VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2021 Adam Moravčík

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

MODBUS WIFI PRO PŘIPOJENÍ PYROMETRU VE SLÉVÁRNĚ
MODBUS WIFI FOR PYROMETER COMMUNICATION IN A FOUNDRY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Adam Moravčík**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Mgr. Karel Slavíček, Ph.D.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Telekomunikační a informační systémy**

Ústav telekomunikací

Student: Adam Moravčík ***ID:*** 211266

Ročník: 3 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Modbus WiFi pro připojení pyrometru ve slévárně

POKYNY PRO VYPRACOVÁNÍ:

Cílem bakalářské práce je navrhnout způsob připojení pyrometru s rozhraním Modbus-RTU přes Wifi síť. Pyrometr slouží pro kontrolu teploty licích forem ve slévárně. Součástí práce je i návrh prostorového umístění pyrometru a způsobu ukládání a zpracování dat. Věcným výstupem bakalářské práce bude praktická realizace přípravku s alternativním řešením napojení do WiFi sítě a návrh způsobu diagnostiky kvality MODBUS komunikace přes WiFi sít v prostředí slévárny.

DOPORUČENÁ LITERATURA:

[1] Klasen, F., Oestreich, V., & Volz, M. (2011). Industrial Communication with Fieldbus and Ethernet. Beltz Verlag.

[2] Modbus [online]. Hopkinton: Modbus Organization, 2020 [cit. 2020-10-21]. Dostupné z: https://modbus.org/

Termín zadání: 1.2.2021 ***Termín odevzdání:*** 31.5.2021

Vedoucí práce: doc. Mgr. Karel Slavíček, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

Abstrakt

Bakalárska práca sa zameriava na návrh spôsobu pripojenia pyrometru komunikujúceho rozhraním MODBUS RTU k WiFi sieti v prostredí Brnenskej zlievarne.
Prvá časť je venovaná protokolu MODBUS, jeho štruktúre, princípom a možnosti prepojenia k sieti TCP/IP, ktorá je využívaná aj vo WiFi.
Druhá časť je zameraná na prehľad dostupných jednodoskových počítačov a mikrokontrolérov vhodných pre realizáciu návrhu.
Tretia časť sa venuje samotnému hardwarovému návrhu zariadenia realizujúceho prepojenie pyrometru k WiFi sieti. Popisuje význam jednotlivých častí a spôsob ich vzájomného prepojenia.
Štvrtá časť sa zaoberá softwarovým návrhom programu. Opisuje jednotlivé bloky programu a ich funkciu.
Piata časť sa venuje testovaniu zariadenia pomocou simulačných programov „Modbus Pool“ a „Modbus Slave“.

Kľúčové slova

MODBUS RTU, MODBUS TCP/IP, ESP 32, RS-485, Mikrokontróler

Abstract

The bachelor's thesis focuses on the design of a method for connecting a pyrometer communicating via the MODBUS RTU interface to a WiFi network in the environment of the Brno foundry.
The first part is devoted to the MODBUS protocol, its structure, principles, and possibilities of connection to the TCP / IP network, which is also used in WiFi.
The second part is focused on an overview of available single-board computers and microcontrollers suitable for the implementation of the design.
The third part deals with the hardware design of the device which realizes the connection of the pyrometer to the WiFi network. It describes the meaning of individual parts and the way of their interconnection.
The fourth part deals with the software design of the program. It describes the individual blocks of the program and their function.
The fifth part deals with testing the device using the simulation programs "Modbus Pool" and "Modbus Slave".

Keywords

MODBUS RTU, MODBUS TCP/IP, ESP 32, RS-485, Microcontroller

Bibliografická citácia:

MORAVČÍK, Adam. Modbus WiFi pro připojení pyrometru ve slévárně. Brno, 2021. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/133374. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce Karel Slavíček.

Vyhlásenie

Vyhlasujem, že svoju bakalársku prácu na tému „Modbus WiFi pro připojení pyrometru ve slévárně“ som vypracoval samostatne pod vedením vedúceho bakalárskej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia S 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

V Brne dna: ……....

……

podpis autora

Poďakovanie

Rád by som poďakoval vedúcemu bakalárskej práce pánovi doc. Mgr. Karel Slavíček, Ph.D.za trpezlivosť, konzultácie a odborné vedenie pri práci.

V Brne dna: …….... ……

podpis autora

Obsah

Zoznam symbolu a skratke

Skratky:

ADU	Application Data Unite
AP	Access Point
ASCII	American Standard Code for Information Interchange
CRC	Cyclic Redundancy Check
DDR2	Double Data Rate 2
DDR3	Double Data Rate 3
DNS	Domain Name System
DPS	Doska plošných spojov
GPIO	General-purpose input/output
IP	Internet Protocol
ISO	International Organization of Standards
LRC	Longitudinal Redundancy Checking
LSB	Leats Significant Bite
MAC	Media Access Control
MBAP	Modbus Application Protocol
Mbps	Megabit per second
MSB	Most Significant Bite
OSI	Open Systems Interconnect
PDU	Protocol Data Unite
RTU	Remote Terminal Unit
TCP	Transmission Control Protocol
TTL	Transistor – Transistor Logic
UART	Universal asynchronous receiver/transmitter
USB	Universal Serial Bus

Zoznam obrázkov

Zoznam tabuliek

1. ÚVOD

Cieľom mojej bakalárskej je navrhnúť spôsob pripojenia zlievarenského pyrometru s rozhraním Modbus RTU k WiFi sieti a s možnosťou alternatívneho pripojenia.

Pri výrobe odliatkov v zlievarni je nevyhnutná neustála kontrola výrobného procesu. Pre kontrolu teploty odlievacích foriem slúži pyrometer, ktorý meria bezkontaktne. Informácie o teplote musia byť dostupné pracovníkom zlievarne.

Brnenská zlievareň Heunisch doposiaľ využívala pyrometer od výrobcu DIAS, ktorý v spojení s PLC WAGO zabezpečoval komunikáciu prostredníctvom MODBUS.

Novo je zaobstaraný pyrometer výrobcu Keller typu PA 83 AF 13/C s rozhraním MODBUS RTU. Aby zamestnanci mali prístup k dátam zachytených pyrometrom je potrebné ich bezdrôtovo prenášať prostredníctvom WiFi siete. Nakoľko je prostredie zlievarne veľmi industriálne, bezdrôtová komunikácia pomocou WiFi by nemusela byť spoľahlivá, a preto je potrebné aj alternatívne pripojenie k firemnej sieti.

V záverečnej práci som ako prvé naštudoval protokol MODBUS, jeho štruktúru, princípy a možnosti prepojenia k sieti TCP/IP, ktorá je využívaná aj vo WiFi. Toto pripojenie je možné realizovať pomocou zariadenia MODBUS gateway, ktoré zaistí preklad dát medzi prostredím sériovej komunikácie a TCP/IP.

MODBUS gateway teda vyžaduje určité množstvo výpočtového výkonu, ktorý zabezpečí preklad dát. K jeho realizácií je možné využiť jednodoskový počítač alebo mikrokontrolér. Ďalším krokom teda bolo vybrať vhodné zariadenie. Na trhu je veľké množstvo týchto výpočtových zariadení od rôznych výrobcov. Podľa potrebných požiadaviek bol zostavený prehlaď najvhodnejších zariadení, z ktorých som vybral vývojový kit ESP32-DevKitC-32U.

Po výbere prišiel na rad hardwarový návrh zapojenia samotného zariadenia. Pyrometer komunikuje sériovým rozhraním RS-485. Vývojový kit disponuje iba sériovým rozhraním UART, ktoré však nie je kompatibilné s RS-485. Na prevod komunikácie medzi rozhraniami bol preto použitý obvod MAX3485. Ako alternatívny spôsob pripojenia k sieti bol použitý Ethernetový modul s integrovaným obvodom ENC28J60 a zásuvkou RJ-45. Pre vizuálnu kontrolu funkčnosti zariadenia slúžia LED diódy, ktoré informujú o zapnutí zariadenia, pripojení k WiFi sieti alebo k Ethernetu a o komunikácií medzi zariadením a pyrometrom. Na základe schémy zapojenia bola navrhnutá doska plošných spojov.

Aby však zariadenie pracovalo ako MODBUS gateway nestačí iba hardware. Ďalším krokom bolo navrhnúť vhodný software. Programovanie bolo realizované s využitím frameworku Arduino v aplikácií Visual Studio Code. Program sa skladá z dvoch hlavných častí. Prvá časť sa stará o nadstavenie a voľbu typu pripojenia (WiFi, Ethernet) k sieti TCP/IP. Užívateľ prostredníctvom webového formulára zadá požadované údaje, ktoré sú následne uložené v pamäti a nie je tak potrebné ich opäť zadávať po reštarte zariadenia. Pre ich odstránenie slúži resetovacie tlačidlo. Druhou častou je samotná aplikácia MODBUS gateway, ktorá vykonáva potrebný preklad dát medzi MODBUS RTU a MODBUS TCP/IP.

Na záver prišlo na rad testovanie, ktoré však vzhľadom k aktuálnej pandemickej situácií nebolo možné realizovať v reálnom prostredí, a preto bolo zvolené alternatívne riešenie v domácich podmienkach. Pre potreby testovania boli zvolené simulačné programy „Modbus Pool“ a „Modbus Slave“, ktoré simulujú MODBUS master a slave zariadenia.

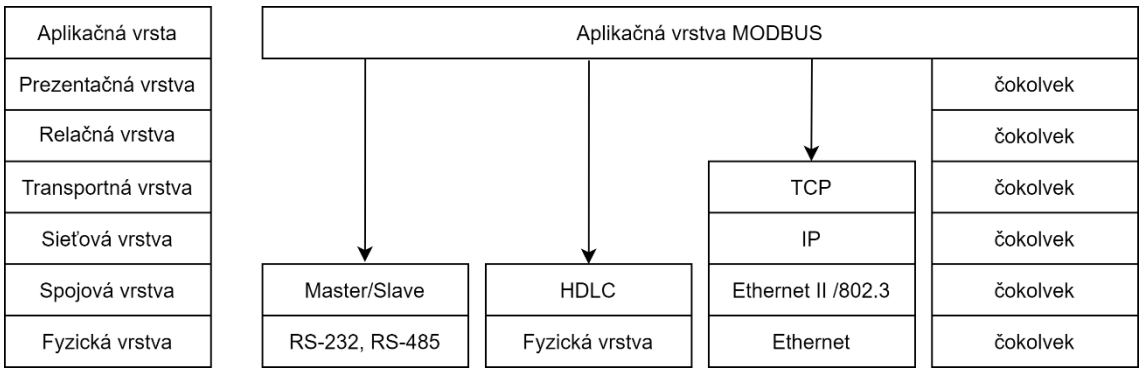
2. MODBUS

2.1 Prehlaď

Protokol MODBUS [1] je otvorený protokol vytvorený firmou Modicon Inc. v roku 1979 a stal sa priemyselným štandardom.

MODBUS je komunikačný protokol siedmej (aplikačnej) vrstvy referenčného modelu ISO/OSI a poskytuje tak zariadeniam komunikáciu typu klient/server (masrer/slave) nezávislú na type pripojenej sieti alebo zbernice. V súčasnej dobe je podporovaná široká škála komunikačných médií napr. siete typu Ethernet, optické a rádiové siete, no najväčšie zastúpenie má na sériových linkách RS-232, RS-422 a RS-485. Taktiež môže komunikovať cez internet a má vyhradené číslo portu 502 v protokolovej sade TCP/IP.

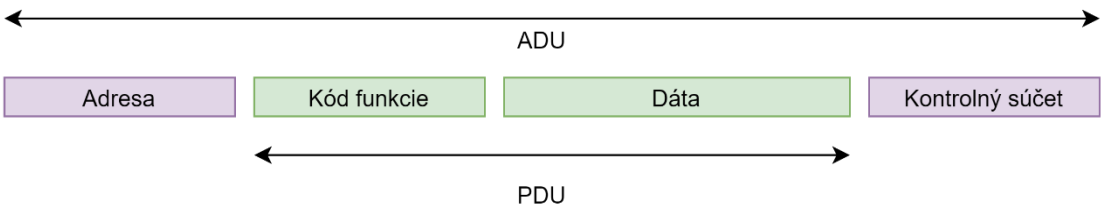
Zariadenia komunikujú metódou požiadavka-odpoveď, kde klient vytvorí požiadavku, ktorá je špecifikovaná pomocou kódov funkcie a server na ňu odpovedá.



Obr. 2-1 Príklad implementácie MODBUS. [2]

2.2 Dátový rámca MODBUS

Protokol MODBUS definuje protokolovú dátovú jednotku (PDU – *Protocol Data Unit*), ktorá je nezávislá na komunikačných vrstvách nižších úrovní. Na špecifických typoch sietí alebo zberníc sa PDU rozširuje o ďalšie časti a vytvorí tak správu na úrovni aplikačnej (ADU – *Application Data Unit*) viď obr. 2-2.



Obr. 2-2 Základná štruktúra MODBUS rámca. [1]

Pole „Adresa“:
Do tohto poľa umiestňuje klient (*master* zariadenie) adresu požadovaného servera (*slave* zariadenia). Pri odpovedi na klientovu požiadavku, vkladá server do tohto poľa svoju vlastnú adresu, aby klient vedel určiť zariadenie od ktorého obdržal odpoveď.

Veľkosť tohoto poľa je 1 bajt čo umožňuje adresovať 256 rôznych adries.

Pole „Kód funkcie“:
Veľkosť tohoto poľa je 1 bajt čo zodpovedá 256 rôznim kódom. Kódom funkcie udáva, akú operáciu má server vykonať, pričom rozsah od 128 po 255 je používaný pre označenie záporných správ (chýb), hodnotu 0 nesmie nadobúdať.

Niektoré kódy funkcie obsahujú tzv. kód podfunkcie, ktorý bližšie upresňuje danú operáciu.

Pole „Dáta“:
Dátová časť, poslaná od klienta k serveru, môže obsahovať doplnkové informácie potrebné k uskutočneniu požadovanej operácie, napr. adresa, hodnota registrov, ktoré majú byť zapísané alebo počet vstupov, ktoré má server prečítať. Keď nie sú potrebné ďalšie dáta k uskutočneniu operácie v tom prípade môže dátová časť chýbať. Pri spätnej komunikácii server do tejto časti vkladá hodnotu vykonanej funkcie.

Maximálna veľkosť dátovej časti je 252 bajtov.

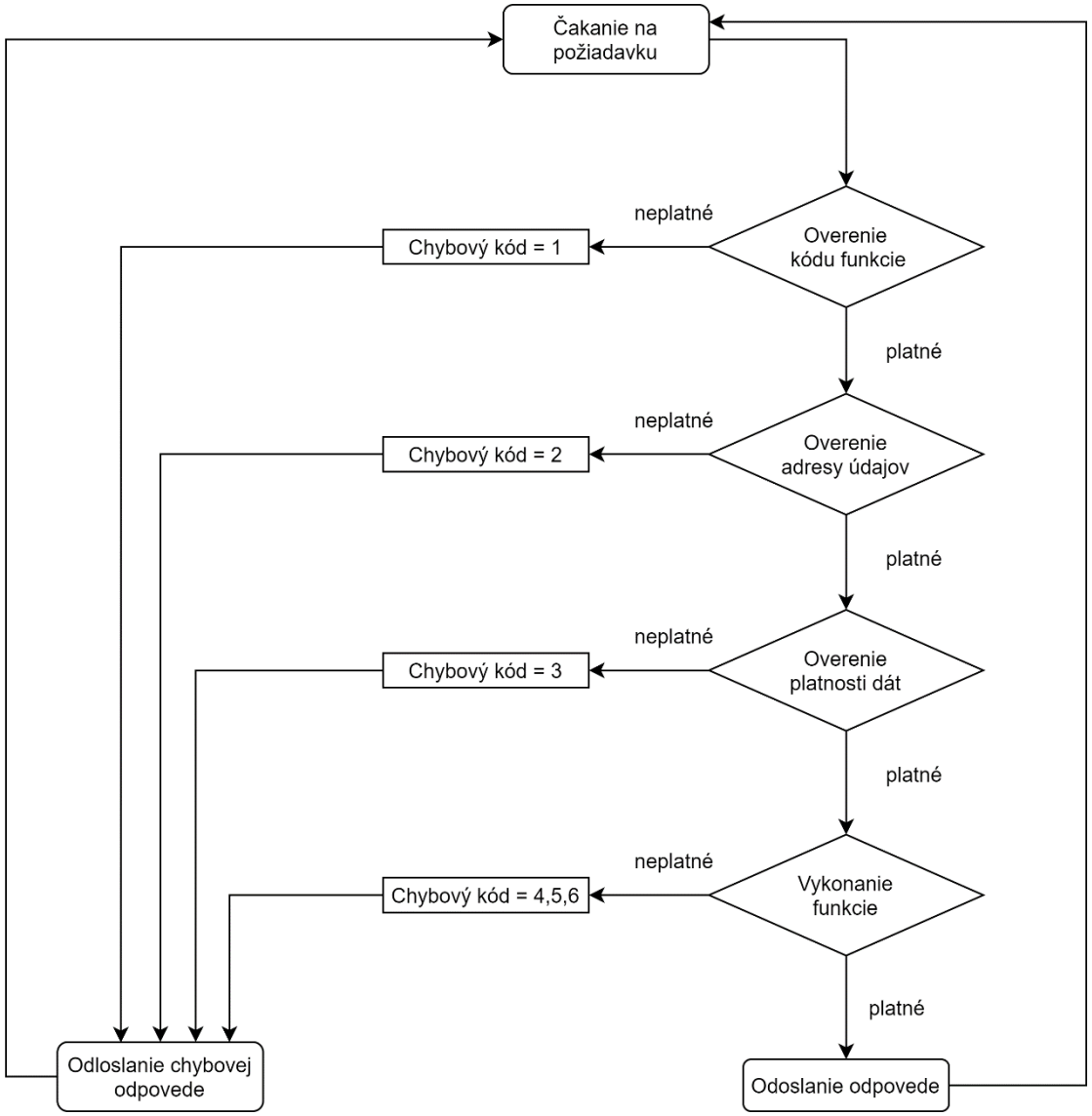
Pole „Kontrolný súčet“:
V tomto poli je uložený kontrolný súčet správy, ktorý slúži na overenie bezchybnosti prijatého rámca. Princíp výpočtu závisí od používaného módu (RTU,ASCII)[4].

2.3 Princíp komunikácie

Klient zahajuje komunikáciu, zostaví ADU s požadovanou adresou servera a kódom funkcie. Rámec je vyslaný na médium a spracovaný iba adresovaným zariadením. Ak nenastane žiadna chyba a server úspešne vykoná operáciu, odpovedá klientovi správou obsahujúcou v poli **kód funkcie**, kód značiaci úspešné vykonanie požiadavky a požadované dáta v **dátovej časti**. V prípade, že nastane chyba je v poli **kód funkcie**, kód požadovanej operácie a MSB tohto poľa je nadstavený na logickú 1 (+0x80). Chybový kód (ExceptionCode) udávajúci príčinu chyby je uložený v **dátovej časti**. Na strane klienta je implementovaný časový limit po dobu ktorého musí obdržať odpoveď od servera, aby klient nečakal donekonečna v prípade straty odpovede alebo požiadavky.

Vzhľadom na správnosť výsledku spracovania, môže nastať jedna z nasledujúcich možností:

- Bezchybná odpoveď:
  - kód funkcie v odpovedi = kód funkcie požiadavky.
- Odpoveď s chybou:
  - kód funkcie v odpovedi = kód funkcie požiadavky + 0x80,
  - je vrátené číslo chyby v závislosti na chybe ktorá sa vyskytla viď obr. 2-3.



Obr. 2-3 Postup spracovania MODBUS požiadavky na strane servera. [1]

Reprezentácia dát v MODBUS je tzv. „Big-endian“ [5]. Pri dátových položkách dlhších ako 1 bajt je ako prvý posielaný najvyšší bajt a posledný najnižší bajt.

2.4 Dátový model

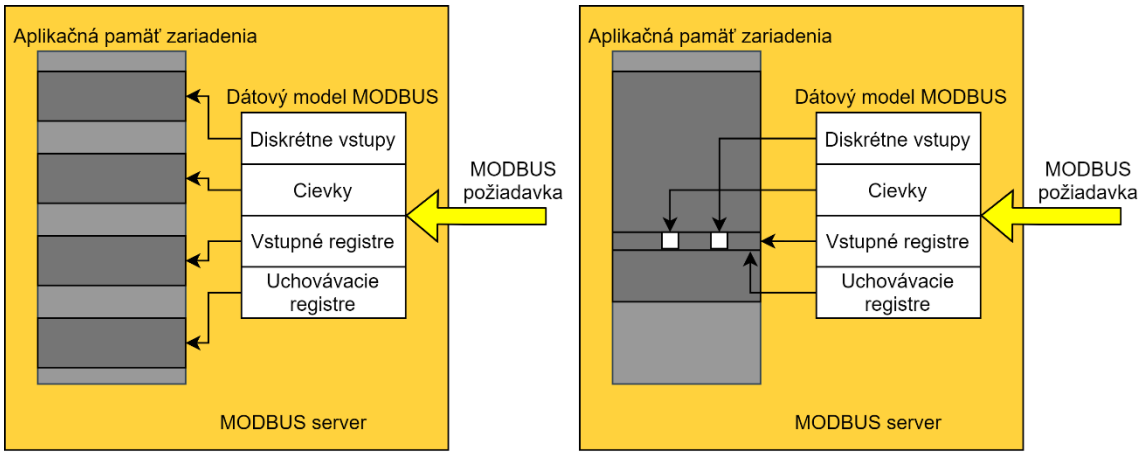
Dátový model MODBUS je založený na sade tabuliek viď tab. 2-2 Sú definované štyri základné tabuľky, kde každá má vlastný charakteristický význam. Tieto tabuľky sú namapované v pamäti zariadenia. Každá tabuľka môže mať vlastný adresný priestor, ktorý sa môže čiastočne alebo úplne prekrývať. Mapovanie je závislé od konkrétneho zariadenia. Tabuľky môžu obsahovať až 65536 položiek. Kvôli zachovaniu spätnej kompatibility sú bloky dlhé iba 10000 položiek. Pomocou príslušnej funkcie je možné pristupovať jednotlivo ku každej položke zvlášť alebo skupinovo k viacerým položkám naraz. Veľkosť skupiny je obmedzená maximálnou veľkosťou dátovej časti.

Tab. 2-1 Rozdelenie dátového modelu MODBUS. [2]

Tabuľka	Typ položky	Prístup	Popis	Adresa (MODICON)
Diskrétne vstupy *(Discrete Inputs)*	1-bit	čítanie	Dáta poskytované I/O systémom	10000 – 19999
Cievky *(Coils)*	1-bit	čítanie/zápis	Dáta modifikovateľné aplikačným programom	0 – 9999
Diskrétne vstupy *(Discrete Inputs)*	1-bit	čítanie	Dáta poskytované I/O systémom	10000 – 19999
Vstupné registere *(Input Registers)*	16-bitov	čítanie	Dáta poskytované I/O systémom	30000 – 39999
Uchovávacie registere *(Holding registers)*	16-bitov	čítanie/zápis	Dáta modifikovateľné aplikačným programom	40000 – 49999

Dáta môžu byť v zariadení zorganizované dvoma spôsobmi:

- Medzi položkami jednotlivých tabuliek nie je žiaden vzťah a teda každá tabuľka má svoj vlastný priestor v aplikačnej pamäti zariadenia viď obr. 2-4 a). K jednotlivým tabuľkám je možne pristupovať prostredníctvom príslušnej funkcie MODBUS.
- Zariadenie používa iba jeden dátový blok viď obr. 2-4 b). Aplikácia pristupuje k položkám pomocou rôznych funkcií MODBUS v závislosti na to čo je v daný moment výhodné.



Obr. 2-4 Dátový model MODBUS: a) s oddelenými blokmi, b) s jedným blokom. [1]

2.5 Adresovací model

V protokole MODBUS sú adresovacie pravidlá v PDU presne definované. Adresovanie dátových položiek v PDU je od 0 do 65535.

Adresovanie v rámci dátového modelu zloženého zo štyroch dátových blokov sú každé položky v bloku číslované od 1 do n.

2.6 Kód funkcie

Kód funkcie je v MODBUS rozdelený do troch skupín:

Verejné:

jednoznačne definované, unikátne, schválené spoločnosťou MODBUS.org, verejne zdokumentované, je k ním dostupný test zhody, zahrňujú verejné kódy funkcie aj nepriradené kódy rezervované pre budúce použite.

Užívateľsky definované:

rozsah kódov funkcií: 65 – 72 a 100 – 110, umožňuje užívateľovi implementovať vlastnú funkciu, ktorá nie je definovaná, nie je garantovaná unikátnosť kódu, je možné ich po prejednaní presunúť do verejných kódov.

Rezervované:

rozsah kódov funkcií: 111 – 127, sú používané niektorými firmami a nie sú dostupné pre verejné použitie.

Tab. 2-2Výpis niektorých funkčných kódov MODBUS. [1]

				Kódy funkcií		
				Kód	hex	Podfunkcia
Prístup k dátam	Bitový prístup	Fyzické diskrétne vstupy	Čítaj diskrétne vstupy	02	02	
		Interné bity alebo fyzické cievky	Čítaj cievky	01	01	
			Zapíš jednu cievku	05	05	
			Zapíš viac cievok	15	0f	
	16 - bitový prístup	Fyzické vstupné registre	Čítaj vstup registru	04	04	
		Interné registre alebo fyzické výstupné registre	Čítaj uchovávajúce registre	03	03	
			Zapíš jeden register	06	06	
			Zapíš viac registrov	16	10	
			Čítaj/zapíš viacej registrov	23	17	
			Zapíš register s maskovaním	22	16	
			Čítaj FIFO frontu	24	18	
	Prístup k záznamu v súboroch		Čítaj záznam zo súboru	20	14	6
			Zapíš záznam do súboru	21	15	6
Diagnostika			Čítaj stav	07	07	
			Diagnostika	08	08	00 - 18, 20
			Čítaj čítač kom. Udalostí	11	0B	
			Čítaj záznam kom. Udalostí	12	0C	
			Povedz identifikáciu	17	11	
			Čítaj identifikáciu zariadenia	43	2B	14
Ostatné			Zapúzdrený prenos	43	2B	13, 14
			CANOpen základný odkaz	43	2B	13

2.7 Popis vybraných kódov funkcie [1]

2.7.1 01 (0x01) Čítaj cievky

Funkcia slúži k čítaniu stavu 1 až 2000 cievok v zariadení. Klient do požiadavky uvedie adresu prvej cievky a počet cievok (adresovanie začína od 0). V jednom bajte odpovedi je zahrnutý stav celkom ôsmich cievok (každý bit predstavuje jednu cievku). Logická 1 = ON, logická 0 = OFF. Stav prvej adresovanej cievky je LSB v prvom bajte.

Tab. 2-3 Prehľad funkcie: 01 - Čítaj cievky.

t ƻ Ő ƛ ŀ Ř ŀ Ø ᶅ			h ŘLJ ƻ Ø Ŝ ř			Chyba	
Kód funkcie	Počiatočná adresa	Počet cievok	Kód funkcie	Počet bajtov	Stavy cievok	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	1 bajt	n bajtov	1 bajt	1 bajt
0x01	0x0000 – 0xFFFF	1 – 2000 (0x7D0)	0x01	N	n = N alebo N+1	0x81	01, 02, 03, 04

Ű سي (ڪ ﺩ ﺗ ﺨ ﺔ ﺘ ﻚ ﺪ ﺝ ﺝ) / ۶ (2.1)

Ak nie je zvyšok po delení nulový tak

Ű سي Ű طى ρ (2.2)

2.7.2 02 (0x02) Čítaj diskrétne vstupy

Funkcia slúži k čítaniu 1 až 2000 diskrétnych vstupov. Klient do požiadavky uvedie adresu prvého vstupu a počet vstupov (adresovanie začína od 0). V jednom bajte odpovedi je zahrnutý stav celkom ôsmich vstupov (každý bit predstavuje jeden vstup). Logická 1 = ON, logická 0 = OFF. Stav prvej adresovaného vstupu je LSB v prvom bajte.

Tab. 2-4 Prehľad funkcie: 02 - Čítaj diskrétne vstupy.

t ƻ Ő ƛ ŀ Ř ŀ Ø ᶅ			h ŘLJ ƻ Ø Ŝ ř			Chyba	
Kód funkcie	Počiatočná adresa	Počet vstupov	Kód funkcie	Počet bajtov	Stavy vstupov	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	1 bajt	N x 1 bajt	1 bajt	1 bajt
0x02	0x0000 – 0xFFFF	1 – 2000 (0x7D0)	0x02	N		0x82	01, 02, 03, 04

Výpočet N je obdobný ako (2.1) a (2.2).

2.7.3 03 (0x03) Čítaj uchovávajúce registre

Funkcia slúži k čítaniu súvislého bloku uchovávajúcich registrov. Klient do požiadavky uvedie adresu prvého registru a počet registrov (adresovanie začína od 0). V odpovedi sú dáta jedného registra interpretované ako dvojica bajtov. Prvý bajt obsahuje 8 najvýznamnejších bitov a druhý bajt obsahuje 8 nižších bitov hodnoty registra.

Tab. 2-5 Prehľad funkcie: 03 - Čítaj uchovávajúce registre.

t ž Ò ă l· Ř l· Ø ł			h ŘLJ ž Ø Ŝ ř			Chyba	
Kód funkcie	Počiatočná adresa	Počet registrov	Kód funkcie	Počet bajtov	Hodnoty registrov	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	1 bajt	N x 2 bajty	1 bajt	1 bajt
0x03	0x0000 – 0xFFFF	1 – 125 (0x7D)	0x03	2 x N		0x83	01, 02, 03, 04

N je počet registrov.

2.7.4 04 (0x04) Čítaj vstupné registre

Funkcia slúži k čítaniu súvislého bloku vstupných registrov. Klient do požiadavky uvedie adresu prvého registru a počet registrov (adresovanie začína od 0). V odpovedi sú dáta jedného registra interpretované ako dvojica bajtov. Prvý bajt obsahuje 8 najvýznamnejších bitov a druhý bajt obsahuje 8 nižších bitov hodnoty registra.

Tab. 2-6 Prehľad funkcie: 04 - Čítaj vstupné registre.

t ž Ò ă l· Ř l· Ø ł			h ŘLJ ž Ø Ŝ ř			Chyba	
Kód funkcie	Počiatočná adresa	Počet registrov	Kód funkcie	Počet bajtov	Hodnoty registrov	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	1 bajt	N x 2 bajty	1 bajt	1 bajt
0x04	0x0000 – 0xFFFF	1 – 125 (0x7D)	0x04	2 x N		0x84	01, 02, 03, 04

2.7.5 05 (0x05) Zapíš jednu cievku

Funkcia slúži k nadstaveniu jedného výstupu do stavu ON alebo OFF. Klient do požiadavky uvedie adresu cievky (adresovanie začína od 0) a hodnotu požadovaného stavu. Hodnota 0xFF00 odpovedá stavu ON, hodnota 0x0000 stavu OFF. Iné hodnoty nie sú povolené. Pri úspešnom zápise cievky je odpoveďou kópia požiadavky.

Tab. 2-7 Prehľad funkcie: 05 - Zapíš jednu cievku.

t ž Ò ƛ ł· Ř ł· Ø			h ŘLJ ž Ø Ŝ ř			Chyba	
Kód funkcie	Adresa výstupu	Hodnota výstupu	Kód funkcie	Adresa výstupu	Hodnota výstupu	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	2 bajty	2 bajty	1 bajt	1 bajt
0x05	0x0000 – 0xFFFF	0x0000 alebo 0xFF00	0x05	0x0000 – 0xFFFF	0x0000 alebo 0xFF00	0x85	01, 02, 03, 04

2.7.6 06 (0x06) Zapíš jeden register

Funkcia slúži k zápisu jedného uchovávajúceho registru. Klient do požiadavky uvedie adresu registra (adresovanie začína od 0) a hodnotu ktorú má doň zapísať. Pri úspešnom zápise registra je odpoveďou kópia požiadavky.

Tab. 2-8 Prehľad funkcie: 06 - Zapíš jeden register.

t ž Ò ƛ ł· Ř ł· Ø			h ŘLJ ž Ø Ŝ ř			Chyba	
Kód funkcie	Adresa registra	Hodnota registra	Kód funkcie	Adresa registra	Hodnota registra	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	2 bajty	2 bajty	1 bajt	1 bajt
0x06	0x0000 – 0xFFFF	0x0000 – 0xFFFF	0x06	0x0000 – 0xFFFF	0x0000 – 0xFFFF	0x86	01, 02, 03, 04

2.7.7 15 (0x0F) Zapíš viac cievok

Funkcia slúži k nadstaveniu stavu 1 až 1968 cievok. Klient do požiadavky uvedie adresu prvého výstupu a počet požadovaných výstupov (adresovanie začína od 0). Hodnoty výstupov sú uvedené v poli dát. Stavu ON odpovedá bitová hodnota logická 1 a stavu OFF zasa logická 0. V prvom bajte dát sú prenášané stavy prvých ôsmich výstupov. V nasledujúcom bajte ďalších osem atď. LSB každého bajtu predstavuje stav prvého výstupu z osmice. Normálna odpoveď vracia kód funkcie, počiatočnú adresu a počet nadstavených cievok.

Príklad zápisu 13 cievok s počiatočnou cievkou 10 a hodnotou výstupov 0xAD0A.
Prvý bajt odpovedá adresám 17 – 10, LSB odpovedá adrese 10.
Druhý bajt odpovedá adresám 18 – 22, LSB odpovedá adrese 18.
0xAD0A binárne: 1010 1101 0000 1010.

Stav:	1	0	1	0	1	1	0	1	0	0	0	0	1	0	1	0
Výstup:	17	16	15	14	13	12	11	10	–	–	–	22	21	20	19	18

Nepoužité bity v poslednom bajte by mali byť nulové.

Tab. 2-9 Prehľad funkcie: 06 - Zapíš viac cievok.

t ž Ò ǎ ł Ř ł Ø ļ ł					h ŘLJ ž Ø Ŝ ř			Chyba	
Kód funkcie	Počiatočná adresa	Počet výstupov	Počet bajtov	Hodnota výstupov	Kód funkcie	Počiatočná adresa	Počet výstupov	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	N bajtov	1 bajt	2 bajty	2 bajty	1 bajt	1 bajt
0x0F	0x0000 – 0xFFFF	1 – 1968 (0x07B0)	N		0x0F	0x0000 – 0xFFFF	1 – 1968 (0x7B0)	0x8F	01, 02, 03, 04

Výpočet N je obdobný ako (2.1) a (2.2).

2.7.8 16 (0x10) Zapíš viac registrov

Funkcia slúži k zápisu súvislého bloku registrov (1 až 123 registrov). Klient do požiadavky uvedie adresu prvého registru a počet registrov (adresovanie začína od 0). Požadované hodnoty sú uvedené v poli dát, kde dva bajty odpovedajú hodnote jedného registra. Normálna odpoveď vracia kód funkcie, počiatočnú adresu a počet zapísaných registrov.

Tab. 2-10 Prehľad funkcie: 16 - Zapíš viac registrov.

t ž Ò ǎ ł Ř ł Ø ļ ł					h ŘLJ ž Ø Ŝ ř			Chyba	
Kód funkcie	Počiatočná adresa	Počet registrov	Počet bajtov	Hodnota registrov	Kód funkcie	Počiatočná adresa	Počet registrov	Kód funkcie	Chybový kód
1 bajt	2 bajty	2 bajty	1 bajt	N x 2 bajty	1 bajt	2 bajty	2 bajty	1 bajt	1 bajt
0x10	0x0000 – 0xFFFF	1 – 123 (0x007B)	2 x N		0x10	0x0000 – 0xFFFF	1 – 123 (0x007B)	0x90	01, 02, 03, 04

N je počet registrov.

2.8 Záporné odpovede

Pri každom odoslaní požiadavky, klient očakáva spätnú odpoveď servera. Môžu nastať štyri situácie:

- Server príjme požiadavku bezchybne a je schopný ju normálne spracovať, vráti klientovi normálnu odpoveď.
- Server nedostane žiadnu požiadavku z dôvodu chyby komunikácie, nevracia žiadnu odpoveď. Na strane klienta dôjde k vypršaniu časového limitu pre odpoveď.
- Server príjme požiadavku, ale zistí chybu na základe kontrolného súčtu, nevracia žiadnu odpoveď. Na strane klienta dôjde k vypršaniu časového limitu pre odpoveď.
- Server príjme požiadavku bezchybne, ale nie je schopný ju spracovať, vráti klientovi zápornú odpoveď spolu s chybovým kódom.

Rozdiel zápornej a normálnej odpovede je v MSB kódu funkcie. V normálnej odpovedi má MSB kódu funkcie vždy hodnotu logickej 0. V prípade zápornej odpovede je tento bit nadstavený na hodnotu logickej 1. Takto je klientský program schopný ľahko rozpoznať, že sa jedná o zápornú odpoveď. Spolu s kódom funkcie je v dátovej časti zápornej odpovede aj chybový kód (*Exception code*), ktorý udáva príčinu chyby viď tab. 2-11.

Tab. 2-11 Popis chybových kódov. [1]

Kód	Názov (anglicky)	Význam
01	ILLEGAL FUNCTION	Server nepodporuje požadovanú funkciu.
02	ILLEGAL DATA ADDRESS	Požadovaná adresa registrov je mimo rozsah servera.
03	ILLEGAL DATA VALUE	Chybná hodnota dát v poli Dáta.
04	SERVER DEVICE FAILURE	Došlo k chybe pri vykonávaní požiadavky.
05	ACKNOWLEDGE	Potvrdenie prijatia požiadavky. Spracovanie bude trvať dlhšie ako doba čakania na odpoveď.
06	SERVER DEVICE BUSY	Server je zaneprázdnený.
08	MEMORY PARITY ERROR	Došlo k chybe pri práci so súborom.
0A	GATEWAY PATH UNAVAILABLE	Súvisí s použitím brány. Brána nie je schopná vyhradiť prenosovú cestu od vstupného portu k výstupnému.
0B	GATEWAY TARGET DEVICE FAILED TO RESPOND	Súvisí s použitím brány. Cieľové zariadenie neodpovedá.

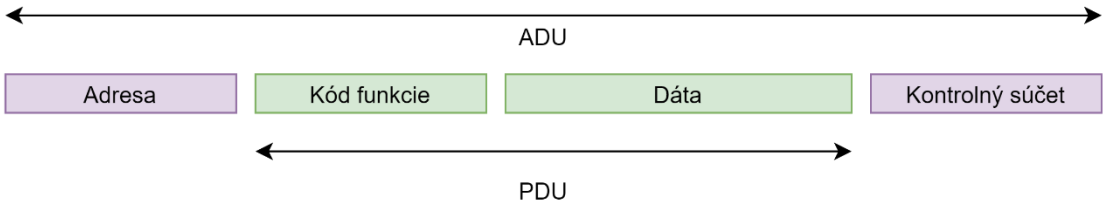
3. MODBUS SERIAL LINE [4]

MODBUS Serial Line pracuje na druhej vrstve referenčného modelu ISO/OSI. Fyzická vrstva môže byť realizovaná rôznymi sériovými rozhraniami, napríklad RS-232, RS-485 a iné.

Protokol je typu master/slave. K zbernici môže byť pripojené iba jedno zariadenie master a niekoľko zariadení typu slave (maximálne 247). Uzol master zadáva príkazy, ktoré majú byť vykonané na uzloch slave. Komunikácia je vždy inicializovaná zariadením master. Slave zariadnia teda nemôžu vysielať žiadne dáta, bez prijatia požiadavky master zariadenia a teda nie sú schopné medzi sebou komunikovať.

3.1 Opis ADU rámca

Na sériovej zbernici je základná dátová jednotka PDU rozšírená o pole „Adresa“, ktoré obsahuje adresu zariadenia slave a o pole „Kontrolný súčet“ obsahujúce hodnotu kontrolného súčtu CRC alebo LRC v závislosti na zvolenom type vysielacieho režimu.



Obr. 3-1 Rámec ADU MODBUS SERIAL LINE. [4]

3.2 Adresácia MODBUS SERIAL LINE

Pole „Adresa“ má veľkosť 1 bajt, čo definuje aj veľkosť adresného priestoru na 256 rôznych adries. V sieti MODBUS majú adresu iba zariadenia typu slave, každé má svoju jedinečnú adresu, zariadenie typu master nemá žiadnu adresu.

Adresný priestor pre slave zariadenia je rozdelený nasledovne:

- broadcast adresa: 0,
- individuálne adresy: 1 – 247,
- rezervované adresy: 248 – 255.

3.3 Režimy prenosu

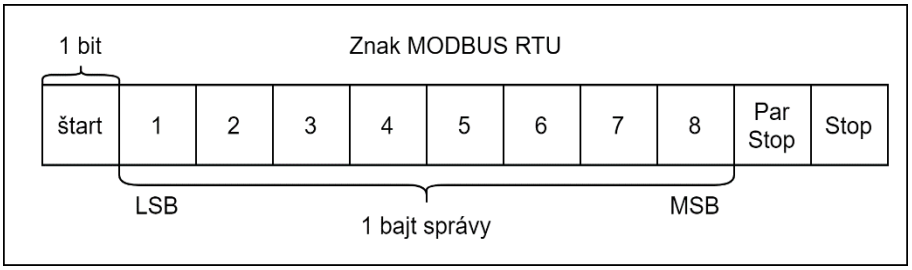
MODBUS SERIAL LINE [4] definuje dva režimy prenosu: RTU (Remote Terminal Unit) a ASCII (American Standard Code for Information Interchange). Prenosový režim určuje formát správy, ako bola zakódovaná a ako má byť dekódovaná. Výrobcovia vo svojich zariadeniach musia vždy implementovať režim RTU, režim ASCII je voliteľný. Zariadenia by mali mať prednastavený režim RTU. Na zbernici MODBUS musia všetky zaradenia pracovať v rovnakom prenosovom režime.

3.3.1 Režim RTU

V režime RTU je každý bajt správy reprezentovaný dvojicou hexadecimálnych znakov. Hlavnou výhodou tohto režimu oproti ASCII je, že dosahuje vyššiu dátovú priepustnosť pri rovnakej modulačnej rýchlosti.

Každá správa musí byť prenášaná ako súvislý prúd MODBUS znakov. Medzi jednotlivými znakmi nesmie byť medzera väčšia ako 1,5 znaku a rozostup medzi správami musí byť minimálne 3,5 znaku.

Každý MODBUS znak je tvorený jedenástimi bitmi vid. obr.3-2.



Obr. 3-2 Sekvencia bitov, znaku MODBUS RTU. [4]

Parita môže byť párna alebo nepárna. Každé zariadenie musí podporovať aspoň párnu paritu. V prípade že nie je parita použitá je tento bit nahradený ďalším stop bitom.

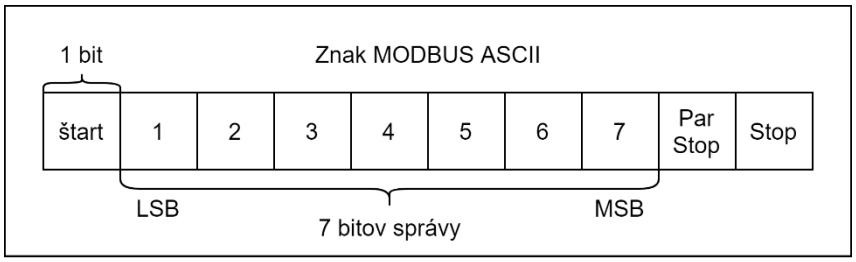
Kontrolný súčet

Pole má veľkosť 2 bajty. Na výpočet kontrolné súčtu sa v režime RTU používa metóda CRC [4] s generujúcim polynómom $x^{16} + x^{15} + x^2 + 1$. Do výpočtu sú zahrnuté iba bity ADU. Štart, stop a parity bity nie sú do výpočtu zahrnuté.

3.3.2 Režim ASCII

V režime ASCII je každý bajt správy reprezentovaný dvojicou ASCII znakov. Medzi jednotlivými MODBUS znakmi môže byť medzera až 1 sekunda. Každá správa musí začínať znakom „ : “ a je ukončená dvojicou znakov CR a LF.

Každý MODBUS znak je tvorený desiatimi bitmi vid. obr. 3-2.



Obr. 3-3 Sekvencia bitov, znaku MODBUS ASCII [4]

Parita môže byť párna alebo nepárna. Každé zariadenie musí podporovať aspoň párnu paritu. V prípade že nie je parita použitá je tento bit nahradený ďalším stop bitom.

Kontrolný súčet

Pole má veľkosť 2 bajty. Na výpočet kontrolné súčtu sa v režime ASCII používa metóda LRC [4]. Do výpočtu sú zahrnuté iba bity ADU. Znaky „ CR, LF a : “ a paritné bity nie sú do výpočtu zahrnuté.

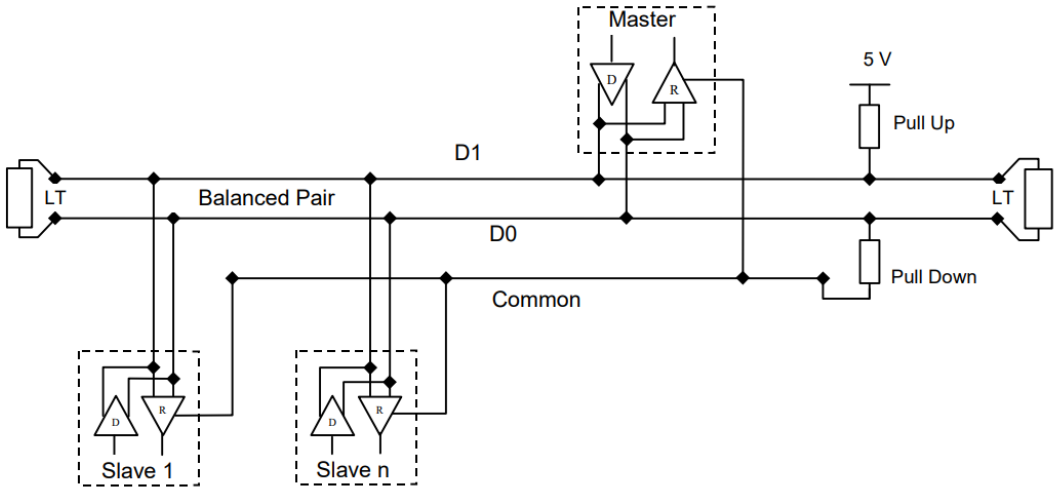
3.4 Fyzická vrstva

Podľa novej normy, by MODBUS Serial Line [4] mal implementovať rozhranie so štandardom EIA/TIA-485 [6] tiež známe ako RS-485. Táto norma umožňuje spojenie bod-bod a viacbodové spojenie v dvojvodičovej alebo štvorvodičovej implementácií. Zariadenie môže implementovať aj rozhranie RS-232.

V štandardnom MODBUSE sú všetky zariadenia paralelne pripojené ku zbernici tvorenej troma vodičmi, z ktorých dva predstavujú symetrický krútený pár (dvojvodičová konfigurácia) pre obojsmernú komunikáciu. Typická bitová rýchlosť je 9600 bitov za sekundu.

3.4.1 Dvojvodičové pripojenie

Dvojvodičové pripojenie musí implementovať rozhranie podľa štandardu EIA/TIA-485. V jeden okamžik môže na zbernici vysielať iba jedno zariadenie. Tretí vodič musí spájať všetky zariadenia.



Obr. 3-4 Dvojvodičové zapojenie. [4]

Vodiče D0 a D1 predstavujú podľa normy EIA/TIA-485 vodiče B/B‘ a A/A‘. Vodič Common predstavuje vodič C/C‘ a môže byť použitý ako napájací alebo signálový vodič. Rezistor LT (line termination) predstavuje ukončenie vedenia.

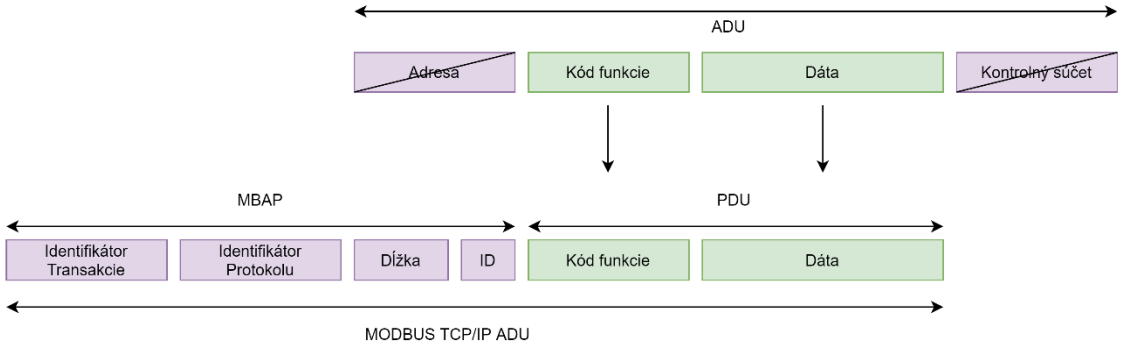
Štvorvodičové zapojenie viď príloha 1.

4. MODBUS TCP/IP [7]

Ako bolo spomenuté, protokol MODBUS je protokol aplikačnej vrstvy a preto je potrebné implementovať súbor mechanizmov a pravidiel pre prenos prostredím TCP/IP. O túto skutočnosť sa stará protokol MODBUS TCP/IP využívajúci protokoly TCP [8] a IP [9]. Protokol TCP sa primárne stará, aby všetky pakety údajov boli bezchybne prijaté, zatiaľ čo protokol IP zabezpečuje správnu adresáciu a smerovanie správ v prostredí TCP/IP a Ethernet . Tieto protokoly žiadnym spôsobom nezasahujú do obsahu dát, starajú sa iba o ich prenos, interpretáciu dát má na starosť protokol MODBUS. Z toho vyplýva, že MODBUS TCP/IP kombinuje reprezentáciu dát MODBUS so sieťovými štandardmi TCP/IP a fyzickou sieťou Ethernet.

4.1 Štruktúra MODBUS TCP/IP

Keďže protokol MODBUS TCP/IP využíva štandardy TCP a IP to znamená, že prenášané dáta aplikačnej vrstvy sú zapuzdrené do protokolov nižších vrstiev. V protokole MODBUS TCP/IP je štruktúra rámca ADU odlišná ako v MODBUS Serial Line viď obr. 3-1.



Obr. 4-1 Štruktúra ADU rámca protokolu MODBUS TCP/IP.[7]

Aplikačná dátová jednotka protokolu MODBUS TCP/IP zachováva polia „Kód funkcie“ a „Dáta“ zo základnej štruktúry MODBUS. Pole „Kontrolný súčet“ nie je potrebné, tento súčet je riešený na linkovej vrstve TCP/IP modelu protokolom Ethernet. Pole „Adresa“ je nahradená polom „ID“ v hlavičke aplikačného protokolu MODBUS (MBAP header – Modbus Application Protocol header). Takto skonštruovaný rámec ADU je zapuzdrený do štandardného segmentu protokolu TCP, odosielaným na známom porte 502, ktorý je špeciálne rezervovaný pre MODBUS. Klient a server naslúchajú a prijímajú dáta na tomto porte.

4.1.1 Opis MBAP header

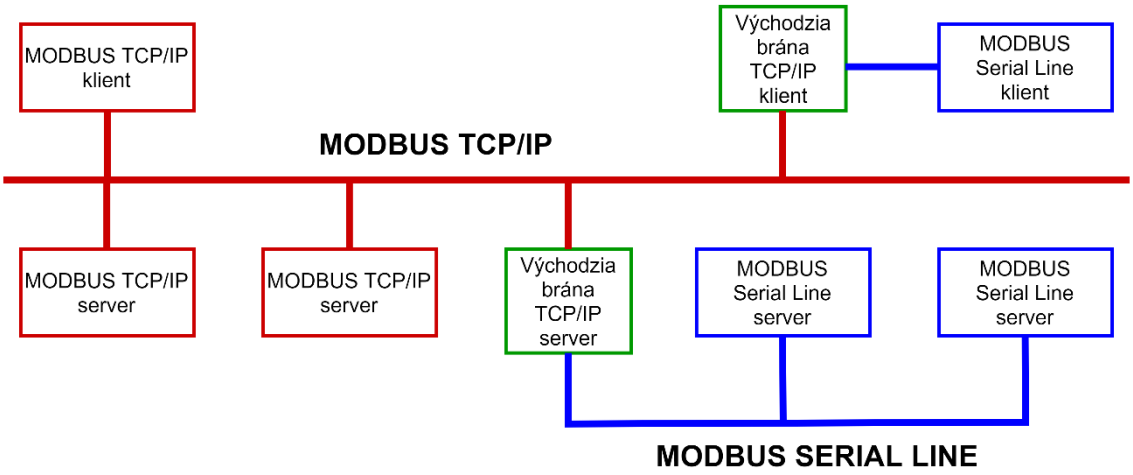
Hlavička MBAP má veľkosť 7 bajtov a obsahuje nasledovné polia:

- **Identifikátor transakcie –** má veľkosť 2 bajty a slúži na párovanie transakcií v prípade, že klient posiela viacej správ v rovnakom TCP spojení bez nutnosti čakania na odpoveď.
- **Identifikátor protokolu** – má veľkosť 2 bajty a pre MODBUS má hodnotu 0. Je rezervované pre budúce použitie.

- **Dĺžka** – má veľkosť 2 bajty a predstavuje počet zvyšných bajtov správy. Zahrňuje polia **ID**, **Kód funkcie** a **Dáta.**
- **ID** – má veľkosť 1 bajt a identifikuje server, ktorý sa nachádza vo vnútornom systéme. Napríklad pri komunikácii cez východziu bránu medzi prostredím TCP/IP a sériovou linkou.

4.2 Adresácia MODBUS TCP/IP

V prostredí TCP/IP môžu byť zariadenia pripojené napriamo alebo cez východziu bránu viď obr. 4-2. Preto je adresácia realizovaná na dvoch úrovniach. Na úrovni IP, kde je zariadenie alebo východzia brána identifikovaná IP adresou. Adresovanie polom ID sa používa pre identifikáciu zariadenia na sériovej linke vo vnútornom systéme prepojeného východzou bránou.



Obr. 4-2 Spôsob pripájania zariadení k MODBUS TCP/IP. [7]

Parametre IP

Každé zariadenie alebo východzia brána musí mať na IP úrovni nadstavené nasledovné parametre:

- lokálna IP adresa,
- maska podsiete,
- predvolená brána.

Adresovanie polom ID

Toto pole sa používa na adresovanie zariadení v podsieti MODBUS Serial Line pripojenej cez východziu bránu. IP adresa identifikuje samotnú východziu bránu a konkrétne zariadenie v podsieti je potom identifikované pomocou poľa „ID“. Pravidlá adresácie pomocou poľa „ID“ sú rovnaké ako pri MODBUS Serial Line.

Zariadenie priamo pripojené k prostrediu TCP/IP je adresované pomocou IP adresy. Pole „ID“ je v tomto prípade zbytočné a jeho hodnota musí byť nadstavená na 0xFF.

5. PREHĽAD DOSTUPNÝCH VÝVOJOVÝCH KITOV

Nasledujúca kapitola bude pojednávať o prehľade vývojových kitov od rôznych výrobcov, ktoré by mohli byť použité pre realizáciu modulu.

Na trhu je dostupná široká škála vývojových kitov rôznych výrobcov od jednoduchých mikrokontrolérov až po výkonné jednodoskové minipočítače. Každý kit je jedinečný a ponúka rôzne možnosti pre vývojára, či už sa jedná o dostupné rozhranie, výkon alebo množstvo pamäte. Výber vhodného vývojového kitu závisí na preferencii vývojára a požiadavkách vyvíjanej aplikácie.

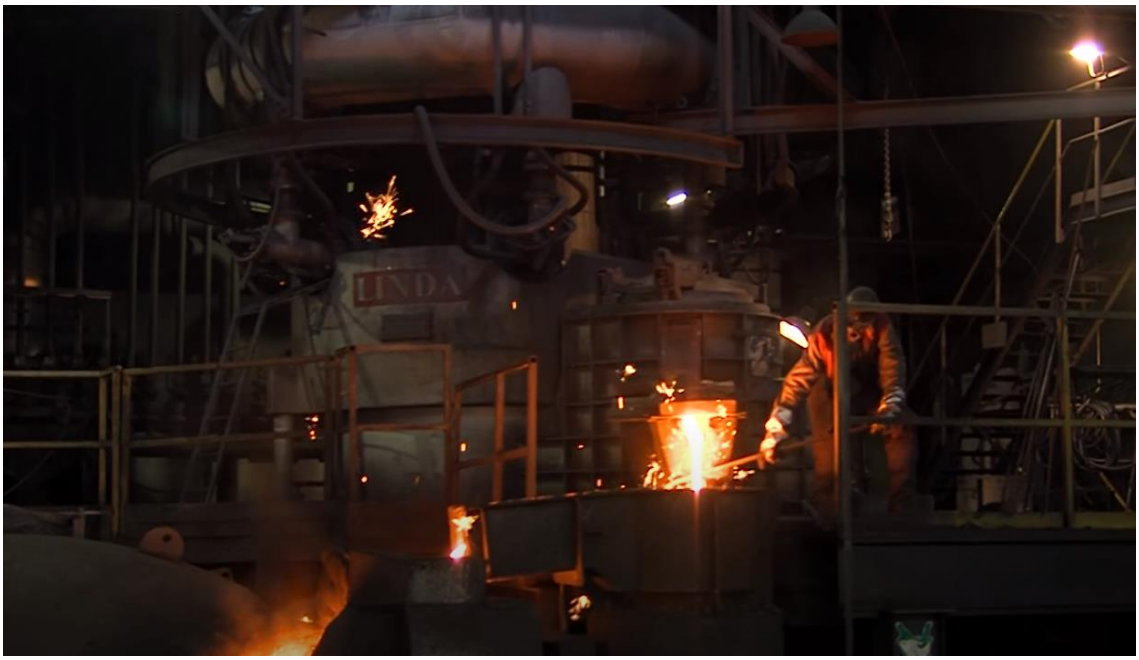
Vzhľadom na nízku úroveň osobných skúseností s používaním mikrokontrolérov, mi bol umožnený prístup k rôznim typom vývojových kitov od rôznych výrobcov pre zoznámenie sa a odskúšanie ich vlastností. Na vybraných kitoch bolo odskúšané ich programovanie jednoduchými príkladmi dostupných z internetu.

Do prehľadu boli vybraté menej známe produkty, ktoré ma zaujali.

5.1 Požiadavky

Navrhovaný modul bude plniť funkciu východzej brány (gateway) pre pripojenie pyrometru s rozhraním RS-485 na WiFi. Preto je nutné aby vývojový kit disponoval rozhraním WiFi a rozhraním umožňujúcim implementáciu sériovej komunikácie.

V prostredí zlievarne sa nachádza veľké množstvo železa, či v tuhej alebo roztavenej forme viď obr. 5-1, ktoré môže nepriaznivo vplývať na WiFi signál. Základná anténa integrovaná na doske vývojového kitu by nemusela postačovať a z toho dôvodu je vhodné aby doska disponovala možnosťou pripojenia externej antény.

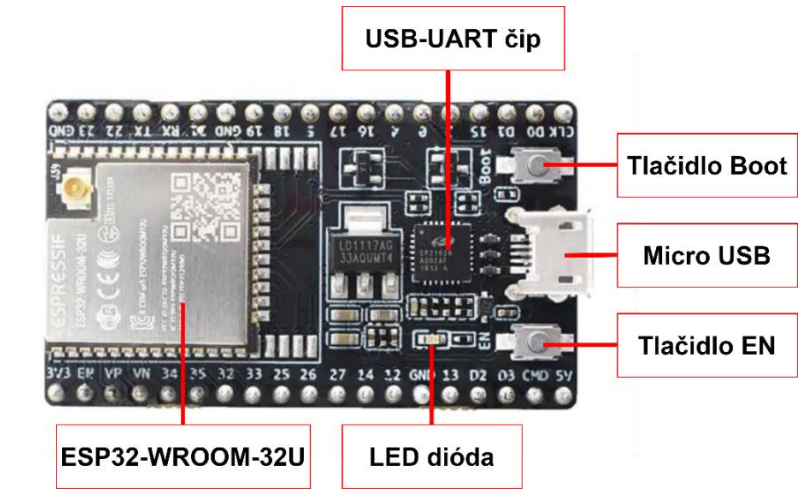


Obr. 5-1 Prostredie zlievarne.

5.2 ESP32-DevKitC-32U [10]

Vývojový kit ESP32-DevKitC V4 je vyrábaný firmou Espressif. Doska má malé rozmery a podporuje rôzne moduly ESP32.

Na doske sa nachádzajú dve tlačidlá: EN a Boot. Tlačidlo EN slúži na reset a Boot tlačidlo na stiahnutie a nahranie programu. Doska je osadená čipom na prevod komunikácie USB na UART. Na komunikáciu s počítačom slúži rozhranie USB typu micro-B . Doska je napájaná jednosmerným napätím 5V. Napájanie môže byť realizované prostredníctvom USB alebo napájacími vstupmi na doske a je indikované LED diódou. ESP32-DevKitC-32U (obr. 5-2) je osadený modulom ESP32-WROOM-32U.



Obr. 5-2 Vývojový kit ESP32-DevKitC-32U

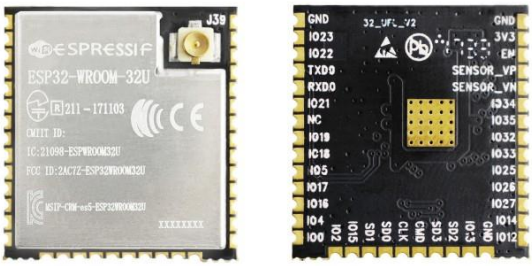
5.2.1 Programovanie platformy ESP32

Mikrokontrolér ESP32 je možné programovať z ľubovoľného operačného systému Windows, Linux alebo macOS pomocou rôznych jazykov v špecifických vývojových prostrediach.

5.2.2 Modul ESP32-WROOM-32U [11]

Modul ESP32-WROOM-32U je výkonný modul s podporou WiFi, BT a BLE zameraný na širokú škálu aplikácií, od sieťových senzorov s nízkou spotrebou až po náročné úlohy. Modul je osadený U.FL konektorom pre pripojenie externej antény.

Jadrom tohto modulu je ESP-D0WD čip, ktorý obsahuje dvoj jadrový 32 bitový procesor Xtensa® LX6. Každé jadro je možné ovládať individuálne s taktovacou frekvenciou od 80MHz do 240MHz. Ďalšie charakteristiky sú uvedené v tab. 5-1.



Obr. 5-3 Modul ESP32-WROOM-32U spredu a zozadu. [12]

Tab. 5-1 Parametre ESP32-WROOM-32U.

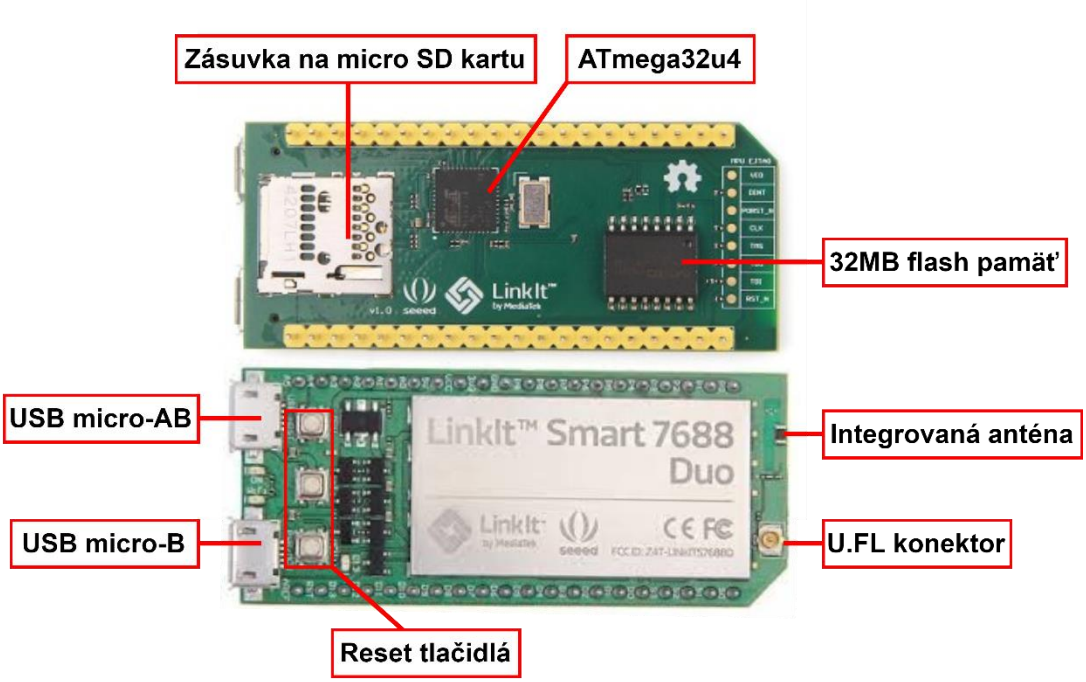
WiFi protokoly	802.11 b/g/n
Integrovaná flash pamäť	4 MB
integrovaný kryštál	40 MHz
Prevádzkové napätie	3 - 3,6 V
Priemerná spotreba prúdu	80 mA
Teplotný rozsah	-40°C až 85°C
Rozhranie	UART, SPI, SDIO, I^2C, I^2S, ADC, DAC

5.3 LinkIt Smart 7688 DUO [13]

Je kompaktná open source vývojová doska, založená na čipovej sade MT7688 a mikrokontroléry ATmega32u4, navrhnutá predovšetkým pre vývoj IoT aplikácií pre domácnosť alebo kanceláriu.

Na doske sa nachádzajú tri tlačidlá pre reset mikroprocesora, mikrokontroléra a Wi-Fi. Ďalej sú na doske dva USB porty typu: micro-B – slúžiace pre napájanie a komunikáciu mikrokontroléra s počítačom a micro-AB – slúžiace ako rozhranie pre pripojenie periférií. Doska je osadená integrovanou anténou a U.FL konektorom pre pripojenie externej antény. Na zadnej strane nájdeme zásuvku na micro SD kartu. Napájanie môže byť realizované prostredníctvom USB (5V) alebo napájacími vstupmi s napätím 3,3V.

Ďalšie špecifikácie sú uvedené v tab. 5-2.



Obr. 5-4 Vývojový kit LinkIt Smart 7688 Duo.

Tab. 5-2 Parametre LinkIt Smart 7688 Duo.

Mikroprocesor	Čipová sada	MT7688AN
	Jadro	MIPS24KEc
	Taktovacia frekvencia	580MHz
Mikrokontrolér	Čipová sada	ATmega32U4
	Jadro	Atmel AVR
	Taktovacia frekvencia	8MHz
Pamäť	Flash	32MB
	RAM	128MB DDR2
WiFi	Protokoly	IEEE 802.11 b/g/n (2.4G)
Externé úložisko	Micro SD karta	
Rozhranie	SPI/SPIS, I2C, UART Lite, USB, ADC, DAC	

5.3.1 Programovanie platformy LinkIt Smart 7688 Duo

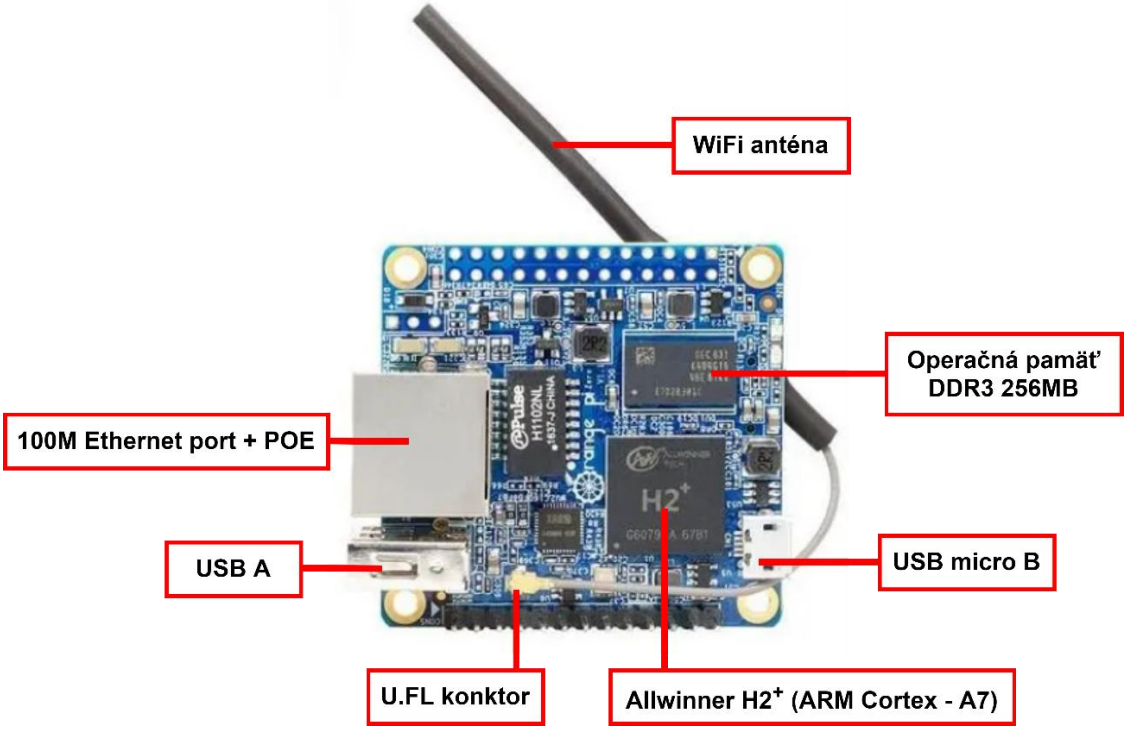
Doska LinkIt Smart 7688 Duo je založená na linuxovej distribúcií OpenWrt a je kompatibilná s Arduino Yún, môže teda využívať aj funkcie tejto dosky. Pre vytváranie aplikácií umožňuje doska využitie programovacích jazykov Phyton, C a Node.js.

5.4 Orange PI Zero 256 [14]

Jedná sa o malý open source jednodoskový minipočítač, ktorý podporuje viacero operačných systémov. Je osadený štvorjadrovým procesorom ARM Cortex – A7 a operačnou pamäťou DDR3 o veľkosti 256 MB.

Na doske sa nachádza rozhranie RJ45 pre Ethernetové pripojenie, ktoré zároveň môže slúžiť pre napájanie dosky prostredníctvom POE. Taktiež doska disponuje možnosťou WiFi pripojenia s využitím externej antény pripojenou U.FL konektorom. Ďalej sa na doske nachádzajú dva USB 2.0 porty. Jeden typu A slúžiaci pre pripojenie periférií a druhé typu micro-B pre napájanie a komunikáciu s počítačom. Potreba pripojenia externých periférií (slúchadlá, mikrofón, USB, TV výstup, IR) je možná prostredníctvom trinástich pinov vyvedených na doske. Doska ponúka aj 26 GPIO pinov. Úložisko je možné rozšíriť pomocou micro SD karty.

Ďalšie špecifikácie sú uvedené v tab. 5-3.



Obr. 5-5 Vývojový kit Orange PI Zero 256

Tab. 5-3 Parametre Orange PI Zero 256.

Grafický procesor	Mali400MP2 GPU @600MHz
WiFi protokoly	IEEE 802.11 b/g/n (2.4G)
Rozhranie	10/100M Ethernet, SPI, UART, I2C, USB
Prevádzkové napätie	5V

5.4.1 Programovanie Orange PI Zero 256

Ako mnohé iné aj platforma Orange PI Zero je možné programovať rôznymi programovacími jazykmi Phyton, Java, C/C++. Avšak minipočítače s označením „PI“ boli vyvíjané so zámerom, že aplikácie budú písané výhradne v Pythone.

5.5 Ďalšie možnosti

Na trhu je dostupné veľké množstvo ďalších vývojových kitov. V prehľade Medzi najznámejšie patria napríklad dosky Arduino alebo Raspberry Pi.

5.5.1 Arduino

Jedná sa o open source platformu založenú na mikrokontroléroch ATmega od spoločnosti Atmel. Programovanie prebieha pomocou vývojového prostredia Arduino IDE s použitím jazyka *Wiring*, ktorý je založený na programovacom jazyku C a C++.

Platforma Arduino ponúka širokú škálu produktov určených k rôznym účelom od domáceho prototypovania, IoT až k modelom určeným pre vzdelávanie.

5.5.2 Raspberry Pi

Raspberry Pi je jednodoskový minipočítač osadený grafickým procesorom a mikroprocesormi architektúry ARM. Primárny operačný systém Paspberry Pi sa nazýva Raspbian, ale podporuje aj iné operačné systémy, ako napríklad rôzne distribúcie Linuxu alebo Windows 10 IoT Core. Ponúka možnosť pripojenia monitoru, prostredníctvom HDMI, a tiež klávesnice a myši pomocou USB. K programovaniu sa využíva hlavne programovací jazyk Phyton ale sú podporované aj ostatné jazyky ako napríklad C, C++.

Platforma ponúka veľké množstvo modelov s rôznym výkonom a dostupnými rozhraniami, určené pre rôzne účely.

Na trhu sú dostupné podobne navrhnuté minipočítače od rôznych výrobcov. Ich názov je odvodený práve od Raspberry Pi. Sú to napríklad: Banana Pi, Orange Pi, Rock Pi atď.

6. HARDWAROVÝ NÁVRH MODUL

WiFi infraštruktúra v zlievarni je zostavená z prvkov firmy Hirschmann, ktoré sú špeciálne zostrojené pre ťažké industriálne prostredie tak, aby boli odolné voči nepriaznivým vplyvom okolia. Z dôvodu aktuálnej situácie ohľadom globálnej pandémie SARS-CoV-2, nebolo možné uskutočniť meranie overujúce spoľahlivosť siete.

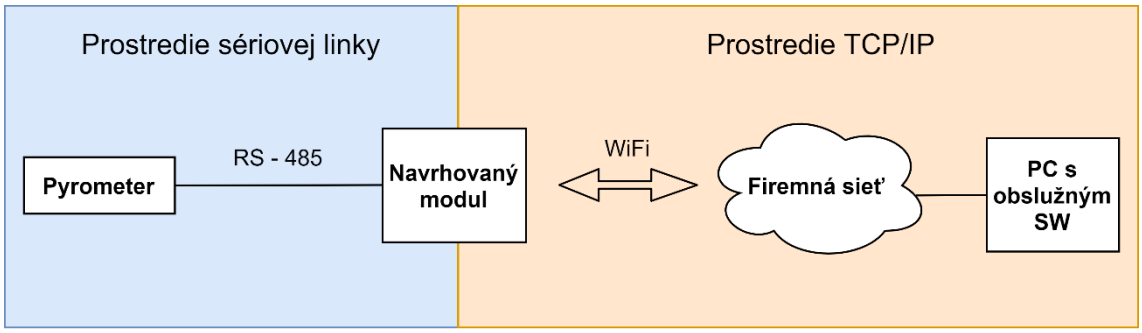
Pri návrhu bolo vychádzané z predpokladu, že je WiFi sieť spoľahlivá a k prenosu bude postačujúci WiFi modul osadený vo vývojovom kite. Avšak v opačnom prípade, bude vývojový kit disponovať alternatívnym pripojením a to eternetovým rozhraním pre možnosť pripojenia výkonného WiFi zariadenia, ktoré zaručí spoľahlivé spojenie.

6.1 Funkcia modulu

Modul bude musieť primárne sprostredkovať komunikáciu medzi pyrometrom firmy Keller typu PA 83 AF 13/C a obslužným softwarom. Pyrometer ponúka pripojenie pomocou konektoru RS-485 s využitím protokolu MODBUS. Software pre obsluhu pyrometru je nainštalovaný na obslužnom počítači, ktorý nie je možné priamo prepojiť s pyrometrom. Počítač je však prepojený k firemnej sieti. Výrobnú halu, teda aj priestor kde je umiestnený pyrometer, pokrýva firemná WiFi sieť. Pre realizáciu bezdrôtového spojenia preto využijeme dostupné WiFi pokrytie. WiFi sprostredkúva prostredie TCP/IP, ktoré je možné využiť aj pre komunikáciu protokolom MODBUS.

Pyrometer so sériovým rozhraním RS-485 ponúka komunikáciu protokolom MODBUS RTU, ktorý však nie je uspôsobený pre komunikáciu v prostredí TCP/IP. Preto navrhovaný modul musí byť schopný previesť komunikáciu po sériovej linke a implementovať dáta do podoby vhodnej pre prenos prostredím TCP/IP. To je možno docieliť s využitím protokolu MODBUS TCP/IP.

Navrhovaný modul bude teda plniť úlohu východzej brány medzi sériovou linkou a TCP/IP prostredím respektíve medzi MODBUS RTU a MODBUS TCP/IP.



Obr. 6-1 Funkcia navrhovaného modulu.

6.2 Výber vývojového kitu

Pre realizáciu modulu bol vybraný vývojový kit ESP32-DevKitC-32U od spoločnosti Espressif. ESP32 je moderný a výkonný mikrokontorlér, ktorý je ľahko programovateľný s využitím frameworku Arduino. Na internete je dostupná široká škála knižníc pre jeho programovanie. Kit má malú spotrebu a disponuje dostatočným výkonom pre realizáciu východzej brány a prípadným rozšírením funkcie.

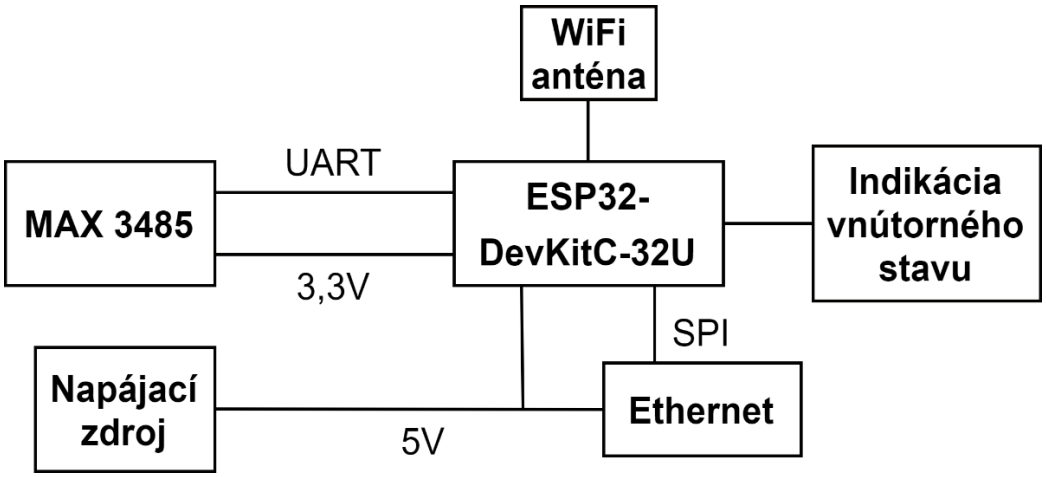
Doska poskytuje veľké množstvo rozhraní viď kapitola 5, avšak neponúka rozhranie typu RS-485 potrebné pre pripojenie pyrometru. Namiesto toho bude využité rozhranie typu UART slúžiace pre asynchrónny prenos dát po dvojlinke. Dátový rámec tohto rozhrania je veľmi podobný rámcu používaného v metóde RTU a z toho dôvodu bude využité práve toto rozhranie.

Bezdrôtová komunikácia bude zaistená integrovaným WiFi modulom s využitím externej antény.

Pre alternatívne pripojenia do siete bude využitý externý ethernetový modul komunikujúci rozhraním SPI.

6.3 Bloková schéma zapojenia

Na obrázku 6-2 je znázornená bloková schéma navrhovaného modulu.

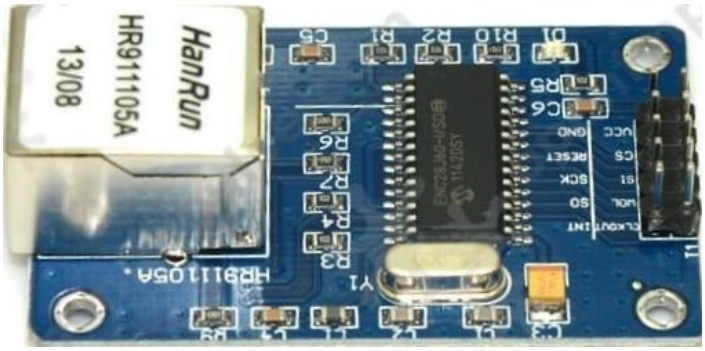


Obr. 6-2 Bloková schéma navrhovaného modulu.

V prípade ak by integrovaný WiFi modul nebol dostatočne spoľahlivý, bude nutné pomocou ethernetového modulu implementovať spoľahlivejší WiFi modul v podobe externého zariadenia, ktoré bude schopné komunikovať v industriálnom prostredí. Napríklad WiFi prístupový bod firmy Hirchmann.

6.4 Ethernetový modul

Modul je založený na integrovanom obvode ENC28J60, ktorý obsahuje všetko potrebné pre realizáciu ethernetového spojenia. Modul komunikuje pomocou rozhrania SPI. Modul je napájaný jednosmerným napätím 5V a obsahuje napäťový delič na napätie 3,3V, s ktorým pracuje integrovaný obvod ENC28J60. Modul plne rešpektuje štandard IEEE 802.3 [15], je kompatibilný so sieťami 10/100/1000 Base-T v polovičnom aj v plnom duplexnom móde.



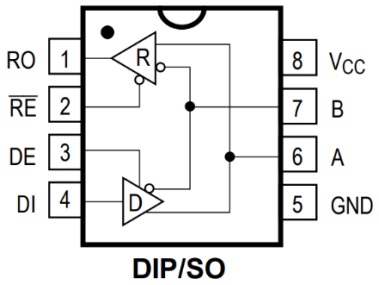
Obr. 6-3 Ethernetový modul s obvodom ENC28J60.

6.5 Prevod UART ↔ RS-485

Rozhranie UART vybraného vývojového kitu využíva TTL logiku, ktorá nie je kompatibilná s rozhraním sériovej linky RS-485 [6]. Preto je nevyhnutný prevod medzi týmito dvoma rozhraniami.

K tomuto prevodu slúži napríklad obvod firmy Maxim integrated, MAX3485.

MAX 3485 [16] je obvod realizujúci prevod medzi logikou TTL a rozhraním RS-485. Poskytuje polo–duplexnú komunikáciu a môže dosiahnuť maximálnu prenosovú rýchlosť 10 Mbps. Je napájaný jednosmerným napätím +3,3V.



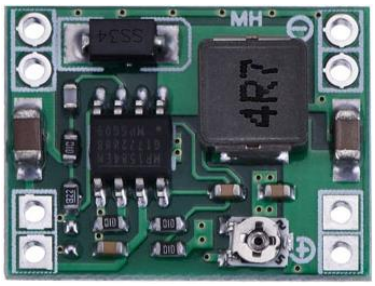
Obr. 6-4 MAX3485.

Popis pinov z obr. 6-3 je nasledovný:

- **RO** – Výstup prijímacej strany. Na výstupe je:
  - logická 1, ak A > B o 200 mV,
  - logická 0, ak A < B o 200 mV.
- **RE** – Slúži na povolenie RO, ak:
  - na vstupe RE je logická 0, tak je RO povolený,
  - na vstupe RE je logická 1, tak je RO nie je povolený.
- **DE –** Slúži na povolenie DI, ak:
  - na vstupe DE je logická 1, tak je DI povolený,
  - na vstupe DE je logická 0, tak je DI nie je povolený.
- **DI** – Vstup vysielacej strany.
- **VCC** – Slúži pre napájanie.
- **B** – Slúži pre pripojenie rozhrania RS-485.
- **A** – Slúži pre pripojenie rozhrania RS-485.
- **GND** – Uzemnenie.

6.6 Napájanie

Zariadenia MODBUS sú napájané jednosmerným napätím +24V, ktoré je obvyklým napájaním v sieťach MAR (Meraní a Regulácií). ESP 32 a eternetový pracujú s jednosmerným napätím +5V. Z tohto dôvodu musí byť napätie +24V zredukované. Túto redukciu zabezpečuje napäťový regulátor OKY3504-2.



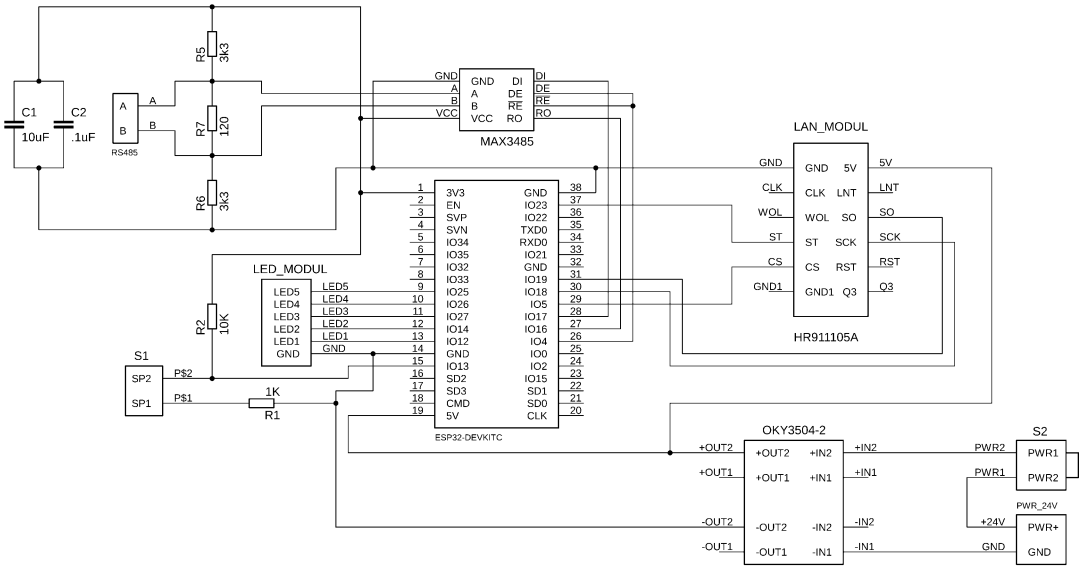
Obr. 6-5 Regulátor napätia OKY3504-2

6.7 Indikácia vnútorného stavu

Pre vizuálne overenie funkčnosti modulu slúžia vyvedené LED diódy. Indikujú či je zariadenie zapnuté, či komunikuje s pripojeným pyrometrom a taktiež pripojenia a typ pripojenia k sieti TCP/IP (Ethernet, WiFi). V prípade poruchy, bude obslužný pracovník schopný ľahšie určiť príčinu.

6.8 Schéma zapojenia

Pre návrh schémy zapojenia bol použitý program EAGLE 9.6.2 so študentskou licenciou. Táto schéma je zobrazená na obr. 6-6.



Obr. 6-6 Schéma zapojenia

6.8.1 Popis zapojenia

Vývojový kit ESP32-DevKitC-32U disponuje 32 GPIO pinmi. Pri návrhu bolo využitých celkovo 13 GPIO pinov.

Napájanie

Vývojový kit je napájaný prostredníctvom jednosmerného napätia 5V, ktoré dodáva napäťový regulátor OKY3504-2. K regulátoru je z konektora **PWR_24V** privedené napätie 24V z vonkajšieho priemyselného rozvodu pre MODBUS. Spínač **S2** slúži k vypnutiu a zapnutiu navrhovaného modulu.

LAN Modul

Ako bolo už spomenuté, ethernetový modul komunikuje prostredníctvom SPI rozhrania k čomu slúžia piny ST, SO, CS a SCK. Vývojový kit poskytuje SPI rozhranie na GPIO: 23 (SPI MOSI), 19 (SPI MISO), 18 (SCK) a 5 (SPI SS). Pin ST je privedený na GPIO_23. Pin SO je privedený na GPIO_19. Na GPIO_18 je privedený pin SCK. Posledný pin CS je privedený na GPIO_5. Modul je napájaný jednosmerným napätím 5V.

MAX3485

Pre komunikáciu s prevodníkom bolo využité rozhranie UART_2 komunikujúce na GPIO_17 (TX) a GPIO_16 (RX), ktoré sú privedené na DI (TX) a RO (RX) piny prevodníka MAX3485. Nakoľko prevodník pracuje iba v polovičnom duplexnom móde je nutné prepínať režim vysielania a prijímania. K tomuto účelu slúžia piny DE a ~~RE~~. Keďže sú piny DE a ~~RE~~ vzájomne negované sú spoločne privedené na GPIO_4, ktorý slúži k ovládaniu smeru komunikácie. S pyrometrom prevodník komunikuje prostredníctvom pinov A a B vyvedených na svorkovnicu. Rezistor R7 slúži ako terminátor linky. Vývojový kit disponuje napäťovým regulátorom s výstupným napätím 3,3V, ktoré je vyvedené aj na pin. Obvod MAX3485 je napájaný napätím 3,3V, ktoré je dostupné práve na zmienenom pine. Kondenzátory C1 a C2 slúžia ako vykrývacie kondenzátory.

LED Modul

Ako už bolo vyššie spomenuté, LED diódy slúžia k indikácii stavu. Anódy diód sú privedené ku GPIO pinom vývojového kitu. Katódy sú cez ochranný odpor 330Ω pripojené k zemi. LED-1 (GPIO_12) indikuje zapnutie zariadenia. LED-2 (GPIO_14) a LED-3 (GPIO_27) indikujú druh pripojenia k TCP/IP sieti. LED-4 (GPIO_26) a LED-5 (GPIO_25) indikujú komunikáciu s pyrometrom.

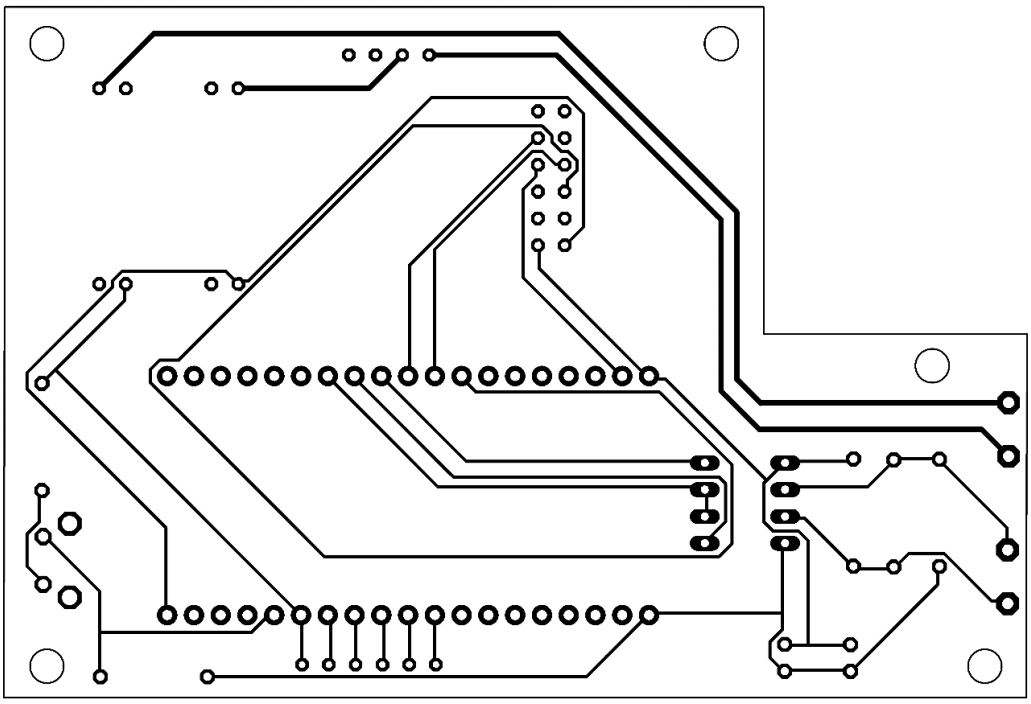
Spínač S1

Spínač slúži na reset nadstavení. Vývojová kit reaguje na zmeny napätia na GPIO_13 pri stlačení spínača.

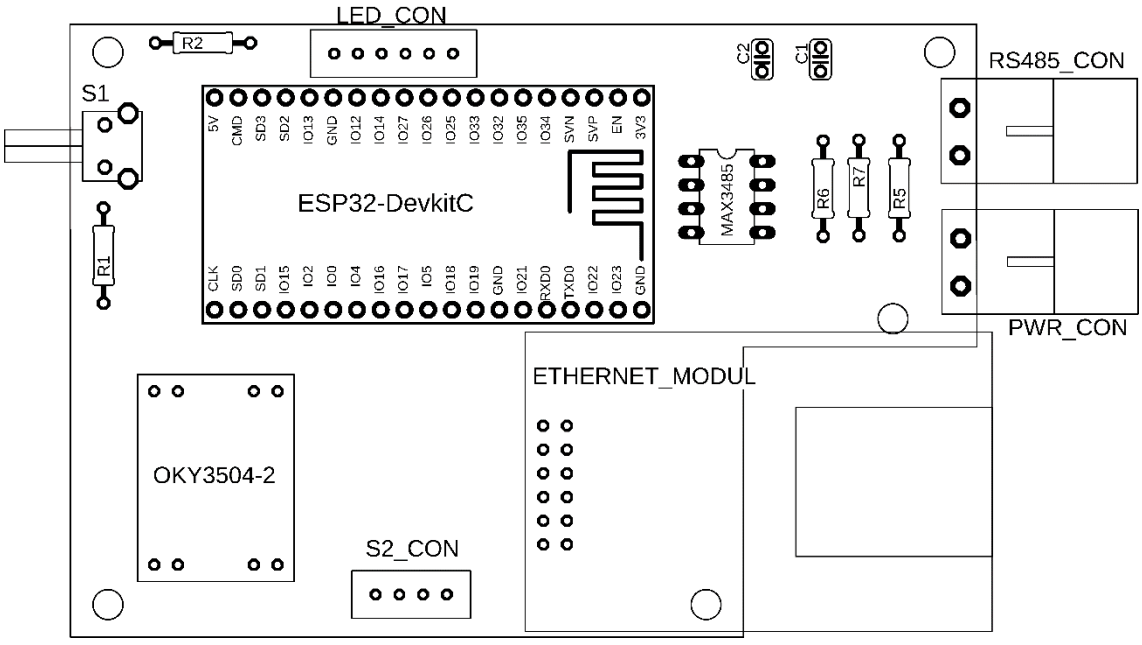
6.9 Návrh dosky plošných spojov

Pre návrh dosky plošných spojov bol využitý program EAGLE 9.6.2 so študentskou licenciou. Doska je navrhnutá s rozmermi 97 x 66 mm. Na doske sa nachádza výrez pod ethernetovým modulom nakoľko je modul osadený dolu hlavou a zásuvka RJ45 by tak zasahovala do dosky. Pre jednoduchosť realizácie boli zvolené bežné vývodové

súčiastky, ktoré boli prepojené automatickým režimom podľa konštrukčnej triedy č.4. DPS zo strany spojov je zobrazená na obr. 6-7 a zo strany súčiastok potom na obr. 6-8.



Obr. 6-7 DPS z pohľadu spojov



Obr. 6-8 DPS z pohľadu súčiastok

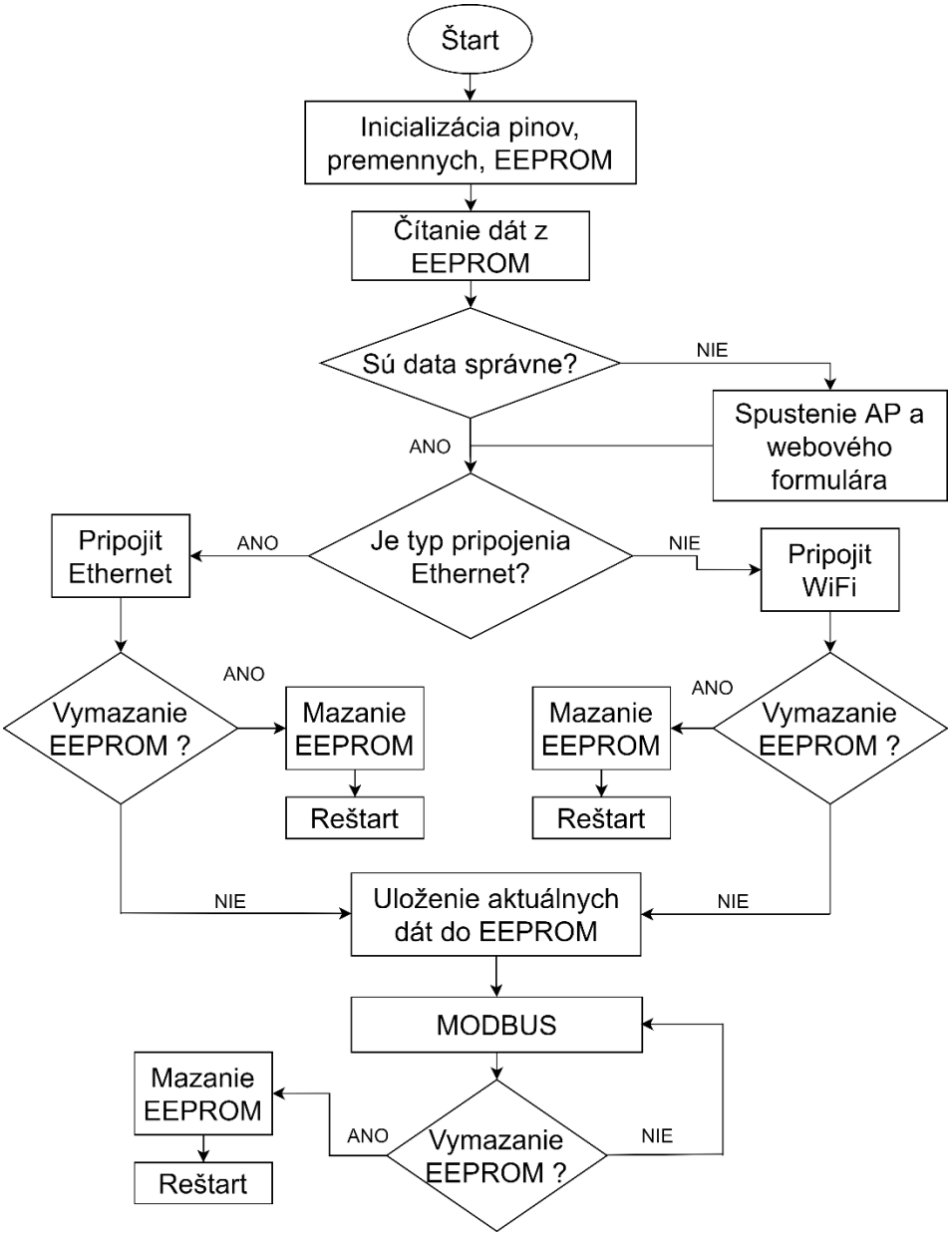
Konektory RS485 a PWR sú schválne umiestnené na okraji DPS aby mohli byť vyvedené z krabičky. Na minimalizáciu vylomenia z DPS sú konektory prilepené o stenu krabičky. Fotografie hotového zariadenia sú dostupné v prílohe č.2.

7. SOFTWAROVÝ NÁVRH MODUL

Programovanie bolo realizované s využitím frameworku Arduino v aplikácií Visual Studio Code.

Hlavnou požiadavkou navrhovaného softwaru bola schopnosť realizácie prekladu komunikácie medzi protokolmi MODBUS RTU a MODBUS TCP/IP. Preto je potrebné pripojenie do siete TCP/IP. K tomuto pripojeniu slúži rozhranie WiFi a Ethernet. Aby bol užívateľ schopný nadstaviť základné parametre a zvoliť spôsob komunikácie (WiFi, Ethernet) bol software rozšírený o webový server so statickou stránkou, ktorá poskytuje rozhranie pre túto konfiguráciu.

7.1 Schéma programu



Obr. 7-1 Schéma navrhnutého programu

7.2 Popis funkcie programových blokov

Program je zostavený zo 6 hlavných častí a využíva princípy objektovo-orientovaného programovania. Navrhnutý program je veľmi rozsiahly, a preto v tejto kapitole budú ukázané iba dôležité časti programu. Celý programový kód je dostupný v elektronickej prílohe.

7.2.1 Main

Programový blok **main** tvorí základný blok pre beh celého softwaru. Main obsahuje dve základné funkcie a to *setup()* a *loop()*. Pre svoju prácu využíva metódy jednotlivých nižšie uvedených tried.

Na začiatku triedy main sú teda pripojené hlavičkové súbory jednotlivých blokov a knižnica Arduino.h.

```
#include <Arduino.h>
#include "my_modbus.h"
#include "my_AP_mode.h"
#include "my_eeprom.h"
#include "my_internet_connection.h"
```

Ďalej sú zadefinované GIOP piny pre kontrolné LED diódy, komunikačné GIOP piny pre rozhranie UART a tiež komunikačná rýchlosť, s ktorou pracuje pyrometer.

```
//uart 2
#define TX 17
#define RX 16
#define BAUD_RATE 57600
//LED piny
#define LED_PIN_POWER 12
#define LED_PIN_WIFI 14
#define LED_PIN_ETHERNET 27
```

Pre jednoduchšiu prácu s programom boli využité princípy objektovo orientovaného programovania a boli vytvorené objekty jednotlivých tried, s ktorými sa ďalej pracuje.

```
//objekty
My_Modbus my_modbus;
My_AP_Mode my_ap_mode;
My_EEPROM my_eeprom;
My_Internet_Connection my_internet_conncetion;
```

V triede main sú definované nasledujúce premenné, ktoré tvoria funkciu ukazovateľa na premenné objektu my_internet_connection.

```
//premenne
String *wifi_name = my_internet_conncetion.getWifiName();
String *wifi_password = my_internet_conncetion.getWifiPassword();
String *connection_type = my_internet_conncetion.getConnectionType();
IPAddress *ip_address = my_internet_conncetion.getIpAddress();
IPAddress *gateway = my_internet_conncetion.getGateway();
IPAddress *mask = my_internet_conncetion.getMask();
```

Význam premenných:

- wifi_name – SSID wifi pripojenia ku ktorému bude zariadenie pripojené,
- wifi_password – heslo k wifi,
- connection_typ – typ pripojenia: WiFi alebo Ethernet,
- ip_addressa – Ipv4 adresa zariadenia,
- gateway – východzia brána smerovača,
- mask – maska siete.

Funkcia setup()

Ako vyplýva z názvu, funkcia má za úlohu nadstaviť základné parametre, obslúžiť pripojenie a EEPROM.

Spustenie sériového rozhrania UART s definovanými parametrami. Parameter SERIAL_8O1 značí spôsob komunikácie UART rozhrania: 8 – značí 8 dátových bitov, O – nepárna parita (odd parity) a 1 – značí jeden stop bit.

Pyrometer pre komunikáciu sériovým rozhraním využíva práve zvolené hodnoty a teda komunikačnú rýchlosť 57600 baud/s a spôsob komunikácie 8O1. [17]

```
//uart 2
Serial2.begin(BAUD_RATE, SERIAL_8O1, RX, TX);
```

Aby užívateľ nemusel po každom spustený zariadenia znova zadávať prihlasovacie údaje k pripojeniu do TCP/IP siete, sú tieto údaje ukladané do EEPROM, ktorú ponúka ESP32. Pred akoukoľvek prácou s ňou musí byť inicializovaná a spustená – funkcia *begin*. Ďalej je volaná funkcia *read*, ktorá vyčíta

```
//start eeprom
my_eeprom.begin();
//citanie eeprom
my_eeprom.read(wifi_name, wifi_password, connection_type, ip_address,
gateway, mask);
```

uložené hodnoty.

Po prečítaní pamäte a uložení údajov je skontrolovaný ich obsah. Ak bola EEPROM prázdna tak premenná connection_type neobsahuje žiadne údaje (`""`) čo značí, že prihlasovací formulár nebol nadstavený. Ak premenná obsahuje `"wifi"` a zároveň nie je

vyplnené SSID, o ktorého kontrolu sa stará funkcia *wifiCredentialsExist* z objektu my_internet_connection, tak užívateľ vybral možnosť pripojenia prostredníctvom WiFi ale nevyplnil údaje o SSID.

V prípade, že nastane jedna z týchto okolností je v navrhovanom module implementovaný WiFi Access Point s prednastaveným SSID, heslom a pevnou IP, ktorý je spustený funkcie *start*.

```
//overenie nadstavenia
//prvý start || wifi s vyplnenym menom wifi_name
if ((*connection_type == "") || (*connection_type == "wifi" && !my_interne
t_conncetion.wifiCredentialsExist()))
    my_ap_mode.start(wifi_name, wifi_password, connection_type, ip_addres
    s, gateway, mask);
```

Ďalej po úspešnom vyplnený formulára nasleduje pripojenie k sieti TCP/IP. Na základe zvoleného typu pripojenia sú volané funkcie z objektu my_internet_connection:

- *connectToEthernet()* – pre pripojenie prostredníctvom Ethernetového modulu,
- *connectToWifi()* – pre pripojenie prostredníctvom WiFi.

Funkcia sa pokúša pripojiť do siete v nekonečnej slučke. Ak je pripojenie úspešné, tak funkcia vracia hodnotu „0“ a rozsvieti sa kontrolná LED dióda. V prípade ak užívateľ vyvolá reset pomocou tlačidla, funkcia vracia hodnotu „1“ a následne sa vymažú údaje z EEPROM a zariadenie sa reštartuje.

```
//vyber pripojenia
bool should_clear_eeprom; //indikátor pre vycistenie eeprom a restart
zariadenia
  if (*connection_type == "ethernet")
  {
    // 1 - vycistenie, 0 - uspesne pripojenie
    should_clear_eeprom = my_internet_conncetion.connectToEthernet();
    if (should_clear_eeprom)
    {
      my_eeprom.clear();
      ESP.restart();
    }
    digitalWrite(LED_PIN_ETHERNET, HIGH);
  }
  else
  {
    // 1 - vycistenie, 0 - uspesne pripojenie
    should_clear_eeprom = my_internet_conncetion.connectToWifi();
    if (should_clear_eeprom)
    {
      my_eeprom.clear();
      ESP.restart();
    }
    digitalWrite(LED_PIN_WIFI, HIGH);
  }
```

Posledným krokom funkcie *setup()* je uloženie údajov do EEPROM.

```
//ulozit do eeprom
my_eeprom.write(wifi_name, wifi_password, connection_type, ip_address,
gateway, mask);
```

Funkcia loop()

Táto funkcia beží do nekonečna a má na starosť hlavnú aplikáciu zariadenia. Volá funkciu *start()* bloku my_modbus, ktorá obsluhuje všetko potrebné k realizácii MODBUS gateway. Funkcia vracia hodnotu „1“ iba ak užívateľ vyžiada reset zariadenia pomocou tlačidla.

```
// 1 - vycistenie
bool should_clear_eeprom = my_modbus.start(connection_type);
  if (should_clear_eeprom)
  {
    my_eeprom.clear();
    ESP.restart();
  }
```

7.2.2 My_eeprom

Programový blok **my_eeprom** má na starosť prácu s EEPROM, ktorá je využívaná k ukladaniu prihlasovacích údajov zadaných užívateľom.
Obsahuje štyri funkcie:

- *begin()* – Funkcia slúži na inicializáciu EEPROM s parametrom potrebnej veľkosti pamäte.
- *write()* – Funkcia slúži pre zápis údajov do EEPROM, funkcii sú predané parametre pomocou ukazovateľov.
  – Využíva základné funkcie z knižnice Arduino.h pre prácu s EEPROM.
- *read()* – Funkcia slúži pre zápis údajov do EEPROM, funkcii sú predané parametre vo forme ukazovateľov.
  – Využíva základné funkcie z knižnice Arduino.h pre prácu s EEPROM.
- *clear()* – Funkcia slúži na mazanie obsahu EEPROM.
  – Cyklus „for“ prechádza celú pamäť a nuluje nenulové pamäťové adresy.

7.2.3 My_AP_mode

Blok **my_AP_mode** zabezpečuje funkciu AP módu a obsluhu web servera. Skladá sa z niekoľkých funkcií.

Funkcia start()

Táto funkcia je volaná z bloku **main** a ako vstupné parametre má ukazovatele na premenné: wifi_name, wifi_password, connection_type, ip_address, gateway a mask. Funkcia nadstaví základné parametre potrebné k vytvoreniu AP a web servera. K AP je možné sa prihlásiť pomocou SSID „Modbus_gateway“ a hesla „123456789“. Webový server naslúcha na porte 80 s IP adresov 192.168.2.1.

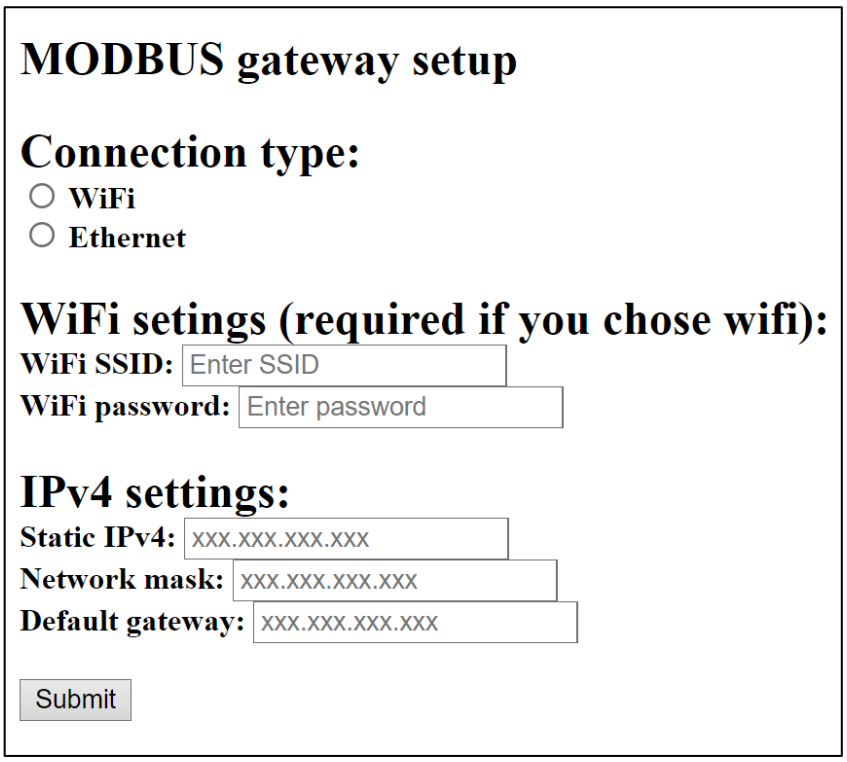
K vytvoreniu AP je využitá funkcia *startAP()*. Obsluhu webové servera má na starosť funkcia *startServer()*.

Funkcia startAP()

Pomocou funkcie *softAPConfig()*, dostupnej z knižnice WiFi.h, nakonfiguruje AP so zvolenými parametrami. Následne funkcia *softAP()* spustí AP. Súčasne sa k zariadeniu môže pripojiť iba jeden užívateľ.

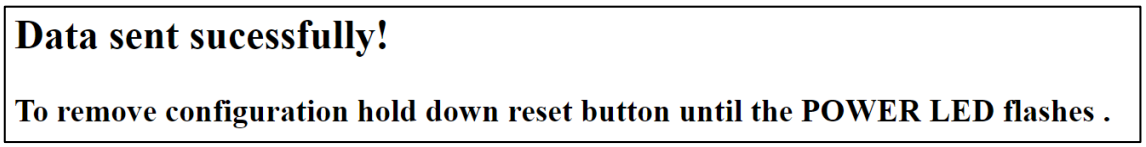
Funkcia startServer()

Funkcia obsluhuje webové rozhranie. Odosiela k užívateľovi webovú stránku s prihlasovacím formulárom viď obr. 7-1. Po úspešnom vyplnení formulára funkcia spracuje odpoveď „GET“, z ktorej sú následne pomocou extrahovacích funkcií vyselektované užívateľom zadané údaje.



Obr. 7-2 Prihlasovací formulár

Po úspešnom odoslaní formulára sa užívateľovi zobrazí nasledovný text viď obr. 7-2.



Obr. 7-3 Potvrdenie úspešného odoslania prihlasovacieho formulára

Extrahovacie funkcie

Extrahovacie funkcie slúžia k načítaniu požadovaných údajov z odpovede „GET“, ktorá je odoslaná po stlačení tlačidla Submit. Každý parameter obsluhuje samostatná funkcia, princíp však majú rovnaký. Ako vstupný parameter slúži práve odpoveď „GET“, ktorá má tvar: „ /wifi?name=H&password=d ”.

Cyklus „for“ prechádza túto odpoveď znak po znaku, ktoré si skladá postupne za seba do pomocnej premennej. Ak narazí na požadovaný výraz napríklad „name=“, spustí sa ďalší cyklus „for“ ktorý opäť prechádza odpoveď znak po znaku, ktoré taktiež zapisuje do pomocnej premennej. Je ukončený ak narazí na znak „&“. Načítaný výraz medzi „name=“ a „&“ odpovedá hodnote, ktorú zadal užívateľ.

7.2.4 My_internet_connection

Tento blok programu má na starosť pripojenie do siete TCP/IP. Obsahuje dve funkcie *connectToWifi()* a *connectToEthernet()*, ktoré obsluhujú pripojenie k WiFi alebo Ethernetu. Funkcia *wifiCredentialsExist()* overuje či užívateľ zadal SSID.

Funkcia connectToWifi()

Nadstaví WiFi do módu stanice a nakonfiguruje jej základné údaje ako IP adresu, gateway a masku podsiete. Následne sa stanica funkciou *begin()* pripojí k požadovanej WiFi sieti na základe SSID a hesla siete, ktoré sú predané funkcii ako parameter. V cykle „while“ sa kontroluje stav pripojenia. Ak je pripojenie úspešné cyklus sa ukončí a funkcia vráti hodnotu „0“. V opačnom prípade je funkciou *shouldClearEeprom()* neustále kontrolovaná požiadavka na vymazanie EEPROM a reštart zariadenia. Ak tak užívateľ učiní funkcia sa ukončí a vracia hodnotu „1“.

Funkcia connectToEthernet()

Funkcia zabezpečuje pripojenie k sieti pomocou Ethernetu. Vo funkcii je volaná funkcia *begin()* so vstupnými parametrami: adresa MAC, IP adresa, adresa DNS, gateway a maska podsiete. Adresa MAC je pevne nadstavená na hodnotu DE-AD-BE-EF-FE-ED a adresa DNS na hodnotu 8.8.8.8. Obdobným spôsobom ako v prípade WiFi pripojenia je kontrolovaný stav pripojenia a požiadavka na vymazanie EEPROM a reštart zariadenia.

7.2.5 My_modbus

Blok **my_modbus** obsluhuje všetky potrebné funkcie k realizácií MODBUS gateway. Skladá sa z 2 hlavných funkcií *start()* a *bridge()* a z niekoľkých pomocných funkcií.

Funkcia start()

Táto funkcia je volaná z bloku **main,** ktorý jej predáva parameter connection_type. Funkcia na začiatku inicializuje výstupné piny pre LED diódy a pin pre ovládanie RX a TX komunikácie na prevodníku MAX3485.

Na základe parametru connection_type je vytvorený buď WiFi server alebo Ethernet server naslúchajúci na porte 502, ktorý je vyhradený pre MODBUS. V nekonečnej slučke „while(1)“ je opäť funkciou *shouldClearEeprom()*, neustále kontrolovaná požiadavka na vymazanie EEPROM a reštart zariadenia. Program v tejto slučke ďalej čaká na príchodzú

komunikáciu na porte 502. Ak je komunikácia úspešne nadviazaná a klient dostupný, spustí sa funkcia *bridge()*.

Funkcia bridge()

Ako prvé funkcia vytvorí premennú tcp_packet_request typu TCP_Packet, ktorej štruktúra je uvedená v hlavičkovom súbore **my_modbus.h**. Do tejto premennej sú pomocnou funkciou *readTcpPacket()* načítané hodnoty „Modbus TCP/IP ADU“ z príchodzieho TCP segmentu.

Následne je vytvorená premenná rtu_packet_request typu RTU_Packet, ktorej štruktúra je opäť uvedená v hlavičkovom súbore **my_modbus.h.** Pomocnou funkciou *createRtuPacket()* sú z premennej tcp_packet_request zapísané potrebné parametre (pre vytvorenie ADU rámca) do premennej rtu_packet_request.

Takto vytvorený rtu rámec je funkciou *writeRtuPacket()* odoslaný rozhraním UART na slave zariadenie. Pred odoslaním je rozsvietená LED dióda indikujúca TX komunikáciu a taktiež je ovládacím pinom zapnutý režim vysielania na prevodníku MAX3485. Po odoslaní správy je prevodník opäť nadstavený do režimu prijímania a LED dióda je opäť zhasnutá.

Pre prijatie rtu odpovede je vytvorená premenná rtu_packet_response typu RTU_Packet, do ktorej sú pomocou funkcie *readRtuPacket()* zapísané dáta prijaté na rozhraní UART. Pred príjmom paketu je rozsvietená LED dióda indikujúca RX komunikáciu a po prijatí všetkých dát je opäť zhasnutá.

Po prijatí dát od zariadenia je vytvorená premenná tcp_packet_response typu TCP_Packet, do ktorej sú z premenných rtu_packet_response a tcp_packet_request pomocou funkcie *createTcpPacket()* uložené potrebné údaje pre vytvorenie Modbus TPC/IP odpovede. Následne je klientovi, pomocou funkcie *writeTcpPacket()*, odoslaná premenná tcp_packet_response, ktorá tvorí odpoveď slave zariadenia.

7.2.6 My_button

Programový blok **my_button** obsahuje jedinú funkciu, a to *shouldClearEeprom()*, ktorá deteguje stlačenie tlačidla pre vymazanie EEPROM a reštart zariadenia. Po stlačení tlačidla funkcia čaká v cykle „while“ po dobu dvoch sekúnd, počas ktorých kontroluje uvoľnenie tlačidla (pre prípad nechceného stlačenia). Pre potlačenie zákmitov spôsobených tlačidlom je pred kontrolou uvoľnenia oneskorenie 10 ms. Po dvoch sekundách je spustená nekonečná slučka „while(1)“, v ktorej je detegované uvoľnenie tlačidla pre úspešné ukončenie funkcie. Aby užívateľ vedel kedy má tlačidlo uvoľniť je v tejto slučke v intervale 30 ms zapínaná a vypínaná LED dióda čo spôsobí efekt blikania, ktorý informuje užívateľa aby uvoľnil tlačidlo.

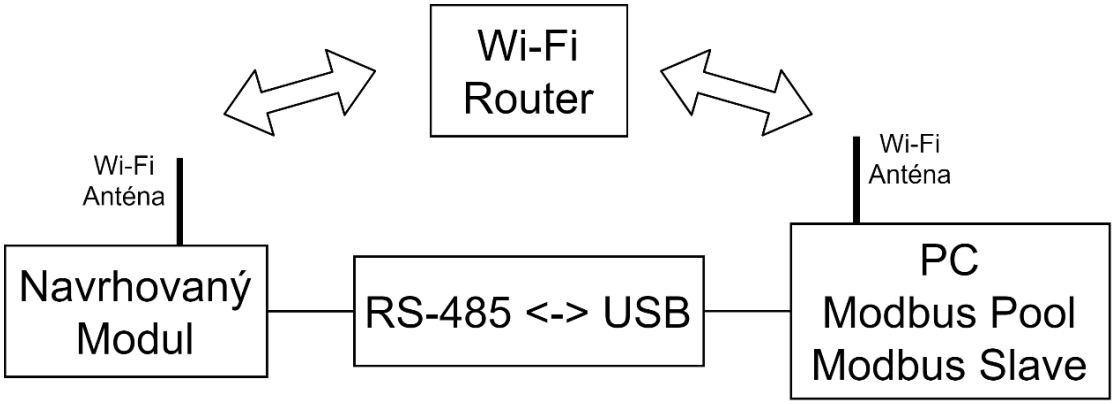
8. TESTOVANIE

Z dôvodu aktuálnej situácie ohľadom globálnej pandémie SARS-CoV-2, nebolo možné zariadenie otestovať v reálnom prostredí Brnenskej zlievarne. Z tohto dôvodu bolo nutné nájsť vhodné alternatívne riešenie, ako otestovať navrhovaný modul v domácom prostredí.

Nakoľko som nemal prístup k žiadnym zariadeniam, ktoré využívajú komunikáciu MODBUS, najvhodnejším spôsobom bolo využiť simulačné programy. Na internete je dostupných mnoho takýchto programov.

Pre potreby testovania boli vybrané dva programy. Prvý program nesie názov „Modbus Poll“ a simuluje MODBUS master zariadenie. Aplikácia poskytuje niektoré základné funkcie MODBUS a taktiež monitor prijatej a odoslanej komunikácie. Druhý program sa nazýva „Modbus Slave“, ktorý ako už názov napovedá simuluje slave zariadenie. Taktiež poskytuje základné funkcie MODBUS a monitor komunikácie. Obe aplikácie sú dostupné z [18].

Na obr. 8-1 je zobrazená bloková schéma testovania navrhovaného modulu. Modul je pripojený k PC pomocou prevodníka rozhraní RS-485 a USB. PC obsluhuje simulačné programy Modbus Pool a Modbus Slave a obe zariadenia sú taktiež pripojené k sieti TCP/IP prostredníctvom WiFi.



Obr. 8-1 Bloková schéma testovania

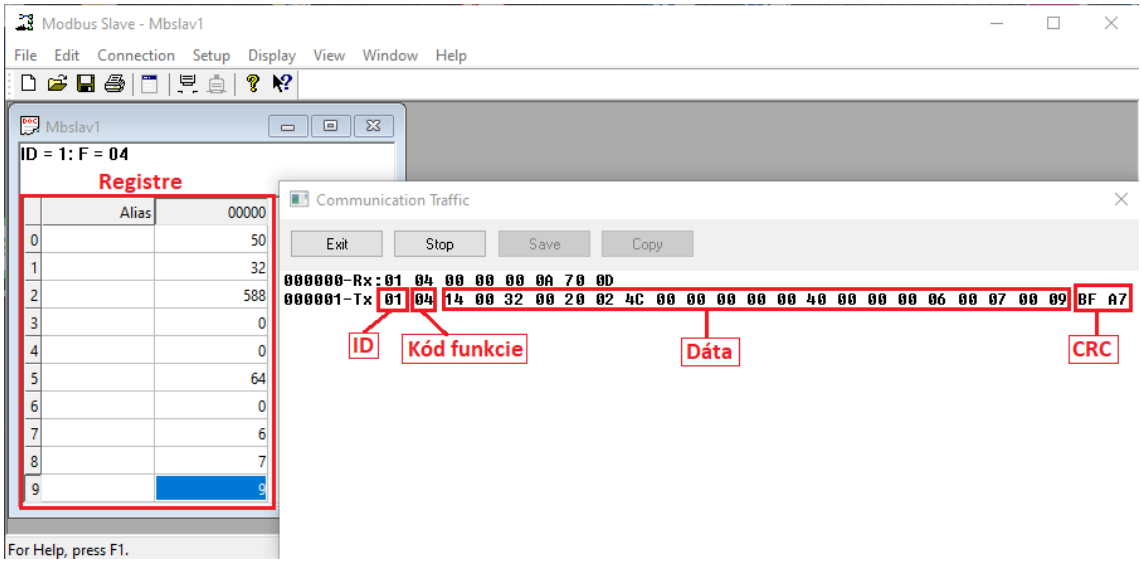
Ukážka testovania

Nasledujúca ukážka demonštruje funkčnosť navrhovaného modulu, pre MODBUS funkciu „0x04“ – čítanie vstupných registrov.

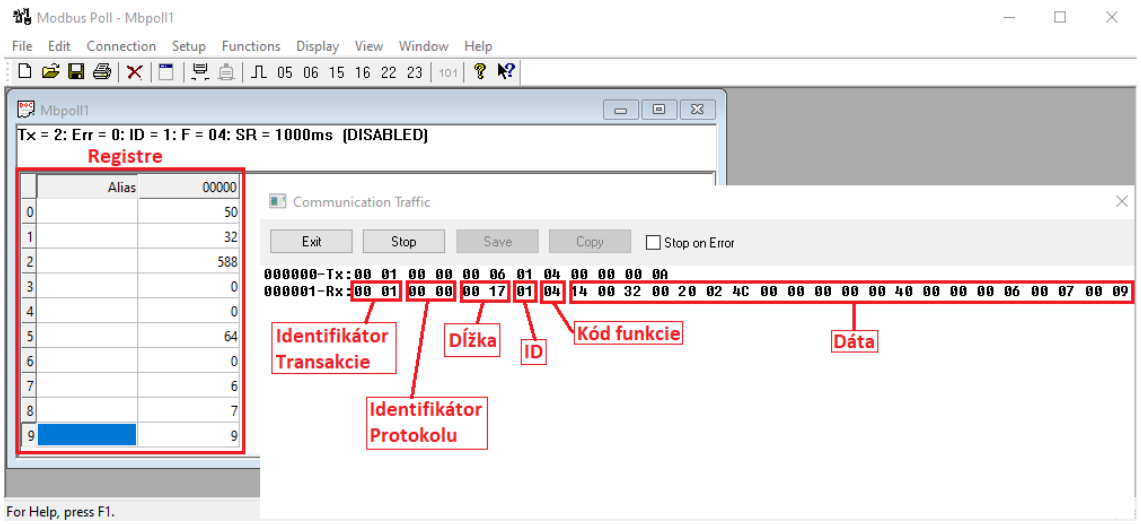
Na obr. 8-2 je zobrazená aplikácia Modbus Slave, ktorá je k navrhovanému zariadeniu pripojená prostredníctvom prevodníka (RS-485←→USB). Na obr. 8-3 je zobrazená aplikácia Modbus Pool, ktorá je pripojená prostredníctvom WiFi.

Aplikácia Modbus Slave má v registroch 0-9 uložené hodnoty. Modbus Pool vyšle navrhovanému modulu požiadavku (Tx:) s kódom funkcie 0x04 pre čítanie hodnôt registrov 0-9. Navrhovaný modul spracuje a odošle aplikácii Modbus Slave túto požiadavku. Aplikácia ju príjme viď obr. 8-2 (Rx:) a následne odošle odpoveď, ktorá obsahuje požadované dáta registrov. Navrhovaný modul spracuje odpoveď do podoby MODBUS TCP/IP a odošle ju aplikácii Modbus Pool, ktorá ich príjme, viď obr. 8-3 (Rx:)

a následne ich zobrazí v odpovedajúcich riadkoch pre registre 0-9. Zobrazené hodnoty registrov sa zhodujú s hodnotami v aplikácii Modbus Slave z čoho vyplýva, že zariadenie pracuje správne a tvorí teda východziu bránu MODBUS.



Obr. 8-2 Aplikácia Modbus Slave



Obr. 8-3 Aplikácia Modbus Pool

9. ZÁVER

Cieľom bakalárskej práce bolo zostrojiť zariadenie, umožňujúce komunikáciu zlievarenského pyrometru prostredníctvom WiFi siete a jeho následná implementácia v reálnom prostredí.

Pyrometer ponúka možnosť prenášania dát rozhraním RS-485, prostredníctvom protokolu MODBUS RTU. Protokol MODBUS je možné prenášať prostredím TCP/IP, ktoré ponúka aj WiFi sieť. Práve táto skutočnosť bola využitá pri návrhu spojenia.

Prvým krokom návrhu bolo podrobné naštudovanie protokolu MODBUS, jeho funkcií a možnosti režimu, ktoré ponúka. Po naštudovaní bolo zistené, že prepojenie sériového rozhrania RS-485, využívaného pyrometrom a prostredia TCP/IP zastúpeného WiFi rozhraním je možné realizovať prostredníctvom východzej brány.

Navrhnuté zariadenie plní práve túto funkciu. Aby bolo zariadenie schopné prekladu medzi MODBUS RTU a MODBUS TCP/IP, musí disponovať určitým výpočtovým výkonom. Ten je zabezpečený mikrokontrolérom.

Výber mikrokontroléru bol pri tomto návrhu kľúčový. Osobné skúsenosti s mikrokontrolérmi boli na nízkej úrovni, preto bolo nutné zoznámiť sa s ich používaním a programovaním. Osobne sa mi najlepšie pracovalo s platformou ESP, ktorá ponúka, rovnako ako platforma Arduino, programovanie s využitím frameworku Arduino. Taktiež je dostupná široká škála knižníc pre rôzne účely. Následne bol zostavený prehľad vývojových kitov ktoré spĺňali požiadavku pre možnosť pripojenia externej WiFi antény.

Pre návrh zariadenia bol vybraný vývojový kit ESP32-DevKitC-32U, ktorý má implementované WiFi rozhranie s možnosťou pripojenia externej antény, avšak nedisponuje rozhraním RS-485. Pre potrebu pripojenia tohto sériového rozhrania bolo využité rozhranie UART, ktoré však nie je kompatibilné s rozhraním RS-485. Preto bolo nutné použiť prevodník týchto rozhraní. K tomuto účelu bol vybraný overený obvod MAX3485, ktorý je využívaný v rôznych aplikáciách dostupných na internete.

S využitím programu EAGLE 9.6.2 bola z vybraných komponentov zostavená schéma zapojenia viď obr. 6-6 a následne vytvorená doska plošných spojov podľa konštrukčnej triedy č.4. Doska bola následne vyrobená externou firmou.

Pre softwarový návrh bola využitá aplikácia Visual Studio Code s využitím frameworku Arduino. Program plní dve hlavné funkcie. Prvou z nich je pripojenie do siete TCP/IP. Aby bolo zariadenie univerzálne a užívateľ ho vedel ľahko pripojiť do siete, disponuje možnosťou nadstavenia parametrov pripojenia pomocou webového prehliadača. Zariadenie vytvorí prístupový bod, ku ktorému sa užívateľ pripojí a po zadaní IP adresy 192.168.2.1 do webového prehliadača sa zobrazí formulár pre nadstavenie pripojenia. Po odoslaní formulára sa údaje uložia do pamäte zariadenia aby ich užívateľ nemusel po vypnutí zariadenia opäť zadávať. Druhou funkciou je samotná aplikácia východzej brány MODBUS medzi rozhraním RS-485 a WiFi sieťou.

Výsledkom tejto práce je funkčné zariadenie, ktoré však nebolo možné otestovať v reálnom prostredí, vzhľadom k aktuálnej pandemickej situácii. Testovanie preto prebehlo v domácom prostredí s využitím simulačných programov „Modbus Pool“ a „Modbus Slave“. Zariadenie je zobrazené v prílohe 2.

Literatúra

[1] *MODBUS APPLICATION PROTOCOL SPECIFICATION*. V1.1b3. Massachusetts: Modbus Organization, 2012. Dostupné také z: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

[2] HORYCH, VLADIMÍR. *ANALÝZA ŘÍDICÍCH PROTOKOLŮ VYUŽÍVANÝCH V PRŮMYSLOVÝCH APLIKACÍCH* [online]. BRNO, 2009 [cit. 2020-10-28]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=15835#page=26&zoom=100,129,260. Diplomová práce. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ. Vedoucí práce ING. MARTIN KOUTNÝ..

[3] ĎUĎÁK, Juraj. *Príspevok k priemyselným komunikačným štandardom* [online]. BRATISLAVA, 2010 [cit. 2020-10-28]. Dostupné z: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiCxebAktfsAhWOsaQKHTFLA6AQFjABegQIAhAC&url=https%3A%2F%2Fis.stuba.sk%2Flide%2Fclovek.pl%3Fzalozka%3D13%3Bid%3D1926%3Bstudium%3D55548%3Bzp%3D21591%3Bdownload_prace%3D1&usg=AOvVaw2iBw6buVE-DFF4B1drMEiy. Disertace. SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ.

[4] *MODBUS over Serial Line: Specification and Implementation Guide*. V1.02. Massachusetts: Modbus Organization, 2006. Dostupné také z: https://modbus.org/docs/Modbus_over_serial_line_V1_02.pdf

[5] Byte Ordering. *Apple Developer* [online]. Kalifornia: Apple, 2009 [cit. 2020-10-28]. Dostupné z: https://developer.apple.com/library/archive/documentation/CoreFoundation/Conceptual/CFMemoryMgmt/Concepts/ByteOrdering.html#//apple_ref/doc/uid/20001150-CJBEJBHH

[6] Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems. TIA-485-A. Virginia: TELECOMMUNICATIONS INDUSTRY ASSOCIATION, 1998.

[7] *MODBUS MESSAGING ON TCP/IP: IMPLEMENTATION GUIDE*. V1.0b. Massachusetts: Modbus Organization, 2006. Dostupné také z: https://modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf

[8] POSTEL, Jon, ed. INFORMATION SCIENCES INSTITUTE. *TRANSMISSION CONTROL PROTOCOL*. September 1981. California: University of Southern California, 1981. Dostupné také z: https://tools.ietf.org/html/rfc793

[9] POSTEL, Jon, ed. INFORMATION SCIENCES INSTITUTE. *INTERNET PROTOCOL*. September 1981. California: University of Southern California, 1981. Dostupné také z: https://tools.ietf.org/html/rfc791

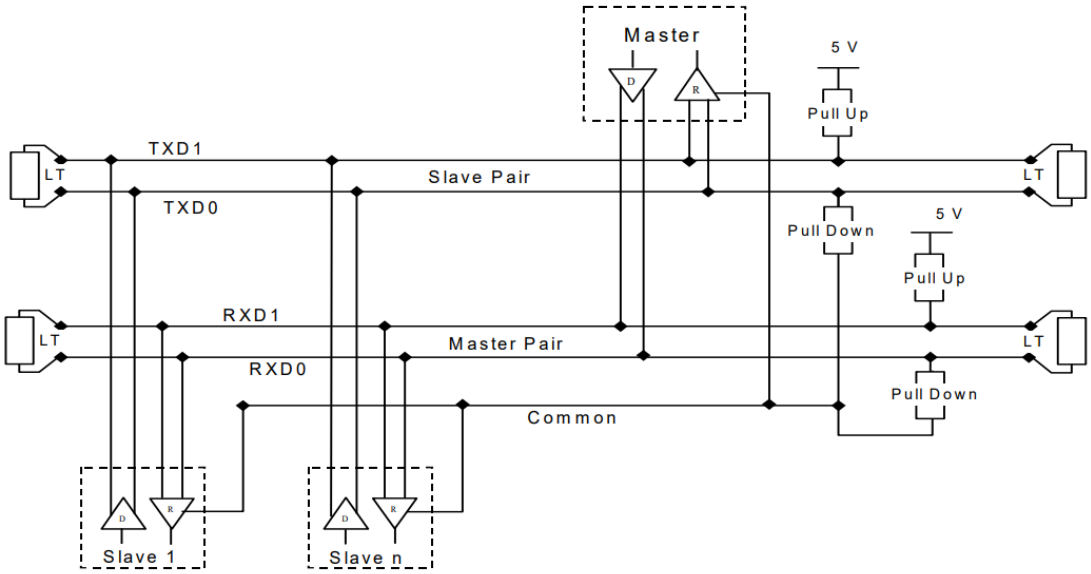
[10] ESP32-DevKitC V4 Getting Started Guide. *ESP-IDF Programming Guide* [online]. Shanghai: Espressif Systems, c2016-2020 [cit. 2020-11-17].

Dostupné z: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html
[11] *ESP32-WROOM-32D & ESP32-WROOM-32U: Datasheet*. Version 1.9. Shanghai: Espressif Systems, 2019. Dostupné také z: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf
[12] ESP32 Modules and Boards. In: *ESP-IDF Programming Guide* [online]. Shanghai: Espressif Systems, c2016-2020 [cit. 2020-11-17]. Dostupné z: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/modules-and-boards.html#esp-modules-and-boards-esp32-wroom-32d-and-u
[13] LinkIt Smart 7688 Duo. *Seeed Wiki* [online]. Shenzhen: Seeed Technology, c2008-2020 [cit. 2020-11-17]. Dostupné z: https://wiki.seeedstudio.com/LinkIt_Smart_7688_Duo/
[14] Orange Pi Zero. *Orange Pi* [online]. Shenzhen: Xunlong Software, c2016 [cit. 2020-11-25]. Dostupné z: http://www.orangepi.org/orangepizero/
[15] IEEE 802.3 Ethernet Working Group. *IEEE 802.3 ETHERNET WORKING GROUP* [online]. Piscataway (New Jersey): Institute of Electrical and Electronics Engineers, 2021 [cit. 2021-04-07]. Dostupné z: https://www.ieee802.org/3/index.html
[16] *3.3V-Powered, 10Mbps and Slew-Rate-Limited*. MAX3485. San Jose: Maxim Integrated, 2019. Dostupné také z: https://datasheets.maximintegrated.com/en/ds/MAX3483-MAX3491.pdf
[17] *Operating manual PA 8x: Pyrometer CellaTemp PA 80, PA 81, PA 83*. 103 398410/2018. Ibbenbüren (Germany): KELLER HCW, 2018. Dostupné také z: https://www.keller.de/dl.php?f=operating-instructions-pa-8x&h=528d6c353293e63L3Zhci93d3cvZnRwL2l0cy9JVFMvw7ZmZmVudGxpY2hlciBCZXJlaWNoIGZyZWllciBadWdhbmcvZW4vTWFudWFscy8wMyBDZWxsYVRlbXAgUEEvTUEgQ2VsbGFUZW1wIFBBOHhfSUQyMDIwX2VuLnBkZg==
[18] Modbus tools: Download. Modbus tools [online]. [cit. 2021-5-13]. Dostupné z: https://www.modbustools.com/download.html

Zoznam Príloh

Príloha 1 - Štvorvodičové zapojenie

MODBUS Serial Line povoľuje implementáciu zbernice ako dve dvojvodičové zbernice, kde dáta na páre (RXD1-RXD0) sú prijímané iba slave zariadeniami a dáta posielané na páre (TXD1-TXD0) sú prijímané iba master zariadením.
Rovnako ako pri dvojvodičovom zapojení, musí aj štvorvodičové zapojenie mať spoločný vodič ku ktorému sú pripojené všetky zariadenia. V jeden okamžik môže na zbernici vysielať iba jedno zariadenie.



Obr. 1-A Štvorvodičové zapojenie.

Vodiče TXD0 a RXD0 predstavujú podľa normy EIA/TIA-485 vodiče A/A‘. Vodiče TXD1 a RXD1 predstavujú vodiče B/B‘. Vodič Common predstavuje vodič C/C‘ a môže byť použitý ako napájací alebo signálový vodič. Rezistor LT (line termination) predstavuje ukončenie vedenia.

Príloha 2 – Fotografie vyrobeného zariadenia

Obr. 2-A Pohľad na zariadenie z vnútra

Obr. 2-B Pohľad na zariadenie zo strany LED diód

Obr. 2-C Pohľad na zariadenie zo strany konektorov

Príloha 3 – Obsah elektronickej prílohy

Obsah elektronickej prílohy:

- Program – obsahuje zdrojové a hlavičkové súbory programu.
- Modbus_wifi_pcb – obsahuje schému zapojenia navrhovaného modulu.
- Modbus_wifi_pcb – obsahuje dosku plošných spoj.