnost průmyslových řídících systémů* [online]. [cit. 17.8.2021] Dostupné z: https://computerworld.cz/securityworld/bezpecnost-prumyslovych-ridicich-systemu-56059

[6] Ing. LOM Michal, doc. Ing. PŘIBYL Ondřej. *Rizika chytrých domácností a jejich zabezpečení* [online]. [cit. 17.8.2021] Dostupné z: https://elektro.tzb-info.cz/inteligentni-budovy/15569-rizika-chytrych-zarizeni-a-jejich-zabezpeceni

[7] ŠKODA Storyboard. *Asistenční systémy ŠKODA* [online]. [cit. 17.8.2021] Dostupné z: https://skoda-storyboard.com/cs/inovace-a-technologie/polopate-asistencni-systemy-skoda/

[8] ADAS. *Základní pojmy v oblasti dopravních informačních systémů* [online]. [cit. 17.8.2021] Dostupné z: http://adas.upol.cz/o-adas.html#prinosy-a-rizika

[9] ROSENKRANC Pavel. *AI ve zdravotnictví usnadní práci a pomůže s diagnostikou* [online]. [cit. 17.8.2021] Dostupné z: https://sciencemag.cz/ai-ve-zdravotnictvi-usnadni-praci-a-pomuze-s-diagnostikou/

[10] HANDL Jan. *Umělá inteligence mění zdravotnictví* [online]. [cit. 17.8.2021] Dostupné z: https://6dhub.cz/uvod/portal/umela-inteligence-meni-zdravotnictvi,-bleskove-pocita-bunky-a-snadno-identifikuje-pacienta/

[11] *Hass.io - vytvoření a obnovení ze zálohy* [online]. [cit. 31.5.2022] Dostupné z: https://hello-future.cz/home-assistant/vytvoreni-a-obnoveni-ze-zalohy/

[12] Wikipedia. *Inteligentní dům* [online]. [cit. 17.8.2021] Dostupné z: https://cs.wikipedia.org/wiki/Inteligentn%C3%AD_d%C5%AFm

[13] Smarteon Systems. *Co je Smart Home* [online]. [cit. 17.8.2021] Dostupné z: https://smarteon.cz/co-je-smart-home/

[14] LOXONE. C*hytrý dům* [online]. [cit. 17.8.2021] Dostupné z: https://loxone.com/cscz/chytry-dum/

[15] Smarteon Systems. *Jak funguje chytrý asistent* [online]. [cit. 17.8.2021] Dostupné z: https://smarteon.cz/hlasovy-asistent-alexa-amazon/

[16] DigiDoupě. *Hlasoví asistenti* [online]. [cit. 17.8.2021] Dostupné z: https://www.digidoupe.upol.cz/index.php/digiseznam/63-hlasovi-asistenti-google-assistant-alexa-siri-a-cortana

[17] LOXONE. *Jde to i bez cloudu* [online]. [cit. 17.8.2021] Dostupné z: https://loxone.com/cscz/produkty/chytry-dum-bez-cloudu/

[18] LOXONE. *Kolik stojí chytrý dům* [online]. [cit. 17.8.2021] Dostupné z: https://loxone.com/cscz/blog/kolik-stoji-chytry-dum-cena/

[19] Economia. *Smart City* [online]. [cit. 17.8.2021] Dostupné z: http://service.ihned.cz/smartcity/

[20] UVTnet. *Smart City: Co jsou to chytrá města?* [online]. [cit. 17.8.2021] Dostupné z: https://uvtnet.cz/smart-city-co-jsou-chytra-mesta

[21] Home Assistant. *Getting Started* [online]. [cit. 17.8.2021] Dostupné z: https://www.home-assistant.io/getting-started/

[22] Hello-Future CZ. *Home Assistant kuchařka* [online]. [cit. 17.8.2021] Dostupné z: https://hello-future.cz/home-assistant-kucharka/

[23] TátaGeek. *Co je Home Assistant* [online]. [cit. 17.8.2021] Dostupné z: https://tatageek.blog/2020/05/26/co-je-home-assistant/

[24] HAVEL František. *Chytrá domácnost Home Assistant – Úvod, info, instalace* [online]. [cit. 17.8.2021] Dostupné z: https://havel.mojeservery.cz/chytra-domacnost-home-assistant-uvod-info-instalace/

[25] *How to create switch to control relay on Home Assistant*[online]. [cit. 31.5.2022] Dostupné z: https://www.kincony.com/how-to-create-switch-to-control-relay-on-home-assistant.html

[26] ŠKODA AUTO. *Průmysl 4.0* [online]. [cit. 17.8.2021] Dostupné z: https://zakazka.cz/prumysl-4-0-skoda-auto-v-zavode-ve-vrchlabi-vyuzila-digitalni-dvojce/

[27] HALE Scott. *Industry 4.0, smart manufacturing and the power of data* [online]. [cit. 17.8.2021] Dostupné z: https://plant.ca/technology-centre/industry-4-0-smart-manufacturing-and-the-power-of-data/

[28] AUSTIN Michael. *Audi lights the way with smart LED headlights* [online]. [cit. 17.8.2021] Dostupné z: https://popularmechanics.com/cars/a9477/frankfurt-auto-show-audi-lights-the-way-with-smart-led-headlights-15912196/

[29] LOXONE. *Inteligentní řízení vnitřního klima* [online]. [cit. 17.8.2021] Dostupné z: https://www.loxone.com/cscz/produkty/topeni-klimatizace/

[30] Onlinesense. *Smart Home hack* [online]. [cit. 17.8.2021] Dostupné z: http://onlinesense.org/smart-home-hack/

[31] LOXONE. *Loxone Miniserver* [online]. [cit. 17.8.2021] Dostupné z: https://www.loxone.com/cscz/produkty/miniserver-extensions/

[32] Internet of business. *Global smart city platform market* [online]. [cit. 17.8.2021] Dostupné z: https://internetofbusiness.com/global-smart-city-platform-market/

[33] Finch and Beak. *Smart cities, smart transit: Bike shares as urban transport solution* [online]. [cit. 17.8.2021] Dostupné z: https://finchandbeak.com/1108/smart-cities-smart-transit-bike-shares.htm

[34] Sons ČR. *Chytré lavičky – nový mobiliář* [online]. [cit. 17.8.2021] Dostupné z: https://www.sons.cz/Chytre-lavicky-novy-mobiliar-P4004702.html#prettyPhoto

[35] Raspberry Pi. *Raspberry Pi3 Model B* [online]. [cit. 17.8.2021] Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[36] Home Assistant. *Demo* [online]. [cit. 17.8.2021] Dostupné z: https://demo.home-assistant.io/#/lovelace/0

[37] ESPHome. *Using with sonoff basic* [online]. [cit. 17.8.2021] Dostupné z: https://esphome.io/devices/sonoff_basic.html

[38] ESPHome. *NodeMCU ESP8266* [online]. [cit. 17.8.2021] Dostupné z: https://esphome.io/devices/nodemcu_esp8266.html

[39] KENLON Seth. *How to share files with Samba* [online]. [cit. 27.3.2022] Dostupné z: https://www.redhat.com/sysadmin/samba-file-sharing

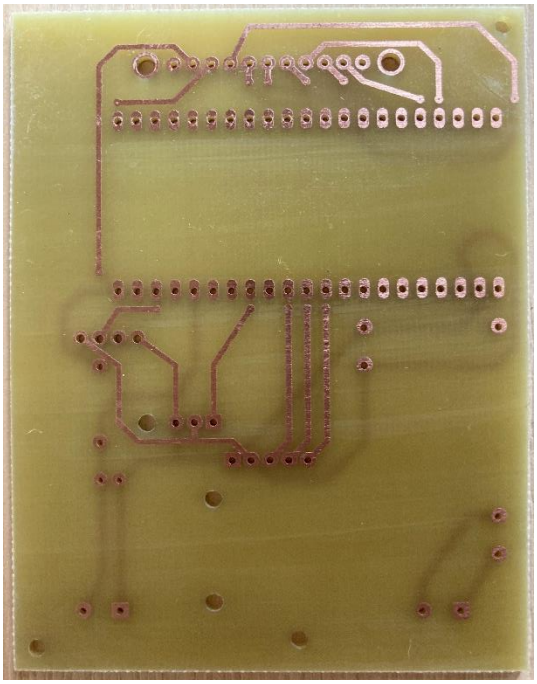
[40] HOLEC Miroslav. *YAML jako nástupce JSON konfigurace v moderním .NETu* [online]. [cit. 27.3.2022] Dostupné z: https://www.miroslavholec.cz/blog/yaml-nastupce-konfigurace-moderni-net-core

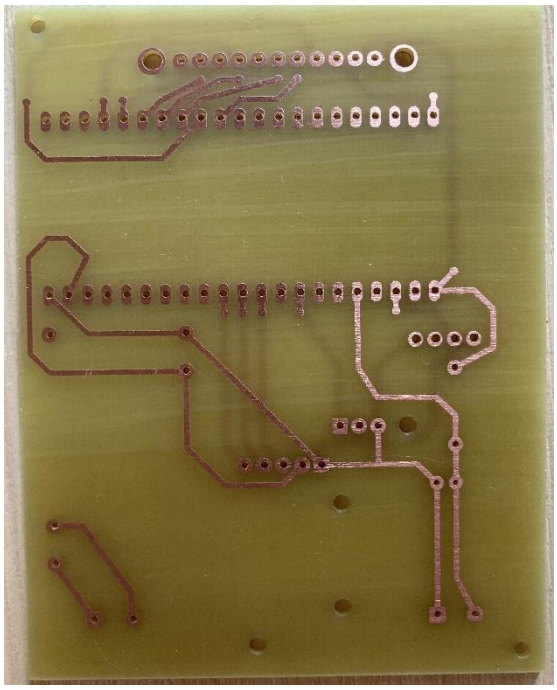
[41] Autodesk Instructables. *Sonoff TH 16 – Tuorial Completo* [online]. [cit. 20.5.2023] Dostupné z: https://www.instructables.com/Sonoff-TH-16-Tutorial-Completo/

[42] Pajenicko.cz. *Dálkově ovládané relé Sonoff MINIR2 (WIFI, 100-240 V AC 10A)* [online]. [cit. 20.5.2023] Dostupné z: https://pajenicko.cz/dalkove-ovladane-rele-sonoff-minir2-wifi-100-240v-ac-10a

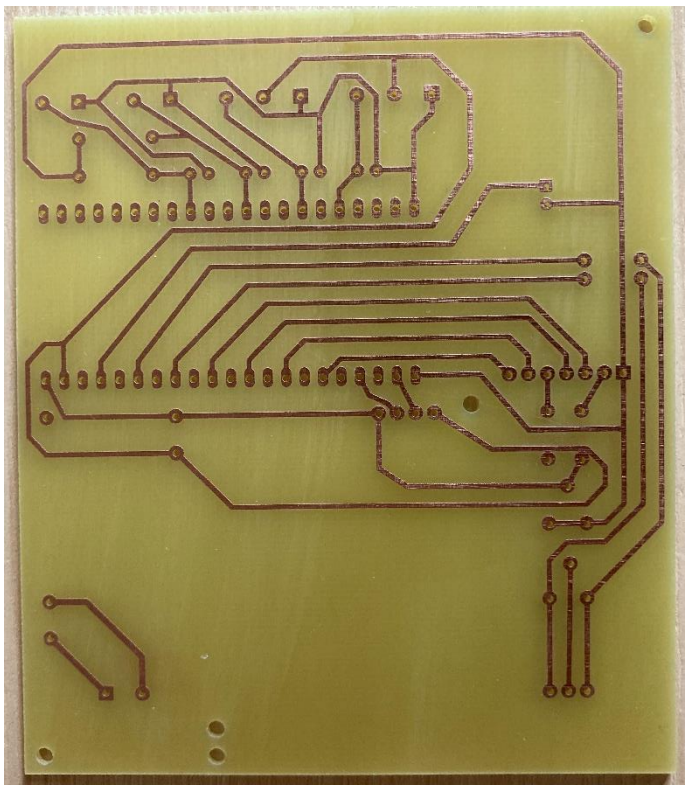
11. Přílohy

Plošný spoj termostatu

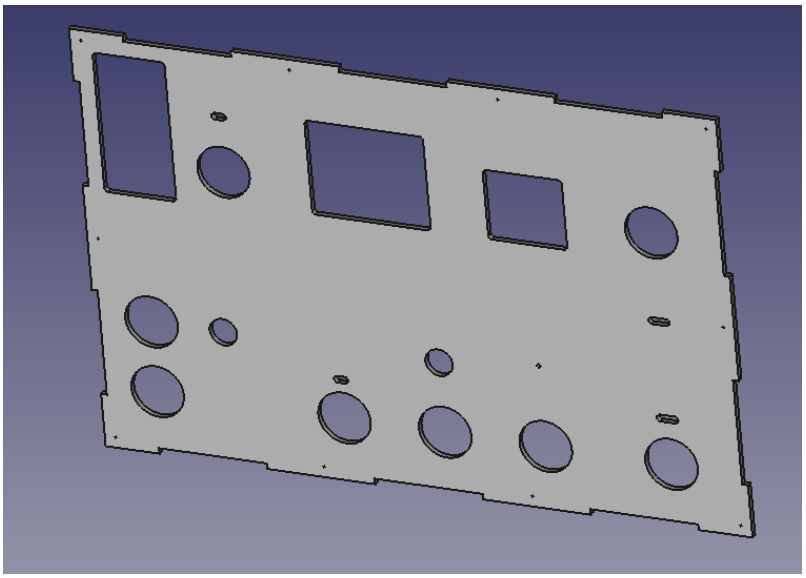


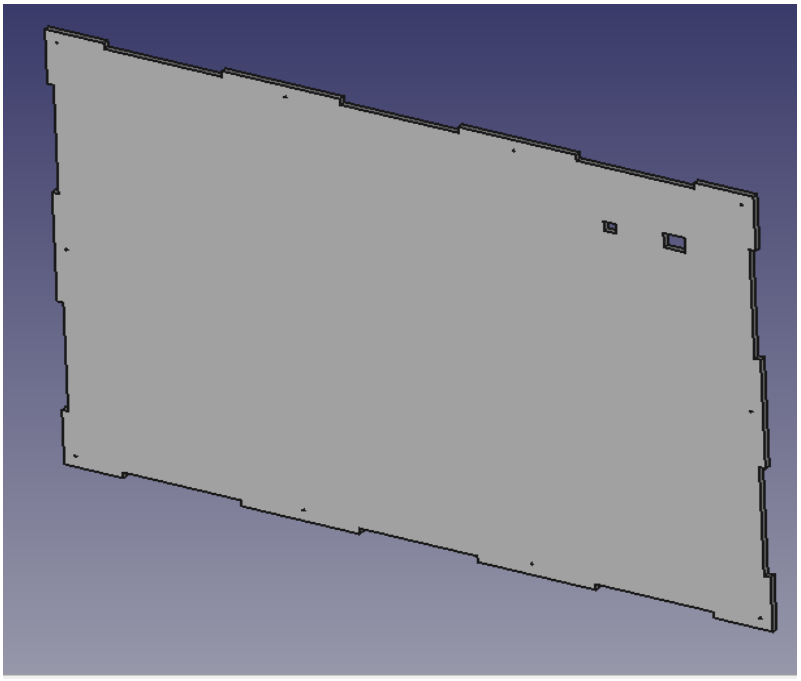


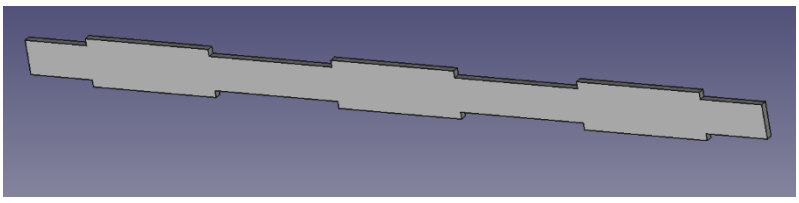
Plošný spoj ovládání garážových vrat/rolet

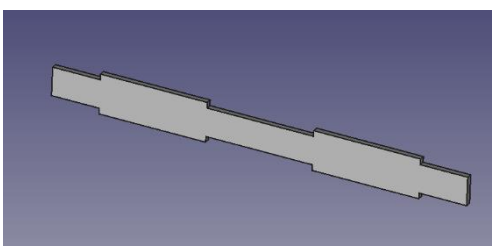


3D model stěn stanoviště









YAML kód termostatu

```yaml
esphome:
  name: termostat
  friendly_name: Termostat
  libraries:
    - "SPI"
    - "wire"
  includes:
    - MyTermostatCode.h   #vložení vlastní C++ knihovny

esp32:
  board: esp32-s2-saola-1
  framework:
    type: arduino

# Enable logging
logger:

# Enable Home Assistant API
api:
  encryption:
    key: "qvDY2oh/axHrrS7rLTSKhY9s5ua9MNjU6s8VPsI56Jw="

ota:
  password: "c6865b16c23a1d9779c028a8daefe973"

wifi:
  ssid: !secret wifi_ssid
  password: !secret wifi_password

  manual_ip:
    static_ip: 192.168.1.218
    gateway: 192.168.1.1
    subnet: 255.255.255.0

  # Enable fallback hotspot (captive portal) in case wifi connection fails
  ap:
    ssid: "Termostat Fallback Hotspot"
    password: "LyVicUccHNPY"

#mdns:
#  disabled: false

captive_portal:

#vypnutí termostatu
switch:
  - platform: template
    name: "Vypnout termostat"
    id: vypinac_termostatu
    optimistic: true

#nastavení požadované teploty
number:
  - platform: template
    name: "Požadovaná teplota"
    id: pozadovana_teplota
    optimistic: true
    min_value: 7
    max_value: 35
    step: 0.5
    initial_value: 20
    unit_of_measurement: °C

#entity pro zobrazování teplot a vlhkosti
```

```yaml
sensor:
  - platform: template
    name: "Teplota v místnosti"
    id: teplota_mistnosti
    unit_of_measurement: °C
    accuracy_decimals: 1
    update_interval: never

  - platform: template
    name: "Teplota topného tělesa"
    id: teplota_topneho_telesa
    unit_of_measurement: °C
    accuracy_decimals: 1
    update_interval: never

  - platform: template
    name: "Vlhost v místnosti"
    id: vlhkost_mistnosti
    unit_of_measurement: '%'
    accuracy_decimals: 0
    update_interval: never

#oznámení o sepnutí topného tělesa

#oznámení o sepnutí topného tělesa
binary_sensor:
  - platform: template
    name: "Žhavení topného tělesa"
    id: zhaveni

#oznamování o vypnutí/zapnutí termostatu
text_sensor:
  - platform: template
    name: "Stav termostatu"
    id: stav_termostatu
    update_interval: never

#instance C++ tříd v podobě ESPHome komponentů
custom_component:
  lambda: |-
    auto ledflash = new LED_Flash();
    auto enpublish = new EncoderPublish();
    auto stoppublish = new StopPublish();
    auto disp = new DisplayST7789(id(teplota_mistnosti), id(vlhkost_mistnosti), id(pozadovana_teplota), id(zhaveni), id(vy
    auto tcontrol = new TermostatControl(id(pozadovana_teplota), id(vypinac_termostatu));
    return {ledflash, enpublish, stoppublish, disp, tcontrol};

  components:
    - id: led_flash_termostat
    - id: encoder_publish_termostat
    - id: vypinani_termostatu
    - id: displej_st7789_termostat
    - id: ovladani_termostatu
```

C++ kód termostatu

```cpp
#include "esphome.h"
#include "Adafruit_NeoPixel.h"
#include "DHT.h"
#include "Adafruit_GFX.h"
#include "Adafruit_ST7789.h"
#include "Adafruit_SPIFlash.h"
#include "Adafruit_ImageReader.h"
#include "SdFat.h"
#include <SPI.h>

//mapování pinů
#define SW_PIN                  8
#define LED_PIN                 18
#define CLK_PIN                 10
#define DT_PIN                  9
#define HEAT_PIN                6
#define DHT_PIN                 1
#define NTC_SENSOR_PIN          3

//časové konstanty
#define STOP_TIME               3000
#define CLICK_FILTER_TIME       50
#define LED_DELAY               500
#define TERM_UPDATE             1000

//ostatní konstanty
#define HEAT_TEMP_MAX           40
#define HYS                     0.1
#define NTC_BETA                3895
#define MAX_ADC                 16383
#define NTC_BALANCE_RESISTOR    10000
#define REF_TEMP                293
#define REF_RESISTANCE          10000

volatile bool ButtonDispFlag = false;
volatile bool EncoderFlag = false;
volatile bool StopFlag = false;
unsigned long ButtonMillis = 0;
unsigned long ButtonPrewMillis = ButtonMillis;
unsigned long StopMillis = 0;
unsigned long PrevStopMillis = StopMillis;
volatile float TempWanted = 20;
float TempWantedStep = 0.1;
float Temp = 20;
float Hum = 0;
float PrevTemp = Temp;
float PrevHum = Hum;
float HeatTemp = 0;
bool HeatState = false;
bool CLKState = true;
bool DTState = true;
bool StopState = false;
volatile bool CLKPrewState = CLKState;
volatile bool EncoderPublishFlag = false;

//přerušovací rutina tlačítka enkodéru
void IRAM_ATTR Button_isr() {
  ButtonMillis = millis();
  if ((ButtonMillis - ButtonPrewMillis > CLICK_FILTER_TIME) || (ButtonPrewMillis > ButtonMillis)){ //filtrování zákmitů a ošetření přetečení registru millis()
    ButtonPrewMillis = ButtonMillis;
    if(digitalRead(SW_PIN) == false){
        ButtonDispFlag = true;
        StopFlag = true;
        PrevStopMillis = millis();
    }else{
        StopFlag = false;
    }
  }
}

//přerušovací rutina enkodéru
void IRAM_ATTR Encoder_isr() {
    CLKState = digitalRead(CLK_PIN);
    DTState = digitalRead(DT_PIN);
    if (CLKPrewState != CLKState){
        EncoderPublishFlag = true;
        EncoderFlag = true;
        if (CLKState != DTState){
            TempWanted = TempWanted + TempWantedStep;
        }else{
            TempWanted = TempWanted - TempWantedStep;
        }
    }
    CLKPrewState = CLKState;
}

//blikání LED diody na desce
class LED_Flash : public Component{
public:

  unsigned long LEDMillis = 0;
  unsigned long LEDPrevMillis = LEDMillis;
  bool LEDstate = false;

  Adafruit_NeoPixel BoardLED = Adafruit_NeoPixel(1, LED_PIN, NEO_RGB + NEO_KHZ800);

  void setup() override {

    BoardLED.begin();
    BoardLED.setBrightness(30);

  }
```

```
  void loop() override {

    unsigned long LEDMillis = millis();
    if (LEDPrevMillis > LEDMillis){
      LEDPrevMillis = LEDMillis;
    }
    if ((LEDMillis - LEDPrevMillis) >= LED_DELAY){
      LEDstate = !LEDstate;
      if (LEDstate  == true){
          BoardLED.setPixelColor(0, BoardLED.Color(0, 0, 255));
          BoardLED.show();
      }else{
          BoardLED.setPixelColor(0, BoardLED.Color(0, 0, 0));
          BoardLED.show();
      }
      LEDPrevMillis = LEDMillis;
    }
  }
};

//vypnutí termostatu tlačítkem enkodéru a odeslání stavu do Home Assistentu
class StopPublish : public Component {
public:

  void setup() override {

  }
  void loop() override {

    if (StopFlag == true){
        StopMillis = millis();
        if ((StopMillis - PrevStopMillis) >= STOP_TIME){
            StopState = !StopState;
            id(vypinac_termostatu).publish_state(StopState);
            StopFlag = false;
        }
    }
  }
};

//odeslání nové nastavené teploty do Home Assistentu
class EncoderPublish : public Component{
public:

  void setup() override {

  }
  void loop() override {

      if (EncoderPublishFlag == true){
        auto EncoderPublishCall = id(pozadovana_teplota).make_call();
        EncoderPublishCall.set_value(TempWanted);
        EncoderPublishCall.perform();
        EncoderPublishFlag = false;
      }
  }
};

//regulace termostatu
class TermostatControl : public Component{
public:

  unsigned long TermMillis = 0;
  unsigned long TermPrewMillis = TermMillis;
  bool HeatAllow = true;

  DHT DHTSensor {DHT_PIN, DHT22};

  //Přijímání požadavků z Home Assistentu
  TermostatControl(esphome::template_::TemplateNumber *&_TempWanted, esphome::template_::TemplateSwitch *&_StopState)
  {

    _TempWanted->add_on_state_callback([this](float TW)
                                       { TempWanted = TW;});

    _StopState->add_on_state_callback([this](bool SS)
                                       { StopState = SS;});
  }

  //přepočet odporu NTC čidla na teplotu
  float NTCRead (){
      unsigned int VoltageRead = analogRead(NTC_SENSOR_PIN);
      float NTCRes = NTC_BALANCE_RESISTOR*((MAX_ADC/VoltageRead) - 1);
      float NTCTemp = (NTC_BETA*REF_TEMP)/(NTC_BETA + (REF_TEMP*log(NTCRes/REF_RESISTANCE))) - 273;
      return NTCTemp;
  }

  void setup() override {

    pinMode(HEAT_PIN, OUTPUT);
    pinMode(NTC_SENSOR_PIN, INPUT);
    DHTSensor.begin();

  }
```

```
  void loop() override {

    TermMillis = millis();
    //perioda regulační smyčky (+ošetření přetečení registru millis())
    if((TermMillis - TermPrewMillis >= TERM_UPDATE) || (TermMillis < TermPrewMillis)){
        TermPrewMillis = TermMillis;

            //načtení hodnot ze senzorů a jejich korekce
            Temp = DHTSensor.readTemperature() - 3;
            Hum = DHTSensor.readHumidity();
            HeatTemp = NTCRead ();

            //filtrace NAN hodnot DHT senzoru
            if((isnan(Temp)) || (isnan(Hum))){
                Temp = PrevTemp;
                Hum = PrevHum;
            }else{
                PrevTemp = Temp;
                PrevHum = Hum;
            }

            //omezení od podlahového čidla
            if((HeatTemp > HEAT_TEMP_MAX + HYS) && HeatAllow == true){
                HeatAllow = false;
            }
            if((HeatTemp < HEAT_TEMP_MAX - HYS) && HeatAllow == false){
                HeatAllow = true;
            }

            //hysterézní regulace
            if(HeatAllow = true){
                if(Temp < TempWanted - HYS){
                    HeatState = true;
                }
                if(Temp > TempWanted + HYS){
                    HeatState = false;
                }
            }else{
                HeatState = false;
            }
            if(StopState == false){
                digitalWrite(HEAT_PIN, HeatState);
            }else{
                digitalWrite(HEAT_PIN, LOW);
            }

            //odeslání hodnot do Home Assistantu
            id(zhaveni).publish_state(HeatState);

            id(teplota_topneho_telesa).publish_state(HeatTemp);

            id(teplota_mistnosti).publish_state(Temp);

            id(vlhkost_mistnosti).publish_state(Hum);

            if(StopState == true){
                id(stav_termostatu).publish_state("Vypnuto");
            }else{
                id(stav_termostatu).publish_state("Zapnuto");
            }
    }
  }
};

class DisplayST7789 : public Component{
public:

    unsigned long DispMillis = 0;
    unsigned long DispPrevMillis = DispMillis;
    unsigned long DispDelay = 30000;
    float TempPrint = 0;
    float HumPrint = 0;
    float TempWantedPrint = 0;
    bool HeatPrint = true;
    bool StopStatePrint = false;
    float PrevTempPrint = TempPrint;
    float PrevHumPrint = HumPrint;
    float PrevTempWantedPrint = TempWantedPrint;
    bool PrevHeatPrint = HeatPrint;
    bool PrevStopStatePrint = StopStatePrint;
    bool DispEnable = false;


    Adafruit_ST7789 tft = Adafruit_ST7789(34, 33, -1);
    SdFat SD;
    Adafruit_ImageReader reader = Adafruit_ImageReader(SD);
    Adafruit_Image       img;
    ImageReturnCode ircode;
```

```
286
287   DisplayST7789(esphome::template_::TemplateSensor *&_TempPrint, esphome::template_::TemplateSensor *&_HumPrint, esphome::template_::TemplateNumber *&_TempWantedPrint, esphome::template_::
288   {
289     _TempPrint->add_on_state_callback([this](float TP)
290                                       { TempPrint = TP;});
291
292     _HumPrint->add_on_state_callback([this](float HP)
293                                       { HumPrint = HP;});
294
295     _TempWantedPrint->add_on_state_callback([this](float TWP)
296                                       { TempWantedPrint = TWP;});
297
298     _HeatPrint->add_on_state_callback([this](bool HP)
299                                       { HeatPrint = HP;});
300
301     _StopStatePrint->add_on_state_callback([this](bool SSP)
302                                       { StopStatePrint = SSP;});
303   }
304
305   //aktualizace hodnot na displeji + odmazávání přebytečných desetinných čísel
306   void TFTPrint (int TextSize, int SetCursorX, int SetCursorY, float Value, float *PrevValue, int ValueIndex, int TextColor){
307     tft.setTextSize(TextSize);
308     tft.setCursor(SetCursorX, SetCursorY);
309     tft.setTextColor(ST77XX_WHITE);
310     tft.print(*PrevValue);
311     tft.setCursor(tft.getCursorX() - TextSize*6, tft.getCursorY());
312     if (ValueIndex == 0){
313         tft.print("0C");
314         tft.setCursor(tft.getCursorX() - TextSize*6 - 20, tft.getCursorY());
315         //tft.setCursor(tft.getCursorX() - TextSize*9 - 2, tft.getCursorY());
316         tft.setTextSize(3);
317         tft.print("o");
318         tft.setTextSize(TextSize);
319     }else{
320         tft.setCursor(tft.getCursorX() - TextSize*3, tft.getCursorY());
321         tft.print("0%");
322     }
323     tft.setCursor(SetCursorX, SetCursorY);
324     tft.setTextColor(TextColor);
325     tft.print(Value);
326     tft.setCursor(tft.getCursorX() - TextSize*6, tft.getCursorY());
327     tft.setTextColor(ST77XX_WHITE);
328     tft.print("0");
329     tft.setTextColor(TextColor);
330     if (ValueIndex == 0){
331         tft.print("C");
332         tft.setCursor(tft.getCursorX() - TextSize*6 - 20, tft.getCursorY());
333         tft.setTextSize(3);
334         tft.print("o");
335         tft.setTextSize(TextSize);
336     }else{
337         tft.setCursor(tft.getCursorX() - TextSize*3, tft.getCursorY());
338         tft.print("%");
339     }
340     *PrevValue = Value;
341   }
342
343   void setup() override {
344
345       //inicializace a přednastavení displeje a SD karty
346       tft.init(240, 240);
347       tft.setRotation(2);
348
349       if(!SD.begin(20, SD_SCK_MHZ(10))) {
350           tft.fillScreen(ST77XX_BLACK);
351           tft.setCursor(180, 40);
352             tft.setTextSize(4);
353           tft.setTextColor(ST77XX_ORANGE);
354           tft.print("SD");
355           tft.setCursor(10, 120);
356           tft.print("Failed");
357       }else{
358           ircode = reader.drawBMP("/HAlogo.bmp", tft, 0, 0); //uvítací logo
359           reader.printStatus(ircode);
360       }
361       delay(2000);
362       tft.fillScreen(ST77XX_WHITE);
363       tft.setTextColor(ST77XX_BLACK);
364   }
365   void loop() override {
366
367     //probuzení displeje enkodérem
368     if ((ButtonDispFlag == true) || (EncoderFlag == true)){
369         DispEnable = true;
370         tft.enableDisplay(true);
371         DispPrevMillis = millis();
372         ButtonDispFlag = false;
373         EncoderFlag = false;
374     }
375
376     //zhasnutí displeje po neaktivitě
377     if (DispEnable == true){
378         DispMillis = millis();
379         if (DispMillis - DispPrevMillis >= DispDelay){
380             DispEnable = false;
381             tft.enableDisplay(false);
382         }
383     }
384
385     //aktualizace hodnot
386     if (PrevTempPrint != TempPrint){
387         TFTPrint (6, 15, 100, TempPrint, &PrevTempPrint, 0, ST77XX_BLACK);
388     }
```

```
    if (PrevHumPrint != HumPrint){
        TFTPrint (4, 5, 5, HumPrint, &PrevHumPrint, 1, ST77XX_BLUE);
    }

    if (PrevTempWantedPrint != TempWantedPrint){
        TFTPrint (5, 30, 190, TempWantedPrint, &PrevTempWantedPrint, 0, ST77XX_BLACK);
    }

    if ((PrevStopStatePrint != StopStatePrint) || (PrevHeatPrint != HeatPrint)){
        if (StopStatePrint == true){
            tft.setTextColor(ST77XX_WHITE);
            tft.setRotation(3);
            tft.setTextSize(6);
            tft.setCursor(5, 50);
            tft.print("~");
            tft.setCursor(5, 35);
            tft.print("~");
            tft.setCursor(5, 20);
            tft.print("~");
            tft.setRotation(2);
            tft.setCursor(160, 5);
            tft.setTextSize(4);
            tft.setTextColor(ST77XX_RED);
            tft.print("OFF");
        }else{
            tft.setTextSize(4);
            tft.setCursor(160, 5);
            tft.setTextColor(ST77XX_WHITE);
            tft.print("OFF");
            if ((PrevStopStatePrint != StopStatePrint) || (PrevHeatPrint != HeatPrint)){
                if (HeatPrint == true){
                    tft.setTextColor(ST77XX_ORANGE);
                }else{
                    tft.setTextColor(ST77XX_WHITE);
                }
                tft.setRotation(3);
                tft.setTextSize(6);
                tft.setCursor(5, 50);
                tft.print("~");
                tft.setCursor(5, 35);
                tft.print("~");
                tft.setCursor(5, 20);
                tft.print("~");
                tft.setRotation(2);
                PrevHeatPrint = HeatPrint;
            }
        }
        PrevStopStatePrint = StopStatePrint;
    }
  }
};
```

YAML kód ovládání garážových vrat/rolet

```yaml
esphome:
  name: garazova-vrata
  friendly_name: Garážová Vrata
  includes:
    - MyGarageDoorCode.h

esp32:
  board: esp32-s2-saola-1
  framework:
    type: arduino

# Enable logging
logger:

# Enable Home Assistant API
api:
  encryption:
    key: "y7DjpDTgVnJBLdiy8itRQZusqDSIyNYMYD5y9LNPVeA="

ota:
  password: "1cabf82a1fd0f88d67e7466315a6f4ce"

wifi:
  ssid: !secret wifi_ssid
  password: !secret wifi_password

  manual_ip:
    static_ip: 192.168.1.240
    gateway: 192.168.1.1
    subnet: 255.255.255.0

  # Enable fallback hotspot (captive portal) in case wifi connection fails
  ap:
    ssid: "Garazova-Vrata Fallback Hotspot"
    password: "4CWfqFrJnpDn"

captive_portal:

#oznámení o stavu vrat/rolet
text_sensor:
  - platform: template
    name: "Stav"
    id: stav_vrat
    update_interval: never

#ovládání vrat/rolet
cover:
  - platform: template
    name: "Garážová vrata"
    id: garazova_vrata
    open_action:
      - number.set:
          id: index_akce_vrat
          value: 1
    close_action:
      - number.set:
          id: index_akce_vrat
          value: 2
    stop_action:
      - number.set:
          id: index_akce_vrat
          value: 3
    optimistic: true
    assumed_state: true
```

```
#pomocná entita ke cover
number:
  - platform: template
    name: "Index akce vrat"
    id: index_akce_vrat
    optimistic: true
    min_value: 0
    max_value: 3
    step: 1
    initial_value: 0
    internal: True

#pouštění režimu ventilace
switch:
  - platform: template
    name: "Režim Ventilace"
    id: ventilace
    optimistic: true

#instance C++ tříd v podobě ESPHome komponentů
custom_component:
  lambda: |-
    auto ledflash = new LED_Flash();
    auto gdcontrol = new GarageDoorControl(id(index_akce_vrat), id(ventilace));
    return {ledflash, gdcontrol};

  components:
    - id: led_flash_vrata
    - id: ovladani vrat
```

C++ kód ovládání garážových vrat/rolet

```cpp
#include "esphome.h"
#include "Adafruit_NeoPixel.h"

//mapování pinů
#define BOARD_LED_PIN           18
#define LED_SIG_PIN             16
#define UP_BUTTON_PIN           39
#define DOWN_BUTTON_PIN         36
#define UP_RELAY_PIN            1
#define DOWN_RELAY_PIN          2
#define UP_HALL_PIN             44
#define DOWN_HALL_PIN           42
#define UP_RECEIVER_PIN         6
#define DOWN_RECEIVER_PIN       10
#define MODE_PIN1               12
#define MODE_PIN2               14

//časové konstanty
#define LED_DELAY               500
#define DOUBLE_CLICK_TIME       1000
#define CLICK_FILTER_TIME       100
#define VENT_TIME               3000
#define BRAKE_DELAY             1000
#define PULSE_TIME              250

volatile bool UpButtonState = false;
volatile bool DownButtonState = false;
volatile bool UpReceiverState = false;
volatile bool DownReceiverState = false;
volatile bool StopFlag = false;
unsigned long ButtonMillis = 0;
unsigned long ButtonPrevMillis = ButtonMillis;
unsigned long GateDoorMillis = 0;
volatile unsigned long GateDoorPrevMillis = GateDoorMillis;
volatile int VentMode = 0;

//přerušovací rutina tlačítka nahoru
void IRAM_ATTR UpButton_isr() {
  ButtonMillis = millis();
  // ošetření proti přetečení registru millis()
  if (ButtonPrevMillis > ButtonMillis){
      ButtonPrevMillis = ButtonMillis;
      GateDoorPrevMillis = ButtonMillis;
  }
  if (ButtonMillis - ButtonPrevMillis > CLICK_FILTER_TIME ){
    ButtonPrevMillis = ButtonMillis;
    if (digitalRead(UP_BUTTON_PIN) == false){
        UpButtonState = true;
    }
  }
}

//přerušovací rutina tlačítka dolu
void IRAM_ATTR DownButton_isr() {
  ButtonMillis = millis();
  if (ButtonPrevMillis > ButtonMillis){
      ButtonPrevMillis = ButtonMillis;
      GateDoorPrevMillis = ButtonMillis;
  }
  if (ButtonMillis - ButtonPrevMillis > CLICK_FILTER_TIME ){
    ButtonPrevMillis = ButtonMillis;
     if (digitalRead(DOWN_BUTTON_PIN) == false){
        DownButtonState = true;
    }
  }
}

//přerušovací rutina tlačítka A (433 MHZ ovladač)
void IRAM_ATTR UpReceiver_isr() {
  ButtonMillis = millis();
  if (ButtonPrevMillis > ButtonMillis){
      ButtonPrevMillis = ButtonMillis;
      GateDoorPrevMillis = ButtonMillis;
  }
  if (ButtonMillis - ButtonPrevMillis > CLICK_FILTER_TIME ){
    ButtonPrevMillis = ButtonMillis;
     if (digitalRead(UP_RECEIVER_PIN) == true){
        UpReceiverState = true;
    }
  }
}

//přerušovací rutina tlačítka B
void IRAM_ATTR DownReceiver_isr() {
  ButtonMillis = millis();
  if (ButtonPrevMillis > ButtonMillis){
      ButtonPrevMillis = ButtonMillis;
      GateDoorPrevMillis = ButtonMillis;
  }
  if (ButtonMillis - ButtonPrevMillis > CLICK_FILTER_TIME ){
    ButtonPrevMillis = ButtonMillis;
     if (digitalRead(DOWN_RECEIVER_PIN) == false){
        DownReceiverState = true;
    }
  }
}

//blikání LED diody na desce
class LED_Flash : public Component{
public:

  unsigned long LEDMillis = 0;
  unsigned long LEDPrevMillis = LEDMillis;
  unsigned long LEDDelay = LED_DELAY;
  bool LEDstate = false;
```

```
  Adafruit_NeoPixel BoardLED = Adafruit_NeoPixel(1, BOARD_LED_PIN, NEO_RGB + NEO_KHZ800);

  void setup() override {

    BoardLED.begin();
    BoardLED.setBrightness(30);

  }
  void loop() override {

    LEDMillis = millis();

    if (LEDPrevMillis > LEDMillis){
      LEDPrevMillis = LEDMillis;
    }

    if ((LEDMillis - LEDPrevMillis) >= LEDDelay){
      LEDstate = !LEDstate;
      if (LEDstate  == true){
          BoardLED.setPixelColor(0, BoardLED.Color(0, 0, 255));
          BoardLED.show();
      }else{
          BoardLED.setPixelColor(0, BoardLED.Color(0, 0, 0));
          BoardLED.show();
      }
      LEDPrevMillis = LEDMillis;
    }
  }
};

//ovládání vrat/rolet
class GarageDoorControl : public Component{
public:

  bool UpButtonPrevState = UpButtonState;
  bool DownButtonPrevState = DownButtonState;
  bool UpReceiverPrevState = UpReceiverState;
  bool DownReceiverPrevState = DownReceiverState;
  bool UpRelayState = false;
  bool DownRelayState = false;
  bool ButtonCollision = false;
  bool FullOpen = false;
  bool FullClose = false;
  int StepByStepCount = 0;
  int HAAction = 0;
  bool HAActionVent = false;
  bool PrevHAActionVent = HAActionVent;
  int PrevHAAction = HAAction;
  unsigned long VentMillis = 0;
  unsigned long PrevVentMillis = VentMillis;
  bool HallState = false;
  int ControlMode = 1;
  bool HAOpenFlag = false;
  bool HACloseFlag = false;

    //Přijímání požadavků z Home Assistentu
    GarageDoorControl(esphome::template_::TemplateNumber *&_HAAction, esphome::template_::TemplateSwitch *&_HAActionVent)
  {

    _HAAction->add_on_state_callback([this](int HAA)
                                     { HAAction = HAA;});

    _HAActionVent->add_on_state_callback([this](bool HAAV)
                                     { HAActionVent = HAAV;});
  }

  void setup() override {

      pinMode (UP_HALL_PIN, INPUT);
      pinMode (DOWN_HALL_PIN, INPUT);
      pinMode (UP_RELAY_PIN, OUTPUT);
      pinMode (DOWN_RELAY_PIN, OUTPUT);
      pinMode (LED_SIG_PIN, OUTPUT);

      //načtení pinů jumperu
      if (digitalRead(MODE_PIN1) == false){
        ControlMode = 1;
      }else if (digitalRead(MODE_PIN2) == false){
        ControlMode = 2;
      }else{
        ControlMode = 0;
      }

      //odeslání výchozích hodnot do Home Assistantu
      if(ControlMode == 1){
        if(digitalRead(DOWN_HALL_PIN) == false){
            PublishState (2);
            id(garazova_vrata).position = COVER_CLOSED;
            id(garazova_vrata).publish_state();
        }else{
            PublishState (1);
            id(garazova_vrata).position = COVER_OPEN;
            id(garazova_vrata).publish_state();
        }
      }else if(ControlMode == 2){
        PublishState (5);
        id(garazova_vrata).position = 0.5;
        id(garazova_vrata).publish_state();
      }else{
          PublishState (7);
      }
  }

  void loop() override {

```

```
    //Akce vykonané při změně požadavku z Home Assistentu
    if ((HAAction != PrevHAAction) || (HAActionVent != PrevHAActionVent)){
        PrevHAAction = HAAction;
        PrevHAActionVent = HAActionVent;
        if(HAActionVent == true){
            HAAction = 4;
            HAActionVent = false;
        }
        switch(HAAction){
            case 1:
                FullOpen = true;
                if (ControlMode == 1){
                    FullClose = false;
                    id(ventilace).publish_state(false);
                    VentMode = 0;
                    PublishState (3);
                }else if (ControlMode == 2){
                    HAOpenFlag = true;
                    PublishState (3);
                }
                break;
            case 2:
                FullClose = true;
                if (ControlMode == 1){
                    FullOpen = false;
                    id(ventilace).publish_state(false);
                    VentMode = 0;
                    PublishState (4);
                }else if (ControlMode == 2){
                    HACloseFlag = true;
                    PublishState (4);
                }
                break;
            case 3:
                if (ControlMode != 3){
                    StopFlag = true;
                    HAAction = 0;
                }
                break;
            case 4:
                if (ControlMode == 1){
                    FullClose = false;
                    FullOpen = false;
                    VentMode = 1;
                    id(garazova_vrata).position = COVER_CLOSED;
                    id(garazova_vrata).publish_state();
                }else{
                    id(ventilace).publish_state(false);
                }
                break;
            default:
                break;
        }
    }

    //Režim ovládání garážových vrat
    if(ControlMode == 1){
        if(StopFlag == false){
            //kontrola koncových spínačů
            if((digitalRead(UP_HALL_PIN) == false) && (HallState == false)){
                HallState = true;
                MotorSignal();
                StepByStepCount = 0;
                FullOpen = false;
                PublishState (1);
                id(garazova_vrata).position = COVER_OPEN;
                id(garazova_vrata).publish_state();
            }
            if ((digitalRead(DOWN_HALL_PIN) == false) && (HallState == false)){
                HallState = true;
                if(VentMode == 2){
                    VentMode = 3;
                }
                MotorSignal();
                StepByStepCount = 2;
                FullClose = false;
                PublishState (2);
                id(garazova_vrata).position = COVER_CLOSED;
                id(garazova_vrata).publish_state();
            }else if(((digitalRead(UP_HALL_PIN) == true) && (digitalRead(DOWN_HALL_PIN) == true)) && (HallState == true)){
                HallState = false;
            }

            //akce vykonané při stisku některého z tlačítek
            if((UpButtonState == true) || (UpReceiverState == true)){
                //resetování vybraných stavů
                UpButtonState = false;
                UpReceiverState = false;
                FullOpen = false;
                FullClose = false;
                VentMode = 0;
                id(ventilace).publish_state(false);
                GateDoorMillis = millis();
                //detekce dvojitého stisknutí
                if ((GateDoorMillis - GateDoorPrevMillis) <= DOUBLE_CLICK_TIME){
                    VentMode = 1;
                    id(garazova_vrata).position = COVER_CLOSED;
                    id(garazova_vrata).publish_state();
                }else{
                    GateDoorPrevMillis = millis();
                    StepByStepCount ++;
                    if (StepByStepCount == 4){
                        StepByStepCount = 0;
                    }
                    MotorSignal();
```

```
                //odeslání stavu do Home Assistantu
                if(StepByStepCount == 1){
                    PublishState (4);
                    id(garazova_vrata).position = COVER_CLOSED;
                    id(garazova_vrata).publish_state();
                }else if(StepByStepCount == 3){
                    PublishState (3);
                    id(garazova_vrata).position = COVER_OPEN;
                    id(garazova_vrata).publish_state();
                }else{
                    PublishState (1);
                    id(garazova_vrata).position = COVER_OPEN;
                    id(garazova_vrata).publish_state();
                }
            }
        }

        //odpočítání pulzů při požadavku na otevření
        if(FullOpen == true){
            switch (StepByStepCount) {
                case 0:
                    MotorSignal();
                    StepByStepCount++;
                    break;
                case 1:
                    MotorSignal();
                    StepByStepCount++;
                    break;
                case 2:
                    MotorSignal();
                    StepByStepCount++;
                    break;
                default:
                    break;
            }
        }

        if(FullClose == true){
            switch (StepByStepCount) {
                case 0:
                    MotorSignal();
                    StepByStepCount++;
                    break;
                case 2:
                    MotorSignal();
                    StepByStepCount++;
                    break;
                case 3:
                    MotorSignal();
                    StepByStepCount = 0;
                    break;
                default:
                    break;
            }
        }

        //vykonávané akce při požadavku na režim ventilace
        if(VentMode == 1){
            switch (StepByStepCount) {
                case 0:
                    MotorSignal();
                    StepByStepCount++;
                    VentMode = 2;
                    PublishState (4);
                    break;
                case 1:
                    VentMode = 2;
                    PublishState (4);
                    break;
                case 2:
                    MotorSignal();
                    StepByStepCount++;
                    break;
                case 3:
                    MotorSignal();
                    StepByStepCount = 0;
                    break;
                default:
                    break;
            }
        }else if(VentMode == 3){
            MotorSignal();
            StepByStepCount = 3;
            VentMode = 4;
            PublishState (3);
            id(garazova_vrata).position = COVER_OPEN;
            id(garazova_vrata).publish_state();
            PrevVentMillis = millis();
        }else if(VentMode == 4){
            VentMillis = millis();
            if ((VentMillis - PrevVentMillis) >= VENT_TIME){
                VentMode = 0;
                MotorSignal();
                StepByStepCount = 0;
                PublishState (6);
                id(ventilace).publish_state(false);
            }
        }
```

```
                    id(ventilace).publish_state(false);
                }
            }
        //vykonánípožadavku na zastavení pohybu
        }else{
            StopFlag = false;
            FullClose = false;
            FullOpen = false;
            VentMode = 0;
            id(ventilace).publish_state(false);
            PublishState (1);
            if(StepByStepCount == 1){
                StepByStepCount = 2;
                MotorSignal();
            }else if(StepByStepCount == 3){
                StepByStepCount = 0;
                MotorSignal();
            }
        }
    //Režim ovládání rolety
    }else if (ControlMode == 2){
        if(StopFlag == false){
                //akce vykonané při stisknutí některého z tlačítek nahoru
                if ((UpButtonState == true) || (UpReceiverState == true)){
                    if (FullClose == true){
                        FullClose = false;
                        digitalWrite(DOWN_RELAY_PIN, false);
                        delay(BRAKE_DELAY);
                    }
                    FullOpen = !FullOpen;
                    digitalWrite(UP_RELAY_PIN, FullOpen);
                    if(FullOpen == false){
                        PublishState (5);
                        id(garazova_vrata).position = 0.5;
                        id(garazova_vrata).publish_state();
                        delay(BRAKE_DELAY);
                    }else{
                        PublishState (3);
                        id(garazova_vrata).position = COVER_OPEN;
                        id(garazova_vrata).publish_state();
                    }
                    UpButtonState = false;
                    UpReceiverState = false;
                }

                //akce vykonané při stisknutí některého z tlačítek dolu
                if((DownButtonState == true) || (DownReceiverState == true)){
                    if (FullOpen == true){
                        FullOpen = false;
                        digitalWrite(UP_RELAY_PIN, false);
                        delay(BRAKE_DELAY);
                    }
                    FullClose = !FullClose;
                    digitalWrite(DOWN_RELAY_PIN, FullClose);
                    if(FullClose == false){
                        PublishState (5);
                        id(garazova_vrata).position = 0.5;
                        id(garazova_vrata).publish_state();
                        delay(BRAKE_DELAY);
                    }else{
                        PublishState (4);
                        id(garazova_vrata).position = COVER_CLOSED;
                        id(garazova_vrata).publish_state();
                    }
                    DownButtonState = false;
                    DownReceiverState = false;
                }

                //vykonání požadavku nahoru z Home Assistantu
                if ((FullOpen == true) && (HAOpenFlag == true)){
                    HAOpenFlag = false;
                    if(FullClose == true){
                        FullClose = false;
                        digitalWrite(DOWN_RELAY_PIN, LOW);
                        delay(BRAKE_DELAY);
                    }
                    digitalWrite(UP_RELAY_PIN, HIGH);
                    id(garazova_vrata).position = COVER_OPEN;
                    id(garazova_vrata).publish_state();
                }

                //vykonání požadavku dolu z Home Assistantu
                if ((FullClose == true) && (HACloseFlag == true)){
                    HACloseFlag = false;
                    if(FullOpen == true){
                        FullOpen = false;
                        digitalWrite(UP_RELAY_PIN, LOW);
                        delay(BRAKE_DELAY);
                    }
                    digitalWrite(DOWN_RELAY_PIN, HIGH);
                    id(garazova_vrata).position = COVER_CLOSED;
                    id(garazova_vrata).publish_state();
                }
        //vykonání požadavku na zastavení pohybu
        }else{
            StopFlag = false;
            StopBlindMove ();
        }
    //servisní režim
    }else{
        digitalWrite(UP_RELAY_PIN, digitalRead(UP_BUTTON_PIN));
        digitalWrite(DOWN_RELAY_PIN, digitalRead(DOWN_BUTTON_PIN));
    }
}
```

```
516
517   //definice signálu pro garážový pohon
518   void MotorSignal(){
519     digitalWrite(UP_RELAY_PIN, HIGH);
520     delay(PULSE_TIME);
521     digitalWrite(UP_RELAY_PIN, LOW);
522     delay(BRAKE_DELAY);
523   }
524
525   //zastavení rolet
526   void StopBlindMove (){
527     digitalWrite(UP_RELAY_PIN, LOW);
528     digitalWrite(DOWN_RELAY_PIN, LOW);
529     FullClose = false;
530     FullOpen = false;
531     PublishState (5);
532     id(garazova_vrata).position = 0.5;
533     id(garazova_vrata).publish_state();
534     delay(BRAKE_DELAY);
535   }
536
537   // odeslání zprávy o stavu do Home Assistantu
538   void PublishState (int State){
539     switch (State){
540         case 1:
541             id(stav_vrat).publish_state("Otevřeno");
542             digitalWrite(LED_SIG_PIN, HIGH);
543             break;
544         case 2:
545              id(stav_vrat).publish_state("Zavřeno");
546             digitalWrite(LED_SIG_PIN, HIGH);
547             break;
548         case 3:
549              id(stav_vrat).publish_state("Otevírání");
550             digitalWrite(LED_SIG_PIN, LOW);
551             break;
552         case 4:
553              id(stav_vrat).publish_state("Zavírání");
554             digitalWrite(LED_SIG_PIN, LOW);
555             break;
556         case 5:
557              id(stav_vrat).publish_state("Nečinné");
558             digitalWrite(LED_SIG_PIN, HIGH);
559             break;
560         case 6:
561              id(stav_vrat).publish_state("Ventilace");
562             digitalWrite(LED_SIG_PIN, HIGH);
563             break;
564         case 7:
565              id(stav_vrat).publish_state("Servisní režim");
566             digitalWrite(LED_SIG_PIN, HIGH);
567             break;
568         default:
569             break;
570     }
571   }
572 };
```