ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Inteligentní termostat řízený mikropočítačem Arduino

Smart thermostat controlled by microcomputer Arduino

Diplomová práce

Studijní program: Inteligentní budovy

Vedoucí práce: prof. Ing. Miroslav Husák, CSc.

Bc. Peter Javnický

Praha 2016

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická
Katedra měření

Akademický rok 2015-16

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	**Bc. Peter Javnický**
Studijní program:	**Inteligentní budovy**
Název tématu česky:	Inteligentní termostat řízený mikropočítačem Arduino.
Název tématu anglicky:	Smart thermostat controlled by microcomputer Arduino.

Pokyny pro vypracování:

1. Proveďte rozbor stávajícího stavu řešení inteligentních termostatů řízených mikropočítači s využitím v inteligentních bytech a budovách.
2. Navrhněte termostatický systém řízený mikropočítačem ArduinoMega2560 s aplikací pro inteligentní byt s možností řízení teploty ve více místnostech. Realizujte laboratorní vzorek navrženého systému.
3. Zjistěte dosažené parametry realizovaného systému, porovnejte s parametry komerčně dosažitelných podobných systémů.
4. Proveďte jednoduchý ekonomický rozbor s úvahou pro výrobu navrženého systému, proveďte úvahu o vhodnosti Vašeho návrhu z hlediska komerčního využití.

Seznam odborné literatury:

1. Neumann, P., Uhlíř, J.: Elektronické obvody a funkční bloky (I, II), ČVUT 2001
2. Web-Enable your Arduino with an Arduino ENC28J60 Ethernet shield …. [Online] [Cited: 11 20, 2015.] http://www.tweaking4all.com/hardware/arduino/arduino-enc28j60-ethernet/.
3. KEYES KY-040 Arduino rotary encoder user manual. [Online] [Cited: 11 20, 2015.] http://henrysbench.capnfatz.com/henrys-bench/keyes-ky-040-arduino-rotary-encoderuser-manual

Vedoucí diplomové práce:	prof. Ing. Miroslav Husák, CSc. (K13134)
Datum zadání diplomové práce:	11. ledna 2016
Platnost zadání do[1]:	30. září 2017

L.S.

Doc. Ing. Jan Holub, Ph.D.
vedoucí katedry

Prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 18. 1. 2016

[1] Platnost zadání je omezena na dobu tří následujících semestrů.

Čestné prehlásenie

Čestne prehlasujem, že na diplomovej práci som pracoval samostatne na základe vlastných teoretických a praktických poznatkov, konzultácií a štúdia odbornej literatúry, ktorej úplný prehľad je uvedený v zozname použitej literatúry. Nemám vážny dôvod proti použitiu tejto práce v zmysle § 60 Zákona č.121/2000 Sb., o autorskom práve, o právach súvisiacich s autorským právom a o zmene niektorých zákonov.

V Prahe dňa 25. mája 2016 .....

Bc. Peter Javnický

Abstrakt

Diplomová práca sa zaoberá návrhom, vývojom a realizáciou inteligentného systému zariadení pre reguláciu teploty v byte (respektíve dome). Cieľom je vytvoriť systém, ktorý bude ovládať teplotu v jednotlivých miestnostiach podľa nastavenia užívateľom. Inteligencia systému spočíva v postupnom sa učení režimu užívateľa a zapamätaní si jednotlivých nastavených teplôt v priebehu dňa a týždňa. Ďalej sa práca zaoberá vývojom webovej aplikácie, ktorá slúži pre zobrazovanie jednotlivých teplôt a nastavovanie požadovaných teplôt v jednotlivých miestnostiach. Takisto má aplikácia zabezpečovať prehľad teplôt, ktoré sa termostat naučil, a tak aj umožniť užívateľovi jednotlivé teploty v priebehu dňa a týždňa nastaviť.

Abstract

The diploma thesis deals with a proposal, design and realization of an intelligent system of devices which controls temperature in an apartment (or in a house). The goal is to create a system that will control a temperature in individual rooms depending on a temperature set by the user. The intelligent part of the system is that it is able to learn the schedule of the user and it remembers all temperature that has been set during a day and during a week. Next thing that the thesis deals with, is a web application used to control and monitor whole system. It also provides user with all the temperatures that had been set or learned by the system during a day and during a week.

Kľúčové slová

termostat, inteligentný termostat, inteligetný dom, riadenie, teplota, mikropočítač, mikrokontrolér, Arduino, MQTT, Raspberry Pi

Kľúčové slová

thermostat, intelligent thermostat, intelligent home, control, temperature, microcomputer, microcontroller, Arduino, MQTT, Raspberry Pi

Obsah

Zoznam obrázkov

Zoznam skratiek

TRV - Termoregulačný Ventil

PI - Proporcionálne Ontegračný (regulátor)

GSM - Globálny Systém pre Mobilnú komunikáciu

AC - Alternating current (striedaný prúd)

DC - Direct Current (jednosmerný prúd)

NO - Normaly Open (bez napätia otvorený)

NC - Normaly Closed (bez napätia zatvorený)

IFTTT - If This Then That

IDE - Integrated Development Environment (Integrované vývojové prostredie)

RX - Receive

TX - Transmit

TTL - Transistor–transistor logic

TTL - Integrated development environment (Integrované vývojové prostredie)

SS - Slave Select

MOSI - Master Output, Slave Input

MISO - Master Input, Slave Output

SCK - Serial Clock

SPI - Serial Peripheral Interface

GND - Ground (zem)

PWM - Pulse Width Modulation

HDMI - High-Definition Multimedia Interface

GPIO - General-Purpose Input/Output

MQTT - Message Queue Telemetry Transport

DHCP - Dynamic Host Configuration Protocol

I2C - Inter-Integrated Circuit

TCP/IP - Transmission Control Protocol and the Internet Protocol

OSI - Open Systems Interconnection model

EEPROM - Electrically Erasable Read-Only Memory

UART - Universal Asynchronous Receiver/Transmitter

SRAM - Static Random-Access Memory

PoE - Power over Ethernet

1 Úvod

Inteligentné termostaty zmenili náš pohľad na kúrenie a jeho reguláciu od základov, keď v roku 2011 dizajnér Tony Fadell predstavil verejnosti termostat *Nest Learning Thermostat*, a poukázal tým na dovtedy prehliadanú súčasť skoro každého domu či bytu. Prvý inteligentný termostat rozbehol vývoj týchto zariadení.

Líšia sa v mnohých veciach od svojich predchodcov, avšak výsledná funkcia je stále rovnaká - dosiahnuť v miestnosti požadovanú teplotu. Inteligentné termostaty sú nazývané inteligentnými práve preto, že sledujú režim užívateľa, a podľa toho zapínajú respektíve vypínajú kúrenie. Niektoré sa dokážu tento režim postupne naučiť. Niektoré sledujú pohyb užívateľov, a podľa toho vedia, kto je doma, v ktorej izbe má byť aká teplota a podobne.

Väčšinu inteligentných termostatov je možné ovládať pomocou mobilnej aplikácie, alebo webovej stránky priamo prostredníctvom internetového prehliadača - čo je pre užívateľa pohodlnejšie ako konfigurovať starší termostat pomocou stláčaní tlačidiel alebo ich kombinácií.

Mojím cieľom je navrhnúť a vyrobiť podobné zariadenie, ktoré sa bude postupne učiť režim užívateľa, respektíve zapamätá si každú nastavenú teplotu, a podľa toho bude riadiť kúrenie. Ovládanie bude pomocou manuálneho otočného prvku priamo na zariadení termostatu a takisto pomocou webovej aplikácie cez internetový prehliadač.

2 Čo nie je možné merať, nie je možné šetriť

Šetrenie energie sa vzťahuje k snahe znížiť spotrebu energie. To je možné docieliť zvýšením efektívneho využívania energie. Je teda dôležité znížiť množstvo používanej energie na dosiahnutie podobného výsledku. Nižšia spotreba má mnoho výhod. Jednak my ako užívatelia ušetríme náklady a zároveň pomôžeme životnému prostrediu. Na výrobu energie sa používajú prírodné zdroje, ktoré boli kedysi samozrejmosťou, ale dnes sú už vzácne. Napríklad uhlie, ropa alebo zemný plyn. Znížená spotreba týchto zdrojov nám ich pomôže zachovať a využívať ich aj v budúcnosti.

Úsporné bývanie znamená premýšľať o tom, ako domácnosť funguje. Uvedomovať si toky energií a zdrojov, ktoré v domácnosti fungujú, a že tieto zdroje nie sú zadarmo. Práve naopak. Ceny energií a vody výrazne stúpajú každý rok. Je dôležité, a dnes už pomaly aj štandard, merať v budove všetko čo je možné, pretože čo nie je možné merať, nie je možné šetriť.

V dnešnej modernej a technologicky vyspelej dobe je možné merať a regulovať takmer všetko. A vďaka rýchlemu rozvoju meracej a regulačnej techniky je možné postaviť automatizáciu za pár eur až do stoviek tisícov. Záleží na presnosti, citlivosti a rozsahu, ktorý si aplikácia vyžaduje.

2.1 Kúrenie

Moja práca pojednáva o jednej z najdôležitejších vecí, ktorú je potrebné regulovať - a tou je kúrenie. V minulosti bola jediná možnosť ako ovládať teplotu v miestnosti jednoduchý škrtiaci ventil pripojený na vykurovacie teleso. Takýto ventil nie je reguláciou v pravom slova zmysle. Regulácia je proces, pri ktorom je jedna veličina stále evidovaná (meraná) a porovnávaná s inou veličinou (riadiacou veličinou) a v závislosti na výsledku porovnania v zmysle prispôsobenia sa meraná veličina upraví. Tento proces prebieha v uzatvorenom obvode tzv. regulačnom obvode. Najjednoduchší príklad je termostatický ventil (TRV). Vysvetlenie sa nachádza v nasledujúcej kapitole.

3 Regulácia prietoku

3.1 Termostatický ventil

K regulácii tepelného výkonu vykurovacích telies sa používa efekt tepelnej objemovej rozťažnosti. Bežnou súčasťou vykurovacích systémov je termostatický ventil, ktorý zabraňuje prekurovaniu miestnosti a reguluje prietok teplej vody podľa teploty v miestnosti. Tým výrazne prispieva k úsporám tepla a financií. Inštalácia takýchto ventilov, na rozdiel od obyčajných, môže priniesť úsporu 5 % až 15 % spotrebovanej energie.

Obr. 1: Termostatický ventil

Dnes existuje množstvo rôznych druhov termostatických ventilov, ale aj napriek tomu je základný fyzikálny princíp podobný. Skladajú sa z ventilovej časti, ktorá je napevno osadená v potrubí, ktorú ovláda prvok rôznych podôb a foriem automatizácie.

V hlavici termostatického ventilu je kvapalina s vysokou hodnotou teplotného súčiniteľa objemovej rozťažnosti, obyčajne sa používa lieh. S rastúcou teplotou vzduchu v miestnosti sa zväčšuje objem tejto kvapaliny. Vďaka rastúcemu objemu vznikne tlak, ktorý sa prenáša v tele ventilu na kuželku ventilu, ktorá začne zatvárať prívod teplej vody. Kvôli menšiemu prietoku teplej vody vo vykurovacom telese sa znižuje jeho výkon a tak sa reguluje teplota vzduchu v miestnosti na potrebnú nastavenú hodnotu.

Keď teplota v miestnosti klesne, kvapalina svoj objem zmenší, kuželka sa z potrubia postupne vytiahne a tým púšťa do vykurovacieho telesa teplú vodu z kotla.

Požadovanú teplotu v miestnosti nastavíme však na termostatickej hlavici.

3.2 Termostatická hlavica

Termostatická hlavica posúva kužeľku termostatického ventilu a tým mu dáva viac alebo menej priestoru pre pohyb. Týmto spôsobom nastavuje užívateľ požadovanú teplotu na stupnici, ktorá zodpovedá rozmedziu teplôt približne 12°C až 28°C. Ďalej sa môžu na stupnici nachádzať symboly 0 (temperovanie) a * (protimrazová ochrana). Nižšia teplota sa nastavuje napríklad na noc alebo pri opustení bytových priestorov. Pri dlhšom otvorení okna je vhodné nastaviť na hlavici číslo 0 aby bol ventil úplne uzatvorený. Inak by bola hlavica ventilu ochladzovaná vonkajším vzduchom a vykurovacie teleso by odovzdávalo veľký tepelný výkon ale všetko teplo by unikalo oknom. Pri dlhšej neprítomnosti je vhodné nastaviť hlavicu na symbol hviezdičky (vločky). Tento symbol predstavuje protimrazovú ochranu miestnosti. Teleso potom nekúri vôbec, až dokým teplota v miestnosti neklesne pod asi 5 - 7 °C (podľa výrobcu). Tým odpadá nebezpečenstvo, že sa byt podchladí až príliš a voda v potrubí zamrzne, zväčší svoj objem a roztrhne ho.

Obr. 2: Termostatická hlavica

Hlavica termostatického ventilu by nemala byť zakrytá závesom alebo schovaná pod krytom telesa, ktoré bráni prúdeniu vzduchu okolo hlavice a tá potom reaguje na nesprávnu teplotu. Pri podlahovom vykurovaní alebo zakrytých telies sa tak musia použiť hlavice s oddeleným tepelným senzorom, ktoré sa umiestni tak, aby dobre reagovalo na teplotu vzduchu v miestnosti.

Ako už bolo spomínané, pokiaľ je termostatická hlavica vhodne nastavená a je zaručené prúdenie okolného vzduchu okolo hlavice, môže tak kombináciou termostatického ventilu a hlavice dôjsť k značným úsporám v kúrení (5 – 15 %). K úsporám vo vykurovaní môže teda dôjsť ak je termostatická hlavica nastavená na hodnotu, pri ktorej nedochádza k zbytočnému prekurovaniu vykurovanej miestnosti. Nastavenie závisí len na užívateľovi a je teda len na ňom, ako bude nastavovať termostatickú hlavicu. Tu vzniká nevýhoda termostatickej hlavice – je nutné ju nastavovať manuálne na požadovanú hodnotu. Užívateľ teda musí pri opustení bytu prísť ku každej hlavici a nastaviť požadovanú teplotu. Inak bude vykurovaný priestor zbytočne vyhrievaný na komfortnú teplotu aj v prípade neprítomnosti osôb. Na druhej strane sa ale môže stať, pri znížení teploty pri odchode, že užívateľ príde do studeného bytu.

Bolo by teda viac ako vhodné, keby došlo k zníženiu alebo zvýšeniu požadovanej

teploty automaticky. Túto funkciu splňujú elektronické termostatické hlavice.

3.3 Elektronická termostatická hlavica

Elektronické termostatické hlavice zaručujú rovnakú funkciu ako štandardné termostatické hlavice. Avšak obsahujú navyše programátor, ktorý umožňuje, aby hlavica udržiavala v požadovanej dobe nastavenú teplotu. Tým dochádza k podstatnej úspore energie bez zníženia komfortu.

Elektronické hlavice obsahujú snímač, regulátor, programátor a akčný člen. Programátor, podľa dopredu nastaveného programu, oznamuje regulátoru akú teplotu užívateľ práve požaduje. Regulátor potom podľa skutočnej teploty a požiadavky užívateľa riadi akčný člen. Akčný člen ovláda prietok teplonosného média do vykurovacieho telesa cez kuželku termostatického ventilu. Na rozdiel od štandardných termostatických hlavíc, nefungujú tieto hlavice na princípe tepelnej rozťažnosti kvapaliny.

Obr. 3: Programovateľná termostatická hlavica

Elektronické termostatické hlavice najčastejšie obsahujú odporový teplotný senzor. U takéhoto snímača sa potom vplyvom zmeny priestorovej teploty v miestnosti zmení odpor. Hodnota odporu je potom prevedená na elektrické napätie a zmeny napätia ovládajú akčný člen. Elektronické termostatické hlavice obsahujú taktiež displej, na ktorom sa pre užívateľa zobrazuje aktuálna teplota a požadovaná teplota. Nastavovanie požadovanej teploty je umožnené napríklad pomocou otočného prvku alebo pomocou tlačidiel.

Pre komfortnejšiu reguláciu je potrebné riadiť elektronickú hlavicu programom (sadou inštrukcií) - firmvérom, ktorý je spojením samotnej regulácie hlavice s užívateľským prostredím pre naprogramovanie rôznych cyklov vykurovania. Firmvér je základným programom, ktorý sprostredkováva vnútornú komunikáciu všetkých obvodov elektronickej hlavice a umožňuje zobrazenie informácií na displeji. Užívateľ si firmvér neprogramuje, ten je súčasťou zariadenia od výroby. Užívateľ ale môže programovať časový program pre každý deň v týždni. Najčastejšie sa skladá tento program z dvoch teplôt. Jedna úroveň je označovaná ako „komfortná teplota“ a

druhá je označovaná ako „úsporná teplota“. Komfortná teplota určuje nastavenie termostatickej hlavice do režimu vykurovania. Je to teplota, ktorá charakterizuje požiadavka na vykurovanie miestnosti keď je užívateľ prítomný. Naopak úsporná teplota označuje teplotu, pri ktorej nie je požadované miestnosť vykurovať. Používa sa teda vtedy, keď nie je užívateľ prítomný.

Takmer každá elektronická hlavica obsahuje tri módy ovládania. Prvý je mód automatický. Pri tomto móde je dodržiavaný užívateľom naprogramovaný režim. Druhý mód je manuálny. V tomto režime je ignorovaný nastavený denný program a hlavica reguluje teplotu v miestnosti len podľa nastavenia pomocou otočného prvku (respektíve pomocou tlačidiel). Tretí je mód, ktorý udržuje vykurovanie miestnosti na protimrazovú teplotu (väčšinou nastavenú na 5 °C od výroby).

Ak by užívateľ zmenil teplotu v automatickom režime, hlavica by vykurovala miestnosť na danú nastavenú teplotu ale program by sa nezmenil. Mení sa len momentálna nastavená teplota. Ak by chcel užívateľ zmeniť režim, musel by sa celý naprogramovať odznovu.

Ďalšou zaujímavou funkciou elektronickej termostatickej hlavice je tzv. funkcia „otvorené okno“. Táto funkcia zaručuje uzatvorenie prietoku teplonosnej látky cez termostatický ventil. Reaguje na prudký pokles teploty v okolí hlavice. Týmto dochádza k značnej úspore energie. Akonáhle je okno zatvorené, teplota v miestnosti začne narastať a hlavica sa vráti do pôvodného režimu vykurovania.

Akým spôsobom reaguje hlavica na zmenu teploty a ako je udržovaná požadovaná teplota popisuje tzv. „princíp regulácie“. Súčasné hlavice sa dodávajú buď s reguláciou PI (proporcionálne integračnou) alebo „Fuzzy logika“. Bežný užívateľ nerozozná aký princíp regulácie je využitý v jeho hlavici. Principiálne ide len o to, akým spôsobom je dosiahnutá požadovaná teplota v miestnosti.

Čo sa týka napájania, existujú dva najrozšírenejšie spôsoby. Prvý spôsob je napájanie pomocou tužkových batérií. Výdrž takýchto batérií je približne dva roky. Ďalším spôsobom je napájanie hlavice pomocou napájacieho adaptéru. Závisí to na výrobcovi. Niektorí výrobcovia umožňujú užívateľovi oba spôsoby.

Medzi ďalšie funkcie elektronických termostatických hlavíc patrí napríklad „automatické precvičenie“ a „detská poistka“. Funkcia automatického precvičenia je výhodná hlavne v letných mesiacoch. Znamená to, že hlavica otvorí ventil naplno a opäť ho uzavrie približne raz za týždeň. Tak sa nemôže stať, že by kužeľka ventilu zatuhla. Funkcia detskej poistky dokáže zablokovať hlavicu pomocou stlačenia kom-

binácie tlačidiel. Potom nie je možné hlavicu nastavovať ani meniť naprogramovaný režim.

Z uvedeného vyplýva, že ak sú na termostatických ventiloch nasadené programovateľné termostatické hlavice a je nastavený vhodný program vykurovania, je tak možné dosiahnuť úspor pri vykurovaní až 20 %. Každé vykurovacie teleso obsahujúce termostatický ventil, by malo správne obsahovať taktiež termostatickú hlavicu, ktorá riadi kuželku ventilu. Takáto regulácia vykurovacieho výkonu je vhodná predovšetkým pre systémy s diaľkovým vykurovaním (teplo dodávané z teplárne), kde je do vykurovacích telies nepretržite prívádzané teplo. Otázkou ostáva aký systém regulácie je vhodný pri lokálnom vykurovaní?

4 Termostat

4.1 Lokálne vykurovanie

Lokálne vykurovanie, používané väčšinou v rodinných domoch, sa dnes stále častejšie používa aj v bytových jednotkách. To znamená, že každá bytová jednotka má vlastný zdroj tepla. V tom spočíva princíp lokálneho vykurovania. Majiteľ objektu je teda zodpovedný za ovládanie svojho zdroja. Distribúcia tepla v bytovej jednotke (resp. rodinnom dome) a regulácia výkonu je len na ňom. Najčastejšie je ako zdroj tepla používaný kotol. Ten môže byť elektrický, plynový (najčastejšie), resp. kondenzačný. Súčasťou každého kotla je čerpadlo, ktoré zabezpečuje distribúciu teplonosného média (spolu s potrubím) do jednotlivých vykurovacích telies.

Aby kotol produkoval teplo, je potrebné zapnúť ohrev vody a následne spustiť jej distribúciu. Podľa druhu použitého kotla je nutné ovládať buď horák kotla, alebo elektrickú špirálu (v elektrických kotloch). Za súčasť kotla je potrebné považovať čerpadlo a prípadne zásobník na teplú vodu, ktorý niektoré kotle obsahujú. Vzniká otázka, ako takýto spôsob kúrenia ovládať a kedy kotol spínať. Bolo by teda vhodné mať zariadenie, ktoré na základe požadovanej teploty dokáže automaticky kotol spínať. Takéto zariadenie sa nazýva priestorový termostat.

4.2 Priestorový termostat

Priestorové termostaty slúžia na porovnávanie aktuálnej a užívateľom požadovanej priestorovej teploty. Súčasťou každého termostatu je tepelný snímač, ktorý meria okolitú priestorovú teplotu. Termostat takisto obsahuje funkčné tlačítka alebo otočný prvok, pomocou ktorých sa nastavuje požadovaná teplota. Termostat ďalej obsahuje spínací kontakt, najčastejšie relé. Toto relé je určené na spínanie kontaktov pripojeného zariadenia (horák, čerpadlo, elektrická špirála, atď.). Princíp termostatu je nasledovný. Odmeraná teplota je porovnávaná s teplotou požadovanou (nastavenou užívateľom). Ak je odmeraná priestorová teplota nižšia ako požadovaná teplota, výstupné relé termostatu sa zopne a tým je spustený ohrev teplonosného média v kotle. Akonáhle odmeraná teplota presiahne hodnotu požadovanej teploty, relé rozopne kontakty a kotol sa vypne. Tento jednoduchý princíp je vhodný použiť práve pri lokálnom spôsobe vykurovania.

4.2.1 Rozdelenie priestorových termostatov

Dnes existuje na trhu veľa rôznych druhov termostatov. Môžeme ich rozdeliť na mechanické a digitálne (podrobnejšie v nasledujúcej kapitole). Ďalším rozdelením je delenie podľa zón na jednozónové a viac zónové. Jedno zónové termostaty merajú teplotu práve v jednej miestnosti a na základe tejto teploty ovládajú zdroj tepla. Za zmienku stojí ešte jedno rozdelenie termostatov a to na drôtové a bezdrôtové. Drôtové termostaty sú priamo spojené so zdrojom tepla. Bezdrôtové termostaty sa skladajú z dvoch jednotiek. Jedna je jednotka priestorového termostatu a druhá je tzv. reléová jednotka. Jednotka priestorového termostatu je umiestnená vo vykurovanej miestnosti a reléová jednotka je umiestnená v blízkosti kotla, ktorý spína. Priestorový termostat v tomto prípade teda neobsahuje relé. To je umiestnené v reléovej jednotke, ktorá je ovládaná bezdrôtovou jednotkou priestorového termostatu. Jedným z obmedzení bezdrôtového systému je dosah. Väčšina výrobcov garantuje dosah 30 m v otvorenom priestore. Nevýhodou bezdrôtových jednotiek je možné vzájomné rušenie s inými bezdrôtovými aplikáciami v rovnakom frekvenčnom pásme (Wi-Fi, bluetooth, mikrovlná rúra, bezdrôtové periférie k počítačom). Tieto siete používajú frekvenčné pásmo 2,4 GHz, resp. 5 GHz, nakoľko to nepodlieha licenčným poplatkom. Výber spôsobu je len na užívateľovi.

4.2.2 Umiestnenie priestorových termostatov

Existuje množstvo doporučení pre umiestnenie termostatu. Nie je však nikde presne definované. Jedným z nich je umiestnenie termostatu minimálne 1,5 m nad podlahu. Nemal by byť umiestnený v blízkosti žiadneho zdroja tepla. Tak isto nie je vhodné termostat umiestňovať ku dverám, k oknu alebo na priame slnečné lúče. Umiestnenie je ideálne v miestnosti, podľa ktorej chceme vykurovať priestor. Ak byt obsahuje viacero miestností, je potrebné umiestniť termostat na miesto kde sa užívateľ zdržuje najčastejšie (obývacia izba, spálňa, atď.). Jednoznačná odpoveď na umiestnenie termostatu však neexistuje.

4.2.3 Možnosti priestorových termostatov

Jednou z užitočných funkcií termostatu je zníženie teploty v prípade neprítomnosti obyvateľov. Jednoduchšie termostaty obsahujú tlačítko, ktoré by mal užívateľ stlačiť pri odchode a tým dať termostatu vedieť, že odchádza a tak môže termostat

vykurovanie vypnúť resp. udržovať teplotu na minimálnej hodnote (obyčajne 15 °C). Po príchode užívateľ stlačí tlačítko opäť, kúrenie sa obnoví a termostat udržuje komfortnú teplotu. Samozrejme, že k úspore energie dôjde, ale môže sa stať, že užívateľ na stláčanie takéhoto tlačítka zabudne. Preto je výhodnejšie keď termostat umožňuje užívateľovi naprogramovať časový rozvrh. Obvykle sa rozvrh nastavuje na týždeň. Túto funkciu obsahujú len digitálne priestorové termostaty. Po nastavení rozvrhu termostat spína kotol podľa neho. Podobne ako to bolo u programovateľných termostatických hlavíc, ktoré boli popísané v predchádzajúcej kapitole.

Jednou z negatívnych vlastností priestorových termostatov je tzv. cyklovanie, resp. cyklické spínanie zdroja tepla. Príliš časté spínanie a vypínanie plynových horákov ich veľmi rýchlo opotrebováva a spotreba plynu rastie. Práve preto niektoré termostaty obsahujú funkciu teplotnej diferencie a funkciu pre nastavenie počtu cyklov zapnutia za požadovanú minimálnu dobu. Teplotná diferencia je výhodná v prípade keď nedochádza k prudkým výkyvom teploty. Ak termostat zopne relé pri určitej teplote a vytápa priestor, nevypne však pri okamžitom prekročení nastavenej teploty ale pokračuje v kúrení nad požadovanú úroveň (väčšinou o 1 °C). Rovnaká funkcia je aj pri znížení teploty pod nastavenú úroveň. Tým sa zabráni častému spínaniu kotla. Avšak ak priestorová teplota príliš kolíše, je vhodnejšie zvoliť funkciu pre nastavenie počtu zopnutí kotla na minimálnu požadovanú hodnotu. Väčšinou sa nastavuje počet zopnutí za hodinu. Takže ak nastavíme počet zopnutí na šesť za hodinu, relé zopne kotol maximálne každých desať minút. Takto môžeme zabrániť príliš častému spínaniu zdroja tepla.

Mechanické priestorové termostaty (obrázok Obr. 5) patria medzi základné priestorové termostaty, ktoré sú v súčasnej dobe ponúkané. Tieto termostaty obsahujú väčšinou iba otočný prvok, na ktorom je tepelná stupnica. Požadovaná teplota sa nastavuje práve pomocou tohoto otočného prvku. Na rozdiel od digitálnych priestorových termostatov, nepotrebujú mechanické priestorové termostaty žiadny zdroj napájania. Ako snímač teploty slúži najčastejšie membránový teplotný element (obrázok Obr. 4), ktorý je naplnený najčastejšie plynom. Vplyvom tepelnej rozťažnosti plynu, pôsobí plyn na membránu, ktorá potom tlačí na príslušný kontakt. Tento kontakt spína výstup priestorového termostatu a teda zapína, resp. vypína kotol. Takýto termostat neobsahuje žiadne relé iba spínacie kontakty. Prípadne prepínacie kontakty, ktoré je potom možné využiť na spínanie chladenia. Tepelný rozsah je najčastejšie od 10 °C do 30 °C. Je tak isto možné tento typ priestorového termostatu použiť na protimrazovú ochranu. V tomto prípade je rozsah od 0 °C do 20 °C.

Obr. 4: Vnútro mechanického priestorového termostatu

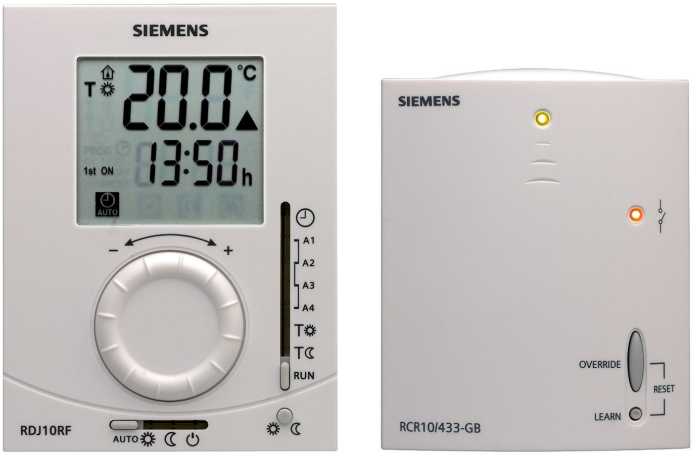
Obr. 5: Mechanický priestorový termostat

4.2.4 Digitálne priestorové termostaty

Na rozdiel od mechanických priestorových termostatov, majú digitálne priestorové termostaty viacero funkcií, ktoré prispievajú k dosiahnutiu optimálnej tepelnej pohody a k šetreniu nákladov na kúrenie. Na začiatku kapitoly je popísaná funkcia týždenného režimu a tak isto funkcia nastavenia počtu cyklov spínania zdroja tepla na minimálnu dobu.

Výhodou digitálnych priestorových termostatov je, že obsahujú displej, ktorý zobrazuje väčšinou tie najdôležitejšie údaje - aktuálna teplota, nastavená požadovaná teplota, dátum, čas, časový program, indikácie zapnutého zdroja tepla, atď. Každý digitálny priestorový termostat obsahuje tri základné režimy. Prvým režimom je vypnutie funkcie priestorového termostatu. Nejde však o úplné vypnutie, je to režim protimrazovej ochrany. Druhým režimom je manuálny režim. V tomto režime si užívateľ ručne nastaví požadovanú teplotu. Táto teplota ostane na termostate nastavená až do doby kým ju užívateľ opäť ručne nezmení. Termostat teda naďalej meria teplotu v priestore, porovnáva ju s teplotou požadovanou a podľa toho spína zdroj tepla. Tretím základným režimom je režim automatický. Tento režim riadi teplotu v priestore práve podľa nastaveného týždenného rozvrhu. Samozrejmosťou je, že užívateľ môže tento rozvrh kedykoľvek meniť a upravovať.

Niektoré priestorové termostaty majú ešte navyše ďalšie režimy ako napríklad párty alebo dovolenka. Nastavením daného režimu (napríklad pomocou extra tlačítka na termostate) sa teplota upraví na dopredu nastavenú. Napríklad režim dovolenka môže nastaviť teplotu na 18 °C po dobu 7 resp. 14 dní. Po uplynutí tejto doby sa termostat vráti naspäť do pôvodného týždenného režimu.



Obr. 6: Digitálny priestorový termoststst - Siemens

Veľkou výhodou niektorých digitálnych priestorových termostatov je pripojenie externého senzoru teploty. Takýto externý senzor môže nahradiť ten vstavaný, alebo môže slúžiť ako informatívny, napríklad pre vonkajšiu teplotu. Niektoré digitálne priestorové termostaty poskytujú možnosť pripojenia vzdialeného ovládania. Takéto ovládanie môže zabezpečovať napríklad GSM[1] modul. Tento GSM modul obsahuje SIM kartu, na ktorú je možné poslať SMS správu (rovnako ako na iný telefón) a tým napríklad vypnúť resp. zapnúť zdroj tepla. Tak isto môže termostat po vyžiadaní poslať SMS správu s informáciou o aktuálnej teplote.

Ďalším rozdielom oproti mechanickým priestorovým termostatom je to, že digitálne priestorové termostaty vyžadujú napájanie a obsahujú relé. Ako zdroj napájania sa často používajú batérie. Výdrž je väčšinou dva roky. Výstupné relé štandardne obsahuje prepínacie kontakty a je tak možné využiť termostat ako pre funkciu kúrenia tak pre funkciu chladenia.

Tieto termostaty je možné zakúpiť ako bezdrôtové. Termostat potom ale neobsahuje relé. Relé je umiestnené v oddelenej jednotke ako je to na obrázku Obr. 6. Vľavo je termostat a vpravo je reléová jednotka.

[1]GSM - Global System for Mobile Communications (globálny systém pre mobilnú komunikáciu)

4.3 Zónové regulačné systémy

Hlavnou nevýhodou priestorových termostatov je, že termostat je umiestnený vždy v jednej miestnosti (referenčnej – najviac obývanej) a na základe okolitej priestorovej teploty v tejto miestnosti je ovládaný zdroj tepla. Celý systém vykurovania je teda riadený na základe teploty v tejto danej miestnosti. V dnešnej dobe už ale existujú systémy, ktoré umožňujú regulovať vykurovaciu sústavu na základe priestorovej teploty vo viacerých miestnostiach nezávisle. Takéto systémy sa nazývajú zónové.

Pojem zóna je adekvátny pojmu priestor. Vo vykurovaní sa ide o vymedzený priestor (konkrétna miestnosť), ktorá sa má vykurovať na požadovanú teplotu. Význam celého pojmu zónový systém teda znamená: Systém umožňujúci individuálne vykurovanie v jednotlivých miestnostiach.

Zónové systémy sa využívajú tak ako v rodinných domoch (s lokálnym kúrením) tak aj v panelových domoch (s diaľkovým kúrením). Rozdielom však je použitie reléovej jednotky, ktorá sa používa pri lokálnom kúrení pri spínaní zdroja tepla podľa skutočnej potreby. U diaľkového kúrenia sú namiesto relé modulov používané servopohony na ventily pre jednotlivé zóny. Základom zónového systému je centrálna riadiaca jednotka. Táto jednotka je správcom všetkých ostatných prvkov v systéme a vydáva povely k ich ovládaniu. Riadiaca jednotka obsahuje často displej, na ktorom zobrazuje stav v jednotlivých zónach a tak isto slúži ako rozhranie medzi užívateľom a ostatnými systémami. V riadiacej jednotke sa celý systém konfiguruje. Je možné napríklad nastavovať, ktoré komponenty patria do ktorej zóny, ktorý senzor patrí ku ktorému relé resp. pohonu.

V samotnej riadiacej jednotke sa zadávajú časové a teplotné programy pre jednotlivé zóny. Riadiaca jednotka môže prijímať informácie o aktuálnych priestorových teplotách z jednotlivých tepelných senzorov umiestnených v jednotlivých zónach (miestnostiach). Rovnako môže vysielať požiadavky na otváranie a zatváranie pohonov ventilov u jednotlivých vykurovacích telesách.

Zónové systémy je možné všeobecne rozdeliť na dve skupiny. Najstaršie zónové systémy boli iba vodičové. To znamená, že jednotlivé komponenty systému boli medzi sebou prepojené vodičmi. Veľkou výhodou vodičového zónového systému je spoľahlivosť. Prepojovanie vodičmi bolo však nevýhodné pri integrácii tohto systému do už vybudovaného vykurovacieho systému. Druhú skupinu zónových systémov tvoria bezdrôtové zónové systémy. U nich je využitá bezdrôtová technológia nízko-

rýchlostnej komunikácie, najčastejšie vo frekvenčnom pásme 868 – 870 MHz.

Okrem riadiacej jednotky sa systém skladá z nasledovných komponentov. Nástenné snímače vnútornej teploty, snímač vonkajšej teploty, termoelektrické pohony, elektronické regulátory vykurovacích telies, zónové regulátory podlahového vykurovania a reléová spínacia jednotka. Ďalej to však môžu byť aj komponenty zabezpečujúce komfort v danej zóne. Napríklad senzor intenzity slnečného žiarenia, senzor rýchlosti vetra, rôzne spínacie jednotky, pohon žalúzií, moduly pre diaľkové ovládanie prostredníctvom GSM a moduly umožňujúce vizualizáciu celého systém na počítači alebo prostredníctvom siete Internet.

4.3.1 Vodičové zónové systémy

Vodičové zónové systémy je možné použiť pre riadenie vykurovania pomocou vykurovacích telies, podlahového vykurovania alebo ich kombinácie. Ako už bolo spomínané, základom celého systému je centrálna riadiaca jednotka. Ďalšou dôležitou súčasťou systému je zónový regulátor. Zónový regulátor slúži pre ovládanie komponentov, ktoré sú k nemu pripojené. Ak budeme uvažovať základnú aplikáciu kúrenia pomocou vykurovacích telies, tak týmito komponentami budú jednotlivé termoelektrické pohony. Termoelektrický pohon je veľkosťou podobný termostatickej hlavici, a takisto sa nasadzuje na radiátorový ventil, ale je možné ho ovládať elektrickým napätím. Najčastejšie je toto napätie buď 230 V AC alebo 24 V AC/DC[2]. Tieto pohony sa najčastejšie vyrábajú v dvoch prevedeniach. Prvý z nich je termoelektrický pohon bez napätia otvorený (normaly open - NO) a druhý je bez napätia uzavretý (normaly closed - NC). Výber závisí na konkrétnej aplikácii. Zónový regulátor môže taktiež ovládať relé, ktoré môže spínať horák kotla. Princíp regulácie je nasledovný. Riadiaca jednotka je prepojená so zónovým regulátorom. K zónovému regulátoru sú pripojené jednotlivé nástenné snímače priestorovej teploty a termoelektrické pohony, ktoré sú nasadené na termostatických ventiloch vykurovacích telies. V riadiacej jednotke sú nastavené časové a teplotné programy pre jednotlivé zóny. Riadiaca jednotka posiela do zónového regulátora požadované teploty. Tieto teploty sú následne porovnávané s aktuálnymi priestorovými teplotami. V prípade, že je aktuálna priestorová teplota nižšia ako teplota požadovaná pre danú zónu, ovláda zónový regulátor termoelektrický pohon. Riadením termoelektrického pohonu sa zdvihne kuželka termostatického ventilu a teplonosné médium je vpustené do telesa. V opačnom prípade, ak je nameraná teplota vyššia ako teplota požadovaná,

[2]AC (Alternating current) - striedavý prúd; DC (Direct current) - jednosmerný prúd.

ventil je rovnakým spôsobom uzavretý. Princíp regulácie je obdobný priestorovému termostatu. Ak je pripojený kotol, tak je horák kotla ovládaný pri požiadavke o kúrenie v akejkoľvek miestnosti. Ak budeme uvažovať aplikáciu pomocou podlahového kúrenia, tak je princíp rovnaký. Jedinou zmenou je to, že v prípade podlahového kúrenia je v systéme umiestnený rozdeľovač. Zónový regulátor je potom umiestnený v priestore rozdeľovača. Princíp ale ostáva nezmenený. Ak je požiadavka na kúrenie (požadovaná teplota je vyššia ako aktuálna), otvorí sa príslušný termoelektrický pohon a podlaha sa začne postupne ohrievať.

4.3.2 Bezdrôtové zónové systémy

Hlavnou nevýhodou vodičového zónového systému je nutnosť prepojenia všetkých komponentov pomocou vodičov. Vďaka tomu, že v bezdrôtovom systéme komunikujú jednotky medzi sebou bezdrôtovo, táto nevýhoda odpadá. Bezdrôtový zónový systém, rovnako ako vodičový, je možné využiť pre riadenie kúrenia vykurovacími telesami, podlahovým kúrením alebo ich kombináciou. Ďalšou výhodou bezdrôtového zónového systému je, že v prípade riadenia vykurovacími telesami nie je nutné použiť zónový regulátor. Zónový regulátor môže byť nahradený bezdrôtovými regulátormi (termostatickými hlavicami) vykurovacích telies, ktoré sú umiestnené v jednotlivých zónach. Princíp bezdrôtovej zónovej regulácie je nasledovný. V riadiacej jednotke je nastavený požadovaný časový program v požadovanými teplotami. Regulátory vykurovacích telies sú nasadené na termostatických ventiloch. Slúžia ako na ovládanie zdvihu kuželky termostatického ventilu, tak aj ako snímače priestorovej teploty. To znamená, že komunikácia medzi riadiacou jednotkou a regulátorom vykurovacieho telesa je obojsmerná. V riadiacej jednotke je potom porovnávaná požadovaná teplota s aktuálnou nameranou teplotou. Podľa výsledku porovnania týchto teplôt je vyslaný signál späť regulátoru na otvorenie alebo uzavretie ventilu.

5 Inteligentné termostaty

Inteligentné termostaty sú zariadenia domácej automatizácie zodpovedné za riadenie vykurovania a občas aj klimatizácie v domácnosti. Umožňujú užívateľovi regulovať teplotu svojho domu počas celého dňa pomocou nastaveného rozvrhu, ako napríklad nastavenie nižšej teploty cez noc. Inteligentné termostaty sú spravidla pripojené k Internetu. Tým umožňujú užívateľom nastavovať požadovanú teplotu pomocou zariadení, ktoré sú tak isto pripojené k sieti Internet – napríklad mobilné

telefóny (smartfóny). To umožňuje užívateľovi zapnúť respektíve vypnúť kúrenie alebo klimatizáciu keď nie je nikto doma. Toto uľahčenie je nevyhnutné pre zabezpečenie úspory energie. Niektoré štúdie ukázali, že domácnosti s programovateľnými termostatmi majú dokonca vyššiu spotrebu energie ako tie domácnosti s jednoduchými termostatmi. Je to najčastejšie spôsobené zvýšenou zložitosťou nastavenia, takže ich užívatelia zle naprogramujú (nastavia časový rozvrh) alebo vypnú ich funkcionalitu úplne. Najznámejšími inteligentnými termostatmi sú *Nest Learning Thermostat*, *Honeywell Lyric* a *Ecobee3* viď obrázok Obr.7.

Obr. 7: *Nest Learning Thermostat*, *Honeywell Lyric* a *Ecobee3*

5.1 Termostaty *so schopnosťou učiť sa*

Niektoré druhy inteligentných termostatov sú schopné sa automaticky *učiť*, teda postupne upravovať program na základe predchádzajúcich vstupov užívateľa. Kedy je užívateľ doma a kedy nie. Takisto sa termostat učí podľa užívateľom nastavených teplôt. Zapamätá si kedy a akú teplotu mal užívateľ nastavenú, a podľa toho vytvára režim. Zo začiatku je termostat v takzvanej *učiacej sa* fáze. V tejto fáze je nastavený od výroby, a postupne sa učí režim novej domácnosti. Táto funkcionalita umožňuje automaticky vykúriť alebo vychladiť domácnosť pred tým než užívateľ príde domov. Ak sa režim v domácnosti zmení, termostat sa postupne naučí a zmenu aplikuje do režimu kúrenia (chladenia). Najznámejším termostatom so schopnosťou učiť sa je *Nest Learning Thermostat*.

5.2 Porovnanie existujúcich zariadení

Najznámejšími predstaviteľmi inteligentných termostatov sú *Nest Learning Thermostat*, *Honeywell Lyric* a *Ecobee3*.

Všetky tri regulujú teplotu, ale iným spôsobom:
Nest sa "učí" pomocou algoritmov. Časom sa naučí kedy užívateľ vstáva, kedy ide spať a kedy odchádza z domu a podľa toho nastavuje teplotu.

Na druhej strane *Lyric* používa *geofence*[3] technológiu na zistenie či je užívateľ prítomný alebo byt opustil. Táto funkcia však vyžaduje aby mal užívateľ smartfón stále zo sebou.

Ecobee3 používa tzv. funkciu *Smart Home/Away*. Proces začína u užívateľa, ktorý si nastaví vlastný rozvrh. Termostat tento rozvrh sleduje a nastavuje podľa neho teplotu. Je ale dostatočne šikovný na to, aby si program zmenil, ak užívateľ príde domov napríklad skôr. To je zabezpečené pohybovými senzormi. Ďalšou funkciou termostatu *Ecobee3* je tzv. funkcia *Follow me*. Umožňuje užívateľovi nastaviť, ktoré senzory má brať do úvahy a ktoré nie. Senzory, ktoré má brať termostat do úvahy následne sledujú aj pohyb a tak zabezpečujú pohodlie v miestnosti, kde sa užívateľ práve nachádza.

Kompatibilita:
Najviac kompatibilný je *Nest*. Spolupracuje napríklad s bezpečnostnými kamerami *Dropcam*, kódovými zámkami od *Yale*, zariadeniami od spoločnosti *Philips* atď. *Nest* má takisto aj vlastnú bezpečnostnú kameru *Nest Cam*, vlastný detektor dymu *Nest Protect*.

Lyric zatiaľ nie je kompatibilný s cudzími zariadeniami. Ale *Honeywell* pracuje na spolupráci s *Apple's Homekit*.

Ecobee3 je súčasťou programu *Homekit* a pracuje s *IFTTT*[4] Práve prostredníctvom *IFTTT* je schopný sa pripojiť k zariadeniam ako *Scout*, *SmartThings* a veľa ďalších. Priamo sa dokáže pripojiť k zariadeniam *SmartThings*, *Control4* a *Vera*.

Hardvér:
Nest obsahuje vstavanú lithium-ion batériu. Napájanie termostatu je ale možné dvoma spôsobmi. Jedným je priložený adaptér a druhým je napájanie priamo z káblov od zdroja tepla (pre 24Voltové systémy).

Lyric obsahuje vymeniteľnú batériu a termostat sleduje a oznamuje keď je

[3]GeoFence – sledovanie opustenia a vstupu do nastavenej zóny

[4]IFTTT - bezplatná webová služba, ktorá umožňuje užívateľom vytvárať reťazce jednoduchých podmienených príkazov, ktoré sa spúšťajú na základe iných webových služieb, ako je Gmail, Facebook, Instagram, atď. Je to skratka z anglického *If This Then That*.

batéria slabá.

Ecobee3 potrebuje spoločný C (common) kábel. V prípade absencie C-káblu v systéme ponúka spoločnosť tzv. *Power Extender Kit*, ktorý tento problém rieši.

Najväčším rozdielom, ktorý odlišuje *Ecobee3* od termostatov *Nest* a *Lyric* je ten, že *Ecobee3* používa termostat a oddelené senzory. V cene termostatu je práve jeden senzor, ale je možné ho rozšíriť až na viac ako 30, ak je to potrebné. Tie je potrebné umiestniť do vzdialenosti maximálne 13,7 metrov (45 stôp) od termostatu. Sú použité na zvýšenie inteligencie termostatu, ktorý tak sníma teplotu vo viacerých miestnostiach. Je to výhodné ak je užívateľ doma, ale nie je v rovnakej miestnosti ako samotný termostat.

Tabuľka 1: Porovnanie inteligentných termostatov

	Nest	Lyric	Ecobee3
Alarm extrémnych teplôt	Áno	Áno	Áno
Vlastné nastavenie teploty	Áno	Áno	Áno
Mód *Away*	Áno	Áno	Áno
Mód spánku	Áno	Áno	Áno
Počasie	Áno	Áno	Áno
Mobilná aplikácia	Áno	Áno	Áno
Nastavenie cez PC	Áno	Nie	Áno
Vymeniteľná batéria	Nie	Áno	Nie
Alarm slabej batérie	Nie	Áno	Len senzory
Velkosť [cm]	7,62	9,53	10x10
Veľkosť LCD [cm]	5,28	3,6	8,9
LCD reagujúci na pohyb	Áno	Áno	Áno
IFTTT	Áno	Nie	Áno
Cena [CZK]	5929,70	4715,00	5739,20

Inteligentné termostaty sú mini a mikro počítače s rôznym prevedením užívateľského rozhrania. Minipočítač je konštrukčne väčšinou jednoprocesorové zariadenie s pomocnými obvodmi pre rozhrania (displej, ovládací panel, sieťové rozhranie a pod.).

Zariadenia rôznych výrobcov sú väčšinou aj konštrukčne takmer zhodné ale líšia sa softvérovým vybavením a dizajnom ovládacích prvkov, teda komfortom obsluhy.

Základ zariadenia tvorí jednočipový mikroprocesor rôznych výrobcov (Atmel, Microchip (General Instrument), Texas Instruments, STMicroelectronic, Freescale Semiconductor (Motorola), a pod.) a podporné obvody, zväčša už v prevedení hotových jednodoskových minipočítačov (Arduino, Raspberry Pi, Banana Pi, a pod.), resp. aj s prevedením s "vážnejším"procesorom a podstatne väčším výkonom, čo je základom aj napríklad pre mobilné telefóny.

V praktickej časti svojej práce som navrhol jednoúčelové ovládanie, ktoré principiálne môže nahradiť inteligentné ovládanie doprogramovaním rozšírenej funkcionality a pripojení ďalších snímacích a ovládacích prvkov.

6 Arduino

Arduino je open-source[5] platforma postavená na mikrokontroléri ATMega od spoločnosti Atmel a grafickom vývojovom prostredí, ktoré vychádza z prostredia Wiring a Processing. Arduino môže byť použité k vytváraniu samostatných interaktívnych zapojení alebo môže byť pripojené k softvéru na počítači.

Systém poskytuje súbor digitálnych a analógových vstupno-výstupných pinov, ktoré tvoria rozhranie s rôznymi rozširujúcimi doskami alebo inými obvodmi.

Dosky majú sériové komunikačné rozhranie, vrátane USB u niektorých modelov, pre nahrávanie programov z osobných počítačov. Pre programovanie mikrokontrolérov Arduino poskytuje integrované vývojové prostredie (IDE[6]) naprogramované v Jave na základe projektu Processing, ktorý zahŕňa podporu programovacích jazykov C a C++.

Dnes sú Arduino dosky pre európsky trh označované ako *Genuino*, obrázok Obr. 8.

Obr. 8: Arduino (Genuino) UNO

[5]open-source je označenie programov, ktorých zdrojový kód bol poskytnutý verejnosti ale hlavne ďalším vývojárom, ktorí ho môžu študovať a väčšinou aj upravovať a tým ďalej vylepšovať.

[6]IDE - Integrated Development Environment (Integrované vývojové prostredie)

6.1 ATMega328

Základom dosky Genuino UNO je mikrokontrolér ATMega328. Tento čip obsahuje 14 digitálnych vstupov a výstupov a 6 analógových vstupov.

Pracovné napätie čipu je 5 V. Flash pamäť o veľkosti 32 kB, z čoho je 0,5 kB použité bootloaderom. Samotný čip ATMega328 obsahuje EEPROM pamäť o veľkosti 1 kB.

Niektoré zo spomínaných pinov majú špeciálne určenie.

- Piny 0(RX) a 1(TX) sú používané na prijímanie (RX) a vysielanie (TX) TTL sériových dát.
- Piny 3, 5, 6, 9, 10, a 11 poskytujú pulznú šírkovú moduláciu.
- Piny 10(SS), 11(MOSI), 12(MISO), 13(SCK) podporujú SPI (Serial Peripheral Interface) komunikáciu.

Každý zo šiestich analógových vstupov označených A0 až A5 poskytuje 10 bitové rozlíšenie (t.j. 1024 hodnôt). Štandardne merajú a počítajú od nuly do piatich voltov. Je samozrejme jednoduché zmeniť tento rozsah pomocou externého referenčného napätia pripojeného na pin AREF.

Digitálne piny sa dajú pomocou softvéru nastaviť ako vstup alebo výstup pomocou jednoduchého príkazu *pinMode()*. Týmto spôsobom môžeme k doske Arduino pripojiť rôzne senzory, akčné členy, rozširujúce dosky atď. Tak isto je možné pripájať na analógové vstupy senzory, ktoré majú analógový výstup.

Aby bolo možné programovať mikrokontrolér ATMega328 pomocou Arduino IDE (popísanom v nasledujúcej kapitole), je potrebné, aby bol na programovanom čipe nahraný bootloader[7].

[7]Bootloader je program, ktorý zavádza operačný systém do počítača (v mojom prípade firmvér do mikrokontroléru).

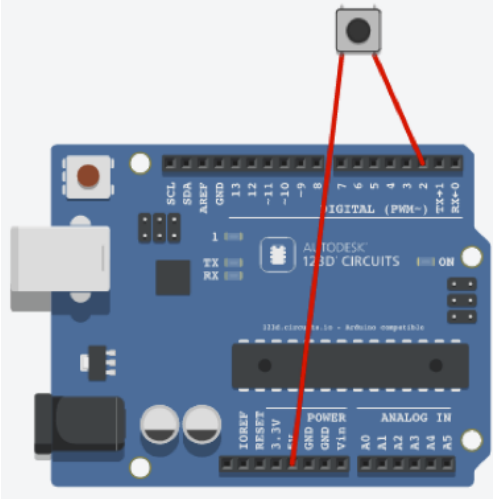
6.2 Arduino IDE

Arduino poskytuje integrované vývojové prostredie (IDE) naprogramované v Jave na základe projektu Processing, ktorý zahŕňa podporu programovacích jazykov C a C++. Avšak vďaka knižniciam, a veľkej podpore od ľudí z komunity okolo Arduina sa programovací jazyk zjednodušil na úroveň vhodnú pre začiatočníkov, ktorí nemajú s programovaním žiadnu skúsenosť. Nasledujúce kapitoly ukazujú príklady jednoduchých zapojení pomocou Arduina.

6.3 Ukážky zapojení

6.3.1 Digitálny vstup

Príkladom jednoduchého digitálneho vstupu je jednoduchý obvod, kde pomocou obyčajného tlačítka a 5V zdroja privedieme na pin nastavený ako vstup kladný impulz (5V). Zapojenie podľa obrázku Obr. 9. Na jeden pin tlačidla je pripojených +5 V zo zdroja, ktorým je samotná doska Arduina, a druhý pin je pripojený na digitálny pin (na obrázku pin 2), ktorý je softvérom nastavený ako vstup — *pinMode(2,INPUT)*. Týchto 5V reprezentuje logickú 1 alebo stav HIGH. Logickú 0 reprezentuje 0V teda stav LOW.



Obr. 9: Arduino — digitálny vstup

Toto v podstate jednoduché zapojenie môže spôsobiť ale v praxi veľký problém.

Obyčajné tlačítko, ako to na Obr. 9, spôsobí po stlačení zákmity. Kvôli týmto zákmitom môže mikrokontrolér dostať nesprávnu informáciu o stlačení respektíve nestlačení tlačítka. Tento problém je možné vyriešiť napríklad pridaním kondenzátora paralelne k tlačítku (hardvérové riešenie) alebo napísať krátku funkciu, ktorá tento problém vyrieši tak, že mikrokontrolér počká určitú dobu, a ak na začiatku aj na konci je tlačítko v rovnakom stave, tak považuje informáciu za správnu. Tento časový interval je dostatočne malý na to, aby to človek nezaznamenal (približne 100ms). Viac o tomto probléme je možné nájsť napríklad na oficiálnych stránkach Arduino [4].

Program pre spracovanie digitálneho vstupu je nasledovný:

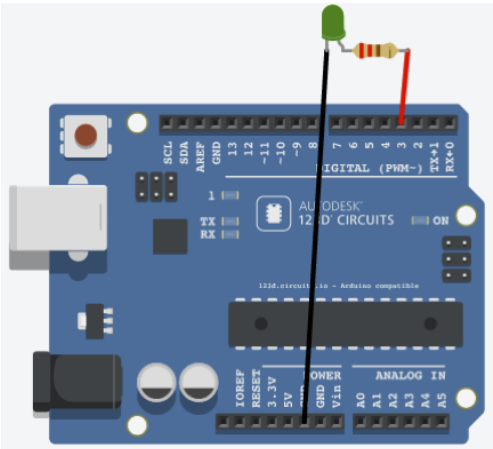
```
int val;

setup(){
    pinMode(2,OUTPUT);
}
loop(){
    val = digitalRead(2);
    if ( val == HIGH ) {
        // prikazy
    } else {
        // prikazy
    }
}
```

Na začiatku je deklarovaná premenná *val*, do ktorej program číta stav pinu 2. Vo funkcii *setup()* je definovaný pin 2 ako vstup. Po tejto definícii nasleduje funkcia *loop()*, v ktorej do premennej *val* prečítame stav na digitálnom vstupe 2 pomocou príkazu *digitalRead()*. Nasleduje klasické rozcestie *if*, ktorým zistíme či je hodnota premennej *val* logická 1, teda stav HIGH. V príklade s tlačítkom by tento jednoduchý program zistil, či je tlačítko stlačené alebo nie. Po tomto vyhodnotení môžeme ďalej pracovať.

6.3.2 Digitálny výstup

Podobným spôsobom dokážeme ovládať akčné členy pripojené k pinom kontroléru, ktoré sú nastavené ako výstupy. Jednoduchým príkladom je LED dióda,

ktorú ak pripojíme na výstupný pin nastavený na stav HIGH, zasvieti. Samozrejme je potrebné ju pripojiť cez oddeľovací rezistor kvôli zníženiu prúdu prechádzajúceho diódou. Na obrázku Obr.10 je zapojená anóda diódy na digitálny pin nastavený ako výstup (na obrázku pin 3) cez 220 Ω rezistor. Katóda diódy je pripojená na pin GND, ktorý reprezentuje uzemnenie, respektíve 0 V.



Obr. 10: Arduino — digitálny výstup

Program pre spracovanie digitálneho výstupu je nasledovný:

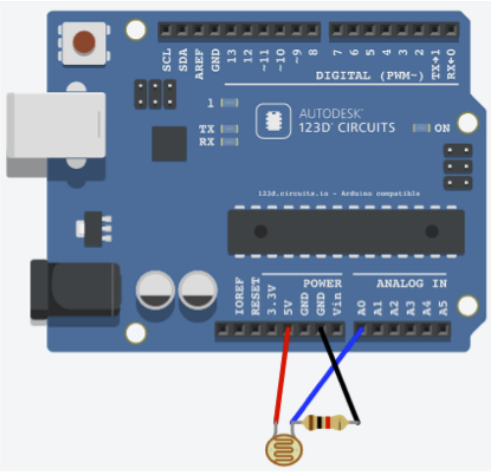
```
setup(){
    pinMode(3,OUTPUT);
}
loop(){
    digitalWrite(3,HIGH);
    delay(1000);
    digitalWrite(3,LOW);
    delay(1000);
}
```

Na začiatku je vo funkcii *setup()* definovaný pin 3 ako výstup. Po tejto definícii nasleduje funkcia *loop()*, v ktorej striedavo zapisujeme stavy HIGH a LOW pomocou funkcie *digitalWrite()*. Medzi jednotlivými zápismi je použitá funkcia *delay()*, ktorá pozastaví program na dobu určenú v milisekundách. Po nahraní tohoto programu bude LED dióda pripojená na pin 3 blikať v sekundových intervaloch.

6.3.3 Analógový vstup

Väčšina senzorov, napríklad na meranie intenzity osvetlenia, má však analógový výstup. Nastaviť analógové vstupy Arduina je rovnako jednoduché ako digitálne. A následné spracovanie signálu je rovnaké. Keďže má Arduino vstup pre USB, vieme zosnímané dáta preniesť pomocou zabudovaného prevodníku TTL - USB do počítača a zobrazovať priamo na obrazovke. Tak isto ako digitálne dáta môžu byť aj analógové dáta spracovávané programom a riadiť akčné členy. Pred spracovaním analógových dát je potrebné ich previesť na digitálne. K tomu slúži zabudovaný 10 bitový analógovo-digitálny prevodník.

Jednoduchú ukážku vieme simulovať na svetlocitlivom rezistore, ktorý mení svoj odpor v závislosti na intenzite osvetlení. Na obrázku Obr.11 je zapojený takýto svetlocitlivý rezistor. Z jednej strany je pripojený na 5V a druhý koniec je pripojený na pull-down[8] rezistor pripojený k zemi. A spojenie medzi premenlivým svetlocitlivým rezistorom a pevným rezistorom je vyvedený na analógový vstup (na obrázku pin A0).



Obr. 11: Arduino — analógový vstup

[8] pull-down/pull-up rezistor: zaisťuje definovanú hodnotu na odpojenom vstupe. Ak teda dôjde z nejakého dôvodu k odpojeniu vstupu, má súčiastka stále nejakú informáciu a nenastane nestabilný stav.

Program pre spracovanie digitálneho vstupu je nasledovný:

```
int LDR = A0;
int readLDR;

setup(){
    Serial.begin(9600);
}
loop(){
    readLDR = analogRead(LDR);
    Serial.println(readLDR);
    delay(250);
}
```

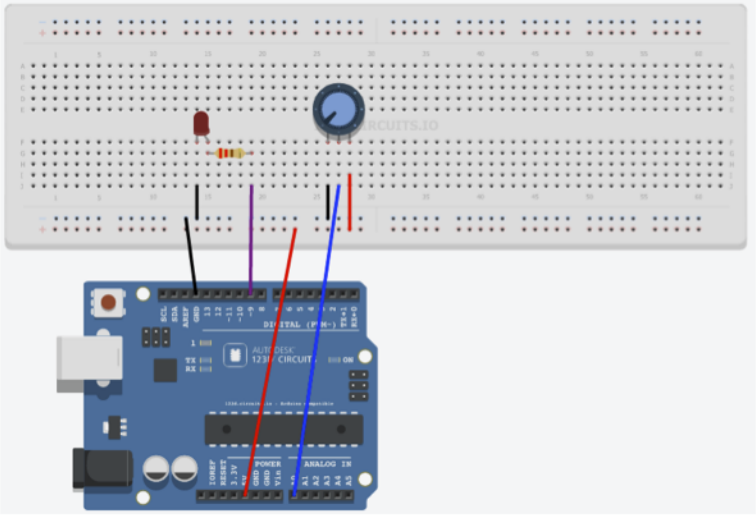
Na začiatku je definovaná premenná LDR, ktorá drží číslo analógového vstupu a premenná *readLDR*, do ktorej bude program ukladať prečítané dáta. LDR je skratka z anglického názvu *Light Dependant Resistor*. V funkcii *setup()* je pomocou príkazu *Serial.begin(9600)* nastavená rýchlosť sériového prenosu medzi Adruinom a počítačom (*baudrate*). Číslo *9600* reprezentuje rýchlosť prenosu dát v bitoch za sekundu. V cykle funkcie *loop()* najprv prečítame hodnotu na analógovom vstupe A0 a hodnotu vložíme ho do premennej *readLDR*. Hodnotu uloženú v premennej *readLDR* vypíšem pomocou príkazu *Serial.println(readLDR)* priamo do dialógového okna na počítači. Pridané písmená *ln* znamenajú výpis na nový riadok. Pozastavenie *delay(250)* je v programe len pre obmedzenie počtu výpisov.

6.3.4 Analógový výstup

Vzhľadom k tomu, že Arduino nemá analógový výstup, môžeme použiť digitálny výstup označený na doske Arduino vlnovkou. Týchto vstupov je na doske Arduino UNO dohromady päť. Je na nich možné použiť pulzne šírkovú moduláciu (PWM - Pulse Width Modulation). Ovládajú sa pomocou príkazu *analogWrite(pin, hodnota)*, kde pin je číslo výstupného pinu a hodnota je číslo v rozsahu jedného bajtu, teda 0 - 255. Frekvencia výstupného signálu je približne 490 Hz. Príkladom je ovládanie jasu LED diódy pomocou potenciometra. Zapojenie podľa obrázku Obr.12.

Program pre spracovanie digitálneho výstupu je nasledovný:

```
int potPin = A0;
int LEDPin = 9;
int readValue;
int writeValue;

setup(){
    pinMode(LEDPin, OUTPUT);
}
loop(){
    readValue = analogRead(potPin);
    writeValue = (255./1023.)*readValue;
    analogWrite(LEDPin, writeValue);
}
```



Obr. 12: Arduino — analógový vstup a výstup

V deklarácií premenných nastavíme pin, na ktorý je pripojený potenciometer do premennej *potPin*. Pin, na ktorý je pripojená LED dióda do premennej *LEDPin*. A zadeklarujeme premennú *readValue*, do ktorej neskôr vložíme hodnotu prečítanú z potenciometra a premennú *writeValue*, do ktorej vložíme hodnotu, pomocou ktorej budeme ovládať jas LED diódy.

Vo funkcii *setup()* nastavíme pin LED diódy ako výstup. V cykle *loop()* najskôr prečítame hodnotu z potenciometra, a následne podľa vzorca odvodeného pre lineár-

nu závislosť medzi hodnotami z potenciometra (0 - 1023) a hodnotou pre ovládanie jasu LED diódy (0 - 255). Túto vypočítanú hodnotu vložíme do pripravenej premennej *writeValue*, ktorú môžeme následne pomocou príkazu *analogWrite()* zapísať na pin LED diódy, kde sa PWM postará o prevedenie hodnôt 0 - 255 na napätie pre diódu.

Arduino, ale samozrejme aj veľa ďalších mikrokontrolérov poskytujú široké možnosti uplatnenia. V nasledujúcej kapitole popisujem konkrétnejšie zapojenia a zariadenia, ktoré som následne využil vo vlastnom projekte.

7 Centrálna jednotka - *Raspberry Pi*

Ako centrálnu riadiacu jednotku a mozog celého systému používam mini počítač *Raspberry Pi 2 model B* (ďalej ako riadiaca jednotka). Je to počítač o niečo väčší ako kreditná karta, ktorý je možné pripojiť ku počítačovému monitoru alebo televízoru pomocou HDMI, a ktorý používa štandardnú klávesnicu a myš. Je to malé ale schopné zariadenie, ktoré umožňuje výuku programovania v jazykoch ako je *Scrach* alebo *Python*. Je schopné všetkého čo by užívateľ mohol očakávať od bežného stolného počítača. Od prehliadania Internetu a prehrávania videa s vysokým rozlíšením po vytváranie tabuľkových a textových dokumentov a hranie hier. Navyše obsahuje *GPIO* piny, ktoré je možné použiť k interakcii s okolím a s najrôznejšími prídavnými zariadeniami.

Na mojej centrálnej jednotke je operačný systém *raspbian* (čo je vlastne odvodená unixová distribúcia pre *Raspberry Pi* od oficiálnej distribúcie *Debian*). Pod týmto operačným systémom som nainštaloval server *MQTT Broker mosquitto*, webový server *Apache* a databázový server *MySQL*. Operačný systém už obsahuje podporu programovacieho jazyka *python*, v ktorom som napísal hlavný riadiaci skript. V centrálnej jednotke teda bežia všetky tri služby - MQTT, web a databáza.

8 Konkrétne zapojenia a jednotlivé časti projektu

V tejto kapitole sú popísané príklady konkrétnych zapojení, ktoré som použil v projekte. Opis je všeobecný pomocou platformy Arduino. Popis konkrétnej dosky, vytvorenej pre mikroprocesor je popísaná v kapitole 10. Takisto však obsahuje mikrokontrolér ATMega328. Kompletný pohľad na systém je na obrázku 25.

8.1 Meranie teploty

Meranie teploty je zabezpečené digitálnym teplomerom DS18B20. Je to jednoduchá súčiastka s relatívne veľkou presnosťou (9 - 12 bit). Komunikuje cez 1-Wire zbernicu, ktorá podľa definície potrebuje na komunikáciu s mikroprocesorom jedinú dátovú linku a uzemnenie. Tým pádom stačí na pripojenie teplomera jednoduchá dvojlinka.

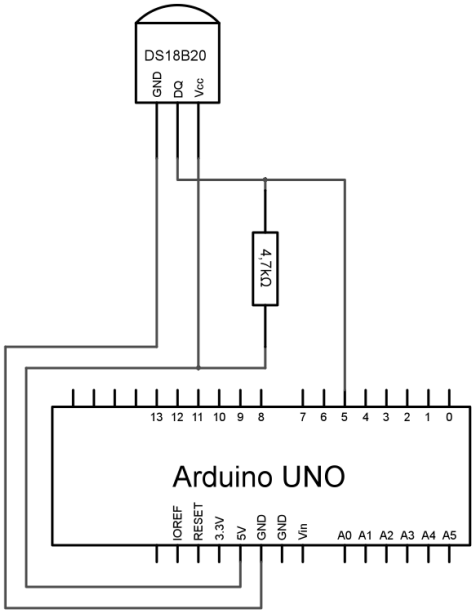
Rozsah teplôt je od -55 °C do +125 °C a presnosťou ±0,5 °C v rozsahu -10 °C až +85 °C. Takáto presnosť stačí pre aplikáciu v mojom projekte. Veľkou výhodou tohto teplomeru je tzv. *parasite power*, čo znamená, že teplomer môže byť napájaný priamo z dátovej linky, takže nepotrebuje externé napájanie. Výhodou tiež je, že teplomer odosiela informáciu o nameranej hodnote priamo v stupňoch Celzia a tak nie je potrebný ďalší prepočet nameraných hodnôt.

Každý teplomer DS18B20 má od výroby dané 64-bitové jedinečné číslo, ktoré slúži ako adresa a tak umožňuje viacerým teplomerom a iným 1-Wire zariadeniam fungovať na jednej zbernici. Teda jeden mikroprocesor dokáže komunikovať s viacerými senzormi a zariadeniami (na tej istej zbernici).

8.1.1 Zapojenie teplomera DS18B20

Senzor DS18B20 má tri piny. Napájanie, dáta a zem. Napájanie je pripojené na 5 V. Zem je spojená s pinom GND a dátový pin je možné pripojiť na ľubovoľný GPIO[9] pin čipu ATMega328. Na obrázku Obr.13 je zvolený GPIO pin číslo 5, ktorý je v programe (v nasledujúcej podkapitole) definovaný ako pin pre 1-Wire zbernicu. Jediným komponentom naviac je rezistor o veľkosti 4,7 kΩ. Tento rezistor slúži ako pull-up rezistor a postačuje jediný pre celú zbernicu.

[9]GPIO – General purpose input/output



Obr. 13: Zapojenie teplomeru DS18B20

8.1.2 Softvér pre teplomer DS18B20

Na jednoduchom príklade ukážem ako sa dá prevádzkovať tento senzor samostatne spolu s mikrokontrolérom. Pre programovanie Arduina používam ArduinoIDE. Výpis softvéru je možné nájsť v prílohe C.

Za zmienku stojí funkcia *getTempC()* (viď prílohu C), ktorá vracia odmeranú teplotu priamo v stupňoch Celzia. Táto funkcia je definovaná v knižnici *DallasTemperature.h*. Program je spomalený pomocou funkcie *delay()*, čo nie je najkorektnejšie riešenie (procesor po dobu 3 sekúnd nemôže nič robiť), ale pre demonštráciu funkcie je to postačujúce. Na termináli sériovej komunikácie budú napríklad nasledovné hlášky o teplote:

Teplota: 24.50

Teplota: 24.50

Teplota: 25.00

Teplota: 26.00

Ak by som potreboval zapojiť viac senzorov na jednu zbernicu, musel by som pridať ďalší teplomer a jeho adresu a samostatne z neho prečítať teplotu. Knižnica *DallasTemperature.h* obsahuje funkcionalitu, ktorá umožňuje prehľadávať celú 1-Wire zbernicu, nájsť všetky senzory na zbernici a postupne vypísať všetky prečítané hodnoty. Pre môj projekt a demonštráciu princípu postačí teplomer s jedným senzorom.

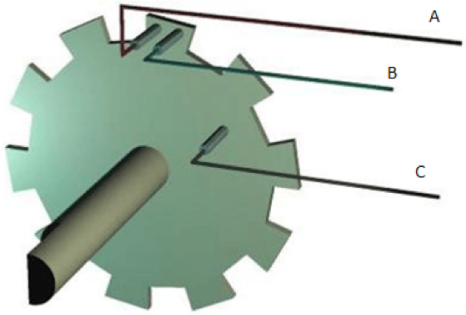
8.2 Rotačný enkóder

Rotačný enkóder je ovládací prvok, pripomínajúci "nekonečný"potenciometer. Je ním ale možné otáčať stále dookola. Je to zariadenie, ktoré poskytuje indikáciu toho „o koľko“ bol enkóder otočený a do ktorej strany ním bolo točené.

Rotačný enkóder má fixný počet pozícií na otáčku. Tieto pozície je jednoduché cítiť ako jemné kliknutia pri otáčaní enkóderom. Enkóder má tri výstupné piny. Často označované ako A, B a C. A ďalšie dva piny pre napájanie Vo vnútri enkóderu sú dva kontakty. Jeden spája A s C a druhý B s C. U každej pozícii sú oba spínače buď otvorené alebo zatvorené. Každé „kliknutie“ spôsobí zmenu stavu oboch kontaktov nasledovne:

- Ak boli oba kontakty zatvorené, otočením enkóderu v smere alebo proti smeru hodinových ručičiek spôsobí otvorenie oboch kontaktov.
- Ak boli oba kontakty otvorené, otočením enkóderu v smere alebo proti smeru hodinových ručičiek spôsobí zatvorenie oboch kontaktov.

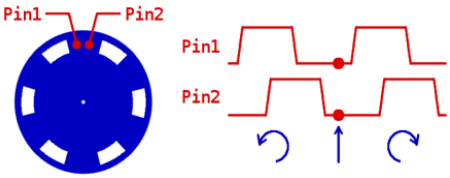
Konštrukcia týchto kontaktov je na obrázku Obr.14



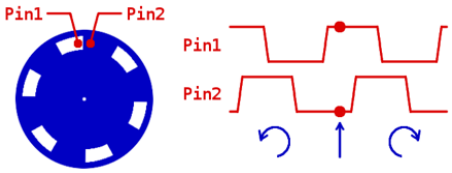
Obr. 14: Konštrukcia rotačného enkóderu

Princíp vysvetlím pomocou nasledujúcich obrázkov.

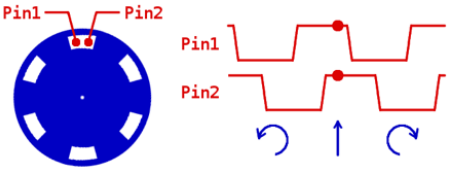
Otočenie v smere hodinových ručičiek:



Obr. 15: Východisková pozícia

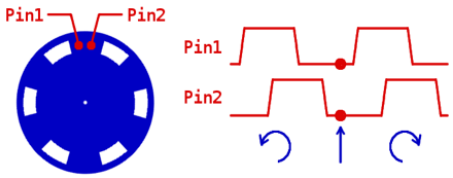


Obr. 16: Otočenie o pól pozície v smere hodinových ručičiek

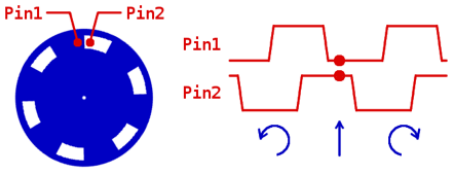


Obr. 17: Otočenie o celú pozíciu v smere hodinových ručičiek

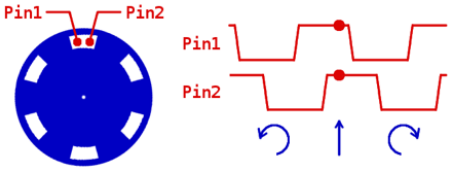
Otočenie proti smeru hodinových ručičiek:



Obr. 18: Východisková pozícia



Obr. 19: Otočenie o pól pozície proti smeru hodinových ručičiek



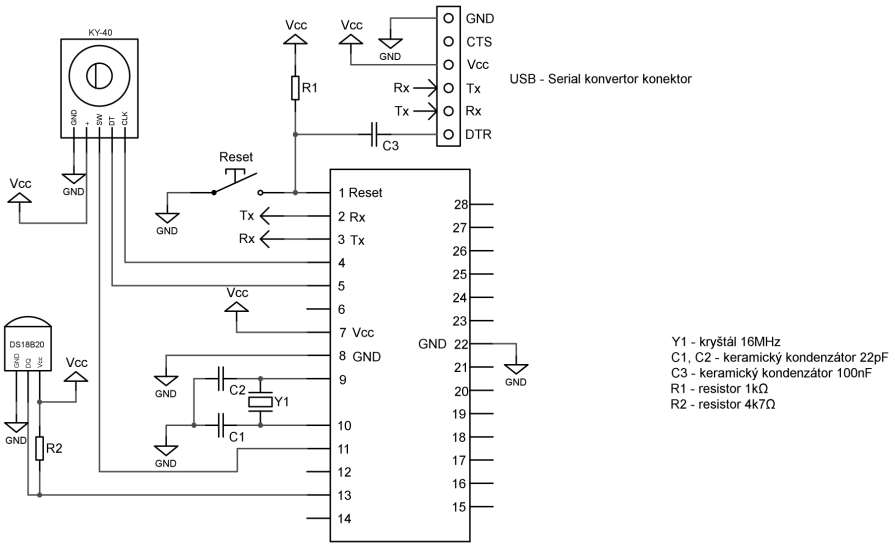
Obr. 20: Otočenie o celú pozíciu proti smeru hodinových ručičiek

Určenie, ktorý kontakt zmenil stav ako prvý, je spôsob určenia smeru otáčania enkóderu. Ak A zmení stav ako prvý, enkóder sa otáča v smere hodinových ručičiek. A ak zmení stav ako prvý kontakt B, enkóder sa otáča proti smeru hodinových ručičiek.

8.2.1 Zapojenie rotačného enkóderu

Modul je navrhnutý tak, že logická 0 je na výstupe, ak sú kontakty zatvorené a logická 1, ak sú otvorené. Logická 0 je generovaná tak, že zem je pripojená na pin C (GND) a pin C je spojený s pinmi CLK a DT, keď sú kontakty zatvorené (obr.15). Logická 1 je generovaná pomocou 5V napájania a pull-up rezistorov. Teda CLK a DT piny sú na úrovni logickej 1 práve vtedy, keď sú oba kontakty otvorené (obr.17).

Naviac je na na enkóderi vbudované tlačidlo, ktoré sa aktivuje zatlačením na hriadeľ enkódera. Toto tlačidlo je *NO – normally open* čo znamená, že po jeho zatlačení sa kontakt SW zopne a privedie sa na neho logická 0. Túto funkcionalitu ale vo svojom projekte nevyužívam.



Obr. 21: Zapojenie rotačného enkóderu KY-040

8.2.2 Softvér pre rotačný enkóder

Jednoduchý skript demonštrujúci funkciu rotačného enkóderu KY-040 je možné nájsť v prílohe D.

Skript je riešený pomocou prerušení (interrupt), ktoré čakajú na zmenu nábežnej hrany na jednom alebo druhom pine. Podľa toho, ktoré prerušenie bolo vyvolané skôr, volá skript príslušnú funkciu. Hlavný cyklus *loop()* je využitý len pre výpis polohy nastavenej enkóderom. Bližšie informácie o čítaní dát z enkóderu je možné nájsť v [5].

8.3 Internet

Najjednoduchším spôsobom ako pripojiť Arduino do siete Internet, je použiť hotové riešenie. To sa skladá z rozširujúcej dosky *Ethernet shield* a pre zjednodušenie programovania existuje základná knižnica, vyvinutá tvorcami Arduina, *Ethernet.h*. Pre Arduino ale existuje viac knižníc, s rôznymi schopnosťami a pre rôzne radiče. Doska *Ethernet shield* je síce pripojený na dosku Arduina, tak že obsadí všetky jeho piny, ale pre komunikáciu používa iba piny rozhrania *SPI* (10, 11, 12, 13). Ďalej

samozrejme napájanie na 5V. Takto sú ostatné piny naďalej prístupné. Oficiálny *Ethernet shield* používa čip W5100.

8.3.1 SPI - Serial Peripheral Interface

SPI rozhranie umožňuje sériovú výmenu dát medzi dvoma zariadeniami. Jeden je označený ako master a druhý ako slave. SPI pracuje v duplex móde. To znamená, že dáta môžu byť prenášané v oboch smeroch súčasne. SPI sa najčastejšie používa na komunikáciu medzi centrálnou jednotkou CPU a perifériami. Taktiež je možné prepojiť pomocou SPI rozhrania dve centrálne jednotky navzájom. SPI zbernica obsahuje štyri logické signály:

- SCLK - Serial Clock (Systémové hodiny), výstup z masteru
- MOSI - Master Input Slave Output
- MISO - Master Output Slave Input
- SS - Slave Select (slúži na voľbu slave zariadenia, s ktorým bude komunikovať)

Výhody tejto zbernice sú nasledovné:

- Plne duplexná komunikácia.
- Dvojčinný[10] (*push-pull*) výstup (na rozdiel od *open drain*[11]).
- Bez limitácie na 8-bitové slová.
- Slave zariadenie používajú master hodiny - odpadá potreba kryštálov resp. oscilátorov.

[10]Dvojčinný (*push-pull*) výstup je typ výstupu digitálneho elektronického obvodu, ktorý je tvorený dvojicou tranzistorov, z ktorých jeden predstavuje spínač voči napájaciemu napätiu a druhý spínač voči zemi. Bod, v ktorom sú tranzistory spojené, predstavuje samotný výstup obvodu.

[11]Otvorený kolektor (*open drain*) je druh výstupu digitálnych elektronických obvodov, kde výstupný obvod pozostáva len z jedného tranzistora, obvykle NPN alebo NMOS, spínajúceho v čase, keď je výstupom logického obvodu nulové napätie, výstup k zemi (zápornému pólu napájacieho napätia). Kolektor (drain) výstupného tranzistora je napojený priamo na výstupnú nožičku (pin) obvodu.

Nevýhody:

- Vyžaduje viac pinov ako napríklad I2C.
- Každé slave zariadenie potrebuje vlastný *Chip Select*, zariadenia nie sú adresovateľné.
- Nepodporuje dynamické pridávanie zariadení.
- Zvyčajne podporuje len jedno master zariadenie.

Priebeh komunikácie je nasledovný: pre komunikáciu nastaví master logickú 0 na SS (*Slave Select*) pine zariadenia, s ktorým chce komunikovať. Potom začne master generovať hodinový signál na SCLK (*Serial Clock*) a v tej chvíli vyšlú obe zariadenia svoje dáta. Akonáhle sú vyslané dáta, môže komunikácia ďalej pokračovať. Master ďalej dodáva hodinový signál a hodnota SS sa nemení. Alebo môže byť komunikácia ukončená. A to tak, že master prestane vysielať hodinový signál a hodnotu SS nastaví na logickú 1. Viac v [6].

8.3.2 Softvér pre Ethernet Shield

Na jednoduchom skripte ukážem ako pripojiť Arduino do siete Internet pomocou Ethernet shieldu. Ako dôkaz o pripojení do lokálnej siete nechám Arduino vypísať IP adresu na *Serial monitor* (Predpokladám, že na lokálnej sieti je DHCP server, respektíve router s touto funkcionalitou).

```
#include <SPI.h>
#include <Ethernet.h>
// MAC adresa shieldu
byte mac[] = { 0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02 };
// Spustenie instancie
EthernetClient client;

void setup() {
    // Spustenie seriovej komunikacie
        Serial.begin(9600);

        // start ethernet pripojenia
        if (Ethernet.begin(mac) == 0) {
```

```
                Serial.println("DHCP zlyhalo");
                for (;;) // nerob nic donekonecna
                        ;
        }
        // ak je pripojenie uspesne, vypis IP adresu
        Serial.print("IP adresa: ");
        Serial.print(Ethernet.localIP());
}
void loop() {

}
```

Po úspešnom pripojení program vypíše na terminál sériovej komunikácie IP adresu. Napríklad:

IP adresa: 192.168.0.102

Vypísanie IP adresy môžem považovať za dôkaz úspešného pripojenia do lokálnej siete (LAN).

8.4 LCD displej

Na zobrazovanie údajov napríklad o aktuálnej nameranej a nastavenej teplote môže slúžiť jednoduchý LCD displej. Takýto jednoduchý displej má obvykle 16 pinov, čo by zabralo väčšinu GPIO pinov mikrokontroléru. Existuje ale prídavná doska, ktorá konvertuje vstup do LCD displeju na *I2C* zbernicu, čím umožňuje pripojiť displej pomocou dvoch dátových liniek a 5V napájania.

8.4.1 I2C zbernica

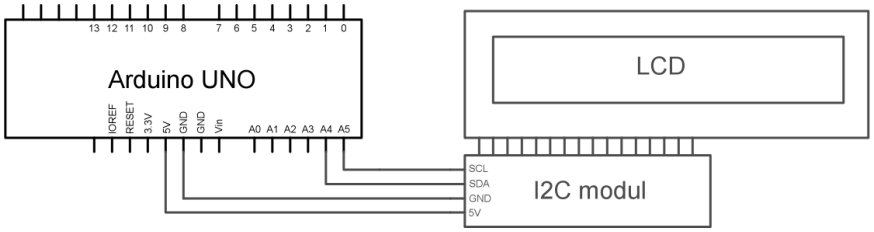
Inter-Integrated Circuit alebo I2C je dvojžilová obojsmerná zbernica vyvinutá firmou Philips začiatkom 90-tych rokov 20. storočia, používa sa predovšetkým na prepojenie pomalých periférnych zariadení s matičnou doskou.

Touto zbernicou je v súčasnosti vybavených veľké množstvo rôznych obvodov.

Zbernica ponúka možnosť jednoduchého rozširovania mikroprocesorových systémov o dodatočné obvody. Viac o zbernici I2C je možné nájsť v oficiálnej špecifikácii [7].

8.4.2 Zapojenie LCD displeja

Zapojenie pomocou I2C modulu je na obrázku Obr.22. Táto zbernica využíva dva analógové piny Arduina (A4 a A5). Tieto piny sú označované ako *SCL - clock line* a *SDA - data line*.



Obr. 22: Zapojenie LCD displeja

8.4.3 Softvér pre LCD displej

Softvér používa knižnicu, ktorá podporuje *I2C* modul a zbernicu *LiquidCrystal_I2C.h*.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// nastav LCD adresu na 0x27 pre 16x2 displej
// Nastav adresu displeja
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7,3,POSITIVE);

void setup() {
  // Nastav LCD ako 16x2 displej
  lcd.begin(16,2);
  // Nastav poziciu kurzoru
  lcd.setCursor(0,0);
  // Vypis hlasku na displej
  lcd.print("Hello, world!");
```

```
  // Pockaj jednu sekundu
  delay(1000);
}
void loop() {

}
```

9 MQTT: Message Queuing Telemetry Transport

MQTT je protokol na efektívnu výmenu krátkych stavových správ. Je jednoduchý, otvorený a navrhnutý tak, aby ho bolo jednoduché implementovať. Tieto vlastnosti sú ideálne pre zariadenia pracujúce s obmedzeným výpočtovým výkonom, malou šírkou pásma (*bandwidth*), slabou konektivitou a nespoľahlivé siete. V princípe ide o publikovanie a následný odber (*publish/subscribe*) informácií.

Tieto vlastnosti vytvárajú perfektné prostredie pre rozvíjajúci sa svet prepojených zariadení (*IoT – Internet of Things*), komunikácie medzi jednotlivými zariadeniami (*m2m – machine-to-machine*) a pre mobilné aplikácie, kde je obmedzená šírka pásma a obmedzené sú aj ďalšie parametre.

Komunikácia prebieha nad protokolom TCP/IP alebo nad inými sieťovými protokolmi, ktoré poskytujú bezstratové a obojsmerné pripojenie. Použitie vzoru správ *publish/subscribe* umožňuje distribúciu správ *one-to-many* a oddelenie aplikácií.

Existujú tri úrovne kvality doručenia, ktoré sa líšia spoľahlivosťou a rýchlosťou (*QoS – Quality of Service*):

- *najviac raz* – Pri tomto spôsobe je možnosť straty správy. Je to ale najrýchlejší spôsob výmeny informácií. Môže byť použitý v aplikácií kde takáto strata nespôsobí vážny problém. Napríklad, ak za jednou hodnotou nasleduje v krátkom časovom intervale ďalšia - tzv. *fire and forget* (vystreľ a zabudni).
- *aspoň raz* – Pri tejto úrovni je isté, že správa prijatá bude, ale môžu sa objaviť duplikáty.
- *práve raz* – Je zaistené, že správa bude prijatá presne jedenkrát. Je to najpomalší, ale najbezpečnejší spôsob odosielania a prijímania správ. Napríklad pre fakturačné systémy, kde neprijatie správy alebo jej duplikát môže spôsobiť problém.

Vzor *publish/subscribe* (publikovanie/odber) je alternatíva k tradičnému modelu klient-server, kde klient komunikuje priamo s koncovým bodom. Avšak *publish/subscribe* oddeľuje klienta, ktorý posiela určitú správu (nazývaný *publisher* od iného klienta (resp. klientov), ktorý prijíma túto správu (nazývaný *subscriber*). To znamená, že klient *publisher* a klient *subscriber* o sebe nevedia. Existuje preto tretí

komponent zvaný *broker*, ktorého obidvaja klienti poznajú. Tento *broker* (server) filtruje všetky prichádzajúce správy a ďalej ich rozdeľuje.

MQTT protokol používa filtrovanie správ na základe predmetu správy. Takže každá správa obsahuje predmet (tému - *topic*). Server (*broker*) používa tento predmet (*topic*) pre filtrovanie správ a teda, či klient prihlásený na odber túto správu dostane alebo nie.

9.1 *MQTT client* - klient

Pojmom *MQTT client* (klient) sa zahŕňajú oba druhy klientov - *publisher* aj *subscriber*. Tak isto môže byť jeden klient aj *publisher* aj *subscriber* zároveň. *MQTT client* môže byť akékoľvek zariadenie od jednoduchého mikrokontroléru po plne rozvinutý server. Takéto zariadenie potrebuje jedine bežiacu MQTT knižnicu a pripojenie k serveru *MQTT Broker* cez akúkoľvek sieť. Napríklad malý bezdrôtový mikrokontrolér s MQTT knižnicou zredukovanou na minimum alebo bežný počítač s grafickým MQTT klientom pre testovacie účely. V podstate akékoľvek zariadenie so sieťovým rozhraním a TCP/IP komunikačným protokolom. Sieťové rozhranie umožňuje (podľa použitého firmware-u, resp. operačného systému) komunikáciu niektorým z komunikačných protokolov (TCP/IP, IPX, a pod.), nad ktorým (teda vo vyššej vrstve OSI[12] modelu) komunikuje aplikácia MQTT protokolom.

Implementácia MQTT protokolu je veľmi jednoduchá a priamočiara. Je to preto jeden z aspektov, prečo je protokol MQTT vhodný pre malé zariadenia. Klientské MQTT knižnice sú k dispozícii pre veľké množstvo programovacích jazykov, ako je napríklad Arduino, C, C++, C#, Java, JavaScript, .NET, atď.

MQTT klienti si môžu nastaviť tzv. *poslednú vôľu* (*Last Will and Testament*). Je to správa, ktorá sa odošle serveru ak sa daný klient odpojí. Slúži teda ako informácia pre server a ostatných užívateľov o odpojení niektorého z klientov.

9.2 *MQTT broker* - server

Druhú časť MQTT komunikácie predstavuje server nazývaný *MQTT broker*, ktorý tvorí srdce akéhokoľvek *publish/subscribe* protokolu. V závislosti na konkré-

[12]OSI model - Open Systems Interconnection model

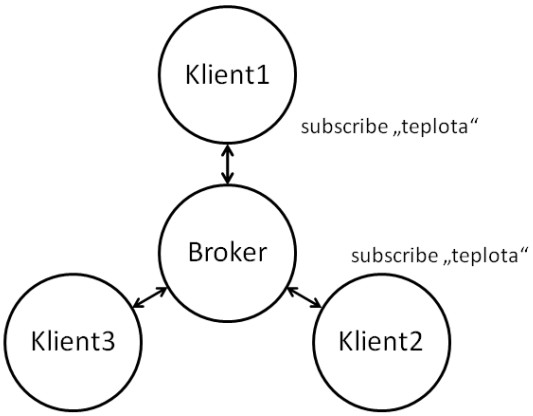
tnom prevedení, *broker* dokáže spracovať až tisíce súčasne pripojených klientov. *Broker* je zodpovedný primárne za prijímanie všetkých správ, za ich filtráciu, zisťovanie, ktorý klient je prihlásený na odber a následné odosielanie týchto správ všetkým klientom, ktorí sú na odber prihlásení. *Broker* je tak isto zodpovedný za prihlasovanie a autorizáciu klientov. Často je *broker* rozšíriteľný, čo umožňuje integrovať vlastný spôsob prihlasovania a autorizácie klientov a takisto integráciu do *back-end* systémov. Táto integrácia je veľmi dôležitá, pretože práve *broker* je pripojený priamo do siete Internet a stará sa o množstvo klientov a správ.

Viac o protokole MQTT je možné nájsť v dokumentácii *MQTT Version 3.1.1* [8] a na blogu *HiveMQ Enterprise* [10].

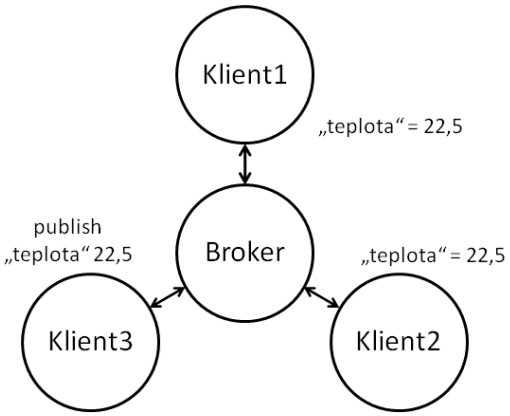
9.2.1 Príklad MQTT komunikácie

Majme jednoduchú sieť troch klientov a jedného centrálneho servera *MQTT broker*. Viď Obr.23 a Obr.24. Všetci klienti otvoria TCP spojenie so serverom. Klienti 1 a 2 sa prihlásia (*subscribe*) na odber správ s predmetom (*topic*) - "teplota". Neskôr klient 3 publikuje správu s hodnotou teploty napríklad 22,5 a odošle ju serveru s predmetom (*topic*) "teplota". Server (*broker*) správu s predmetom *topic* "teplota"prijme a následne pošle iba klientom, ktorí sú prihlásení na odber správ s týmto predmetom.

Tento model umožňuje komunikáciu *one-to-one*, *one-to-many* a *many-to-one*.



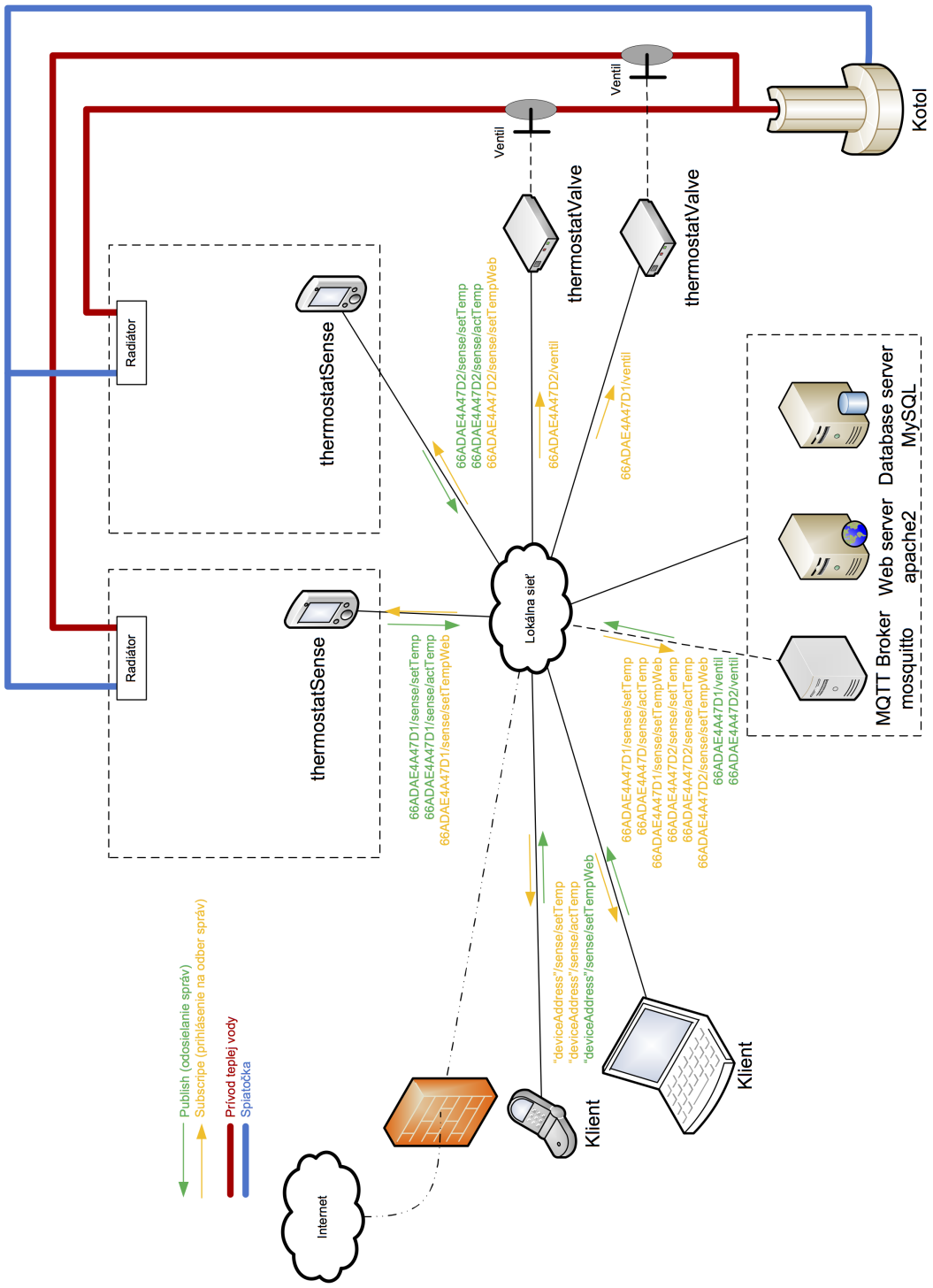
Obr. 23: MQTT subscribe



Obr. 24: MQTT publish

10 Vlastný projekt

Práca sa zaoberá návrhom, vývojom a realizáciou projektu regulácie teploty vo viacerých miestnostiach v byte s lokálnym zdrojom tepla (teoreticky je projekt použiteľný aj na diaľkový zdroj tepla). Každá miestnosť (zóna) musí obsahovať zariadenie, ktoré bude merať priestorovú teplotu v danej miestnosti (zóne) a umožní užívateľovi nastavovať požadovanú teplotu pre túto miestnosť (zónu). Toto zariadenie bude pripojené k lokálnej sieti pomocou kábla a Ethernet rozhrania. Zariadenie bude odosielať správy o nastavenej požadovanej teplote a aktuálne odmeranej priestorovej teplote pomocou MQTT protokolu na server (*MQTT broker*). Server, ktorý bude zároveň predstavovať riadiacu jednotku, tieto informácie vyhodnotí a odošle príkaz konkrétnemu ventilu, či sa má otvoriť alebo zavrieť, prípadne odošle správu pohonu o tom ako veľmi sa otvoriť respektíve zatvoriť. Kompletný pohľad na systém je na obrázku 25.



Obr. 25: Celkový pohľad na problematiku (Big Picture)

10.1 Mikroprocesor ATMega328

Pre svoj projekt som zvolil jednoduchý čip ATMega328. Tento čip obsahuje 32 kB flash pamäte 1 kB EEPROM pamäte, 2 kB SRAM pamäte. Ďalej obsahuje 23 GPIO pinov, USART programovacie rozhranie, SPI sériový port a 10-bitový A/D prevodník.

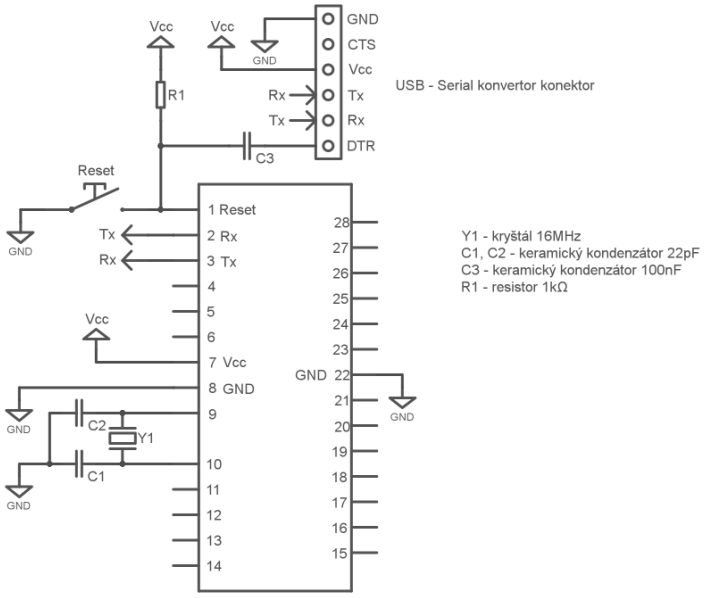
Pre tento čip som vytvoril základnú dosku, ktorá mu umožňuje fungovať bez Arduino platformy. Pre svoju funkciu potrebuje čip 16 MHz kryštál, dva keramické kondenzátory s hodnotou 22 pF a jednosmerné napájanie 5 V.

Takéto základné zapojenie je pre samostatnú funkciu čipu postačujúce (Obr.27). Aby bolo možné čip programovať, je potrebné k doske pridať konektor pre pripojenie *USB-Serial* konvertoru. Takýto konvertor má 6 pinov a je na obrázku Obr.26.

- GND - Pripojenie k zemi, resp. 0 V.
- CTS - *Clear To Send*, nevyužitý pin.
- Vcc - Napájanie 5 V z portu USB počítača.
- Rx - *Receive* (príjem) sériovej komunikácie.
- Tx - *Transmitt* (vysielanie) sériovej komunikácie.
- DTR - *Data Terminal Ready* Kontrolný signál sériovej komunikácie.

Obr. 26: USB - Serial konvertor

V tomto štádiu je možné čip programovať pomocou konvertoru (Obr.26) a softvéru Arduino IDE v počítači.



Obr. 27: Základný obvod pre čip ATMega328

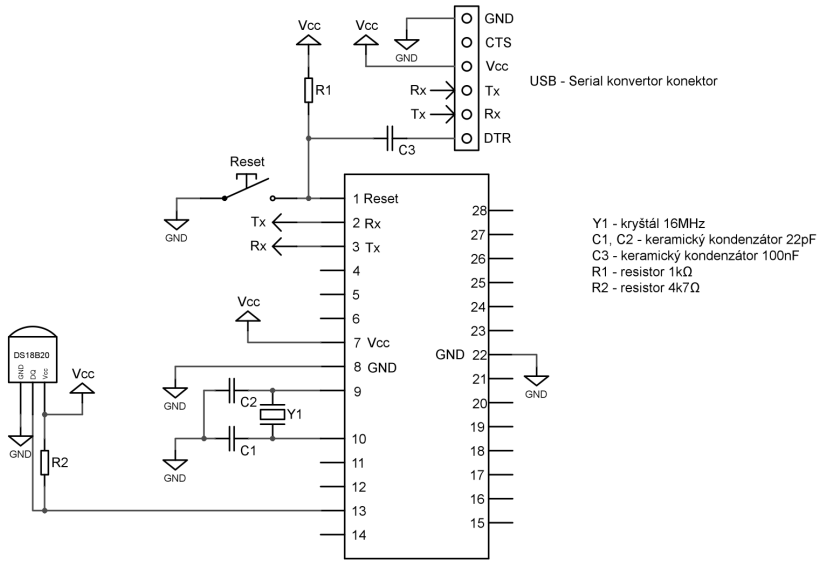
Táto doska sa stane základom pre termostat v miestnosti, ktorý meria teplotu a umožňuje jej nastavovanie a takisto aj pre vybavovacie (relé) zariadenie pre ventil (pohon).

10.2 Zariadenie *thermostatSense*

Prvé zariadenie som nazval *thermostatSense*. Toto zariadenie je umiestnené v miestnosti (resp. zóne), meria aktuálnu priestorovú teplotu v miestnosti, umožňuje užívateľovi nastavovať požadovanú teplotu a zobrazuje obe teploty na displeji. Ďalej zariadenie *thermostatSense* odosiela správy o aktuálnej priestorovej teplote a nastavenej požadovanej teplote do centrálnej riadiacej jednotky, ktorá následne vyhodnocuje nastavenie a spúšťanie kúrenia.

10.2.1 Meranie aktuálnej teploty - *DS18B20*

Prvým krokom bolo pripojenie digitálneho teplomera DS18B20 k doske. Pripojenie je nakreslené na obrázku Obr.28.



Obr. 28: ATMega328 doska s DS18B20

Kód pre teplomer je podobný kódu s kapitoly 8.1.1 s jediným rozdielom. Keďže používam len jeden teplomer, nepotrebujem jeho adresu. Je možné adresovať tento senzor ako index 0 na 1-Wire zbernici. Nasledujúca funkcia prečíta informáciu o teplote z teplomeru s indexom 0:

```
temperature = sensors.getTempCByIndex(0);
```

Táto informácia je následne konvertovaná na dátový typ *String* a odoslaná do riadiacej jednotky ako správa pod témou *66ADAE4A47D1/sense/actTemp* pomocou protokolu *MQTT*, kde *66ADAE4A47D1* je MAC adresa zariadenia. Meranie a odoslanie prebieha každé tri sekundy.

```
dtostrf(temperature, 5, 2, temperatureBuff);
mqttClient.publish("66ADAE4A47D1/sense/actTemp",
                   temperatureBuff,
                   true); // retain
```

Posledný argument funkcie *publish()* - *true* znamená, že každá odoslaná správa ostane uložená na MQTT serveri (*retain*). To znamená, že po reštarte zariadenia je táto správa odoslaná naspäť do zariadenia a to môže s touto informáciou pracovať, až do prijatia ďalšej správy pod rovnakou témou. Každá nasledujúca správa pod rovnakou témou zmaže správu predchádzajúcu a uloží novú správu.

Aktuálna nameraná teplota je takisto vypísaná na displej zariadenia:

```
lcd.setCursor(0,1); // Nastavenie pozicie kurzora
lcd.print("Act:␣");
lcd.print(temperature);
lcd.print((char)223); // char(223) je symbol stupna
lcd.print("C");
```

10.2.2 Nastavenie požadovanej teploty - *KY-040*

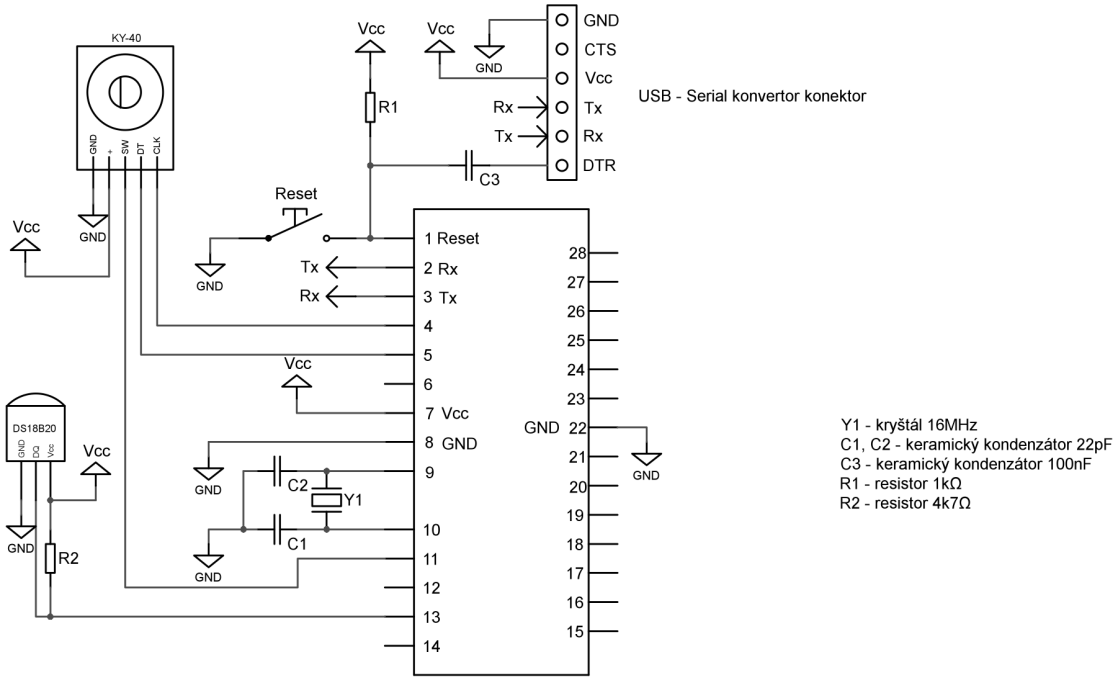
Pre nastavovanie požadovanej teploty slúži rotačný enkóder *KY-040* (zapojenie podľa Obr.29). Funkcia je popísaná v kapitole 8.2. Kód je rovnaký ako v kapitole 8.2.2. S rozdielom, že finálny softvér počíta kroky rotačného enkóderu po 0,5. Následne je poloha enkóderu uložená ako požadovaná teplota *setTemp*. Táto teplota je rovnako vypísaná na displej zariadenia a odoslaná do riadiacej jednotky pod témou *"66ADAE4A47D1/sense/setTemp"*, kde *66ADAE4A47D1* je MAC adresa zariadenia.

```
setTemp = encoderPos;

lcd.setCursor(0, 0); // Nastavenie pozicie kurzoru
lcd.print("Set:␣");
lcd.print(setTemp);
lcd.print((char)223); // char(223) je symbol stupna
lcd.print("C");

dtostrf(setTemp, 5, 2, setTempBuff);
mqttClient.publish("66ADAE4A47D1/sense/setTemp",
                   setTempBuff,
                   true); // retain
```

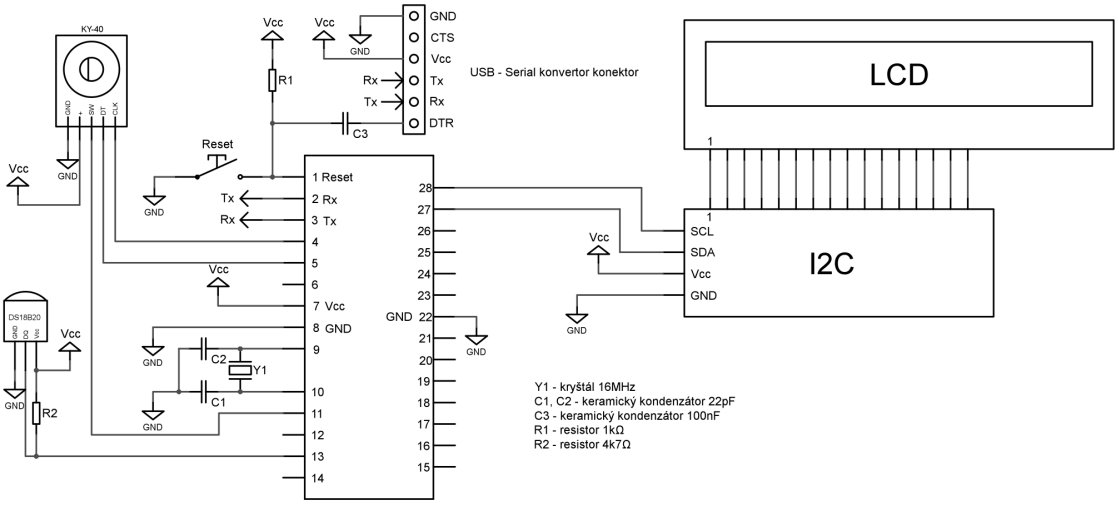
Nastavenie požadovanej teploty je obmedzené od 15 °C do 30 °C.

10.2.3 Zobrazovanie - LCD displej

Pre zobrazenie informácií pre užívateľa slúži už spomínaný LCD displej, ktorý je pripojený k základnej doske pomocou zbernice *I2C* (viď obrázok Obr.30)



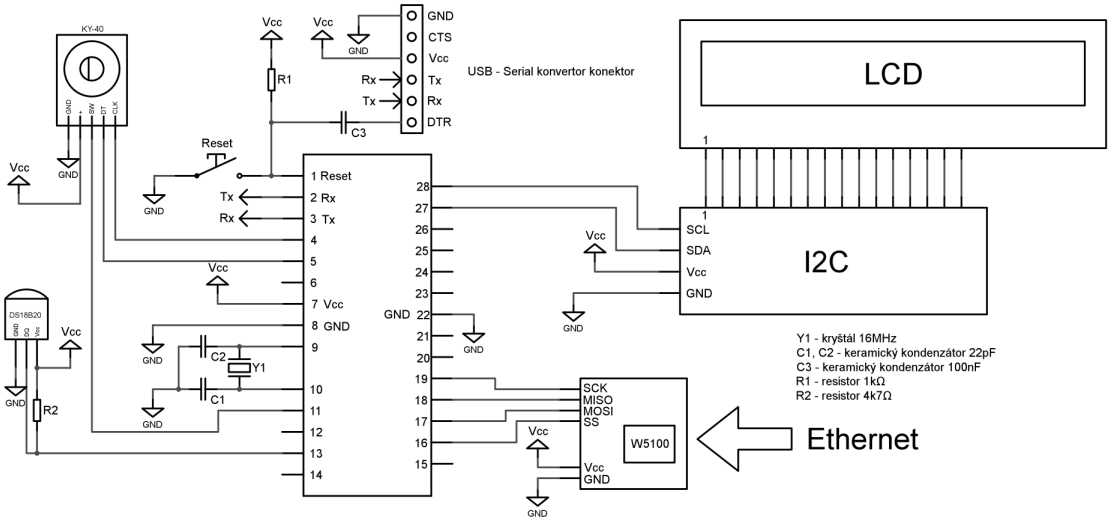
Obr. 29: ATMega328 doska s DS18B20 a KY-040



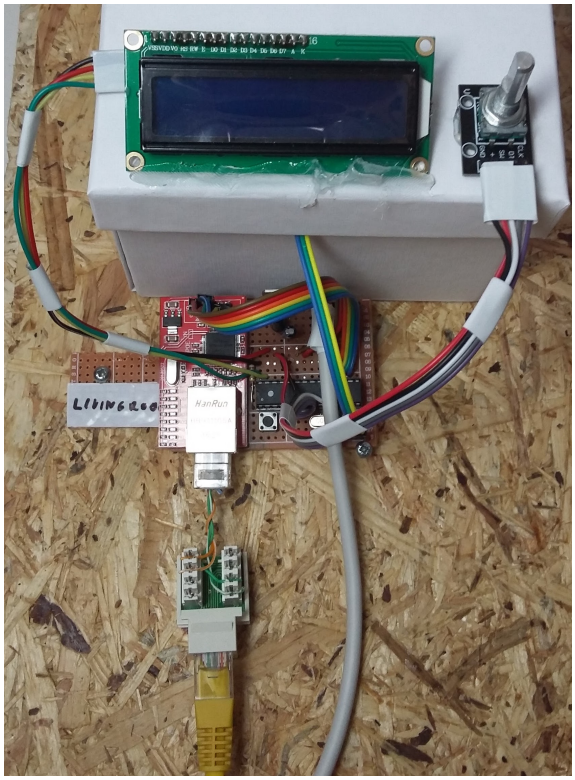
Obr. 30: ATMega328 doska s DS18B20, KY-040 a LCD

10.2.4 Pripojenie k lokálnej sieti - Ethernet

Ako je uvedené v predchádzajúcich kapitolách, zariadenie *thermostatSense* odosiela informácie o aktuálnej nameranej teplote a nastavenej požadovanej teplote do centrálnej jednotky. To je zabezpečené pomocou protokolu *TCP/IP* po lokálnej sieti *LAN*. Preto som pridal do zariadenia *Ethernet* modul, ktorý túto komunikáciu zabezpečí. Modul obsahuje konektor *RJ45* pre pripojenie Ethernet káblu, čip *W5100* (rovnako ako oficiálny *Ethernet shield* od spoločnosti Arduino) a *SPI* rozhranie, ktorým som ho pripojil k základnej doske. Softvér pre ovládanie tohto modulu je rovnaký ako pre oficiálny *Ethernet shield* a rovnako využíva oficiálnu knižnicu *Ethernet.h*. Softvér je popísaný v kapitole 8.3.2. Zapojenie pomocou zbernice *SPI* je na obrázku Obr.31. Obrázok Obr.31 je zároveň kompletnou schémou zariadenia *thermostatSense*



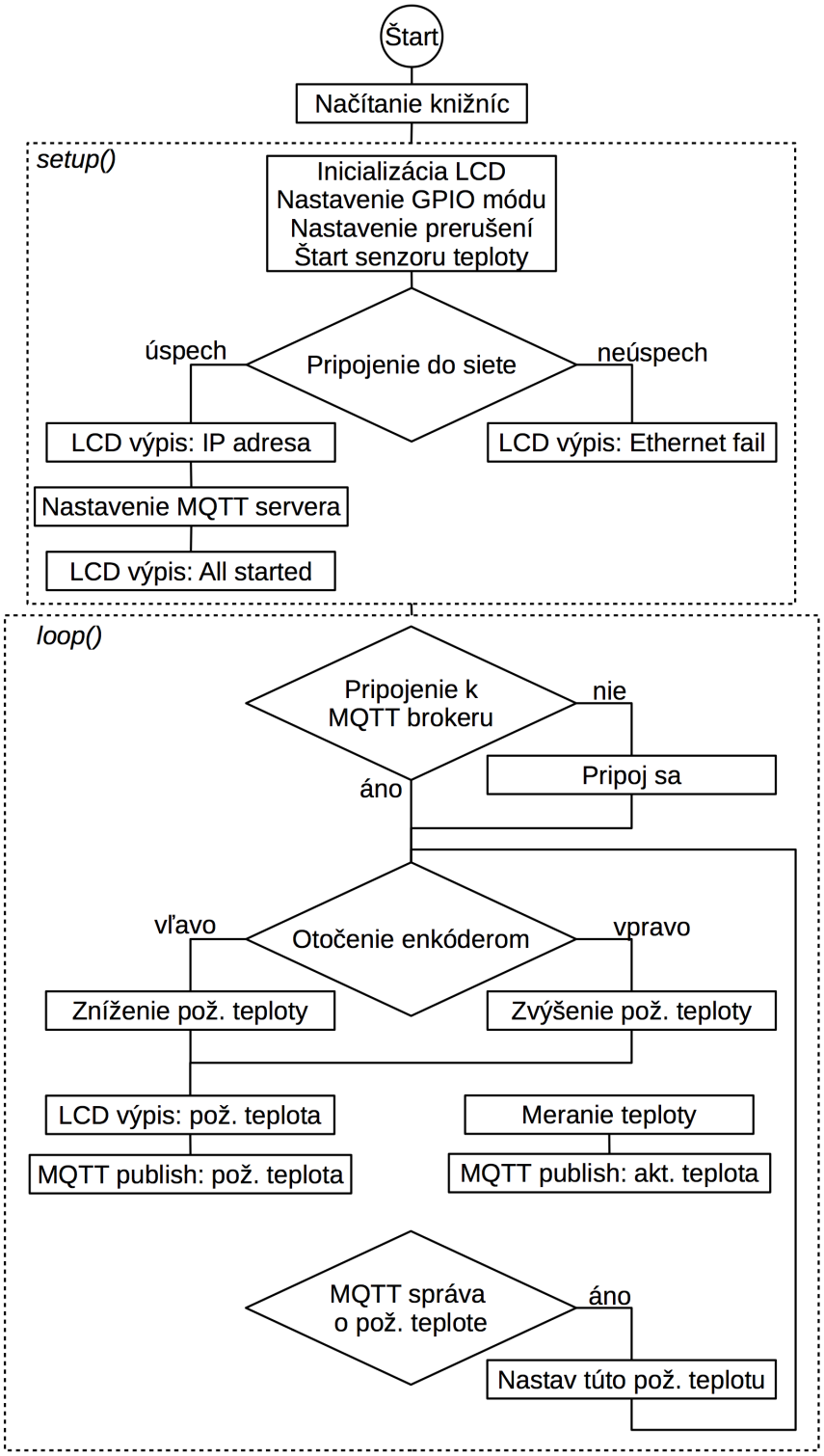
Obr. 31: ATMega328 doska s DS18B20, KY-040, LCD a Ethernet



Obr. 32: Fotografia zariadenia *thermostatSense*

Firmvér zariadenia *thermostatSense* je možné nájsť v prílohe A.

Pri testovaní a simulácii som použil papierovú škatuľu, ktorá simuluje miestnosť, v ktorej sa kúri pomocou halogénovej žiarovky. Na obrázku Obr.32 je to biela škatuľa hore. Na nej je pripevnený LCD displej, vedľa ktorého je rotačný enkóder na nastavovanie požadovanej teploty. Pod displejom je na obrázku Obr.32 vidieť spomínanú dosku pre mikrokontrolér ATMega328, ktorá je rozšírená práve o LCD, o rotačný enkóder a Ethernet modul. Pripojenie k Ethernetu je riešené dvoma konektormi RJ45 zapojenými za sebou. Je to z dôvodu prípravy na napájanie zariadenia pomocou *PoE - Power over Ethernet* (napájanie pomocou štruktúrovanej kábeláže, bez použitia externého zdroja napájania). Z dôvodu tejto prípravy je na dosku pre mikrokontrolér pridaný aj regulátor napätia LM7805.



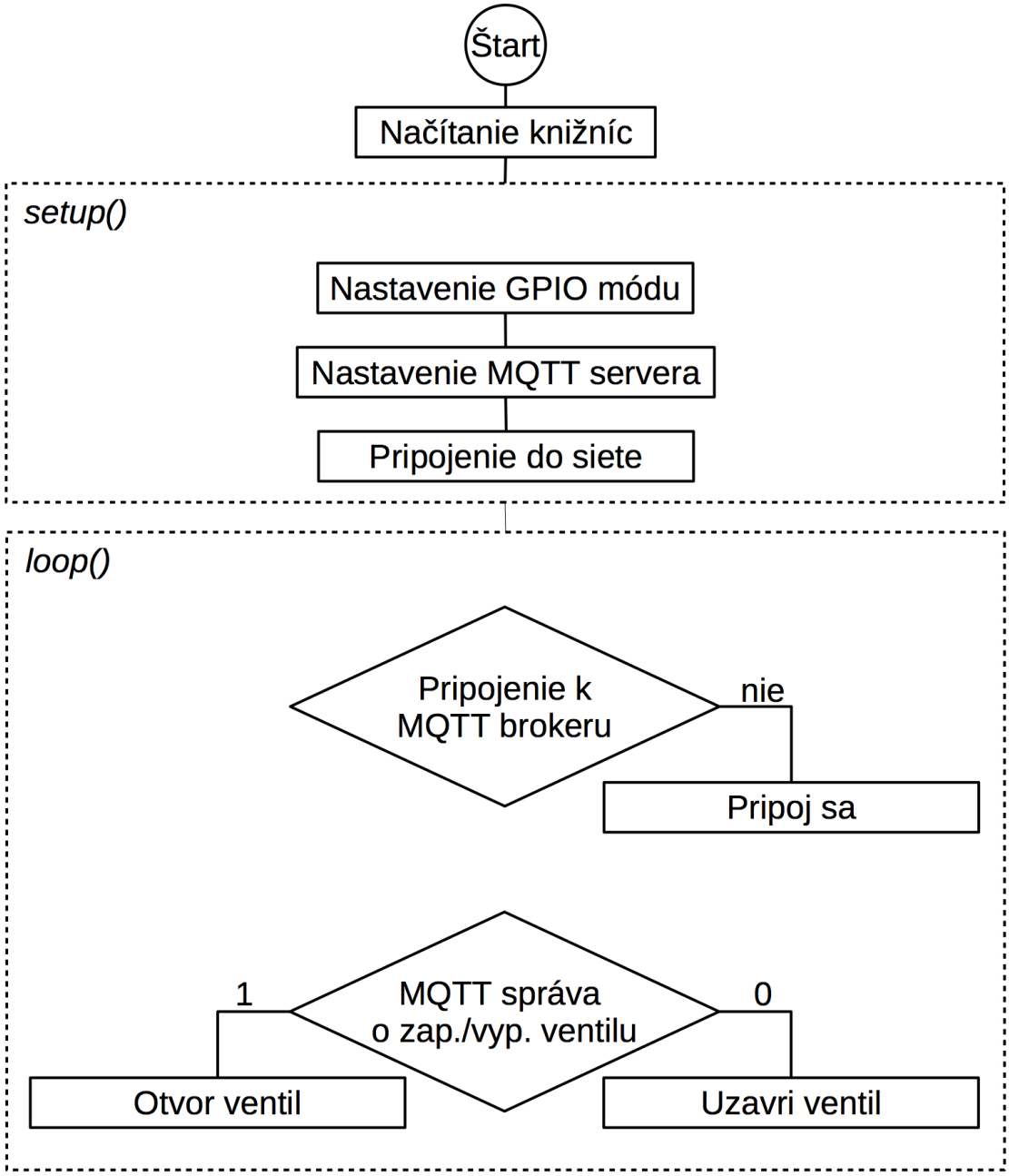
Obr. 33: Vývojový diagram - *thermostatSense*

10.3 Zariadene *thermostatValve*

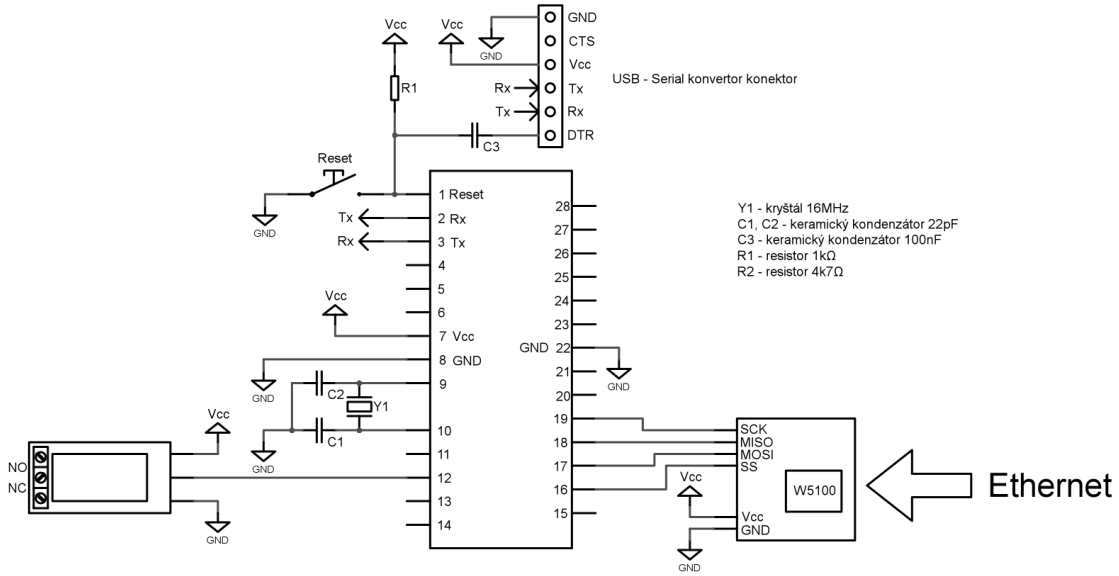
Druhou časťou systému je vybavovacia relé jednotka, ktorá spína kontakty zdroja tepla. V svojej práci simulujem vykurovacie telesá halogénovými žiarovkami, ktoré spínajú práve tieto relé. Základom je rovnaká doska ako bola popísaná v kapitole 10.1. Toto zariadenie však neobsahuje teplomer, LCD displej ani rotačný enkóder na nastavovanie teploty. So zariadením *thermostatSense* majú spoločnú len základnú dosku a Ethernet modul. Zariadenie *thermostatValve* obsahuje naviac relé modul, ktorým spína simuláciu kúrenia. Schéma zapojenia zariadenia *thermostatValve* je na obrázku Obr.35.

Zariadenie *thermostatValve* je prihlásené na odber správ, ktoré posiela riadiaca jednotka s predmetom *"66ADAE4A47D1/valve"*. Kde *66ADAE4A47D1* je adresa nadradeného zariadenia *thermostatSense*. Týmto spôsobom sú jednotky *thermostatSense* a *thermostatValve* prepojené. Správa od centrálnej jednotky je buď zapnúť alebo vypnúť zdroj tepla. Zariadenie *thermostatValve* má takisto svoju MAC adresu. Tá slúži na prihlásenie zariadenia do centrálnej riadiacej jednotky. Tieto adresy sú samozrejme jedinečné pre každé zariadenie. Tým pádom aj predmety správ sú jedinečné. Dvojice zariadení *thermostatSense* a *thermostatValve* sú definované vo firmvéri jednotlivých zariadení *thermostatSense*.

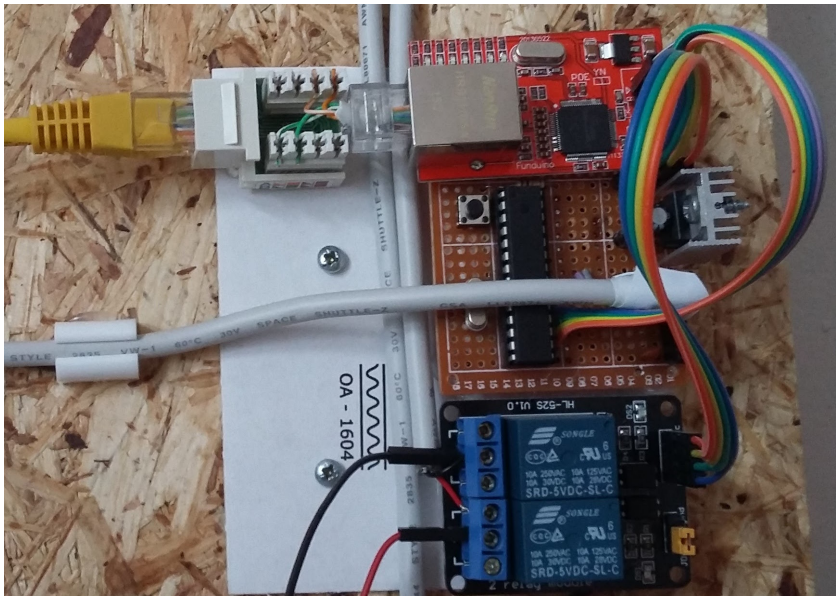
Firmvér zariadenia *thermostatValve* je možné nájsť v prílohe B.



Obr. 34: Vývojový diagram - *thermostatValve*



Obr. 35: Zariadenie thermostatValve



Obr. 36: Fotografia zariadenia *thermostatValve*

Zariadenie *thermostatValve* sa skladá tak isto zo základnej dosky popísanej v kapitole 10.1. Tá je rozšírená o pripojené relé, ktoré simuluje spínanie kúrenia. Spína kladný pól 12Voltového napájania so žiarovkou, ktorá simuluje kúrenie. Rovnako ako pri zariadení *thermostatSense* je doska mikrokontroléru rozšírená o Ethernet modul. Použil som tak isto RJ45 konektory ako adaptér pre PoE. Je tak isto použitý regulátor napätia LM7805.

10.4 Logika hlavného riadiaceho programu

V centrálnej jednotke beží spomínaný hlavný riadiaci program. Pripája sa k lokálnemu MQTT server, od ktorého získava informácie o všetkých zariadeniach v systéme. Ako prvé čaká program na prihlásenie sa jednotlivých zariadení *thermostatSense* a *thermostatValve*. Následne očakáva správy o aktuálnych teplotách a teplotách nastavených. Nastavené teploty získava prostredníctvom MQTT správ s predmetom *setTemp* (v prípade nastavenia požadovanej teploty priamo na zariadení *thermostatSense*) alebo *setTempWeb* (v prípade nastavenia požadovanej teploty prostredníctvom webovej aplikácie). Požadované teploty ďalej získava program z databáze, kde sú uložené všetky naučené a nastavené požadované teploty. To znamená, že každú hodinu sa program spýta databáze, aká je požadovaná teplota v danú hodinu. Následne si túto teplotu nastaví a informáciu pošle do zariadenia *thermostatSense*.

Samotné riadenie teploty vykonáva takisto hlavný program. Zariadenia *thermostatSense* merajú a odosielajú aktuálnu teplotu každé tri sekundy. Po prijatí tejto správy o teplote, program porovná túto odmeranú teplotu s teplotou požadovanou, a podľa toho rozhodne o zapnutí alebo vypnutí kúrenia v danej miestnosti. Následne odošle MQTT server (broker) správu o zapnutí resp. vypnutí kúrenia zariadeniu *thermostatValve* s predmetom *"66ADAE4A47E1/valve/1"*, kde *1* znamená zapnúť kúrenie (zopnúť relé) a *"66ADAE4A47E1/valve/1"*, kde *0* znamená vypnúť kúrenie (rozopnúť relé).

Schopnosť učiť sa režim užívateľa je rovnako zabezpečený v hlavnom programe. Akonáhle centrálna jednotka (hlavný program) obdrží informáciu o nastavení požadovanej teploty na niektorom zo zariadení *thermostatSense*, uloží túto informáciu do databáze s dňom, hodinou a adresou zariadenia pre ktoré bola teplota nastavená. Následne centrálna jednotka číta túto databázu ako bolo spomínané v predchádzajúcom odseku.

10.5 Webová aplikácia

Pre nastavovanie požadovanej teploty a sledovanie teploty aktuálne nameranej som vytvoril jednoduchú webovú aplikáciu pomocou HTML, PHP a JavaScriptu. Táto aplikácia beží a je uložená v riadiacej jednotke. Ako som spomínal v kapitole 7, moja simulácia obsahuje dve miestnosti, takže aplikácia vyzerá ako na obrázku Obr.37.

Obr. 37: Webová aplikácia

Prvé číslo vľavo (žltej farby) zobrazuje aktuálnu nameranú teplotu v miestnosti. Druhé číslo vpravo (zelenej farby) zobrazuje nastavenú požadovanú teplotu. Požadovanú teplotu je možné nastaviť priamo fyzicky za zariadení *thermostatSense*, ale aj pomocou tejto webovej aplikácie. Sú k tomu určené farebné ikony pod číslom zobrazujúcim nastavenú požadovanú hodnotu (červená ikona teplotu zvýši a modrá, naopak zníži). Logika je pre obe simulované izby rovnaká.

Ďalej sa v aplikácii nachádza nastavenie jednotlivých požadovaných teplôt na každú hodinu a každý deň v týždni. Táto tabuľa (obrázok Obr.38) slúži zároveň ako prehľad teplôt, ktoré sa termostat naučil - teplôt, ktoré boli nastavené v jednotlivé hodiny. Opäť je tabuľka rozdelená na dve časti, ktoré zodpovedajú simulovaným miestnostiam.

Monday:

0:00	1:00	2:00	3:00	4:00	5:00	6:00	7:00	8:00	9:00	10:00	11:00	12:00	13:00
18.00	18.00	18.00	18.00	18.00	18.00	18.00	22.00	22.00	22.00	22.00	22.00	22.00	22.00

Tuesday:

0:00	1:00	2:00	3:00	4:00	5:00	6:00	7:00	8:00	9:00	10:00	11:00	12:00	13:00
18.00	18.00	18.00	18.00	18.00	18.00	18.00	25.00	25.00	24.00	22.00	22.00	22.00	22.00

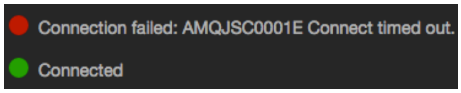
Wednesday:

0:00	1:00	2:00	3:00	4:00	5:00	6:00	7:00	8:00	9:00	10:00	11:00	12:00	13:00
18.00	18.00	18.00	21.00	18.00	18.00	18.00	25.00	25.00	24.00	22.00	25.00	22.00	22.00

Obr. 38: Nastavenie celého týždňa

10.5.1 Logika webovej aplikácie

Design aplikácie som naprogramoval pomocou HTML, PHP, CSS a frameworku Bootstrap. Aplikácia získava informácie o teplotách, ktoré následne zobrazuje od serveru *MQTT Broker* pomocou *MQTT* protokolu. O pripojenie k serveru sa stará javascript. Pomocou jednoduchého skriptu (v prílohe na CD *webapp/js/mymqtt.js*) sa aplikácia pripojí k *MQTT Brokeru* a prihlási sa na odber správ s daným predmetom. Pripojenie je indikované zeleným kruhom a hlásením *connected* pod nadpisom aplikácie. V prípade chyby pripojenia je kruh červený a aplikácia vypíše chybové hlásenie (viď obrázok Obr.39). Po úspešnom pripojení javascript príma správy o teplotách v jednotlivých miestnostiach a zobrazuje ich. Nastavenie požadovanej teploty prebieha takisto pomocou javascriptu. Po kliknutí na ikonu (zvýšiť alebo znížiť) je predchádzajúca nastavená teplota zvýšená (resp. znížená) o pól stupňa Celzia a následne odoslaná ako nová správa s predmetom nastavenej teploty pomocou webovej aplikácie - *SetTempWeb*. Týmto spôsobom sa dostane informácia o novej nastavenej teplote najprv do riadiacej jednotky a následne do zariadenia *thermostatSense* a tým je nastavená nová požadovaná teplota.



Obr. 39: Webová aplikácia

Užívateľovi je umožnené nastavovať požadovanú teplotu v jednotlivé hodiny jednotlivých dní v týždni. Pre túto funkciu slúži spomínaná tabuľka v druhej časti aplikácie (viď obrázok Obr.38). Tabuľku je potrebné vypĺňať po jednom dni. Každý deň je možné uložiť pomocou tlačidla na konci riadku pre každý deň. Tieto hodnoty sú uložené v databáze, ktorá je tak isto v riadiacej jednotke.

10.5.2 Ekonomický rozbor

Cieľom práce bolo nahradiť proprietárne riešenia od rôznych výrobcov otvoreným riešením, ktoré by mohlo byť aj prostriedkom pre spojenie rôznych systémov. Profesionálne riešenia termostatov (napríklad zo zariadení opísaných v kapitole 5) sa pohybuje medzi 250 - 300 USD viď tabuľka Tab. 1 (v prepočte 6000 - 7200 Českých korún). Takéto riešenie je však uzavreté. To znamená, že prípadné užívateľské požiadavky nad rámec naprogramovanej funkcionality je prakticky nemožné implementovať.

Práca popisuje riešenie, ktorého cena sa pohybuje do tisíc Českých korún (cena za popísané zariadenia v kapitole 10 a ich počet). V cene nie je zahrnutý vývoj, programátorské práce, testovací čas a podobne. V prípade väčšieho projektu, teda nasadenia väčšieho počtu komponentov, je možné rovnaké komponenty zaobstarať za približne dve tretiny ceny. V prípade hromadnej výroby, respektíve pri veľkom projekte, je možná zákazková výroba komponentov, čo by mohlo priniesť ešte väčšie úspory.

11 Záver

Cieľom práce bolo vytvoriť otvorený systém, a teda využiť čo najviac systémy open-source softvéru a takisto open-source hardvéru. Takýto open-source predstavuje protokol MQTT a platforma Arduino.

V blízkej budúcnosti je očakávaný nárast inteligentných zriadení, ktoré vyžadujú jednotný komunikačný štandard. Ten je použitý aj v prípade vlastného projektu. Tento projekt je pripravený aj na pripojenie takýchto zariadení a prakticky iba úpravou softvéru je pripravený pre riadenie akéhokoľvek zariadenia so štandardným rozhraním.

Vlastný projekt je základom pre prakticky ľubovolnú aplikáciu pre ovládanie, úpravou softvéru, doplnením prvkov a rozšírením na distribuovaný systém. Doplnením zabezpečenia komunikácie (prostredníctvom VPN, respektíve certifikátom) vznikne možnosť bezpečne riadiť a kontrolovať rôzne zariadenia (a rôzne objekty) cez Internet.

Na použitých prvkoch vlastného projektu a použitím rovnakej filozofie je možné postaviť IoT (Internet of Things) aplikácie rôznej úrovne. Principiálne bude vždy aplikácia pozostávať z riadiaceho člena, ovládacieho člena, konektora pre komunikáciu rôznych prvkov (MQTT), a viac či menej komfortného užívateľského rozhrania (v tomto prípade web rozhranie).

Prínos práce by mal byť hlavne v uvedomení si nastupujúceho trendu ovládania a riadenia aj tých najjednoduchších spotrebičov, čo by malo priniesť na jednej strane úsporu energie spotrebiteľov, bezpečnosť (napríklad doplnením o hlásenie hraničných hodnôt, automatické zabezpečovacie akcie - vypnutie elektrického prúdu, plynu, vody, atď.), na druhej strane vyváženú dodávku energií od výrobcov a distribútorov.

Plánované je postupné nasadenie v areáli kúpeľov vo Vysokých Tatrách, kde sa nachádza deväť objektov (každý s vlastným plynovým kotlom) v spolupráci so spoločnosťou, ktorá zabezpečuje pre tieto kúpele informačno-komunikačné služby. V každom objekte je 20 až 30 izieb a ďalších prevádzkových miestností. Rozpočet na tento projekt nie je určený, plánuje sa ale postupné nasadzovanie z nasledovným vyhodnocovaním efektivity.

Literatúra

[1] *Ako môžeme šetriť energiou?*. [online]. URL: http://www.futurenergia.org/ww/sk/pub/futurenergia/activity/save_energy.htm

[2] *Mechanické a digitální prostorové termostaty, možnosti úspor při vytápění domů*. [online]. URL: http://vytapeni.tzb-info.cz/mereni-a-regulace/6092-mechanicke-a-digitalni-prostorove-termostaty-moznosti-uspor-pri-vytapeni-domu

[3] *Zónové regulační systémy a jejich využití při úsporném efektivním vytápění*. [online]. URL: http://vytapeni.tzb-info.cz/mereni-a-regulace/6203-zonove-regulacni-systemy-a-jejich-vyuziti-pri-uspornem-efektivnim-vytapeni

[4] *Arduino*. [online]. URL: http://www.arduino.cc

[5] *Reading Rotary Encoders*. [online].
URL: http://playground.arduino.cc/Main/RotaryEncoders

[6] *SPI library*. [online]. URL: https://www.arduino.cc/en/Reference/SPI

[7] *Official I2C specification*. [online].
URL: http://www.nxp.com/documents/user_manual/UM10204.pdf

[8] BANKS Andrew, GUPTA Rahul. *MQTT OASIS Standard*. 2014-10-14. [online].
URL: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html

[9] KLAS Marcel. *Porovnání protokolů pro M2M komunikaci*. Brno 2014. 37 s. Bakalárska práca na Fakulte informatiky MUNI. Vedúci práce: RNDr.Daniel Tovarňák.

[10] *HiveMQ Blog*. [online]. URL:http://www.hivemq.com/blog/

[11] *Honeywell Lyric vs Nest vs Ecobee 3 Smart Thermostat*. [online]. URL: http://www.securitygem.com/honeywell-lyric-vs-nest-vs-ecobee-3/

Prílohy

A Firmware zariadenia *thermostatSense*

thermostatSense.ino:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Ethernet.h>
#include <PubSubClient.h>

#include "livingroom.h"
#include "config.h"
#include "functions.h"

void setup() {
  lcd.begin(16, 2); // initialize the lcd
  lcd.setCursor(0, 0);
  lcd.print(F("Thermostat-Sense"));

  pinMode(pinA, INPUT);
  pinMode(pinB, INPUT);
  pinMode(pinSW, INPUT);

  // attach interupt to pin pinA (CLK)
  attachInterrupt(digitalPinToInterrupt(pinA),
                  doEncoderA,
                  CHANGE);
  // attach interupt to pin pinB (DT)
  attachInterrupt(digitalPinToInterrupt(pinB),
                  doEncoderB,
                  CHANGE);

  sensors.begin();
  if (Ethernet.begin(mac) == 0) {
    lcd.setCursor(0, 1);
```

```
    lcd.print(F("Ethernet␣failed!"));
  } else {
    lcd.setCursor(0, 1);
    lcd.print(F("IP:␣"));
    lcd.print(Ethernet.localIP());
  }
  delay(1500);

  mqttClient.setServer(mqttBroker, mqttPort);
  mqttClient.setCallback(callback);

  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print(F("All␣started..."));
  delay(1500);
  lcd.clear();
}

void loop() {
  if (!mqttClient.connected()) {
    connectMqtt();
  }

  rotating = true; // reset the debouncer
  if (encoderPos >= MAX_POS) {
    encoderPos = MAX_POS;
    lastReportedPos = MAX_POS - 0.5;
  } else if (encoderPos <= MIN_POS) {
    encoderPos = MIN_POS;
    lastReportedPos = MIN_POS + 0.5;
  } else {
    if (lastReportedPos != encoderPos) {
      setTemp = encoderPos;
      lcd.setCursor(0, 0);
      lcd.print(F("Set:␣"));
      lcd.print(setTemp);
      lcd.print((char)223);
      lcd.print(F("C"));
      lastReportedPos = encoderPos;
```

```
      dtostrf(setTemp, 5, 2, setTempBuff);
      mqttClient.publish(setTempTopic, setTempBuff, true);
    }
  }

  currentMillis = millis(); // save milliseconds now
  // if the difference is more or equal to interval value:
  if (currentMillis - previousMillis >= interval) {
    //update previous milliseconds
    previousMillis = currentMillis;
    sensors.requestTemperatures(); // request temperature
    // store an actual temperature as float in variable.
    temperature = sensors.getTempCByIndex(0);
    // Index 0 because I have only one sensor on the bus.
    lcd.setCursor(0, 1);
    lcd.print(F("Act: "));
    lcd.print(temperature);
    lcd.print((char)223);
    lcd.print(F("C"));

    // creates string from float
    // to pass to the publish function
    dtostrf(temperature, 5, 2, temperatureBuff);
    // publishes an actual temperature to broker
    mqttClient.publish(actTempTopic,temperatureBuff,true);
  }
  mqttClient.loop();

}
```

config.h:

```
// ENCODER
enum PinAssignments {
  pinA = 3,   // DT pin
  pinB = 2,   // CLK pin
  pinSW = 5    // SW pin
};
```

```
// Position of the encoder is stored here
volatile float encoderPos = 22.0;
// change management
float lastReportedPos = 22.5;
// debounce management
static boolean rotating = false;

// interrupt service routine variables
boolean A_set = false;
boolean B_set = false;

#define MAX_POS 30.0
#define MIN_POS 15.0


//TEMPERATURE
#define ONE_WIRE_BUS 7

// Setup a oneWire instance to communicate
// with any OneWire devices:
OneWire oneWire(ONE_WIRE_BUS);
// Pass the oneWire reference
// to Dallas Temperature:
DallasTemperature sensors(&oneWire);

// measured temperature
float temperature = 0.0
unsigned long previousMillis = 0;
const int interval = 3000;
unsigned long currentMillis = 0;

float setTemp = encoderPos;
char tempBuff[10];
char setTempBuff[10];
char temperatureBuff[10];

// LCD
// Set the LCD I2C address
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7,3,POSITIVE);
```

```
// Ethernet
EthernetClient ethClient;

// MQTT
PubSubClient mqttClient(ethClient);
IPAddress mqttBroker(192, 168, 0, 13);
int mqttPort = 1883;
```

functions.h:

```
// Interrupt handling on pinA change state
void doEncoderA() {
  if (rotating) delay (10);
  if (digitalRead(pinA) != A_set) {
    A_set = !A_set;
    if (A_set && !B_set) encoderPos += 0.5;
    rotating = false;
  }
}
// Interrupt handling on pinB change state
void doEncoderB() {
  if (rotating) delay (10);
  if (digitalRead(pinB) != B_set) {
    B_set = !B_set;
    if (B_set && !A_set) encoderPos -= 0.5;
    rotating = false;
  }
}
// MQTT
void callback(char* topic,
              byte* payload,
              unsigned int length) {
  for (int i = 0; i < length; i++) {
    if (i==0) setTemp = 10 * ((int)payload[i] - 48);
    if (i==1) setTemp = setTemp + ((int)payload[i] - 48);
    if (i==3) setTemp = setTemp+0.1*((int)payload[i]-48);
    if (i==4) setTemp = setTemp+0.01*((int)payload[i]-48);
  }
  encoderPos = setTemp;
```

```
}

void connectMqtt() {
  while (!mqttClient.connected()) {
    lcd.setCursor(0, 1);
    lcd.print(F("MQTT␣connect...␣"));
    if (mqttClient.connect(deviceID)) {
      lcd.setCursor(0, 1);
      lcd.print(F("Broker␣connected"));
      mqttClient.subscribe(setTempWebTopic);
      mqttClient.publish(readyTopic,deviceID);
      lcd.clear();
    } else {
      lcd.setCursor(0, 1);
      lcd.print(F("failed!␣rc="));
      lcd.print(mqttClient.state());
      lcd.print(F("␣␣␣␣"));
      delay(500);
      lcd.setCursor(0, 1);
      lcd.print(F("Retry␣in␣3␣sec.␣"));
      delay(1000);
      lcd.setCursor(0, 1);
      lcd.print(F("Retry␣in␣2␣sec.␣"));
      delay(1000);
      lcd.setCursor(0, 1);
      lcd.print(F("Retry␣in␣1␣sec.␣"));
      delay(1000);
      lcd.clear();
    }
  }
}
```

Pre každú izbu a zároveň dvojicu zariadení, existuje konfiguračný súbor. Príklad konkrétnej izby:

```
byte mac[] = { 0x66, 0xAD, 0xAE, 0x4A, 0x47, 0xD2 };
char deviceID[] = "66ADAE4A47D1";
char deviceName[] = "sense_livingroom";
char actTempTopic[] = "66ADAE4A47D2/sense/actTemp";
char setTempTopic[] = "66ADAE4A47D2/sense/setTemp";
```

```
char setTempWebTopic[] = "66ADAE4A47D2/sense/setTempWeb";
char readyTopic[] = "66ADAE4A47D2/sense/ready";
```

B Firmware zariadenia *thermostatValve*

thermostatValve.ino:

```
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>

#include "livingroom.h"

char ventilBuff[4];
int heatPin = 6;

IPAddress server(192, 168, 0, 13);

void callback(char* topic,
              byte* payload,
              unsigned int length) {
  int i = 0;
  for(i=0; i<length; i++) {
    ventilBuff[i] = payload[i];
  }
  ventilBuff[i] = '\0';

  String ventilString = String(ventilBuff);

  if (ventilString == "1") {
    digitalWrite(heatPin,LOW);
  } else if (ventilString == "0") {
    digitalWrite(heatPin,HIGH);
  }
}

EthernetClient ethClient;
PubSubClient client(ethClient);

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
```

```
    Serial.print("Attempting␣MQTT␣connection...");
    // Attempt to connect
    if (client.connect(deviceName)) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish(readyTopic,"ready");
      // ... and resubscribe
      client.subscribe(ventilTopic);
    } else {
      Serial.print("failed,␣rc=");
      Serial.print(client.state());
      Serial.println("␣try␣again␣in␣5␣seconds");
      // Wait 5 seconds before retrying
      delay(3000);
    }
  }
}

void setup() {
  pinMode(heatPin,OUTPUT);

  Serial.begin(9600);

  client.setServer(server, 1883);
  client.setCallback(callback);

  Ethernet.begin(mac);
  // Allow the hardware to sort itself out
  delay(1500);
}

void loop()
{
  if (!client.connected()) {
    reconnect();
  }

  client.loop();
}
```

Pre každú izbu a zároveň dvojicu zariadení, existuje konfiguračný súbor. Príklad konkrétnej izby:

```
byte mac[] = { 0x66, 0xAD, 0xAE, 0x4A, 0x47, 0xE1 };
char deviceID[] = "66ADAE4A47E1";
char deviceName[] = "ventil_livingroom";
char ventilTopic[] = "66ADAE4A47D1/ventil";
char readyTopic[] = "66ADAE4A47E1/ventil/ready";
```

C Softvér pre teplomer DS18B20

```
// Potrebne kniznice
#include <OneWire.h>
#include <DallasTemperature.h>

// Datovy pin senzoru je pripojeny na port 5 Arduina
#define ONE_WIRE_BUS 5

// Instancia, pre komunikaciu s 1-Wire zariadeniami
OneWire oneWire(ONE_WIRE_BUS);

// Posunutie tejto instancie kniznici DallasTemperature
DallasTemperature sensors(&oneWire);

// Adresa senzoru
DeviceAddress teplomer = {
    0x28, 0xFF, 0xE3, 0xD4, 0x67, 0x14, 0x03, 0x39
    };

// funcia setup() prebehne len raz
void setup() {
    //Start senzoru
    sensors.begin();

    // Nastavenie rozlisenia senzoru. 9-12bit
    sensors.setResolution(teplomer, 12);

    // Spustenie seriovej komunikacie s baudrate=9600
    Serial.begin(9600);
}

// Hlavny cyklus
void loop() {
    // Vyziadanie teploty
    sensors.requestTemperatures();

    // Precitanie teploty v stupnoch Celzia
    // a priradenie do premennej
```

```
    float teplota = sensors.getTempC(teplomer);

    // Vypis teploty na terminalu - Serial Monitor
    Serial.print("Teplota:");
    Serial.println(teplota);

    // Zdrzanie 3 sekundy
    delay(3000);
}
```

D Softvér pre rotačný enkóder KY-040

```
enum PinAssignments {
    pinA = 2, // DT pin
    pinB = 3, // CLK pin
    switchPin = 4 // SW pin
};
// vychodiskova pozicia nastavena na 0
volatile  int encoderPos = 0;
// vychodiskova predchadzajuca pozicia
int lastReportedPos = 1;
// ochrana proti zakmitom
static boolean rotating = false;

// premenne pre ISR (interrupt service routine)
boolean A_set = false;
boolean B_set = false;

void setup() {
    pinMode(pinA, INPUT_PULLUP); // nastavenie pinov
    pinMode(pinB, INPUT_PULLUP);
    pinMode(switchPin, INPUT_PULLUP);

    // prerusenia
    attachInterrupt(digitalPinToInterrupt(pinA),
                    doEncoderA,
                    CHANGE);
    attachInterrupt(digitalPinToInterrupt(pinB),
```

```
                    doEncoderB,
                    CHANGE);

    Serial.begin(9600); // spustenie seriovej komunikacie
}
void loop() {
    rotating = true;  // reset pre ochranu proti zakmitom
    if (encoderPos == -1 ){
        encoderPos = 0;
        lastReportedPos = 1;
    } else {
        if (lastReportedPos != encoderPos) {
            Serial.print("Poloha: ");
            Serial.println(encoderPos, DEC);
            Serial.println(lastReportedPos);
            lastReportedPos = encoderPos;
        }
    }
    // po stlaceni hriadele - reset na nulu
    if (digitalRead(switchPin) == LOW )  {
        encoderPos = 0;
        lastReportedPos = 1;
        delay (10);
    }
}
// Prerusenie pri zmene na pine A
void doEncoderA(){
    // mala prestavka kvoli zakmitom
        if ( rotating ) delay (1);

        if( digitalRead(pinA) != A_set ) {
                A_set = !A_set;
                // posun sa o jednu poziciu
                if ( A_set && !B_set ) encoderPos += 1;
                rotating = false;
        }
}
// Prerusenie pri zmene na pine B
// (to iste ako pre A ale na opacnu stranu)
```

```
void doEncoderB(){
    if ( rotating ) delay (1);
    if( digitalRead(pinB) != B_set ) {
        B_set = !B_set;
        if( B_set && !A_set ) encoderPos -= 1;
        rotating = false;
    }
}
```