**České vysoké učení technické**

**Fakulta strojní**

**Ústav přístrojové a řídící techniky**

# Vzdálené řízení PLC SCADA systémem pomocí protokolu MODBUS

Bakalářská práce

*Filip Šrámek*

Bakalářský program: Strojírenství

Studijní obor: Informační a automatizační technika

Vedoucí: Ing. Pavel Trnka

Praha, 2018

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně s tím, že její výsledky mohou být dále použity podle uvážení vedoucího bakalářské práce jako jejího spoluautora. Souhlasím také s případnou publikací výsledků bakalářské práce nebo její podstatné částí, pokud budu uveden jako spolu autor.

Dne …….  Podpis…….

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Šrámek** Jméno: **Filip** Osobní číslo: **437145**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídící techniky**

Studijní program: **Strojírenství**

Studijní obor: **Informační a automatizační technika**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Vzdálené řízení PLC SCADA systémem pomocí protokolu MODBUS**

Název bakalářské práce anglicky:

**Remote control of PLC using MODBUS protocol and SCADA**

Pokyny pro vypracování:

1) Nastudujte a popište komunikační možnosti protokolu MODBUS.
2) Prověřte komunikační možnosti programovatelného automatu Tecomat Foxtrot.
3) Prověřte komunikační možnosti nadřazeného SCADA systému myScada myPro.
4) Navrhněte vhodný způsob komunikace mezi SCADA systémem a PLC.
5) Vytvořte spojení a vyzkoušejte.
6) Navrhněte a realizujte demonstrační úlohu seřízení regulátoru na vzdáleném pc.

Seznam doporučené literatury:

[1] Petr Kocourek, Jiří Novák: Přenos informace. Skriptum. ČVUT Praha, 2004.
[2] Modbus application protocol v1.1b. [online], Modbus-IDA, 28.12.2006, Link: <http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf>
[3] Modbus Technical Resources. [online], Modbus-IDA, Link: <http://www.modbus.org/tech.php>

Jméno a pracoviště vedoucí(ho) bakalářské práce:

**Ing. Pavel Trnka, U12110.3**

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **18.04.2018** Termín odevzdání bakalářské práce: **15.06.2018**

Platnost zadání bakalářské práce: _____

| Ing. Pavel Trnka<br>podpis vedoucí(ho) práce | podpis vedoucí(ho) ústavu/katedry | prof. Ing. Michael Valášek, DrSc.<br>podpis děkana(ky) |
|---|---|---|

## III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ Datum převzetí zadání

_____ Podpis studenta

Poděkování

Rád bych touto cestou vyjádřil poděkování panu Ing. Pavlu Trnkovi za čas, který mi věnoval při vedení této bakalářské práce. Mé poděkování patří též panu Mgr. Ing. Jakubovi Jurovi, Ph.D. za neocenitelné rady, které mi poskytl během vypracovávání mé bakalářské práce. V neposlední řadě musím poděkovat své rodině za neocenitelnou podporu během celého studia.

Název: Vzdálené řízení PLC SCADA systémem pomocí protokolu MODBUS

Abstrakt

První část této práce je věnována rešerši o komunikačních protokolech podporovaných produkty firmy Teco a.s. Druhá část je věnována navázání komunikace mezi PLC a SCADA systémem pomocí protokolu MODBUS spolu s řízením soustavy pomocí PID regulátoru SimplePID.

Klíčová slova: MODBUS, SCADA, PLC, řízení, PID regulátor, MOSAIC, Tecomat Foxtrot

Title: Remote control of PLC using MODBUS protocol and SCADA

Abstract

The first part of this bachelors thesis is dedicated to the research of communication protocols of supported products of Teco a.s. The second part is dedicated to the communication between the PLC and the SCADA system using the MODBUS protocol together with the control system using the SimplePID PID controller.

Keywords: MODBUS, SCADA, plc, řízení, PID Controller, MOSAIC, Tecomat Foxtrot

# Obsah

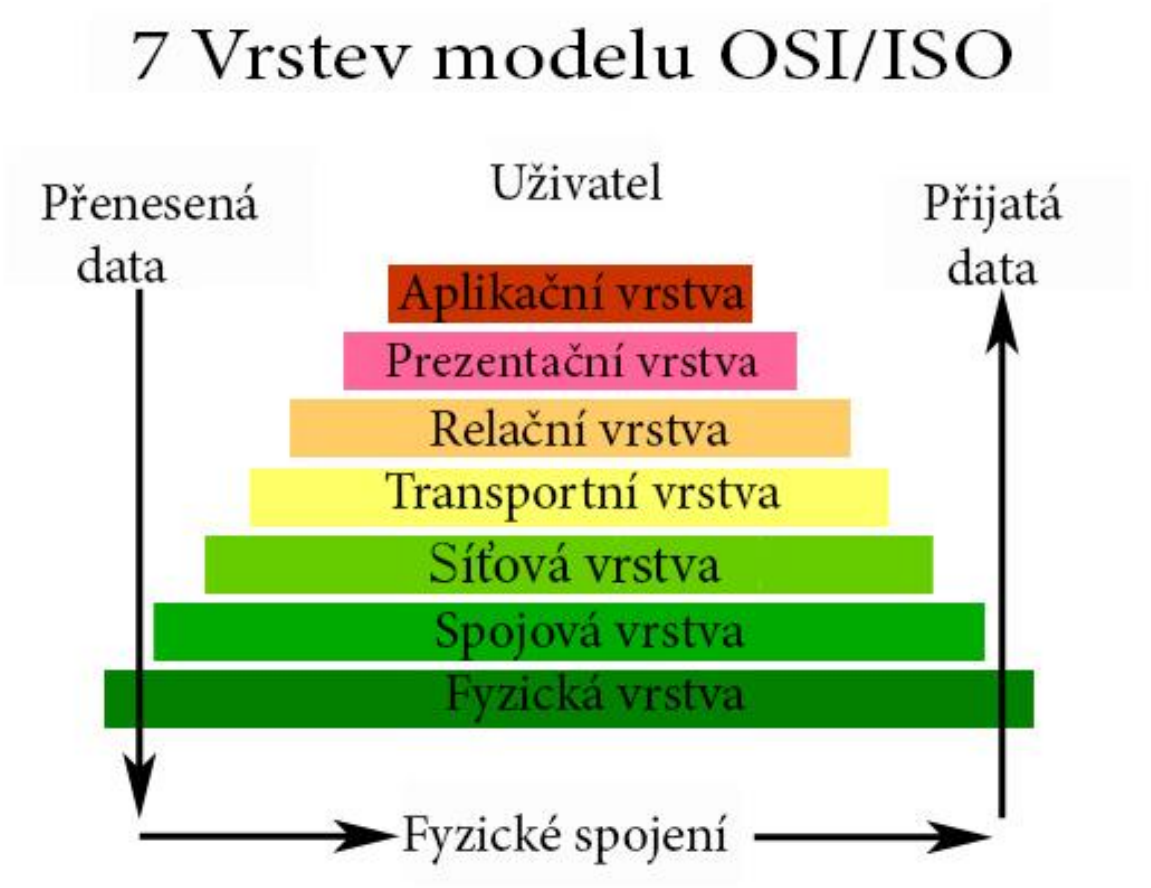
# Úvod

Cílem této práce je zprovoznění komunikace mezi PLC a SCADA systémem pomocí protokolu MODBUS s následnou emulací PID regulátoru uvnitř PLC. Celý proces řízení bude na dálku řízen pomocí SCADA systému dodávaným firmou mySCADA. Pro realizaci této úlohy byl vybrán programovatelný automat Tecomat Foxtrt CP – 1015 vyráběný firmou Teco a.s. Tento automat plně implementuje normu IEC -61131-3, což umožňuje přenositelnost programu i na jiné automaty.

# 1 Vysvětlení pojmů

Model ISO/OSI

Model ISO/OSI je základním referenčním komunikačním modelem označeným zkratkou slovního spojení „International Standards Organization/Open Systém interconnection“, což v překladu znamená Mezinárodní organizace pro normalizaci/propojení otevřených systémů. Tedy jedná se o doporučený model definovaný organizací ISO. Komunikační protokoly jsou řazeny do sedmi vrstev, z nichž každá má svůj specifický úkol, graficky tyto vrstvy znázorňuje obrázek č. 1 [15; 16].




Obrázek č. 1 – 7 Vrstev modelu OSI/ISO; [35]

Fyzická vrstva (Physical layer)

Aplikační vrstva definuje nejen přenosové médium jako např. optické vlákno, koaxiální kabel apod., ale i topologii sítě, kanálové kódování, připojovací konektory či modulaci [15; 16].

Spojová vrstva (Link layer)

Spojová vrstva, též někdy nazývána linková vrstva definuje způsob předání zprávy v síti. Funkce protokolu se nejčastěji dělí do dvou podvrstev, a to MAC (Medium Access Control)a LLC(Logical link Control). MAC vrstva se využívá k řízení přístupu uzlů k médiu, její úhlavní role spočívá v systémech s časovým sdílením komunikačního kanálu. LLC podvrstva přenáší zprávy dat v rámcích, které jsou touto vrstvou definovány [15; 16].

Síťová vrstva ( Network layer)

Síťová vrstva definuje směrování, kterými se pohybují pakety v síti a mezi sítěmi. U mnohabodových decentralizovaných sítí musíme určit způsob předání paketu řetězem stanic umístěných mezi koncovými uživateli. Stanicím propojující různé linky říkáme spojovací uzly [15; 16].

Transportní vrstva ( Transport layer)

Transportní vrstva nám zajišťuje vytvoření dočasných spojení mezi uzly, a to nám dává možnost přizpůsobení nižších vrstev požadavkům vyšších vrstev. Transportní vrstva mimo jiné rozkládá zprávy do paketů a na přijímací straně zajišťuje složení zprávy ve správném pořadí [15; 16].

Relační vrstva ( Session layer)

Zahrnuje, jak autorizaci uživatele ale také i řízení a rušení relací mezi komunikujícími uzly [15; 16].

Prezentační vrstva (Presentation layer)

Prezentační vrstva specifikuje transformaci dat do forem vhodných pro přenos, pro nekompatibilní uzly provádí převody kódů a formátů [15; 16].

Aplikační vrstva (Application layer)

Aplikační vrstva je nejvyšší vrstva modelu, kterou využívají programy v síti. Vzhledem k velkému množství a různorodosti aplikací, zachovává se z aplikačních protokolů pouze „společné" jádro [15; 16].

PDU

PDU v překladu protokolová datová jednotka umožňuje přenášet informace jako jeden celek mezi entitami stejné úrovně v počítačové síti. Obvykle obsahuje řídící informace (např. adresa) a uživatelská data. Ve vícevrstvých systémech PDU se skládá z řídící informace a případných uživatelských dat dané vrstvy [18].

ADU

Z andlického application data unit do češtiny přeloženo aplikační datová jednotka. Jedná se tedy o rozšířenou protokolovou datovou jednotku o další informace, jako jsou například adresa nebo kontrolní součet.

LAN

Lan popřípadě lokální či místní síť označuje síť, která pokrývá malé území (domácnost, firmy). Vyznačuje se systematickou topologií sítě s vyššími přenosovými rychlostmi [17].

### WAN

WAN přeloženo z anglického spojení wide area, znamená v překladu rozsáhlá síť, která pracuje s nižšími přenosovými rychlostmi (cca 64 kbps) na rozdíl od sítě LAN. Topologie sítě WAN je nepravidelná s nižší spolehlivostí přenosových cest a velkým přenosovým zpožděním [17].

### HTTP

Http je jeden z nejdůležitějších protokolů pro přenos dat v síti internet. Tento protokol má na starosti přenos dat mezi serverem a klientem.

### FTP

FTP je protokol určený pro přenos dat v sítích TCP/IP. Předchůdce protokolu http [10].

### DNS

DNS je název pro protokol s hierarchickým systémem doménových jmen. Součástí tohoto systému jsou tzv. DNS servery, které schraňují informace o spravovaných doménách. Protokol DNS pracuje na principu rozsáhlého stromu, v jehož nejvyšších strukturách se o každou větev stará jeden DNS server a v nižších strukturách obsluhuje DNS server více struktur najednou [13; 14].

### DHCP

Jedná se o protokol, jehož úkolem je automatické přidělení IP adresy počítači připojenému do sítě. DHCP využívá architektury typu klient/server, což ve výsledku znamená, že žádost o konfigurační informace přijme DHCP server a posléze odešle konfigurační informace zpět [19; 20].

UDP

User data protokol je jedním ze dvou protokolů na transportní vrstvě. Tento protokol si na úrovni síťové vrstvy ponechává sporný a nespojovaný způsob fungování protokolu IP [21].

HMI

Human Maschine Interface do češtiny přeloženo rozhraní člověk/stroj, reprezentuje rozhraní mezi přístrojem a jeho operátorem, tedy člověkem [22].

RTU

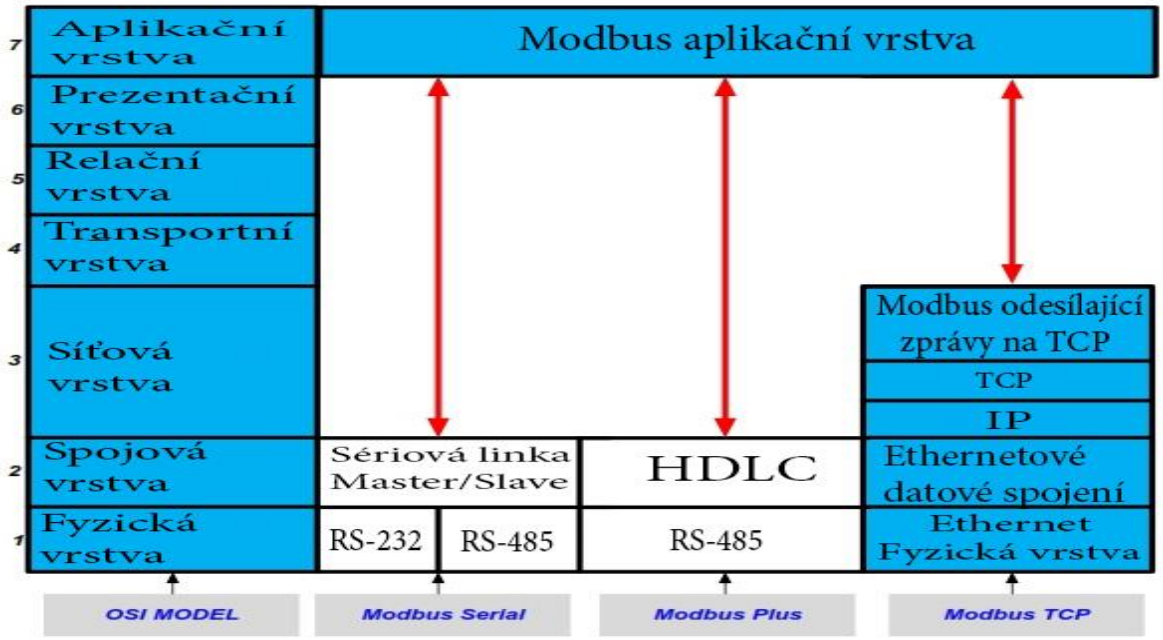
Mikroprocesorem ovládané elektronické zařízení, které propojuje senzoriku se SCADA systémy [23].

PLC

Programovatelný logický automat je relativně malý průmyslový počítač řízený mikroprocesorem s vlastním operačním systémem.

# 2 Protokol MODBUS

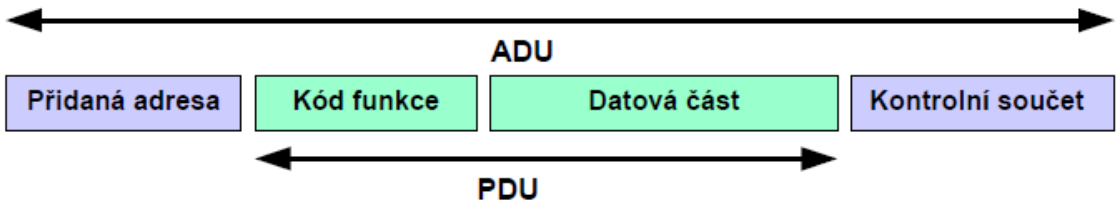
MODBUS je otevřený komunikační protokol vyvinutý firmou Modicon, která jej vytvořila v roce 1979. Jedná se o metodu, používající pro přenos informací sériové linky mezi elektronickými zařízeními např. RS-232, RS-422, RS-485 či síť Ethernet s využitím protokolu TCP/IP. Komunikace probíhá metodou request/responce (žádost/odpověď). Ve standardní síti MODBUS je jeden nadřízený prvek a až 247 podřízených prvků [5; 4; 24; 34].




Obrázek č. 2 – Model OSI/ISO pro protokol MODBUS; [6]

## 2.1 Použití protokolu MODBUS

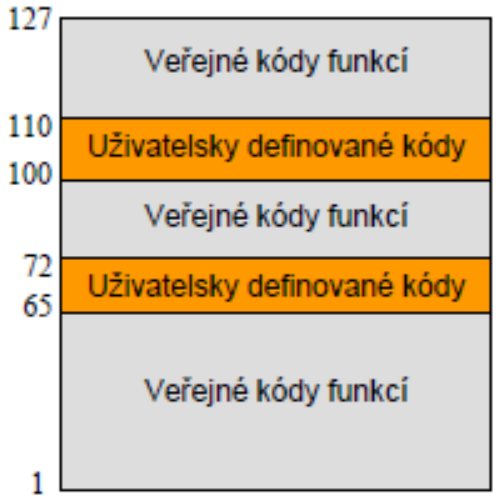
MODBUS je volně použitelný protokol, což znamená, že výrobci ho mohou bezplatně použít ve všech svých zařízeních. MODBUS se stal standardním komunikačním protokolem v průmyslu a je hojně používán pro přenos signálů z přístrojové a řídící jednotky zpět do hlavního řídicího systému [4; 24; 34].




Obrázek č. 3 – Základní tvar MODBUS zprávy; [24]

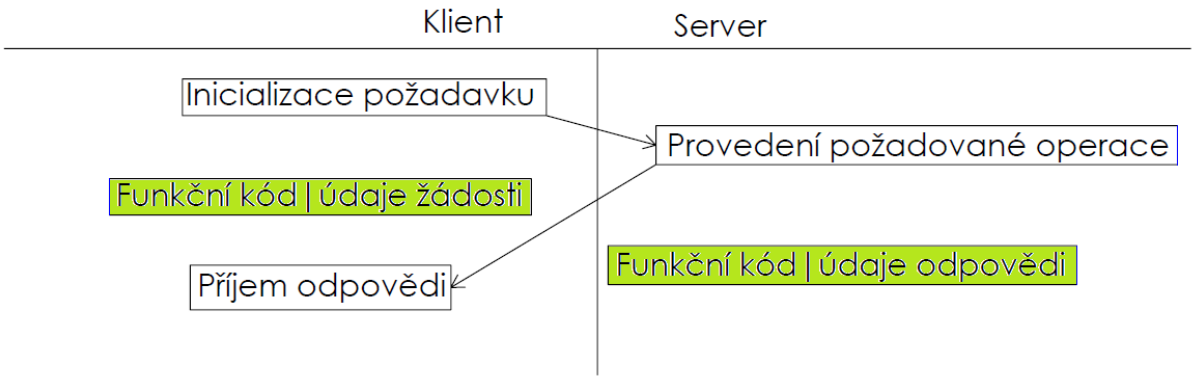
## 2.2 Popis protokolu

Protokol MODBUS je definován strukturou zprávy na úrovni PDU (Protocol Data Unit) nezávisle na typu komunikační vrstvy. V závislosti na typu sítě je PDU rozšířena o další části tvořící tak zprávu na ADU (Application Data Unit). Kód funkce datové jednotky udává jaký druh operace se má provést. Rozsah kódů je 1 až 255 (rozsah 1-65 a 111-127 je určen pro veřejné kódy funkcí a rozsah 128-255 je vyhrazen pro oznámení záporné odpovědi (chyby)). Některé kódy funkcí obsahují i kód dílčí funkce k upřesnění požadované operace. Datové pole zpráv odesílaných ze zařízení klienta na server obsahuje další informace jako například: adresa, počet vstupů nebo hodnota registrů, na které má server zapsat. U některých operací může tato datová zpráva chybět, jelikož pro danou funkci nejsou zapotřebí další data [4; 24; 34].



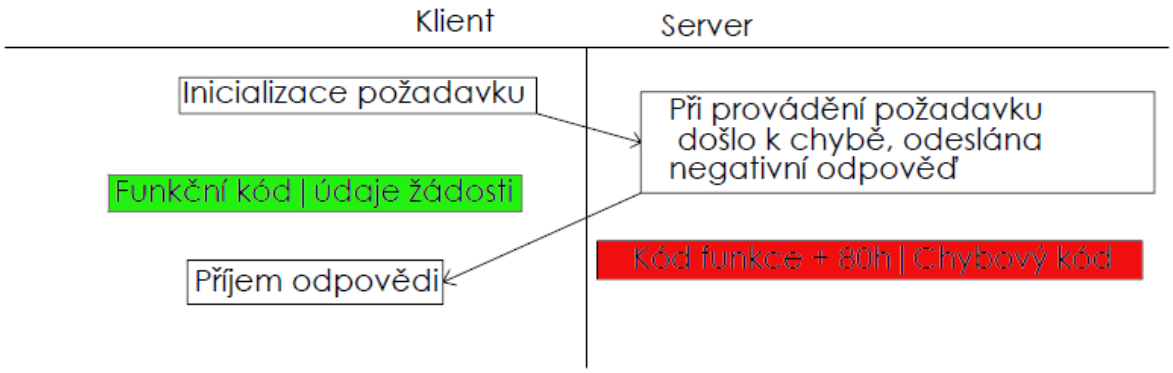

Obrázek č. 4 - Kategorie funkčních kódů; [24]

Pokud nedojde při provádění dané operace k chybě, odpovídá server zprávou, která obsahuje kód v poli kód funkce požadované funkce jako indikaci kladně vykonaného požadavku. V datové části předává server data klientovi, pokud jsou nějaká [4; 24; 34] .




Obrázek č. 5 – MODBUS transakce; [24]

Pokud dojde během požadavku operace k chybě, je v poli kód funkce vrácen kód žádané funkce s nastaveným nejvyšším bitem indikujícím neúspěch. Datová část vrací chybový kód (exception code), upřesňující důvod selhání požadavku [4; 24; 34].




Obrázek č. 6 – MODBUS transakce s chybou při provádění požadavku; [24]

Velikost MODBUS PDU je limitována velikostí zděděnou z první implementace MODBUSu na sériovou linku RS485 ADU = 256 bytů, tomu odpovídá maximální velikost PDU =253 bytů.

Proto:

Maximální velikost PDU na sériové lince = 256 - adresa serveru (1 byte)- kontrolní součet CRC (2 byty)=253 bytů.

Odtud:

Velikost ADU na TCP/IP=253 bytů PDU+MBAP=260 bytů.

Protokol MODBUS má 3 základní typy zpráv (PDU):

1. Request PDU, mb_req_pdu

Request PDU je tvořeno kódem funkce, která má velikost 1 byte a Datová část (adresa, proměnné, počet proměnných atd.) má velikost n bytů.

2. Responce PDU, mb_rsp_pdu

Responce PDU se skládá z funkčního kódu o velikosti 1 byte a datové odpovědi s velikostí m bytů (korekce dat, sub funkční kódy atd.).

3. Exception response PDU, mb_excep_rsp_pdu

Exception response PDU je tvořeno kódem funkce (1 byte)+80h (indikace selhání požadavku) a chybovým kódem (1byte) [4; 24; 34].

### 2.2.3 Kódování dat

Protokol MODBUS využívá tzv. „Big-endian" reprezentaci dat. To znamená, že při odesílání dat větších než 1 byte odesílá první nejvyšší byte a jako poslední posílá poslední nejnižší byte [4; 24; 34].

| Velikost položky | hodnota | |
|---|---|---|
| 16 – bits | 0x1234 | prvně odesílá 0x12 a posléze 0x34 |

| Tabulka | Typ položky | Přístup | Popis | Adresa |
|---|---|---|---|---|
| Diskrétní vstupy (Discrete Inputs) | 1-bit | Pouze čtení | Data poskytovaná I/O systémem | 10000÷19999 |
| Cívky (Coils) | 1-bit | Čtení/zápis | Data modifikovatelná aplikačním programem | 0÷9999 |
| Vstupní registry (Input Registers) | 16-bitové slovo | Pouze čtení | Data poskytovaná I/O systémem | 30000÷39999 |
| Uchovávající registry (Holding Registers) | 16-bitové slovo | Čtení/zápis | Data modifikovatelná aplikačním programem | 40000÷49999 |

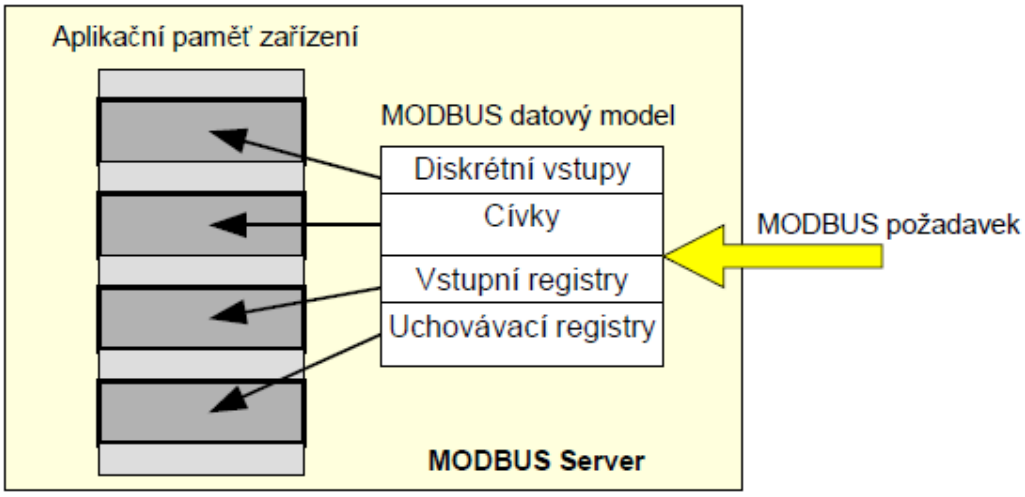
Tabulka č. 1 – Datový model MODBUS; [24]

## 2.3 MODBUS datový model

MODBUS zakládá svůj datový model na řadě tabulek, které mají charakteristické vlastnosti. Definovány jsou 4 základní tabulky [4; 24; 34].

Mapování tabulek do adresního prostoru je závislé na konkrétním zařízení. Každá z adresních tabulek může, mít svůj vlastní prostor popřípadě se mohou částečně či úplně překrývat z důvodu zpětné kompatibility zbývá adresní prostor rozdělen bloky o velikosti cirka 10000 položek, tak jak je uvedeno ve sloupci adresa přístupná je každá položka samostatně nebo je možné přistupovat ke skupině položek najednou velikost skupiny položek, je limitována maximální velikost datové části

zprávy, na obrázcích níže jsou znázorněny dva možné způsoby organizace dát v zařízení. Do jednotlivých tabulek je možné vstupovat prostřednictvím daných funkcí MODBUSU [4; 24; 34].




Obrázek č. 7 – Datový model MODBUS se 4 oddělenými cívkami; [24]

# 3 Adresovací model

Protokol MODBUS exaktně definuje adresovací pravidla ve zprávách (PDU) :

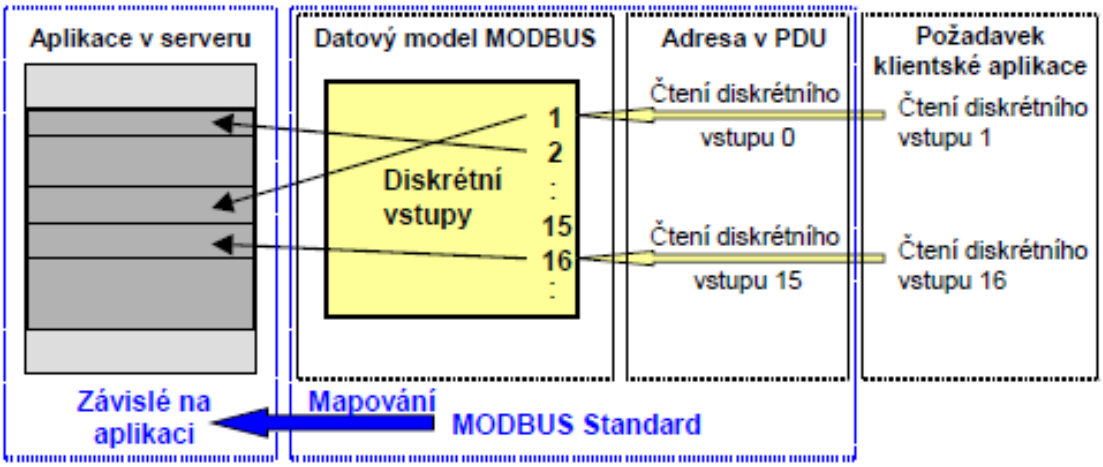
*„V MODBUS zprávách jsou datové položky adresovány od nuly do 65535“* [24]

Také je definováno adresování v rámci datového modelů sestávajícího ze čtyř datových tabulek [34]

*„V MODBUS datovém modelu jsou položky v datových blocích číslovány od jedné do n.“* [24]

Mapování položek MODBUS datového modelů do aplikace serveru je zcela v režii výrobce [34]

*„Na obrázku č. 8 je zobrazen příklad adresování od požadavku klienta až po aplikace v serveru. Z obrázku je patrné, že datová položka Y v datovém modelu je v PDU adresována jako položka Y“ [24]*




Obrázek č. 8 – Příklad adresování dle protokolu MODBUS; [24]

# 4 Kategorie kódů funkcí

Protokol MODBUS formuluje tři základní skupiny kódů funkcí:

Veřejné kódy funkcí [4; 24; 34]

- Jasně definované
- S garantovanou unikátností
- Schvalovány společností MODBUS python.org
- Veřejně zdokumentované
- Je k nim veřejně dostupný test shody
- Zahrnují veřejné přiřazené kódy funkcí i nepřiřazené kódy rezervované pro budoucí použití

Uživatelsky definované kódy funkcí

- Dva rozsahy uživatelsky definovaných kódu funkci: V 65 ÷ 72 a 100 ÷ 110
- Umožňují uživateli implementovat funkci, která není definována touto specifikací
- Není garantována unikátnost kódu
- Lze je po projednání přesunout do veřejných kódů

Rezervované kódy funkcí

- Kódy funkci, které jsou v současnosti používány některými firmami a které nejsou dostupné pro veřejnou funkci.

## 4.1 Definice funkčních kódů

| | | | | Kódy funkcí | | |
|---|---|---|---|---|---|---|
| | | | | Kód | Podfunkce | Hex |
| Přístup k datům | Bitový přístup | Fyzické diskrétní | Čti diskrétní vstupy | 2 | | 2 |
| | | Interní bity nebo fyzické cívky | Čti cívky | 1 | | 1 |
| | | | Zapiš jednu cívku | 5 | | 5 |
| | | | Zapiš více cívek | 15 | | 0F |
| | 16-bitový přístup | Fyzické vstupní | Čti vstupní registr | 4 | | 4 |
| | | Interní nebo fyzický výstupní registry | Čti uchovávací registry | 3 | | 3 |
| | | | Zapiš jeden registr | 6 | | 6 |
| | | | Zapiš více registr | 16 | | 10 |
| | | | Čti/zápiš více registrů | 23 | | 17 |
| | | | Zapiš registr s maskováním | 22 | | 16 |
| | | | Čti FIFO frontu | 24 | | 18 |
| | Přístup k záznamům v souborech | | Čti záznam ze souboru | 20 | 6 | 14 |
| | | | Zapiš záznam do souboru | 21 | 6 | 15 |
| Diagnostika | | | Čti stav | 7 | | 7 |
| | | | Diagnostika | 8 | 00-18,20 | 8 |
| | | | Čti čítač komunikačních událostí | 11 | | 0B |
| | | | Čti záznam komunikačních události | 12 | | 0C |
| | | | Sděl identifikaci | 17 | | 11 |
| | | | Čti identifikaci zařízení | 43 | 14 | 2B |
| Ostatní | | | Zapouzdřený přenos | 43 | 13,14 | 2B |
| | | | CANOpen základní | 43 | 13 | 2B |

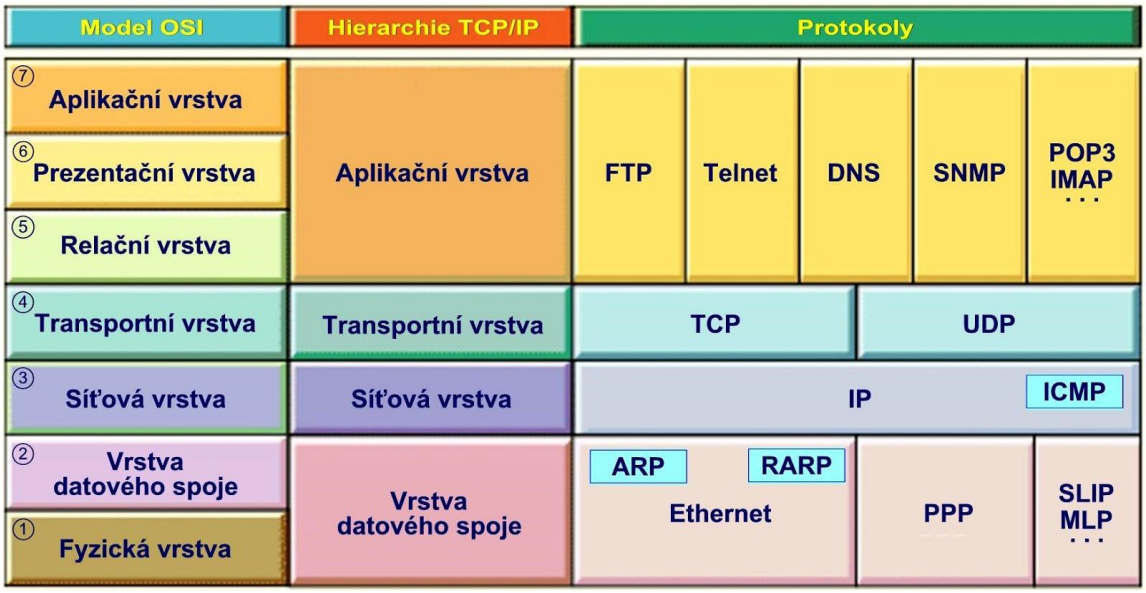
Tabulka č. 2 – Definice funkčních kódů; [4; 24]

# 5 Protokol TCP/IP

Skupina protokolů TCP/IP obsahuje sadu protokolů pro komunikaci v počítačové síti a v celosvětové síti Internet. Komunikační protokol je soubor pravidel, která určují skladbu a význam daných zpráv při komunikaci. Počátky internetového protokolu lze najít ve výzkumu americké agentury DARPA, který probíhal na konci 60. let. Původní účel protokolu měl být sjednotit počítačové komunikace v rámci ARPANET. Dnes je tento protokol součástí prakticky všech operačních systémů a je využíván ke komunikaci i v síti Internet. Model TCP/IP je nezávislý na přenosovém médiu a je určen jak pro LAN, WAN tak i pro sériové linky [11; 30; 34].

## 6.1 Architektura TCP/IP

Zatímco referenční model ISO/OSI počítá se sedmi vrstvami, protokol TCP/IP počítá pouze se čtyřmi vrstvami. Pro zajištění spolehlivosti na vyšších vrstvách jsou spolehlivé přenosové služby, na nižších vrstvách jsou umístěny přenosové služby s nespolehlivou přenosovou službou. Každá vrstva pro komunikaci využívá služeb vrstvy nižší a poskytuje své služby vrstvě vyšší [31; 34].




Obrázek č. 9 – Porovnání architektury OSI a TCP/IP; [31]

Aplikační vrstva

Aplikační vrstva dle modelu ISO/OSI sdružuje vrstvy Aplikační, Prezentační a Relační. V této vrstvě jsou provozovány základní aplikace v rámci protokolu. Aplikační vrstva zaštiťuje přenos a srozumitelnost zpráv. Jsou to protokoly, které slouží k přenosu konkrétních dat např. HTTP, FTP, DNS či DHCP. Aplikační protokoly vždy využívají jednu ze dvou základních služeb transportní vrstvy: TCP popřípadě UDP nebo obě dvě. Pro rozlišení aplikační vrstvy se využívají tzv. porty, což jsou smluvená číselná označení aplikací [34]. Každé síťové spojení je jednoznačně určeno číslem portu, transportním protokolem a adresou počítače [11].

Transportní Vrstva

Poskytuje transportní služby pro kontrolu celistvosti dat, a to při kontrolovaném a nekontrolovaném spojení. Kontrolované spojení je kontrolováno protokolem TCP a nekontrolovaná spojení jsou kontrolována méně spolehlivým protokolem UDP (user data protocol). Transportní vrstva je aplikována až v koncových zařízeních, zejména jsou to počítače.

Transportní vrstva dále umožňuje přizpůsobit chování sítě potřebám dané aplikace [34].

Síťová vrstva

Síťová vrstva představuje především síťovou adresaci, směrování a předávání datagramů (základní jednotka, která je přepravována v počítačové síti). Síťová vrstva využívá služeb protokolů, jako jsou např.: ARP, ICMP nebo IP. IP protokol se snaží doručovat data co nejrychleji, přes uzly až ke koncovému příjemci. To znamená, že nabízí nespolehlivou přenosovou službu nespojovaného charakteru. Z tohoto vyplívá, že se jedná nespolehlivý protokol s částečnou detekcí chyb. Protokol ICMP přenášející zprávy o chybách, který je nedílnou součásti protokolu IP, má bohužel stejnou spolehlivost. Nicméně pomocí protokolu ICMP lze zjistit propustnost a to pomocí utility PING (Packet Internet Grouper), popřípadě topologii sítě a to pomocí utility Traceroute. Utilita PING vysílá pomocí servisního protokolu (ICMP) sérii testovacích paketů, které příjemce ihned vrací zpět. Změřením doby odezvy jsme schopni zjistit dostupnost konkrétního uzlu. Hodnota propustnosti je udávána v ms a její jednotky se pohybují od jednotek až po tisíce. Utilita Traceroute pro operační systémy DOS a Windows je schopna najít požadovaný uzel i přes 20 mezi uzlů. Program provede standardně 3 pokusy o ping na jednotlivé uzly. Podle jmen jednotlivých uzlů a z odezvy je schopen rekonstruovat celou cestu [34].

Funkce protokolu IP [30; 34]:

- Adresování v síťové vrstvě
- Směrování datagramů
- Propojení síťové a transportní vrstvy. Protokolům jsou přiřazena pevná čísla např.: TCP - 6 a UDP – 17
- Fragmentace a sestavení datagramů podle MTU (maximumTransfer Unit).

Nejdůležitější zprávy protokolu ICMP:

- Echo request a echo response - žádost a odpověď na ping
- Parametr problém – chyba v hlavičce IP paketu
- Redirect – upozornění pro koncový uzel
- Destination unreacheble

Vrstva síťového rozhraní

Vrstva síťového rozhraní je nejnižší vrstva, která umožňuje přístup k přenosovému médiu. Je jiná pro každou síť v závislosti na její aplikaci. Jsou to sítě jako Ethernet, Token ring či SMDS [34].

# 7 Controller Area Network

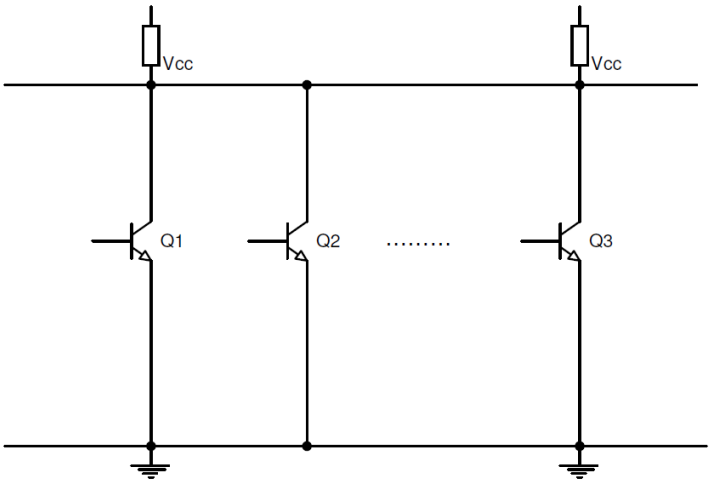
*„Controller Area Network (CAN) je sériový komunikační protokol vyvinutý firmou Bosch pro nasazení v automobilech. Vzhledem k faktu, že výrobci mikroprocesorů implementovali podporu protokolu CAN do svých zařízení, je protokol široce využíván také v průmyslových aplikacích. Samotná specifikace CANu definuje pouze protokol linkové vrstvy. Existuje několik různých variant protokolu fyzické vrstvy ( ISO 11898, ISO 11519, J1939 …) i různé protokoly aplikační vrstvy jako jsou např. : CANopen, DeviceNet atd.*

*CAN je navržen, tak aby umožnil distribuované řízení systémů v reálném čase s přenosovou rychlostí do 1Mbit/s a vysokým stupněm zabezpečení přenosu proti chybám. Jedná se o protokol typu multi-master, pro řízení přístupu k médiu je použita sběrnice s náhodným přístupem, která řeší kolize na základě prioritního rozhodování. Jednotlivé zprávy neobsahují žádnou informaci o cílovém uzlu, kterému jsou určeny, a jsou přijímaný všemi uzly připojenými ke sběrnici. Každá zpráva je uvozena identifikátorem, který definuje obsah přenášené zprávy a její prioritu.*

*Protokol CAN zajišťuje, aby zpráva s vyšší prioritou byla v případě kolize více zpráv doručena přednostně." [2].*

## 7.1 Fyzická vrstva protokolu

Elementárním požadavkem na přenosové fyzické médium protokolu CAN je realizace logického součinu. Standard definuje dvě vzájemně doplňující se hodnoty bitů na sběrnici, a to dominantní a recesivní. V podstatě se jedná o univerzální ekvivalent logických úrovní, jejichž hodnoty nejsou definovány a skutečná prezentace záleží na konkrétní realizaci fyzické vrstvy. Pravidla pro stav na sběrnici jsou triviální a jasně určená. V případě, že všechny uzly sběrnice vysílají recesivní hodnotu bitu, pak na sběrnici je recesivní úroveň. Pokud vysílá alespoň jeden uzel dominantní hodnotu, v takovém případe je na sběrnici dominantní úroveň. Příkladem takového chování může být například sběrnice buzená hradly s otevřeným kolektorem [2; 34].

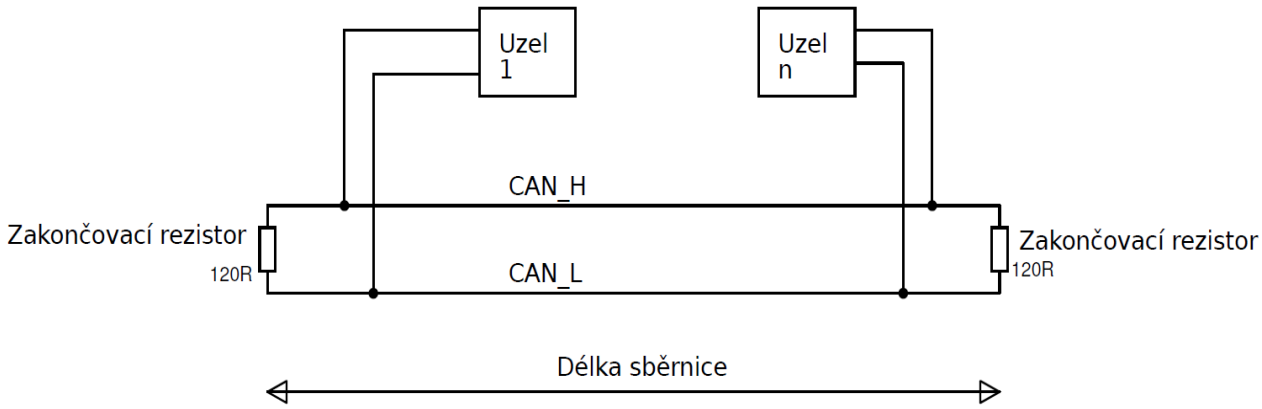



Obrázek č. 10 – Příklad realizace fyzické vrsty; [2]

V takovém to případě dochází k dominantnímu stavu logická nula a recesivnímu stavu logická jednička, popřípadě je-li jeden tranzistor sepnut, bude na sběrnici úroveň logické nuly a dále nezáleží, již zda jsou ostatní

tranzistory sepnuty či nejsou. Pokud jsou všechny tranzistory rozepnuty, je na sběrnici úroveň logické jedničky [2; 34].

Při návrhu přenosového fyzického média se hojně využívá rozdílová sběrnice definovaná dle normy ISO 11898. Sběrnice je tvořena dvěma vodiči, a to CAN_H a CAN_L. Úroveň sběrnice je definována pomocí rozdílového napětí na těchto dvou vodičích. Recesivní úroveň na sběrnici je definována normou na nominální úrovni rozdílového napětí $V_{diff}$ = 0 V, dominantní úrovni přísluší nominální rozdílové napětí $V_{diff}$ = 2 V. Oba konce vedení jsou vybaveny rezistory o velikosti 120 Ω, a to kvůli eliminaci rušivých odrazů na vedení. Na sběrnici můžeme teoreticky připojit nekonečný počet uzlů, nicméně s ohledem na možné zatížení sběrnice a zajištění korektních statických a dynamických parametrů sběrnice, norma udává, že maximální počet uzlu, které lze připojit je 30. Dále norma uvádí maximální délku 40 metrů pro přenosovou rychlost 1Mbit/s, pro jiné přenosové rychlosti norma neuvádí [2; 34].




Obrázek č. 11 – Fyzické uspořádání sítě CAN dle ISO 11898; [2]

## 7.2 Linková vrstva protokolu

Linkovou vrstvu protokolu tvoří dvě podvrstvy a to MAC a LLC. Podvrstva MAC (Medium Acces Control) má na starosti řízení přístupu k médiu, kódování dat a vkládání doplňkových bitů do komunikace (Stuffing/Destuffing), řízení přístupu všech uzlů k médiu s rozlišením priority

zpráv, detekce chyb a hlášení o chybách a potvrzení správně přijatých zpráv. Druhá podvrstva LLC ( Logical Link Control) zajišťuje filtrování přijatých zpráv (Acceptance Filtering) a hlášení o přetížení (Overload Notification) [2; 34].

Nachází-li se sběrnice ve stavu Bus free (sběrnice je volná), může libovolný uzel začít vysílat. Začne-li jeden z uzlů vysílat dříve než ostatní, zabírá si tímto celou sběrnici pro sebe a ostatní uzly mohou začít vysílat až po skončení uzlu, který na sběrnici zahájil vysílání jako první. Jedinou výjimkou v tomto jsou chybové hlášky, které může vysílat kterýkoliv uzel a kdykoliv [2; 34].

V případě, že několik uzlů zahájí vysílání zprávy současně, získá přístup na sběrnici ten uzel, který přenáší zprávu s vyšší prioritou (nejnižší hodnota identifikátoru). Identifikátor je zasílán na začátku zprávy. Každý vysílač srovnává hodnotu právě zasílaného bitu s hodnotou na sběrnici a zjistí-li, že na sběrnici se nachází jiná hodnota, než vysílá (tento jev nastává pouze pokud, vysílač vysílá recesivní bit a na sběrnici je dominantní úroveň), neprodleně ukončí další vysílání. Tímto mechanismem je zaručeno, že zpráva s vyšší prioritou bude zaslána přednostně a že nedojde k jejímu poškození. Uzel, který při střetu nedostal přístup na sběrnici, musí počkat do doby, než bude sběrnice opět volná (Bus free) a posléze se pokusí zprávu vyslat znovu [2; 34].

Zprávy zasílané za pomoci protokolu CAN jsou chráněny hned několika mechanismy, které jsou v činnosti současně.

- Monitoring – je již zmiňovaná metoda, kdy vysílač porovnává hodnotu právě vysílaného bitu s úrovní zjištěnou na sběrnici. V případě, že jsou obě hodnoty shodné, vysílač pokračuje ve vysílání. V opačném případě je-li detekování jiná úroveň, než odpovídá vysílanému bitu v momentě, kdy se právě odehrává řízení přístupu na sběrnici, je okamžitě přerušeno vysílání a přístup na sběrnici získá uzel zasílající zprávu s vyšší prioritou.

V případě, je-li rozdílnost vysílané a zjištěné úrovně v jiné části rámce, je okamžitě indikována chyba bitu.

- CRC kód (Cyclic Redundancy Check) – na konci každé zprávy je předložen 15 bitový CRC kód, který se skládá ze všech předcházejících bitů příslušné zprávy generujícím polynomem:

$$G(x) = X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1 \qquad (2.1)$$

Vkládání bitu (bit stuffing) – V případě, je-li na sběrnici vysíláno pět po sobě jdoucích bitů je stejné úrovně, je do zprávy vložen bit opačné úrovně. Toto opatření jednak vede k detekci chyb a také k správnému časové synchronizaci přijímačů v jednotlivých uzlech.

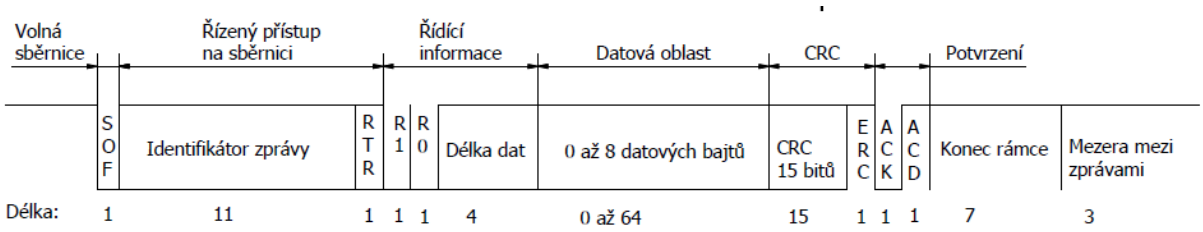
- Kontrola formátu zprávy (Message Frame Check) – zpráva je kontrolována dle formátu daným specifikací a v případě, že je zakázaná hodnota na nějaké pozici bitu, je detekována chyba.
- Potvrzení přijetí zprávy (Acknowledge) – Pakliže je zpráva v pořádku přijata libovolným uzlem, je tato skutečnost potvrzena změnou hodnoty jednoho bitu zprávy (ACK). Vysílač vždy na tomto bytu vysílá recesivní úroveň. Zachytí-li dominantní úroveň generovanou ostatními uzly, pak je vše v nejlepším pořádku. Potvrzování přijetí zprávy provádí všechny uzly připojené k sběrnici bez ohledu na to, zda mají filtrování zpráv zapnuté či vypnuté. Pokud jeden z uzlů detekuje chybu, vyšle na sběrnici tzv. chybový rámec, který se skládá z šesti dominantních bitů. Chyba je detekována dalšími uzly a to na základě porušení pravidla o, vkládaní bitů. Dodatečné ochranný mechanismus zajistí, aby poškozený uzel neblokoval celou sběrnici chybovými rámci. Každý uzel má zabudovaná dva interní čítače chyb, které počítají chyby, jak při příjmu zprávy, tak i při vysílání. Podle obsahu čítače může uzel přecházet mezi třemi stavy: aktivní – uzel je schopný aktivně se podílet na komunikaci po sběrnici. Pokud dojde k detekci libovolné chyby v právě přenášené zprávě, vysílá na sběrnici chybový rámec, pasivní – uzel v tomto stavu se také podílí na komunikaci, nicméně v případě hlášení chyby vysílá pouze recesivní chybový rámec, čímž dojde ke zničení právě vyslané zprávy. Odpojené uzly v tomto stavu

nemají žádný vliv na sběrnici, jejich výstupní budiče jsou vypnuty. V případě, že uzel generuje příliš mnoho chyb, je uzel automaticky odpojen (Bus off). [2; 7; 34]

Protokol CAN definuje čtyři typy zpráv:

- Datová zpráva (data frame)
- Žádost o data (remote request frame) – touto zprávou žádá uzel ostatní uzly na sběrnici o zaslání žádaných dat.
- Zpráva o chybě
- Zpráva o přetížení

První dva typy se týkají komunikace po sběrnici. Datová zpráva je elementární prvek komunikace po sběrnici a umožňuje vyslat zprávu dlouhou až 8 datový bajtů. Pro jednoduché povely jako: zapni/ vypni, není třeba vysílat žádná data, tato data mohou být obsažena v identifikátoru zpráv, což zrychluje přenos v protokolu CAN a zvyšuje propustnost sběrnice. Odpovědí na tento požadavek jsou data od uzlu, který má požadovaná data k dispozici. Zbylé dva typy slouží k řízení sběrnice, primárně k signalizaci detekovaných chyb a eliminaci chybných zpráv. [2; 34]




Obrázek č. 12 – Datová zpráva dle specifikace CAN 2.0A; [2]

*„Význam jednotlivých částí datové zprávy je následující:*

- *Začátek zprávy SOF(Start of Frame) – 1 dominantní bit*
- *Řízení přístupu ke sběrnici a identifikátor zprávy (Arbitration Field),*
  - *Identifikátor zprávy – 11 bitů, udává význam přenášené zprávy*

- *o RTR (Remote Request) – 1 bit, příznak udává, zda se jedná o datovou zprávu, nebo o žádost o vyslání dat. V datové zprávě je tento bit dominantní, žádosti o data recesivní.*

- *Řídicí informace (Control Field)*
  - *o R0 a R1 rezervované bity.*
  - *o Délka dat – 4 bity, počet datových bajtů ve zprávě. Povolené hodnoty 0 až 8.*
- *Datová oblast (Data Field) – datové bajty zprávy. Maximálně 8 bajtů.*
- *CRC (CRC Field) – 2 bity*
- *Potvrzení (ACK Field) - 2bity*
  - *o ACK (acknowledge bit) – 1 dominantní bit*
  - *o ACD(Acknowledge delimeter) – 1 recesivní bit*
- *Konec zprávy (End of Frame) – 7 recesivních bitů*
- *Mezera mezi zprávami (Interfame Space) – 3 recesivní bity, odděluje dvě zprávy".[2]*

## 7.3 Aplikační vrstva protokolu

Aplikační vrstva protokolu je definována několika desítkami navzájem nekompatibilních protokolů jako jsou například CANopen, CAL, CAN Kingdom a DeviceNet. Nejrozšířenějším protokolem v průmyslové automatizaci ze zde zmíněných protokolů je protokol CANopen. Struktura aplikačního protokolu CANopen je poměrně složitá. Ve zjednodušené podobě ji znázorňuje obr. č. 13 [2; 34]. „*Protokol obsahuje dvě skupiny protokolů. První z nich tvoří protokol CMS (CAN Message specification), který definuje způsoby přenosu procesních dat různých typů a jejich kódování do linkových rámců, druhá skupina protokolů slouží pro řízení a správu sítě a patří do ní protkoly NMT ( Network Management), DBT (Distributor) a LMT (Layer Management)". [2]*

| Profily zařízení | Profily rozhraní | Aplikační profily | CANopen manažer | Specifické profily výrobce |
|---|---|---|---|---|
| Rámce<br>Komunikační profil<br>Aplikační vrstva CAL | | | | |
| Linková vrstva dle ISO 11898 | | | | |
| Fyzická vrstva dle ISO 11898 | | | | |

Obrázek č. 12 – Struktura protokolu CANopen; [2]

# 8 Produkty firmy Teco a.s.

## 8.1 PLC Tecomat TC700

Obrázek č. 13 – Produkty firmy Teco a.s.; [28]

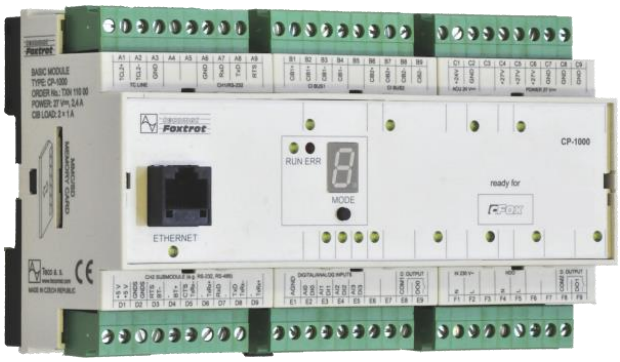
Tecomat TC700 je řídicí systém na modulární bázi se zvýšenou provozní spolehlivostí, vhodný pro širokou škálu středních a velkých aplikací v nejrůznějších odvětvích (strojírenství, doprava, řízení průmyslových

procesů nebo technické zařízení budov). Všech osvědčených vlastností dosahuje díky důsledné implementaci mezinárodní normy IEC 61131. Mezi hlavní přednosti tohoto systému patří vyšší výkon, za který vděčí procesorové jednotce s 32 bitovým RISC procesorem, větší paměť a propracované modularitě systému, která umožňuje decentralizovat periferie, které mohou pracovat až na vzdálenost 1,700 m.

## 8.2 Tecomat Foxtrot

Tecomat typu Foxtrot je osvědčený modulární řídící a regulační systém, který se může pyšnit značnou kompaktností. Díky možnosti montáže na DIN lištu, je vhodným adeptem pro malé a střední aplikace s možností montáže přímo do rozvaděčů. Tyto systémy můžeme nazvat řídicími systémy nové generace a to zejména díky výkonné procesorové jednotce a bohaté komunikační schopnosti.

CP-1003




Obrázek č. 14– PLC TECOMAT CP-1003; [25]

Technické specifikace

| | |
|---|---|
| Digitální vstupy | 8 |
| Reléové výstupy | 4 |
| Analogové vstupy | 4 |
| Analogové výstupy | 4 |
| CPU | 32 bit RISC procesor |
| Doba sepnutí kontaktu | 10 ms/ 4 ms |
| Doba cyklu PLC | 0,2 ms / 1 k instrukcí |
| Hodiny reálného času | Ano |
| Komunikace | Ethernet 100/10, CIB, 1× RS-232, TCL2(345 bit/s) |
| Paměť | Paměťový slot pro karty SD/SDHC/ MMC |
| Napájení | + 24 V |
| Maximální příkon | 10 W |
| Stupeň krytí IP IEC 529 | IP 20 |

Tabulka č. 3 - Technické specifikace CP-1003; [25]

CP-1015


Obrázek č. 15– PLC TECOMAT CP-1015; [26]

## Technické specifikace

| | |
|---|---|
| Digitální vstupy | 6 |
| Reléové výstupy | 2 |
| Analogové vstupy | 6 |
| Analogové výstupy | 2 |
| CPU | 32 bit RISC procesor |
| Doba sepnutí kontaktu | 10 ms/ 4 ms |
| Doba cyklu PLC | 0,2 ms / 1 k instrukcí |
| Hodiny reálného času | Ano |
| Komunikace | Ethernet 100/10, 2× CIB, 1× RS-232, 1× volitelné, TCL2(345 bit/s) |
| Paměť | Paměťový slot pro karty SD/SDHC/ MMC |
| Napájení | + 24 V |
| Maximální příkon | 10 W |
| Stupeň krytí IP IEC 529 | IP 20 |

Tabulka č. 4 - Technické specifikace CP-1015; [26]

CP-1018


Obrázek č. 16 – PLC TECOMAT CP-1018; [27]

## Technické specifikace

| | |
|---|---|
| Digitální vstupy | 10 |
| Reléové výstupy | 4+ 2 Solid State Relay |
| Analogové vstupy | 10 |
| Analogové výstupy | 4 |
| CPU | 32 bit RISC procesor |
| Doba sepnutí kontaktu | 10 ms/ 4 ms |
| Doba cyklu PLC | 0,2 ms / 1 k instrukcí |
| Hodiny reálného času | Ano |
| Komunikace | Ethernet 100/10, 2× CIB, 1× RS-232, 1× volitelné, TCL2(345 bit/s) |
| Paměť | Paměťový slot pro karty SD/SDHC/ MMC |
| Napájení | + 24 V |
| Maximální příkon | 10 W |
| Stupeň krytí IP IEC 529 | IP 20 |

Tabulka č. 5 - Technické specifikace CP-1018; [27]

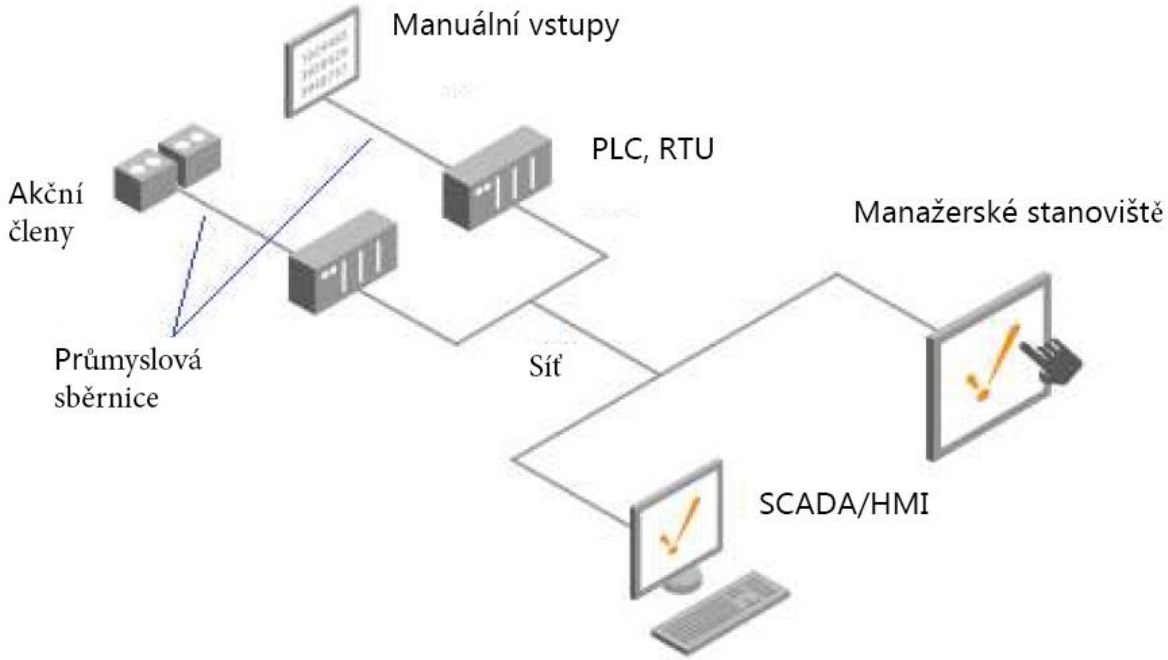
# 9 SCADA systémy

SCADA systémy nahradily dřívější ovládací panely, v porovnání s moderním PC gigantických rozměrů. Oba tyto systémy řízení umožňovaly náhled průmyslových dat v reálném čase. Nespornou výhodou SCADA systému oproti ovládacím panelům je nesmírná variabilita nastavení dle daného problému a uživatele, například zobrazení trendů pro libovolnou proměnnou dle operátorových požadavků, které mu umožní efektivně ale hlavně bezpečně řídit danou úlohu či experiment [9].

SCADA (Supervisory Control and Data Acquisition) bývá do češtiny volně překládáno jako dispečerské řízení a sběr dat. Jedná se o software využívaný v průmyslové automatizaci, který hromadně sbírá a vyhodnocuje data technického charakteru např. údaje o teplotě z termočlánků. Výsledky takovéhoto vyhodnocování jsou posléze zobrazována například pomocí webové aplikace ve webovém prohlížeči, popřípadě v aplikaci nainstalované na daném počítači. Díky tomuto zobrazení může operátor vidět aktuálně nastavené hodnoty popřípadě reálný stav zařízení, který může signalizovat, že systém spěje k poruchovému stavu, což umožňuje operátorovi zavčasu zasáhnout a odvrátit hrozící nebezpečí, které by nemuselo být pouze lokálního charakteru nýbrž globálního. Vzdálené řízení a regulace, které SCADA systémy umožňují, nám dovolují jak nastavovat požadované hodnoty (teploty, talku, atd.) tak i například otáčky motoru nebo rychlost výroby na výrobní lince [9].

Samá podstata fungování SCADA systémů začíná u sběru dat pomocí vstupních senzorů a snímačů, které mohou být jak digitální tak analogové. Tyto senzory a snímače zajišťují pro automatizovaný proces nezbytná data, bez kterých by se žádný SCADA systém neobešel. Tyto vstupní objekty jsou umístěny přímo na zkoumaném objektu a přenášejí data, která sbíráme pomocí PLC či RTU jednotek a tato zařízení předávají data dále samotnému SCADA softwaru. SCADA software bývá rozdělen na několik dílčích částí, každá z nich má své specifické určení a oprávnění. Typickým rozdělením je

dělení na operátorské stanoviště určené zejména pro řízení a manažerské (vizualizační) stanoviště, jehož hlavním úkolem je vizualizace dat managementu společnosti. Toto typické rozdělení zobrazuje pro názornou představu níže přiložený obrázek č. 18 [9].




Obrázek č. 17 – Obecné schéma rozdělení SCADA systému; [33]

# 10 Další prostředky vizualizace dat

Mezi další neméně důležité prostředky vizualizace průmyslových dat patří HMI, bez kterých by se většina dnešních strojů neobešla. Jelikož se jedná o systém zobrazující průmyslová data jako systémy SCADA, proto často bývá v literatuře uváděn termín HMI spolu s termínem SCADA [9].

## 10.1 Operátorské panely

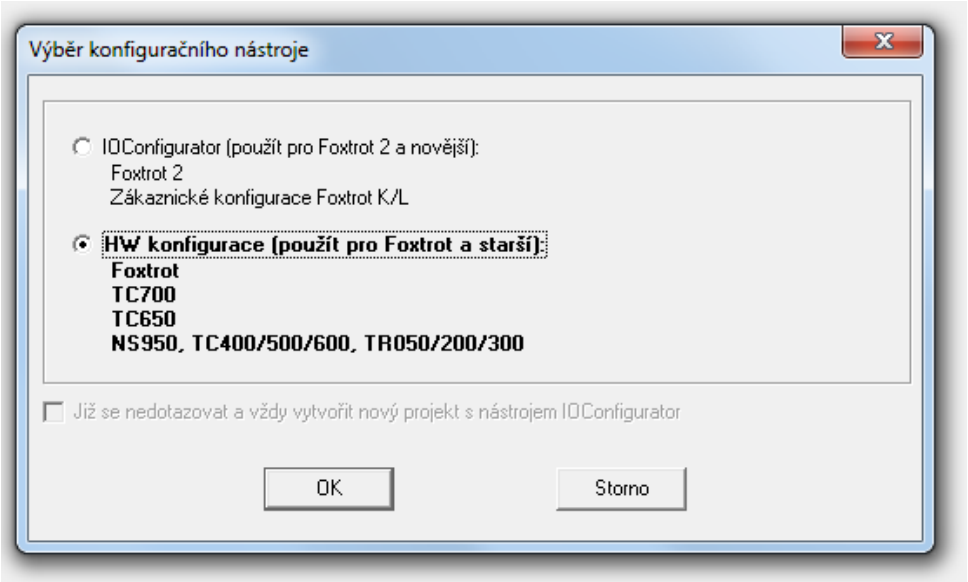
Operátorské panely jsou v dnešní době nastávající čtvrté průmyslové revoluce po boku SCADA systémů jedním z nejdůležitějších prvků této


Obrázek č. 18 – Operátorské panely; [29]

revoluce. Operátorský panel reprezentuje dva druhy aplikací a to aplikace samostatně stojící a integrované. Samostatně stojící aplikace bývají přímo na strojích popřípadě na menších výrobních linkách, oproti integrovaným aplikacím, které bývají spojeny s řídícím popřípadě s hostitelským počítačem a softwarem, který je nahrán v operátorském panelu. Aplikace běžící v operátorských panelech vynikají svou variabilitou, což v praxi znamená, že mohou fungovat na bázi například jako PC, panelové PC, terminálu, popřípadě na mobilním zařízení [9].

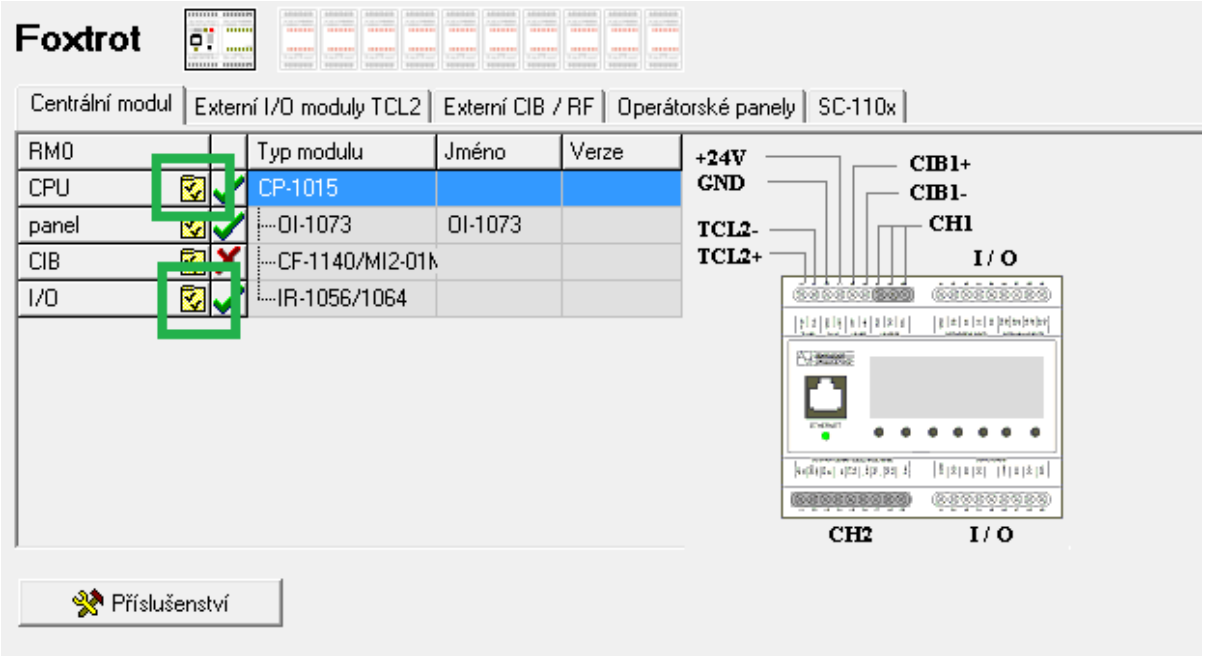
# 11 Navázání komunikace pomocí protokolu MODBUS

Prvním krokem k navázání komunikace mezi PLC a PC je vytvoření skupiny projektů v programu Mosaic. V okně, které se automaticky otevírá po dvojkliku na ikonu vytvoření nového projektu, pojmenujeme danou skupinu a vybereme adresář, kam budeme posléze ukládat data. Po stisknutí tlačítka OK nás program přesune k dalšímu potřebnému kroku a to je výběr konfiguračního nástroje, zde si zvolíme jednu ze dvou nabízených možností dle typu použitého PLC od firmy TECO a.s.



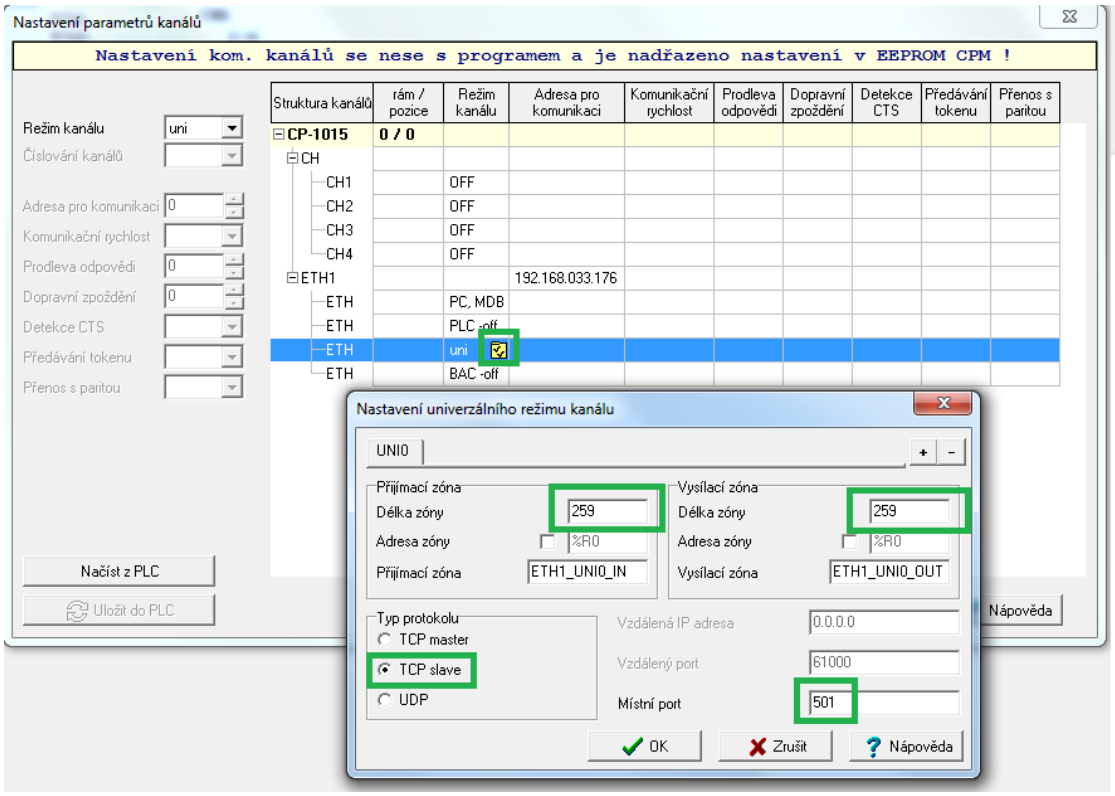

Obrázek č. 19 – Výběr konfiguračního nástroje; [s1]

Po úspěšném výběru a odsouhlasení jsme programem odkázáni k pojmenování aktuálního projektu, popřípadě můžeme importovat již vytvořený projekt na jiném zařízení. Po dalším odsouhlasení jsme přesunuti do části, kde si volíme název hlavního programu a jazyk, ve kterém si přejeme programovat. Jazyky jsou dle normy IEC 1131-3. Po úspěšném vyplnění a odsouhlasení jsme přesunuti již do prostředí, ve kterém lze psát programy. Jedná se vcelku o standardní a poměrně přehledné prostředí schopné konkurovat zavedeným programům například od firmy Siemens. Nyní musíme v manažeru projektu vyprat správný model našeho použitého PLC.



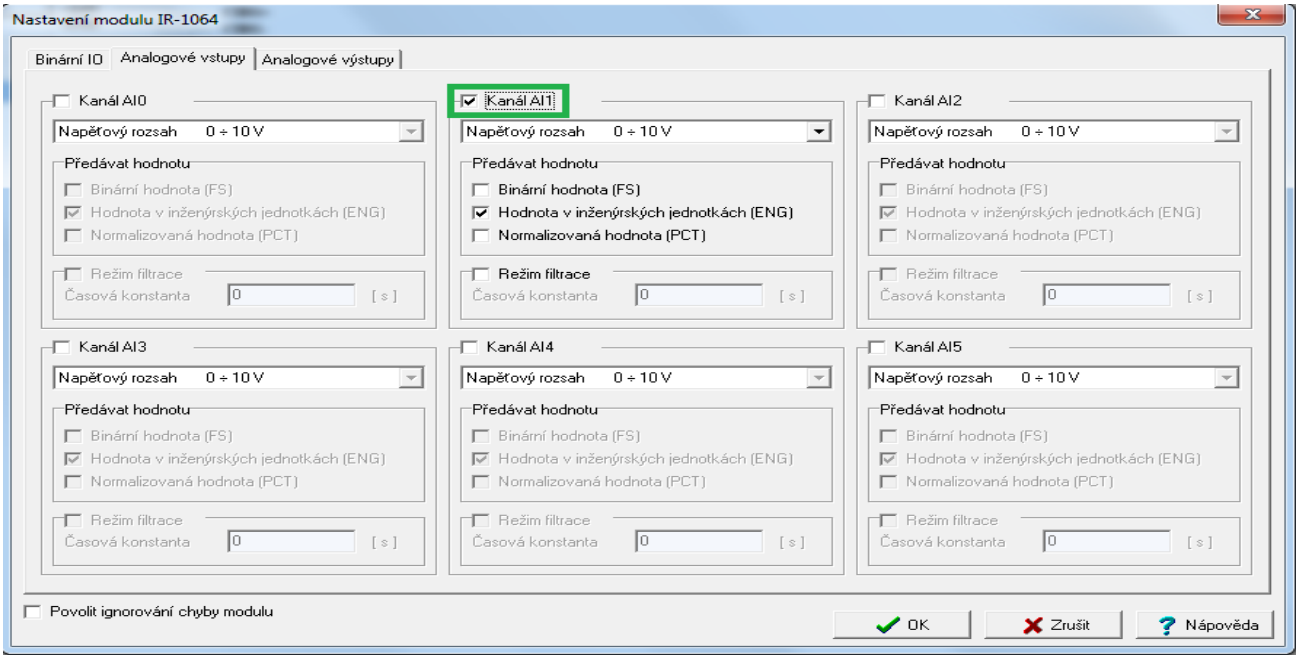

Obrázek č. 20 – Nastavení HW konfigurace; [s1]

Po výběru PLC se přesuneme o záložku níže a nakonfigurujeme PLC dle manuálu, který vytvořila firma Teco a.s., dodávající uživatelskou knihovnu MODBUSRTU_V33_20180109, kterou posléze po konfiguraci budeme hledat v průzkumníku knihoven. Jelikož nemáme na PC, které jsem pro svou bakalářskou práci využil HW klíč, dále popisovaná konfigurace CPU, bude odpovídat Lite licenci. Prvně musíme vypnout všechny komunikační kanály až na kanál ETH1, kde na prvním řádku v tabulce vybereme z výběru „PCMDB" a kanál uvedeme do univerzálního režimu, zbytek řádků nastavíme do stavu „off".




Obrázek č. 21 – Nastavení komunikačního kanálu; [s1]

Nyní nastavíme univerzální režim dle manuálu uživatelské knihovny MODBUSRTU. Po HW konfiguraci se přesuneme do části, konfigurace portů PLC. Pro správné fungování jak programu, tak samotné úlohy postačí, pokud bude zapnut pouze analogový vstupní kanál AI1 a analogový výstupní kanál AO0. Jak u analogového vstupního i výstupního kanálu je nutné nastavit, aby byly informace předávány jako „Hodnoty v inženýrských jednotkách“.




Obrázek č. 22 – Nastavení analogových vstupů; [s1]

Až nyní po konfiguraci je PLC připraveno pro naprogramování komunikace. Prvně musíme přidat k našemu projektu uživatelskou knihovnu MODBUSRTU a přidáme funkční blok MODBUSTCPslave a pomalu začneme přiřazovat proměnné dle manuálu dodavatele knihovny. Jelikož proměnné inputs, coils, inputRegs a holdingRegs odkazují na první vstup či výstup v poli. Prvně musíme tato pole vytvořit a přiřadit jim dostatečnou délku. Samotná pole vedeme jako globální proměnné a jejich maximální délku budeme vest jako globální konstanty, což pro názornost zobrazuje obrázek č. 24. Je nutné dodržet datové typy polí, pokud bychom chtěli přenášet jiné datové typy než UINT a BOOL došlo by ke kolizi, která by měla fatální

následky pro funkčnost komunikace.

```
VAR_GLOBAL CONSTANT
 MODBUS_INPUTS_COUNT        : UINT := 64;
 MODBUS_COILS_COUNT         : UINT := 64;
 MODBUS_INPUT_REG_COUNT     : UINT := 32;
 MODBUS_HOLDING_REG_COUNT   : UINT := 128;
END_VAR


VAR_GLOBAL
 ModbusInputs      : ARRAY [1..MODBUS_INPUTS_COUNT]       OF BOOL;
 ModbusCoils       : ARRAY [1..MODBUS_COILS_COUNT]        OF BOOL;
 ModbusInputReg    : ARRAY [1..MODBUS_INPUT_REG_COUNT]    OF UINT;
 ModbusHoldingReg  : ARRAY [1..MODBUS_HOLDING_REG_COUNT]  OF UINT;
```

Obrázek č. 23 – Příklad deklarace polí pro funkční blok komunikace; [s1]

Nyní se dostáváme do fáze, kdy musíme přiřadit proměnné funkčnímu bloku, který bude mít na starosti komunikaci. Jako příklad přiřazení proměnných slouží obrázek č. 24.

```
PROGRAM prgMain
 VAR
     fbModbusTCPslave1 : fbModbusTCPslave;  //Volání funkčního bloku komunikace
      SimplePID1 : fbSimplePID;    //Volání funkčního bloku regulátoru
 END_VAR

 //Přiřazení proměnných funkčnímu bloku komunikace
 fbModbusTCPslave1(     UnitID := 1, //Identifikátor podřízeného (slave) zařízení
                        chanCode := ETH1_uni0, //Kód komunikačního kanálu
                        inputsCnt := MODBUS_INPUTS_COUNT,  //Počet diskrétních vstupů
                        coilsCnt := MODBUS_COILS_COUNT,   //Počet diskrétních výstupů
                        inputRegCnt := MODBUS_INPUT_REG_COUNT,  //Počet vstupních registrů
                        holdingRegCnt := MODBUS_HOLDING_REG_COUNT,   //Počet registrů
                        inputs := ModbusInputs[1],  //První diskrétní vstup v poli (musí být BOOL)
                        coils := ModbusCoils[1],   // První diskrétní výstup v poli (musí být BOOL)
                        inputRegs := ModbusInputReg[1], // První vstupní registr v poli
                        holdingRegs := ModbusHoldingReg[1]); // První registr v poli
```

Obrázek č. 24 – Deklarace proměnných ve funkčním bloku komunikace; [s1]

# 12 Regulace pomocí PID regulátoru

Pro zajištění PID regulace má program Mosaic několik možností řešení, a to za prvé pomocí implementovaného programu PID maker anebo pomocí funkčních bloků, které se nacházejí v uživatelských knihovnách. Pro řešení této úlohy jsem zvolil možnost regulace pomocí funkčního bloku SimplePID. Tento regulátor je schopný pracovat ve dvou režimech, a to v automatickém režimu a v manuálním režimu. Pro reprezentaci datových typů vstupních a výstupních veličin byl zvolen datový typ REAL, a to pro jeho snazší implementaci do řídícího algoritmu. Pro nastavení chodu slouží proměnné typu BOOL přivedené na funkční blok regulátoru. Pakliže je nastavena hodnota manual TRUE, slouží jako výstup z regulátoru hodnota

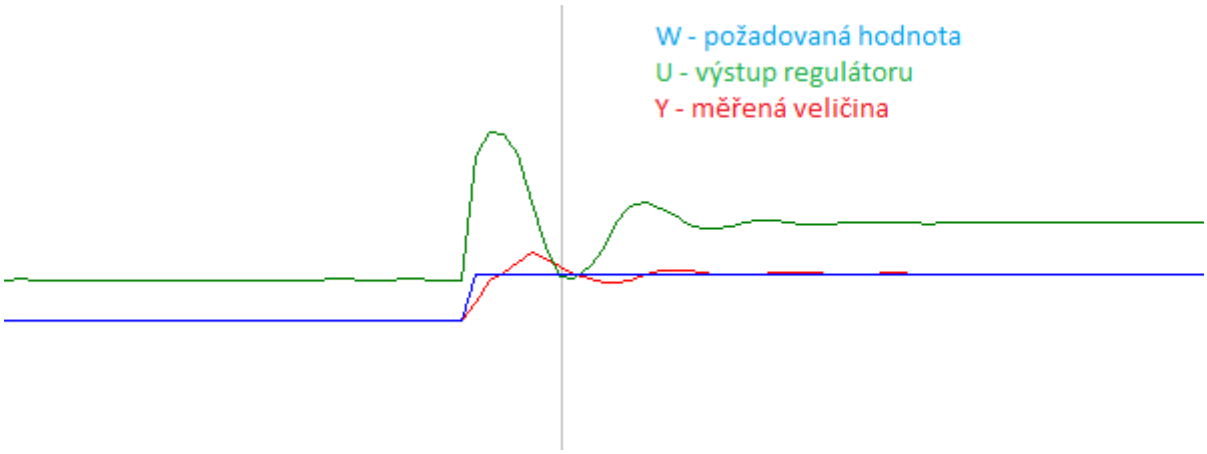
*u_man*. Tento regulátor umožňuje nastavení maximální a minimální hodnoty, které mají za úkol omezit výstup z regulátoru *u*. Na obrázku č. 25 je zobrazen program volající funkční blok regulátoru s přiřazenými vstupními veličinami.

```
 //Přiřazení proměnných funkčnímu bloku
SimplePID1(y      := Y_AO0,
           w      :=varReal[2] ,
           u_man  :=varReal[3] ,
           Gain   :=varReal[4]/1000 ,
           Ti     := varReal[5]/1000,
           Td     := varReal[6]/1000,
           T      := varReal[7]/1000,
           Tf     := varReal[8]/1000,
           dz     := varReal[9]/1000,
           max_u  := varReal[10],
           min_u  := varReal[11],
           e      =>E,
           manual := varBool[1],
           u      => u_AI0);

           IF varBool[1] THEN
              varReal[3] := SimplePID1.u;
                        END_IF; //část programu starající se o man. řízení


END_PROGRAM
```

Obrázek č. 25 – Vyplnění a volání funkčního bloku regulátoru; [s1]




Obrázek č. 26 – Závislost řízeného výstupu na požadované hodnotě; [s1]

Na obrázku č. 26 je znázorněn průběh závislosti řízeného výstupu *y* na požadované hodnotě *w* při nastavení regulátoru: Zesílení: 0,848, Ti: 0,0375,Td: 0,00625 a T: 0,01.

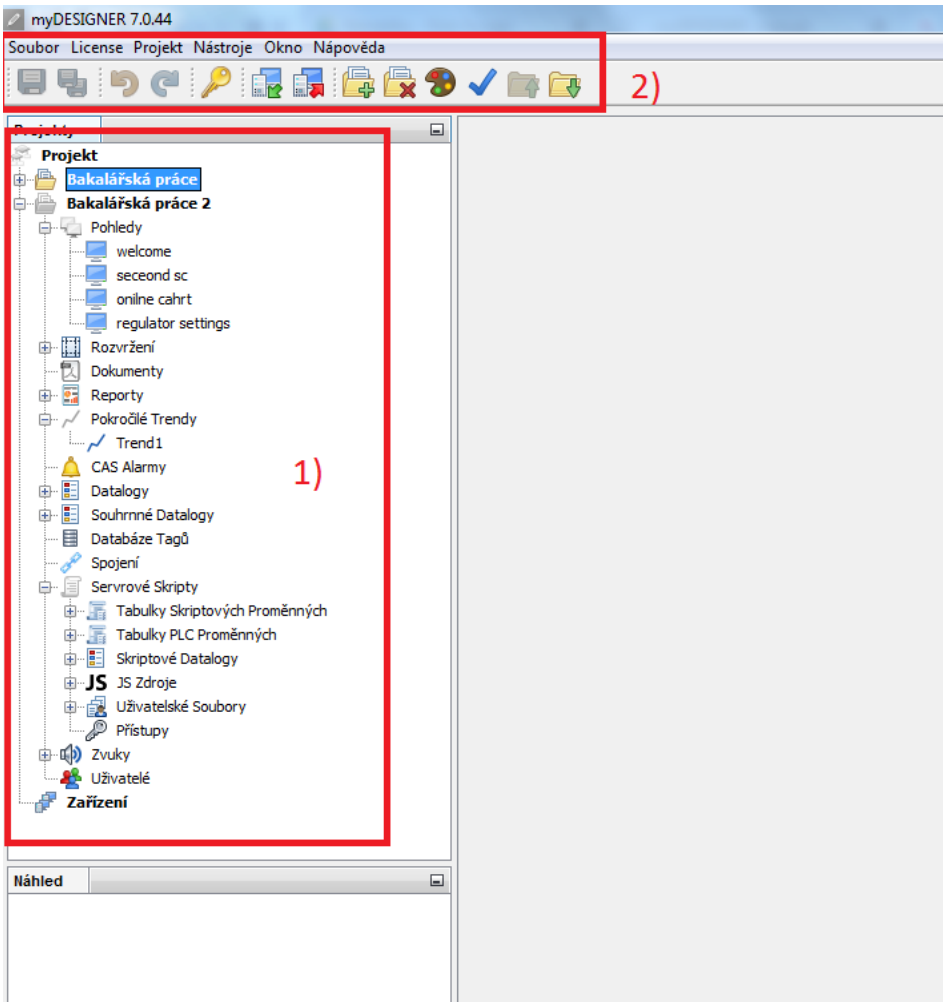
# 13 myDESIGNER

Program myDESIGNER je rychle se vyvíjející platforma pro vizualizace průmyslových dat. Nespornou výhodou tohoto softwaru je uživatelská přívětivost a to zejména intuitivní uživatelské rozhraní. Při tvorbě vlastních rozraní nejsme ničím limitování, avšak dodavatel softwaru již implementoval poměrně bohatou knihovnu prvků, které lze využít v nezměněném stavu, popřípadě je možné prvek kompletně předefinovat. Díky faktu, že je zde grafika kompletně vektorová, není problém ani velikost prvků, kterou si můžeme libovolně volit.

## 13.1 Uživatelské rozhraní

Uživatelské rozhraní programu myDESIGNER je velice uživatelsky přívětivé a i nezkušený uživatel si na něj ve velice krátké době zvykne a osvojí. V levé části se nachází hlavní menu (označené 1) na obrázku č 28.), ve kterém nalezneme ty nejdůležitější položky, jako jsou spojení, tabulka PLC proměnných, pohledy a databáze tagů (proměnných). V hlavním panelu nástrojů (označené 2) na obrázku č. 28) nalezneme funkce, jako je ukládání

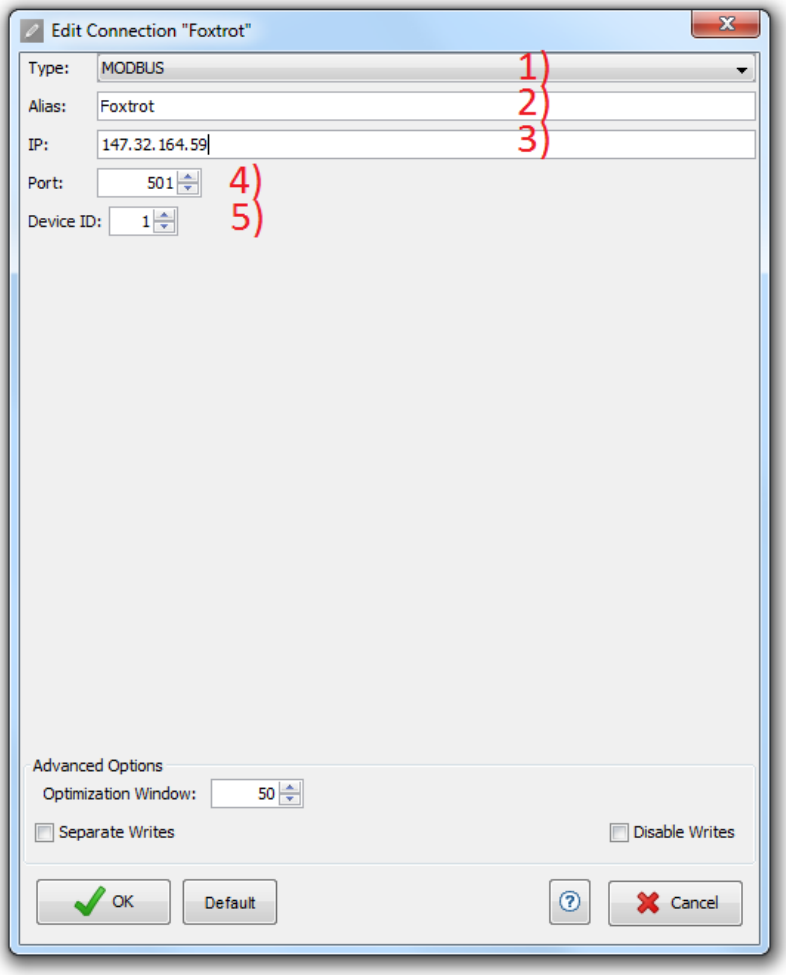
a nahrávání projektu do zařízení. O prvcích, které lze naleznout při vytváření pohledu a přidávání prvků do projektu, bude psáno v podkapitole: „Napojení prvku z knihovny na reálná data z PLC“.




Obrázek č. 27 – Dialogová menu v programu myDESIGNER; [s2]

## 13.2 Navázání komunikace se serverem myPRO

Popis navázání komunikace se serverem myPRO nejlépe vystihuje obrázek č. 29, který je níže popsán.




Obrázek č. 28 – Konfigurace serveru myPRO pro MODBUS komunikaci s PLC; [s2]

Postup vytvoření připojení:

1) Volba typu připojení. Pro tuto úlohu volíme typ připojení pomocí protokolu MODBUS

2) Alias, tedy přiřazení jména připojení

3) Do kolonky IP je třeba zadat IP adresu PLC

4) Do kolonky Port zadáme hodnotu místního portu viz. obrázek č. 21

5) Zde vyplníme identifikátor zařízení, který jsme mu přidělily při deklaraci funkčního bloku komunikace viz. obrázek č. 24.

Pro správné navázání komunikace je třeba nejprve uvést PLC do stavu „halt" a poté je možné nahrát projekt z programu myDESIGNER. PLC uvedeme do stavu

„run“ teprve až po nahrání programu na server. Při nedodržení tohoto postupu se komunikace nenaváže.

## 13.3 Napojení prvku z knihovny na reálná data z PLC

Pro napojení prvků na prvky z knihovny programu myDESIGNER budeme využívat proměnné uložené v polích holdingReg a inputReg. Pro úspěšné propojení proměnných musíme vytvořit a pojmenovat nový seznam PLC proměnných v hlavním menu.

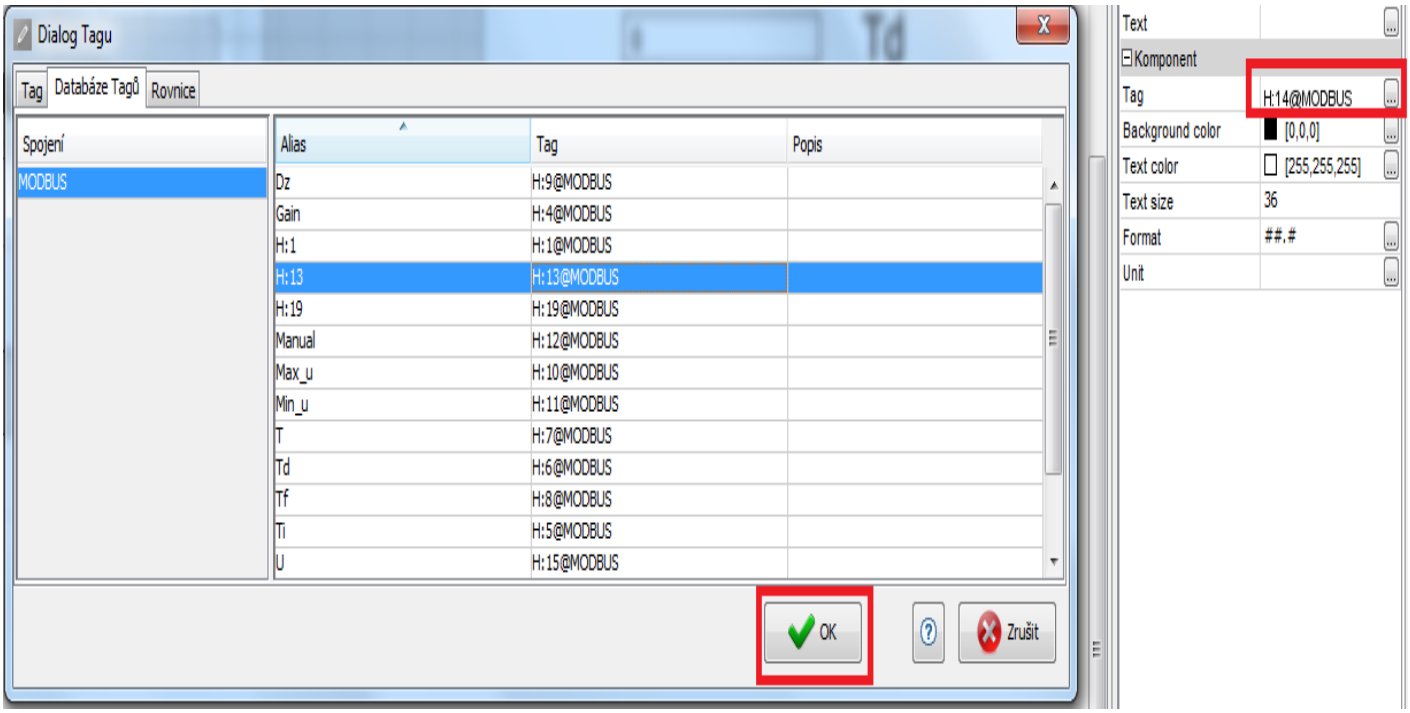
| Jméno proměnné | Tag@Spoj/*Alias | Typ | Počet Elem... |
|---|---|---|---|
| w | H:1@MODBUS | Číslo | 1 |
| u_man | H:2@MODBUS | Číslo | 1 |
| gain | H:3@MODBUS | Číslo | 1 |
| ti | H:4@MODBUS | Číslo | 1 |
| td | H:5@MODBUS | Číslo | 1 |
| t | H:6@MODBUS | Číslo | 1 |
| tf | H:7@MODBUS | Číslo | 1 |
| dz | H:8@MODBUS | Číslo | 1 |
| max_u | H:9@MODBUS | Číslo | 1 |
| min_u | H:10@MODBUS | Číslo | 1 |
| y | H:13@MODBUS | Číslo | 1 |
| u | H:14@MODBUS | Číslo | 1 |
| manual | H:11@MODBUS | Číslo | 1 |
| w1000 | H:19@MODBUS | Číslo | 1 |
| | | Číslo | 1 |

| Tabulky PLC Proměnných | |
|---|---|
| Jméno | PLC_proměnné |
| Výchozí PLC | MODBUS |
| Obnovení [msec] | 500 |
| Počet tagů | 13 |

Obrázek č. 29 – Příklad vytvoření tabulky s PLC proměnnými; [s2]

Po dokončení předešlého kroku musíme zadat a pojmenovat proměnné, které si přejeme zobrazovat. Při zadávání proměnných musíme brát na zřetel umístění proměnné v poli. Po úspěšné deklaraci proměnných se přesuneme v hlavním menu na položku „pohledy“ a vytvoříme nový pohled. Pro přidání prvků slouží malá ikona v podobě knihy. Po kliknutí na ikonu se otevře knihovna prvků a my si můžeme zvolit jak nabízené prvky výrobcem, tak si můžeme nadefinovat nové prvky. Pro úspěšné svázání prvku s proměnnou (tagem) je nutné danou proměnou přiřadit k danému prvku a to přes vlastnosti prvku, kde dvojklikem na tlačítko nacházející se

na řádku „Tag“ otevřeme okno „Dialog Tagu“ a v databázi proměnných si vybereme požadovanou proměnnou.




Obrázek č. 30 – Příklad propojení proměnné s prvkem; [s2]

# 14 Závěr

V této práci byly představeny komunikační protokoly logických automatů dodávaných českou firmou Teco a.s. Tyto protokoly byly v teoretické části rešeršně popsány a v praktické části byl použit komunikační protokol MODBUS k navázání komunikace mezi logickým automatem a SCADA systémem.

Během praktické části, která se věnovala navázání komunikace, jsem se potýkal s nesčetnými problémy, které pramenily z neznalosti programovacího prostředí Mosaic. Po důkladném seznámení se s tímto programovacím prostředím jsem byl schopen zprovoznit komunikaci mezi PLC a SCADA systémem pomocí protokolu MODBUS.

Pro vzdálené řízení úlohy bylo vytvořeno uživatelské rozhraní pomocí programu myDESIGNER. Při vytváření rozhraní byl kladen důraz na jednoduchost a přehlednost. Nespornou výhodou byl fakt, že firma dodávající program myDESIGNER vytvořila celou sadu užitečných návodů, které mi při vytváření rozhraní přišly vhod.

Celá úloha byla vyzkoušena a odladěna v laboratoři číslo 111 a 109 Ústavu přístrojové a řídící techniky při ČVUT.

# Seznam zkratek

PDU - Protocol data unit

ADU - application data unit

LAN - Local Area Network

WAN - Wide Area Network

Http - Hypertext Transfer Protocol

FTP - File Transfer Protocol

DNS - Domain Name System

DHCP - Dynamic Host Configuration Protocol)

UDP - User Datagram Protocol

HMI - Human Machine Interface

RTU - Remote Terminal Unit

PLC - Programmable Logic Controller

ICMP - Internet Control Message Protocol

PING - Packet InterNet Groper

CAN - Controller Area Network

CRC - Cyclic redundancy check

ACK - Acknowledge bit

ACD - Acknowledge delimeter

CMS - CAN Message specification

NMT - Network Management

DBT - Distributor

LMT - Layer Management

SCADA - Supervisory Control And Data Acquisition

CPU - Central Processing Unit

# Zdroje

[1] Zezulka František, *Automatizační prostředky*. Skripta VUT FEKT v Brně, 1999, ISBN 80-214-1482-0

[2] KOCOUREK, Petr a Jiří NOVÁK. *Přenos informace*. Praha: Vydavatelství ČVUT, 2004. ISBN 80-010-2892-5.

[3] VOJÁČEK, Antonín. MODBUS [online]. [cit. 2018-06-09]. Dostupné z: https://automatizace.hw.cz/clanek/2004070701

[4; 24] MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b3 [online]. In: . 2012, 26.4.2012, s. 50 [cit. 2018-06-09]. Dostupné z: www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

[5] Modbus TCP/IP [online]. Kanada [cit. 2018-06-09]. Dostupné z: http://www.simplymodbus.ca/TCP.htm

[6] MODBUS PROTOCOL ON OSI MODEL. In: Protoconvert [online]. [cit. 2017-11-13]. Dostupné z: http://www.protoconvert.com/TechnicalResourses/Tutorials/Modbus.aspx

[7] POLÁK, Karel. Elektrorevue. Elektrorevue [online]. [cit. 2018-02-014]. Dostupné z: http://www.elektrorevue.cz/clanky/03021/index.html

[8] MODBUS transaction. In: FreeSCADA [online]. [cit. 2017-11-13]. Dostupné z: http://free-SCADA.org/ls4/blog/tutorials/28.html

[9] PRIMUS, Tomáš. Systémy SCADA a nástroje pro sběr, vizualizaci a analýzu průmyslových dat [online]. Praha, 2017 [cit. 2018-06-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/70825. Bakalářská práce. ČVUT. Vedoucí práce Doc. Ing. Ivo Bukovský, Ph.D.

[10] PLICKA, Martin. FTP (File Transfer Protocol) [online]. In: . [cit. 2018-06-09]. Dostupné z: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=7&ved=0ahUKEwiOuuTQkeDaAhVCaFAKHRHiBTYQFghpMAY&url=https%3A%2F%2Fmplicka.cz%2F__media%2Fdownload%2Fsos%2Fftp.pdf&usg=AOvVaw2CUytCnAJ3xr3gWPt4BAqk

[11] TCP/IP. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-06-09]. Dostupné z: https://cs.wikipedia.org/wiki/TCP/IP

[12] PETERKA, Jiří. Síťový model TCP/IP [online]. 1992 [cit. 2018-06-09]. Dostupné z: http://www.earchiv.cz/a92/a231c110.php3

[13] DNS [online]. [cit. 2018-06-09]. Dostupné z: http://www.adaptic.cz/znalosti/slovnicek/dns/

[14] O doménách a DNS. Cz.nic [online]. [cit. 2018-06-09]. Dostupné z: https://www.nic.cz/page/312/o-domenach-a-dns/

[15] Počítačové sítě - Model ISO/OSI [online]. [cit. 2018-06-09]. Dostupné z: http://site.the.cz/index.php?id=4

[16] PETERKA, Jiří. Eferenční model ISO/OSI - sedm vrstev [online]. [cit. 2018-06-09]. Dostupné z: http://www.earchiv.cz/a92/a213c110.php3

[17] PETERKA, Jiří. LAN vs. WAN [online]. 1996 [cit. 2018-06-09]. Dostupné z: http://www.earchiv.cz/a96/a615k150.php3

[18] Protokolová datová jednotka [online]. [cit. 2018-06-09]. Dostupné z: http://cs.dbpedia.org/page/Protokolov%C3%A1__datov%C3%A1__jednotka

[19] DHCP [online]. [cit. 2018-06-09]. Dostupné z: https://it-slovnik.cz/pojem/dhcp

[20] Automatické přidělení IP adres pomocí DHCP serveru Více na: https://www.zive.cz/clanky/automaticke-prideleni-ip-adres-pomoci-dhcp-serveru/sc-3-a-5765/default.aspx [online]. [cit. 2018-06-09]. Dostupné z: https://www.zive.cz/clanky/automaticke-prideleni-ip-adres-pomoci-dhcp-serveru/sc-3-a-5765/default.aspx

[21] PETERKA, Jiří. TCP a UDP [online]. 1999 [cit. 2018-06-09]. Dostupné z: http://www.earchiv.cz/anovinky/ai1864.php3

[22] HMI (vizualizace a ovládání) [online]. [cit. 2018-06-09]. Dostupné z: http://plc-automatizace.cz/knihovna/hmi.htm

[23] KUMAR, Shubham. Remote terminal unit. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-06-09]. Dostupné z: https://en.wikipedia.org/wiki/Remote__terminal__unit

[24] RONEŠOVÁ, Andrea. Přehled protokolu MODBUS [online]. 2005 [cit. 2018-03-09]. Dostupné z: home.zcu.cz/~ronesova/bastl/files/modbus.pdf

[25] CP-1003 [online]. [cit. 2018-06-13]. Dostupné z: https://www.tecomat.cz/Products/cz/plc-tecomat-foxtrot/zakladni-moduly/125-cp-1003/

[26] CP-1015 [online]. Kolín [cit. 2018-06-09]. Dostupné z: https://www.tecomat.cz/Products/cz/plc-tecomat-foxtrot/zakladni-moduly/131-cp-1015/

[27] CP-1018 [online]. Kolín [cit. 2018-06-09]. Dostupné z: https://www.tecomat.cz/Products/cz/plc-tecomat-foxtrot/zakladni-moduly/133-cp-1018/

[28] Tc700 [online]. In: . [cit. 2018-06-09]. Dostupné z: http://docplayer.ru/54809395-Smart-city-tehnologii-i-novye-vozmozhnosti.html

[29] Simatic basic panel [online]. In: . [cit. 2018-06-09]. Dostupné z: https://gbu-dialog.ru/simatic-basic-panel.html

[30] 3. TCP/IP [online]. [cit. 2018-06-11]. Dostupné z: www.ped.muni.cz/wtech/03_studium/teps/teps-03.pdf

[31] Síťový model TCP/IP [online]. [cit. 2018-06-11]. Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=10011

[32] JAK SE DOSPĚLO K INTERNETU [online]. In: . 2014 [cit. 2018-04-06]. Dostupné z: http://c4-isr.blogspot.com/2014/04/

[33] BEER, Jan. SCADA systémy pro průmyslové aplikace. Plzeň, 2015. Bakalářská práce. Západočeská univerzita v Plzni.

[34] ŠRÁMEK, Filip. Komunikační protokoly. Praha, 2018. Projekt. České vysoké učení technické. Vedoucí práce Ing. Pavel Trnka.

[35] The ISO OSI Reference Model [online]. In: . [cit. 2018-06-13]. Dostupné z: https://www.jackcola.org/2012/11/the-iso-osi-reference-model/

# Seznam obrázků

# Seznam příloh

Příloha A: …… Přiložené CD – Bakalářská_práce_šrámek, Mosaic.zip, Rozhraní.zip

- Bakalářská_práce_šrámek.pdf – elektronická kopie bakalářské práce
- Mosaic.zip – program vytvořený v prostředí Mosaic
- Rozhraní.zip –uživatelské rozhraní pro SCADA systém

Příloha B: …… Výpis programu

# Použitý software

[s1] Mosaic 2018.1 SP1

[s2] myDESIGNER

# Seznam rovnic

# Seznam tabulek

# Příloha B

```
VAR_GLOBAL CONSTANT
 MODBUS_INPUTS_COUNT        : UINT := 32;
 MODBUS_COILS_COUNT         : UINT := 32;
 MODBUS_INPUT_REG_COUNT     : UINT := 16;
 MODBUS_HOLDING_REG_COUNT   : UINT := 32;
END_VAR


VAR_GLOBAL
 ModbusInputs      : ARRAY [1..MODBUS_INPUTS_COUNT]       OF BOOL;
 ModbusCoils       : ARRAY [1..MODBUS_COILS_COUNT]        OF BOOL;
 ModbusInputReg    : ARRAY [1..MODBUS_INPUT_REG_COUNT]    OF UINT;
 ModbusHoldingReg  : ARRAY [1..MODBUS_HOLDING_REG_COUNT] OF UINT;
 varReal      : ARRAY[1..20] OF REAL;
 varBool      : ARRAY[1..2] OF BOOL;
 LastError    : STRING;
 MdbError     : STRING;
 MdbData      : ARRAY [0..1] OF UINT;
 E:REAL:=0;
 w0:REAL;
 Y_AO0 AT r0_p3_AI1.ENG:REAL;
 u_AI0 AT r0_p3_AO0.ENG:REAL;
```

```
 Manual         : BOOL;
 SetPoint       : REAL;
 ManualValue    : REAL;
 U_man          : REAL;
 b              : ARRAY [1..1] OF UINT;
 W2 :UINT;

END_VAR

PROGRAM prgMain
 VAR
   ModbusSlave1 : fbModbusTCPslave;
   SimplePID1 :   fbSimplePID;
 END_VAR
 ModbusHoldingReg[14]:=REAL_TO_UINT(Y_AO0*1000);
 ModbusHoldingReg[15]:=REAL_TO_UINT(u_AI0*1000);
 ModbusSlave1( UnitID := 1,
                chanCode := ETH1_uni0,
                inputsCnt := MODBUS_INPUTS_COUNT,
                coilsCnt := MODBUS_COILS_COUNT,
                inputRegCnt := MODBUS_INPUT_REG_COUNT,
                holdingRegCnt := MODBUS_HOLDING_REG_COUNT,
```

```
                inputs := ModbusInputs[1],
                coils := ModbusCoils[1],
                inputRegs := ModbusInputReg[1],
                holdingRegs := ModbusHoldingReg[1]
                );

IF GetModbusErrTxt(ErrorCode := ModbusSlave1.ErrCode, ErrTxt := MdbError) THEN
   LastError := MdbError;
 END_IF;
   w2:=ModbusHoldingReg[2];
   varReal[2]  :=   UINT_TO_REAL(ModbusHoldingReg[2]);//W
   varReal[3]  :=   UINT_TO_REAL(ModbusHoldingReg[3]);//u_man
   varReal[4]  :=   UINT_TO_REAL(ModbusHoldingReg[4]);//Gain
   varReal[5]  :=   UINT_TO_REAL(ModbusHoldingReg[5]);//Ti
   varReal[6]  :=   UINT_TO_REAL(ModbusHoldingReg[6]);//Td
   varReal[7]  :=   UINT_TO_REAL(ModbusHoldingReg[7]);//T
   varReal[8]  :=   UINT_TO_REAL(ModbusHoldingReg[8]);//Tf
   varReal[9]  :=   UINT_TO_REAL(ModbusHoldingReg[9]);//dz
   varReal[10]  :=  UINT_TO_REAL(ModbusHoldingReg[10]);//max_u
   varReal[11]  :=  UINT_TO_REAL(ModbusHoldingReg[11]);//min_u
   varBool[1]  :=   UINT_TO_BOOL(ModbusHoldingReg[12]);//manuální režim=0 automatický režim=1
```

```
 ModbusHoldingReg[20]:=W2*1000;
SimplePID1(y      := Y_AO0,
           w      :=varReal[2] ,
           u_man  :=varReal[3] ,
           Gain   :=varReal[4]/1000 ,
           Ti     := varReal[5]/1000,
           Td     := varReal[6]/1000,
           T      := varReal[7]/1000,
           Tf     := varReal[8]/1000,
           dz     := varReal[9]/1000,
           max_u  := varReal[10],
           min_u  := varReal[11],
           e      =>E,
           manual := varBool[1],
           u      => u_AI0);

                  IF varBool[1] THEN
                     varReal[3] := SimplePID1.u;
                                 END_IF;


END_PROGRAM
```