Středoškolská technika 2017

Setkání a prezentace prací středoškolských studentů na ČVUT

KEEPCUBE – SYSTÉM AUTOMATIZACE DOMÁCNOSTI

Marek Šefl, Dominik Kadera, Richard Šimeček, Ondrej Telka, Dominik Dosoudil, Narek Vardanjan

Střední průmyslová škola elektrotechnická
V Úžlabině 320, Praha 10

„Prohlašuji, že jsem svou práci SOČ vypracoval samostatně a použil jsem literárních pramenů a informací, které cituji a uvádím v seznamu použité literatury a zdrojů informací.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb. o právu autorském a o změně některých zákonů (autorský zákon v platném znění).“

V Praze dne ....

.....

.....

podpis autora (jméno a příjmení)

Anotace

Práce se zabývá levným, otevřeným a cenově dostupným systémem inteligentní domácnosti od návrhu až po realizaci. Uživateli je tak umožněn sběr dat ze senzorů nebo ovládání automatizovaných systémů.

Klíčová slova: Chytrá domácnost, inteligentní budova, automatizace

Annotation

The project, which we have been working on is an Open Source and affordable solution of a smart home, designed from scratch. Allowing user to collect data and remotely control home systems.

Key words: Intelligent house, intelligent building, automatization

Obsah

Úvod

KeepCUBE byla naplánována jako výrobně i prodejně levnější alternativa běžných komerčních chytrých domácností. V bytě či domě není potřeba nic vybourávat kvůli senzorům, všechny části jsou určeny pro použití v interiérech. Od toho se odvíjí i design celého projektu, v němž jsme vsadili na to nejjednodušší, na krychle.

Hlavním cílem naší práce tedy bylo vytvořit centrum chytré domácnosti, které bude pomocí různých senzorů a i na dálku ovladatelných spínačů schopné zajišťovat běh domácnosti, inteligentně šetřit elektrickou energii tam, kde zrovna není potřeba, a v neposlední řadě také dokáže nahradit domácí meteostanici.

Celý projekt uvolníme pod licencí Open Source, takže zdrojové kódy pro řízení KeepCUBE, případně i modely šasi budou po dokončení projektu k dispozici pro případné zájemce o vlastní výrobu nebo „fajnšmekry“, kteří by si chtěli model doupravit nějakým vlastním způsobem.

1 Idea projektu

Hlavní myšlenkou projektu je již od začátku vytvoření systému chytré domácnosti, která bude disponovat licencí Open Source, bude cenově dostupná a zároveň nebude uživatele přímo nutit kupovat proprietární hardware. Ale naopak bude kompatibilní s jinými neproprietárními chytrými prvky, které budou uživateli dostupné za mnohem nižší cenu, než u jiných podobných systémů. Jako příklad můžeme uvést chytré vypínače světel. Běžný vypínač na dálkové ovládání je možné sehnat okolo pětiset korun. Oproti tomu proprietární vypínače určené k jednomu konkrétnímu systému chytré domácnosti jsou téměř trojnásobně dražší.

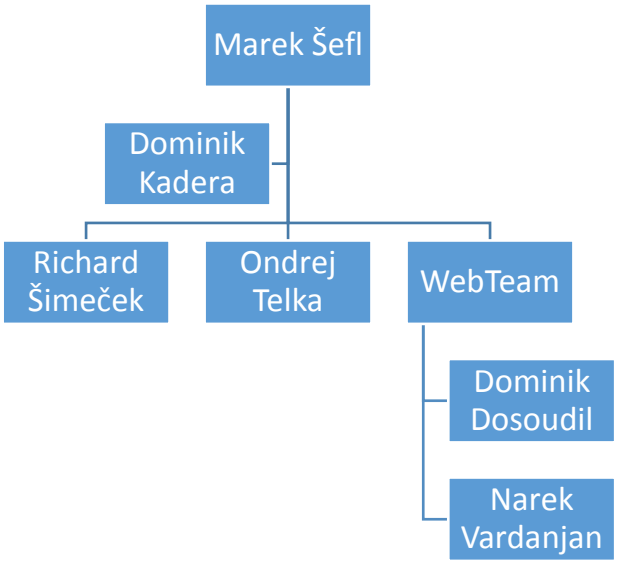
Jednou z předností tohoto systému by měla být schopnost učit se kompatibilitě s novými chytrými prvky, a případně následné sdílení této kompatibility s jinými zařízeními KeepCUBE. To samozřejmě může zahrnovat i uživateli navržený a vyrobený hardware. Nový hardware je možné přidat pomocí webového rozhraní, kde bude možné po zvolení „Přidat typ zařízení“ definovat, které funkce a parametry bude zařízení potřebovat a jakým způsobem s ním má systém KeepCUBE zacházet.

Obrázek 1: Prezentační plakát projektu

2 KeepCUBE tým

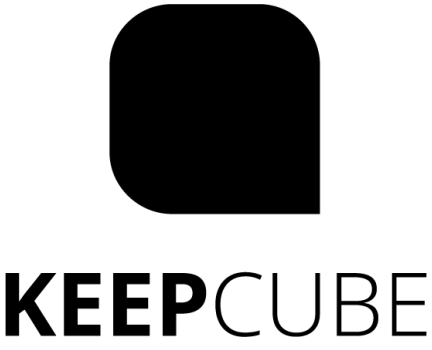
Původně jsme se do projektu pustili pouze dva (Marek Šefl a Dominik Kadera). Postupně, jak jsme koncept inteligentní domácnosti začali vyvíjet, začal se rozrůstat i náš tým.

V současnosti je hlavním koordinátorem Marek Šefl. Richard Šimeček má na starosti kompletní zajištění hardwarové stránky projektu. Ondrej Telka hardware následně programuje. Dominik Dosoudil společně s Narkem Vardanjanem vytvářejí webové rozhraní. Dominik Kadera napojuje jednotlivé části projektu na sebe.



3 Logo projektu

Máme připravené logo projektu s různými variantami (barevné, černobílé), viz např. níže zobrazené.



4 Inteligentní budovy

Inteligentní budova může být složena z mnoha rozličných prvků. Většinu z nich jsme se snažili zahrnout i do našich řešení.

Aktivní systém inteligentní budovy může ovládat například osvětlení, otevírat a zavírat okna, zatahovat rolety, lze do něj například integrovat senzor pro měření teploty a tak proces udržení teploty zautomatizovat, systém může také kontrolovat např. dovření dveří, otevírat a zavírat garážová vrata apod. Podle počasí a teploty může pak ovládat nastavení klimatizace a topení.

Systém se dá použít i jako zabezpečovací systém, který pozná neoprávněnou osobu v bytě (přičemž se k systému dají připojit i bezpečnostní kamery a čidla), případně kontroluje, jestli není spuštěn plyn nebo elektřina tam, kde to není potřeba. Ve finále by takto fungující systém měl být i schopen rozpoznat, zda je spuštěná televize nebo například sporák ještě využíván a v případě potřeby může sám ukončit činnost těchto zařízení.

Kromě těchto činností může celý systém fungovat i jako multimediální centrum, spravovat rodinné fotografie a videa, uchovávat je, případně zálohovat na webové servery. V neposlední řadě také může splňovat komunikační funkci, například v případě nouze nebo potřeby zobrazit informační zprávu na nejbližší obrazovce člověku či organizaci, pro které je určena.

Podle různých způsobů a možností fungování inteligentní budovy je možné tyto systémy rozdělit do několika kategorií, jak bude dále popsáno.

Inteligentní budova tedy při svém fungování může plnit skutečně mnoho úloh. Právě proto jsme se rozhodli, že by pro nás bylo velmi výhodné a zajímavé pokusit se sestavit alespoň v základech fungující systém.

4.1 Dům obsahující automatické systémy

Tento typ budovy máme doma vlastně skoro každý. Dá se o něm mluvit již ve chvíli, kdy je použito například světlo kontrolované fotobuňkou, klimatizace, která hlídá teplotu, nebo automaticky se stahující rolety. Podstatou této první kategorie inteligentního domu

je, že žádné z těchto věcí nejsou propojené do jedné sítě, že by se například rolety před okny zatáhnuly ve chvíli, kdy by klimatizace zjistila, že je v místnosti moc velké teplo. Takto to u prvního stupně nefunguje, každá součást „hlídá“ jen sebe.

4.2 Dům obsahující komunikující inteligentní systémy

U druhého stupně inteligentní budovy dochází k propojení samostatných jednotek do jednoho komunikujícího systému. K tomuto efektu se dá dojít dvěma způsoby. Buď dochází k vytvoření MESH sítě, kdy každá jednotka komunikuje se všemi dalšími, které má v dosahu a posílá přes ně zprávy dál, nebo se všechna data posílají, vyhodnocují a zpracovávají na centrálním prvku inteligentní sítě. Může tak dojít například k výše zmiňovanému příkladu, že mezi sebou navzájem komunikuje klimatizace a topení a tak zůstává v místnosti zachována vždy požadovaná teplota.

4.3 Propojený dům (Connected house)

V propojeném domě dochází k připojení systému komunikujícího systému k internetu. Můžete tak například na dálku ovládat světla, topení, klimatizaci nebo rolety přes váš telefon nebo internetové rozhraní, nebo naopak ze senzorů zjišťovat teploty v jednotlivých místech bytu, zkontrolovat, zda jsou zavřená všechna okna nebo dveře, nebo například zjistit, jestli je spuštěný sporák.

Mezi komerční výrobce těchto produktů patří například společnost D-Link nebo v této oblasti možná známější firma ABB.

4.4 Učící se dům (learning house)

Učící se domácnost patří již do kategorie, která není zatím komerčně příliš vyráběna, zatím je spíše v experimentální fázi. Inteligentní učící se systém je schopen vyhodnocovat chování uživatelů, zapamatovat si ho a následné kroky provádět již namísto uživatelů. Například když systém zjistí, že uživatel ve čtvrtek ráno stahuje topení, je schopen si toto chování zapamatovat a následující měsíc již topení tlumit automaticky.

4.5 Pozorný dům

Tato kategorie inteligentní budovy je jakýmsi pomyslným vrcholem automatizace domácnosti. Dovede vám například dokonce připravit ranní kávu a k ní vybrat správně vyváženou snídani. Pokud by ovšem v lednici chyběly suroviny k přípravě nebo jich bylo nedostatek, systém automaticky přidá chybějící položku na nákupní seznam.

Je otázkou, zda takový stupeň automatizace již není přehnaný. Objevují se totiž i názory, že v takto „přeautomatizovaném“ domě je mnohem jednodušší člověka například zabít. Pokud by se totiž zloděj – hacker – dostal k řízení budovy, byl by schopen v ní uživatele na dálku zamknout a přetížením zásuvky nebo spuštěním prázdné rychlovarné konvice způsobit požár. Ze zamčeného bytu poté není úniku. Vždy by proto měla existovat nějaká možnost záchrany, například centrální odstavení celého systému.

5 Hardware

5.1 KeepCUBE base

Hlavní výpočetní jednotkou KeepCUBE base je Compute modul 3 Lite od společnosti Raspberry Pi. Jedná se o jiný formfaktor populárního stejnojmenného mikropočítače od této společnosti, který je Open Source. Compute Module 3 Lite (dále jen Compute Module) je vybaven čtyřjádrovým procesorem ARM Cortex-A53 BCM2837 o taktovací frekvenci 1,2 GHz a 64bitovou architekturou. Compute Module také obsahuje 1GB RAM. Připojení je realizováno 200pinovým konektorem DDR2 SODIMM. Modul má k dispozici různá komunikační rozhraní: Secondary Memory Interface (SMI), Display Parallel Interface (DPI), SD/SDIO Interface, CSI (MIPI Serial Camera), DSI (MIPI Serial Display), USB, HDMI, Composite (TV Out) a dále také sběrnice I2C, SPI a UART. My jsme využili pouze USB, SPI a sériový port. Žádný vizuální interface nebyl potřeba, jelikož bude pro konfiguraci Compute Module použit zabezpečený komunikační protokol Secure Shell.

5.2 Napájení

CM požaduje několik napájecích napětí. Konkrétně 1,8 V pro interní IO Expander, 5 V pro vnitřní spínaný zdroj procesoru, 2,5 V pro kompozitní videosignál a 3,3 V pro vše ostatní. Proud je dodáván prostřednictvím mikro USB, díky tomu je napětí 5 V použito, kde je potřeba a zároveň je napájen pulzní nastavitelný zdroj NCP1532MUAATXG s dvěma výstupy, které jsou pomocí zpětné vazby konfigurovány na 3,3 V a 1,8 V s maximálním výstupním proudem 1 A na kanál, což je pro naše potřeby plně dostačující. Protože jsme nevyužili zdroj kompozitního video signálu, nemusíme vytvářet 2,5 V, stačí podle datasheetu daný napájecí pin připojit na 3,3 V.

5.3 Konektivita

Pro připojení Compute Module do internetové sítě je použit mikročip ENC28J60 pro ethernetové připojení s rychlostí 10 Mb/s a velice rozšířeným RJ45 konektorem. Tento čip je napájen 3,3 V a pracuje na frekvenci 25 MHz. S Compute Module je propojen přes

SPI komunikační protokol, podporující až 20MHz clock signál. V naší KeepCUBE Base je také implementován jeden USB port typu A.

K Compute Module je také připojený mikrokontroler ATmega328p, známý především z vývojových desek Arduino, s taktem 16 MHz. Mikrokontroler řídí veškerou DSC komunikaci v mesh síti a je k Compute Module připojen přes rozhraní UART a je přes něj zároveň programovatelný.

K mikrokontroleru je přes SPI připojen modul pro 2,4GHz bezdrátovou komunikaci nRF24L01 s přenosem až 2 Mb/s. Modul nRF24l01 je stejně jako ATmega napájen 3,3 V ze spínaného zdroje.

Na mikrokontroleru je připojen také 433MHz vysílací modul pro jednosměrnou komunikaci se zařízeními ovládanými na této frekvenci. Vstupní napájení modulu je 3–12 V. V našem případě bude napájen přímo z napájecího mikro USB, tedy 5 V.

Dále je zde pomocí NRZ komunikace připojeno také šestnáct adresovatelných LED diod WS2812b. Tento typ komunikace má tu výhodu, že data lze přenášet pouze jedním vodičem. V KeepCUBE Base je také teplotní a vlhkostní senzor SI7006-A20-IM komunikující přes sběrnici I2C. Veškeré tyto součástky jsou napájeny 3,3 V, pouze WS2812b diody a modul pro 433 MHz jsou připojeny na 5 V. Do budoucna je plánované i bezdrátové připojení přes WiFi.

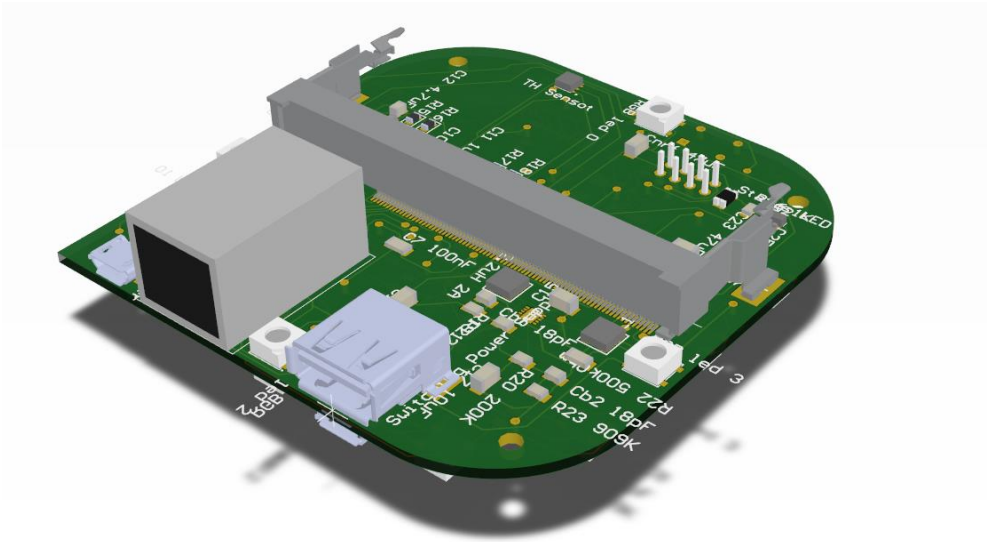
5.4 KeepCUBE příslušenství

Pod příslušenství KeepCUBE momentálně patří malá deska se stejným kontrolorem ATmega328p, obdobně jako u KeepCUBE Base s nRF24l01, transricieverem pro 2.4Ghz bezdrátovou komunikaci se základnou. ATmega spíná pomocí PWM na této desce tři MOSFETy typu N, které řídí LED pásek se společnou anodou. PWM nám umožňuje smíchávat různé složky RGB a tím nastavovat libovolnou barvu.

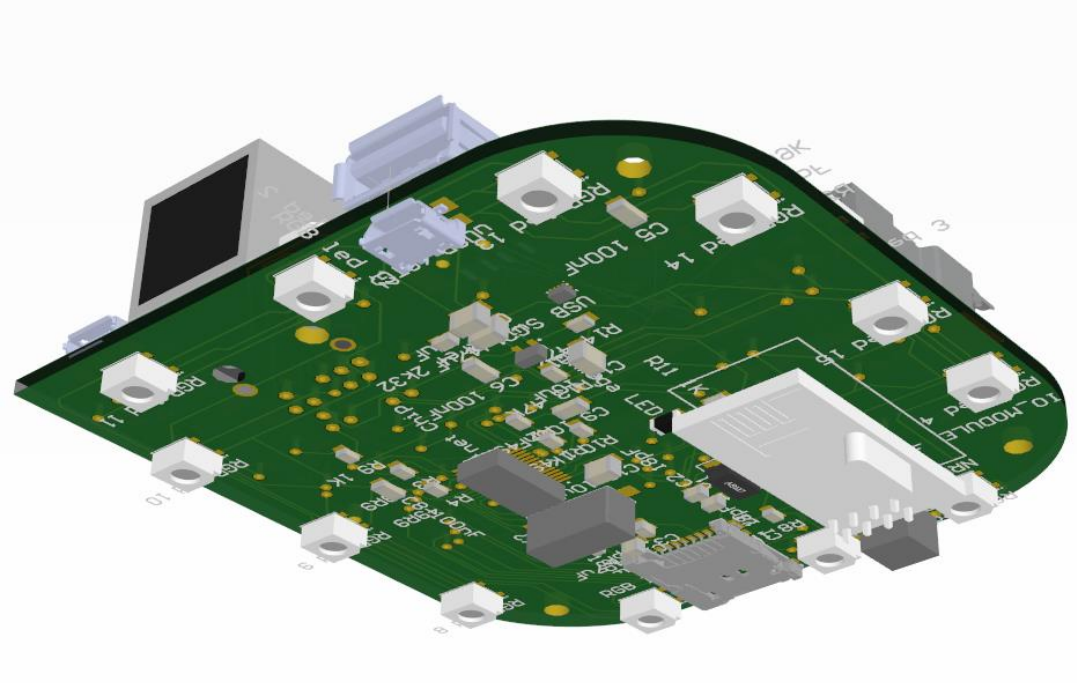
Dále ATmega reaguje na dotykové tlačítko pro změnu intenzity světla LED pásku. Na desce je 3,3V lineární napěťový regulátor pro nRF24l01. Kontroler ATmega je napájen přímo ze zdroje, jelikož deska mikro USB má napájecí konektor s 5 V. Není zde potřeba

měnit 5V řídící signál z kontroleru, protože modul nRF24l01 je s těmito řídícími signály kompatibilní.

Těchto typů desek bude nakonec několik s různými funkcemi a senzory. Od teplotních a vlhkostních, přes senzory měřící koncentraci různých plynů či znečištění až po senzory pohybové.



Obrázek 2: 3D model návrhu DPS



Obrázek 3: 3D model návrhu DPS

6 Descriptive String Communication (DSC)

DSC je námi vymyšlená jednoduchá a „lightweight“ forma zápisu i formátování příkazů a obecně dat v systému KeepCUBE. Používá se při posílání dat sériovým rozhraním mezi Raspberry Pi Compute Modulem a mikrokontrolerem, v mesh síti, napříč Redis channely a všude jinde, kde je potřeba manipulovat s daty v rámci více platforem. DSC vychází ze struktury JSON.

Struktura DSC příkazů je pevně daná. Každý příkaz začíná znakem “#” a končí znakem “;”. Bezprostředně za uvozovacím znakem se nachází identifikátor – tři velká písmena (od A do Z), která symbolicky naznačují účel a použití příkazu.

Za identifikátorem následují parametry, obsahující přenášené hodnoty. Příkaz většinou obsahuje několik parametrů, ve speciálních případech však nemusí obsahovat žádný. Každý parametr musí obsahovat nějakou hodnotu, tzn. nesmí zůstat prázdný. Názvy parametrů jsou vždy svázány s identifikátorem příkazu, tzn. pro příkaz SLP může parametr D znamenat něco jiného, než pro příkaz AAA.

Datovým typem parametru může být buď celé číslo, desetinné číslo, nebo textový řetězec. Textové řetězce mají na začátku a na konci speciální uvozovací znak “&”. Tento znak byl zvolen, protože se neplánuje jeho využití uvnitř řetězců. Pokud by k tomu došlo, musí se tento znak zapsat tzv. escape sekvencí.

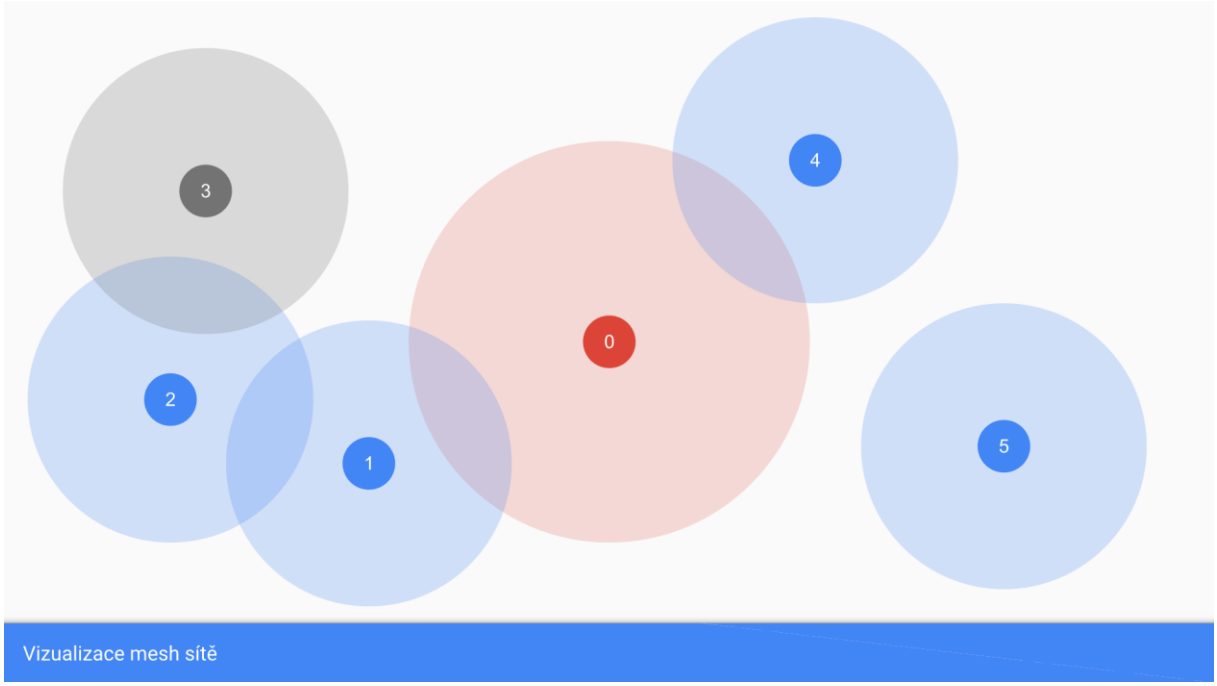
Příkazy mohou být zapouzdřovány do jiných příkazů jako parametr s hodnotou typu string. Zapouzdřování se využívá v KeepCUBE Base k určení, co se má s příkazem stát. Mezi možné způsoby zpracování příkazu patří odeslání modulem nRF24L01+, odeslání pomocí 433 MHz systému. S přibývajícími technologiemi zpracování bude počet možností zapouzdření přibývat.

Abychom mohli využít formát DSC napříč všemi zařízeními, museli jsme naprogramovat parsery příkazů pro všechny platformy, tzn. PHP, Python, C++, a Java. Tyto parsery umí sestavit celý DSC příkaz, manipulovat s jeho parametry, nebo je přidávat a mazat. Konkrétně parser pro využití v mikrokontrolerech ATmega, sepsaný v jazyce C++, má pouhých 11 KB, přičemž zvládne plně editovat daný příkaz.

7 Interní bezdrátová síť mesh

Všechny prvky v KeepCUBE síti komunikují na principu bezdrátové centralizované mesh sítě, kde jednotlivé uzly sítě jsou tvořeny moduly nRF24L01+ spojenými s řídícími mikrokontrolery ATmega328p, nebo případně ATtiny1634, které zpracovávají veškeré požadavky.

Každý datový celek odeslaný do sítě a přijatý příjemcem je kontrolován pomocí položky ACK (Acknowledgement, potvrzení příjmu). V případě poškození nebo ztráty dat se data odešlou znovu.



Obrázek 4: Vizualizace mesh sítě

Obrázek 4 obsahuje vizualizaci reálného příkladu mesh sítě. Červená (ID 0) je Base, modré jsou aktivní uzly, šedivé jsou pasivní uzly. Kdyby uzel s *ID = 2* chtěl poslat zprávu Base
(*ID = 0*), musel by zprávu poslat nejdřív uzlu s *ID = 1*, a až teprve potom by uzel s *ID = 1* mohl zprávu odeslat Base (*ID = 0*). Uzel s *ID = 5* má problém, jelikož se jeho dosah signálu nepřekrývá s žádným dalším, tudíž zprávu nemá komu poslat. Uzel s *ID = 4* nebo uzel
s *ID = 1* zprávu posílají rovnou Base (*ID = 0*). V opačném případě, kdyby měla Base (*ID =*

0) poslat zprávu uzlu s *ID = 3*, musela by ji nejdřív poslat uzlu s *ID = 1*, potom uzlu s *ID = 2*, a nakonec, až by uzel s *ID = 2* poslal zprávu, i uzlu s *ID = 3*.

7.1 Dělení uzlů podle typu

V síti existují tři druhy uzlů:

- Centrální prvek
- Aktivní uzel
- Pasivní uzel

7.1.1 Centrální prvek

Centrální prvek řídí všechen provoz v síti a je jejím hlavním prvkem. Je v síti nepostradatelný, jelikož právě on nastavuje všechny parametry LED páskám a ostatním bezdrátovým periferiím, právě jemu se posílají veškerá data ze senzorů a detektorů.

7.1.2 Aktivní uzel

Vyznačuje se neustálou připojeností do sítě a schopností komunikovat a přenášet zprávy ostatním uzlům sítě.

7.1.3 Pasivní uzel

Je navržen pro nízkou spotřebu elektrické energie, protože jako zařízení nemá přímé napájení z elektrické sítě. Tento uzel funguje v mesh síti jen částečně. Jako plnohodnotný prvek mesh sítě se jeví jen v případě, když není v režimu nízké spotřeby elektrické energie.

Pokud má toto zažízení aktivováno režim nízké spotřeby elektrické energie, je modul nRF24L01+ vypnutý, takže je zařízení odjinud ze sítě nedosažitelné. Ve většině případech používá toto zařízení jako zdroj elektrické energie monočlánky nebo akumulátory. Jedná se zejména o senzory, které se zapínají jednou za určený interval, provedou měření, odešlou data a zase se přepnou do režimu nízké spotřeby energie.

Některé uzly mohou přecházet z aktivního režimu na pasivní a naopak. To má za následek nižší spotřebu energie a tím pádem delší výdrž na baterie.

7.2 Objevování nových zařízení v síti

Když si zákazník dokoupí nebo vyrobí další nový senzor nebo jinou periferii, bude počítat s tím, že mu bude fungovat s již zavedenou sítí a že nebude muset nic dokupovat, nebo měnit stávající konfiguraci. Nové zařízení bude mít nastaveno speciální výrobní ID, podle kterého Base pozná, že se jedná o nové zařízení. Proces přidávání zařízení je velmi jednoduchý a zvládnutelný v několika krocích:

1. Na webu (nebo v aplikaci) uživatel klikne na možnost přidat nové zařízení.
2. Spustí se vyhledávání nových zařízení, přičemž Base začne odesílat požadavky na zmíněnou speciální výrobní adresu a bude sledovat, jestli se jí vrátí validní odpověď. Nové zařízení bude samozřejmě od výroby dostupné na speciální adrese.
3. Po úspěšném přijetí odešle Base novému zařízení příkaz k nastavení nového ID, které mu Base přidělí podle své databáze už zavedených prvků.
4. Nové zařízení si uloží novou adresu do své nevolatilní paměti (EEPROM), ze které ji vždy při dalším startu přečte a nastaví.
5. Nové zařízení se restartuje a při startu si automaticky nastaví přidělené ID.
6. Pokud dojde v průběhu k jakýmkoli potížím, nastaví si zařízení jako poslední možnost speciální servisní (nouzové) ID.
7. Po odpojení zařízení se speciálním výrobním ID bude Base čekat na potvrzovací zprávu od toho samého zařízení (jen s novým přiděleným ID), že je vše v pořádku. V tomto případě vše končí a nové zařízení je úspěšně přidáno.
8. Pokud do stanoveného časového intervalu potvrzující zpráva nedorazí, aktivuje se na Base nouzový stav. Base nejdřív znovu prověří výrobní ID, potom přidělené ID a nakonec servisní (nouzové) ID. Pokud dostane odpověď, opakuje výše popsanou sekvenci pro přidávání zařízení pro stanovený počet opakování, dokud se zařízení nepřipojí.
9. Pokud odpověď nedostane, jedná se o fatální chybu systému, která mohla být způsobena poruchou hardwaru, špatným signálem, nebo jinými příčinami. V tomto extrémním případě je uživatel upozorněn, že bylo přidáno nové zařízení, ale není možné se s ním spojit. Uživatel dostane doporučení, že by měl zařízení restartovat, nebo jej umístit na místo s lepším pokrytím signálem. To je ale tak extrémní možnost, že nastává opravdu jen výjimečně.

8 Software

Základem všech procesů, které probíhají na řídicí desce KC Base, je neustále běžící skript sepsaný v jazyce Python. Tento skript se stará o předávání příkazů DSC mezi jednotlivými vrstvami, tedy webem a nižší vrstvou, kterou tvoří mikrokontroler ATmega a jeho periferie. Python skript se připojí při startu k aplikačnímu serveru služby Redis a začne poslouchat. V případě, že zachytí nějaký příkaz na jeho kanále, spustí se jeho zpracování. Podle hlavičky příkazu skript ví, kam má příkaz poslat, odstraní jednu úroveň zapouzdření a příkaz pošle zadanou cestou.

Zdrojové kódy jsou v příloze.

9 Webové rozhraní

Jelikož samotná architektura komunikace pro inteligentní domácnost potřebuje i systém pro její správu, vytváříme systém, který bude řešen jako webová aplikace. Z toho plyne, že jediné, co bude potřeba, je prohlížeč. Není třeba instalovat žádnou jinou aplikaci.

Protože inteligentní domácnost KeepCUBE funguje decentralizovaně, respektive neexistuje žádný jediný kontrolní server pro všechny KeepCUBE systémy, je webová aplikace hostována v lokální síti KeepCUBE Base.

Pro přehlednost, modifikovatelnost a hlavně znovupoužitelnost jsme rozdělili web na dvě části – API a grafické rozhraní.

9.1 Standard REST API

Slouží ke komunikaci s KeepCube v lokálním síti. Využívá k tomu protokol HTTP a je plně bezstavový, tudíž se o přihlášeném uživateli na serveru neukládají informace a vše, co je ke komunikaci potřeba, se posílá při každém dotazu. Standard REST API vznikl hlavně pro získávání dat pro webový frontend, ale v budoucnu by API mohla sloužit i pro jiné frontendy. Například pro nativní mobilní aplikace či desktop klienta.

9.1.1 Web Framework

Celý backend je napsaný v jazyce PHP, přesněji ve frameworku Laravel. Jde o soubor nástrojů zaměřených na rychlý vývoj. Výhodou je také velká komunita a spoustu již hotových Open Source balíčků k použití. Samotná KeepCUBE bude také licencovaná Open Source a všechno bude veřejně dostupné na GITHUBu. Proto není licenční problém tyto balíčky využívat.

9.1.2 Response z endpointu

Odpověď je vždy ve tvaru validního JSON. V hlavní větvi JSON jsou dva atributy, *ok* a *data*. V atributu *ok* je uložena pravdivostní hodnota, která určuje, jestli akce proběhla v pořádku. K tomuto účelu také slouží hlavička HTTP pojmenovaná *status*. V případě úspěšného požadavku vrací hodnotu začínající číslicí 2.

Pokud je chyba na straně uživatele (uživatel se například ptá na data, která mu nejsou zpřístupněna) začíná chyba číslicí 4. Chyby začínající číslicí 5 znázorňují problémy na serverové straně. V data atributu pak najdeme vyžádané informace společně s metainformacemi. Metainformace se týkají členění obsahu či informací ohledně stránkování. Můžou obsahovat i URL odkazy na jiné zdroje, které bychom chtěli využít.

9.1.3 Redis

Aplikace bude také komunikovat mimo HTTP Redis pub/sub. Python skript se na přihlásí na channel a začne ho odebírat. Cokoli co se odešle na channel se dostane také k Python skriptu, který podle channelu zavolá patřičnou metodu. Channel nemusí být odposloucháván jenom Python skriptem, a tak si může koncový pořizovatel napsat vlastní kód, který z backendu přijímá zprávy. Není tudíž omezován naší implementací.

9.1.4 Autentizace

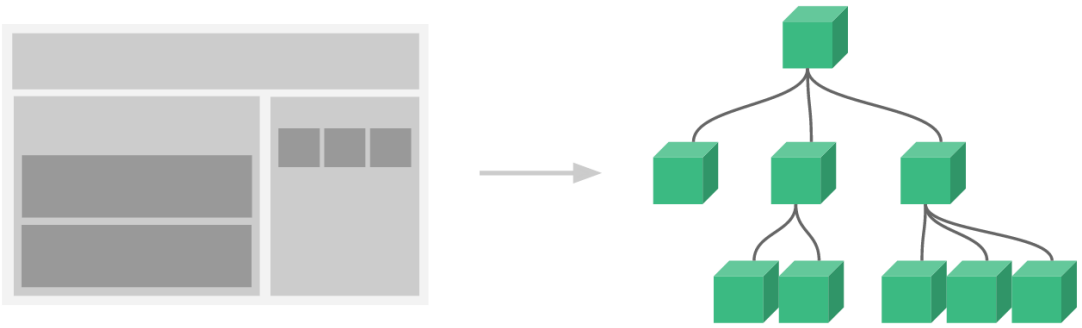
K ověřování totožnosti využíváme protokol JWT, který je přímo určený pro práci s REST EndPoints. Princip spočívá v neustálém generování tokenů, které se při každé poslané žádosti ověří. Server vyhodnotí, jestli máme validní token, a poté nám v odpovědi zašle token nový. Token je validní jen pro jeden dotaz. Nemůže se tedy stát, že by někdo získal náš neplatný token a mohl přes něj bezproblémově pracovat.

9.2 Grafické rozhraní

Na toto API napojíme grafické rozhraní, které si sice udržuje svůj stav, nicméně nekomunikuje s ničím jiným, než právě s API. Výhodou tohoto provedení je možnost zcela předělat a optimalizovat API (například použitím jiného jazyka), ale při zachování API EndPoints je grafické rozhraní naprosto zachováno a jeho funkčnost zůstává stejná.

9.2.1 Vue

Grafické rozhraní je naprogramováno pomocí jazyka JavaScript, konkrétně frameworku Vue. Princip tohoto frameworku spočívá ve vytváření tzv. „webových komponent“, které do sebe můžeme vnořovat, ale snažíme se o jejich nezávislost. Díky této vlastnosti jsme „nuceni“ stavět přehlednou a lehce upravitelnou aplikaci.



Obrázek 5: Vnořování (inheritance) webových komponent

9.3 Vuex

Komponenty nicméně občas potřebují data uchovávat, předávat, nebo ukládat do databáze. Abychom neměli data uchovaná v různých proměnných (nebo hůře v globálních proměnných) a nemuseli řešit jejich předávání mezi komponentami, využíváme Vuex. Jak už vyplývá z názvu, tato „služba“ byla připravena přímo pro framework Vue. Vuex je „stavový manažer“, což znamená, že neslouží k ničemu jinému, než k uchování stavu aplikace.

Díky tomu, že data jsou uchovávána pomocí manažeru Vuex, máme k němu přístup odkudkoliv, což nám zaručuje přehlednost. Vuex navíc umožňuje komponentám operovat s daty jen očekávanými způsoby. To je zaručeno tak, že si nastavíme několik metod pro přístup/modifikaci (tzv. gettery a mutace). Další velkou výhodou je možnost „poslouchat“ změny stavu a při jakékoliv modifikaci ihned odesílat nové hodnoty do databáze pomocí API.

Závěr

Tento projekt, na kterém pracujeme již více než rok, nám přinesl mnoho starostí, problémů, řešení chyb, ale také spoustu zkušeností při hledání řešení. Podle našich představ by měl být projekt KeepCUBE po dokončení volně dostupný jako Open Source pod licencí GPL, tedy pro každého zdarma stažitelný a distribuovatelný.

Zároveň bychom však chtěli nabídnout možnost nákupu již hotových kostek přímo od nás. Víme totiž, že například výroba vlastního Linux boardu není vůbec jednoduchá záležitost a ne každý zájemce o KeepCUBE by byl schopen si desku doma sestavit.

Co se týče otevřenosti, chceme být pro uživatele co nejotevřenější, tedy již od základu počítáme s možnostmi přidávání vlastních zařízení a vlastních částí DSC příkazů, případně celých kusů kódu, a na toto připravujeme i interní programy a rozhraní.

Ve výsledku by se tedy KeepCUBE měla stát levnou a každému dostupnou otevřenou platformou chytré domácnosti, která bude podporovat i ostatní řešení a bude s nimi spolupracovat.

Seznam literatury a ostatních zdrojů

ATmega 328P Datasheet. *Www.mikcrochip.com* [online]. [cit. 2017-03-29]. Dostupné z: ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

Standalone Arduino. *Www.arduino.cc* [online]. [cit. 2017-03-29]. Dostupné z: https://www.arduino.cc/en/Tutorial/ArduinoToBreadboard

Sodimm soket DDR2 Datasheet. *Http://cz.farnell.com/* [online]. [cit. 2017-03-29]. Dostupné z: http://www.farnell.com/cad/1469832.pdf?_ga=1.231052703.1010699264.1487712143

Schéma zapojení čipu ENC28J60. *Www.easyeda.com* [online]. [cit. 2017-03-29]. Dostupné z: https://easyeda.com/editor#id=50e6922b7b884f38a1b854ae8a1156cb|199189eb5bd44909ad181ae35aac8eca

ENC28J60 Datasheet. *Www.microchip.com* [online]. [cit. 2017-03-29]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/39662e.pdf

Tepelná pojistka série MF-MSMF. *Http://cz.farnell.com/* [online]. [cit. 2017-03-29]. Dostupné z: http://www.farnell.com/datasheets/2248897.pdf?_ga=1.190143531.1010699264.1487712143

Schéma desky Arduino Nano. *Www.arduino.cc* [online]. [cit. 2017-03-29]. Dostupné z: https://www.arduino.cc/en/uploads/Main/ArduinoNano30Schematic.pdf

Datasheet čipu LAN9512. *Www.microchip.com* [online]. [cit. 2017-03-29]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/9512.pdf

Datasheet řídícího čipu spínaného zdroje NCP1532. *Http://www.onsemi.com* [online]. [cit. 2017-03-29]. Dostupné z: https://www.onsemi.com/pub/Collateral/NCP1532-D.PDF

Datasheet adresovatelné LED WS2812B. *Https://www.adafruit.com/* [online]. [cit. 2017-03-29]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf

Datasheet senzoru teploty a vlhkosti Si7006-A20. *Https://www.silabs.com/* [online]. [cit. 2017-03-29]. Dostupné z: https://www.silabs.com/documents/public/data-sheets/Si7006-A20.pdf

Schéma mikropočítače Raspberry Pi 3 Model B. *Https://www.raspberrypi.org/* [online]. [cit. 2017-03-29]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/raspberrypi/schematics/RPI-3B-V1_2-SCHEMATIC-REDUCED.pdf

Datasheet Compute Modulu 3 Lite. *Https://www.raspberrypi.org/* [online]. [cit. 2017-03-29]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/computemodule/RPI-CM-DATASHEET-V1_0.pdf

Schéma základní desky pro Compute Module od společnosti Raspberry Pi. *Https://www.raspberrypi.org/* [online]. [cit. 2017-03-29]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/computemodule/RPI-CMIO-V3_0-SCHEMATIC.pdf

Schéma Compute Module 3 od společnosti Raspberry Pi. *Https://www.raspberrypi.org/* [online]. [cit. 2017-03-29]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/computemodule/RPI-CM3-V1_0-SCHEMATIC.pdf

Návod na zapojení periférií. *Https://www.raspberrypi.org/* [online]. [cit. 2017-03-29]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/computemodule/cm-peri-sw-guide.md