VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

AUTONOMNÍ ŘÍDÍCÍ SYSTÉM PRO VYTÁPĚNÍ KOMPOZITNÍCH MATERIÁLŮ

AUTONOMOUS CONTROL SYSTEM FOR COMPOSITE MATERIALS HEATING

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Adam Lapčák

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Pavel Šteffan, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Mikroelektronika**

Ústav mikroelektroniky

Student: Bc. Adam Lapčák ***ID:*** 220881

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Autonomní řídící systém pro vytápění kompozitních materiálů

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte autonomní řídící systém využitelný pro vytápění dílců z kompozitních materiálů. Navrhněte komunikační modul schopný rozšířit autonomní řídicí systém o vzdálené nastavení parametrů vytápění a monitorování teplot kompozitních dílců. Zařízení navrhněte pro použití v sítích GSM.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce.

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 21.5.2024

Vedoucí práce: doc. Ing. Pavel Šteffan, Ph.D.

doc. Ing. Lukáš Fujcik, Ph.D.
předseda rady studijního programu

Abstrakt

Tato práce se zabývá návrhem autonomního systému určeného pro vytápění ploch z vodivého kompozitního materiálu. Návrh spočívá ve vytvoření chytré elektroniky pro nasazení do rozvaděčů a možností ovládat výkonové spínací prvky. Důraz je také kladen na možnost vzdáleného ovládání systému prostřednictvím internetu. Pro jednoduché nasazení na libovolné lokaci bude autonomní systém využívat mobilní síť. Ovládání systému pak zajistí vlastní řídicí HTTP API server společně s vlastní uživatelskou aplikací.

Klíčová slova

Internet věcí, HTTP, API, mobilní síť, GSM, GPRS, SIM

Abstract

This thesis focuses on designing an autonomous system intended for heating surfaces made of conductive composite material. The design involves smart electronics design for integration into distribution boards and the ability to control power switching elements. Emphasis is also placed on the possibility of remote system control via the internet network. For an easy deployment in any location, the autonomous system will use a mobile network. System control will be realized with a dedicated HTTP API server along with a custom user application.

Keywords

Internet of things, HTTP, API, mobile network, GSM, GPRS, SIM

Bibliografická citace

LAPČÁK, Adam. *Autonomní řídící systém pro vytápění kompozitních materiálů* [online]. Brno, 2024 [cit. 2024-05-21]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/159938. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mikroelektroniky. Vedoucí práce Pavel Šteffan.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	*Adam Lapčák*
VUT ID studenta:	*220881*
Typ práce:	*Diplomová práce*
Akademický rok:	*2023/24*
Téma závěrečné práce:	*Autonomní řídící systém pro vytápění kompozitních materiálů*

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 21. května 2024 ----------

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Pavlu Šteffanovi, Ph.D. za časté konzultace a aktivní vedení práce. Dále děkuji Ing. Jiřímu Pospišilíkovi, Ph.D. za zhodnocení textu práce a Bc. Ondřeji Pokornému a Ing. Vladimíru Levkovi, Ph.D za praktické rady ohledně realizace zařízení.

V Brně dne: 21. května 2024 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

V dnešní době se společnost stále více zaměřuje na implementaci chytrých technologií, které zvyšují kvalitu každodenního života. Implementace není patrná pouze ve spotřební sféře, ale také v městské infrastruktuře, kde nalezne uplatnění například v autonomním vytápění pozemních komunikací či chodníků. Právě na tuto problematiku se tato práce zaměřuje, konkrétně se zabývá návrhem autonomního systému vytápění s monitoringem teplot a proudového odběru pro plochy vytvořené z vodivého kompozitního materiálu.

Důležitým požadavkem na autonomní systém je také jeho vzdálené ovládání prostřednictvím internetu. Vzhledem k možnosti jeho nasazení na libovolné lokaci bude autonomní systém rozšířen o SIM modul a možnost připojení k mobilnímu internetu.

Před samotným návrhem práce objasňuje strukturu IoT sítě a také uvádí a vysvětluje použité komponenty a technologie k realizaci systému. Mezi důležité využité komponenty patří senzor proudu založený na Hallovém jevu, termistorový snímač teploty NTC a mikrokontroler. Z použitých periferií mikrokontroleru se využívá GPIO, AD převodníku, čítače/časovače a komunikačních sběrnic UART, CAN a SPI. Zároveň práce před návrhem systému objasňuje mobilní síť, tvorbu síťových služeb a tvorbu uživatelských aplikací.

V další části práce je vysvětlen princip funkce navrženého systému, který má za úkol udržovat požadovanou teplotu vytápěné plochy na základě zpětné vazby o teplotě vytápěné plochy. Zároveň bude systém vybaven módem vytápění, který umožní vytápět materiál v případě, kdy by se na povrchu mohla tvořit vrstva náledí. K vyhodnocení, zda se vrstva náledí může vytvořit, bude systém využívat předpověď počasí.

Samotný návrh autonomního systému zahrnuje návrh elektronických desek, které lze vložit do modulové krabičky určené pro rozvaděče s DIN lištou. Nedílnou součástí návrhu je také programové vybavení mikrokontrolerů a příklad zapojení systému v rozvaděči.

Vzdálené ovládání a monitoring autonomního sytému bude zajišťovat vlastní HTTP API server. K serveru se bude autonomní systém připojovat prostřednictvím mobilního internetu a předdefinovanými API dotazy. Zároveň bude server plnit funkci zálohy historie naměřených dat.

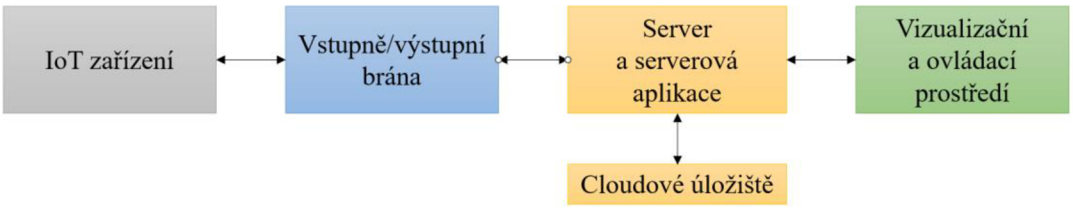
Server bude také uživateli umožňovat získat přehled naměřených dat z autonomního systému a nastavit parametry vytápění prostřednictvím internetu. Pro zjednodušení obsluhy serveru a zpřehlednění naměřených dat bude vytvořena uživatelská aplikace.

1. Úvod do internetu věcí

Internet věcí popisuje síť fyzických zařízení, které jsou určeny pro vzdálený sběr a sdílení dat. Dnes se internet věcí využívá například v chytrých domácnostech (chytré termostaty, osvětlení, rolety), v průmyslu (sběr dat ze strojů pro optimalizaci výroby), v nákladní dopravě (sledování zásilek) a také v tzv. chytrých městech. Chytrá města za pomocí internetu věcí zvyšují bezpečnost a kvalitu života občanů, řídí efektivně dopravu aj. [1] [2]

Důležitými prvky internetu věcí jsou [1] [3] [4]:

- IoT zařízení: hardware, který má za úkol interagovat s okolím na základě řídících digitálních dat, nebo naopak fyzikální jevy v okolí a jejich změny převádět na digitální signál. Zařízení proto obsahuje senzory s digitálním výstupem a akční členy.
- Vstupně/výstupní brána: Má za úkol sběr dat ze senzorů a jejich přenos do serverové aplikace. Zároveň na základě příkazů ze serverové aplikace předávají parametry ovládání pro akční členy do IoT zařízení. Pro přenos dat se serverem jsou vybaveny možností připojení k internetu nebo k jiné síti.
- Serverové systémy a aplikace: Starají se o převod dat mezi vstupně/výstupní bránou a ovládacím či vizualizačním prostředím. Jsou spuštěny na pozadí a uživatel k nim nemá přímý přístup. Data mohou také šifrovat z důvodu bezpečnosti nebo komprimovat z důvodu úspory dat. Příkladem jsou API servery nebo MQTT broker.
- Cloudové úložiště: Ne vždy potřebná, ale výhodná část internetu věcí. Slouží k ukládání velkého množství dat v síti pro retrospektivní analýzu bez nutnosti použití lokálního úložiště.
- Vizualizační aplikace a ovládací prostředí: Aplikace určené pro koncové zákazníky. Slouží k ovládání IoT zařízení a přehledu sledovaných dat. Obsahují vizuální prvky pro přehlednou orientaci, např. tlačítka, přepínače, signalizační prvky, seznamy nebo grafy.



Obrázek 1.1 Blokové schéma systému založené na IoT síti (překresleno z [3])

Před vytvořením IoT sítě je potřeba zvážit její kybernetickou bezpečnost, konektivitu zařízení, datovou náročnost a finanční náklady z pohledu pořízení hardwaru, provozu systému a licenčních poplatků. [1]

2. POPIS FUNKČNÍCH BLOKŮ INTERNETU VĚCÍ A PŘEHLED VYUŽITÝCH TECHNOLOGIÍ PRO NÁVRH

V této kapitole je uveden popis jednotlivých bloků IoT struktury, které jsou uvedeny na obrázku 1.1. Bude zde uvedeno, čím mohou být bloky představovány, a také přehled klíčových technologií, které byly využity pro návrh a realizaci autonomního sytému s jeho vzdáleným ovládáním prostřednictvím internetu.

2.1 IoT zařízení a vstupně/výstupní brána

První částí IoT systému je IoT zařízení se vstupně/výstupní bránou. Tato část představuje systémy, které interagují s okolním světem a zjištěná data předávají prostřednictvím lokální nebo internetové sítě.

IoT zařízení je obsahuje senzory a akční členy. Obsahuje také mikrokontroler s firmwarem, který přebírá data ze senzorů a řídí akční členy na základě dat ze vstupně/výstupní brány.

Vstupně/výstupní brána má za úkol sběr dat z jednoho nebo více IoT zařízení a předávání informací o jeho řízení. Komunikace mezi IoT zařízením a vstupně/výstupní bránou může být realizována bezdrátově i drátově. Mezi bezdrátové spojení patří např. Bluetooth nebo Zigbee, pro drátové spojení se využívají např. sériové sběrnice. Brána má zároveň za úkol předávat informace o IoT zařízení do lokální či internetové sítě. Obsahuje tedy možnost připojení k síti, například prostřednictvím LAN portu, Wi-Fi, LoRa nebo GSM modulu.

Často může být vstupně/výstupní brána implementována v IoT zařízení. Jedná se např. o chytré prvky domácnosti, které se připojují k její lokální síti.

Pro autonomní systém byla navržena vstupně/výstupní brána označována jako komunikační modul. Modul obsahuje mikrokontrolér, který využívá komunikace UART pro ovládání SIM modulu, který zajišťuje propojení komunikačního modulu s internetem. Zároveň je v komunikačním modulu využita sběrnice SPI pro připojení OLED displeje a microSD karty.

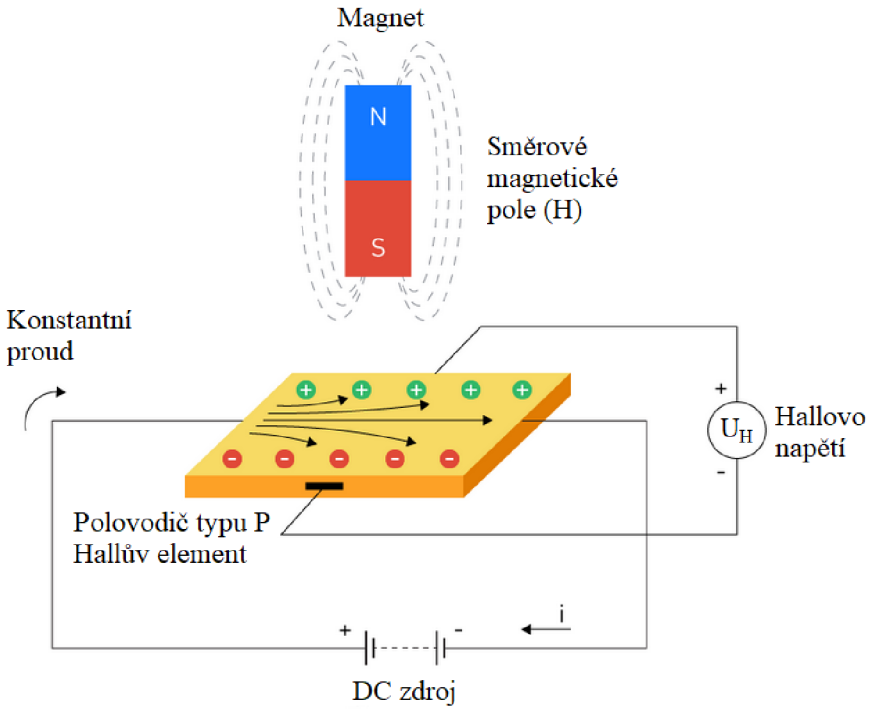
Dále bylo pro autonomní systém navrženo IoT zařízení označené jako výkonový modul. Bude mít za úkol ovládat spínací prvky (SSR relé a stykač) pomocí GPIO mikrokontroleru. Zároveň bude měřit spotřebu proudu pomocí Hallova snímače a teplotu vytápěných ploch pomocí NTC termistorů. Vyhodnocení analogových dat bude zajišťovat AD převodník mikrokontroleru.

Pro komunikaci mezi moduly je využita sériová sběrnice CAN.

V následujících podkapitolách bude uveden podrobně princip využitých snímačů, periferií mikrokontroléru, komunikačních sběrnic a technologie GSM pro vytvoření komunikačního a výkonového modulu.

2.1.1 Snímače proudu založené na Hallovém jevu

Hallův jev je zobrazen na obrázku 2.1. Při průtoku proudu tenkou destičkou tvořenou vodičem nebo polovodičem (nejčastěji polovodič typu P) a působení magnetického pole na destičku vzniká vlivem stáčení dráhy volných nosičů náboje na okrajích destičky napětí.[5]



Obrázek 2.1 Princip Hallova jevu (převzato z [5])

Vzniklé napětí na okrajích destičky (Hallův element) se nazývá Hallovo a je dáno vztahem [6]:

$$U_H = R_H \cdot \left(\frac{I}{h} \cdot B\right) \; [V; \frac{m^3}{C}, A, mm, T] \,, \tag{2.1}$$

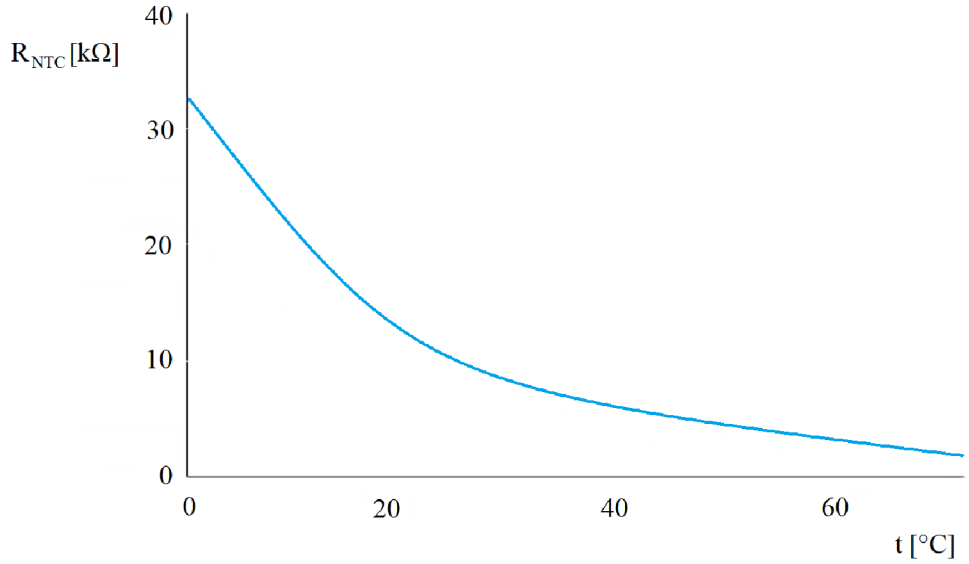
kde U_H označuje velikost Hallova napětí, R_H koeficient Hallova jevu, I protékající proud Hallovým elementem, h tloušťku Hallova elementu a B intenzitu směrového magnetického pole.

V případě lineárního obousměrného snímače proudu ACS758LCB-050B od spolčenosti Allegro Microsystems, který byl v práci využit, generuje vstupní proud na kontaktech *IP+* a *IP-* magnetické pole, jehož magnetická indukce je měřena Hallovým snímačem a převáděna na napěťovou hodnotu. Výstupní napětí tak vykazuje lineární závislost na vstupním proudu s citlivostí 40 mV/A. Výstupní napětí má v případě obousměrného snímače nastavený napěťový posun o $U_{CC}/2$, kde U_{CC} je napájecí napětí snímače, aby bylo možné snímat kladnou i zápornou polaritu proudu. [7]

2.1.2 NTC Termistory

Termistor je elektronická součástka, jejíž elektrický odpor je závislý na teplotě. Termistory se děli na dva typy, termistory se záporným teplotním koeficientem (NTC) a s kladným teplotním koeficientem (PTC). Teplotní koeficient vyjadřuje, jak se mění elektrický odpor v závislosti na zvyšující se teplotě, tedy v případě záporného teplotního koeficientu elektrický odpor klesá, v případně kladného se elektrický odpor zvyšuje. [8]

Pro měření teploty je častěji využíván NTC termistor. Teplotní závislost elektrického odporu je exponenciální, příklad závislosti je uveden na obrázku 2.2.



Obrázek 2.2 Příklad teplotní závislosti odporu NTC termistoru (převzato z [9])

V praxi se ovšem místo teplotní závislosti elektrického odporu využívá parametr β. Tento parametr linearizuje průběh elektrického odporu ve vymezené oblasti teplot (např. 25 až 85 °C) a zjednodušuje přepočet elektrického odporu na snímanou teplotu. Dalšími důležitými parametry u NTC termistorů je typická hodnota elektrického odporu při referenční teplotě (nejčastěji 25 °C) a rozsah teplot, pro které lze NTC termistor použít. [9]

Parametr β je dán vztahem [9]:

$$\beta = \frac{\ln\left(\frac{R_1}{R_2}\right)}{\left(\frac{1}{T_1}-\frac{1}{T_2}\right)}\ [K;\Omega,\Omega,K,K]\ , \tag{2.2}$$

kde R_1 udává odpor termistoru při nejnižší teplotě rozsahu T_1 a R_2 hodnotu odporu termistoru při nejvyšší teplotě rozsahu T_2.

Z rovnice (2.2) lze odvodit vztah pro výpočet naměřené teploty T NTC termistorem (rovnice (2.3)), případně elektrický odpor R při dané teplotě (rovnice (2.4)):

$$T = \frac{1}{\frac{\ln\left(\frac{R}{R_0}\right)}{\beta} - \frac{1}{T_0}} \; [K; \Omega, \Omega, K, K] \,, \tag{2.3}$$

$$R = R_0 \cdot e^{\beta \cdot \left(\frac{1}{T} - \frac{1}{T_0}\right)} \; [\Omega; \Omega, K, K, K] \,, \tag{2.4}$$

Pozn. Symbol R_0 udává typickou hodnotu elektrického odporu danou výrobcem při referenční teplotě T_0.

2.1.3 Mikrokontrolér

Mikrokontroler je integrovaný obvod, který obsahuje mikroprocesor, paměť a periferní zařízení v jednom čipu. Jeho hlavní funkci je vykonávat program, pomocí kterého řídí připojená elektronická zařízení. Program je reprezentován jako programový kód např. v jazycích C, C++ nebo Python. V práci byl využit mikrokontroler ATMega32M1, proto budou funkce mikrokontroleru a využité periferie popsány podle katalogového listu výrobce. [10][11]

Mikroprocesor je část mikrokontroleru, která vykonává instrukce programu. Skládá se z [10]:

- Aritmeticko-logické jednotky (ALU) – Provádí aritmetické a logické výpočty
- Pracovních registrů – Malá rychlá paměťová místa, které slouží k ukládaní mezivýsledků během vykonávání instrukce
- Programového čítače (Program Counter) – Udržuje adresu následující instrukce
- Ukazatele zásobníku (Stack Pointer) – Odkazuje na zásobník v paměti. Do zásobníku se ukládají dočasná data a návratové adresy při vykonávání podprogramů.
- Obvod hodinového signálu – Generuje hodinový signál, který určuje rychlost provádění operací
- Obvody přerušení – Umožňují přerušení běhu běžného programu a okamžitě zpracují prioritní událost

Mezi paměti, které jsou v mikrokontroleru ATMega32M1, patří [11]:

- Flash paměť – Slouží k uložení kódu programu. Je nevolatilní, tedy uchovává data i v případě vypnutí napájení
- SRAM paměť – Slouží k uložení proměnných a dočasných dat. Je volatilní, tedy data se po odpojení napájení ztratí
- EEPROM paměť – Nevolatilní paměť, slouží pro uživatelské a konfigurační data.

Periferie v mikrokontroleru umožňují připojení externích zařízení k mikrokontroleru nebo rozšiřují jeho funkce. Mezi využité periferie mikrokontroleru v této práci patří:

- GPIO
- Analogově digitální převodník
- Čítač/časovač
- Sběrnice CAN
- Sběrnice SPI
- Sběrnice UART

Využité periferie mikrokontroleru budou podrobně popsány v následujících kapitolách.

2.1.4 GPIO

GPIO (general purpose input/output) je označení pro piny mikrokontroleru, které se využívají jako digitální vstupy a výstupy. V mikrokontroleru ATMega32M1 jsou tyto piny rozděleny do skupin označované jako porty, a to konkrétně porty s názvy *B*, *C*, *D* (každý obsahuje 8 pinů) a *E* (obsahuje 3 piny). Ke každému portu se pojí 3 registry (x označuje název portu) [11]:

- Data direction register (*DDRx*) – Registr nastavuje směr dat jednotlivého pinu, tedy zda se pin chová jako digitální vstup nebo výstup
- Data register (*PORTx*) – Zapisuje digitální úroveň na výstupní piny
- Input pins register (*PINx*) – Uchovává stav na vstupním pinu

GPIO jsou v práci využity pro ovládání výkonových spínacích prvků a volbu periferií na sběrnici SPI.

2.1.5 Analogově digitální převodník

Analogově-digitální převodník (zkráceně AD převodník) je periferie, která převádí napětí vstupního analogového signálu na digitální hodnotu. Mikrokontroler ATMega32M1 disponuje převodníkem typu SAR s rozlišením 10 bitů, tedy digitální hodnota je vyjádřena v 10bitovém čísle. Převodník dokáže převádět napětí vztažené proti zemi *GND* (jedno-koncové připojení), nebo rozdíl napětí na diferenciálním vstupu. [11]

Pro výslednou hodnotu vrácenou AD převodníkem při převodu v případě využití jedno-koncového připojení platí [11]:

$$ADC = \frac{U_{in} \cdot 1024}{U_{ref}} \; [-; V, V] \,, \tag{2.5}$$

kde U_{in} označuje hodnotu napětí vstupního signálu a U_{ref} hodnotu napětí reference.

V případě využití diferenciálního vstupu platí pro výslednou hodnotu AD převodníku vztah [11]:

$$ADC = \frac{(U_p - U_n) \cdot GAIN \cdot 512}{U_{ref}} \; [-; V, V, -, V] \,, \tag{2.6}$$

kde $(U_p - U_n)$ označuje hodnotu rozdílového napětí vstupního diferenčního signálu, *GAIN* zvolené zesílení zesilovače a U_{ref} hodnotu napětí reference.

Zesílení *GAIN* v případě využití diferenciálního vstupu se nastavuje v konfiguračních registrech zesilovačů *AMPxCSR*. Referenční napětí V_{REF} se nastavuje bity *REFS1* a *REFS0* v registru *ADMUX* a zároveň bity *ISRCEN* a *AREFEN* v registru *ADCSRB*. [11]

Pro výběr kanálu, který bude převáděn, slouží bity *MUX0-4* v registru *ADMUX*. Převod se spustí nastavením bitu *ADSC* do log. 1 v registru *ADCSRA*. V tomto registru se také spouští AD převodník bitem *ADEN* a nastavuje předdělička, která nastaví vstupní hodinový signál převodníku z frekvence oscilátoru mikrokontroleru. Pro výsledný hodinový signál převodníku platí rovnice [11]:

$$f_{ADC} = \frac{f_{osc}}{D}\ [Hz; Hz, -]\ , \tag{2.7}$$

kde f_{ADC} označuje hodinový signál AD převodníku, f_{osc} frekvenci oscilátoru a *D* dělící faktor daný předděličkou.

Pro správnou operaci vyžaduje převodník vstupní hodinový signál v rozmezí 50 kHz až 2 MHz. [11]

2.1.6 Čítač/časovač

Čítač/časovač (zkráceně č/č) je periferie mikrokontroleru, která se využívá k časování (mód časovač), nebo k čítání impulsů z externích zdrojů (z pracovních pinů mikrokontroleru, mód čítač) [11]

V této práci je využit mód časovač pro vytvoření časových intervalů. Zavedení časových intervalů zmenší zatížení mikrokontroleru, především omezí časté snímání analogových veličin.

Funkcí časovače je čítání impulsů hodinového signálu. Časovače v ATMega32M1 jsou 8bitové (č/č 0) a 16bitové (č/č 1), mohou tedy čítat maximálně 256, resp. 65536 impulsů před přetečením. Doba trvání jednoho impulsu hodinového signálu (*t*) rovna [11]:

$$\mathrm{t} = \frac{1}{f_{osc}}[\mu s; MHz]\ , \tag{2.8}$$

kde f_{osc} označuje frekvenci hodinového signálu mikrokontroleru.

Maximální doba, kterou tak může časovač časovat před přetečením (*T*), je [11]:

$$T = 2^n \cdot \frac{1}{f_{osc}}\ [\mu s; -, MHz]\ , \tag{2.9}$$

kde *n* označuje počet bitů čítače/časovače.

Časovaná doba lze prodloužit předděličkou, která se nastavuje v konfiguračních registrech čítače/časovače. Hodnoty předděličky u ATMega32M1 mohou být 1, 8, 64, 256 nebo 1024. Vztah pro dobu čítání při použití předděličky je roven [11]:

$$T = 2^n \cdot \frac{D}{f_{osc}}\ [\mu s; -, -, MHz]\ , \tag{2.10}$$

kde *D* označuje hodnotu předděličky.

Počet čítaných impulsů je uložen v registru *TCNxT* (*x* označuje konkrétní č/č). Příznak uplynutí časované doby může být dán přetečením časovače (z hodnoty 255, resp. 65535 na hodnotu 0) nebo porovnáním hodnoty *TCNTxT* s komparačním registrem *OCRx*. Do něj je možno nastavit menší počet impulsů, než je maximální hodnota *TCNTxT*, a tím optimalizovat dobu časování. [11]

Pro zjištění počtu impulzů, které je potřeba čítat pro určitý časový úsek, slouží rovnice (2.11) [11]:

$$N = \frac{T}{t \cdot D}\ [-; \mu s, \mu s, -]\ , \tag{2.11}$$

kde *N* označuje potřebný počet impulzů, *T* označuje požadovaný časový úsek, *t* dobu jednoho impulzu hodinového signálu a *D* hodnotu předděličky.

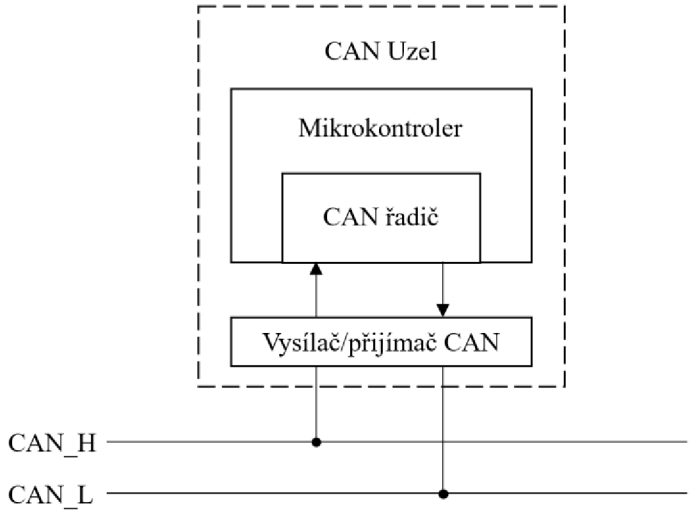
Příznaky přerušení při přetečení či shodě registru *TCNTxT* s *OCRx* jsou generovány registrem *TIFRx*. Povolení příznaků přerušení se provádí v registrech *TIMSKx*. [11]

2.1.7 Sběrnice CAN

Sběrnice CAN je sériová sběrnice, která je určená pro komunikaci mezi elektronickými řídícími jednotkami. Využívaná je především v automobilovém průmyslu pro komunikaci mezi elektronickými jednotkami v automobilech. Komunikace je zprostředkována zprávami, které jsou rozlišeny prostřednictvím identifikátorů ID. Jedná se o centralizovaný systém, tedy nepotřebuje ke své funkci zařízení typu server. [12]

Výhody CAN sběrnice [12]:

- Jednoduché a finančně nenáročné propojení
- Vysoká odolnost vůči EMI záření
- Vysoká robustnost komunikace při deformaci komunikačního signálu
- Umožňuje upřednostnit a okamžitě odeslat kritické zprávy nezávisle na předchozím přenosu (např. chybové zprávy).



Obrázek 2.3 Schéma zapojení zařízení (uzlu) k sběrnici CAN (převzato z [12])

Každé zařízení, které umožňuje připojení na CAN sběrnici, musí být vybaveno mikrokontrolerem s řadičem CAN a vysílačem/přijímačem sběrnice CAN. Fyzické spojení mezi zařízeními (přijímači/vysílači) je realizováno diferenciálním párem s označením vodičů CAN_H a CAN_L. Každé zařízení připojené ke sběrnici CAN se poté označuje jako uzel. Schéma zapojení je uvedeno v obrázku 2.3. [12]

Mezi standardy sběrnice CAN patří Low-speed CAN, High-speed CAN a CAN FD. Standardy Low-speed a High-speed CAN definují 4 typy zpráv (rámců) [12]:

- Datové rámce pro odesílání dat
- Rámce pro žádost dat z ostatních zařízení
- Rámce pro ohlášení chybových stavů
- Rámce pro ohlášení přetížení na lince

Struktura základního datového rámce je uvedena na obrázku 2.4 a podrobně popsána v tabulce 2.1.

Obrázek 2.4 Struktura základního datového rámce (převzato z [12])

Tabulka 2.1 Popis struktury základního datového rámce CAN [12][27]

Název pole	**Počet bitů**	**Funkce pole**
SOF	1	Indikuje začátek zprávy
Identifikátor	11	Identifikátor zprávy (čím nižší identifikátor, tím vyšší priorita)
RTR	1	Žádost o odeslání dat z jiného uzlu na sběrnici
IDE	1	Označuje, zda se jedná o základní nebo rozšířený datový rámec
r0	1	Rezervní bit
DLC	4	Udává délku datového pole
DATA	0-64	Pole s odesílanými daty
CRC	16	Pole s CRC kontrolou
ACK	2	Příznak úspěšného přijetí zprávy
EOF	7	Indikuje konec zprávy
IFS	3+	Časové okno pro uložení zprávy

Struktura rámců pro žádost dat z ostatních zařízení má obdobnou strukturu datového rámce, neobsahuje pouze pole *DATA*. Pro žádost o data se využívá především pole *RTR*, které na sběrnici ohlásí žádost o data a pole *Identifikátor* určí, z jakého uzlu jsou data požadována. [12]

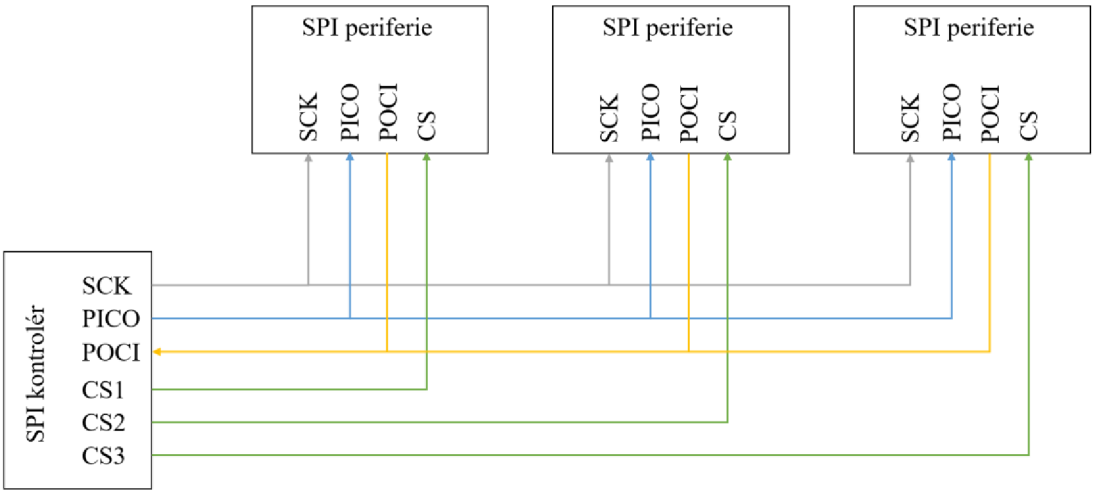
V moderních systémech, které vyžadují vyšší datovou propustnost, se využívá Standard CAN FD (Flexible Data Rate). Výhodou standardu oproti High-speed CAN je zvýšení maximální délky dat na 64 bytů, navýšení maximální přenosové rychlosti až na 8 Mb/s a schopnost dynamicky měnit velikost zprávy a přenosovou rychlost na základě požadavků systému v reálném čase. Oproti Low-speed a high-speed standardům je délka CRC pole v rámci 28 nebo 33 bitů. [12]

2.1.8 SPI

SPI je sériová sběrnice, která je běžně využívána v embedded systémech pro připojení externích periferií (označovány jako SPI periferie) k mikrokontroleru (označován jako SPI kontrolér). Jedná se o synchronní komunikaci, tedy rychlost komunikace určuje hodinový signál, který se přenáší mezi zařízeními. Datový přenos mezi zařízeními je realizován jako full-duplex, vysvětlení je uvedeno v obrázku 2.6. [13]

Propojení zařízení při využití SPI sběrnice je realizováno minimálně 4 vodiči (uvedeno na obrázek 2.5) [13]:

- *SCK* (Serial Clock) – Zajišťuje synchronizaci připojených zařízení pomocí hodinového signálu.
- *POCI* (Peripheral-out/controller-in) – Slouží pro přenos dat z periferie do kontroléru. V minulosti byl označován jako *MISO*.
- *PICO* (Peripheral-in/controller-out) – Slouží pro přenos dat z kontroléru do periferie. V minulosti byl označován jako *MOSI*.
- *CS* (Chip select) – Slouží pro aktivaci periferie, se kterou bude kontrolér aktuálně komunikovat. V případě připojení více SPI periferií musí každá periferie mít svůj *CS* signál, pomocí kterých bude SPI kontrolér přepínat komunikaci s periferiemi. V minulosti byl vodič *CS* označován jako *SS*.



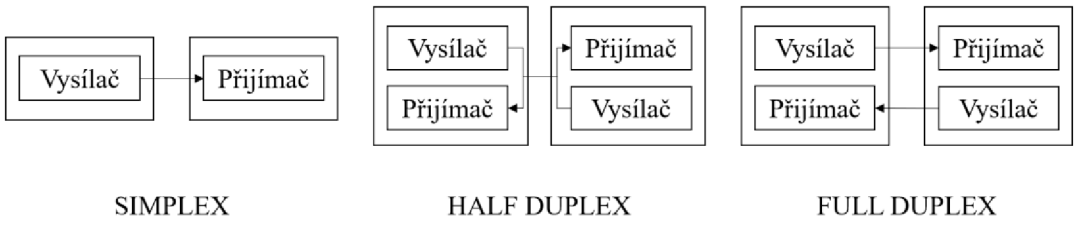
Obrázek 2.5 Schéma SPI sběrnice (převzato z [13])

Na rozdíl od protokolů jiných sběrnic nemá protokol SPI sběrnice definován žádné rámce nebo packety, odesílají se pouze čistá data, která se rozklíčují využitím hodinového signálu a délkou jednoho znaku (nejčastěji 8 bitů). Začátek a konec komunikace mezi SPI zařízeními je dán změnou signálu *CS* (obvykle je přenos blokován při stavu log. 1 a povolen při stavu log. 0). [14]

2.1.9 UART

UART je protokol k sériovému přenosu dat mezi dvěma zařízeními. Jeho výhodou je asynchronní přenos dat, tedy ke správné funkci se nemusí přenášet hodinový signál mezi zařízeními. Spojení mezi zařízeními může v případě použití UART sériové komunikace být realizováno ve formách [15]:

- Simplex: Komunikace probíhá pouze 1 směrem. Propojení mezi zařízeními je realizováno dvěma vodiči – linka a *GND*
- Half-duplex: Komunikace probíhá oběma směry, ale pouze po jedné lince. Směr toku dat se tedy musí přepínat a v jednom čase může být realizován přenos pouze jedním směrem. Propojení je opět realizováno dvěma vodiči – linka a *GND*
- Full duplex: Komunikace probíhá oběma směry souběžně po 2 linkách. Propojení je realizováno pomocí tří vodičů – linka *TX*, linka *RX*, *GND*



Obrázek 2.6 Typy spojení mezi zařízeními

Jelikož se mezi zařízeními nepřenáší hodinový signál, musí se před začátkem komunikace definovat oběma zařízením, jakou přenosovou rychlostí se budou data přenášet. Ta se u UARTu uvádí v jednotkách Baud, kdy 1 Baud odpovídá 1 bit/s. Mezi nejpoužívanější rychlosti UART komunikace patří 9600, 19200, 38400, 57600 a 115200 Baud. [15]

Kromě rychlosti přenosu musíme oběma zařízením definovat formu 1 packetu. Ten se skládá ze 4 částí [16]:

- Start bit: Značí začátek přenosu dat. V případě, že na lince neprobíhá přenos dat, udržuje linka úroveň log. 1. V případě začátku odesílání dat vysílač na jeden hodinový cyklus nastaví linku do log. 0. Přijímač při této změně z vysoké úrovně do nízké začíná číst data předdefinovanou přenosovou rychlostí.
- Datová část: Obsahuje data, která chceme přenášet. Velikost dat může být 5 – 8 bitů, v případě nepoužití parity až 9 bitů.
- Paritní bit: Slouží k částečné kontrole přenášených dat. Zjišťuje, zda přenášená data obsahují sudý nebo lichý počet bitů o úrovni log. 1. Kontrola parity nemusí být nastavena.

- Stop bity: Slouží k ukončení komunikace. Může být použit jeden nebo dva stop bity. Po dobu trvání jejich cyklu je linka v úrovni log. 0, po uplynutí cyklu je linka nastavena do úrovně log.1 a připravena k přijetí dalších packetů.

Start bit (1 bit)	DATA (5-9 bitů)	Paritní bit (0 nebo 1 bit)	Stop bit (1 nebo 2 bity)

Obrázek 2.7 UART packet (překresleno z [16][16])

2.1.10 Technologie mobilních sítí – GSM a GPRS

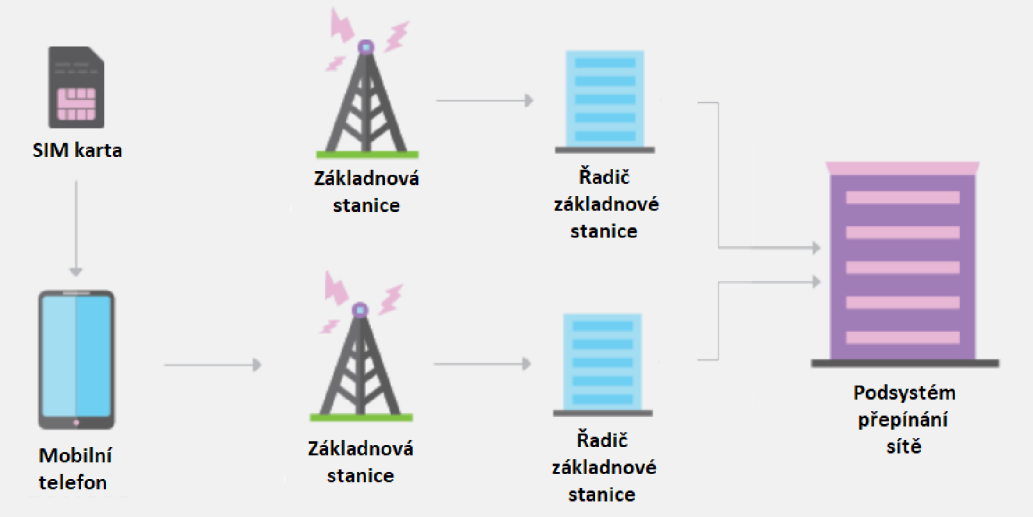
GSM je standart pro digitální mobilní síť a jeho hlavním cílem je umožňovat hlasovou komunikaci mezi uživateli, případně odesílat krátké zprávy (SMS). [17]

GSM využívá takzvanou buněčnou strukturu. To znamená, že území pokryté GSM signálem je rozděleno do buněk a každá z nich má svou základnovou stanici. Základnová stanice obsahuje vysílač/přijímač, který slouží k připojení uživatelů do sítě. [17]

Uživatel musí mít zařízení, které umožňuje využívat hlasové a datové služby v síti GSM (mobilní telefon, SIM modul), a SIM kartu, která poskytuje identifikační údaje uživatele (telefonní číslo). Přenos mezi zařízením uživatele a základnovou stanicí probíhá rádiově (dnes v ČR na frekvencích 800, 900, 1800, 2100 MHz). [17][18]

Základnové stanice jsou pomocí řadičů připojeny do tzv. podsystému přepínání sítě s řídícími centry. Podsystém má za úkol vyhledávat a spojit uživatele, případně určit jejich polohu. Je propojen pevným spojením z důvodu vyšší datové propustnosti. [17]

Výhodou této buněčné struktury je efektivita a spolehlivost. Každá buňka má omezenou kapacitu připojených zařízení, nedochází tak k přetížení vysílače. Pokud je v jedné buňce kapacita připojených zařízení naplněna, přesměruje uživatelské zařízení na jinou buňku. [17]



Obrázek 2.8 Blokové schéma GSM sítě (překresleno z [17])

GPRS je nadstavba sítí GSM a jejím hlavním cílem je připojení k internetu přes mobilní data. Rozdílem mezi sítí GSM a GPRS je způsob přenosu dat. GSM udržuje po dobu spojení jeden kanál mezi mobilním zařízením a základnovým vysílačem, zatímco u GPRS jsou data přenášeny pomocí packetů. Tím se zvýšila efektivita sítě (uživatel, který aktuálně neodesílá nebo nepřijímá data, nemusí stále udržovat spojení, a může prostor nabídnout jinému uživateli). [19]

Připojení k internetu pomocí mobilních poskytuje operátor, který SIM kartu vydal. Každý operátor má svoje APN (access point name), což je webová adresa, pomocí které se do sítě uživatel připojí. V některých případech je potřeba také k připojení zadat uživatelské jméno a heslo pro autentifikaci. [19]

Generace mobilních sítí [20]:

- 1G: Mobilní síť byla založena na analogovém rádiovém přenosu, který umožňoval pouze hovory, síť nebyla bezpečná a spolehlivá.
- 2G: Tato generace přinesla přenos dat v síti pomocí digitálního signálu. Síť tak byla spolehlivější, bezpečná a umožnila přenos SMS a MMS.
- 2,5G: Rozšíření 2G sítě o GPRS (umožnila připojení k internetu pomocí mobilních dat, ale pouze k posílání a příjem e-mailů).
- 3G: Zrychlení datového přenosu oproti síti 2G. To umožnilo prohlížet webové stránky, uskutečnit video hovory nebo sledovat on-line videa.
- 4G: Zrychlení datového přenosu oproti síti 3G (teoretická rychlost 100 Mbit/s, umožnila streamování videí ve vysoké kvalitě, připojení s nízkou latencí).
- 5G: Nejnovější mobilní síť, nejrychlejší přenos dat (teoretická rychlost 2000 Mbit/s), vytvořeno pro velké datové přenosy v IoT sítích (např. chytrá města).

2.2 Server a serverová aplikace

Dalším prvkem IoT systému je server a serverová aplikace. Serverová aplikace plní funkci přenosu dat mezi vstupně/výstupní bránou a ovládací aplikací. Server tak představuje centrální úložiště IoT systému, kde se nachází data o všech IoT zařízeních.

Data se mohou uchovávat v paměti serverové aplikace, na lokálním úložišti serveru nebo na cloudovém úložišti. Při návrhu serveru je potřeba zvážit, jak jsou důležitá jednotlivá data. Například historii naměřených dat je vhodné ukládat na lokální úložiště serveru nebo na cloudové úložiště, kdy v případě selhání serverové aplikace zůstanou data uložena. Mezi cloudové úložiště patří např. Google Cloud Platform nebo Azure od Microsoftu. Nepodstatná data, např. která jsou použita pouze pro jednorázové nastavení IoT zařízení, je zase vhodné uchovávat v paměti serverové aplikace.

Pro serverovou aplikaci je vždy potřeba zvážit, jak responzivní má být IoT systém a jaký objem dat je možno sítí přenášet. Navržený autonomní systém nevyžaduje sledování vytápění v reálném čase, jelikož tepelná kapacita betonu neumožňuje jeho okamžitou změnu teploty. Zároveň je systém připojen do internetové sítě prostřednictvím

mobilního internetu, je tedy potřeba do návrhu zahrnout úsporu dat. Proto bude serverová aplikace realizována jako API. Pro vytvoření API bude využita platforma ASP.NET a pro přehlednost zdrojového kódu bude využit architektonický styl MVC.

2.2.1 Platforma ASP.NET

ASP.NET je rozšířením platformy .NET a umožňuje vytvářet webové aplikace. Platforma byla vyvinuta společností Microsoft. Pro programování využívá programovací jazyky C# nebo F#. [21]

ASP.NET rozšiřuje platformu .NET o [21]:

- Knihovny pro vytváření webových požadavků v jazyce C# nebo F#
- Šablonu Razor, která kombinuje programovací jazyky HTML, C#, CSS a JavaScript pro vytvoření vizuálu webové stránky
- Architektonický styl MVC (Model-View-Controller)
- SignalR – protokol pro vytváření aplikací v reálném čase
- Další knihovny, např. knihovny pro autentifikaci s použitím externích účtů, například Google účtu

ASP.NET umožňuje vytvářet [21]:

- Klasické webové stránky a služby
- API (Rozhraní pro programování aplikací) – umožňuje dvěma různým systémům nebo aplikacím si vzájemně předávat data bez toho, aby bylo nutné znát vnitřní podrobnosti obou systémů.
- Aplikace běžící v reálném čase – např. služby pro živý přenos, využívá protokolu SignalR
- Mikroslužby – služby běžící na modulech s omezeným výpočetním výkonem na platformě Docker. Každá služba je zabalena do standardizovaného kontejneru, který se spouští na hostitelském systému. Každému kontejneru se může přiřadit omezený počet jader procesoru a limit využití operační paměti.

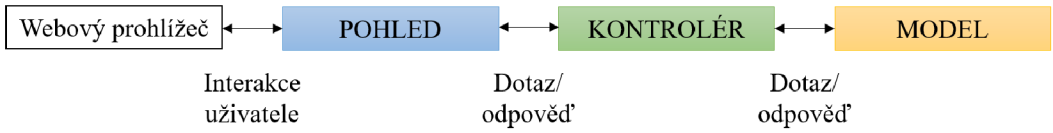
2.2.2 Architektonický styl MVC (Model-View-Controller)

Styl MVC slouží k přehlednému oddělení aplikace do tří komponent – Model, Pohled a Kontrolér. Má důležitou roli při vytváření webových aplikací za použití objektově orientovaného programování. [22]

Funkce komponent [22]:

- Model – Tato komponenta se zabývá datovou vrstvou. Její úlohou je definování struktury dat, kterou aplikace využívá k zápisu či odesílání dat.
- Pohled – Využívá se u webových stránek a aplikaci s vizuálním uživatelským rozhraním, kde definuje vzhled stránky a prvky, které budou zobrazovat předaná data. V ASP.NET se pohledy vytváří v šabloně Razor.

- Kontrolér – Zajišťuje logiku mezi modelem a pohledem. Jeho úlohou může být načtení dat z databáze a předání do pohledu, nebo naopak nahrání dat z webového formuláře do databáze.



Obrázek 2.9 Blokové schéma architektonického stylu MVC (překresleno z [22])

2.2.3 REST API

REST API je jedním ze stylů pro Web API, což je rozhraní pro programování aplikací, které využívá k přenosu dat hypertextový protokol (HTTP). Jsou na něm postaveny služby pracující na pozadí, které slouží k práci s databází, přenos dat ve webových službách, nebo k přenášení dat z/do IoT zařízení. Využívá architekturu server-klient, kdy klient odesílá požadavek na server a server následně klientu odešle zpětnou vazbu, např. ve formě chybového kódu nebo požadovaných dat. [23]

REST API definuje 4 metody, kterými může klient pracovat s daty na API serveru, případně v databázi [24]:

- Metoda GET: Slouží klientovi k získání dat.
- Metoda POST: Slouží k odeslání nových dat.
- Metoda PUT: Slouží k aktualizaci dat podle určeného identifikátoru.
- Metoda DELETE: Slouží k smazání dat podle určeného identifikátoru.

Každá vytvořená metoda, kterou může klient využít pro požadavek, má svojí URL adresu. Ta je ve formátu [24]:

- Pro metody GET a POST:

```
http(s)://<URL_adresa_API_serveru>/<Metoda>?<parametr1>=<hodnota1>&<parametr2>=<hodnota2>
```

- Pro metody PUT a DELETE:

```
http(s)://<URL_adresa_API_serveru>/<Metoda>/<ID>?<parametr_1>=<hodnota1>&<parametr_2>=<hodnota2>
```

Parametry za znakem ‚*?*‘ jsou nepovinné. Používají se například pro identifikaci klienta, který odeslal požadavek na server, nebo o odeslání jiných doplňujících dat. [24]

Pokud chceme serveru odeslat data metodou POST nebo PUT, odesíláme data přednostně v těle HTTP požadavku. Data můžeme odesílat ve formátech JSON, HTML, XLT, Python, PHP nebo jako prostý text [25].

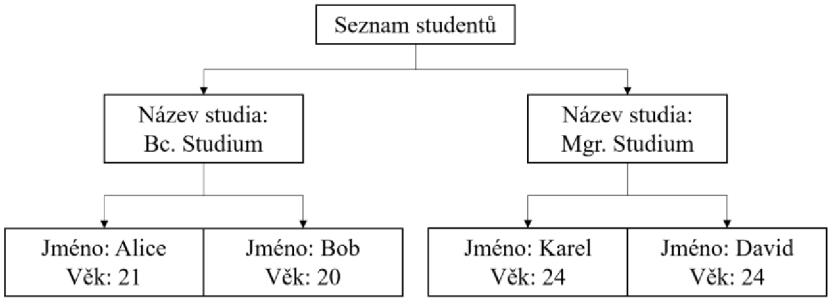
Stejnými formáty také získáváme data pomocí GET metody. Dnes je nejpoužívanějším formátem JSON. Jeho největší výhodou je přehlednost složitějších struktur. Příkladem formátu dat JSON může být [26]:

```
{
  "SeznamStudentů":
  [
    {
      "názevStudia": "Bc. Studium",
      "studenti":
      [
        {
          "jméno": "Alice",
          "věk": 21
        },
        {
          "jméno": "Bob",
          "věk": 20
        }
      ]
    },
    {
      "názevStudia": "Mgr. Studium",
      "studenti":
      [
        {
          "jméno": "Karel",
          "věk": 24
        },
        {
          "jméno": "David",
          "věk": 24
        }
      ]
    }
  ]
}
```

Dekódování JSON formátu [26]:

- Základem formátu jsou objekty, které jsou ve zprávě uzavřeny znaky *‚{‘* a *‚}‘*
- Každý objekt má své parametry:
  - Textový parametr má zápis *„parametr“: „hodnota_parametru“*
  - Parametry jiného datového typu mají zápis: *„parametr“: hodnota_par*
- Obsahuje-li zpráva pole objektů, je ve zprávě uzavřené znaky *‚[‘* a *‚]‘*
- Parametry a objekty jsou odděleny čárkou

Výše uvedený příklad JSON zprávy tedy obsahuje strukturu:



Obrázek 2.10 Znázornění výše uvedeného příkladu JSON struktury

Pro odeslání dat ve formátu JSON pomocí metod PUT a POST je potřeba do hlavičky HTTP požadavku definovat parametr *„Content-Type“* jako *„application/json“*. [26]

Výhodou REST API je také využití HTTP stavových kódů při volání API metod. Kódy jsou reprezentovány trojciferným čísly a dělíme je do pěti skupin [27]:

- Informační kódy (1xx)
- Kódy úspěšného dokončení (2xx)
- Kódy přesměrování (3xx)
- Chybové kódy, kdy chyba pochází ze strany klienta (4xx)
- Chybové kódy, kdy chyba pochází ze strany serveru (5xx)

Nejčastější stavové kódy jsou uvedeny v tabulce 2.2:

Tabulka 2.2 Nejčastější stavové kódy REST API [27]

Stavový kód	Popis	Podrobný popis
200	OK	Žádost byla úspěšná
400	Bad request	Syntaxe požadavku je neplatná
401	Not authorized	Požadavek potřebuje doplnit o přihlašovací údaje
403	Forbidden	Uživatel nemá dostatečná práva pro požadavek
404	Not found	Server nemůže najít danou URL adresu
500	Internal server error	Chyba při zpracování požadavku na serveru

2.3 Vizualizační a ovládací aplikace

Při návrhu vizualizační a ovládací aplikace je důležité rozdělit návrh na část běžící na pozadí a na vizualizační část.

Část běžící na pozadí musí umožňovat přenos dat ze serveru a serverové aplikace. Musí tedy umět pracovat se síťovou kartou počítače, na kterém je aplikace nasazena a který je připojen k IoT lokální síti či k internetu. Zároveň musí umět volat požadavky pro příjem nebo odesílání dat definované serverem, v případě navrženého systému vytápění se jedná API dotazy prostřednictvím HTTP protokolu.

Část vizualizační obsahuje ovládací prvky a indikátory dat. Při návrhu vizualizační části je důležité vytvořit minimalistický panel, nejlépe tvořený imitacemi vzhledu fyzických ovládacích a indikačních prvků (tlačítka, posuvný číselník). Pro přehledné uspořádání množství dat je potřeba využívat tabulek a grafů. Vhodné je také zavést zpětnou vazbu při vykonání akce (např. rozsvítit LED při stisku tlačítka) a rozdělit ovládací prvky do oken, mezi kterými může uživatel přepínat.

Vhodnou volbou pro vytváření vizualizační a ovládací aplikace je programovací prostředí LabVIEW, které rozděluje program na vizualizační část a část běžící na pozadí. [28] Zároveň obsahuje databázi knihoven funkcí pro část běžící na pozadí, je tak možno v aplikaci volat HTTP dotazy nebo rozklíčovat JSON strukturu dat, které požaduje navržená serverová aplikace.

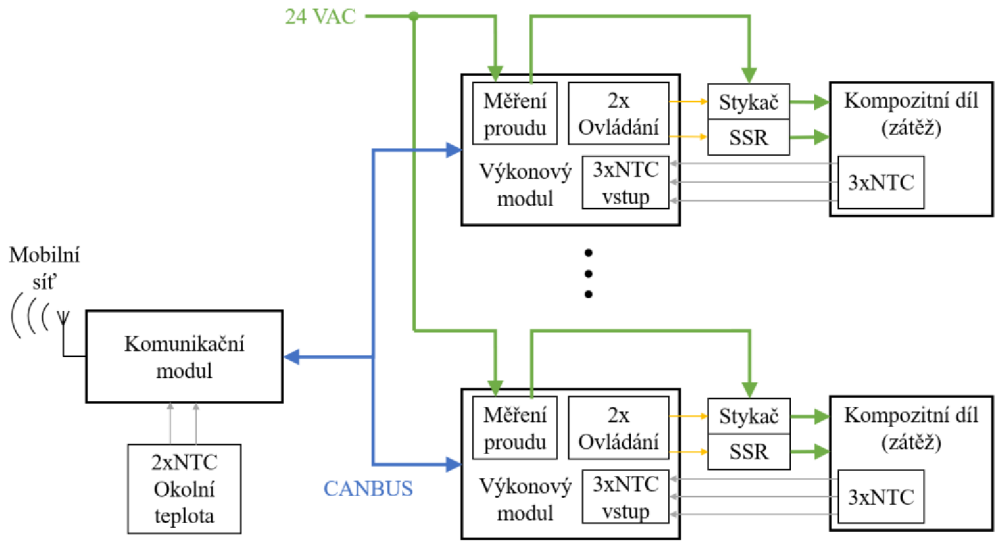
3. NÁVRH ŘEŠENÍ

K vytápění dílců z kompozitních materiálů bude navržen autonomní systém s monitoringem teplot dílců a proudového odběru. Kompozitní materiál je vodivý v celém svém objemu a mezi protilehlými konci dílce vykazuje odpor v řádu jednotek Ω, vytápění dílců je proto realizováno přivedením bezpečného střídavého napětí 24 VAC přímo na kompozitní dílce. Autonomní systém bude regulovat vytápění spínáním tohoto napětí prostřednictvím stykače a SSR relé na základě zpětné vazby o teplotě kompozitních dílců.

Autonomní systém bude prostřednictvím SIM modulu připojen k mobilnímu internetu, který je potřeba pro připojení na server vzdáleného řízení a monitoringu autonomního systému. Server bude realizován jako vlastní API server s metodami REST API. Bude tak připraven pro snadnou implementaci nových funkcí a zároveň bude umožňovat připojení a ovládání více systému z jednoho serveru. K přehledné vizualizaci naměřených dat a jednoduchému ovládání systému bude vytvořen ovládací panel v programovacím prostředí LabVIEW.

3.1 Autonomní systém vytápění

Autonomní systém vytápění se skládá ze 2 modulů: komunikačního a výkonového. Blokové schéma propojení modulů je uvedeno na obrázku 3.1. V IoT struktuře představuje výkonový modul IoT zařízení, komunikační modul vstupně/výstupní bránu.



Obrázek 3.1 Blokové schéma autonomního systému vytápění

3.1.1 Výkonový modul

Výkonový modul řídí spínání napětí pro vytápění kompozitních dílců. Obsahuje 2 výstupy pro připojení ovládání spínacích prvků – jeden je určený pro SSR relé, druhý pro stykač. K vytápění je využito bezpečné střídavé napětí 24 V. Vytápění je regulováno zpětnou vazbou o teplotě betonových dílů. K měření teploty betonového dílu se využívá NTC termistorů, modul umožnuje celkem připojit až 3 NTC termistory. Regulace teploty dílců je prováděna s hysterezí 1 °C.

Výkonový modul zároveň monitoruje proudový odběr připojené zátěže Hallovým snímačem proudu. Monitoring proudu zároveň slouží jako softwarová pojistka – při překročení maximální povolené proudové spotřeby při vytápění odpojí napětí od kompozitních dílců.

Modul může vytápět v režimech NORMAL, ECO nebo SMART. Režimy NORMAL a ECO regulují vytápění pouze na základě zpětné vazby o teplotě vytápěných dílců. V režimu ECO se výkon vytápění sníží na polovinu z důvodu spínání napětí pouze v jedné polovině přednastavené periody. V tomto módu se uplatní jako spínací prvek pouze SSR. V režimu NORMAL není vytápění nijak výkonově omezeno a uplatní se oba spínací prvky. V režimu SMART se uplatňuje získaná předpověď počasí a modul určuje vytápění na základě okolní teploty a výskytu srážek. Vytápění není výkonově omezeno a uplatňuje se opět obou spínacích prvků.

Nastavení požadované teploty betonových dílů, povolení vytápění a volbu režimu vytápění přebírá od komunikačního modulu prostřednictvím komunikace CAN. Komunikaci CAN zároveň využívá pro odesílání naměřených teplot betonových dílů do komunikačního modulu.

3.1.2 Komunikační modul

Modul zajišťuje sběr dat z výkonových modulů prostřednictvím komunikace CAN. Naměřená data poté jednou za 10 minut odesílá na řídící server. Po odeslání dat ze serveru stáhne aktualizované nastavení pro vytápění kompozitních dílců a předpověď počasí pro další hodinu na základě zeměpisných souřadnic modulu. Pomocí sběrnice CAN poté stažené informace odešle do výkonových modulů.

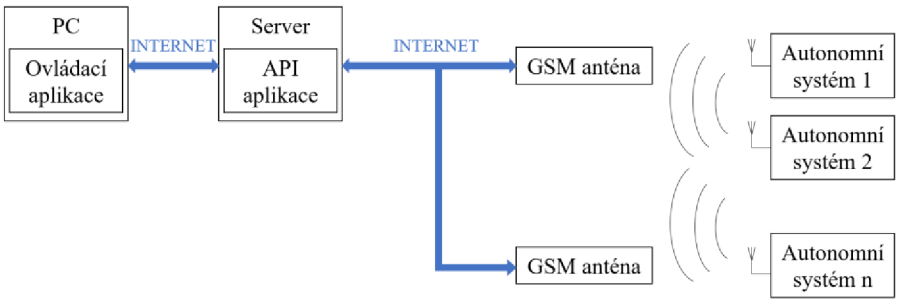
Modul zároveň pro kontrolu měří okolní teplotu pomocí 2 NTC termistorů, kterou srovnává s teplotou získanou z předpovědi počasí pro aktuální hodinu. Pokud je naměřená teplota nižší alespoň o 2 °C než teplota získána z předpovědi počasí, předá výkonovým modulům informaci o naměřené okolní teplotě místo teploty z předpovědi počasí.

3.2 Návrh vzdáleného řízení a monitoringu autonomního systému prostřednictvím mobilní sítě

Návrh zahrnuje vytvoření vlastního řídícího serveru a připojení autonomního systému k internetu. Vzhledem k tomu, že autonomní systém může být nasazen v místech, kde je nedostupné LAN nebo Wi-Fi připojení, zvolila se možnost připojení do internetu prostřednictvím mobilní sítě a SIM modulu.

Pro připojení autonomního systému k mobilní síti bude využit SIM modul SIM7000E NB-IOT HAT od společnosti Waveshare [29], který bude osazen v komunikačním modulu autonomního systému. Komunikaci mezi mikrokontrolerem komunikačního modulu a SIM modulem zprostředkuje UART sériová komunikace.

Řízení komunikačního modulu a získávání měřených dat prostřednictvím internetu bude realizováno HTTP API dotazy ve formátu REST API, které budou definované metodami na vlastním HTTP API serveru. Zároveň návrh předpokládá připojení více autonomních systému k jednomu řídícímu serveru. API server plní funkci aplikace běžící na pozadí v IoT struktuře.



Obrázek 3.2 Blokové schéma monitoringu a řízení autonomního systému

3.2.1 HTTP API Server

API server bude vytvořen pomocí vývojářské platformy ASP.NET. Server bude zajišťovat 3 funkce:

- Definování API metod pro řízení a sběr dat, které budou rozděleny do dvou kontrolérů (metody určené pro komunikační moduly a pro ovládací aplikaci)
- Zjištění předpovědi počasí pro daný komunikační modul na základě zeměpisných souřadnicích lokace modulu
- Ukládání historie naměřených dat pro analýzu v ovládací aplikaci

Struktura uložených dat bude definována v modelech. Každý model může být vypsán pomocí JSON struktury.

3.2.2 Rozšíření komunikačního modulu o SIM modul

Modul od společnosti Waveshare je založen na SIM modulu SIM7000E. Je ovládán pomocí AT příkazů, které jsou uvedeny v katalogových listech modulu. [29] [30] [31]

[32] [33] AT příkazy budou odesílány z mikrokontroleru komunikačního modulu pomocí sériové linky UART. Je důležité, aby příkazy obsahovaly na konci svého textového řetězce oddělovací znaky „\r\n“.

Obrázek 3.3 Modul Waveshare SIM7000E NB-IOT HAT (převzato z [29])

Samotný SIM modul SIM7000E komunikuje na linkách UART s napěťovou úrovní 1,8 V. [31] Modul od společnosti Waveshare však obsahuje implementovaný převodník úrovní pro UART linky, je tedy možné k modulu přímo připojit UART linky z mikrokontroleru komunikačního modulu pracujících na úrovni napětí 3,3 V.

SIM modul zároveň obsahuje automatické nastavení přenosové rychlosti UART linek, ovšem pouze pro rychlosti 9600, 19200, 38400, 57600 a 115200 Baud. Jelikož nebude nutná vysoká přenosová rychlost, bude zvolena rychlost 9600 baud. Pokud by bylo potřeba nastavit nestandartní přenosovou rychlost, musí se využít příkaz *„AT+IPR“*.

Modul je ovládán pomocí příkazů AT. Návod, jak tyto příkazy využít a SIM modul připojit do mobilní sítě, k mobilnímu internetu a vytvářet pomocí něj HTTP API dotazy, je uveden v Příloha A - Návod k využití AT příkazů. Sekvence AT příkazů budou rozděleny do funkcí v knihovně programu mikrokontroleru. Knihovna bude obsahovat funkce pro připojení, odeslání naměřených dat a příjem parametrů pro nastavení autonomního systému vytápění.

3.3 Ovládací aplikace

Ovládací aplikace bude vytvořena v programovacím prostředí LabVIEW. [28] Je zástupcem vizualizačního a ovládacího prostředí v IoT struktuře. Využívá přichystaných metod API serveru, které jsou uvedeny v kapitole 7.5.

Aplikace bude umožňovat připojení k API serveru s použitím jména klienta a URL adresy. Za běhu aplikace bude také umožněno přepojení na jiný server. Po připojení klienta vypíše do seznamu identifikátory autonomních systému připojených k API serveru.

Po výběru jednoho ze systému bude přehledně zobrazovat poslední naměřená data teplot a proudového odběru, a to i detailně na každém výkonovém modulu. Zároveň bude aplikace obsahovat ovládací prvky pro nastavení vytápění každého z připojených systémů a parametry vytápěných ploch.

4. NÁVRH ELEKTRONIKY KOMUNIKAČNÍHO A VÝKONOVÉHO MODULU

Tato část se zabývá návrhem elektroniky komunikačního a výkonového modulu. Požadavky na oba moduly bylo, aby umožňovaly montáž do elektrických rozvaděčů s DIN lištou, 2 typy napájecího napětí (12 VDC a 24 VAC) a spolehlivou komunikaci mezi moduly.

Důležitým požadavkem na elektroniku komunikačního modulu je připojení k internetu prostřednictvím GPRS sítě. Dalšími požadavky jsou měření okolní teploty, možnost zálohovat naměřená data lokálně a zobrazení důležitých informací v místě instalace.

Požadavky na elektroniku výkonového modulu jsou možnost měření proudu do rozsahu hodnot minimálně ±30 A, možnost snímání teploty vytápěného materiálu ve 3 bodech objemu a možnost ovládat spínání 1x SSR relé a 1x stykač.

V následujících kapitolách je uvedeno řešení jednotlivých požadavků. Elektronická schémata a DPS obou modulu byly realizovány v programu EAGLE [34].

4.1 Návrh komunikačního modulu

Elektronika komunikačního modulu byla navržena pro zabudování do krabičky D4MG [35] od společnosti GAINTA pro použití v rozváděčích s lištami DIN. Krabička v rozvaděči zabírá 4 modulové pozice. Logiku modulu obstarává mikrokontroler ATMega32M1 od společnosti ATMEL [11]. Procesor byl vybrán z důvodu zabudovaného CANBUS řadiče pro komunikaci po sběrnici CAN.

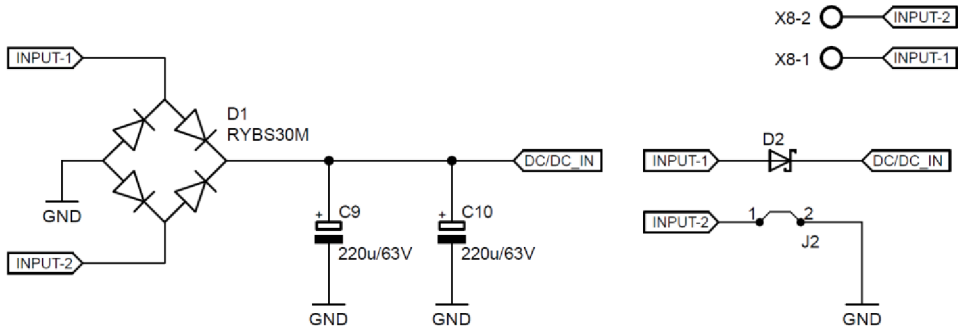
Modul bude zároveň obsahovat již dříve zmíněný SIM modul SIM7000E pro připojení k mobilnímu internetu, komunikovat s mikrokontrolerem bude prostřednictvím sběrnice UART. Také bude obsahovat microSD slot pro zálohování historie naměřených dat, ten bude propojen s mikrokontrolerem prostřednictvím sběrnice SPI. K SPI sběrnici bude dále připojen OLED displej pro rychlý přehled informací na místě v rozvaděči. Pro ladění modulů bude sloužit skupina 3 LED a 3 tlačítek. Ke snímání okolní teploty je využita dvojice NTC termistorů připojených do odporových děličů a vyhodnocení teploty zajišťuje AD převodník mikrokontroleru.

4.1.1 Systém napájení

Modul byl navržen pro napájení přímo z bezpečného střídavého napětí 24 VAC, které je využito pro vytápění kompozitního materiálu, nebo pro napájení ze zdroje 12 VDC. Napájení se připojuje konektorem *X8*, a to při obou verzích napájecího napětí. Při volbě napájecího napětí 12 VDC je zajištěna ochrana proti přepólování napětí Schottkyho diodou. Při volbě napětí 24 VAC je vstupní napětí převedeno na stejnosměrné napětí přibližně 36 VDC dané vztahem (4.1), a to prostřednictvím Graetzova můstku *D1*.

$$U_{DC} = U_{AC} \cdot \sqrt{2} = 24 \cdot \sqrt{2} = 33{,}94\,V\,, \tag{4.1}$$

V obou případech je poté vstupní stejnosměrné napětí filtrováno dvojicí kondenzátorů C9 a C10. Schéma napájení komunikačního modulu je uvedeno v obrázku 4.1.



Obrázek 4.1 Schéma zapojení napájení komunikačního modulu

Napájení SIM modulu připojeného do komunikačním modulu zajišťuje Step-down DC/DC měnič LMR33630CDDA od společnosti Texas Instruments. Jedná se o měnič s rychlostí spínaní 2100 kHz, s rozsahem vstupního napětí 3,8-36 V, rozsahem výstupního napětí 1-24 V a maximálním výstupním proudem 3 A [36].

Výstupní proud byl nejdůležitějším parametrem při výběru DC/DC měniče, jelikož SIM modul může mít proudovou spotřebu až 2 A při módu vysílání GPRS [37], který bude pro komunikaci s řídícím serverem využit. Dalšími požadavky na měnič byl rozsah vstupního napětí alespoň 12-36 VDC, který měnič splňuje, a poté možnost výstupního napětí 5 V, které požaduje SIM modul.

Pro výstupní napětí 5 V byly využity doporučené externí součástky DC/DC měniče, případně dopočítány v rovnicích (4.2)(4.1) až (4.5), podle katalogového listu výrobce [36]. Schéma zapojení je uvedeno na obrázku 4.2:

Volba rezistorů pro dělič určující výstupní napětí:

Katalogový list výrobce doporučuje, aby rezistor mezi výstupním napětí a pinem *FB* (rezistor R_6) měl hodnotu odporu 100 kΩ. Pro výpočet druhého odporu v děliči (R_7) platí vztah [36]:

$$R_7 = \frac{R_6}{\frac{U_{out}}{U_{ref}} - 1} = \frac{100 \cdot 10^3}{\frac{5}{1} - 1} = 25\ k\Omega\,, \tag{4.2}$$

kde U_{out} označuje výstupní napětí měniče a U_{ref} fixní hodnotu napětí na pinu FB $U_{ref} = 1$ V.

Jako R_7 byl tedy zvolen rezistor 24.9 kΩ.

Volba cívky:

Pro velikost indukčnosti cívky platí vztah [36]:

$$L = \frac{U_{IN} - U_{OUT}}{f_{sw} \cdot K \cdot I_{outmax}} \cdot \frac{U_{out}}{U_{in}}, \tag{4.3}$$

kde U_{IN} označuje vstupní napětí, U_{out} výstupní napětí, f_{sw} frekvenci spínaní, na které pracuje měnič (f_{sw} = 2100 kHz), I_{outmax} maximální výstupní proud a K činitel zvlnění výstupního proudu (katalogový list doporučuje použít hodnotu K = 0,3) [36].

Pro obě možná vstupní napětí (12 VDC a 36 VDC) jsou potřeba cívky indukčnosti:

$$L = \frac{12-5}{2100 \cdot 10^3 \cdot 0{,}3 \cdot 3} \cdot \frac{5}{12} = 1{,}54\ \mu H, \tag{4.4}$$

$$L = \frac{36-5}{2100 \cdot 10^3 \cdot 0{,}3 \cdot 3} \cdot \frac{5}{36} = 2{,}28\ \mu H, \tag{4.5}$$

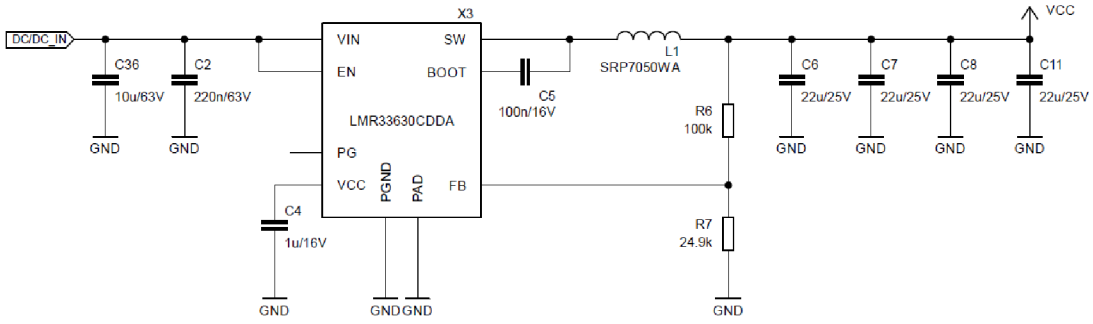
Pro testovanou verzi byla zvolena cívka SRP7050WA-1R5M s hodnotou indukčnosti 1,5 µH [38], jelikož napájení prototypu bude realizováno zdrojem 12 VDC. Při volbě napájecího napětí 24 VAC je tedy potřeba cívku nahradit za cívku s indukčností 2,2 µH.

Volba výstupních kondenzátorů:

Katalogový list výrobce doporučuje pro výstupní napětí U_{out} = 5 V dva výstupní kondenzátory v pouzdře SMD1206 s malým ESR (doporučené dielektrikum alespoň X5R) a hodnotou kapacit 22 µF. Navržené schéma a DPS ovšem umožnění připojení až 4 kondenzátorů v případě, kdy by bylo požadované výstupní napětí jiné nebo byl zvolen obdobný integrovaný obvod měniče s nižší frekvencí spínání, což by vyžadovalo až 4 výstupní kondenzátory podle katalogového listu výrobce [36].

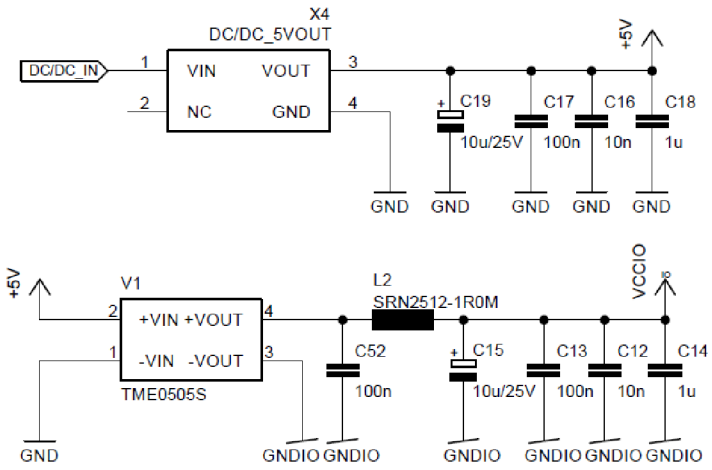
Volba ostatních komponent:

Ostatní komponenty měniče byly vybrány podle doporučených hodnot a SMD pouzder z katalogového listu výrobce pro výstupní napětí U_{out} = 5 V [36]. Jedná se o vstupní kondenzátory 10 µF + 220 nF, kondenzátor mezi piny *SW* a *BOOT* 100nF a kondenzátor na pinu V_{CC} 1 µF.



Obrázek 4.2 Schéma zapojení DC/DC měniče pro napájení SIM modulu

Pro napájení ostatní elektroniky byly využity DC/DC měniče CUI PXO7805-500-M-TR [39] a TRACO TME0505S [40]. Měnič CUI využívá vstupní napětí 12 VDC nebo 36 VDC a vytváří napájecí větev *+5V*, která je použita pro napájení neoddělené části sběrnice CAN a jako vstupní napětí DC/DC měniče TRACO. Ten vytváří napájecí větev V_{CCIO} a odděluje ji od předešlé napájecí struktury. Větev V_{CCIO} slouží k napájení mikrokontroleru a připojených periferií.



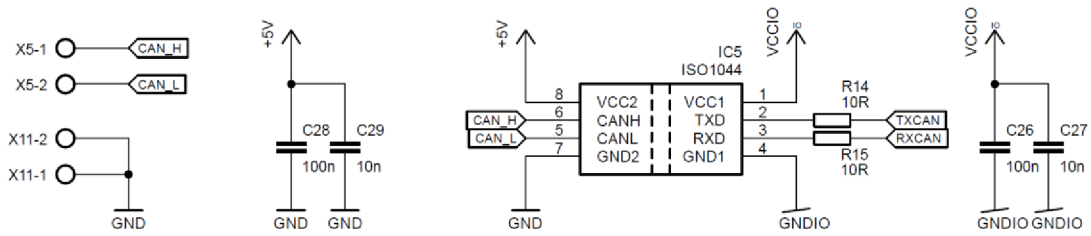
Obrázek 4.3 Schéma zapojení DC/DC měničů pro napájení ostatní elektroniky

4.1.2 Mikrokontroler

Použitý mikrokontroler ATMega32M1 a jeho nejdůležitější části jsou zapojeny podle katalogového listu výrobce [11]. Pro napájení mikrokontroleru a AD převodníku je přidána sada filtračních keramických kondenzátorů o velikostech kapacit 1 uF + 100 nF + 10 nF. K mikrokontroleru je připojen krystal NX3225GD [41] o frekvenci 8 MHz s dvojicí kondenzátorů 12 pF. Programování mikrokontroleru je možné prostřednictvím 6pinového konektoru AVR ISP. Nevyužité piny mikrokontroleru jsou připojeny externím pulldown rezistorem k zemi *GNDIO* pro eliminaci rušení.

4.1.3 Sběrnice CAN

Pro komunikaci prostřednictvím sběrnice CAN je k mikrokontroleru připojen CANBUS vysílač/přijímač ISO1044 od společnosti Texas Instruments [42]. Ten zároveň odděluje kroucenou dvojlinku sběrnice CAN od mikrokontroleru, tedy chrání mikrokontroler v případě poruchy na neoddělené části. K napájení vysílače/přijímače jsou jak na oddělené, tak na neoddělené straně přidány filtrační kondenzátory o kapacitách 100 nF a 10 nF. Kroucená dvojlinka sběrnice CAN se propojuje s výkonovými moduly pomocí konektoru *X5*. Pro vyrovnání zemního potenciálu mezi moduly, které je požadováno pro správnou funkci sběrnice CAN, slouží konektor *X11*.

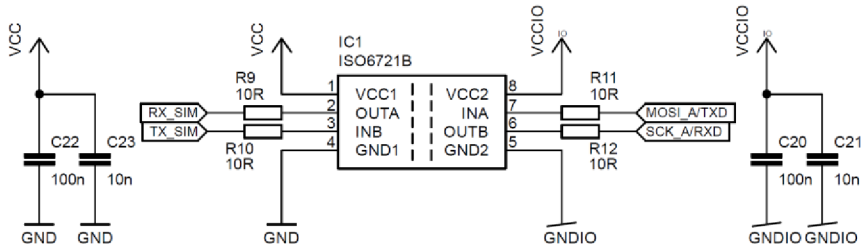


Obrázek 4.4 Schéma zapojení sběrnice CAN komunikačního modulu

4.1.4 SIM modul a sběrnice UART

Pro oddělené připojení komunikace mezi SIM modulem a mikrokontrolerem prostřednictvím sběrnice UART byl použit UART izolátor ISO6721B od společnosti Texas Instruments [43]. Napájení oddělené i neoddělené části je doplněno o filtrační kondenzátory o kapacitách 100 nF a 10 nF. Stejné filtrační kondenzátory jsou použity i u napájení SIM modulu.

SIM modul je k desce připojen piny o rozteči 2,54 mm. Je umístěn do DPS na výšku, ovšem modul je delší, než je výška krabičky D4MG. Bude proto pro přední panel potřeba vytvořit vlastní přední kryt, který poté umožní odhalit konektory pro připojení antén SIM modulu, které se nachází ve vyčnívající části DPS SIM modulu.



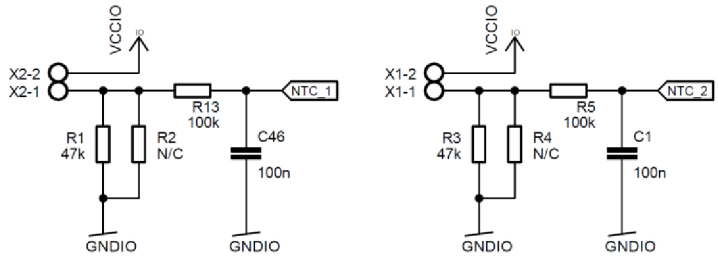
Obrázek 4.5 Schéma zapojení sběrnice UART

4.1.5 SPI periferie (OLED displej a microSD karta)

K mikrokontroleru jsou prostřednictvím SPI sběrnice připojeny 2 periferie – modul s OLED displejem a modul s mircoSD slotem pro microSD kartu. Obě periferie se k desce připojí pomocí pinů s roztečí 2,54 mm, u OLED displeje se jedná o piny prodloužené (délka 50 mm), aby byl displej přiblížen k přednímu panelu krabičky D4MG a čitelný tak pro uživatele. Volba periferie, se kterou se bude na sběrnici SPI komunikovat, se provádí piny *CS* jednotlivých modulů. Ty jsou připojeny a ovládány prostřednictvím GPIO mikrokontroleru.

4.1.6 Zapojení NTC termistorů

NTC termistory se ke komunikačnímu modulu připojují pomocí konektorů *X1* a *X2*. NTC poté s rezistorem *R1*, případně *R3*, tvoří napěťový dělič, ze kterého AD převodník mikrokontroleru měří napětí na rezistorech *R1* a *R3* a přepočítává naměřené napětí na teplotu.



Obrázek 4.6 Schéma zapojení NTC termistorů

Mezi rezistorovým děličem a analogovým vstupem převodníku je vložen RC filtr typu dolní propust, který filtruje nechtěné střídavé složky o frekvenci vyšší než 16 Hz. Je tvořen rezistorem s hodnotou odporu 100 kΩ a kondenzátorem o kapacitě 100 nF. Výpočet mezního kmitočtu je uveden v rovnici (4.6).

$$f_p = \frac{1}{2\pi RC} = \frac{1}{2\pi \cdot 100 \cdot 10^3 \cdot 100 \cdot 10^{-9}} = 15{,}92\ Hz, \tag{4.6}$$

Rezistorový dělič byl navržen pro měření teploty v rozsahu -25 až 25 °C (nepředpokládá se provoz autonomního systému v letním období) pomocí NTC termistoru TT02-10KC3-1D-T105-1500 [44]. Dělič je připojen na napětí V_{CCIO} (5 V). Termistor má hodnotu odporu 10 kΩ při teplotě 25 °C a parametr B = 3977 K. Při nejnižší hodnotě teploty (-25 °C) je hodnota odporu NTC termistoru rovna (využita rovnice (2.4)):

$$R = R_0 \cdot e^{\beta \cdot \left(\frac{1}{T} - \frac{1}{T_0}\right)} = 10 \cdot 10^3 \cdot e^{3977 \cdot \left(\frac{1}{248{,}15} - \frac{1}{298{,}15}\right)} = 147\ k\Omega\,, \tag{4.7}$$

kde teploty T a T_0 byly dosazeny v Kelvinech.

Pro využití většiny rozsahu 5 V napájení je vhodné použití rezistoru *R1*, resp. *R3*, s hodnotou odporu 47 kΩ. Krajní hodnoty napětí, které odpovídají teplotám -25 °C a 25 °C a budou snímány AD převodníkem, jsou uvedeny v rovnicích (4.8) a (4.9):

$$U_{-25°C} = U_{VCCIO} \cdot \frac{R_{1,3}}{R_{NTC} + R_{1,3}} = 5 \cdot \frac{47 \cdot 10^3}{147 \cdot 10^3 + 47 \cdot 10^3} = 1{,}2\ V, \tag{4.8}$$

$$U_{25°C} = U_{VCCIO} \cdot \frac{R_{1,3}}{R_{NTC} + R_{1,3}} = 5 \cdot \frac{47 \cdot 10^3}{10 \cdot 10^3 + 47 \cdot 10^3} = 4{,}12\ V\,, \tag{4.9}$$

Dělič, do kterého se připojuje NTC termistor, obsahuje navíc pozici pro rezistor R2, resp. R4. Tato pozice nemusí být obsazena, ale byla vyvedena na DPS v případě, kdy by bylo potřeba snížit chybu hodnoty odporu R1, resp. R3 vlivem tolerance rezistoru.

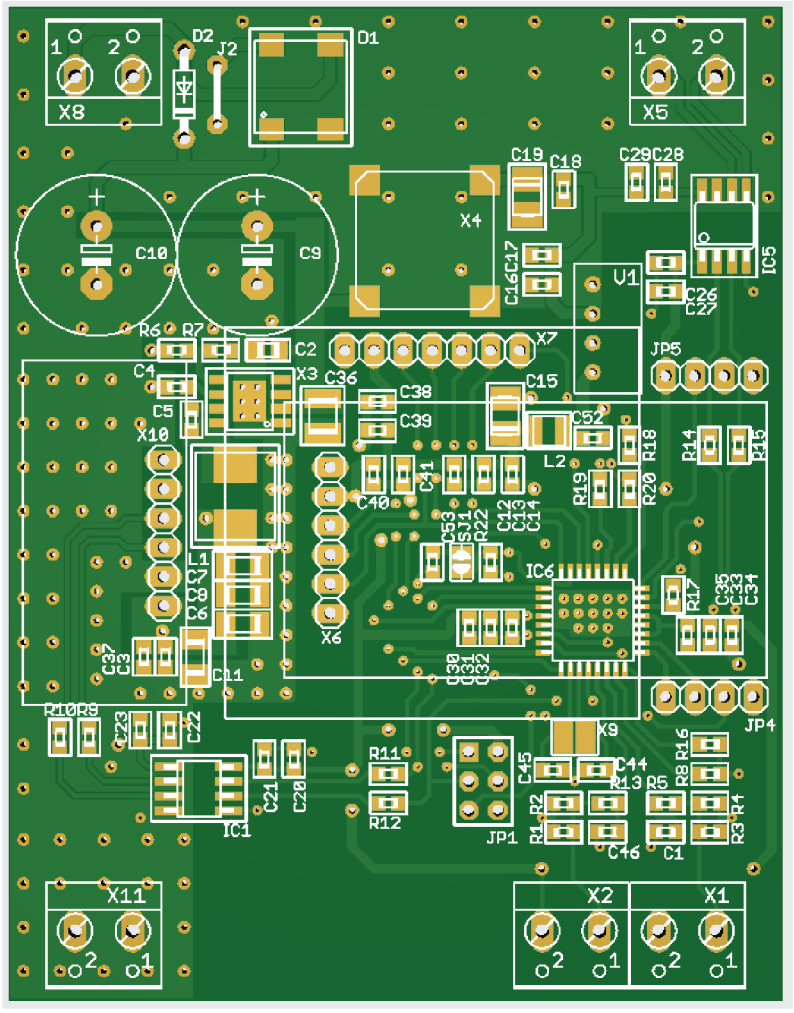
Výslednou hodnotu odporu paralelní kombinace R1 a R2 je následně nutné upravit v kódu mikrokontroleru komunikačního modulu pro správný přepočet napětí na teplotu.

4.1.7 Návrh desky plošných spojů (DPS) komunikačního modulu

Navržená DPS komunikačního modulu je uvedena na obrázku 4.7. DPS má rozměr daný výrobcem krabičky D4MG [35]. Na DPS topologie DC/DC měniče pro SIM modul dodržuje důležitá návrhová pravidla uvedená v katalogovém listu výrobce [36].

Pro vhodné propojení v rozvaděči s výkonovým modulem je konektor pro sběrnici CAN umístěn na horní straně DPS a společná neoddělená zem GND na spodní straně DPS. Konektor pro napájení je umístěno v levém horním rohu DPS a konektory pro NTC v pravém dolním rohu DPS.

DPS byla navržena podle návrhových pravidel výrobce DPS (rozlišení 0,15 mm).



Obrázek 4.7 DPS komunikačního modulu

4.2 Výkonový modul

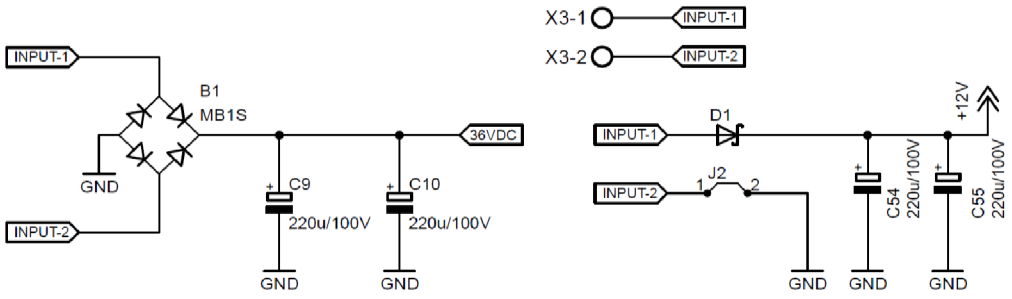
Elektronika výkonového modulu byla opět navržena pro zabudování do krabičky D4MG od společnosti GAINTA [35], tedy v rozvaděči obsazuje 4 modulové pozice. Logiku modulu obstarává opět mikrokontroler ATMega32M1 od společnosti ATMEL [11] se zabudovaným řadičem sběrnice CAN, který bude využit pro komunikaci s komunikačním modulem prostřednictvím CAN sběrnice.

Hlavní funkcí modulu je řízení spínání SSR relé a stykače pro vytápění kompozitních materiálů. Řízení je řešeno dvojicí MOSFET tranzistorů, které spínají napětí 12 V pro ovládání SSR relé a stykače řízeného stejnosměrným napětím 12 V, nebo optotriakem v případě využití stykače ovládaného střídavým napětím 24 V. Spínání tranzistorů je stejně jako spínání optotriaku odděleno opticky.

Pro měření teplot kompozitního materiálu se využívá trojice NTC termistorů, připojených prostřednictvím izolačních zesilovačů k AD převodníku mikrokontroleru. Pro měření proudového odběru při vytápění slouží Hallův snímač ACS758.

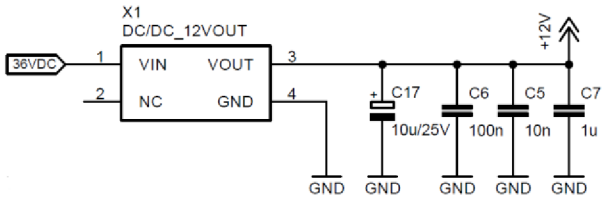
4.2.1 Systém napájení

Napájení výkonového modulu je řešeno obdobným způsobem jako u komunikačního modulu. Výkonový modul je možné opět napájet bezpečným střídavým napětím 24 VAC nebo stejnosměrným napětím 12 VDC. Stejně jako u komunikačního modulu je zde ochrana proti přepólování Schottkyho diodou v případě 12 VDC, převod 24 VAC na 36 VDC Graetzovým můstkem (rovnice (4.1)), nebo filtrace vstupního napětí dvěma kondenzátory 220 µF.



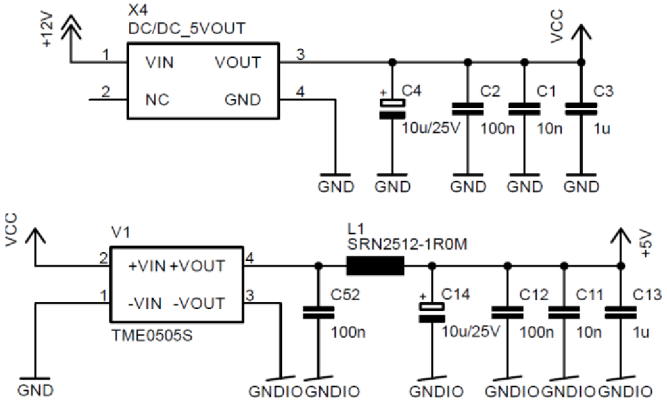
Obrázek 4.8 Schéma zapojení napájení výkonového modulu

V modulu se využívá 12 VDC pro spínání SSR relé nebo stykače řízený 12 VDC. Pokud je zapojeno napájení 24 VAC, 12 VDC větev vytváří Step-down DC/DC měnič CUI PXO7812-500-M-TR [39]. Pokud je zapojeno napájení 12 VDC, není potřeba pozici DC/DC měniče osazovat.



Obrázek 4.9 Schéma zapojení DC/DC měniče pro vytvoření 12 VDC větve

Neoddělenou V_{CC} napájecí větev s napětím 5 V vytváří step-down DC/DC měnič CUI PXO7805-500-M-TR [39]. V_{CC} napájecí větev se využívá pro napájení neoddělené strany izolačních zesilovačů, rezistorového děliče pro snímání teploty pomocí NTC, neoddělené stany vysílače/přijímače sběrnice CAN ISO1044, neoddělené části optočlenu ACPL-227 a jako vstupní napětí izolovaného DC/DC měniče TRACO TME0505S [40]. Tento měnič vytváří oddělenou větev *+5V*, která se využívá k napájení elektroniky v oddělené části.



Obrázek 4.10 Schéma zapojení DC/DC měničů pro vytvoření 5 V větví

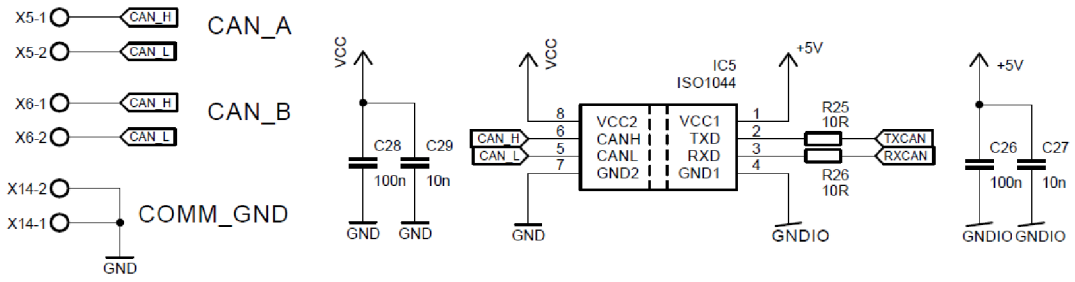
4.2.2 Mikrokontroler

Jelikož logiku výkonového modulu obstarává stejný mikrokontroler jako u komunikačního modulu, byl ve výkonovém modulu mikrokontroler zapojen stejně, tj. byl použit krystal NX3225GD [41] s dvojicí 12 pF kondenzátorů, stejná sada filtračních kondenzátorů, programování bude možné prostřednictvím 6 pinového konektoru AVR ISP a nevyužité piny jsou externími pulldown rezistory připojeny k zemi GND. Podrobnější popis je uveden v kapitole 4.1.2.

4.2.3 Sběrnice CAN

Komunikace prostřednictvím CAN je realizována obdobně jako u komunikačního modulu, tedy byl využit přijímač/vysílač ISO1044 od společnosti Texas Instruments [42] a filtrační kondenzátory 100 nF a 10 nF pro napájení oddělené i neoddělené části. Konektor pro kroucenou dvojlinku sběrnice CAN byl zdvojen pro snadnější propojení

modulů v rozvaděči, jelikož se předpokládá více připojených výkonových modulů k jednomu komunikačnímu. Konektory jsou na DPS označeny *X5* a *X6*. Konektor s označením *X14* slouží k vyrovnání potenciálů zemí mezi moduly, které je u sběrnice CAN vyžadováno.



Obrázek 4.11 Schéma zapojení sběrnice CAN výkonového modulu

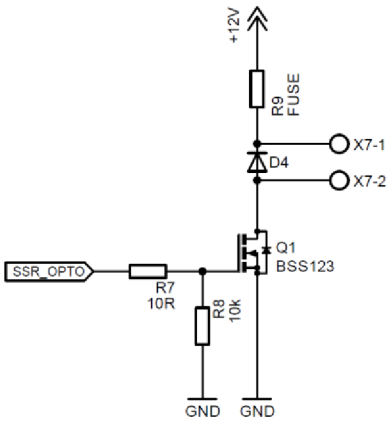
4.2.4 Měření proudu

Pro připojení vodičů určených k měření proudu slouží konektor Phoenix Contact PC 6/ 2-GU-7,62 s vyšší proudovou odolností až 41 A [45]. Samotné měření proudu zajišťuje Hallův snímač ACS758 od společnosti Allegro MicroSystems [7]. Je schopen měřit proudy do velikosti ±50 A a informaci převádí na velikost napětí v rozsahu 0-5 V s citlivostí 40 mV/A. Převedenou informaci poté vyhodnocuje AD převodník mikrokontroleru. Hallův snímač zároveň odděluje silovou část od elektroniky mikrokontroleru. Napájecí napětí snímače a napětí s výstupní informací jsou doplněny o filtrační kondenzátory 100 nF a 10 nF.

4.2.5 Spínání SSR relé a stykače

Jako SSR pro spínaní vytápění bylo zvoleno SSR relé Fotek-40 DA s maximálním řídícím proudem 20 mA a stejnosměrným řídícím napětím v rozsahu 3 až 32 V [46]. Pro SSR relé bylo zvoleno řízení z napěťové větve 12 VDC. Spínání řídícího proudu zajišťuje MOSFET tranzistor BSS123 od společnosti Onsemi, který umožňuje spínat maximální kontinuální proud 170 mA bez použití chladiče pro teplotu okolí 25 °C [47]. Vyšší teplota v rozvaděči se nepředpokládá, jelikož autonomní systém bude provozován pouze v zimním období.

Pro připojení řídícího napětí k SSR relé slouží konektor *X7*. Mezi výstupy konektoru je vložena paralelně v závěrném směru ochranná dioda *D4*. Před konektorem je také zapojena ochranná SMD pojistka v pouzdře SMD R1206.

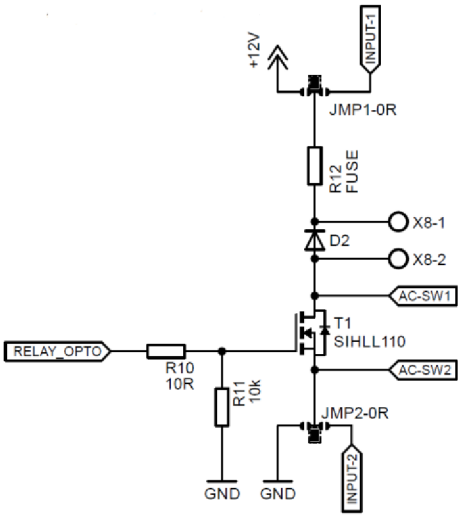


Obrázek 4.12 Schéma zapojení ovládání SSR relé

Výstup pro řízení stykače je navržen univerzálně jak pro stykače řízené napětím 12 VDC, tak napětím 24 VAC.

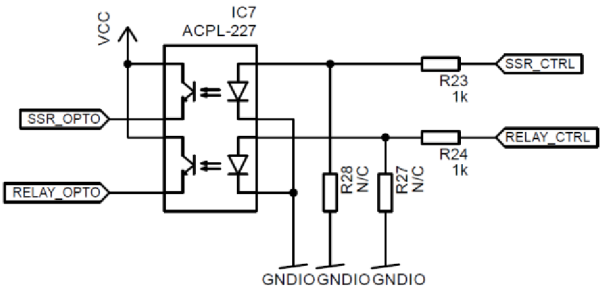
V případě použití stykače řízený 12 VDC je jeho ovládání řešeno obdobně jako u řízení SSR relé, ale MOSFET tranzistor BSS123 byl nahrazen MOSFET tranzistorem VISHAY SIHLL110TR-GE3 [48]. Ten umožňuje maximální kontinuální proud tranzistorem bez chladiče 1,5 A pro teplotu okolí 25 °C. Předpokládaný maximální řídící proud stykačů do 40 A (vzhledem ke konektoru určený pro měření proudu) je 300 mA, konkrétně vybraný stykač HAGER ESL225SDC, který bude v testovacím zapojení použit, má proudový odběr 183 mA [49]. Tranzistor tedy splňuje vyšší výkonové zatížení, než je zatížení v případě využití SSR relé.

Řídící obvod v případě využití 12 VDC řídícího napětí je opět doplněn o tavnou SMD pojistku v pouzdře SMD R1206 a ochrannou diodou D2, která zabraňuje zpětným proudovým rázům při rozpínaní stykače.



Obrázek 4.13 Schéma zapojení ovládání stykače

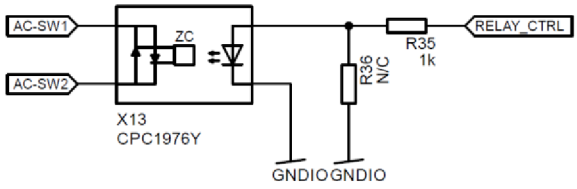
Řízení obou tranzistorů (BSS123 a SIHLL110) zajišťuje mikrokontroler přes optické oddělení pomocí optočlenu ACPL-227 od společnosti Broadcom [50].



Obrázek 4.14 Optické oddělení pro spínání MOSFET tranzistorů

V případě využití stykače řízeným napětím 24 VAC se na DPS nebude osazovat tranzistor *T1* (SIHLL110) a dioda *D2*, ale ke spínání řídícího obvodu se využije optotriaku CPC1976Y od společnosti IXYS. Optotriak umožňuje spínat napětí v rozsahu 20-240VAC a maximální proud 2 A. Má v sobě integrován systém spínaní v nule (zero-cross switching). [51] Řídící obvod bude nadále jištěn tavnou pojistkou v pouzdře SMD R1206.

Řídící napětí je potřeba zvolit SMD propojkami v pouzdře SMD R1206. Na DPS je u propojek pro přehlednost uvedena legenda.



Obrázek 4.15 Zapojení optotriaku

4.2.6 Zapojení NTC termistorů

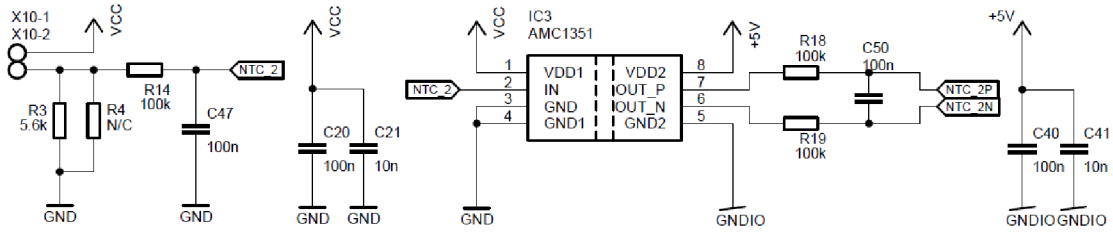
Pro měření teploty kompozitního materiálu budou využity stejné NTC termistory jako pro měření okolní teploty u komunikačního modulu. Budou zapojeny do stejných rezistorových děličů se stejným filtrem dolní propusti (viz kapitola 4.1.6). Změna zapojení oproti NTC termistorů u komunikačního modulu spočívá ve vložení izolačního zesilovače AMC1351 od spolčenosti Texas Instruments [52] před AD převodník mikrokontroleru.

Izolační zesilovač odděluje napětí na děliči od elektroniky, čímž plní funkci ochrany elektroniky v případě poruchy. Jelikož NTC s přívodními kabely budou zapuštěny v kompozitních materiálech, může nastat případ proniknutí napětí 24 VAC do elektroniky při poruše izolace přívodních kabelů NTC. Izolační zesilovač tak sníží při této poruše rozsah poškození elektroniky.

Použitý izolační zesilovač převádí napětí na vstupu *IN* na diferenciální pár *OUT_P* a *OUT_N* s fixním zesílením *A* = 0,4 (-). Diferenciální pár je poté přiveden přímo do AD převodníku mikrokontroleru, který umožňuje připojit a vyhodnotit až 3 diferenciální páry, proto modul umožňuje měřit teplotu kompozitu ve třech bodech objemu. Diferenciální páry jsou doplněny o filtry dolní propusti o mezní frekvenci dané rovnicí (4.10).

$$f_p = \frac{1}{2\pi R \cdot 2C} = \frac{1}{2\pi \cdot 100 \cdot 10^3 \cdot 2 \cdot 100 \cdot 10^{-9}} = 7{,}96\ Hz, \tag{4.10}$$

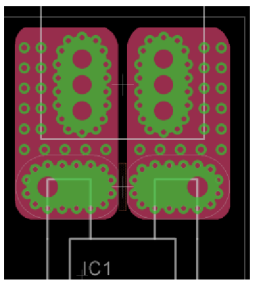
Napájení oddělené i neoddělené části izolačního zesilovače je doplněno o filtrační kondenzátory 100 nF + 10 nF.



Obrázek 4.16 Zapojení NTC termistoru k výkonovému modulu

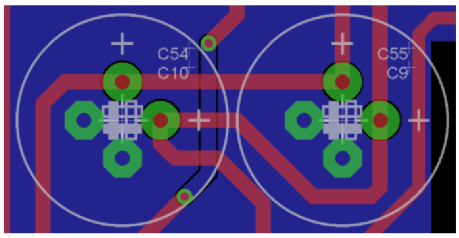
4.2.7 Návrh desky plošných spojů (DPS) výkonového modulu

Navržená DPS výkonového modulu je uvedena na obrázku 4.19. DPS má rozměr daný výrobcem krabičky D4MG [35]. Krabička bude upravena pro vložení zesíleného konektoru určeného k měření proudu. Ten je umístěn v pravém horním rohu DPS, společně s Hallovým snímačem ACS758. Trasa na DPS pro měření proudu je výkonové přizpůsobena.



Obrázek 4.17 Výkonové přizpůsobení vodivých cest pro měření proudu

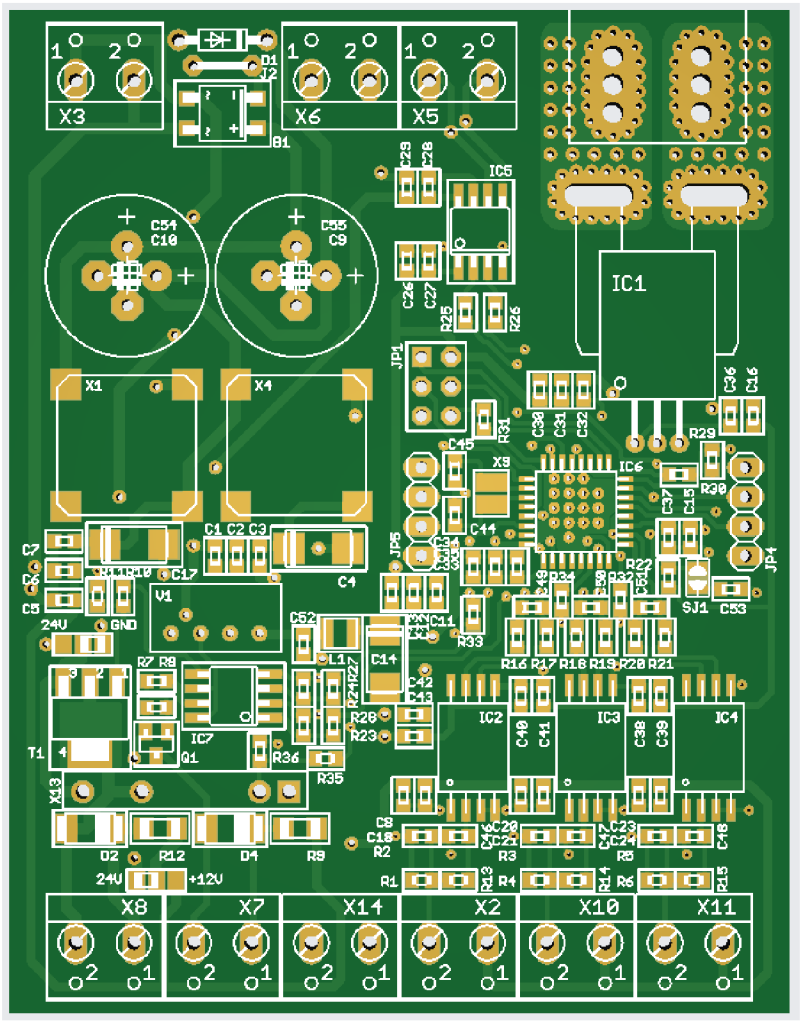
Pro optimalizaci místa na DPS byly využity stejné pozice kondenzátorů pro filtraci vstupního napětí jak z 12 VDC, tak 24 VAC. Jedná se o pozice C54 + C10 a C55 + C9. Pozice kondenzátorů jsou vůči sobě otočeny o 90°, osadí se podle zvoleného vstupního napětí.



Obrázek 4.18 Optimalizace umístění filtračních kondenzátoru vstupního napětí

Pro vhodné propojení v rozvaděči s komunikačním modulem a ostatními výkonovými moduly jsou konektory sběrnice CAN umístěn na horní straně DPS a společná neoddělená zem GND na spodní straně DPS. Konektor pro napájení je umístěno v levém horním rohu DPS, konektory pro NTC v pravém dolním rohu DPS a konektory pro výkonové spínací prvky v pravém dolním rohu DPS.

DPS byla navržena podle návrhových pravidel výrobce DPS (rozlišení 0,15mm).



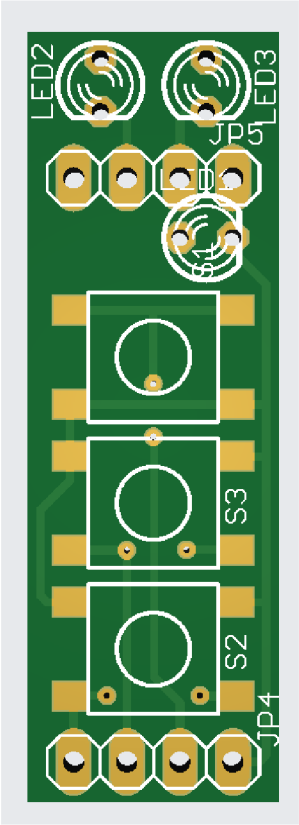
Obrázek 4.19 DPS výkonového modulu

4.3 Ovládací panel

Ovládací panel slouží k ovládání komunikačního a výkonového modulu na místě v rozvaděči. Obsahuje sadu 3 tlačítek a 3 LED. Jedna LED je využita pro indikaci zapnutí modulu, jedno tlačítko je využito k resetu modulu. Ostatní 2 LED a 2 tlačítka jsou určeny pro vlastní funkce, které může uživatel v budoucnu zavést.

4.3.1 Návrh desky plošných spojů (DPS) ovládacího panelu

DPS ovládacího panelu byla navržena univerzálně pro oba typy předchozích modulů. K DPS jednotlivých modulů se připojuje pomocí dvou 4 pinových konektorů s roztečí 2,54 mm, které jsou na opačných stranách DPS a zajišťují tak stabilitu ovládacího panelu v případě stisku tlačítek. K propojení budou využity prodloužené piny s délkou 50 mm. Návrh DPS ovládací desky je uveden na obrázku 4.20. DPS byla navržena podle návrhových pravidel výrobce DPS (rozlišení 0,15mm).



Obrázek 4.20 DPS ovládacího panelu

5. PROGRAMOVÉ VYBAVENÍ AUTONOMNÍHO SYSTÉMU VYTÁPĚNÍ

V této kapitole bude popsáno programové vybavení komunikačního a výkonového modulu autonomního systému. Jak již bylo zmíněno, logiku obou modulů zajišťuje mikrokontrolér ATMega32M1 [11] s integrovaným řadičem sběrnice CAN, která byla využita pro komunikaci mezi moduly. Programování mikrokontroleru bylo realizováno v jazyce AVR C a programovém prostředí ATMEL studio 7 [53]. Program je uveden v Příloha H - Elektronická příloha.

5.1 Kód mikrokontroléru komunikačního modulu

U komunikačního modulu se z periferií mikrokontroleru využívá AD převodníku, který jednobodovým měřením vyhodnocuje teplotu okolí ze 2 připojených NTC, komunikace UART pro zadávání příkazů SIM modulu a 16bitového čítače časovače (č/č 1) pro časování úkonů v hlavním programu. Zároveň byla připravena knihovna SPI pro zápis dat na SD kartu a komunikaci s OLED displejem pro budoucí implementaci periferií.

5.1.1 Měření okolní teploty

NTC termistory pro snímání teploty okolí se připojují do rezistorového děliče, ze kterého AD převodník snímá napětí na fixním rezistoru připojeným k zemi (rezistor má hodnotu odporu 47 kΩ, viz kapitola 4.1.6. Rozsah pro snímaní teplot byl nastaven na -25 až 25 °C. Pro tento rozsah teplot pak odpovídá rozsah elektrického odporu NTC čidel a rozsah napětí na rezistoru 47 kΩ uvedený v kapitole 4.1.6, tedy 10 až 147 kΩ a 1,2 až 4,12 V.

Rozsah digitálních kódů získaných z AD převodníku odpovídající rozsahu měřených teplot je pak dán rovnicemi (5.1) a (5.2):

$$ADC = \frac{U_{in} \cdot 1024}{U_{ref}} = \frac{1{,}2 \cdot 1024}{5} = 246\ (-)\ , \tag{5.1}$$

$$ADC = \frac{U_{in} \cdot 1024}{U_{ref}} = \frac{4{,}12 \cdot 1024}{5} = 844\ (-)\ , \tag{5.2}$$

Převod napětí na digitální kód zajišťuje funkce *adcRead(char channel)* v knihovně *adc.c*. V kódu byl rozsah výstupního kódu ořezán o 2 LSB na 0-255, aby byla informace uchována v 1 bytu. Po ořezání 2 LSB je pak rozsah digitálních kódů 61 až 211. Parametr channel udává kanál AD převodníku, v případě snímání okolní teploty z NTC termistorů jsou to kanály 2 a 3.

Vrácení teploty v odpovídajícím digitálním kódu zajištuje funkce *read_temp()* v knihovně *ntc.c*. Funkce postupně zjišťuje výsledný kód z každého kanálů AD převodníku a následně vrací průměrnou naměřenou hodnotu z obou NTC.

Při měření se zároveň zjišťuje, zda jsou připojeny NTC termistory k modulu (pokud ne, je na rezistoru 47 kΩ v rezistorovém děliči napětí 0 V, což se projeví jako kód 0 na výstupu AD převodníku. Funkce *read_temp()* tedy považuje za správný výsledek měření kód vyšší než 25 (zavedení tolerance). Pokud funkce vyhodnotí, že je výsledek nevyhovující, měření se opakuje s omezením na 3 pokusy.

Pokud je některý z NTC nepřipojen, nevrací funkce *read_temp()* průměrnou hodnotu z obou kanálu ADC, ale pouze hodnotu z kanálu, ke kterému je připojen NTC termistor. Pokud není ani jeden NTC termistor připojen, vrací funkce hodnotu 255.

5.1.2 Komunikace se serverem prostřednictvím SIM modulu

SIM modul umožňuje připojení a přenášení dat mezi autonomním systémem a API serverem prostřednictvím API metod, které jsou uvedeny v kapitole 7.4. Příkazy z mikrokontroléru do SIM modulu jsou odesílány prostřednictvím UART sběrnice.

UART sběrnice v ATMega32M1 umožňuje nastavit počet vzorků pro vyhodnocení 1 bitu při datovém přenosu (registr *LINBTR*). [11] V případě nastavení UART sběrnice pro SIM modul bylo zvoleno 8 vzorků na bit a nastavená přenosová rychlost je 9600 Baud. Pro nastavení přenosové rychlosti je potřeba do registru *LINBRR* nastavit hodnotu dělení hodinového signálu, která je dána vztahem [11]:

$$LINBRR = \frac{f_{clk}}{n_{vz} \cdot (BAUD+1)} = \frac{8 \cdot 10^6}{8 \cdot (9600+1)} = 104\,(-)\,, \tag{5.3}$$

kde f_{clk} označuje frekvenci hodinového signálu, n_{vz} počet vzorků na 1 bit a *BAUD* požadovanou přenosovou rychlost UART sběrnice.

Dále bylo potřeba zvolit full-duplex komunikaci v registru *LINCR* a povolení přerušení dokončení příjmu a odesílání v registru *LINENIR*. Všechny výše uvedeny nastavení se v hlavním programu provedou funkcí *uart_init()*.

Pro odeslání dat do SIM modulu je definována funkce *uart_transmit(char byte_data[])*. Ta odesílá prostřednictvím UART linky textový řetězec *byte_data[]* postupně po jednom znaku.

K přečtení dat z UART linky slouží funkce *char* uart_get_line()*. Ta využívá funkce *uart_char_receive()*, která postupně ukládá do zásobníku buffer jednotlivé znaky z odpovědi SIM modulu, a následně celou odpověď vrací jako textový řetězec.

Výše uvedené funkce pro nastavení UART sběrnice a odeslání či příjem dat na UART sběrnici jsou definovány v knihovně *uart.c*. Implementovány jsou pak v knihovně *simHTTP.c*, která definuje funkce pro připojení SIM modulu k internetu a API serveru a funkce pro odeslání či příjem dat ze SIM modulu.

K inicializaci SIM modulu slouží funkce *sim_init()*. Postup inicializace je uveden v kapitole 3.2.2 a Příloha A - Návod k využití AT příkazů. Při inicializaci SIM modul synchronizuje rychlost UART linky s mikrokontrolerem, připojí SIM kartu do mobilní sítě a k mobilnímu internetu, a nakonec nastaví technologii přenosu dat po LTE síti. Do

SIM modulu se vložila SIM karta od společnosti T-Mobile, která požaduje tyto přihlašovací údaje [54]:

- APN: internet.t-mobile.cz
- Přihlašovací jméno: ponechat prázdné
- Heslo: ponechat prázdné

Pro připojení autonomního systému k serveru slouží funkce *sim_connect()*, která volá HTTP API dotaz GET s názvem *Client/Connect*. Podrobnosti o URL adrese dotazu a odeslání doplňujících parametrů jsou uvedeny v kapitole 7.4. Sekvence pro vytvoření HTTP API dotazu GET ze SIM modulu je uvedena v Příloha A - Návod k využití AT příkazů.

Po připojení SIM modulu k HTTP API serveru je autonomní systém připraven opakovaně odesílat naměřená data a přijímat nové nastavení vytápění či novou předpověď počasí.

K přijetí nových parametrů vytápění a nové předpovědi počasí slouží funkce *sim_get_parameters()*. Ta volá HTTP API dotaz GET s názvem *Client/GetSettings* s parametrem názvu modulu. Podrobnosti o metodě jsou uvedeny v kapitole 7.4. Sekvence pro vytvoření HTTP API příkazu GET je opět uvedená v Příloha A - Návod k využití AT příkazů.

Přijaté hodnoty, které se nachází v odpovědi od serveru, se prostřednictvím CAN komunikace okamžitě předají výkonovým modulům. Definice formátu přijatých dat je uvedena v kapitole 5.3.

Pro odesílání naměřených dat na server slouží systému funkce *sim_upload_info()*. K odeslání dat se využívá HTTP API dotaz POST s názvem *Client/PostModuleInfo*. Data se posílají v těle HTTP požadavku. Sekvence příkazů pro SIM modul v případě volání HTTP API dotazu POST je uvedena v Příloha A - Návod k využití AT příkazů. Naměřená data se předávají ze systému v datovém řetězci, který má formát:

```
<kód_teploty_okolí>;<data_výkonový_modul_1>;<data_výkonový_modul_2>; … ;<data_výkonový_modul_n>
```

Kdy data z výkonových modulů jsou neupravená data převzaté z CAN komunikace (viz kapitola 5.3) a seřazeny za sebou do textového řetězce.

Do těla HTTP požadavku jsou data zapsána do proměnné *statusString* v JSON formátu:

```
{"statusString":"<kód_teploty_okolí>;<data_výkonový_modul_1>;<data_výkonový_modul_2>; … ;<data_výkonový_modul_n>"}
```

5.1.3 Hlavní program komunikačního modulu

Na začátku hlavního programu se inicializují potřebné periferie prostřednictvím definovaných funkcí *<periferie>_init()* v knihovnách dané periferie.

Pro časování hlavní smyčky programu byl využit čítač/časovač 1, který vytváří časové zpoždění 1 s. K dosažení tohoto zpoždění byla zvolena předdělička 1024 (nastaveno v registru *TCCR1B*). Počet impulzů, které je potřeba čítat k dosažení zpoždění 1 s, je (viz kapitola 2.1.6):

$$N = \frac{T}{t \cdot D} = \frac{T}{\frac{1}{f_{clk}} \cdot D} = \frac{1}{\frac{1}{8 \cdot 10^6} \cdot 1024} = 7813\ (-), \tag{5.4}$$

kde T označuje požadovanou dobu zpoždění, f_{clk} frekvenci hodinového signálu a D hodnotu nastavené předděličky čítač/časovače.

K vygenerování příznaku 1 s bylo využito přetečení čítače/časovače 1. K tomu je potřeba vždy na začátku nového čítání nastavit hodnotu čítacího registru *TCNT1* na hodnotu:

$$TCNT1 = 65535 - \mathrm{N} = 65535 - 7813 = 57722\ (-)\ , \tag{5.5}$$

Při přetečení čítače se spouští funkce *timer1_overflow_callback()*. Ta čítá postupně počet uplynutých sekund pomocí proměnné *sec_cnt*.

Dále funkce obsahuje 2 podmínky. První podmínka se využívá pouze při zapnutí autonomního systému, kdy po 5 s od zapnutí se komunikační modul připojuje k mobilnímu internetu a API serveru pomocí funkcí *sim_init()* a *sim_connect()* (kapitola 5.1.2). Po připojení se volá funkce *sim_get_parameters()*, která stáhne prvotní nastavení vytápění ze serveru, které se odešle se CAN komunikací do výkonových modulů.

Druhá podmínka ve funkci *timer1_overflow_callback()* kontroluje časové zpoždění 10 minut a využívá se opakovaně ve smyčce. Po uplynutí této doby funkce získává teplotu okolí z NTC, naměřená data z výkonových modulů a následně získaná data odesílá na server prostřednictvím funkce *sim_upload_info()*. Následně znovu stáhne nové nastavení vytápění funkcí *sim_get_parameters()* a parametry odešle do výkonových modulů. Komunikace mezi moduly je uvedena v kapitole 5.3.

Blokové schéma hlavního programu komunikačního modulu je uvedeno v Příloha F - Blokové diagramy kódů.

5.2 Kód mikrokontroléru výkonového modulu

U výkonového modulu se z periferií mikrokontroleru využívá AD převodníku, který jednobodovým měřením vyhodnocuje proud protékající do zátěže prostřednictvím proudového snímače ACS758, a také vyhodnocuje diferenciálně teplotu vytápěných kompozitních dílců z připojených NTC. Dále se využívá GPIO pro ovládání výkonových prvků a 16bitového čítače časovače (č/č 1) pro časování úkonů v hlavním programu.

5.2.1 Měření proudu

Funkce pro měření proudu se nachází v knihovně *acs758.c* jako *read_ac_current()*.

K zaznamenání aktuální hodnoty proudu se využívá funkce *adcRead(char channel)*, která zjišťuje prostřednictvím AD převodníku odpovídající napětí na výstupu proudového snímače ACS758 a převádí jej na digitální kód. Digitální kód má rozlišení 10 bitů, kód je ale ve funkci ořezán o 2 nejnižší bity. To umožňuje uchovávat informaci o naměřeném proudu v jednom bytu, čímž se ušetřily požadavky na obsazení paměti. Parametr *channel*

udává kanál AD převodníku, který se k měření využívá, v případě měření proudu se využívá kanál 7.

Jelikož se k vytápění využívá střídavé napětí 24 V s frekvencí 50 Hz, je vhodné pro uživatele uvést jako výsledek efektivní hodnotu protékajícího proudu. K výpočtu efektivní hodnoty proto funkce navzorkuje jednu periodu protékajícího proudu se zvolenou vzorkovací frekvencí 1 kHz. Ta zajistí v průběhu periody 20 vzorků protékajícího proudu (dáno rovnicí (5.6)).

$$n_{vz} = \frac{f_{vz}}{f_s} = \frac{1000}{50} = 20\ (-), \tag{5.6}$$

kde n_{vz} označuje počet vzorků, f_{vz} vzorkovací ferkvenci v Hz a f_s frekvenci vstupního proudu v Hz.

Počítadlo vzorků ve funkci zajišťuje cyklus *for*. Vzorkovací frekvence je zajištěna zpožděním o délce 1 ms. Vzorky jsou postupně ukládány do pole *samples[]*.

Po navzorkování jedné periody proudu je vypočítán ze vzorků kód průměrné hodnoty proudu (stejnosměrná složka, ideálně 0 A) a zjištěn kód odpovídající maximální hodnotě naměřeného proudu. Odečtením těchto dvou hodnot je určena amplituda proudu a převedena na hodnotu efektivní pomocí rovnice (5.7). Vypočtený kód efektivní hodnoty proudu je pak funkcí *read_ad_current()* vrácen.

$$I_{ef} = \frac{I_{max}}{\sqrt{2}}\ [\mathrm{A; A}], \tag{5.7}$$

kde I_{ef} označuje efektivní hodnotu proudu a I_{max} amplitudu měřeného proudu.

Jelikož se předpokládá, že průměrná hodnota proudu bude 0 A, bude vrácená hodnota efektivního proudu nabývat čísel 0-127. Proud 0 A totiž převodník ACS758 vyjadřuje jako napětí 2,5 V na výstupním pinu [7], které AD převodník vyhodnotí jako kód 128 v osmibitovém rozlišení (dáno rovnicí (5.8)), tedy jako 1 MSB. Odečtení 1 MSB od amplitudy pak zajistí, že rozlišení hodnoty efektivního proudu bude 7bitové.

$$ADC = \frac{U_{in} \cdot 256}{U_{ref}} = \frac{2{,}5 \cdot 256}{5} = 128\ (-), \tag{5.8}$$

Jelikož je závislost mezi měřeným proudem a zjištěným kódem lineární [7], lze převodní konstantu mezi efektivní hodnotou proudu a odpovídajícím digitálním kódem určit rovnicí (5.9):

$$k = \frac{I_{max}}{128} = \frac{50}{128} = 0{,}39\ A/d, \tag{5.9}$$

kde k je převodní konstanta mezi proudem a digitálním kódem a I_{max} je maximální hodnota proudu, kterou může snímač proudu ACS758 vyhodnotit. Tato konstanta je pro převod využita na API serveru v kontroléru *Client* (viz kapitola 7.4).

V rámci mikrokontroleru se v hlavním programu pracuje s vráceným kódem z AD převodníku, přepočet na proud v A je proveden až na API serveru (ušetřil se tak výpočetní výkon a paměť mikrokontroléru).

5.2.2 Snímání teplot kompozitních dílců

NTC termistory pro snímání teplot kompozitních dílců se připojují do rezistorového děliče, ze kterého AD převodník snímá napětí na fixním rezistoru připojeným k zemi (rezistor má hodnotu odporu 5.6 kΩ, viz kapitola 4.2.6. Rozsah pro snímaní teplot byl nastaven na -25 až 15 °C. Pro tyto teploty NTC termistorů odpovídá elektrický odpor čidel:

$$R = R_0 \cdot e^{\beta \cdot \left(\frac{1}{T} - \frac{1}{T_0}\right)} = 10 \cdot 10^3 \cdot e^{3977 \cdot \left(\frac{1}{288,15} - \frac{1}{298,15}\right)} = 15{,}9\ k\Omega\,, \tag{5.10}$$

$$R = R_0 \cdot e^{\beta \cdot \left(\frac{1}{T} - \frac{1}{T_0}\right)} = 10 \cdot 10^3 \cdot e^{3977 \cdot \left(\frac{1}{248,15} - \frac{1}{298,15}\right)} = 147\ k\Omega, \tag{5.11}$$

Rozsah napětí, které se vyskytne na odporu R 5.6 kΩ v odporovém děliči, je dán rovnicemi:

$$U_{-25°C} = U_{VCCIO} \cdot \frac{R}{R_{NTC}+R} = 5 \cdot \frac{5{,}6 \cdot 10^3}{147 \cdot 10^3 + 5{,}6 \cdot 10^3} = 0{,}183\ V, \tag{5.12}$$

$$U_{15°C} = U_{VCCIO} \cdot \frac{R}{R_{NTC}+R} = 5 \cdot \frac{5{,}6 \cdot 10^3}{15{,}9 \cdot 10^3 + 5{,}6 \cdot 10^3} = 1{,}3\ V\,, \tag{5.13}$$

Snímané napětí se k AD převodníku mikrokontroleru prostřednictvím izolačních zesilovačů, které převádí jednobodové měření na rozdílové napětí se zesílením A = 0,4 (-) (kapitola 4.2.6) [52]. Pro vyhodnocení diferenčního signálu byl zvolen v mikrokontroléru rozdílový zesilovač se zesílením *GAIN* = 5 (-) [11].

Maximální hodnotu kódu, kterou může AD převodník vrátit z diferenčního měření, je 511. Jelikož se nepředpokládá záporné napětí na rezistorovém děliči, které by způsobilo změnu polarity OUTP a OUTN, bude výstupní rozsah hodnot z AD převodníku 0-511, což vyjadřuje 9 bitů. Hodnota 511 se objeví v případě snímaného napětí na děliči (U_{in}) přibližně 1,25 V (dáno rovnicí (5.14) [11]), což odpovídá přibližně maximální hodnotě teploty z měřeného rozsahu, rozsah AD převodníku byl tedy plně využit.

$$ADC = \frac{(U_{in} - U_n) \cdot A \cdot GAIN \cdot 512}{U_{ref}} = \frac{(1{,}25 - (-1{,}25)) \cdot 0{,}4 \cdot 5 \cdot 512}{5} = 512\ (-), \tag{5.14}$$

Převod napětí na digitální kód zajišťuje funkce *adcReadDiff(char channel)* v knihovně *adc.c*. V kódu byl rozsah výstupního kódu ořezán o 1 LSB na 0-255, aby byla informace uchována v 1 bytu. Parametr *channel* udává kanál AD převodníku, v případě diferenčních vstupů odpovídá hodnotám 14, 15 a 16. V hlavičkovém souboru knihovny *adc.h* byly tyto hodnoty předefinovány vlastními názvy na *AMP0*, *AMP1* a *AMP2* pomocí direktivy *#define*.

Vrácení teploty v odpovídajícím digitálním kódu zajištuje funkce *read_temp()* v knihovně *ntc.c*. Funkce postupně zjišťuje výsledný kód z každého kanálů AD převodníku a následně je vrací v poli datového typu *unsigned char*.

Při měření se zároveň zjišťuje, zda jsou připojeny NTC termistory k modulu (pokud ne, je na rezistoru 5.6 kΩ v rezistorovém děliči napětí 0 V, což se projeví jako kód 0 na výstupu AD převodníku. Funkce *read_temp()* tedy považuje za správný výsledek měření kód vyšší než 10 (zavedení tolerance). Pokud funkce vyhodnotí, že je výsledek nevyhovující, měření se opakuje s omezením na 3 pokusy.

V rámci mikrokontroleru se v hlavním programu pracuje s vráceným kódem z AD převodníku, přepočet na teplotu ve °C je proveden až na API serveru (ušetřil se tak výpočetní výkon a paměť mikrokontroléru).

5.2.3 Vytápění kompozitních dílců

Výkonové prvky spíná prostřednictvím MOSFET tranzistorů nebo optotriaku GPIO mikrokontroleru. Pro dané piny (*PD7* a *PB2*) byla napsána knihovna *pwr_ctrl.c*, která obsahuje funkci *gpio_init()* pro inicializaci pinů jako výstupní pomocí *DDR* registrů a funkce pro ovládání SSR relé a stykače - *set_ssr(char state)* a *set_relay(char state)*. Proměnná *state* nastavuje stav výkonových prvků (log. 0 vypnuto, log. 1 sepnuto).

Pro samotné vytápění kompozitních dílců byly vytvořeny 3 módy vytápění: NORMAL, ECO a SMART. Ve všech 3 módech udržuje výkonový modul teplotu betonových dílců na nastavené požadované hodnotě s hysterezí ±1 °C.

V režimu NORMAL je spínání napětí pro vytápění realizováno stykačem. Při sepnutí a rozepnutí stykače se na 1 sekundu sepne SSR relé, aby přemostilo zátěž a eliminovalo tak opalování kontaktů, tedy prodloužilo životnost stykače.

V režimu ECO je vytápění eliminováno na polovinu maximálního výkonu. Využívá se zde pouze SSR relé, které při vytápění spíná zátěž pouze v polovině přednastavené periody (3 sekundy zapnuto, 3 sekundy vypnuto). Jelikož se jedná o časté spínání a rozpínání zátěže, není pro tuto aplikaci vhodný stykač.

Režim SMART vytápí obdobně jako režim NORMAL, ale vytápění spouští pouze v případě, že předpověď počasí na další hodinu hlásí teplotu pod 0 °C a srážky vyšší než 0.1 mm. Režim má tak zabránit vytváření námrazy na povrchu kompozitních dílců.

Vytápění a jeho režimy nastavuje funkce *heating_settings()*, definována v souboru hlavního programu *main.c*.

5.2.4 Hlavní program výkonového modulu

Na začátku hlavního programu se inicializují potřebné periferie prostřednictvím definovaných funkcí *<periferie>_init()* v knihovnách dané periferie.

Pro časování hlavní smyčky programu byl využit obdobně jako v případě komunikačního modulu čítač/časovač 1, který ovšem vytváří časové zpoždění 3 s. Nastavení generování zpoždění je stejné jako v případě komunikačního modulu (viz kapitola 5.1.3), pouze se musela upravit hodnota *TCNT1* registru, kterou je na počátku časování potřeba nastavit:

$$TCNT1 = 65535 - 3 \cdot \mathrm{N} = 65535 - 3 \cdot 7813 = 42096\ (-)\ , \qquad (5.15)$$

Při přetečení čítače se spouští funkce *timer1_overflow_callback()*, která obsahuje funkce, které se v hlavní smyčce programu opakují. Obsluhují měření teploty, měření proudu a nastavení vytápění pomocí výše uvedených funkcí. Zároveň z naměřených hodnot teplot vypočítávají průměrnou teplotu kompozitního dílce potřebnou pro nastavení vytápění a ukládají záznam o naměřeném proudu do pole naměřených hodnot. Z těch se následně vypočítává průměrný proudový odběr v případě odeslání dat komunikačnímu modulu (komunikace mezi moduly je popsána v kapitole 5.3).

Zároveň se v hlavním programu kontroluje, zda naměřený proud není vyšší než 32 A. Pokud ano, modul zahlásí chybový stav a odpojí spínací prvky od zátěže. Ke kontrole slouží funkce *overcurrent_flag(unsigned char current)*, která je definována v knihovně *acs758.c*.

Blokový diagram hlavního programu výkonového modulu je uvedeno v Příloha F - Blokové diagramy kódů.

5.3 Komunikace mezi moduly

Komunikace mezi výkonovým a komunikačním modulem je realizována sběrnicí CAN BUS. Zprávy na CAN BUS se předávají v tzv. MOB – Message object, kdy každý MOB definuje jeden typ zprávy. Pro komunikaci mezi výkonovým a komunikačním modulem bylo využito 5 MOB. Pro každý MOB byl definován název direktivou *#define* v hlavičkovém souboru knihovny CAN *canbus.h*. Jedná se o:

- *MOB0 – MOB_TOTAL_STOP*
- *MOB1 – MOB_ERROR*
- *MOB2 – MOB_ADDRESSING*
- *MOB3 – MOB_PARAMETERS*
- *MOB4 – MOB_DATA_REQUEST*

Při komunikaci mezi moduly má vždy MOB s nižším číslem vyšší prioritu než MOB s číslem vyšším, proto byly *MOB0* a *MOB1* vyhrazené pro chybové stavy. Prostřednictvím *MOB0* může komunikační modul okamžitě odpojit vytápění u všech výkonových modulů (odesláním zprávy *„TS“*). Prostřednictvím *MOB1* pak výkonové moduly hlásí, pokud se nachází v chybovém stavu. V poslední verzi softwaru je zaveden 1 chybový stav při proudovém přetížení (zpráva *„OC“*).

MOB2 slouží pro přiřezování adres výkonových modulů. Daný výkonový modul nejdříve zasílá zprávu *„RQ“*, na kterou dostane odpověď od komunikačního modulu, která obsahuje dostupnou adresu pro výkonový modul ze seznamu.

V *MOB3* předává komunikační modul výkonovým modulům parametry pro nastavení vytápění. Předává je hned po stažení parametrů z API serveru. Data jsou předávána v textovém řetězci o délce 5 bytů (znaků). Rozklíčování znaků v řetězci je uvedeno v tabulce 5.1.

Tabulka 5.1 Popis textového řetězce s parametry vytápění

Pořadí znaku	Název parametru	Možné hodnoty
1	Mód vytápění	0 – NOMRAL, 1 – ECO, 2 - SMART
2	Minimální požadovaná teplota	0-255, Kód požadované teploty snížené o 1 °C
3	Maximální požadovaná teplota	0-255, Kód požadované teploty zvýšené o 1 °C
4	Předpověď srážek	0 – Bez srážek, 1 – předpoklad srážek
5	Teplota z předpovědi počasí	0-255, Kód teploty z předpovědi počasí

V *MOB4* předávají výkonové moduly své adresy a naměřené hodnoty ze snímačů po přijetí požadavku o odeslání dat ze strany komunikačního modulu. Komunikační modul žádá o data zprávou *„RQ“*. Jakmile komunikační modul data obdrží, postupně jednotlivé odpovědi skládá za sebe do datového řetězce s oddělovačem *„;“*, na začátek navíc přiloží naměřenou teplotu okolí a následně datový řetězec předává na odeslání do SIM modulu. Data jsou mezi moduly předávána v textovém řetězci o délce 5 bytů (znaků). Rozklíčování znaků v řetězci je uvedeno v tabulce 5.2.

Tabulka 5.2 Popis textového řetězce s naměřenými daty výkonovým modulem

Pořadí znaku	Název parametru	Možné hodnoty
1	Adresa výkonového modulu	1-255
2	Proudová spotřeba	0-255, Kód průměrné proudové spotřeby
3	Teplota NTC1	0-255, Kód teploty naměřené snímačem NTC3
4	Teplota NTC2	0-255, Kód teploty naměřené snímačem NTC3
5	Teplota NTC3	0-255, Kód teploty naměřené snímačem NTC3

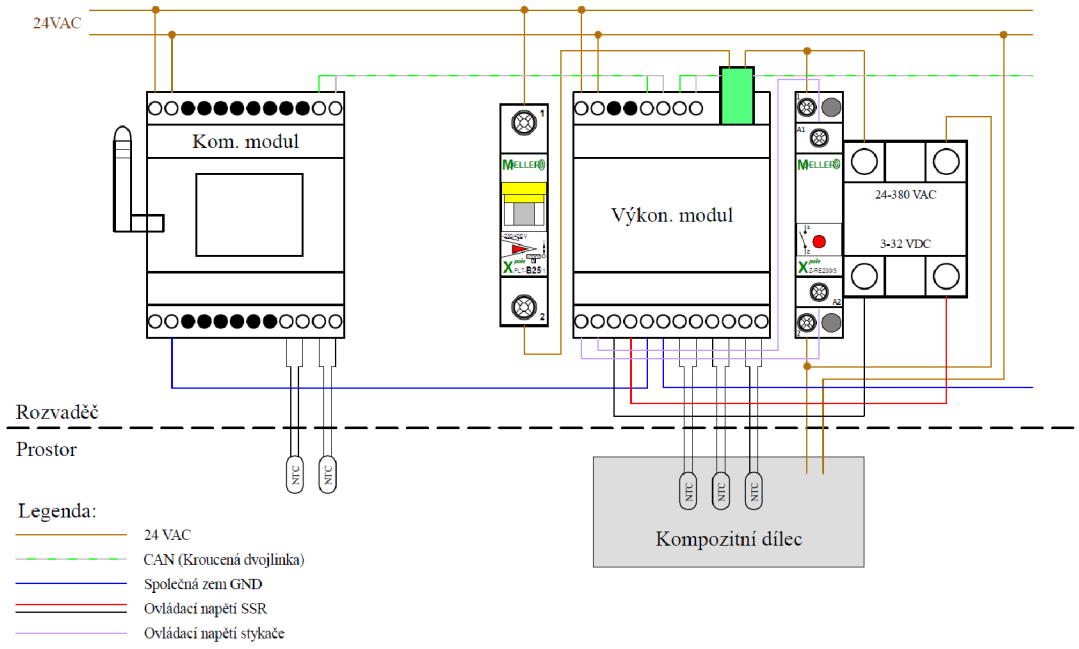
Pro inicializaci jednotlivých MOB, přerušení CAN sběrnice a rychlosti přenosu dat slouží funkce *can_init()* v knihovně *canbus.c*. V této knihovně je také definovaná funkce pro odeslání dat v MOB a nastavení MOB pro příjem zpráv. Přijatá zpráva pak generuje přerušení, při kterém se data zpracovávají ve funkci *canbus_receive_callback()*. Ta definuje zpracování dat pro každý MOB zvlášť podle výše uvedeného využití MOB.

Princip předávání dat v jednotlivých MOB je uveden v Příloha F - Blokové diagramy kódů.

6. PŘÍKLAD ZAPOJENÍ AUTONOMNÍHO SYSTÉMU

Na obrázku 6.1 je zobrazen příklad zapojení autonomního systému v rozvaděči. Jedná se o systém, který ovládá vytápění jednoho kompozitního dílce pomocí stykače řízeného napětím 24 VAC a SSR relé. Vytápění je jištěno proti přetížení a zkratu předřazeným jističem. Maximální proudové zatížení vytápění zátěže je tak omezeno těmito externími prvky, nebo maximální proudovou zatížitelností konektoru pro měření proudu nacházejícím se ve výkonovém modulu (41 A, kapitola 4.2.4). V případě, kdy by bylo potřeba ovládat vytápění s vyšší proudovou spotřebou, je možno měření proudu realizovat s pomocí přídavného proudového transformátoru.

Napájení komunikačního a výkonového modulu je v tomto případě zajištěno napětím 24 VAC. Pro názornost připojení dalších výkonových modulů s výkonovými prvky a jističi do autonomního systému jsou ve schématu vyvedeny vodiče sběrnice CAN a společné země GND. Zapojení NTC, výkonové a ovládací části je shodné. Schéma zapojení bylo vytvořeno v programu ProfiCAD [55].



Obrázek 6.1 Příklad zapojení autonomního systému v rozvaděči

7. ŘÍDÍCÍ API SERVER

Server využívá síťový protokol HTTP a architektonický styl MVC. API server je aplikace běžící na pozadí, tedy využívá pouze z MVC stylu komponenty model a kontrolér. Komponentu model používá server k definování struktur, do kterých se budou ukládat aktuální naměřená data, jejich historie a přijatá data z předpovědi počasí. Komponentu kontrolér využívá server pro definování API metod, konkrétně má definován 2 kontroléry – jeden je určený pro řízení a sběr dat autonomního systému a druhý je určen pro výměnu dat s ovládací aplikací. Server také obsahuje službu pro stažení předpovědi počasí na základě zeměpisných souřadnic lokace modulu.

Server byl vytvořen v prostředí Microsoft Visual Studio [56]. Soubory projektu jsou uvedeny v Příloha H - Elektronická příloha.

7.1 Modely

Modely jsou v projektu uvedeny ve složce Models. Vytvořené modely definují v programu vlastní datové typy.

Prvním definovaným modelem je *CommunicationModule*, který definuje strukturu dat týkající se jednoho připojeného komunikačního modulu. Parametry, které struktura obsahuje, jsou uvedeny v tabulce 7.1.

Tabulka 7.1 Definice parametrů komunikačního modulu

Název parametru	Datový typ	Definice
ModuleName	string	Název připojeného modulu
Longitude	float	Zeměpisná délka pozice modulu
Latitude	float	Zeměpisná šířka pozice modulu
Index	int	Přiřazené ID modulu po připojení k serveru
OutdoorTemp	int	Poslední naměřená hodnota teploty okolí
ForecastTemp	int	Teplota z předpovědi počasí
Forecast	string	Předpověď srážek
Cm_settings	CM_settings	Parametry vytápění zátěže (tabulka 7.2)
Pm_details	PM_details[]	Detaily z výkonových modulů (tabulka 7.3)
Material_parameters	Material_parameters	Detaily připojené zátěže (tabulka 7.4)

Tabulka 7.2 Definice parametrů vytápění CM_settings

Název parametru	Datový typ	Definice
HeatingAllowed	int	Příznak povolení vytápění
Mode	string	Volba módu vytápění
RequiredTemp	int	Hodnota požadované teploty materiálu

Tabulka 7.3 Definice detailů výkonových modulů PM_details

Název parametru	Datový typ	Definice
Id	int	Identifikátor výkonového modulu (CANBUS ID)
Current	float	Naměřená proudová spotřeba
Ntc_1	int	Naměřená teplota NTC termistorem 1
Ntc_2	int	Naměřená teplota NTC termistorem 2
Ntc_3	int	Naměřená teplota NTC termistorem 3
Error_code	int	Chybový kód modulu
Error_source	string	Detailní popis chyby

Tabulka 7.4 Definice detailů připojené zátěže Material_parameters

Název parametru	Datový typ	Definice
Volume	float	Objem materiálu
Density	float	Hustota materiálu
Spec_heat_capacity	float	Měrná tepelná kapacita materiálu
Heat_transfer_coef	float	Součinitel přestupu tepla

Druhý model *LogItem* definuje strukturu jednoho záznamu historie naměřených dat. Parametry, které každý záznam uchovává, jsou uvedeny v tabulce 7.5:

Tabulka 7.5 Parametry záznamu historie dat

Název parametru	Datový typ	Definice
DateTime	DateTime	Datum a čas vytvoření záznamu
ModuleName	string	Název modulu, který data odeslal
OutdoorTemperature	int	Naměřená venkovní teplota
RoadTemperature	int	Naměřená teplota silnice
Current	int	Naměřený proudový odběr

Model Weather data definuje strukturu dat, kterou API server přijímá data o předpovědi počasí. Podrobnější popis je uveden v kapitole 7.3.

7.2 Globální proměnné

Globální proměnné jsou uvedeny ve třídě *GlobalVariables.cs*. V aplikaci se využívají 3 globální proměnné:

- *availableIndex* – přiřazuje index nově připojenému modulu. Po připojení nového modulu se zvýší o 1. Je datového typu *int*.
- *commModules* – list, který uchovává nejnovější informace o připojených modulech. Je datového typu *List<CommunicationModule>* (tabulka 7.1).
- *log* – list, který uchovává historii záznamů o naměřených hodnotách teplot a proudového odběru z autonomního systému. Je datového typu *List<LogItem>* (tabulka 7.5).

7.3 Služba předpovědi počasí

Služba je vytvořena ve třídě *WeatherService.cs*. Předpověď počasí zjišťuje služba HTTP API dotazem na server *https://api.open-meteo.com/v1/forecast*. Do URL adresy jsou doplněny parametry, které požadují předpověď teplot následující 3 dny na základě zeměpisných souřadnic lokace modulu. [57] Parametry jsou uvedeny v tabulce 7.6.

Tabulka 7.6 Parametry dotazu o předpovědi počasí

Název parametru	Hodnota	Popis
latitude	*Dle modulu*	Zeměpisná délka pozice modulu
longitude	*Dle modulu*	Zeměpisná šířka pozice modulu
hourly	temperature_2m,precipitation	Požaduje se hodinová předpověď teplot a srážek
timezone	Europe%2fBerlin	Formát času předpovědi (UTC+01:00, Berlín)
forecast_days	3	Počet dní předpovědi

Pro získání hodnot velikosti srážek a teplot z předpovědi počasí slouží metoda *GetTemperatureForecastAsync()*. Ta zařizuje roztřídění teplot z přijatých dat do pole o vlastním datovém typu *FilteredWeatherDataModel* s parametry *int[] Temperatures* a *List<float> Precipitation*. Toto pole následně vrací jako odpověď.

7.4 Kontrolér Client

Kontrolér obsahuje 3 REST API metody, které jsou určeny pro komunikační moduly autonomních systémů.

První metoda má název *Connect* a slouží k připojení autonomního systému k serveru. Jedná se o metodu HTTP GET a požaduje parametry polohy zařízení a názvu připojeného modulu. Název může být zvolen libovolně, ovšem nesmí se shodovat s názvem jiného připojeného modulu, jelikož se používá k rozpoznání souvisejících nastavení a dat. Zároveň nesmí obsahovat znak ‚;‘, ten se používá k oddělování textů na API serveru. Poloha zařízení je potřebná ke stažení dat o předpovědi počasí. URL adresa dotazu je ve formátu:

```
http://<url_adresa_serveru>/Client/Connect?commModuleName=<nazev_modul
u>&latitude=<zemepisna_sirka>&longitude=<zemepisna_delka>
```

Druhou metodou je metoda *GetSettings*, která slouží systému pro stažení nového nastavení vytápění. Jedná se o metodu HTTP GET a jako parametr je potřeba uvést název modulu, podle kterého server vyhledá a odešle data z globální proměnné *commModules* (uvedeno v kapitole 7.2). URL adresa dotazu je ve formátu:

```
http://<adresa_serveru>/Client/GetSettings?commModuleName=<nazev_modul
u>
```

Data, které metoda odesílá, jsou textovém řetězci uvedeném v kapitole 5.3 a tabulce 5.1.

Poslední metodou kontroléru je metoda *PostModuleInfo*, která slouží systému pro odeslání naměřených dat na server. Formát dat, která odesílá systém, je uveden v kapitole

5.1.2. Data jsou odesílána v těle HTTP dotazu. Funkce přijatá data podle oddělovačů rozklíčuje a předá je do globální proměnné *commModules* (uvedeno v kapitole 7.2). Při každém přijetí dat navíc vytvoří záznam do historie měření (globální proměnné *log*, uvedeno v kapitole 7.2). URL adresa dotazu je ve formátu:

```
http://<url_adresa_serveru>/Client/PostModuleInfo?commModuleName=<nazev_modulu>
```

7.5 Kontrolér LabVIEW

Kontrolér obsahuje REST API metody, které jsou určeny pro nasazení do ovládacího panelu vytvořeném v prostředí LabVIEW. Metody se ovšem dají využít v jakémkoliv jiném programu, který je schopen volat API dotazy a rozklíčovat datovou strukturu JSON.

K přihlášení a odhlášení ovládacího panelu k serveru slouží metody typu GET s názvy *Connect()* a *Disconnect()*. K přihlášení je potřeba zadat ID ovládacího panelu. Tvar URL adres dotazu obou metod je:

```
http://<adresa_serveru>/LabVIEW/Connect?lvClient=<ID_ovladacího_panelu>
http://<adresa_serveru>/LabVIEW/Disconnect?lvClient=<ID_ovladacího_panelu>
```

Pro vypsání seznamu připojených autonomních systémů k API serveru slouží metoda typu GET *GetConnectedCommModules()*. Funkce vytvoří seznam názvů modulů z globální proměnné *commModules* (uvedeno v kapitole 7.2). Pro oddělení jednotlivých názvů modulů je využito znaku *„;“*. Tvar URL adresy dotazu je:

```
http://<adresa_serveru>/LabVIEW/GetConnectedCommModules
```

Informace o konkrétním autonomním systému vrací metoda *GetCommModuleInfo()*. Je typu GET a vyhledá záznam o komunikačním modulu v globální proměnné *commModules* (uvedeno v kapitole 7.2) podle názvu modulu a vrátí veškeré jeho aktuální informace v JSON formátu. Název modulu je nutné uvést v parametru *moduleName*. Struktura odeslaných dat je určená modelem *CommunicationModule* (uvedeno v kapitole 7.1). V případě nenalezení modulu vrátí chybovou hlášku *„Modul nebyl nenalezen“*. URL adresa dotazu je:

```
http://<adresa_serveru>/LabVIEW/GetCommModuleInfo?moduleName=<nazev_modulu>
```

Pro předání nového nastavení vytápění z ovládacího panelu na API server slouží metoda *UpdateCMSettings()*, která je typu POST. Parametry nastavení se odesílají v těle HTTP požadavku a musí být poslány ve struktuře JSON podle modelu *CM_settings* (uvedeno v kapitole 7.1). Do parametrů URL adresy se uvádí název komunikačního modulu, kterému se mají aktualizovat nastavení. URL adresa dotazu je ve tvaru:

```
http://<adresa_serveru>/LabVIEW/UpdateCMSettings?moduleName=<nazev_modulu>
```

Obdobným principem se aktualizují materiálové konstanty vytápěné zátěže. K tomu slouží POST metoda *SetMaterialParameters()*. Parametry se opět posílají v těle HTTP požadavku a definované struktury modelu *Material_parameters* (uvedeno v kapitole 7.1).

Do URL adresy je opět potřeba zapsat název komunikačního modulu. URL adresa dotazu je ve tvaru:

```
http://<adresa_serveru>/LabVIEW/SetMaterialParameters?moduleName=<naze
v_modulu>
```

Poslední metodou je GET metoda s názvem *GetLogFile()*. Tato metoda vrací historii naměřených dat v JSON struktuře jako list prvků datového typu *LogItem* (uvedeno v kapitole 7.1). URL adresa dotazu je ve tvaru:

```
http://<adresa_serveru>/LabVIEW/GetLogFile
```

7.6 Provoz vytvořeného serveru na Raspberry Pi

Pro provoz vytvořeného serveru byl využit malý jednodeskový počítač Raspberry Pi 4, uvedený na obrázku 7.1. Jeho operačním systémem je „*Raspberry Pi OS*“, který je založen na operačním systému „*Debian GNU/Linux*“. [58]

Pro nasazení bylo potřeba do operačního nainstalovat Docker a vytvořit kontejner s obrazem Ubuntu. Do kontejneru byl nainstalován ASP.NET Core runtime a byla vytvořena složka s aplikací serveru. Zároveň zde byl nainstalován reverzní proxy HTTP server Apache, který pomáhá efektivně spravovat API server.

Pro zpřístupnění aplikace byla využita služba Cloudflare, u které je potřeba mít zakoupenou svou doménu. Zároveň byla služba zabezpečena pomocí funkce „*Zero Trust*“, která umožňuje připojení k serveru pouze ověřeným uživatelům. [59]

Podrobný návod k zavedení aplikace na server je uveden v Příloha B - Postup nasazení ASP.NET Core aplikace na Raspberry Pi 4.

Obrázek 7.1 Raspberry Pi 4

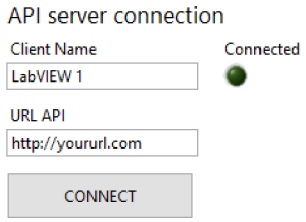
8. UŽIVATELSKÁ APLIKACE

Uživatelská aplikace byla vytvořena za účelem jednoduššího získání dat o autonomním systému ze serveru a zpřehlednění stažených dat pomocí grafických prvků. Zároveň umožňuje nastavení parametrů vytápění pro jednotlivé systémy. Aplikace je vytvořena v programovém prostředí LabVIEW [28] a uvedena v Příloha H - Elektronická příloha.

8.1 Vizuální část uživatelské aplikace

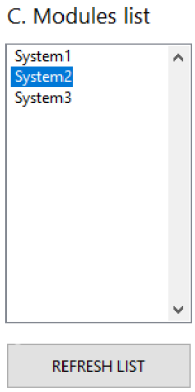
Hlavní obrazovka uživatelské aplikace je rozdělená do hlavních 3 částí – „*API server connection*“, „*Connected Modules list*“ a „*Module_details*“. Celkový snímek hlavní obrazovky je uveden v Příloha G - Vizuální část aplikace LabVIEW. Následující snímky obsahují v jednotlivých datových polích z důvodu bezpečnosti smyšlené hodnoty, které však názorněji zobrazují funkci aplikace.

První část „*API server connection*“ slouží pro uživatele k připojení na API server. Pro přihlášení k serveru zadává uživatel název klienta (pro rozpoznání aktivity na serveru) a URL adresu serveru. Stisknutím tlačítka „*CONNECT*“ se připojí k serveru. V případě úspěšného připojení signalizuje LED dioda „*Connected*“ rozsvícením úspěšné přihlášení k serveru a tlačítko „*CONNECT*“ je nahrazeno tlačítkem „*DISCONNECT*“ pro pozdější odhlášení od serveru.



Obrázek 8.1 Pole pro připojení aplikace k API serveru

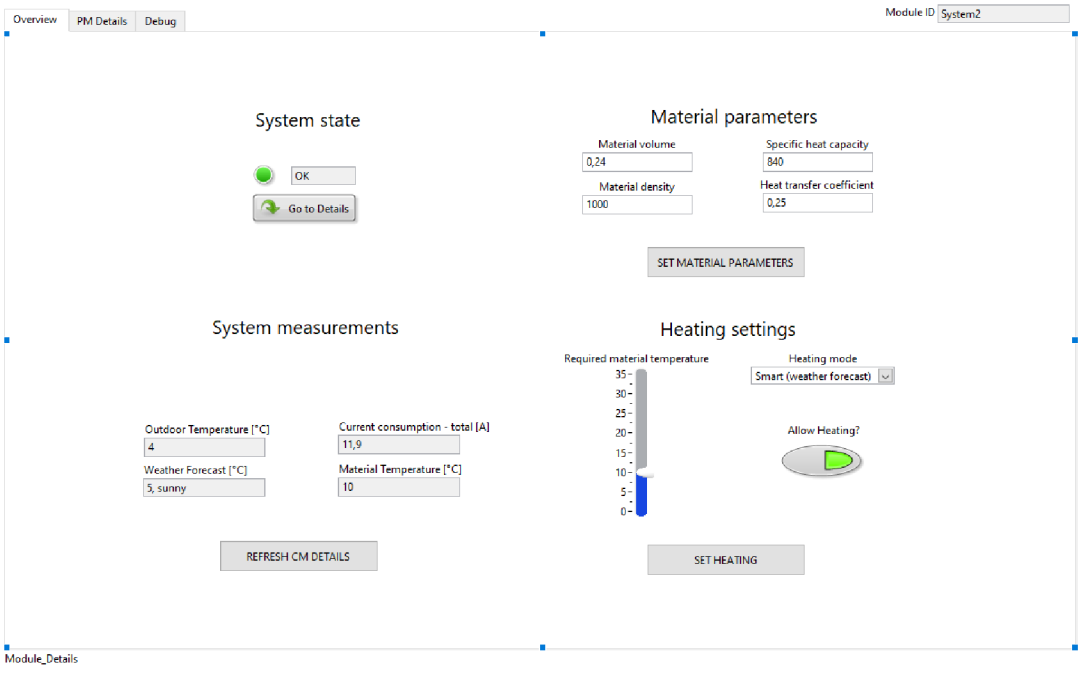
Část „*Connected Modules list*“ vypíše po připojení k serveru list komunikačních modulů, tedy list autonomních systémů, které jsou k API serveru připojeny. List lze aktualizovat tlačítkem „*REFRESH*“. V případě práce v aplikaci s některým systémem ze seznamu je potřeba dvojklikem aktivovat systém (identifikátor systému se vypíše do pole „*Module ID*“).



Obrázek 8.2 Seznam připojených autonomních systémů

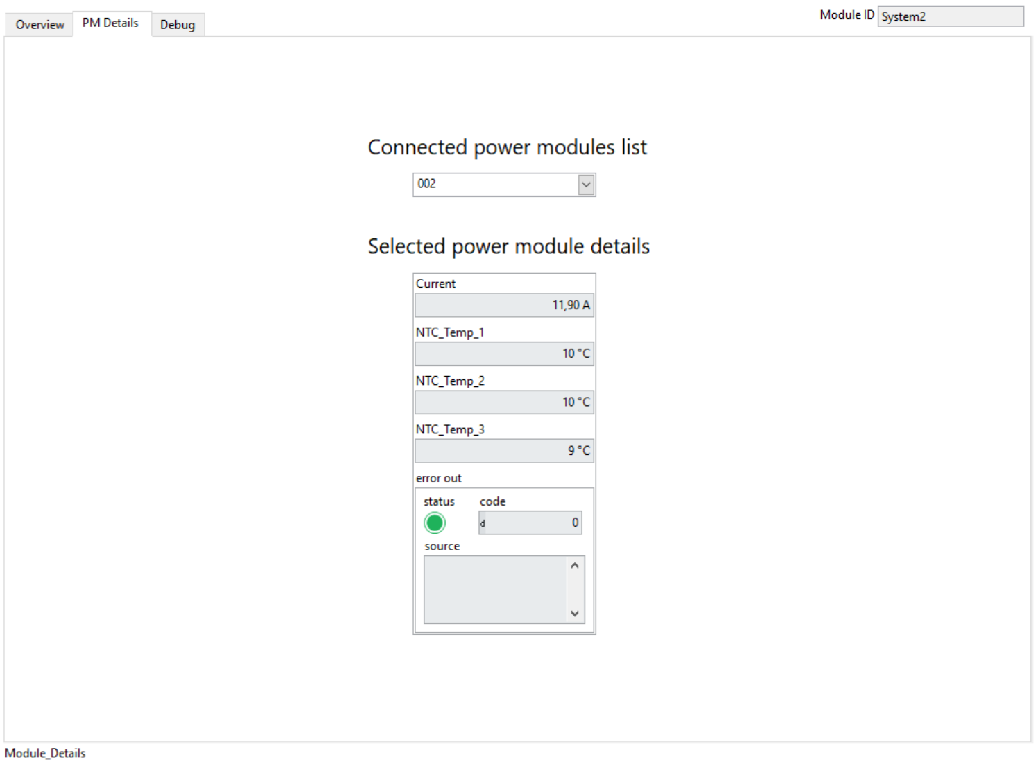
Po dvojkliku na autonomní systém ze seznamu se vypíší známá data o systému do odpovídajících polí v okně „*Module_details*“ (3. část hlavní obrazovky). Toto okno je pro přehlednost rozděleno do 3 záložek – „*Overview*“, „*PM Details*“ a „*Debug*“.

Záložka „*Overview*“ obsahuje přehledně nejdůležitější měřená data a pole pro nastavení parametrů vytápění a parametrů vytápěného materiálu. U každé části je příslušné tlačítko pro stažení dat ze serveru nebo aktualizace dat na server.



Obrázek 8.3 Přehled nejdůležitějších parametrů autonomního systému

Záložka „*PM details*“ obsahuje seznam připojených výkonových modulů ke komunikačnímu modulu daného systému. V seznamu jsou uvedeny identifikátory jednotlivých modulů na sběrnici CAN. Při zvolení jednoho z výkonových modulů se v okně „*Selected power modul details*“ vypíší podrobné detaily výkonového modulu.



Obrázek 8.4 Podrobný přehled informací výkonových modulů

Poslední záložka „*Debug*“ není určena pro uživatele, slouží pouze k nastavení správného formátu dat při komunikaci se serverem.

Doplňujícími prvky na hlavní obrazovce jsou indikátory „*State*“ a „*Response time*“, které udávají aktuální stav v programu a plynulost programu. Pro ukončení aplikace slouží tlačítko „*STOP*“.

8.2 Programová část uživatelské aplikace

Programová část je rozdělená do 3 hlavních částí – inicializace, hlavní program a ukončení. Ke komunikaci s API serverem byla využita LabVIEW knihovna *DataCommunication/Protocols/HTTPClient*. V této knihovně jsou předdefinovány funkce pro volání dotazů HTTP GET, které jsou zapotřebí ke stažení dat ze serveru, a HTTP POST, které jsou využívány pro nastavení nových parametrů vytápění a parametrů vytápěného materiálu.

V inicializaci se definují a nastavují proměnné do výchozích hodnot. Jedná se především o smazání dat ze seznamů, vynulování číselných ukazatelů a nastavení výchozí hodnoty booleovských indikátorů. Zároveň se definuje první stav hlavní smyčky programu.

Hlavní smyčka je řešená stavovým automatem se čtyřmi stavy – *Init*, *Login*, *Run* a *Logout*. Blokové schéma stavového automatu je uvedeno v Příloha F - Blokové diagramy kódů. Hlavní smyčka běží v taktu 100 ms.

Ve stavu *Init* se čeká, zda uživatel zadá příkaz k připojení na server. Pokud ano, přechází se do stavu *Login*, pokud ne, zůstává program ve stavu *Init*.

Ve stavu *Login* se uživatel přihlašuje názvem klienta k API serveru pomocí metody *LabVIEW/Connect*. Po úspěšném přihlášení pak aplikace automaticky stahuje seznam připojených komunikačních modulů k API serveru prostřednictvím metody *LabVIEW/GetConnectedCommModules*. Následně program přechází do stavu *Run*.

V tomto stavu může uživatel vybírat dvojklikem komunikační modul, se kterým chce v aplikaci pracovat. Po dvojkliku se automaticky volá dotaz na metodu *LabVIEW/GetCommModuleInfo*, který stahuje poslední známé informace o komunikačním modulu ze serveru. Pro jejich uchování byla v programu vytvořena globální proměnná o struktuře, která odpovídá modelu komunikačního modulu na API serveru. K proměnné se poté může přistupovat kdekoliv v programu. Identifikátor vybraného modulu se uloží do proměnné „*Module ID*“.

V rámci stavu Run se také obsluhují jednotlivá funkční tlačítka. Jedná se o tlačítka:

- *REFRESH*: aktualizuje seznam připojených komunikačních modulů k API serveru prostřednictvím metody *LabVIEW/GetConnectedCommModules*
- *REFRESH CM DETAILS*: aktualizuje informace o vybraném komunikačním modulu prostřednictvím metody *LabVIEW/GetCommModuleInfo*
- *SET MATERIAL PARAMETERS*: nastaví parametry vytápěného materiálu prostřednictvím metody *LabVIEW/SetMaterialParameters*
- *SET HEATING*: nastaví parametry vytápění prostřednictvím metody *LabVIEW/UpdateCMSettings*

Ve stavu *Run* je také možné zobrazit podrobné informace o jednotlivých výkonových modulech v záložce „*PM Details*“ okna „*Module_details*“. Podrobné informace se vypíšou při výběru jednoho z výkonových modulů ze seznamu „*Connected power modules list*“.

Ve stavu *Run* aplikace zůstává do doby, než se uživatel odpojí od API serveru stiskem tlačítka *DISCONNECT*. Poté přechází aplikace do stavu *Logout*, kde nastavuje indikátory parametrů do výchozích hodnot a odhlašuje se od API serveru pomocí metody *LabVIEW/Disconnect*. Následně aplikace přechází zpět do stavu *Init* a je připravena se připojit k jinému API serveru.

Hlavní smyčku lze ukončit tlačítkem *STOP*. Při zastavení hlavní smyčky se automaticky odpojí klient od API serveru (metoda *LabVIEW/Disconnect*) a nastaví se opět indikátory parametrů do výchozích hodnot. Aplikace je poté automaticky ukončena.

9. ZÁVĚR

Hlavním cílem diplomové práce bylo navrhnout autonomní systém pro vytápění ploch vytvořených z vodivého kompozitního materiálu. Systém reguluje vytápění plochy na základě zpětné vazby o její teplotě. Zároveň je systém využitelný v průmyslu 4.0, jelikož umožňuje připojení k mobilnímu internetu prostřednictvím sítě GSM, čímž umožňuje i vzdálené ovládání modulu či vzdálený monitoring dat.

Navržený autonomní systém vytápění se skládá ze 2 typů modulů – komunikačního a výkonového modulu. Komunikační modul představuje vstupně/výstupní bránu systému, ke které se připojují výkonové moduly, které zajišťují vytápění dílců. Komunikace mezi moduly zprostředkovává sběrnice CAN. Moduly mohou být napájeny stejnosměrným napětím 12 V, nebo střídavým 24 V.

Vytápění kompozitních dílců je realizováno bezpečným střídavým napětím 24 V. Každý výkonový modul ovládá SSR relé a stykač, které toto napětí spínají a přivádí jej tak ke kompozitním dílcům. Vytápění je regulováno zpětnou vazbou o teplotě dílců, kdy se pro měření teplot využívá až 3 NTC termistorů.

Výkonový modul zároveň umožňuje měřit proudovou spotřebu při vytápění dílců. Jako snímač proudu je použit Allegro ACS758, který snímá proud pomocí Hallova jevu. Pro připojení výkonové částí ke snímači je využit konektor Phoenix Contact PC6/2-GU-7,62, který má maximální proudové zatížení 41 A.

Maximální proudové zatížení, které může přímo ovládat výkonový modul, je dáno maximálním povoleným zatížením spínacích prvků a konektoru pro měření proudu. V softwaru je zavedená softwarová pojistka proti proudovému přetížení a zkratu nad 32 A. Přesto je doporučeno spínanou zátěž jistit přídavným jističem.

V případě, kdy by bylo požadováno připojení proudové zátěže pro jeden výkonový modul vyšší než 32 A, je možno měření proudu realizovat pomocným měřicím transformátorem proudu. Při vyšším proudovém zatížení se ovšem doporučuje realizovat spínání stykače prostřednictvím pomocného relé, jelikož stykače pro vyšší proudové zatížení vyžadují vyšší ovládací proud (výkonový modul limituje ovládací proud stykače na max. 1,5 A při použití 12 VDC stykače nebo 2 A při použití 24 VAC stykače).

Komunikační modul zajišťuje sběr dat z výkonových modulů prostřednictvím sběrnice CAN a měří okolní teplotu v místě vytápěné plochy. Byl vybaven SIM modulem Waveshare SIM7000E NB-IOT HAT, který umožňuje připojení autonomního systému k síti GSM, a tím i k mobilnímu internetu. SIM modul je ovládán z mikroprocesoru komunikačního modulu prostřednictvím sběrnice UART a AT příkazů.

Systém může vytápět ve 3 módech: NORMAL, ECO a SMART. V režimu NORMAL není vytápění nijak výkonově omezeno, pouze reguluje teplotu kompozitních dílců na požadovanou teplotu s přednastavenou hysterezí v programu komunikačního modulu. V tomto módu se využívá kombinace stykače a SSR relé zároveň.

V případě módu ECO je vytápění omezeno na polovinu/třetinu maximálního výkonu. V tomto módu se využívá pouze SSR relé, jelikož se při vytápění spíná napětí pouze v polovině/třetiny přednastavené periody. Jedná se tak o časté spínání, čímž by se životnost stykače výrazně snížila.

V režimu SMART se využívá předpovědi počasí. Mód je určen pro provoz v zimním období a má zabránit namrzání vytápěných ploch. Vytápění se tedy spouští pouze v případě kombinace srážek a nízkých teplot. Režim není výkonově omezen a opět využívá kombinace stykače a SSR relé.

Pro vzdálené nastavení parametrů vytápění a sběr dat z autonomního systému byla navržena architektura, ve které je jádrem HTTP API server s vlastními definovanými API metodami. Server je připojen k internetu a je navržen pro připojení více autonomních systémů vytápění najednou. Uchovává si aktuální stav připojených systémů, historii naměřených dat a zjišťuje předpověď počasí na další 3 dny.

API metody na serveru jsou rozděleny do dvou skupin – kontrolérů. První kontrolér s názvem *Client* definuje API metody pro komunikaci autonomního systému se serverem. Mezi ně patří metoda pro připojení systému k API serveru, metoda pro odeslání naměřených dat na server a metoda pro přijetí nového nastavení vytápění ze serveru. Autonomní systém pak volá jednotlivé metody pomocí HTTP dotazů a mobilního internetu. Dotazy pro stažení nových parametrů vytápění a pro odeslání naměřených dat volá opakovaně po 10 minutách.

Druhý kontrolér s názvem *LabVIEW* definuje API metody pro komunikaci mezi ovládací aplikací a API serverem. Formát předávaných dat mezi aplikací a serverem je typu JSON a jejich struktura je definována modely na API serveru.

Ovládací aplikace usnadňuje uživateli nastavení připojených autonomních systému k API serveru a přehlednou vizualizaci naměřených dat a detailů z jednotlivých modulů. Aplikace byla vytvořena v programovém prostředí LabVIEW.

Pro navržený autonomní systém byly vytvořeny 2 prototypové moduly, 1 výkonový a 1 komunikační. Ty nemohly být vzhledem k časovým podmínkám otestovány v zimním období. Bylo by tedy vhodné v tuto dobu prototyp odzkoušet a doladit režim SMART, stejně jako další případné nedostatky. Fotografie výsledných modulů jsou uvedeny v Příloha E - Fotodokumentace elektroniky systému.

LITERATURA

[1] IBM. *What is the internet of things?* Online. Dostupné z: https://www.ibm.com/topics/internet-of-things. [cit. 2024-01-02].

[2] TWI LTD. *What is a Smart City? – Definition and Examples*. Online. Dostupné z: https://www.twi-global.com/technical-knowledge/faqs/what-is-a-smart-city. [cit. 2024-01-02].

[3] GILLIS, Alexander S. *What is IoT (Internet of Things) and How Does it Work?* Online. TechTarget. 2023, August 2023. Dostupné z: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT. [cit. 2024-01-02].

[4] TECHSLANG. *What is a Backend System?* Online. Dostupné z: https://www.techslang.com/definition/what-is-a-backend-system/. [cit. 2024-01-02].

[5] MONOLITHIC POWER SYSTEMS. *Hall Effect Sensors: A Comprehensive Guide*. Online. Dostupné z: https://www.monolithicpower.com/en/hall-effect-sensors-a-comprehensive-guide. [cit. 2024-05-06].

[6] ELECTRONICSTUTORIALS. *Hall Effect Sensor*. Online. Dostupné z: https://www.electronics-tutorials.ws/electromagnetism/hall-effect.html. [cit. 2024-05-06].

[7] ALLEGRO MICROSYSTEMS, 2022. *ACS758xCB*. Online. ACS758-DS, Rev. 18 MCO-0000203. Dostupné z: https://www.allegromicro.com/en/products/sense/current-sensor-ics/fifty-to-two-hundred-amp-integrated-conductor-sensor-ics/acs758. [cit. 2024-05-06].

[8] WAVELENGTH ELECTRONICS. *What is a thermistor?* Online. Dostupné z: https://www.teamwavelength.com/thermistor-basics/. [cit. 2024-05-06].

[9] NORTH STAR SENSORS, LLC. *Calculating Temperature from Resistance.* Online. Dostupné z: https://www.northstarsensors.com/calculating-temperature-from-resistance. [cit. 2024-05-06].

[10] AYALA, Kenneth J.; a BARILE, George, STENDAHL, Roslyn (ed.), 1991. *The 8051 Microcontroller*. Online. United States of America: WEST PUBLISHING COMPANY. ISBN 0-314-77278-2. Dostupné z: https://buddhiprakash.weebly.com/uploads/4/5/3/2/45327319/8051microcontroller-ayala.pdf. [cit. 2024-05-06].

[11] ATMEL, 2015. *ATmega16M1/ATmega32M1/ATmega64M1/ ATmega32C1/ATmega64C1 Automotive*. Online. Rev.: 7647O–AVR–01/15. Dostupné z: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7647-Automotive-Microcontrollers-ATmega16M1-32M1-64M1-32C1-64C1_datasheet.pdf. [cit. 2024-05-06].

[12] SMITH, Grant Maloy, 2024. *What Is CAN Bus (Controller Area Network) and How It Compares to Other Vehicle Bus Networks*. Online. DEWESOFT.

Dewesoft. Tuesday, February 13, 2024. Dostupné z: https://dewesoft.com/blog/what-is-can-bus. [cit. 2024-05-06].
[13] SHELDON, Robert, 2023. *What is a serial peripheral interface (SPI)?* Online. Techtarget. August 2023. Dostupné z: https://www.techtarget.com/whatis/definition/serial-peripheral-interface-SPI. [cit. 2024-05-06].
[14] *The SPI Bus*. Online. RealDigital. Dostupné z: https://www.realdigital.org/doc/6c55fef7bba4a22ff35dce46a3c359af. [cit. 2024-05-06].
[15] ROHDE & SCHWARZ USA, INC. *Understanding UART*. Online. Dostupné z: https://www.rohde-schwarz.com/us/products/test-and-measurement/essentials-test-equipment/digital-oscilloscopes/understanding-uart_254524.html. [cit. 2024-01-02].
[16] GRACE LEGASPI, Mary a PEŇA, Eric. *UART: A Hardware Communication Protocol, Understanding Universal Asynchronous Receiver/Transmitter*. Online. ANALOG DEVICES, INC. Analog Dialogue. 2020, DEC 2020. Dostupné z: https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html. [cit. 2024-01-02].
[17] NDUNGU, Solomon a MIXON, Erica. *What is GSM (Global System for Mobile communication)?* Online. TechTarget. 2021, March 2021. Dostupné z: https://www.techtarget.com/searchmobilecomputing/definition/GSM. [cit. 2024-01-02].
[18] FCC PRŮMYSLOVÉ SYSTÉMY S.R.O. *Pásma LTE/UMTS/EDGE/GSM používaná v České republice*. Online. FCC průmyslové systémy. Dostupné z: https://www.fccps.cz/pasma-lteumtsedgegsm-pouzivana-v-ceske-republice-1379. [cit. 2024-01-02].
[19] ZOLA, Andrew. *What is GPRS (General Packet Radio Services)?* Online. TechTarget. 2021, March 2021. Dostupné z: https://www.techtarget.com/searchmobilecomputing/definition/GPRS. [cit. 2024-01-02].
[20] THALES. *Generations of mobile networks explained*. Online. Thales Group. 2023, December 2023. Dostupné z: https://www.thalesgroup.com/en/markets/digital-identity-and-security/inspired/basics-of-mobile-networking/milestones. [cit. 2024-01-02].
[21] MICROSOFT. *What is ASP.NET? | .NET*. Online. Dostupné z: https://dotnet.microsoft.com/en-us/learn/aspnet/what-is-aspnet. [cit. 2024-01-02].
[22] SHELDON, Robert. *What is model-view-controller (MVC)?* Online. TechTarget. 2023, September 2023. Dostupné z: https://www.techtarget.com/whatis/definition/model-view-controller-MVC. [cit. 2024-01-02].

[23] THELIN, Ryan. *What are REST APIs? HTTP API vs REST API*. Online. Educative. 2021, May 19, 2021. Dostupné z: https://www.educative.io/blog/what-are-rest-apis. [cit. 2024-01-02].
[24] GUPTA, Lokesh. *HTTP Methods*. Online. 2023, November 4, 2023. Dostupné z: https://restfulapi.net/http-methods/. [cit. 2024-01-04].
[25] RED HAT. *What is the REST API?* Online. 2020, May 8, 2020. Dostupné z: https://www.redhat.com/en/topics/api/what-is-a-rest-api. [cit. 2024-01-02].
[26] *JSON Basics*. Online. Opentext. Dostupné z: https://www.microfocus.com/documentation/silk-performer/195/en/silkperformer-195-webhelp-en/GUID-91180EA7-4C8E-4957-90D5-7551B5548A63.html. [cit. 2024-01-02].
[27] GUPTA, Lokesh. *HTTP Status Codes*. Online. 2023, November 4, 2023. Dostupné z: https://restfulapi.net/http-status-codes. [cit. 2024-01-02].
[28] NATIONAL INSTRUMENTS. *LabVIEW*. Online. Dostupné z: https://www.ni.com/en/support/downloads/software-products/download.labview.html#521715. [cit. 2024-05-06].
[29] WAVESHARE. *SIM7000E NB-IoT HAT User Manual*. Online. Rev1.0. June 8, 2018. Dostupné z: https://www.waveshare.com/w/upload/7/76/SIM7000E-NB-IoT-HAT-Manual-EN.pdf. [cit. 2024-01-02].
[30] SIMCom. *SIM7000 Series_HTTP_Application Note*. Online. Version 1.01. 2017. Dostupné z: https://www.waveshare.com/w/upload/5/57/SIM7000_Series_HTTP_Application_Note_V1.01.pdf. [cit. 2024-01-02].
[31] SIMCom. *SIM7000 Series UART Application Note_V1.00*. Online. Version 1.00. 2017. Dostupné z: https://simcom.ee/documents/SIM7000x/SIM7000%20Series%20UART%20Application%20Note_V1.00.pdf. [cit. 2024-01-02].
[32] CONEL. *AT commands APLICATION NOTE*. Online. 2012. Dostupné z: https://www.induo.com/wp-content/uploads/2014/03/at-kommandon-conel-router.pdf. [cit. 2024-01-02].
[33] *HTTP testing with Simcom SIM7000 and SIM800 Modules*. Online. M2MSupport.net. Dostupné z: https://m2msupport.net/m2msupport/http-testing-with-simcom-sim7000-and-sim800-modules/. [cit. 2024-01-02].
[34] AUTODESK. *Autodesk Eagle: PCB design made easy for every engineer*. Online. Dostupné z: https://www.autodesk.com/products/eagle/overview?term=1-YEAR&tab=subscription. [cit. 2024-05-06].
[35] GAINTA INDUSTRIES LTD., 2010. *D4MG-ASSEMBLY*. Online. REVISID: A. Dostupné z: https://www.tme.eu/Document/700083e69be2fc709cd21ba7c9f4006c/D4MG.pdf. [cit. 2024-05-06].
[36] TEXAS INSTRUMENTS, 2017. *LMR33630 SIMPLE SWITCHER® 3.8-V to 36-V, 3-A Synchronous Step-down Voltage Converter*. Online. SNVSAN3F –

AUGUST 2017 – REVISED NOVEMBER 2020. Dostupné z: https://www.ti.com/lit/ds/symlink/lmr33630.pdf?ts=1714979738039&ref_url=https%253A%252F%252Fwww.google.com%252F. [cit. 2024-05-06].
[37] SIMCom. *SIM7000 _Hardware Design*. Online. Version 1.05. 2018. Dostupné z: https://simcom.ee/documents/SIM7000E/SIM7000%20Hardware%20Design_V1.05.pdf. [cit. 2024-01-02].
[38] BOURNS. *SRP7050WA Series - Shielded Power Inductors*. Online. C1753 05/17/18R. Dostupné z: https://www.farnell.com/datasheets/3930707.pdf. [cit. 2024-05-06].
[39] CUI INC, 2022. *Series: PXO78-500-M*. Online. Rev. 1.0. Dostupné z: https://www.farnell.com/datasheets/3773832.pdf. [cit. 2024-05-06].
[40] TRACO POWER, 2024. *TME Series, 1 Watt*. Online. Rev. February 7, 2024. Dostupné z: https://www.tracopower.com/int/tme-datasheet. [cit. 2024-05-06].
[41] CRYSTAL UNITS. *NX3225GD*. Online. 120306_NX3225GD_e. Dostupné z: https://cz.mouser.com/datasheet/2/905/c_NX3225GD_STD_CRA_3_e-1317534.pdf. [cit. 2024-05-06].
[42] TEXAS INSTRUMENTS, 2020. *ISO1044 Isolated CAN FD Transceiver in Small Package*. Online. SLLSFB0A – MARCH 2020 – REVISED JULY 2020. Dostupné z: https://www.ti.com/lit/ds/symlink/iso1044.pdf?ts=1714972242515&ref_url=https%253A%252F%252Fwww.google.com%252F. [cit. 2024-05-06].
[43] TEXAS INSTRUMENTS, 2020. *ISO672x General Purpose Basic Dual-Channel Digital Isolators with Robust EMC*. Online. SLLSFJ0F – JANUARY 2020 – REVISED FEBRUARY 2023. Dostupné z: https://www.ti.com/lit/ds/symlink/iso6721.pdf?ts=1714920441911&ref_url=https%253A%252F%252Fwww.mouser.fr%252F. [cit. 2024-05-06].
[44] TEWA TEMPERATURE SENSORS, 2015. *TT02-10KC3-T105-1500*. Online. Revision: 00. Dostupné z: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/876/TT02-10KC3-T105-1500.pdf. [cit. 2024-05-06].
[45] PHOENIX CONTACT. *PC 6/ 2-GU-7,62* - Kontaktová lišta desky plošných spojů. Online. 1136002. Dostupné z: https://www.phoenixcontact.com/product/pdf/api/v1/MTEzNjAwMg?_realm=cz&_locale=cs-CZ&blocks=commercial-data%2Ctechnical-data%2Cdrawings%2Capprovals%2Cclassifications%2Cenvironmental-compliance-data%2Call-accessories. [cit. 2024-05-06].
[46] FOTEK. *SSR Series*. Online. Dostupné z: https://datasheetspdf.com/pdf-down/S/S/R/SSR-10DA-Fotek.pdf. [cit. 2024-05-06].
[47] ONSEMI, 2003. *BSS123*. Online. November, 2021 − Rev. 10. Dostupné z: https://cz.mouser.com/datasheet/2/308/1/BSS123_D-2310466.pdf. [cit. 2024-05-06].

[48] VISHAY SILICONIX. *IRLL110, SiHLL110*. Online. S21-0322-Rev. G, 05-Apr-2021. Dostupné z: https://www.vishay.com/docs/91320/sihll110.pdf. [cit. 2024-05-06].
[49] HAGER. *ESL225SDC*. Online. S21-0322-Rev. G, 05-Apr-2021. Dostupné z: https://www.hager.cz/katalog-produktu/distribuce-energie/modularni-pristroje/spinaci-a-signalizacni-pristroje/stykace/esl225sdc/82022.htm. [cit. 2024-05-06].
[50] AVAGO TECHNOLOGIES. *ACPL-227 / ACPL-247*. Online. Dostupné z: https://www.farnell.com/datasheets/72466.pdf. [cit. 2024-05-06].
[51] IXYS, 2021. *CPC1976Y*. Online. DS-CPC1976Y-R07. USA. Dostupné z: https://cz.mouser.com/datasheet/2/240/media-3323922.pdf. [cit. 2024-05-06].
[52] TEXAS INSTRUMENTS, 2021. *AMC1351 Precision, 5-V Input, Reinforced Isolated Amplifier*. Online. SBASAA8 – DECEMBER 2021. Dostupné z: https://www.ti.com/lit/ds/symlink/amc1351.pdf?ts=1714906815221&ref_url=https%253A%252F%252Fwww.mouser.de%252F. [cit. 2024-05-06].
[53] MICROCHIP. *AVR® and SAM MCU Downloads Archive*. Online. Dostupné z: https://www.microchip.com/en-us/tools-resources/archives/avr-sam-mcus. [cit. 2024-05-06].
[54] T-MOBILE. *Mobilní internet*. Online. Dostupné z: https://www.t-mobile.cz/podpora/technicka-podpora/internet-a-e-mail/mobilni-internet. [cit. 2024-01-02].
[55] JEDLIČKA, Václav. *ProfiCAD*. Online. Dostupné z: https://www.proficad.cz/. [cit. 2024-05-06].
[56] MICROSOFT. *Visual Studio: Integrované vývojové prostředí (IDE) a editor kódu pro vývojáře softwaru a týmy*. Online. Dostupné z: https://visualstudio.microsoft.com/cs/. [cit. 2024-01-04].
[57] *Weather Forecast API*. Online. Open-Meteo. Dostupné z: https://open-meteo.com/en/docs. [cit. 2024-01-04].
[58] RASPBERRY PI. *Raspberry Pi OS*. Online. Dostupné z: https://www.raspberrypi.com/software/. [cit. 2024-05-06].
[59] CLOUDFLARE. *Cloudflare*. Online. Dostupné z: https://www.cloudflare.com/. [cit. 2024-05-06].

SEZNAM SYMBOLŮ A ZKRATEK

Zkratky:

AD	Analogově-digitální
ALU	Areitmeticko-logická jednotka
API	Rozhraní pro aplikace
APN	Název přístupového bodu
CAN	Controller Area Net (řídicí oblastní síť)
CRC	Cyklická redundantní kontrola
Č/Č	Čítač/časovač
DPS	Deska plošných spojů
EEPROM	Elektricky mazatelná programovatelná paměť
EMI	Elektromagnetické rušení
ESR	Ekvivalentní sériový elektrický odpor
FD	Flexible data rate (flexibilní datový tok)
GPIO	Univerzální vstupy a výstupy
GPRS	General Packet Radio Service
GSM	Globální systém pro mobilní komunikaci
HTTP	Hypertextový přenosový protokol
ID	Identifikátor
IOT	Internet věcí
ISP	In-system programming
JSON	JavaScriptový zápis objektů
LAN	Místní síť
LED	Elektroluminiscenční dioda
LORA	Long-Range (dlouhý dosah)
LSB	Nejméně významný bit
LTE	Long Term Evolution (dlouhodobý vývoj)
MOB	Message object (objekt zprávy)
MOSFET	Polem řízený tranzistor se strukturou kov-oxid-polovodič
MQTT	MQ Telemetry Transport (Telemetrický přenos MQ)
MSB	Nejvíce významný bit
MVC	Model-Pohled-Kontrolér
NTC	Záporný teplotní koeficient
OLED	Organická elektroluminiscenční dioda
PTC	Kladný teplotní koeficient
SIM	Identifikační modul uživatele
SMD	Součástka pro povrchovou montáž
SMS	Služba krátkých textových zpráv
SPI	Sériové periferní rozhranní

SRAM	Statická paměť s náhodným přístupem
SSR	Solid state relé
UART	Univerzální asynchronní přijímač/vysílač
URL	Jednotný lokátor zdroje
VAC	Voltů střídavých
VDC	Voltů stejnosměrných
WI-FI	Wireless Fidelity (bezdrátová věrnost)

Symboly:

A	Zesílení izolačního zesilovače	(-)
ADC	Vrácená digitální hodnota z AD převodníku	(-)
B	Magnetická indukce	(T)
β	Převodní koeficient změny odporu na teplotě NTC termistoru	(K)
CAN_H	Kladný pól kroucené dvojlinky CAN sběrnice	(-)
CAN_L	Záporný pól kroucené dvojlinky CAN sběrnice	(-)
CS	Pin SPI sběrnice chip-select	(-)
D	Hodnota předděličky	(-)
f	Frekvence	(Hz)
$GAIN$	Zesílení diferenčního zesilovače	(-)
GND	Společná zem	(-)
h	Tloušťka	(mm)
I	Proud	(A)
k	Převodní koeficient mezi analogovou hodnotou proudu a digitálním kódem	(A/d)
N	Počet impulzů	(-)
n	Počet bitů	(-)
$PICO$	Pin SPI sběrnice Peripheral-in-controller-out	(-)
$POCI$	Pin SPI sběrnice Peripheral-ou-controller-in	(-)
R	Elektrický odpor	(Ω)
RX	Přijímací linka	(-)
SCK	Pin SPI sběrnice Serial-clock	(-)
T	Teplota	(K)
t	Čas 1 impulzu hodinového signálu	(s)
TX	Vysílací linka	(-)
U	Napětí	(V)

SEZNAM PŘÍLOH

Příloha A - Návod k využití AT příkazů

Pro synchronizaci přenosové rychlosti mezi SIM modulem a mikrokontrolerem slouží příkaz *„AT“*. Pokud byla synchronizace úspěšná, tedy SIM modul dokáže komunikovat s mikrokontrolerem, odešle SIM modul odpověď *„OK“*.

Po synchronizaci je potřeba zjistit příkazem *„AT+CPIN?“*, zda je přítomna SIM karta. Odpovědi SIM modulu mohou být následující:

- *+CPIN: ERROR* – SIM karta není přítomna
- *+CPIN: READY* – SIM karta je přítomna a připravena k použití
- *+CPIN: SIM PIN* – SIM karta je přítomna, ale vyžaduje PIN kód
- *+CPIN: SIM PUK* – SIM karta je přítomna, ale vyžaduje PUK kód

Pokud SIM karta vyžaduje PIN nebo PUK kód, zadá se AT příkazem ve tvaru *„AT+CPIN=\"<PIN/PUK>\"“*.

Pro přihlášení SIM karty do mobilní sítě slouží příkaz *„AT+CREG=1“*.

K přihlášení SIM karty do internetu (GPRS sítě) slouží příkaz *„AT+CIICR“*. Předtím je ovšem potřeba definovat parametry připojení od poskytovatele SIM karty – APN, přihlašovací jméno a heslo. APN je URL adresa, která zprostředkovává připojení SIM karty do internetu přes mobilního operátora. Parametry se předávají pomocí příkazu *„AT+CSTT=\"<APN>\",\"<přihlašovací_jméno>\",\"<heslo>\"“*.

Dále je potřeba definovat technologii připojení k internetu. K tomu slouží příkazy *„AT+SAPBR=<příkaz>,<profil>“*. Pro technologii LTE je *<profil>* = 1).

- Dotaz *„AT+SAPBR=2,1“* – zjišťuje, zda je přenos pomocí technologie LTE dostupný
- Příkaz *„AT+SAPBR=1,1“* – umožní přenos dat pomocí technologie LTE
- Příkaz *„AT+SAPBR=0,1“* – zakáže přenos dat pomocí technologie LTE

Po této sekvenci připojení SIM karty k mobilní síti a internetu je modul připraven odesílat HTTP API požadavky (pouze HTTP GET a HTTP POST). Ty jsou volány pomocí sekvence AT příkazů:

- *„AT+HTTPINIT“* – Oznamuje SIM modulu, že bude proveden HTTP API dotaz
- *„AT+HTTPPARA=\"URL\",\"<URL_adresa>\"“* – zadání URL adresy API metody
- *„AT+HTTPPARA=\"CID\",1“* – zadává context ID, který určuje QoS (kvalitu služeb). Zde je QoS = 1.
- *AT+HTTPPARA=\"CONTENT\",\"<formát>\"“* (pouze u metody HTTP POST) – Určuje, v jakém formátu budou odeslány data v těle dotazu. Pro účely této práce je využit formát JSON, parametr *<formát>* musí tedy být nastaven jako *„application/json“*.
- *„AT+HTTPDATA=<délka>,<časový_limit>“* (pouze u metody HTTP POST) – Oznamuje SIM modulu, že v následujícím čase definovaným

parametrem <*časový_limit*> má očekávat příjem dat o délce podle parametru <*délka*>. Po odeslání tohoto příkazu je potřeba odeslat textový řetězec s daty. Časový limit je definován v milisekundách.

- *„AT+HTTPACTION=<ID_metody>“* – určuje typ metody a zároveň odešle dotaz na server. Pro HTTP GET je *<ID_metody>* = 0, pro HTTP POST je *<ID_metody>* = 1.
- *„AT+HTTPREAD“* – SIM modul vrátí zpětnou vazbu serveru
- *„AT+HTTPTERM“* – Oznamuje SIM modulu, že HTTP API dotaz byl dokončen

Příloha B - Postup nasazení ASP.NET Core aplikace na Raspberry Pi 4

1. Instalace operačního systému „*Raspberry Pi OS*“ (založeno na Debian GNU/Linux):

- Připojení SD karty k PC, která bude následně využita v Raspberry Pi
- Stažení a instalace programu „*Raspberry pi imager*“ na PC z oficiálních stránek
- Volba Raspberry Pi desky, požadovaného operačního systému a média, na které se bude systém nainstalovat
- V pokročilém nastavení povolení připojení přes SSH a nastavení heslo.
- Po instalaci vyjmutí SD karty z počítače a vložení do Raspberry Pi.
- Připojení Raspberry Pi k napájení
- Připojení Raspberry Pi k internetu prostřednictvím konzole a SSH zabezpečení

2. Instalace Dockeru

- Aktualizace operačního systému a ostatních programů prostřednictvím příkazů „*sudo apt update*“ a „*sudo apt upgrade*“ v konzoli
- Pro instalaci Dockeru slouží příkaz „*curl -ssl https://get.docker.com | sh*“, který stáhne instalační script z oficiálních stránek Dockeru.
- V případě nastavení více uživatelů v Raspberry Pi je možno nastavit práva pro ostatní uživatele, v případě jednoho uživatele tento krok přeskočit
- Funkcionalitu Dockeru po instalaci lze ověřit příkazem „*docker run hello-world*“

3. Vytvoření kontejneru s instalací Ubuntu

- Stažení image Ubuntu z databáze dockeru příkazem „*sudo docker pull ubuntu*“
- Po stažení vytvoření nového kontejneru s instalací Ubuntu pomocí příkazu „*sudo docker run -ti --rm ubuntu /bi/bash*“. Vlajka „*-ti*“ označuje, že instalace bude dostupná v interaktivním terminálovém režimu. Vlajka „*--rm*“ určuje, že pokud bude kontejner zastaven, instalace bude smazána. Argument „*/bin/bash*“ spustí terminál shellu Bash.
- Po instalaci a spuštění je nutné provést aktualizace a nainstalovat ostatní software příkazem „*apt update && apt install lsb-core*“

4. Instalace ASP.NET Core aplikace

- Pro instalaci ASP.NET Core runtime slouží příkaz „*sudo apt-get install -y aspnetcore-runtime-7.0*“
- Instalaci lze ověřit příkazem „*dotnet--info*“

- Vytvoření složky pro ASP.NET aplikace a nastavení spouštěcích práv (příkaz „*mkdir -m 755 Root_Flash/Ubuntu/API*“ v Raspberry Pi, příkaz „*mkdir -m 755 var/www/app*“ v kontejneru, vlajka „*-m 755*“ značí udělení práv pro čtení a zápis)
- Vložení ASP.NET aplikace do vytvořené složky
- Ověřit funkčnost aplikace příkazem „*dotnet <název_aplikace>.dll*“

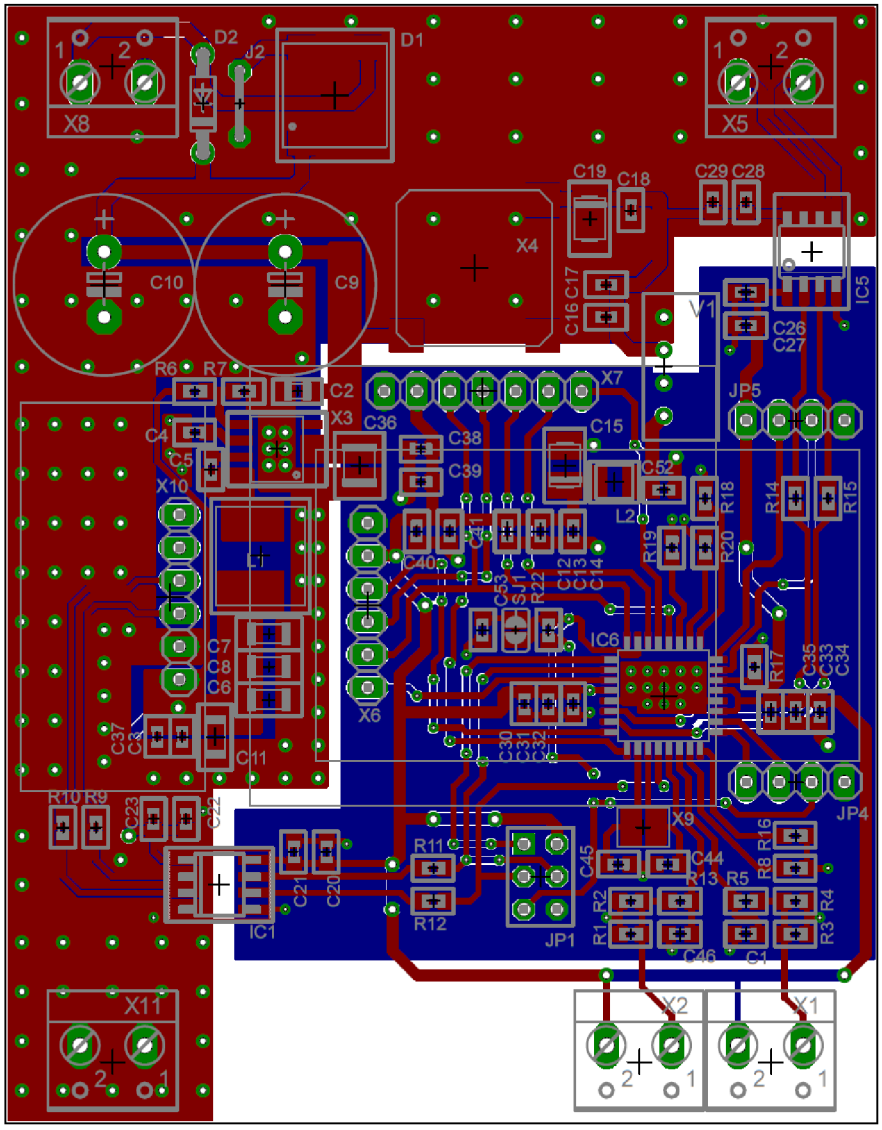
5. **Instalace Apache HTTP Proxy Serveru**
   - Instalace pomocí příkazu „*sudo apt-get install -y apache2*“
   - Konfigurace proxy serveru příkazem „*sudo a2enmod prox proxy_http*“
   - Vytvoření konfiguračního souboru pomocí příkazu „*nano /etc/apache2/sites-available/app.conf*“
   - Do souboru vložit nastavení:

```
<VirtualHost *:80>
    ProxyPreserveHost On
    ProxyPass / http://127.0.0.1:5000/
    ProxyPassReverse / http://127.0.0.1:5000/
    ErrorLog ${APACHE_LOG_DIR}/app-error.log
    CustomLog ${APACHE_LOG_DIR}/app-access.log common
</VirtualHost>
```

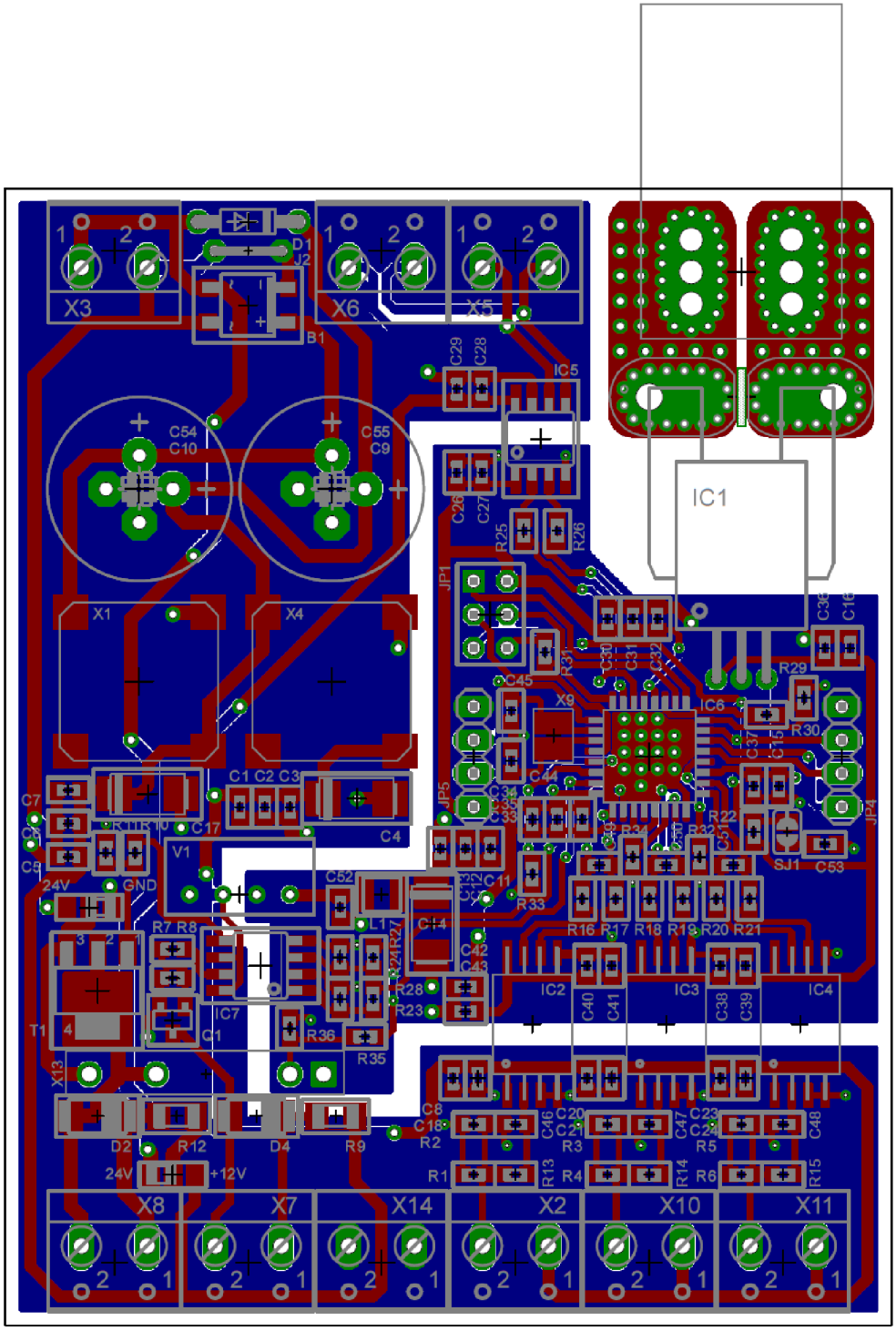
   - Po nastavení restart Apache serveru pomocí příkazů „*sudo a2dissite 000-default*“, „*sudo a2ensite app*“ a „*sudo systemctl restart apache2*“
   - Po nastavení lze získat data prostřednictvím API příkazu (port 80)
   - Uložení kontejneru pomocí příkazu „*docker commit -p container_id <jméno_kontejneru>*“
6. **Spuštění ASP.NET Core aplikace**
   - K spuštění aplikace se využívá uloženého kontejneru. Následující příkaz spouští uložený kontejner společně s webserverem a ASP.NET aplikace. Aplikaci poté přesměrovává na port 8080.
   - Celý příkaz: „*sudo docker run -ti -v /Root_Flash/Ubuntu/API:/var/www/app -p 8080:80 --name <container_name> -w /var/www/app --restart unless-stopped <image_name> /bin/sh -c 'service apache2 start && dotnet <název_aplikace>.dll'*“
7. **Zpřístupnění ASP.NET Core aplikace k internetu, např. pomocí veřejné IP adresy nebo služby Cloudflare**

Příloha C - Layout DPS autonomního systému

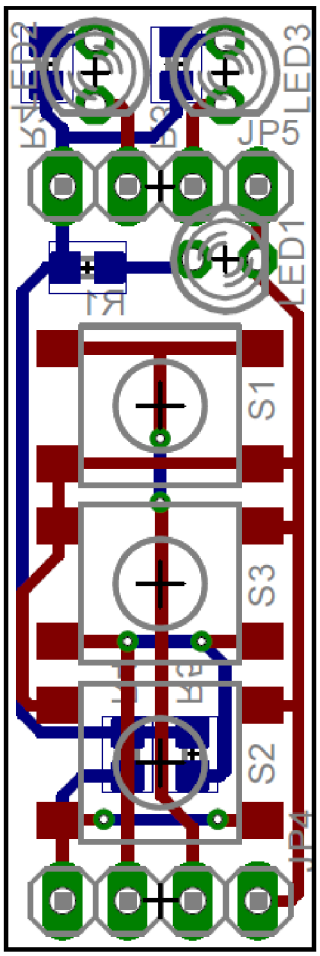
C.1 Komunikační modul



C.2 Výkonový modul

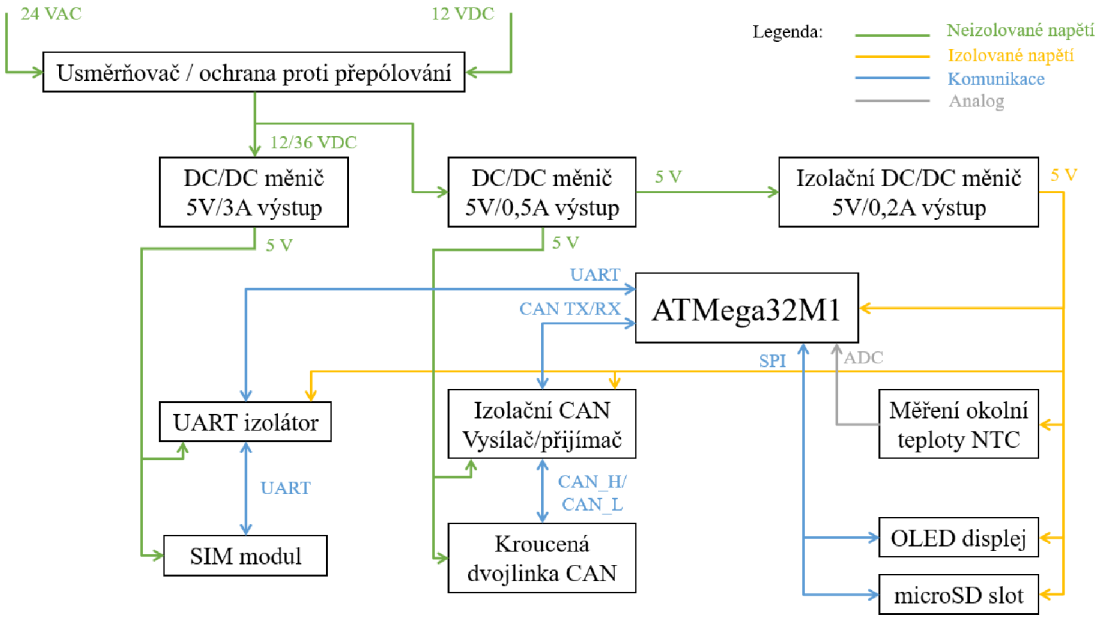


C.3 Ovládací panel

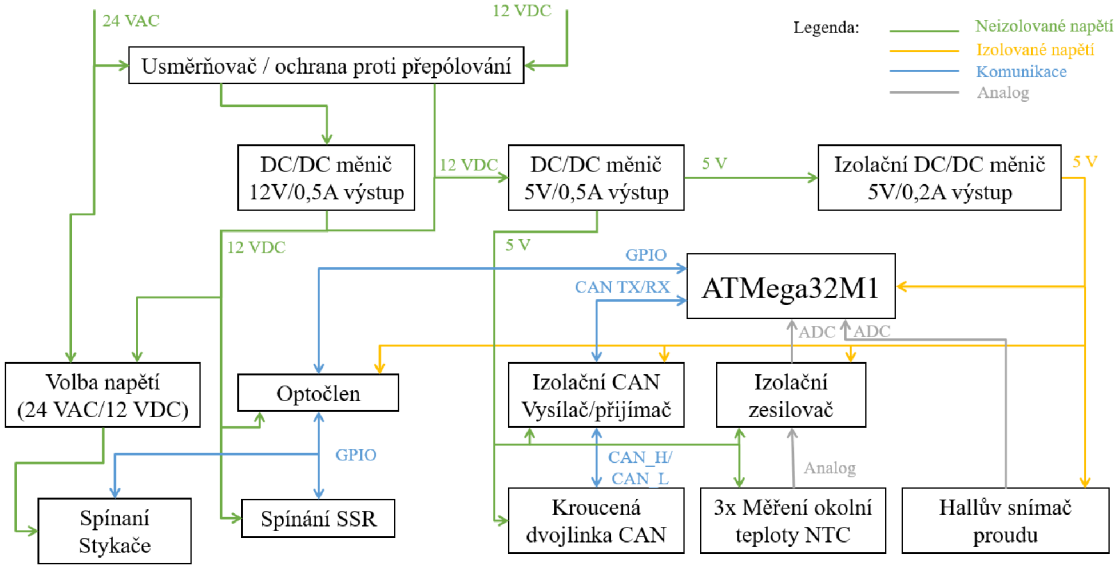


Příloha D - Bloková schémata komunikačního a výkonového modulu

D.1 Komunikační modul

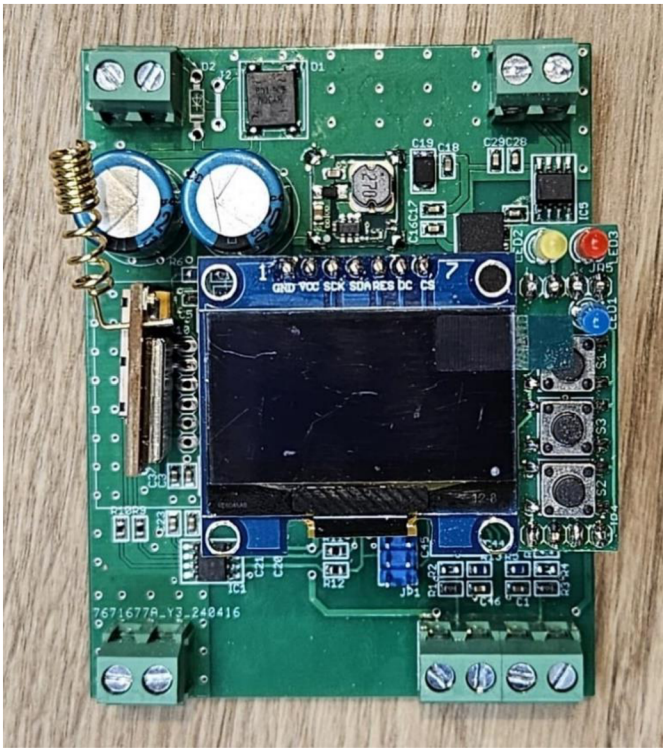


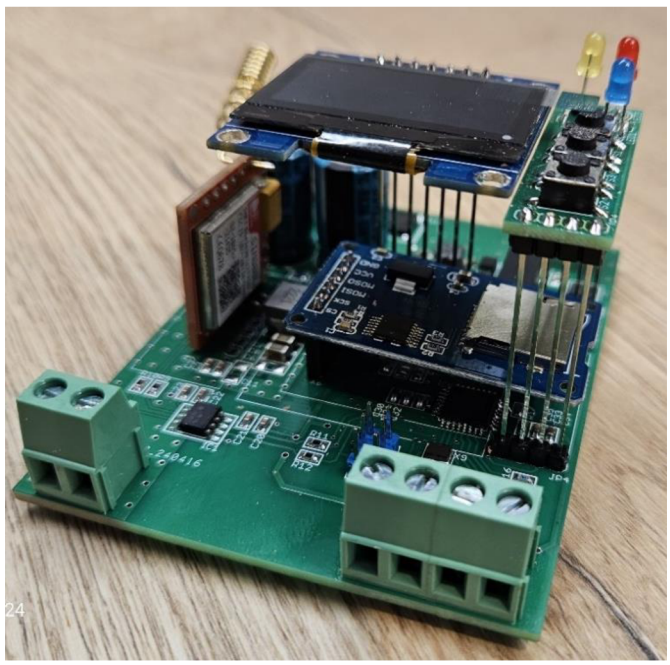
D.2 Výkonový modul



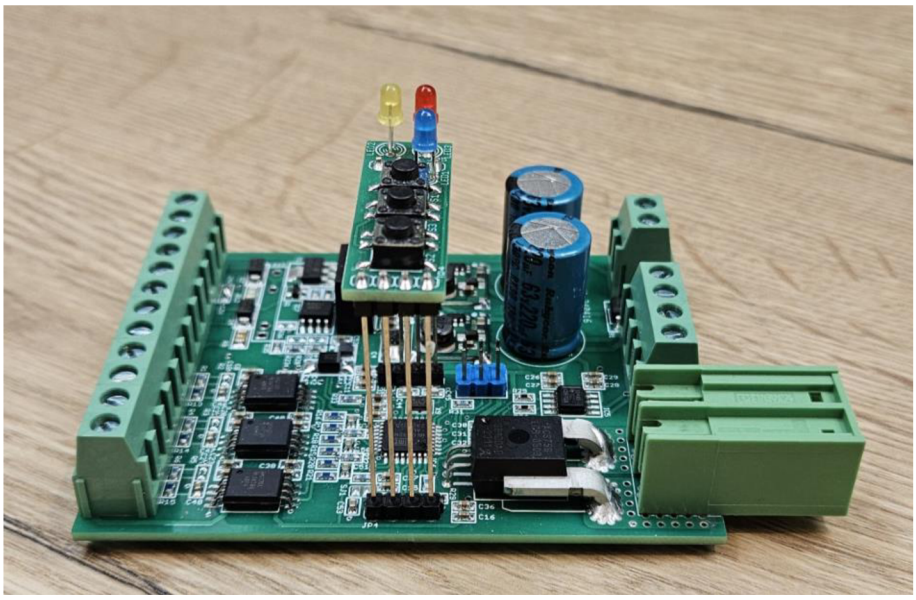
Příloha E - Fotodokumentace elektroniky systému

E.1 Komunikační modul



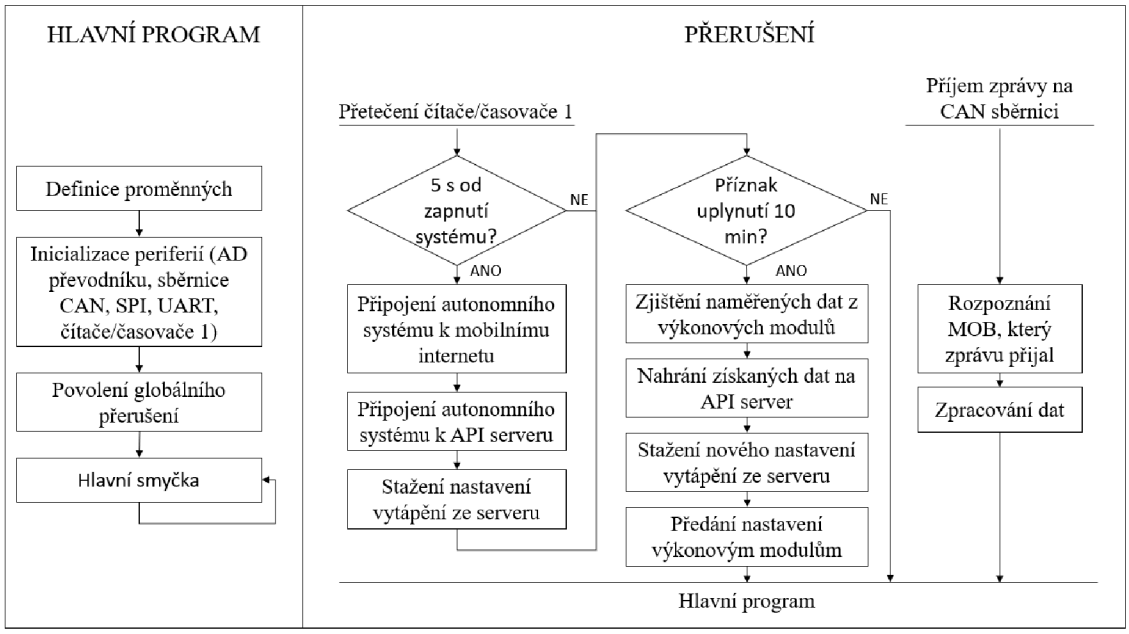


E.2 Výkonový modul

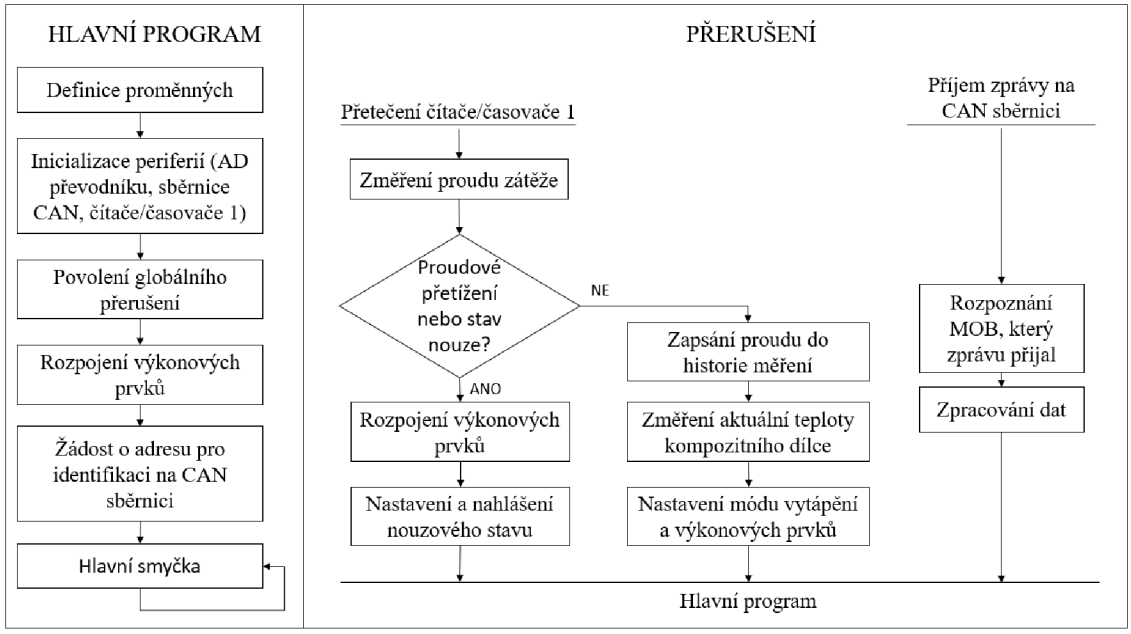


Příloha F - Blokové diagramy kódů

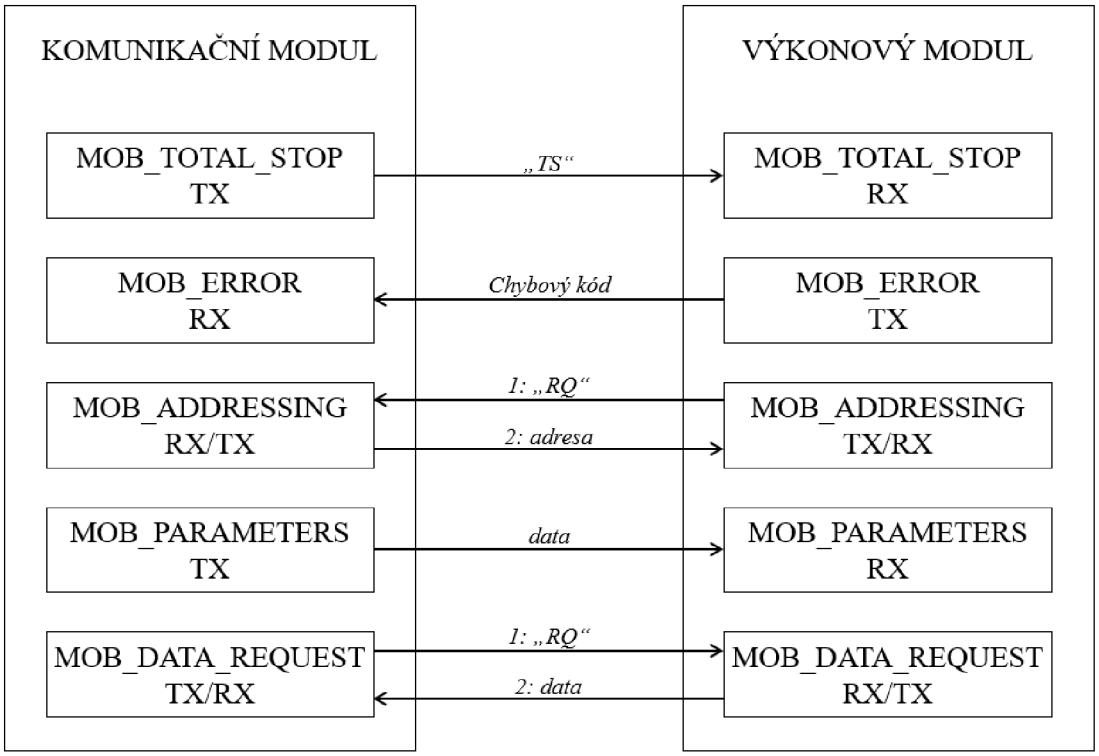
F.1 Kód komunikačního modulu



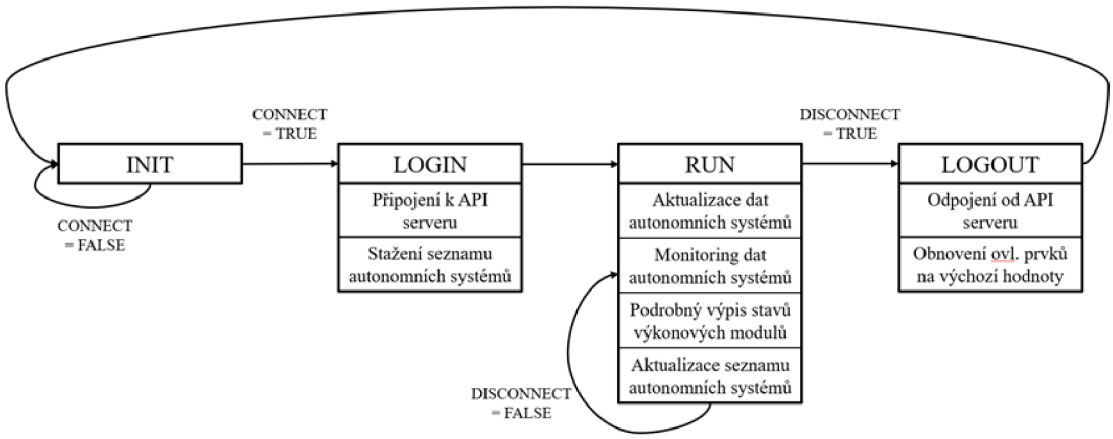
F.2 Kód výkonového modulu



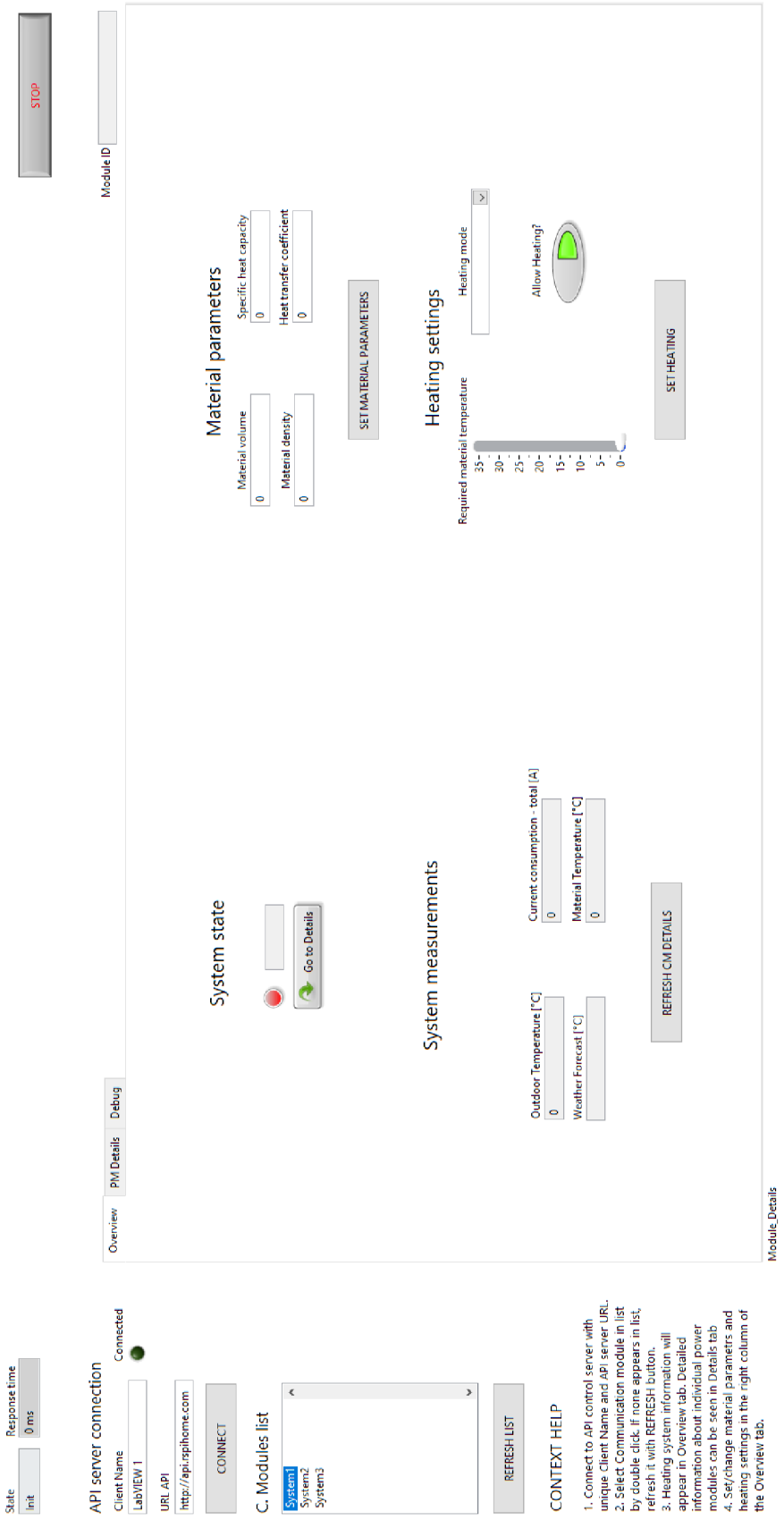
F.3 Blokový diagram CAN sběrnice



F.4 Stavový automat LabVIEW



Příloha G - Vizuální část aplikace LabVIEW



Příloha H - Elektronická příloha

Elektronická příloha je odevzdaná v IS VUT pod názvem *Přílohy_220881.zip*.

Příloha obsahuje

- Elektronický návrh autonomního systému v programu EAGLE
- Zdrojové kódy mikrokontrolérů
- Zdrojový kód API aplikace serveru
- Uživatelskou aplikaci LabVIEW