

# **Využití standardního protokolu MODBUS pro přenos informace z čteček RFID karet v energetice**

The Exploitation of Modbus Protocol for the Transmission of  
Information  
from an RDIF Card in Power Engineering

Bc. Vratislav Křivák



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky  
akademický rok: 2011/2012

## **ZADÁNÍ DIPLOMOVÉ PRÁCE**

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Vratislav KŘIVÁK**  
Osobní číslo: **A10486**  
Studijní program: **N 3902 Inženýrská informatika**  
Studijní obor: **Bezpečnostní technologie, systémy a management**

Téma práce: **Využití standardního protokolu Modbus pro přenos informace z bezkontaktních čteček karet v energetice**

Zásady pro vypracování:

1. Definujte funkci standardního protokolu Modbus po sériové lince a jeho využití pro instalaci bezpečnostních zařízení v energetických systémech.
2. Vysvětlete funkce řídicích systémů v energetice s hlediska možné implementace čteček RFID karet.
3. Vytvořte program, který umožní sledovat vzájemnou komunikaci mezi zařízeními pomocí protokolu Modbus RTU a popište jeho funkci.
4. Implementujte bezkontaktní čtečku karet do energetických systémů ASŘ a navrhnete projekt integrace pro zvýšení bezpečnosti práce v energetickém průmyslu.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Process Automation College: UxS System Administration. Technical Reference Manual, vydané 27.8 1996.**
2. **PROKOPOVA, Zdenka. Databázové systémy MySQL + PHP. Vyd. 1. Zlín : Univerzita Tomáše Bati ve Zlíně, 2006. 126 s. ISBN 80-7318-486-9.**
3. **KAINKA, Burkhard ; BERNT, Hans-Joachim. Využití rozhraní PC pod Windows. Vyd. 1. Ostrava : HEL, 2000. 151 s. ISBN 80-86167-13-5.**
4. **Modicon Modbus Protocol Reference Guide. North Andover, Massachusetts : Modicon, Inc., 1994. 115 s.**
5. **RONESOVÁ, Andrea. [Http://home.zcu.cz/ronesova/index.php?menuitem=bastlirna](http://home.zcu.cz/ronesova/index.php?menuitem=bastlirna) [online]. Upravené vydání. 2005 [cit. 2011-11-28]. [Http://home.zcu.cz/ronesova/bastl/files/modbus.pdf](http://home.zcu.cz/ronesova/bastl/files/modbus.pdf). Dostupné z WWW: [<http://home.zcu.cz/>].**
6. **Process Automation College: Local/UCN/APM Maintenance. Technical Reference Manual, vydané 4.6 1992.**

Vedoucí diplomové práce:

**Ing. Rudolf Drga**

Ústav bezpečnostního inženýrství

Datum zadání diplomové práce:

**24. února 2012**

Termín odevzdání diplomové práce:

**15. května 2012**

Ve Zlíně dne 24. února 2012

prof. Ing. Vladimír Vašek, CSc.  
*děkan*



doc. RNDr. Vojtěch Křesálek, CSc.  
*ředitel ústavu*

## ABSTRAKT

MODBUS je jedním z nejrozšířenějších protokolů v energetických systémech ASŘ, který slouží ke vzájemné komunikaci různých zařízení umožňující přenos dat po nehomogenních systémech, sběrnicích a sítích, pracujících na principu předávání datových zpráv mezi klientem a serverem. Cílem této diplomové práce je implementace prvků bezpečnostních systémů do systémů ASŘ v energetice a vyzvednutí univerzálnosti protokolu MODBUS pro vzájemnou integraci těchto systémů. Diplomová práce obsahuje návrh vzájemné integrace, specifikace a popis konečného řešení samotného projektu za účelem zvýšení bezpečnosti práce na energetickém zařízení.

Klíčová slova: Modbus RTU, Honeywell, Access\_Energo, Access\_ComPort\_Modbus, PLC Fatek FBs-40MCR2-D24, Wie485.

## ABSTRACT

MODBUS is one of the most common protocols in energetic automatical systems. It is used for the communication between devices that allow the data transmission within the non-homogenous systems, as well as the collectors and cites that are used in data transportation between the server and the client. The goal of this essay is the implementation of the elements in the security systems of the power engineering, highlighting the importance of the universal MODBUS protocol, and its usage in the integration of the security systems . The essay contains the proposal how to integrate and incorporate the systems; it also has a detail description for the final resolution of the whole project. Overall, the purpose of the proposal is to produce increase in safety of the working environment within the power engineering institutions.

Keywords: Modbus RTU, Honeywell, Access\_Energo, Access\_ComPort\_Modbus, PLC Fatek FBs-40MCR2-D24, Wie485.

Děkuji tímto svému vedoucímu bakalářské práce Ing. Rudolfu Drgovi za odborné vedení, cenné rady a připomínky, které mi poskytoval během konzultací.

**Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji,**

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....  
podpis diplomanta

## OBSAH

<b>ÚVOD .....</b>	<b>11</b>
<b>I TEORETICKÁ ČÁST .....</b>	<b>12</b>
<b>1 STANDARDNÍ PROTOKOL MODBUS.....</b>	<b>13</b>
1.1 KÓDOVÁNÍ DAT A DATOVÝ MODEL.....	15
1.2 DEFINICE MODBUS TRANSAKCE .....	16
1.3 TRANSAKCE V MODBUS SÍTÍCH .....	18
1.4 TRANSAKCE NA DALŠÍCH TYPECH POČÍTAČOVÝCH SÍTÍ .....	18
<b>2 PŘENOSOVÉ MÓDY MODBUSU POUŽÍVANÉ NA SÉRIOVÉ LINCE.....</b>	<b>20</b>
2.1 ASCII A RTU MÓD.....	20
2.2 RÁMEC SE ZPRÁVOU NA MODBUSU .....	21
2.2.1 ASCII rámec.....	21
2.2.2 RTU rámec .....	22
2.2.3 Získání adresného prostoru a funkčního pole .....	22
<b>3 OBSAH CHYBOVÉHO SOUČTU A JEHO METODY .....</b>	<b>25</b>
3.1 ASCII.....	25
3.2 RTU .....	25
3.3 METODY KONTROLNÍHO SOUČTU .....	26
3.4 KONTROLA PARITY .....	26
3.5 LRC CHECKING .....	27
3.6 CRC CHECKING .....	27
3.6.1 Užití Součtového bajtu.....	28
3.7 DATA ADRESOVANÉ V MODBUS ZPRÁVĚ .....	28
3.8 ZÁPORNÉ ODPOVĚDI.....	29
<b>4 POPIS FUNKČNÍCH KÓDŮ .....</b>	<b>30</b>

4.1	01 (0x01) ČTI CÍVKY (READ COILS).....	30
4.2	02 (0x02) ČTI DISKRÉTNÍ VSTUPY (READ DISCRETE INPUTS) .....	31
4.3	03 (0x03) ČTI UŽIVATELSKÉ REGISTRY (READ HOLDING REGISTERS) .....	31
4.4	04 (0x04) ČTI SYSTÉMOVÉ REGISTRY (READ INPUT REGISTERS) .....	32
4.5	05 (0x05) ZAPIŠ JEDNU CÍVKU (WRITE SINGLE COIL).....	32
4.6	06 (0x06) ZAPIŠ JEDEN REGISTR (WRITE SINGLE REGISTER).....	33
4.7	07 (0x07) ČTI STAV (READ EXCEPTION STATUS) – POUZE PRO SÉRIOVOU LINKU .....	34
4.8	08 (0x08) DIAGNOSTIKA (DIAGNOSTICS) – POUZE PRO SÉRIOVOU LINKU.....	34
4.9	11 (0x0B) ČTI ČÍTAČ KOMUNIKAČNÍCH UDÁLOSTÍ (GET COMM EVENT COUNTER) – POUZE PRO SÉRIOVOU LINKU .....	35
4.10	12 (0x0C) ČTI ZÁZNAM KOMUNIKAČNÍCH UDÁLOSTÍ (GET COMM EVENT LOG) – POUZE PRO SÉRIOVOU LINKU .....	36
4.11	15 (0x0F) ZAPIŠ VÍCE CÍVEK (WRITE MULTIPLE COILS) .....	36
4.12	16 (0x10) ZAPIŠ VÍCE REGISTRŮ (WRITE MULTIPLE REGISTERS).....	37
4.13	17 (0x11) SDĚL IDENTIFIKACI (REPORT SLAVE ID) – POUZE PRO SÉRIOVOU LINKU .....	37
4.14	20 / 6 (0x14 / 0x06) ČTI ZÁZNAM ZE SOUBORU (READ FILE RECORD) .....	37
4.15	21 / 6 (0x15 / 0x06) ZAPIŠ ZÁZNAM DO SOUBORU (WRITE FILE RECORD).....	38
4.16	22 (0x16) ZAPIŠ REGISTR S MASKOVÁNÍM (MASK WRITE REGISTER) .....	38
4.17	23 (0x17) ČTI/ZAPIŠ VÍCE REGISTRŮ (READ/WRITE MULTIPLE REGISTERS) .....	38
4.18	24 (0x18) ČTI FIFO FRONTU (READ FIFO QUEUE).....	39
4.19	43 / 14 (0x2B / 0x0E) ČTI IDENTIFIKACI ZAŘÍZENÍ (READ DEVICE IDENTIFICATION).....	39
<b>5</b>	<b>DEFINICE ŘÍDICÍHO SYSTÉMU V ENERGETICE HONEYWELL S POHLEDU MOŽNÉ IMPLEMENTACE ČTEČKY RFID KARET .....</b>	<b>40</b>
<b>6</b>	<b>SOUČASNÝ STAV ŘEŠENÍ IMPLEMENTACÍ ČTEČEK RFID KARET DO ŘÍDICÍCH SYSTÉMŮ PO MODBUSU.....</b>	<b>45</b>
6.1	SUBMODUL MX-0301 PRO PŘIPOJENÍ ČTEČKY KARET WIEGAND .....	45
6.2	RFID ČTEČKY EMS COBALT C.....	46
6.3	I/O CONTROLLER PL.....	47
6.4	REGULÁTOR A INTEGRÁTOR PRO APLIKACE AUTOMATIZACE BUDOV – HAWK.....	48
6.5	Wie485 - PŘEVODNÍK WIEGAND NA RS485 .....	49
<b>II</b>	<b>PRAKTICKÁ ČÁST.....</b>	<b>51</b>
<b>7</b>	<b>DEFINICE A NÁVRH SYSTÉMU ACCESS_ENERGO.....</b>	<b>52</b>
7.1	DEFINICE POŽADOVANÉ INTEGRACE ZA ÚČELEM ZVÝŠENÍ BEZPEČNOSTI PRÁCE V ENERGETICE ZA POMOCÍ ČTEČKY RFID KARET .....	52
7.2	ANALÝZA PROSTŘEDÍ A VSTUPNÍ INFORMAČNÍCH DAT DO PROCESU INTEGRACE.....	53
7.3	IMPLEMENTACE ČTEČEK RFID KARET DO SYSTÉMŮ ASŘ HONEYWELL.....	56
7.4	KONCEPČNĚ PROGRAMOVÉ POJETÍ PROJEKTU IMPLEMENTACE ČTEČEK .....	58
<b>8</b>	<b>ZPRACOVÁNÍ DATABÁZE PRACOVNÍKŮ A OBJEKTŮ.....</b>	<b>60</b>



8.1	PROPOJENÍ PHP+MySQL .....	60
8.2	VYTVOŘENÁ DATABÁZE PROJEKTU V MySQL .....	61
8.2.1	Databázová tabulka ZAMĚSTNANCI .....	62
8.2.2	Databázová tabulka ACCESS_ROZVODNY a VÝVODY .....	63
8.2.3	Databázová tabulka KVALIFIKACE .....	64
8.2.4	Databázová tabulka NAPĚTÍ .....	65
8.2.5	Databázová tabulka OBJEKT .....	65
8.2.6	Databázová tabulka ODDĚLENÍ a POVEŘENÍ .....	65
8.2.7	Databázová tabulka ROZVODNY .....	66
<b>9</b>	<b>VIZUALIZACE PROJEKTU POMOCÍ EDITORU WEBOVÝCH STRÁNEK HTML A SKRIPTOVACÍHO JAZYKU PHP .....</b>	<b>67</b>
9.1	ZÁKLADNÍ WEBOVÁ APLIKACE A ZOBRAZENÍ ROZHRANÍ .....	69
9.1.1	Zobrazení stránek po výběru tlačítkem – Výběr zaměstnanců .....	71
9.1.2	Zobrazení stránek po výběru tlačítkem – Výběr zaměstnanců .....	72
9.1.3	Zobrazení stránek po výběru tlačítkem – Přístup do rozvodu .....	73
<b>10</b>	<b>HLAVNÍ APLIKACE ACCESS_COMPORT_MODBUS .....</b>	<b>78</b>
10.1	FORMULÁŘ ZÁKLADNÍ APLIKACE A JEHO POPIS .....	80
10.2	POPIS FUNKCÍ HLAVNÍ APLIKACE ACCESS_COMPORT_MODBUS .....	84
10.2.1	Vlákno pro čtení dat z COM1 .....	84
10.2.2	Vlákno s dotazem žádostí čtení registrů z PLC .....	85
10.2.3	Hlavní aplikace Access_ComPort_Modbus .....	86
10.2.4	Kontrolní součet CRC pro přenos informace po Modbusu .....	89
10.2.5	ADT Seznam pro ukládání informací .....	92
<b>11</b>	<b>PROGRAMOVACÍ AUTOMAT A AKČNÍ ČLEN INTEGRACE .....</b>	<b>94</b>
11.1	PROGRAMOVACÍ AUTOMAT PLC FATEK FBS-40MCR2-D24 A VYUŽITÍ KOMPONENTY MODBUS PRO KOMUNIKACI MEZI ZAŘÍZENÍMI .....	96
11.2	FYZICKÉ ZAPOJENÍ A NASTAVENÍ KOMUNIKACE NA SÉRIOVÝCH PORTECH PLC .....	105
11.2.1	Nastavení komunikačních portů PLC Fatek FBS-40MCR2-D24 v projektu Access_Energo .....	108
11.2.1.1	Nastavení Portu 0 .....	109
11.2.1.2	Nastavení Portu 1 .....	110
11.2.1.3	Nastavení portu 2 .....	111
<b>12</b>	<b>PŘENOS INFORMACE Z ČTEČKY KARET JA-80 DO PLC FATEK FBS-40MCR2-D24 POMOCÍ PŘEVODNÍKU WIE485 .....</b>	<b>113</b>
12.1	PŘEVODNÍK WIE485 A JEHO CHARAKTERISTIKA .....	113
12.1.1	Holding Register .....	114
12.1.2	Input Register .....	114
12.2	ČTEČKA RFID KARET JA-80 .....	117
<b>13</b>	<b>VYUŽITÍ SYSTÉMU ACCESS_ENERGO A JEHO VÝZNAM V ENERGETICKÉM PRŮMYSLU .....</b>	<b>119</b>

13.1	ALGORITMUS PRO POVOLENÍ MANIPULACÍ NA ENERGETICKÉM ZAŘÍZENÍ .....	120
<b>14</b>	<b>MODELOVÝ SYSTÉM ACCESS_ENERGO .....</b>	<b>124</b>
	<b>ZÁVĚR .....</b>	<b>129</b>
	<b>ZÁVĚR V ANGLIČTINĚ .....</b>	<b>130</b>
	<b>SEZNAM POUŽITÉ LITERATURY .....</b>	<b>131</b>
	<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK .....</b>	<b>133</b>
	<b>SEZNAM OBRÁZKŮ .....</b>	<b>135</b>
	<b>SEZNAM TABULEK .....</b>	<b>138</b>
	<b>SEZNAM PŘÍLOH .....</b>	<b>140</b>

## ÚVOD

Komunikační protokol MODBUS je velmi rozšířený protokol určený pro komunikaci různých zařízení, což umožňuje maximální automatizaci a zjednodušení procesů. V energetickém průmyslu je tento protokol velmi rozšířený pro vzájemnou komunikaci zařízení, které slouží pro výrobu a distribuci elektrické energie. Také integrace bezpečnostních systémů prožívá velký rozmach a na trhu se objevují stále novější a modernější produkty. Tyto systémy zvyšují nejen samotnou ochranu objektů a zařízení, ale také umožňují provádět různé regulační a řídicí procesy, např. ovládat osvětlení nebo kontrolovat činnosti vzduchotechniky v závislosti na pohybu osob v daných objektech. Tyto regulační procesy byly dříve výhradně doménou systémů automatického řízení procesů, avšak dnes, kdy zaznamenáváme rychlý rozvoj technologií integrovaných bezpečnostních systémů, je tento trend poměrně rychle vyrovnáván. Největším nedostatkem těchto systémů se jeví různorodost komunikačních protokolů, což je častou příčinou nemožnosti provádět zamýšlenou integraci z důvodu jejich nestandardizace a nekompatibility. Nejznámějším protokolem určený ke komunikaci bezpečnostních prvků je protokol Wiegand, který je považován za nejrozšířenější komunikační protokol pro přenos informace z čtečky RFID karet. Tento protokol však není rozšířen v jiných odvětví průmyslu, proto prvky bezpečnostního průmyslu (např. čtečky karet RDIF) je nutné přizpůsobit z hlediska komunikace a komunikačního protokolu tak, aby si tyto prvky rozuměly s ostatními průmyslovými zařízeními. Na trhu se objevují převodníky protokolů Wiegand/MODBUS, které daný problém odstraňují a usnadňují tak implementaci prvků objektové bezpečnosti do oblasti automatizace průmyslu.

Cílem této práce je návrh, realizace a implementace čteček RFID karet s řídicími systémy technologických procesů energetických zařízení. Tento návrh propojení bezpečnostních systémů do technologických procesů umožní obsluze a dispečerům lépe provozovat zařízení distribuční sítě a zvýšit bezpečnost práce na energetickém zařízení.

## **I. TEORETICKÁ ČÁST**

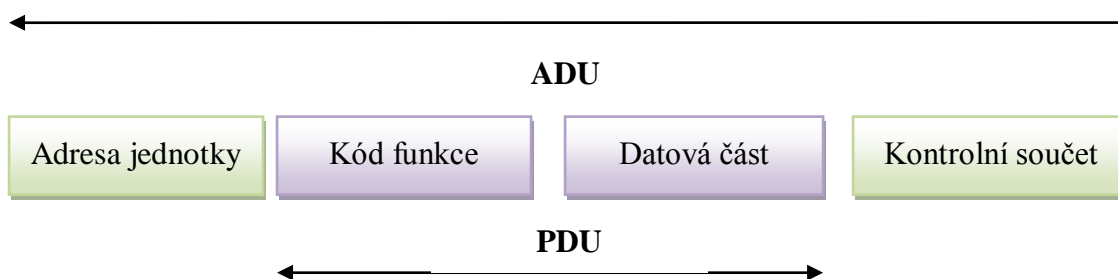
## 1 STANDARDNÍ PROTOKOL MODBUS

Standardní protokol MODBUS je univerzální komunikační protokol, který byl vyvinut především pro komunikaci na sériovém rozhraní RS232 již v 70. letech minulého století. K velkému rozšíření a standardizaci tohoto protokolu se dostalo v 80. a 90. letech s rozvojem výpočetní techniky a především s příchodem univerzálních PLC zařízení, které zjednodušily a zrychlily automatizaci všech odvětví průmyslu. Do průmyslu komerční bezpečnosti však proniká velmi pomalu, přestože umožňuje rozsáhlé, univerzální a levné implementace bezpečnostních systémů do integrovaných systémů v PKB.

Protokol MODBUS je rozdělen do 5ti základních typů, které používají společné služby a funkce. Rozdělení spočívá v jiném přenosovém rámci s různými způsoby adresování jednotlivých zařízení:

- MODBUS RTU - sériový binární protokol typu Master/Slave
- MODBUS ASCII - sériový ASCII protokol typu Master/Slave
- MODBUS TCP/IP - klasický Ethernet TCP/IP s rychlostí 10/100 Mbit/s.
- JBUS - omezená množina komunikačních zpráv typu MODBUS RTU
- MODBUS PLUS - deterministický token LAN, peer to peer protokol, 1Mbit/s.

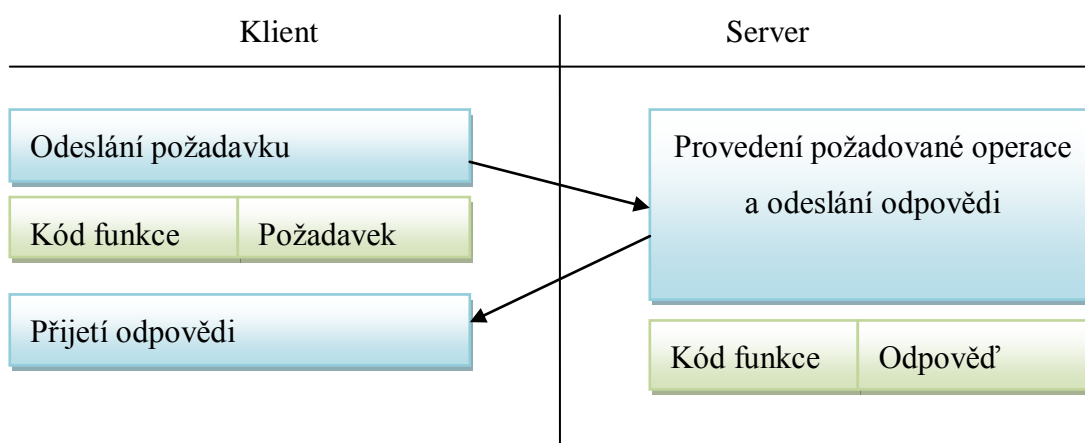
Protokol MODBUS definuje strukturu zprávy na úrovni protokolu (PDU – Protocol Data Unit) nezávisle na typu komunikační vrstvy. V závislosti na typu sítě, na které je protokol použit, je PDU rozšířena o další části a tvoří tak zprávu na aplikační úrovni (ADU – Application Data Unit). [1]



*Obr. 1 Základní tvar MODBUS zprávy*

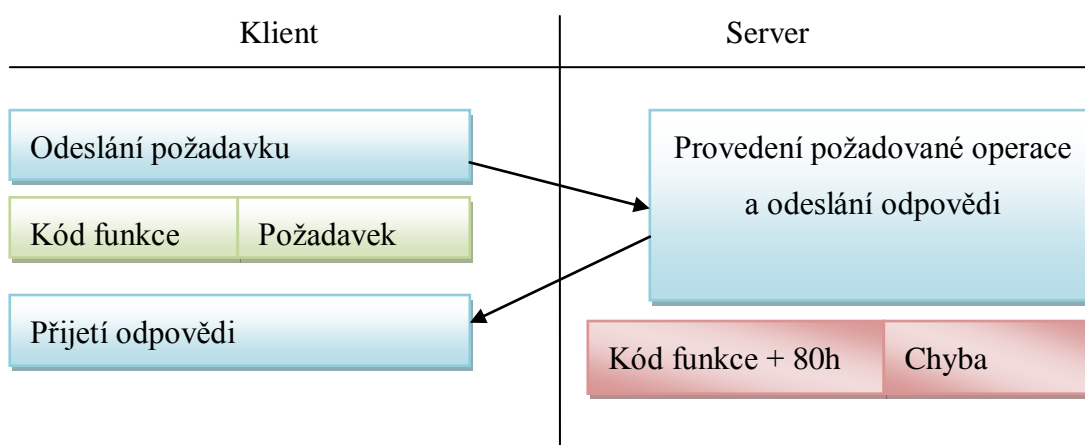
Kód funkce udává serveru jaký druh operace má provést. Rozsah kódů je 1 až 255, přičemž kódy 128 až 255 jsou vyhrazeny pro oznámení záporné odpovědi (chyby). Některé kódy funkcí obsahují i kód podfunkce upřesňující blíže požadovanou operaci. Obsah

datové části zprávy poslané klientem slouží serveru k uskutečnění operace určené kódem funkce. Obsahem může být například adresa a počet vstupů, které má server přechít nebo hodnota registrů, které má server zapsat. U některých funkcí nejsou pro provedení operace zapotřebí další data. V tomto případě může datová část ve zprávě úplně chybět. Pokud při provádění požadované operace nedojde k chybě (Obr. 2), odpoví server zprávou, která v poli Kód funkce obsahuje kód provedené (požadované) funkce jako indikaci úspěšného vykonání požadavku. V datové části odpovědi předá server klientovi požadovaná data. [1]



Obr. 2 Bezchybná transakce

Pokud při vykonávání požadované operace dojde k chybě (Obr. 3), je v poli kód funkce vrácen kód požadované funkce s nastaveným nejvyšším bitem indikujícím neúspěch (exception response). V datové části je vrácen chybový kód (exception code) upřesňující důvod neúspěchu. [1]



Obr. 3 Chybová transakce

Maximální velikost PDU je zděděna z první implementace protokolu MODBUS na sériové lince RS-485, kde byla maximální velikost ADU 256 bytů. Tomu odpovídá maximální velikost PDU 253 bytů. [1]

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

Z toho vyplývá:

Velikost ADU na RS-485 = 253 bytů PDU + adresa (1 byte) + CRC (2 byty) = 256 bytů

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

## 1.1 Kódování dat a datový model

MODBUS používá tzv. „Big-endian“ reprezentaci dat. To znamená, že při posílání datových položek delších než 1 byte je jako první poslán nejvyšší byte a jako poslední nejnižší byte.

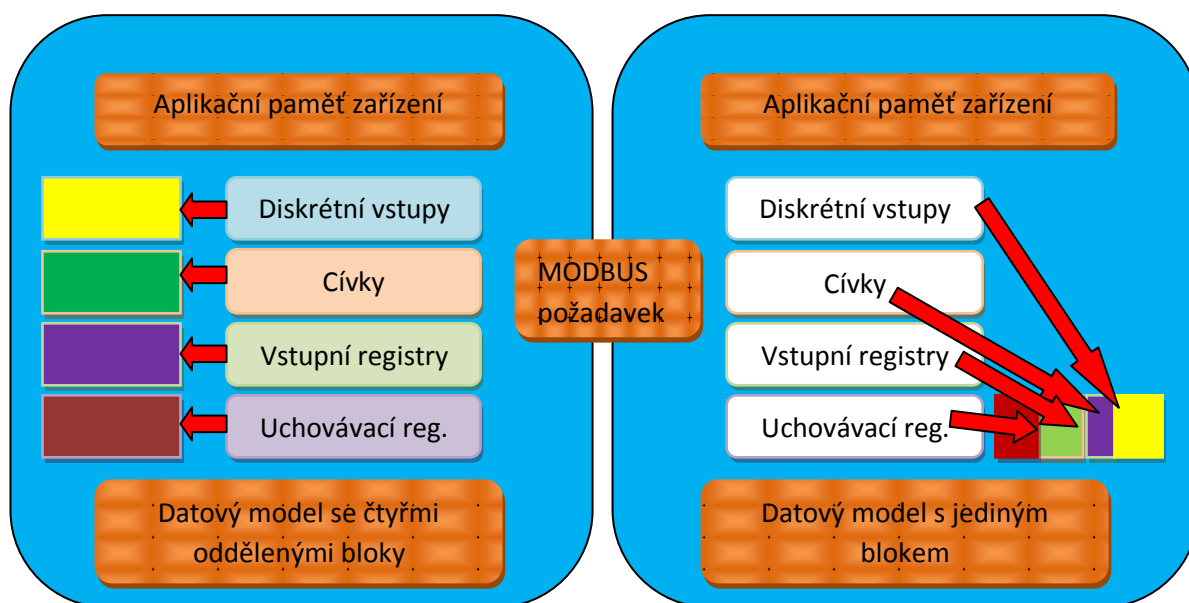
Například u 16bitové položky s hodnotou 1234h je nejprve poslán byte 12h, poté byte 34h.

Datový model MODBUS je založen na sadě tabulek, s charakteristickým významem. Definovány jsou čtyři základní tabulky:

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

Tab. 1 Datový model MODBUS

Mapování tabulek do adresovatelného prostoru je závislé na konkrétním zařízení. Každá z tabulek může mít vlastní adresní prostor nebo se mohou částečně či úplně překrývat. Každá z tabulek může mít dle protokolu až 65536 položek. Z důvodu zpětné kompatibility bývá adresní prostor rozdělen na bloky o velikosti 10000 položek, tak jak je uvedeno ve sloupci „Adresa“ tabulky 2.1. Přístupná je každá položka jednotlivě nebo lze přistupovat ke skupině položek najednou. Velikost skupiny položek je omezena maximální velikostí datové části zprávy. [1]



Obr. 4 Datový model MODBUS s oddělenými a neoddělenými bloky

Na obrázku 4 jsou znázorněny dva možné způsoby organizace dat v zařízení. Náčrty v levé části obrázku znázorňují zařízení, kde není žádný vztah mezi položkami jednotlivých tabulek a každá tabulka má tedy svůj oddělený prostor v aplikační paměti zařízení. Do jednotlivých tabulek lze přistupovat prostřednictvím příslušné funkce MODBUS. V pravé části obrázku je znázorněno zařízení, které má pouze jeden datový blok. K položkám lze přistupovat prostřednictvím různých funkcí MODBUS v závislosti na tom, co je pro aplikaci v daném okamžiku výhodné. [1]

## 1.2 Definice MODBUS transakce

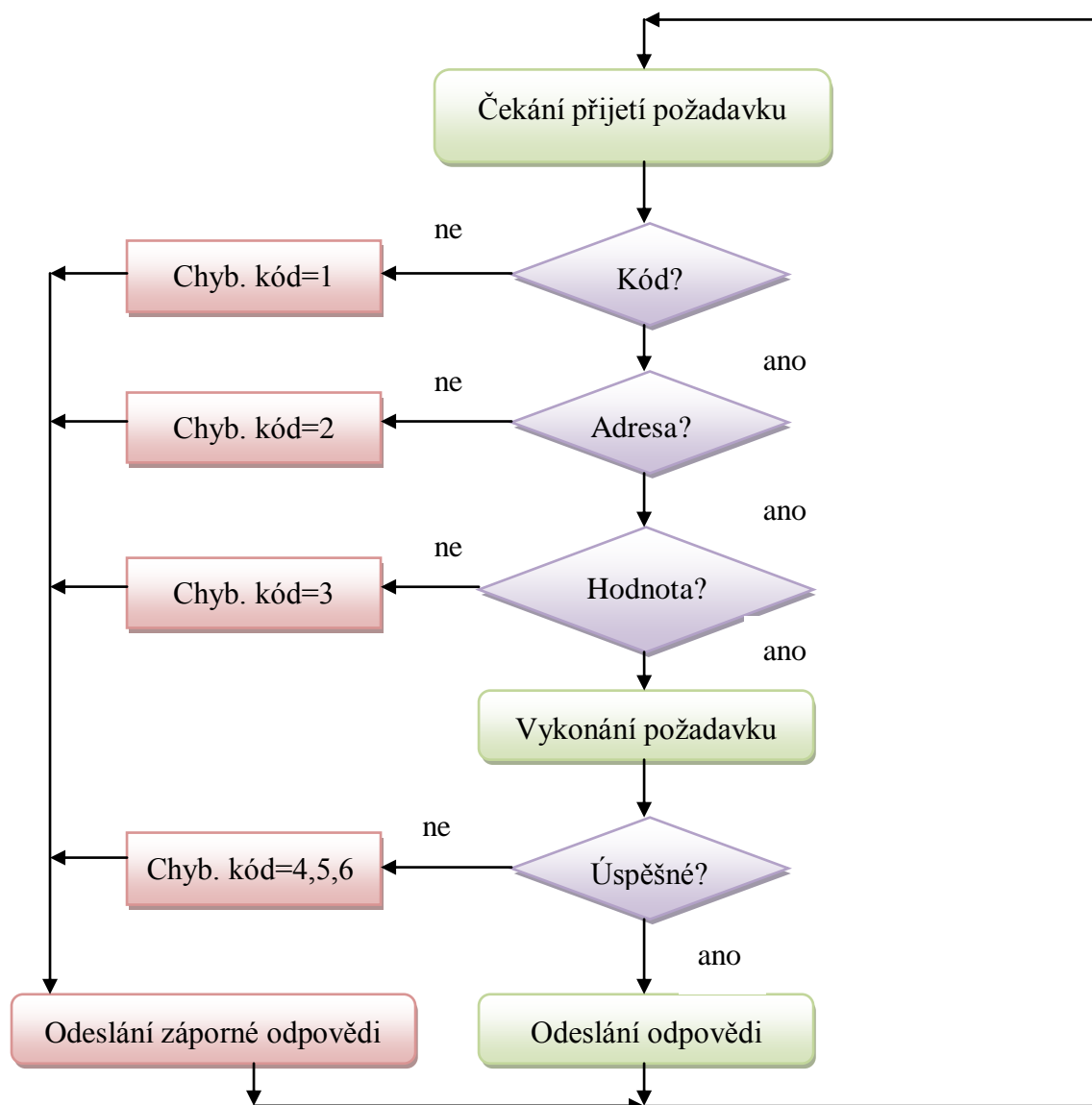
**Dotaz (Request PDU)** – Funkční kód v žádosti od masteru říká adresovanému slave zařízení jaký druh instrukce se má provést nebo vykonat. Data obsahují přidanou informaci o tom co má slave zařízení provést.

Například: funkční kód 03 požádá slave jednotku, aby přečetl obsah registrů. Datové pole musí obsahovat informaci říkající Slave zařízení, které registry a kolik registrů má zařízení přečíst. Chybový výpočet zajistí metodu potvrzující integritu přenášených dat.

**Odezva (Response PDU)** – Pokud slave jednotka pracuje s integritní zprávou přijatou od masteru, funkční odezvou je poté odpověď na funkční kód v dotazu poslanou zpět k master zařízení. Data obsahují všechny posbíraná data ze slave jednotky, dále hodnoty registrů a jejich status. Pokud nastane jakákoliv chyba v integritě dat, tak funkční kód MODBUS je



modifikován, aby indikoval v odpovědi chybovou zprávu a data obsahovaly chybový kód, který chybu upřesňuje. Chybový výpočet umožňuje masteru zpracovat a vyhodnotit chybovou hlášku. [2]



Obr. 5 Obecný postup zpracování MODBUS požadavku na straně serveru

**Záporná odpověď** (Exception Response) – Jestliže se objeví záporná odpověď, tak její identifikace je složena s 1 bajtu, který vytváří kód funkce + 80 hex přiznávající identifikaci neúspěchu a dalšího bajtu, který nazýváme chybový kód, kde se vypíše identifikace chyby. Stavový diagram na obrázku 5 popisuje obecný postup zpracování MODBUS požadavku na straně serveru. Jakmile server zpracuje požadavek (ať úspěšně či neúspěšně), sestaví odpověď a odešle ji klientovi. [2]

V závislosti na výsledku zpracování požadavku je vytvořena jedna ze dvou možných odpovědí:

- Pozitivní odpověď (Response):  
kód funkce v odpovědi = kód funkce v požadavku
- Negativní odpověď (Exception Response)  
kód funkce v odpovědi = kód funkce v požadavku  
+ 80h je vrácen kód chyby udávající důvod neúspěchu

### 1.3 Transakce v MODBUS sítích

Standardní MODBUS port využívá RS-232C kompatibilní sériové rozhraní, které definuje konektory, kabeláž, přenosovou rychlost, paritu apod. Ovladače mohou být definovány přímo prostřednictvím daného zařízení. Komunikace je založena na technologii Master-Slave, ve kterém pouze jedno zařízení se chová jako master (klient) a může inicializovat transakce tzv. dotaz (queries). Další zařízení (slave) odesílají odezvu na master jednotku nebo provádějí požadované procesy odeslané master zařízením.

Master může zasílat instrukce na jednotlivou slave jednotku nebo může inicializovat broadcastovou zprávu do všech slave zařízení. Každá slave jednotka individuálně vrací zprávu (odpověď-response). Odpověď master zařízení na slave dotaz však není opětovně vráceno broadcastově, ale pouze tomu slave zařízení, které dotaz zaslalo. MODBUS je založen na systému dotaz-odpověď pouze v prostoru jeho funkčního adresovaného okolí, jeho funkce definuje odezvu na žádost zařízení komunikujících pod tímto protokolem, zpracující data a provádějící kontrolní chybový výpočet. Slave odezva je také podřízena tomuto protokolu. Pokud slave zařízení přijme chybovou instrukci nebo není schopné tuto instrukci provést, pak následně vypracuje chybovou informaci pro master zařízení a tuto informaci mu zašle jako instrukci s výpisem chybového kódu. [2]

### 1.4 Transakce na dalších typech počítačových sítí

Mimo oblast působení standardního MODBUS protokolu některé PLC mohou mezi sebou komunikovat pomocí protokolu MODBUS Plus užívající tzv. built-in porty přes MAP, které jsou vybaveny standardním síťovým adaptérem.

Na těchto sítích PLC automaty mohou inicializovat komunikaci s jinými adaptéry pomocí peer-to-peer technologie, kdy je možné nastavit komunikaci s dalšími kontroléry. Takto může kontroler pracovat jednou jako master a podruhé jako slave jednotka v oddělených transakcích. Z důvodu provádění současně probíhajících procesů jsou vytvářeny plně multiplexové interní cesty. V komunikaci typu peer-to-peer je v přenosové zprávě také obsažena principiálně žádost typu master-slave. Pokud kontrolér, který pracuje jako master vypracuje zprávu, tak očekává odezvu od slave zařízení. Podobně je tomu naopak, pokud slave přijme informaci od master zařízení vypracuje odpověď a tu odešle jako odezvu zpět k masteru. [2]

## 2 PŘENOSOVÉ MÓDY MODBUSU POUŽÍVANÉ NA SÉRIOVÉ LINCE

Kontroléry mohou používat pro komunikaci na sériové lince dva přenosové módy – ASCII nebo RTU. Uživatel se může libovolně rozhodnout, který mód použít. Záleží na rozhodnutí podle kritérií, jako je přenosová rychlost, počet stop bitů, parity apod. Metoda módu však musí být shodná pro celou komunikační síť používající protokol MODBUS. Výběr se ovšem vztahuje ke standardním MODBUS sítím, které definují obsah bitové integrity přenášené na této síti. Tato bitová integrita stanovuje, jak jsou jednotlivé bity zabaleny do rámců a jak jsou následně dekodovány.

Na dalších sítích jako je MAP nebo MODBUS Plus jsou zprávy zabaleny do rámců, které nejsou schopné být přenášeny po sériové lince standardním MODBUS protokolem. Příkladem může být čtení registrů, které se vztahuje ke dvěma kontrolérům pracující na MODBUS Plus. MODBUS plus posílá svá data najednou, nikoliv však bit po bitu jako to je u standardního modbusu.[2]

### 2.1 ASCII a RTU Mód

Pokud jsou kontroléry nastavené, aby komunikovaly na MODBUS síti pomocí ASCII protokolu, je každý 8bitový bajt posílán jako dvojice ASCII znaků. Výhodou tohoto módu je čas přenosu mezi jednotlivými znaky, který je až do 1 sekundy, aniž by byla vyhlášena chyba přenosu dat.

#### Kódovací systém ASCII:

- Hexadecimální, ASCII znaky 0-9, A-F
- Jeden hexadecimální znak obsahuje v každé zprávě danou zprávu

#### Popis bitové zprávy v bajtu

- 1 start bit
- 7 bitová zpráva, první přichází nejnižší bit
- 1 bit pro příznak bitové parity
- 1 bit, pokud je parita použita
- 2 bity, pokud je přenos bez kontroly parity

#### Kontrolní součet

- Longitudinal Redundancy Check (LRC)

Pokud je kontrolér nastavený do RTU módu, tak každý 8mý bit v bajtu obsahuje dva 4bitové hexadecimální znaky. Výhodou tohoto módu je větší znaková hustota, která umožňuje větší přenos informace při stejné rychlosti jakou má ASCII mód. Každá zpráva musí být zaslána jako posloupný tok bitů. [2]

#### **Kódovací systém RTU:**

- 8bitový binární, hexadecimálně 0-9, A-F
- 2 hexadecimální znaky obsahující v každé zprávě 8bitové pole znaků

#### **Popis bitové zprávy v bajtu**

- 1 start bit
- 8 bitová zpráva, první přichází nejnižší bit
- 1 bit pro příznak bitové parity
- 1 bit, pokud je parita použita
- 2 bity, pokud je přenos bez kontroly parity

#### **Kontrolní součet**

- Cyclical Redundancy Check (CRC)

## **2.2 Rámec se zprávou na MODBUSu**

V obou MODBUS módech jsou vytvářeny rámce se zprávou, které znají počátečný a koncový bod zprávy. To umožňuje zařízení, aby okamžitě po obdržení zprávy byl definovaný adresní prostor a z něj se určilo, které zařízení je již adresováno. Tento způsob definování adresného prostoru se umožňuje dozvědět, zdali byla zpráva kompletní nebo nikoliv. Pokud je detekována chybná zpráva, je možné podle provedeného chybového výpočtu tuto chybu identifikovat. [2]

### **2.2.1 ASCII rámec**

V ASCII módu je zpráva započata dvojtečkou - ( : ). U ASCII kódu jde o 3A (hex) a ukončena tzv. návratem netisknutelného znaku (carriage return – line feed) (CRLF), což jsou znaky #13 a #10, ze kterých se nepočítají kontrolní součty. V ASCII modu se jedná o 0D a 0A hexadecimálně.

Přípustné znaky pro všechny ostatní části zprávy jsou v hexadecimálním kódu 0-9, A-F. Zpráva vždy začíná dvojtečkou a zařízení sleduje její objevení ve zprávě. Pokud je dvojtečka detekována, tak systém dekóduje nové pole dalších dvou přijatých znaků jako adresní prostor a zjistí, zdali je zařízení adresováno. Intervaly nad jednu vteřinu mezi znaky mohou být uvnitř zprávy ztraceny. [2]

Začátek	Adresa	Funkce	Data	LRC	Konec
znak ":"	2 znaky	2 znaky	0 až 2*252 znaků	2 znaky	2 znaky CR, LF

Tab. 2 ASCII rámeček zprávy

### 2.2.2 RTU rámeček

V RTU módu je začátek a konec zprávy modifikován pomlčkou na sběrnici, která je delší než 3,5 znaku. Jedná se o nejjednodušší implementaci multiplexního přenosu dat užívané na síti. První pole 8 bitů určují adresu zařízení. Všechny přenášené znaky jsou v hexadecimální podobě. Vysílání zprávy musí být souvislé a to včetně pomlky. Po přijetí prvního pole znaků je každé zařízení dekódováním pole identifikováno. Pokud není v adresaci při dekódování zařízení rozpoznáno, tak nastane chyba, která je následně vyhodnocena. Závěrečných 3,5 znaku je vyhodnoceno jako konec zprávy a za počátek nové zprávy je považováno ukončení vysílací sekvence 3,5 znaků.

Celá zpráva musí být vysílána jako kontinuální sekvence znaků. Jestliže dojde k pomlce, která je delší než 1,5 znaků před komplementací rámce, tak přijímací strana zbylé data zahodí a čeká na další rámeček, který je uzavřen kontrolním součtem. Podobně, pokud je započato vysílání zprávy před dokončením vysílání závěrečných 3,5 znaků z předchozí zprávy, tak jsou tyto data doručeny, ale kontrolní součet je vyhodnotí jako chybový. Každá jednotka musí podporovat sudou paritu. Pokud není použita parita, tak je tato nahrazena druhým stop bitem. [2]

Začátek	Adresa	Funkce	Data	CRC	Konec
>3,5 znaků	8 bitů	8 bitů	N*8 bitů	16 bitů	>3,5 znaků

Tab. 3 RTU rámeček zprávy

### 2.2.3 Získání adresného prostoru a funkčního pole

Adresní prostor rámce zprávy obsahuje dvojice znaků v ASCII módu nebo 8bitový bajt zprávy. Hodnota slave zařízení je v prostoru 0-247 (decimálně). Individuální adresa slave

jednotky je 1-247. Master jednotka adresuje slave jednotky podle odkazu adresného prostoru v zasláné zprávě. Když slave zařízení odpovídá na dotaz masteru, tak slave jednotka uvede svojí vlastní adresu v adresném poli. Tím Master jednotka pozná, které slave zařízení posílá odezvu na dotaz. Adresa 0 je užívána pro broadcastové adresy. Pokud je používána broadcastová adresa, tak master jednotka může rozpoznat všechny slave jednotky v síti. Pokud je MODBUS protokol užíván na vyšších úrovních modelu ISO OSI, metoda broadcastového volání nemusí být plně funkční. V tomto případě musí být nahrazena jinými metodami. Například MODBUS Plus užívá sdílenou globální databázi, která může být obnovována v každém dotazu při rotaci daného tokenu. [2]

Platnost kódu je dimenzována v prostoru 1-255 decimálně. Ve zprávě masteru určené slave jednotce funkční kód říká jakou instrukci má slave jednotka provést. Příkladem může být čtení ON/OFF stavů diskrétních cívek nebo vstupů, čtení obsahu dat ve skupině registrů, diagnostikování statusu slave jednotky, zápis do označených cívek nebo registrů, povolení načítání, nahrávání a verifikaci uvnitř slave jednotky.

Když slave jednotka odpovídá masteru, tak užívá funkční kód pole, aby indikoval buď normální odpověď, nebo zprávu s chybovým hlášením. V bezchybné zprávě, slave jednotka navrátí původní zprávu funkčního kódu. V záporné odpovědi navrací kód, který je ekvivalentní původní zprávě s přidanou logickou 1 na počátku zprávy.

Příklad čtení zprávy z Master jednotky do slave zařízení : 0000 0011 (hexadecimálně 03)

Pokud slave jednotka přijme zprávu bezchybně, tak navrací tu samou zprávu. Pokud se však vyskytne chyba, je navracená tato hodnota: 1000 0011 (hexadecimálně 83)

Mimo tuto modifikaci ještě slave jednotka umísťuje unikátní kód do datového pole odezvy. Tento kód řekne master jednotce o jakou případnou chybu se jednalo.

Master jednotka musí být vybavena funkčním kódem, který tyto chybové hlášky získává a zpracovává. Typickým procesem je zaslání charakteristické zprávy slave jednotce a informování operátora nebo obsluhu zařízení.

Datové pole je určeno užitím dvou hexadecimálních čísel v adresném prostoru 00 až FF. Tyto mohou být ve formátu dvojice ASCII znaků nebo jednoho RTU znaku podle sériového přenosového módu. Datové pole z masteru do slave zařízení obsahuje přidanou informaci s instrukcí co má slave jednotka provést. Tyto informace jsou určeny pro adresy

registrů nebo diskrétních cívek. Velikost položek, se kterými má být manipulováno a počet aktuálních bitů v poli je volitelné podle požadovaného kódu.

Například, jestliže master žádá slave jednotku, aby přečetl skupinu registrů (funkční kód 03), tak pole dat specifikuje počáteční registr a počet registrů, které mají být přečteny. V odpovědi je přiřazena každému registru dvojice bajtů. Pokud Master jednotka zapíše do skupiny registrů funkční kód 10 (hexadecimálně), tak datové pole specifikuje startovní registr, kolik registrů zapsat, počet bajtů a data zapsané do registrů. Pokud nenastane žádná chyba, tak odezva ze slave zařízení do master jednotky je totožná jako volaná žádost. Pokud je generována nějaká chybová hláška, tak master rozhodne, jaké se provedou následující instrukce nebo procesy.

Data mohou být obsaženy ve zprávě také jako pole dat nulové délky. Například v žádosti z masteru zařízení pro slave jednotku, aby odpověděla událostí log (funkční kód 0B hexadecimálně), slave zařízení nežádá žádnou přidanou informaci. Funkční kód sám specifikuje akci, která se má provést, neboli vrátí data požadavku. [2]



### 3 OBSAH CHYBOVÉHO SOUČTU A JEHO METODY

Standardní protokol MODBUS používá dvě metody kontrolního součtu. Tyto součty jsou závislé na používané metodě. Tyto metody jsou podrobně vysvětleny ve třetí kapitole této diplomové práce. [2]

Zpráva standardního MODBUS protokolu je odeslána v tomto pořadí:

Least Significant Bit (LSB) + Most Significant Bit (MSB)

#### 3.1 ASCII

Pokud je používáno ASCII kódování, kontrolní součet obsahuje ASCII znaky. Tyto dva znaky jsou dány výsledkem kontrolního výpočtu (Longitudinal Redundancy Check – LRC), která představuje ve zprávě obsah od prvotní dvojtečky až po ukončovací znaky #13. LRC znaky jsou přidány do zprávy do posledního pole předchozí zprávy.



Tab. 4 ASCII pole pořadí bitů

#### 3.2 RTU

V RTU módu je kontrolní součet obsažen v 16bitové zprávě představované dvěma 8bitovými bajty. Kontrolní součet je dán výsledkem algoritmu Cyclical Redundancy Check výpočtu (CRC). Kontrolní součet je přidán na poslední místo v přenášené zprávě. Když je součet dokončen, tak jako první je do posledního pole zprávy umístěn tzv. horní bajt a za ním následuje přidáný tzv. dolní bajt. CRC horní bajt je posledním bajtem, který je v odeslané zprávě obsažen. [2]



*Tab. 5 RTU pole pořadí bitů*

### 3.3 Metody kontrolního součtu

Standardní MODBUS používá dva druhy kontrolního součtu. Kontrola parity na sudou a lichou může být aplikována pro každý znak. Kontrolní součet je obsažen v přenášené zprávě. Jak kontrolní výpočet, tak implementace znaků do kontrolní zprávy je generována v master jednotce a je zaslána do slave jednotky. Před odesláním zprávy slave jednotka kontroluje každý znak ve zprávě při přijímání zprávy.

Master jednotka je nastavena uživatelem tak, aby byl nastaven daný timeout interval před tím, nežli bude transakce zrušena. Tento interval musí být nastaven na délku, která odpovídá bezproblémové komunikaci na straně slave jednotky s masterem. Pokud slave jednotka detekuje chybný výpočet, tak instrukce zprávy nebudou provedeny. Slave zařízení nezpracovává odpověď pro Master. Naprogramovaná master jednotka podle vypršené časové odezvy stanovuje chybu události. Zpráva odeslána do neexistující slave jednotky bude detekována také jako timeout než vyprší časová odezva na zaslanou zprávu. [2]

### 3.4 Kontrola parity

Uživatel se může rozhodnout, zdali používat kontrolu na sudou, lichou nebo případně žádnou paritu. Tímto je určeno, jaká parita je nastavena v každém znaku.

Pokud je sudá nebo lichá parita specifikována, velikost výsledného bitu je spočítána po částech v každém znaku (7 datových bitů u ASCII nebo 8 datových bitů u RTU). Paritní bit poté bude nastavena na 0 nebo 1 dle žádané a nastavené parity.

Například pro 8 datových bitů obsažené v RTU zprávě 1100 0101 (binárně) je celková hodnota jednoho bajtu ve zprávě rovna 4. Pokud je užívána sudá parita, tak parita zprávy bude 0 a celkový součet zůstává opět roven 4. Pokud je použita lichá parita, tak paritní bit musí být nastaven na 1 a celková hodnota bitů ve zprávě je 5.

Když je zpráva odeslána, tak paritní bit je vypočítán a aplikován do zasílané zprávy. Příjemci jednotka tuto zprávu dekóduje, spočítá velikost prvního bajtu, a pokud dojde ve výpočtu k nerovnosti v počtu přijatých bitů s odeslanými bity, tak vypracuje chybovou hlášku. Sudá nebo lichá parita může být detekována pouze v případě samotného přenosu dat. Pokud je nastavena lichá parita, tak může být detekována pouze ztráta lichého počtu bitů ve zprávě. Pokud není zvolena žádná parita, tak při přenosu dat není žádný kontrolní součet parity proveden. [2]

### 3.5 LRC Checking

V ASCII módu obsahuje zpráva chybovou hlášku založenou na LRC algoritmu. LRC metoda sleduje obsah zprávy, který začíná dvojtečkou a končí dvojicí CRLF znaků. Tato metoda je aplikována bez ohledu na výběr parity znaků. LRC je jednobajtové pole obsahující 8bitovou binární hodnotu. LRC je vypočítána z obsahu bitů v přenosové cestě a tento LRC výsledek je přidán do posílané zprávy. Příjemci zařízení vypočítává LRC během přenosu a srovnává vypočtenou hodnotu s aktuální přenášenou hodnotou RLC. Pokud obě hodnoty nejsou totožné, tak je vyhlášena chyba přenášených dat. LRC má za cíl kontrolu přenosu společných jdoucích po sobě 8bitových bajtů zprávy, zahození vadných dat, a nakonec zkompletování výsledku přenosu. CKSM funkce vypočítává LRC z kontextu přenášené zprávy. [2]

### 3.6 CRC Checking

V RTU metodě zpráva zahrnuje CRC algoritmus pro výpočet chybného přenosu dat. CRC pole kontroluje obsah celé zprávy, která je aplikována bez ohledu na nastavení parity požadovaného přenosu. CRC je 2 bajtové pole, obsahující 16bitovou hodnotu. CRC hodnota je vypočítána z přenášených dat, a tento CRC výsledek je přidán do přenášené zprávy. Příjemci strana tento výpočet porovnává a vyhodnocuje případný chybný přenos.

CRC kód je započat před úvodním 16bitovým registrem, kde jsou všechny bity nastaveny na log1. Poté je proces aplikován následujícím 8bitovým bajtem do obsahu registru. Pouze každý 8 bit je použit pro výpočet CRC kódu. Start a stop bit, pokud je použit, není zahrnut do CRC algoritmu.

Během generování CRC kódu je každý 8bitový znak v obsahu registru v logickém obvodu vytvářen tzv. Exclusive OR logikou. Poté je výsledek přesunut do nejméně významného bitu s nulou vloženou do nejvyššího bitu. LSB je získána s přenosu data a posléze

přepočítána. Jestliže LSB byl log1, tak registr je exclusive OR s přednastavenou stálou hodnotou. Jestliže LSB byl 0, tak k žádnému výsledku v exclusive OR nedochází. Tento proces je opakován, dokud není přesunuto v přenosu všech 8 znaků. Po přenesení posledního 8 znaku je celý proces výpočtu opět zopakován. Celkový součet registrů všech přenesených dat vytváří CRC součet.

### 3.6.1 Užití Součtového bajtu

Pokud je skládána odezva v bufferu, je nutné užít bitový součet, který je roven součtu 8bitového bajtu ve zprávě. Hodnota je vyhrazená pro všechny ostatní obsahy polí zahrnující bitový součet.

## 3.7 Data adresované v MODBUS zprávě

Všechna data adresované v MODBUS komunikaci jsou prezentovány od 0. Tedy první odkazová hodnota je adresována do adresy 0.

- Adresa první cívky klientské aplikace je adresována pod indexem 0000 v poli MODBUS zprávy.
- Cívka 127 decimálně je adresována jako 007E (126 decimálně)
- Uživatelské registr 40001 je adresován jako registr 0000 v adresovacím poli zprávy. Tento funkční kód pole je již specifikován uživatelském registrem operace. Proto je 4XXXXX reference brána jako implicitní
- Uživatelské registr 40108 je adresován jako registr 006B hex (107 decimálně).

Dotaz Master jednotky se nazývá v tomto případě Read Holding Registers (Čti uživatelské registry), aby přečetl dotaz slave jednotky na adrese 06. Zpráva žádá o data ze tří uživatelských registrů 40108 až 40110. Zpráva je adresována startujícím registrem adresované jako 0107 (006B hex). Slave jednotka odpovídá ve funkčním kódu. Součet hodnoty bajtů specifikuje, kolik 8bitových položek je právě navraceno. Tím je možné se dozvědět celkový součet 8bitových bajtů, které sledujeme. V ASCII kódování je tato hodnota polovina aktuálního součtu ASCII znaků. V ASCII každá 4bitová hexadecimální hodnota žádá jeden ASCII znak o příznak své hodnoty.

Například, hodnota 63 hex je odeslán jako jeden osmibitový bajt v RTU módu (01100011). Ta samá hodnota odeslána v ASCII módu vyžaduje dva bajty. Jeden pro ASCII

„6“ (0110110) a druhá pro „3“ (0110011). Součtový bajt počítá s těmito daty jako s jednou 8bitovou položkou, bez ohledu na charakteru používané metody (RTU nebo ASCII). [2]

### 3.8 Záporné odpovědi

Jestliže klient posílá serveru požadavek, očekává od něj odpověď. Můžou nastat tři situace:

- Jestliže server požadavek nepřijme z důvodu komunikační chyby, není vrácena žádná odpověď. Na straně klienta dojde k vypršení časového limitu pro příjem odpovědi.
- Jestliže server přijme požadavek, ale detekuje komunikační chybu (parita, CRC), tak nevrací žádnou odpověď. Na straně klienta dojde k vypršení časového limitu pro příjem odpovědi.
- Jestliže server přijme bezchybně požadavek, ale není schopen jej normálně zpracovat, vrátí klientovi zápornou odpověď s udáním důvodu neúspěchu.

Normální a záporná odpověď se liší nejvyšším bitem kódu funkce. Je-li bit nulový, jedná se o normální odpověď, je-li bit nastavený na nenulovou hodnotu, jedná se o zápornou odpověď. V případě záporné odpovědi je v datové části předán kód chyby. V následující tabulce je seznam možných chybových kódů:

Modbus chybové kódy		
kód	jméno	Význam
01	Ilegální funkce	Požadovaná funkce není serverem podporována
02	Ilegální adresa dat	Zadaná adresa je mimo podporovaný rozsah
03	Ilegální hodnota dat	Předávaná data jsou neplatná
04	Selhání zařízení	Při provádění požadavku došlo k chybě
05	Potvrzení	Server hlásí přijetí platného požadavku. Vykonání však bude trvat delší dobu
06	Zařízení je zaneprázdněné	Hlásí zaneprázdnění serveru vzhledem k dlouhotrvajícímu příkazu
08	Chyba parity paměti	Hlášení chyby parity při čtení souboru
0A	Brána-přenosová cesta nedostupná	Hlásí pravděpodobnost chybného nastavení brány (gateway)
0B	Brána-cílové zařízení neodpovídá	Cílové zařízení neodpovídá nebo není přítomné

Tab. 6 Chybové kódy v protokolu MODBUS

## 4 POPIS FUNKČNÍCH KÓDŮ

Podporované kódy:

V přehledové tabulce jsou znázorněny nejvíce používané kódy pro komunikaci mezi periferiemi komunikující prostřednictvím protokolu MODBUS na sériové lince. [1]

Instrukce				kódy funkcí			
				kód	podfunkce	hex	
Přístup k datům	Bitový přístup	Fyzické diskrétní vstupy	Čti diskrétní vstupy	02		02	
			Čti cívky	01		01	
		Interní bity nebo fyzické cívky	Zapiš jednu cívku	05		05	
			Zapiš více cívek	15		0F	
	16- bitový přístup	Fyzické vstupní registry	Čti vstupní registr	04		04	
		Interní registry nebo fyzické výstupní registry	Čti uchovávací registry	03		03	
			Zapiš jeden registr	06		06	
			Zapiš více registrů	16		10	
			Čti zapiš 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	06	14	
	Zápiš záznam do souboru		21	06	15		
	Diagnostika			Čti stav	07		07
				Diagnostika	08	00-18,20	08
Čti čítač kom. Událostí				11		0B	
Čti záznam kom. Událostí				12		0C	
Sděl identifikaci				17		11	
Čti identifikaci zařízení				43	14	2B	

Tab. 7 Definice funkčních kódů

### 4.1 01 (0x01) Čti cívky (Read Coils)

Tato funkce slouží ke čtení stavu 1 až 2000 cívek. V požadavku je specifikována adresa první cívky a počet cívek. V odpovědi je v jednom bytu přenášen stav celkem 8 cívek. Nejnižší bit prvního bytu je stav první (adresované) cívky. [1]

Požadavek		
Kód funkce	1 byte	0x01
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Počet cívek	2 byty	1 až 2000 (0x7D0)
Odpověď		
Kód funkce	1 byte	0x01
Počáteční adresa	2 byty	N
Stavy cívek	N bytů	
N = počet cívek/8, je-li zbytek po dělení nenulový, N=N+1		
Chybový kód		
Kód funkce	1 byte	0x81
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 8 Čti cívky

## 4.2 02 (0x02) Čti diskretní vstupy (Read Discrete Inputs)

Tato funkce slouží ke čtení stavu 1 až 2000 diskretních vstupů. V požadavku je specifikována adresa prvního vstupu a počet vstupů. V odpovědi je v jednom bytu přenášen stav celkem 8 vstupů. Nejnižší bit prvního bytu je stav prvního (adresovaného) vstupu. [1]

Požadavek		
Kód funkce	1 byte	0x02
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Počet cívek	2 byty	1 až 2000 (0x7D0)
Odpověď		
Kód funkce	1 byte	0x02
Počáteční adresa	2 byty	N
Stavy cívek	N bytů	
N = počet cívek/8, je-li zbytek po dělení nenulový, N=N+1		
Chybový kód		
Kód funkce	1 byte	0x82
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 9 Čti diskretní vstupy

## 4.3 03 (0x03) Čti uživatelské registry (Read Holding Registers)

Tato funkce slouží ke čtení obsahu souvislého bloku až 125 uživatelských registrů. V požadavku je specifikována adresa prvního registru a počet registrů. V odpovědi odpovídá každému registru dvojice bytů. [1]

Požadavek		
Kód funkce	1 byte	0x03
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Počet cívek	2 byty	1 až 125 (0x7D)
Odpověď		
Kód funkce	1 byte	0x03
Počáteční adresa	2 byty	2*N
Stavy cívek	2*N bytů	
N = počet cívek/8, je-li zbytek po dělení nenulový, N=N+1		
Chybový kód		
Kód funkce	1 byte	0x83
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 10 Čti uživatelské registry

#### 4.4 04 (0x04) Čti systémové registry (Read Input Registers)

Tato funkce slouží ke čtení obsahu souvislého bloku až 125 vstupních registrů. V požadavku je specifikována adresa prvního registru a počet registrů. V odpovědi odpovídá každému registru dvojice bytů. [1]

Požadavek		
Kód funkce	1 byte	0x04
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Počet cívek	2 byty	1 až 125 (0x7D)
Odpověď		
Kód funkce	1 byte	0x04
Počáteční adresa	2 byty	2*N
Stavy cívek	2*N bytů	
N = počet cívek/8, je-li zbytek po dělení nenulový, N=N+1		
Chybový kód		
Kód funkce	1 byte	0x84
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 11 Čti systémové registry

#### 4.5 05 (0x05) Zapiš jednu cívku (Write Single Coil)

Tato funkce slouží k nastavení jednoho výstupu do stavu ON nebo OFF. V požadavku je specifikována adresa výstupu, který se má nastavit a hodnota, na kterou se má nastavit. 0x0000 znamená OFF, 0xFF00 znamená ON. Normální odpověď je kopií požadavku. [1]



Požadavek		
Kód funkce	1 byte	0x05
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Počet cívek	2 byty	0x0000 nebo 0xFF00
Odpověď		
Kód funkce	1 byte	0x05
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Stavy cívek	2*N bytů	0x0000 nebo 0xFF00
N = počet cívek/8, je-li zbytek po dělení nenulový, N=N+1		
Chybový kód		
Kód funkce	1 byte	0x85
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 12 Zapiš jednu cívku

#### 4.6 06 (0x06) Zapiš jeden registr (Write Single Register)

Tato funkce slouží k zápisu jednoho uživatelského registru. V požadavku je specifikována adresa registru, který se má zapsat a hodnota, která se má zapsat. Normální odpověď je kopií požadavku a je vrácena poté, co je registr zapsán. Tato funkce slouží k zápisu jednoho uživatelského registru. V požadavku je specifikována adresa registru, který se má zapsat a hodnota, která se má zapsat. Normální odpověď je kopií požadavku a je vrácena poté, co je registr zapsán. [1]

Požadavek		
Kód funkce	1 byte	0x06
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Počet cívek	2 byty	0x0000 až 0xFFFF
Odpověď		
Kód funkce	1 byte	0x06
Počáteční adresa	2 byty	0x0000 až 0xFFFF
Stavy cívek	2*N bytů	0x0000 až 0xFFFF
N = počet cívek/8, je-li zbytek po dělení nenulový, N=N+1		
Chybový kód		
Kód funkce	1 byte	0x86
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 13 Zapiš jeden registr

#### 4.7 07 (0x07) Čti stav (Read Exception Status) – pouze pro sériovou linku

Tato funkce slouží ke čtení stavu osmi stavových výstupů. Normální odpověď obsahuje stav těchto výstupů, přenášený v jednom bytu. [1]

Požadavek		
Kód funkce	1 byte	0x07
Odpověď		
Kód funkce	1 byte	0x07
Stav výstupů	1 byte	0x0000 až 0xFFFF
Chybový kód		
Kód funkce	1 byte	0x87
Chybový kód	1 byte	01 nebo 04

Tab. 14 Čti stav

#### 4.8 08 (0x08) Diagnostika (Diagnostics) – pouze pro sériovou linku

Tato funkce slouží k provedení série testů pro zkontrolování komunikace mezi klientem (Master) a serverem (Slave) nebo ke kontrole různých interních chybových stavů serveru. Funkce používá dvoubajtový kód podfunkce, který specifikuje požadovaný typ testu. Normální odpověď obsahuje kopii požadavku případně další data, pokud jsou výsledkem testu. [1]

Požadavek		
Kód funkce	1 byte	0x08
Podfunkce	2 byty	tabulka podfunkcí
Počet cívek	N*2 bitů	
Odpověď		
Kód funkce	1 byte	0x08
Podfunkce	2 byte	tabulka podfunkcí
Počet cívek	N*2 bitů	
Chybový kód		
Kód funkce	1 byte	0x88
Chybový kód	1 byte	01,03 nebo 04

Tab. 15 Diagnostika

Kód podfunkce		Název
Hex	Dec	
00	00	Vrať data požadavku
01	01	Registruj komunikaci
02	02	Vrať diagnostický registr
03	03	Změň ASCII oddělovací znak
04	04	Přejdi do pasivního režimu
	05...09	REZERVOVÁNO
0A	10	Vynuluj čítače a diagnostický registr
0B	11	Vrať počet zpráv
0C	12	Vrať počet komunikačních chyb
0D	13	Vrať počet negativních odpovědí
0E	14	Vrať počet zpracovaných zpráv
0F	15	Vrať počet nezodpovězených zpráv
10	16	Vrať počet zpráv s negativním potvrzením
11	17	Vrať počet zpráv s příznakem zaneprázdnění
12	18	Vrať počet ztracených znaků
13	19	REZERVOVÁNO
14	20	Vynuluj čítač ztracených znaků
	21...65535	REZERVOVÁNO

Tab. 16 Kódy podfunkcí podporované sériovými zařízení

#### 4.9 11 (0x0B) Čti čítač komunikačních událostí (Get Comm Event Counter) – pouze pro sériovou linku

Tato funkce slouží k získání stavového slova a hodnoty čítače komunikačních událostí. Čítač událostí je inkrementován po každém úspěšném dokončení požadavku. Normální odpověď obsahuje dvoubajtové stavové slovo a dvoubajtový počet událostí. [1]

Požadavek		
Kód funkce	1 byte	0x0B
Odpověď		
Kód funkce	1 byte	0x0B
Status	2 byty	0x0000 až 0xFFFF
Počet událostí	2 byty	0x0000 až 0xFFFF
Chybový kód		
Kód funkce	1 byte	0x8B
Chybový kód	1 byte	01 nebo 04

Tab. 17 Čti čítač komunikačních událostí

#### 4.10 12 (0x0C) Čti záznam komunikačních událostí (Get Comm Event Log) – pouze pro sériovou linku

Tato funkce slouží k získání stavového slova, hodnoty čítače komunikačních událostí, čítače zpráv a záznamu komunikačních událostí. Stavové slovo a čítač událostí má stejný význam jako u funkce 11 (0x0B). Normální odpověď obsahuje dvoubajtové stavové slovo, dvoubajtový počet událostí, dvoubajtový počet zpráv a pole obsahující 0 až 64 bytů záznamu událostí. [1]

Požadavek		
Kód funkce	1 byte	0x0C
Odpověď		
Kód funkce	1 byte	0x0C
Počet bytů	1 byte	N
Status	2 byty	0x0000 nebo 0xFFFF
Počet zpráv	2 byty	0x0000 až 0xFFFF
Záznam událostí	2 byty	0x0000 až 0xFFFF
Počet událostí	(N-6) bytů	
Chybový kód		
Kód funkce	1 byte	0x8C
Chybový kód	1 byte	01 nebo 04

Tab. 18 Čti záznam komunikačních událostí

#### 4.11 15 (0x0F) Zapiš více cívek (Write Multiple Coils)

Tato funkce slouží k nastavení až 1968 cívek do stavu ON nebo OFF. Požadavek je adresa prvního výstupu, který se má nastavit a hodnoty, na které se mají výstupy nastavit. [1]

Požadavek		
Kód funkce	1 byte	0x0F
Počáteční adresa	2 byty	0x0000 nebo 0xFFFF
Počet výstupů	2 byty	1 až 1968 (0x7B0)
Počet bytů	1 byty	N
Hodnota výstupů	N bytů	
Odpověď		
Kód funkce	1 byte	0x0F
Počáteční adresa	2 byty	0x0000 nebo 0xFFFF
Počet výstupů	2 byty	1 až 1968 (0x7B0)
Chybový kód		
Kód funkce	1 byte	0x8F
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 19 Zapiš více cívek

#### 4.12 16 (0x10) Zapiš více registrů (Write Multiple Registers)

Tato funkce slouží k zápisu bloku až 120 registrů. V požadavku je specifikována adresa prvního registru, který se má zapsat, počet registrů a hodnoty, které se mají zapsat. Normální odpověď obsahuje počáteční adresu a počet zapsaných registrů. [1]

Požadavek		
Kód funkce	1 byte	0x10
Počáteční adresa	2 byte	0x0000 nebo 0xFFFF
Počet výstupů	2 byty	1 až 120 (0x78)
Počet bytů	1 byty	2*N
Hodnota výstupů	N bytů	
Odpověď		
Kód funkce	1 byte	0x10
Počáteční adresa	2 byty	0x0000 nebo 0xFFFF
Počet výstupů	2 byty	1 až 120 (0x78)
Chybový kód		
Kód funkce	1 byte	0x90
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 20 Zapiš více registrů

#### 4.13 17 (0x11) Sděl identifikaci (Report Slave ID) – pouze pro sériovou linku

Tato funkce slouží ke zjištění typu zařízení, současného stavu a dalších informací o zařízení. Konkrétní obsah odpovědi je závislý na typu zařízení. [1]

Požadavek		
Kód funkce	1 byte	0x11
Odpověď		
Kód funkce	1 byte	0x11
Počet bytů	1 byte	
Indikátor běhu	1 byte	0x00=OFF,0xFF=ON
Chybový kód		
Kód funkce	1 byte	0x91
Chybový kód	1 byte	01 nebo 04

Tab. 21 Sděl identifikaci

#### 4.14 20 / 6 (0x14 / 0x06) Čti záznam ze souboru (Read File Record)

Tato funkce slouží ke čtení záznamu ze souboru. Soubor je složen až z 10000 záznamů číslovaných od 0 do 9999. Délka záznamu je udávána v počtu 16bitových registrů. Funkce

může číst několik bloků současně. Každý blok je v požadavku definován v samostatném sub-požadavku o délce 7 bytů. Normální odpověď je sérií sub-odpovědí, jedné pro každý sub-požadavek. [1]

#### **4.15 21 / 6 (0x15 / 0x06) Zapiš záznam do souboru (Write File Record)**

Tato funkce slouží ke zápisu záznamu do souboru. Soubor je složen až z 10000 záznamů číselovaných od 0 do 9999. Délka záznamu je udávána v počtu 16bitových registrů. Funkce může zapisovat několik bloků současně. Každý blok je v požadavku definován v samostatném sub-požadavku o délce 7 bytů + data. Normální odpověď je kopií požadavku. [1]

#### **4.16 22 (0x16) Zapiš registr s maskováním (Mask Write Register)**

Tato funkce slouží k modifikaci uživatelského registru použitím AND a OR masky. Funkci lze použít k nastavení nebo vynulování jednotlivých bitů registru. V požadavku je specifikována adresa registru, AND maska a OR maska. Algoritmus funkce je následující:  $\text{Registr} = (\text{Registr AND And\_Maska}) \text{ OR } (\text{Or\_Maska AND (NOT And\_Maska)})$  Normální odpověď je kopií požadavku a je vrácena poté, co je registr modifikován. [1]

#### **4.17 23 (0x17) Čti/Zapiš více registrů (Read/Write Multiple Registers)**

Tato funkce provádí kombinaci čtení a zápisu registrů v jedné MODBUS transakci. Operace zápisu je provedena před operací čtení. V požadavku je specifikována adresa prvního registru a počet registrů, které se mají číst a adresa, počet registrů a hodnoty, které se mají zapsat. [1]

Požadavek		
Kód funkce	1 byte	0x17
Poč. adresa pro čtení	2 byty	0x0000 nebo 0xFFFF
Počet registrů pro čtení	2 byty	1 až 118 (0x0076)
Počáteční adresa pro zápis	2 byte	0x0000 nebo 0xFFFF
Počet registrů pro zápis	2 byty	1 až 118 (0x0076)
Počet zapisovaných bytů	1 byty	2*N
Hodnoty registrů	N*2 bytů	
N=Počet registrů pro čtení		
Odpověď		
Kód funkce	1 byte	0x17
Počáteční adresa	2 byty	M*2
Počet výstupů	M*2 bytů	
M=Počet registrů pro psaní		
Chybový kód		
Kód funkce	1 byte	0x97
Chybový kód	1 byte	01,02,03 nebo 04

Tab. 22 Čti/Zapiš více registrů

#### 4.18 24 (0x18) Čti FIFO frontu (Read FIFO Queue)

Tato funkce umožňuje číst obsah FIFO fronty registru. Funkce vrací počet a obsah registrů ve frontě. Může být přečteno až 32 registrů; délka fronty + až 31 registrů ve frontě. Je-li fronta delší než 31 registrů, je vrácen chybový kód 03. [1]

#### 4.19 43 / 14 (0x2B / 0x0E) Čti identifikaci zařízení (Read Device Identification)

Tato funkce umožňuje čtení identifikace a dalších údajů týkajících se popisu zařízení. Identifikace zařízení je složena z množiny objektů, z nichž každý má svou identifikaci. Existují tři skupiny objektů:

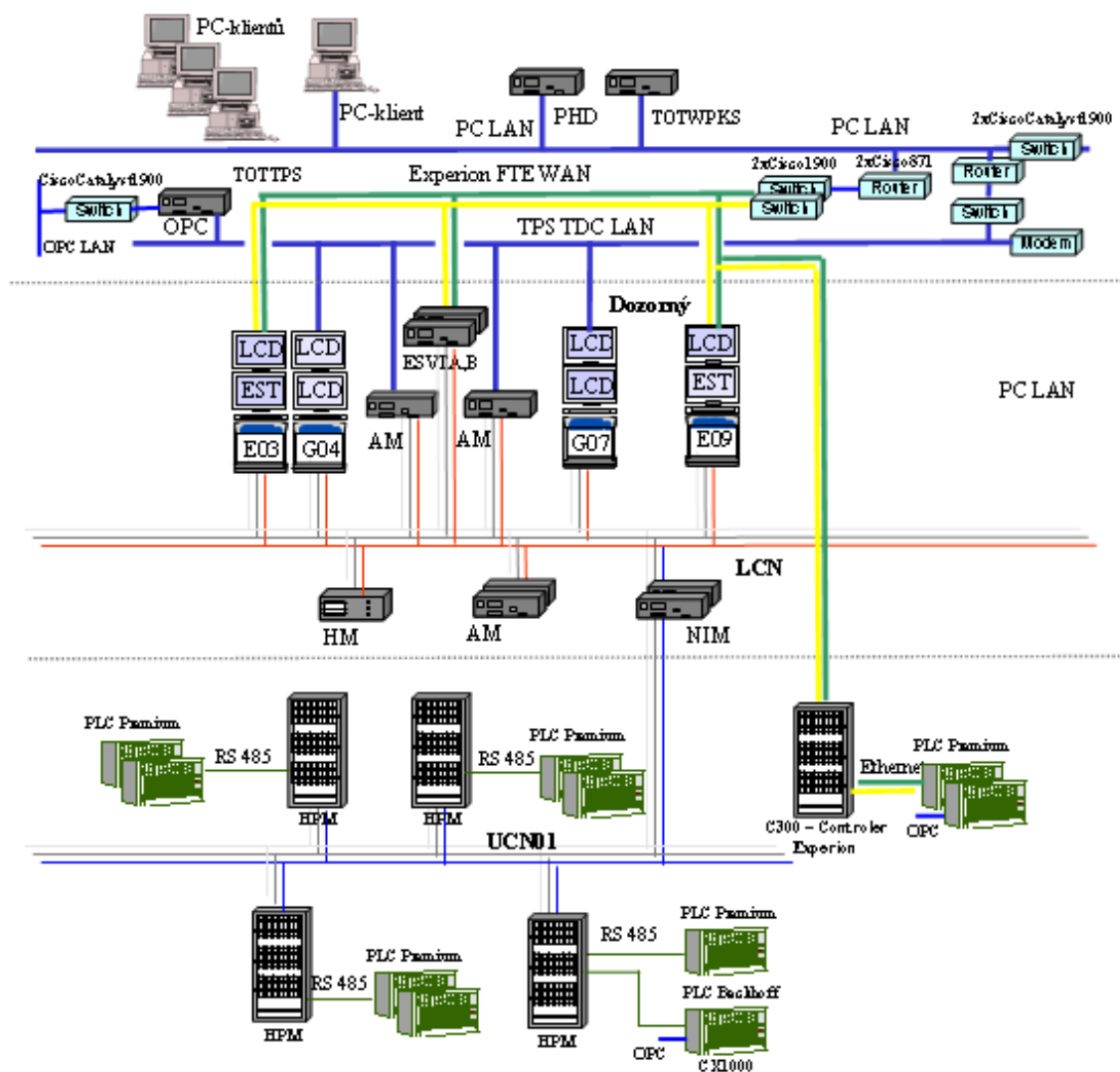
- Základní identifikace zařízení
- Obvyklá identifikace zařízení
- Rozšířená identifikace zařízení

## **5 DEFINICE ŘÍDICÍHO SYSTÉMU V ENERGETICE HONEYWELL S POHLEDU MOŽNÉ IMPLEMENTACE ČTEČKY RFID KARET**

Global User Station (GUS) byla vyvinuta firmou Honeywell pro průmyslový řídicí systém TDC3000/TPS a nyní také pro zcela nový systém EXPERION jako prostředek na systémové a softwarové práce, řízení průmyslových procesů a pro prezentaci nebo sběr technicko-ekonomických dat. Schéma je zobrazeno na obrázku 6. Zásadním rozdílem v činnosti těchto stanic je rozdílné komunikační rozhraní. Původní sériové komunikační rozhraní (TDC 3000) je nahrazováno rozhraním Ethernet (EXPERION). GUS pracuje pod operačním systémem Microsoft WINDOWS NT. Technické zpracování stanice umožňuje připojení jak na místní síť, tak i v reálném čase na Local Control Network (LCN). Obsluha je umožněn přímý vstup k procesním údajům a řízení procesů z displejů nakonfigurovaných v rámci softwarového vybavení GUS (Native Window). GUS může disponovat celou řadou přídatných periférií, jako jsou zařízení pro záznam dat (Hard disk, jednotek CD-ROM, Floppy disků a zařízení pro tisk kompatibilních s Windows NT.

Systémy TDC3000/TPS (Totally Distributed Control system / Total Plant Solution) a EXPERION jsou distribuované řídicí systémy s definovanou hierarchií. To znamená, že všechny jejich řídicí, regulační, dohlížecí, prezentační a další prvky jsou rozděleny mezi řadu specializovaných procesorů mezi sebou navzájem komunikujících v rámci příslušné sítě na základě pravidel daných strukturou sítě a na základě vzájemné hierarchie mezi jednotlivými prvky na síti. Běžný uživatel nepozná, zda pracuje na systému TDC 3000 nebo EXPERION, protože koncové zařízení jsou společná. Rozdílné je pouze technologie přenosu dat. Technik nebo projektant, který bezpečnostní prvky implementuje do těchto zařízení, musí být obeznámen správcem ASŘ do kterého z těchto systému budou data importována. Technická dokumentace a technická podpora obou systémů jsou na slušné úrovni. Silnější technická podpora je všeobecně sledována v systému TDC 3000. Systém EXPERION ještě není tolik rozšířený, ale postupně nahrazuje původní systém TDC 3000 pracující na sériovém rozhraní. Přístup k procesním údajům, operátorské stanice a technologie zpracování dat zůstala zachována. Oba systémy tedy pracují v bez kolizní součinnosti. [3]





Obr. 6 Schéma ASŘ Honeywell

Hierarchicky se síť skládá z několika částí:

- lokální řídicí síť (LCN), které představují nejvyšší úroveň a nemají přímý styk s technologickým procesem.
- jedné nebo více procesních sítí, které přímo komunikují s technologickým procesem, ale mají omezené funkce, zejména prezentační (X-vrstva, vrstva s WINDOWS NT).
- Součástí LCN několik základních typů specializovaných procesorů.
- Operátorská stanice, jinak Universal Station (US). US má dva hlavní režimy práce:
  - operátorský - je možná přímá interakce (řízení) s technologickým procesem

- inženýrský - není možná interakce s technologickým procesem, ale zato je možné mít přístup ke všem detailům a prostředkům operačního systému. Pouze v tomto režimu je možné např. vytvářet grafická schémata, provádět softwarové práce a systémové operace.

Na US se pracuje pouze ve standardním programovém prostředí TDC 3000.

- Operátorská stanice s X-vrstvou, jinak Universal Station X (U<sup>X</sup>S), která kromě plné funkčnosti jako US umožňuje i přístup na ethernetovou podnikovou informační síť pomocí tzv. X-vrstvy. U<sup>X</sup>S umožňuje práci nejen ve standardním programovém prostředí TDC 3000, ale i v programovém prostředí X-vrstvy.
- Operátorská/inženýrská stanice Global User Station (GUS). Na GUS se pracuje v programovém prostředí WINDOWS NT. Kromě základního softwarového vybavení GUS operačním systémem WINDOWS NT a softwarem umožňujícím přístup do prostředí odpovídajícího US - tzv. Native Window, může být stanice navíc vybavena (podle rozsahu dodávky určené zákazníkem) řadou softwarových podpůrných prostředků umožňujících efektivní provádění inženýrských prací na TDC3000 a ovládání procesu z nestandardních uživatelsky navržených a nakonfigurovaných displejů.
- Aplikační modul (AM), jedná se o pomocný programový modul, na kterém obvykle běží programy, starající se o vazby mezi různými procesními proměnnými, programy pro směnové, denní a měsíční přehledy a bilance apod.
- Modul historie (HM), což je záznamová jednotka (disk), který jednak obsahuje všechny soubory patřící k operačnímu systému TDC3000 a EXPERION, jako jsou konfigurační programy, textový editor, obrazový editor, kompilátor programovacího jazyka a mnoho dalších a jednak jsou na něm ukládány vybrané procesní hodnoty, procesní alarmy, zásahy operátora a další hodnoty, dokumentující činnost řízeného technologického procesu a jeho okolí
- Computer Gateway (CG) zprostředkovává styk mezi LCN a jiným počítačem (PC), který je použit pro další funkce, nesloužící bezprostředně k řízení technologie, ale vyžadující znalost hodnot z technologického procesu (např. zpracování údajů docházkového systému)
- Network Gateway (NG) spojuje 2 různé LCN:

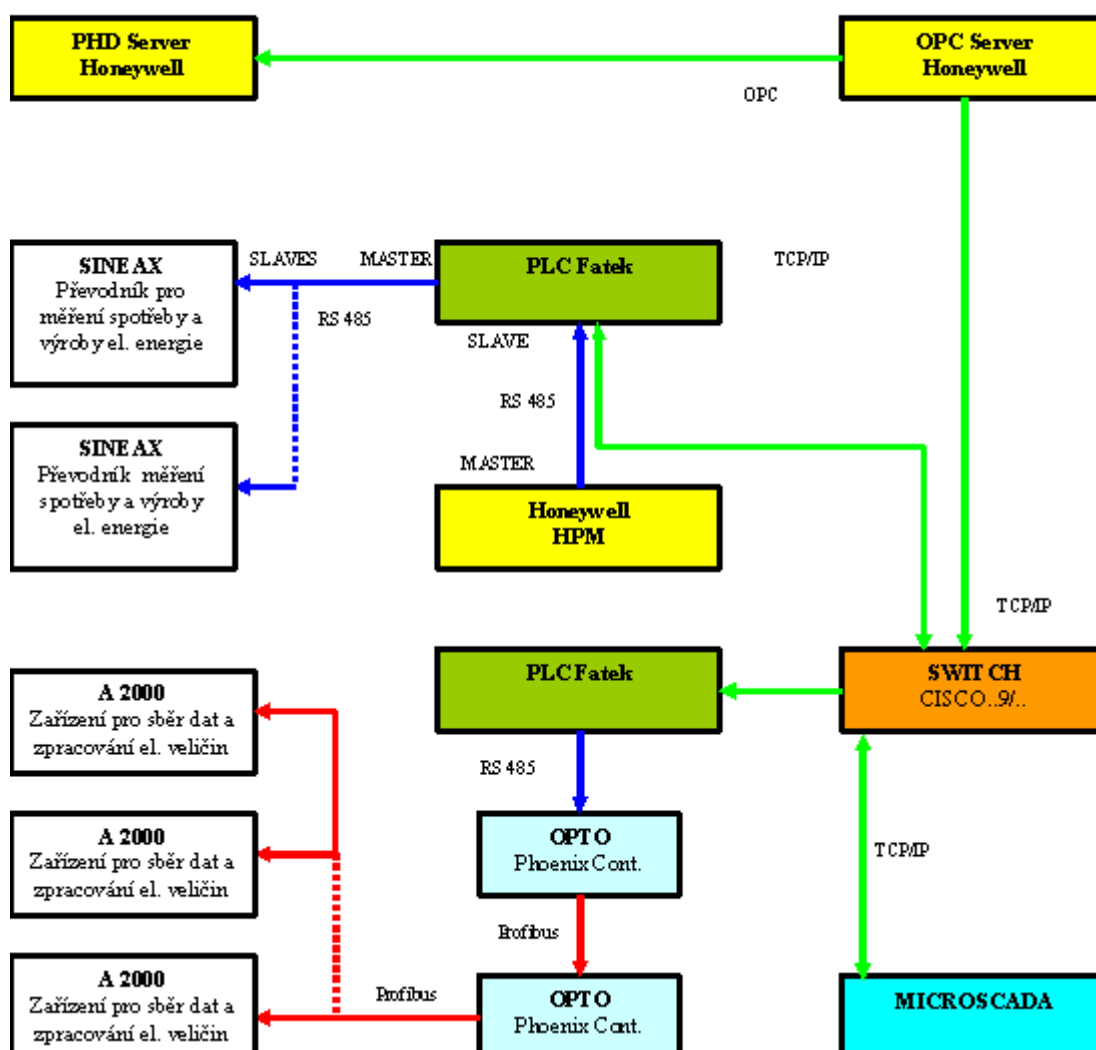
- Network Interface Module (NIM) spojuje LCN s Universal Control Network (UCN). UCN integruje nejvýkonnější zařízení pro řízení provozu, technologie a koordinaci systémů.

Universal Control Network (UCN) je vysokorychlostní a poměrně funkčně spolehlivá procesní síť. K UCN je možné připojit řadu zařízení pro přímé řízení procesů jako např. Advanced Process Manager (APM), High-Performance Process Manager / History Process Modul (HPM) pro systém TDC 3000 a Controller C300 pro systém Experion, Logic Manager (LM) nebo libovolnou kombinaci těchto zařízení. [4]

K základním součástem těchto zařízení patří I/O Modul a Process Modul. I/O Modul poskytuje širokou paletu vstupně-výstupních (I/O) funkcí jak pro vlastní ovládání pohonů, tak pro monitorování dat. Process Modul obstarává řídicí funkce včetně regulačních, logických, sekvenčních, binárních.

Současná vzájemná komunikace řídicích systémů Honeywell je založena na propojení všech prvků pomocí ethernetového fyzického média a přepínače (např. Cisco Catalyst 1900). Tyto systémy jsou komunikačním protokolem MODBUS RTU standardně vybaveny. Starý systém TDC 3000 je určený pro komunikaci pomocí sériové linky, zatímco novější systém EXPERION je určený pro komunikaci pomocí TCP-IP protokolu. Současné propojení systémů ASŘ je zakresleno na obr. 7. [3]

Nejdůležitější funkcí programovatelného automatu (PLC) je možnost SW a HD konverze, převodu rozhraní a překladu komunikačních protokolů. Příkladem může být příjem dat z převodníků pomocí sériové linky RS485 komunikujících protokolem Wiegand a konverzí tohoto protokolu na jeho výstupu pomocí vhodného převodníku již můžeme použít protokol MODBUS na sériové lince. Například přes rozhraní RS485 mohou být přenášeny data protokolu Wiegand do PLC. Tyto data jsou překládány z protokolu Wiegand do protokolu MODBUS pomocí převodníku Wie485. Konvertovaný protokol MODBUS je poté možné využít k přenosu dat buď po sériové lince, nebo přes ethernet. Záleží pouze na dalším aplikovaném modulu v programovatelném automatu. Tímto způsobem je možno realizovat implementaci čtečky RFID karet do systému řízení energetických procesů Honeywell. Integraci čtečky RFID karet do systému řízení je věnována celá praktická část této diplomové práce.



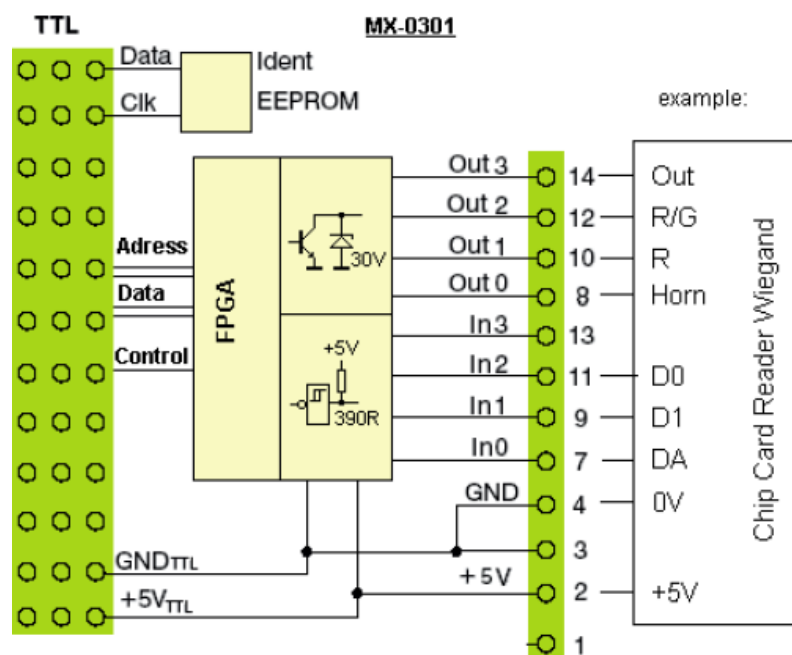
Obr. 7 Systém ASŘ Honeywell z hlediska napojení externích zařízení

## **6 SOUČASNÝ STAV ŘEŠENÍ IMPLEMENTACÍ ČTEČEK RFID KARET DO ŘÍDICÍCH SYSTÉMŮ PO MODBUSU**

V současnosti na českém trhu zcela chybí čtečky RFID karet, které přímo komunikují po komunikačním protokolu MODBUS. Integrace čteček do systémů řízení ASŘ je vytvářena pouze pomocí převodníků nebo integrátorů, a tyto zařízení také nejsou na domácím trhu příliš rozšířena. Dalším problémem jak integrovat čtečku RFID karet do systému řízení, které komunikuje na sériovém rozhraní je to, že integrátory pracují na protokolu MODBUS TCP/IP, přestože v automatizaci je stále nejoblíbenější a nejpoužívanější sériové rozhraní pro komunikaci mezi zařízeními a koncovými akčními členy. Převodník mezi protokoly Wiegand a Modbus na sériové lince je na českém trhu jediný – Wie485. To není příjemné zjištění, chceme-li provést zamýšlenou integraci čtečky RFID do nejrozšířenějšího systému řízení ASŘ Honeywell TDC 3000. Integrace do systému Honeywell EXPERION je jednodušší v tom, že lze použít integrátorů, které jsou určené pro komunikaci na protokolu TCP/IP. Na českém trhu se nabízí několik zařízení, které lze použít pro překlad protokolu Wiegand na protokol Modbus na TCP nebo zařízení, které lze použít pro integraci nepřímo pomocí dalších převodníků nebo překladačů. Ty nejvhodnější popisují následující podkapitoly.

### **6.1 Submodul MX-0301 pro připojení čtečky karet Wiegand**

Submodul MX-0301 od společnosti TECO obsahuje obvody rozhraní bez galvanického oddělení pro připojení čtečky čipových identifikačních karet PR500 s kódováním Wiegand. Submodul je obsluhován a diagnostikován z programovacího prostředí MOSAIC. Toto prostředí není však volně uvolněno pro komerční použití a je nutné jej zakoupit. Tímto se celá integrace čteček RFID karet velmi prodražuje. Nehledě na to, že submodul MX-0301 je určen pro modulární řídicí systém TC700, tedy je nutné dokoupit další zařízení pro možnou integraci. Na druhou stranu systém TC700 umožňuje integraci prvků určené pro řízení vytápění, vzduchotechniky a klimatizace (HVAC), měření a řízení spotřeby všech typů energií (Energy management), integraci bezpečnostních prvků budovy (EVS/EPS) nebo také integraci přístupových systémů - dveře, brány, průchody (Access control). Výsledkem může být dokonalá koordinace procesů budovy, optimalizace řízení zabezpečující vyšší úspory spotřeby médií.



Obr. 8 Subsystem Tecomat MX-0301

## 6.2 RFID čtečky EMS Cobalt C

Dalším prvkem vhodným pro integraci, který se objevil na českém trhu je čtečka RFID karet EMS Cobalt C. Řada C RFID čteček Cobalt je nejmenší RFID čtečka zahrnující integrovanou anténu. Jedná se o kompaktní zařízení s průmyslovým krytím IP 66 zapouzdřené v odolném plastu. Tato RFID čtečka pracuje na frekvenci HF a je vhodná jak pro kancelářské tak i průmyslové aplikace. Řada C zahrnuje dva základní modely RFID čtečky. Jedná se o model C405 s obecným použitím a C1007, který má možnost připojení externí antény a je speciálně navržen pro montáž na kov. Zvláštností této řady produktů je možnost čtení pod hladinou vody, oleje a jiných kapalin. RFID čtečky Cobalt C je možné připojit přímo k počítači, nebo programovatelnému automatu pomocí rozhraní RS-232, RS-422, RS-485, USB. Za použití EMS hub, nebo EMS gateway je možné čtečky propojit do sítě a využít tak protokolů Modbus TCP, DeviceNet, případně ethernet IP. K integraci do systémů řízení Honeywell TD3000 není vyžadováno dalšího externího zařízení (např. převodníky atd.).



*Obr. 9 Čtečka RFID karet EMS Cobalt C*

### 6.3 I/O Controller PL

Na českém trhu se nedávno objevil I/O Controller společnosti HW Group. I/O Controller je jednotka připojená do sítě Ethernet, obsluhující 8x binární vstup, 8x binární výstup a sériové rozhraní RS-232/485 (Terminal server). Všechna rozhraní jsou přístupná po počítačové síti pomocí M2M protokolu na TCP/IP. Specifickou aplikací je vzdálený přístup na technologii, která je ovládána přes RS-232 nebo RS-485. I/O Controller je ideální zařízení, pokud někde potřebujeme vyvést nějakou binární signalizaci, ze serveru zjistit stav nějakého binárního vstupu, nebo tyto případy spojit zároveň se sériovou linkou RS-232 pro připojení například čtečky čárového kódu, RFID čtečky, displeje a podobně. Pomocí HW VSP můžeme k zařízení přistupovat pomocí originálního softwaru. Vzdálené ovládání napájení a jiné technologie je řešeno přímo po RS-232 a RS-485. Kontrolér je možné použít pro tyto možné aplikace - vzdálené ovládání displejů a světelné signalizace, vzdálené čtení stavu vstupů PIR čidel nebo dveřních kontaktů a zabezpečovacích systémů, čtečky RFID karet, bezpečnostní alarmy, docházkové systémy, zařízení ovládání a logování UPS, skenery čárových kódů. Značnou nevýhodou se jeví podporované protokoly ARP, TCP + NVT (Network Virtual, Terminal), které nepatří mezi příliš rozšířené, také nepříznivá je vysoká cena.



Obr. 10 I/O Controller PL

#### 6.4 Regulátor a integrátor pro aplikace automatizace budov – HAWK

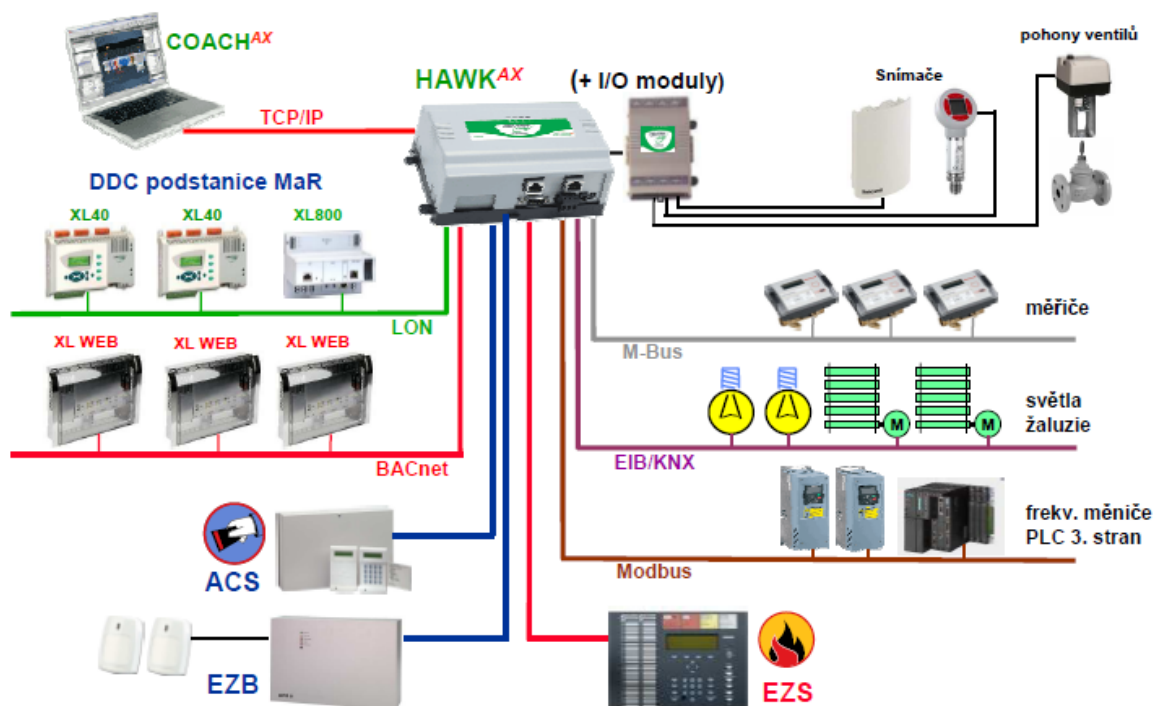
Společnost Honeywell ve svém sortimentu nabízí velmi zdařilý integrátor HAWK. Integrátor HAWK je kompaktní regulátor postavený na Niagara<sup>AX</sup> Framework<sup>(R)</sup> určený pro integraci systémů automatizace budov jako jsou řídicí systémy vytápění, ventilace, klimatizace, systémy sledování spotřeb energií a ovládání osvětlení, systémy komfortní individuální regulace a dalších systémů nutných pro chod budovy. HAWK je univerzální zařízení poskytující uživateli komplexní služby, jako řízení jednotlivých technologií budovy, sběr dat, sběr a distribuce alarmových hlášení, nastavení časových programů a dozor celé technologie přes uživatelsky příjemné prostředí, a to vše přes internetové připojení. HAWK obsahuje vestavěný web server a jeho součástí je i vývojové prostředí pro vytváření aplikace.



Obr. 11 Integrátor a regulátor HAWK



Integrátor HAWK umožňuje komunikaci na různých protokolech: Modbus, M-Bus, BACnet, Echelon, EIB-KNX nebo OPC. Integrátor také umožňuje navrhnout a aplikovat rozličnou architekturu.



Obr. 12 Příklad architektury s využitím integrátoru HAWK<sup>AX</sup>

Značnou nevýhodou pro integraci čtečky RFID karty do systému řízení Honeywell TDC300 je chybějící rozhraní sériové komunikace, což však u systémů Honeywell EXPERION není potřeba. Další minus pro možnou integraci do stávajících systémů je velmi vysoká cena integrátoru a nutnost zakoupeného vývojového prostředí. Pro samotnou integraci čtečky do systémů řízení v energetice je integrátor HAWK sice vhodný, ale pro účel jednoduché integrace na sériovém rozhraní pomocí protokolu Modbus velmi neekonomický.

## 6.5 Wie485 - Převodník Wiegand na RS485

Velmi vhodným zařízením pro integraci čteček RFID karet do systému řízení v energetice Honeywell, MicroSCADA nebo AISE je převodník Wiegand na RS485 – Wie485. Wie485 je obousměrný převodník rozhraní Wiegand na RS485 pro systémy s bezkontaktními čtečkami. Wie485 umožňuje připojit čtečku přes průmyslovou sběrnici RS485 a číst kódy protokolem MODBUS RTU. Běžné bezkontaktní čtečky karet tak lze snadno připojit k počítači PC nebo k jinému systému. Výstupem z Wie485 je buď číslo přiložené karty nebo

přímo sada všech bitů přijatých protokolem Wiegand, čitelná přes RS485 protokolem MODBUS RTU. Wie485 umí také generovat Wiegand na základě informací přijatých pomocí MODBUS RTU. Wie485 je MODBUS Slave zařízení. Linka RS485 na převodníku je kompletně galvanicky oddělena od ostatních částí zařízení. Velkým plusem v možné integraci je nízká cena převodníku, která činí 2200 Kč. Pro univerzálnost, nízkou cenu a výbornou technickou podporu, kterou výrobce Papouch s.r.o. nabízí je ze všech jmenovaných zařízení pro integraci nejvhodnější a pro účel integrace jsem se pro také pro převodník Wie485 rozhodl. Veškeré informace o převodníku Wie485, jeho technických parametrech, včetně návrhu zapojení jsou popsány v kapitole 12.



Obr. 13 Převodník Wiegand na RS485 – Wie485

## **II. PRAKTICKÁ ČÁST**

## 7 DEFINICE A NÁVRH SYSTÉMU ACCESS\_ENERGO

Návrh systému Access\_Energó vychází z požadavku dosažení maximální bezpečnosti při práci, které je prováděno obsluhou na energetickém zařízení. Základem je za pomoci čtečky karet Jablotron JA-80 navrhnout a následně vytvořit systém, který maximálně zvýší bezpečnost práce na energetickém zařízení a minimalizuje možné hrozby a rizika, které z těchto činností vyplývají. Protože drtivá většina zařízení v energetickém průmyslu vzájemně komunikují pomocí komunikačního protokolu Modbus, je cílem této práce implementovat standardní čtečku RFID karet, která používá protokol Wiegand do energetického systému ASŘ a provést integraci dvou zcela odlišných odvětví průmyslu a to energetiky a průmyslu komerční bezpečnosti.

Projekt Access\_Energó je kombinací samostatně běžících aplikací, které mezi sebou spolupracují a vytvářejí tak celkový jednotný bezpečnostní systém. Pro projekt byly vytýčeny tyto požadavky:

- Definování koncepčně programové pojetí projektu implementace čteček.
- Zpracování databáze pracovníků a objektů.
- Vizualizace projektu pomocí editoru webových stránek HTML a skriptovacího jazyku PHP.
- Vytvoření hlavní aplikace Access\_ComPort\_Modbus s veškerými algoritmy, požadavkem na umožnění funkční sériové komunikace a kontrolním CRC kódem pro komunikaci na Modbusu.
- Vhodná volba PLC a jeho následné naprogramování.
- Vyřešit přenos informace z čtečky karet do PLC pomocí převodníku Wie485.
- Vytvořit funkční modelový systém Access\_Energó.

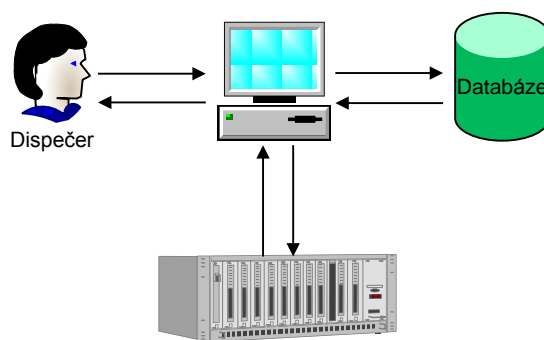
### 7.1 Definice požadované integrace za účelem zvýšení bezpečnosti práce v energetice za pomoci čtečky RFID karet

Definování požadovaných postupů pro zvýšení bezpečnosti práce lze zahrnout do několika bodů:

- Vedoucí práce se při vstupu do rozvodny identifikuje osobní kartou. Identifikační číslo karty je porovnáno s identifikačním číslem pracovníka, který se o práci na zařízení uchází.
- Pokud vše koresponduje se zadanými hodnotami v PC na dispečinku, které zadává dispečer při předávání příkazu “B” na VN zařízení nebo “BIP” na NN s údaji načtenými s čtečky zaměstnance, tak je systémem uvolněna pouze ta kobka nebo linka, na které jsou určené pracovní činnosti povoleny.
- Jestliže si obsluha zmýlí rozvodnu, linku nebo kobku, tak nenastane k žádnému pracovnímu úrazu, protože zařízení bude uzamknuto elektrickým zámkem. Tím je zabráněno nebezpečnému dotyku na živé části elektrického zařízení.
- Na každých dveřích je v projektu počítáno s adresovatelným kontaktem, jehož mechanická západka se uvolní až po verifikaci navolených údajů s údaji ze čtečky RFID.
- Pokud by došlo k otevření nesouhlasného pole, může být zařízení nastaveno tak, aby se rozezněl alarm jak v rozvodně, tak u dispečera na dispečinku. Tento alarm by upozornil pracovníka, že otevřel dveře nesouhlasné kobky a nedošlo by tak k těžkému pracovnímu úrazu elektrickým proudem.
- Informace o tom jestli se na zařízení pracuje nebo zdali již jsou práce ukončeny, může být zaneseno do spínacích algoritmů systému řízení energetických procesů. Těmto algoritmům je věnována kapitola 13.

## 7.2 Analýza prostředí a vstupní informačních dat do procesu integrace

Celý projekt lze popsat a znázornit na následujícím obrázku 14. Z obrázku lze vypožorovat, že základem integrace je výměna dat a informací mezi systémy a obsluhou. Obsluha je v našem případě dispečer, který ovládá hlavní aplikaci. Hlavní aplikace si vyměňuje data nebo informace s databázovým systémem a programovatelným automatem, který se nachází v provozu rozvodny a do kterého jsou zaneseny informace ze čteček RFID karet. Z obrázku lze také vypožorovat, že hlavní aplikace, která dostala pracovní pojmenování Access\_ComPort\_Modbus, a která je nainstalována na pracovním PC dispečera, tak se chová ke všem svým komunikačním partnerům jako Server Station. Stejně tak PLC se chová jako server jednotka ke čtečce RFID karet, která je do PLC implementována.



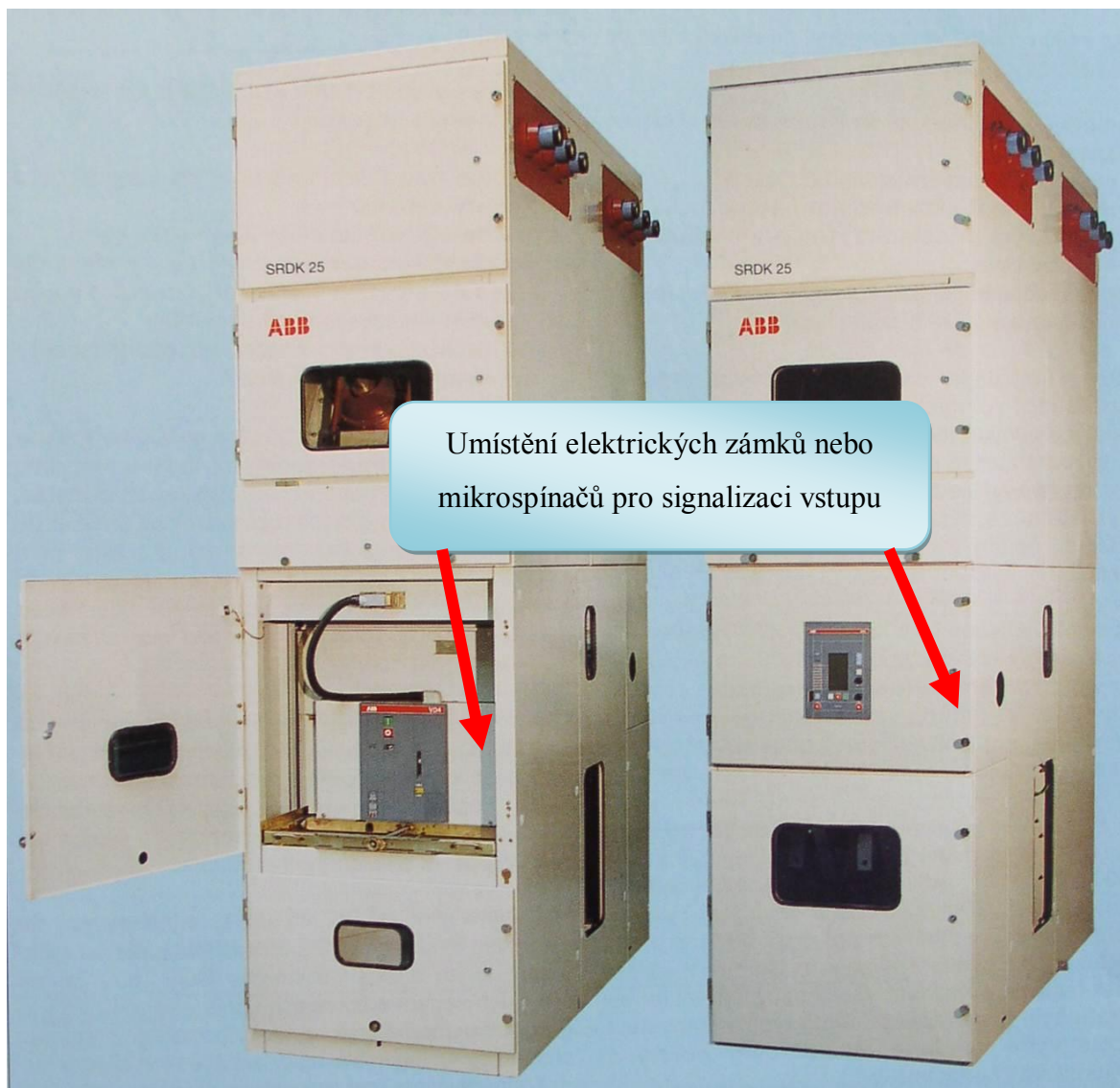
Obr. 14 Aplikační členění projektu

Klientem aplikace je dispečer. Dispečer ukládá požadované data do databázového systému. Tyto data jsou v tomto systému uložena tak dlouho, dokud práce na zařízení není ukončena a nejsou tak samotným dozorným z databáze vymazána. Hlavní běžící aplikace Access\_ComPort\_Modbus tyto data uloží do indexovaného ADT seznamu. Uložená data jsou srovnávána s informacemi z čteček karet, a pokud dojde ke shodě indexu a čísla karty v databázi, tak teprve poté je umožněn vstup do objektu. Stejně tak jsou umožněny práce na požadované lince, kobce nebo vývodu. Data jsou z ADT seznamu odstraněna společně s daty, které jsou uloženy v databázi SQL serveru.

Na obrázku 15 je znázorněno umístění elektrického zámku popřípadě mechanického kontaktu. Mechanických kontaktů by bylo použito pouze v případě, jestliže bychom sbírali pouze informace o otevření dveří kobky.

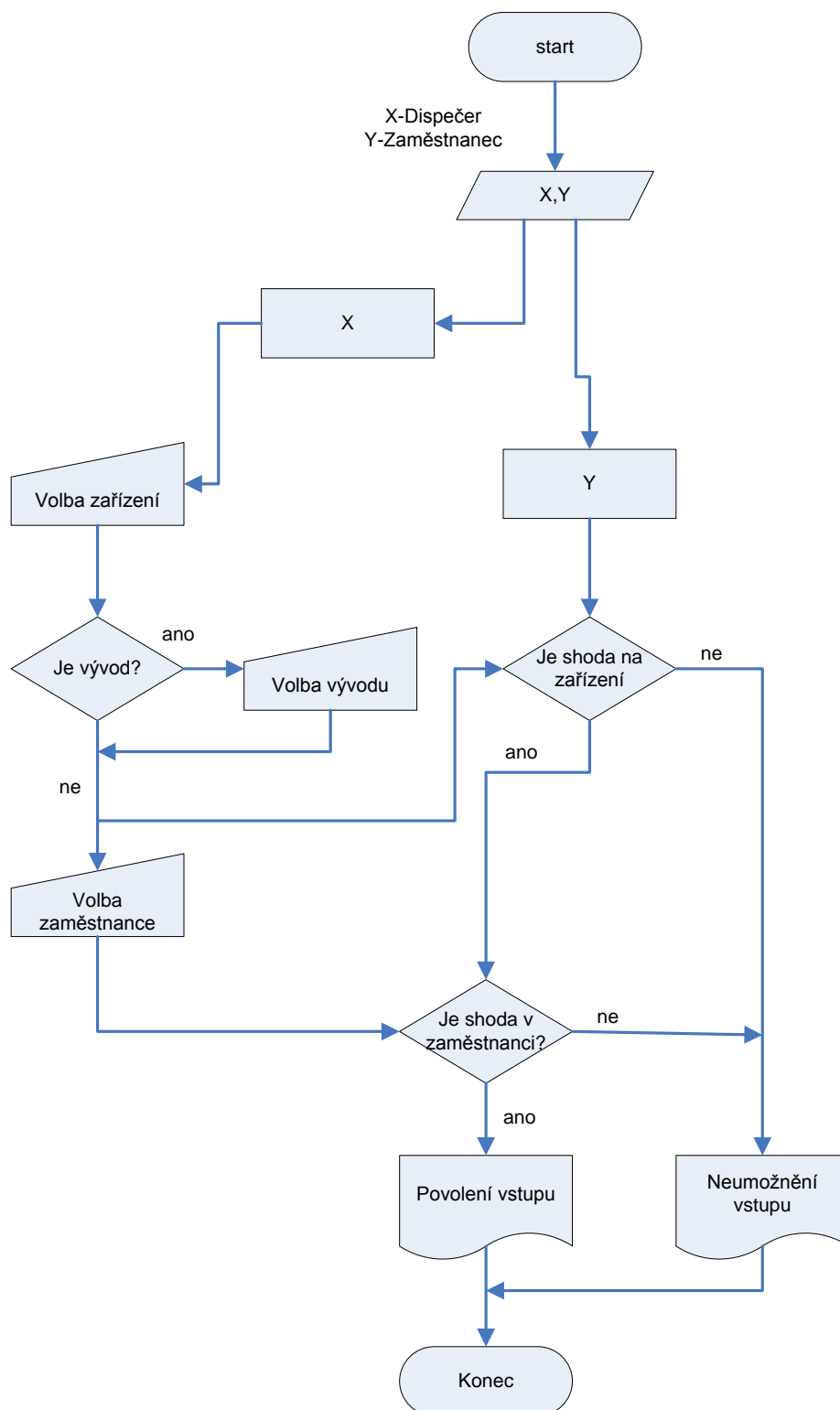
Varianta s mechanickými kontakty je jednodušší a méně náročná na softwarovou aplikaci, protože z kobek by přicházely do PLC pouze informace o otevření adresované kobky. Varianta s elektrickými zámky je náročnější na zpracování, protože je nutné pro vzájemnou komunikaci mezi čtečkou karet, PLC a PC s běžící hlavní aplikací nejen zpracovat přijaté data, ale také data zapisovat. K tomu je v systému Access\_Energo použit komunikační protokol MODBUS RTU, který nám poskytuje výměnu informací mezi výše zmiňovanými zařízeními.

Projekt je navržen tak, aby v každé kobce nebo vývodu byl nainstalován elektrický zámek, který uvolní pouze to zařízení k pracovní činnosti, které bylo navoleno dispečerem. Pokud zařízení v databázi souhlasí s informací ze čtečky karet, kterým se pracovník identifikoval, tak nemůže dojít k lidskému selhání a možnému vniknutí do zařízení, která jsou pod napětím.



*Obr. 15 Umístění zámku nebo signalizačního mikropínače*

Následující diagram, který je zobrazen na obrázku 16 popisuje algoritmus přístupu do objektu z důvodu požadovaných prací na vyhrazených elektrických zařízeních. Algoritmus spočívá ve společné volbě požadovaných pracovních činností mezi dispečerem a zvoleným pracovníkem, který je znázorněn v okně X, Y. Odtud se činnosti pracovníka a dispečera rozcházejí. Dispečer pouze navolí rozvodnu a vývod, na kterém mají probíhat pracovní činnosti. Pracovník se identifikuje vlastní kartou, a pokud zadané data, které dispečer uložil do databáze, souhlasí s identifikačními informacemi s karty, tak je umožněno pracovníkovi vstup do objektu a následné odemknutí požadované kobky nebo linky.



Obr. 16 Algoritmus pro povolení přístupu do objektu

### 7.3 Implementace čteček RFID karet do systémů ASŘ Honeywell

Celý projekt realizace implementace čteček RFID karet je postaven na převodníku rozhraní Wiegand z bezkontaktních čteček na RS485 od výrobce Papouch s.r.o. Tento převodník

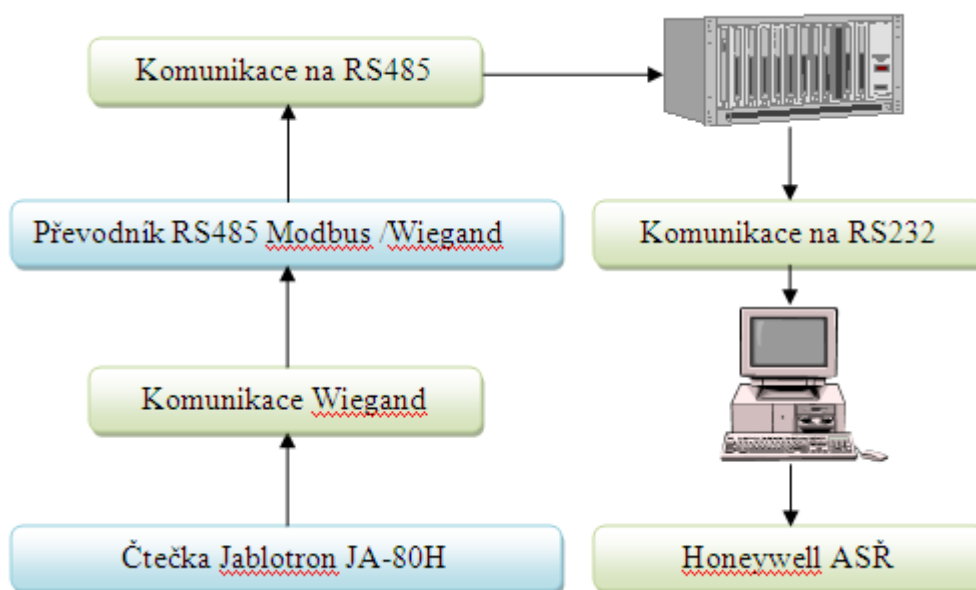


byl pro realizaci vybrán z důvodu kvalitně zpracované technické dokumentace. Na trhu (převážně zahraničním) jsem objevil několik převodníků Wiegand/MODBUS, avšak často výrobci uvádí v technické dokumentaci neúplné informace o zapojení převodníků, nedokonalé informace o rozložení paměti uživatelských nebo vstupních registrů.

Dalším prvkem, který jsem zvolil kvůli dokonale zpracované technické dokumentaci je PLC Fatek FBs-40MCR2-D24 od výrobce SEA spol. s r.o. Tato společnost bezplatně uvolnila vývojové prostředí pro programování programovatelných automatů Fatek, což je nezbytně nutné pro ovládání dveřních kontaktů a dalších akčních členů stejně tak pro automatizaci systémů. Obrovskou výhodou těchto PLC je modularita, která umožňuje rozšířit zařízení o další komponenty.

Při hledání vhodné čtečky RFID karet pro případnou integraci se objevily problémy s vhodným výběrem čtečky podle zadaných kritérií. Výrobci čteček v České Republice se striktně drží komunikačního protokolu Wiegand a jiné komunikační protokoly zařazují do svých výrobních projektů jen výjimečně. Na zahraničních trzích se objevují čtečky RFID, které mají přímo implementovaný komunikační protokol Modbus, ale pro českého spotřebitele je technická podpora ze strany zahraničních výrobců žalostná. Řešení se objevilo v použití převodníku Wiegand/Modbus, který s výbornou technickou podporou nabízí společnost Papouch s.r.o. Pomocí převodníku Wie485 lze nyní velmi jednoduše integrovat přímo jakoukoliv čtečku RFID karet, která komunikuje se svým okolím pomocí komunikačního protokolu Wiegand do jakéhokoliv systému řízení v energetice, protože Modbus je velmi univerzální a standardní protokol. Velmi rozšířené systémy řízení MicroSCADA od výrobců ABB používají ke komunikaci Profibus, ale to nečiní v integraci žádný problém, protože převodníků Modbus/Profibus je na trhu obrovské množství. Převodník Wiegand/Profibus jsem bohužel našel pouze jeden - SNAP 2-Ch Wiegand Interface Serial Communication Module.

Návrh implementace čteček bezkontaktních karet se systémy ASŘ Honeywell je zobrazen na obrázku 17.



Obr. 17 Návrh implementace čtečky RFID do systému ASŘ

Čtečka karet Jablotron JA-80H, která poskytuje data v protokolu Wiegand 26b je připojena podle katalogového zapojení na převodník rozhraní Wiegand z bezkontaktních čteček na RS485. Výstup z tohoto převodníku je připojen přímo na komunikační PORT2 programovatelného automatu, který komunikuje po rozhraní RS485. PORT1 je nastaven na komunikaci po sériové lince RS232 se vzdáleným serverem na kterém běží základní aplikace systému Access\_ComPort\_Modbus. Komunikace mezi převodníkem Wie485, PLC Fatek FBs-40MCR2-D24 a vzdáleným serverem je zprostředkováno prostřednictvím komunikačního protokolu MODBUS RTU.

#### 7.4 Konceptně programové pojetí projektu implementace čteček

Projekt implementace čtečky bezkontaktních karet je rozdělen do několika samostatně rozdělených oddílů:

- Vytvoření databáze pracovníků a objektů v prostředí MySQL a jejich vzájemné relační propojení.
- Vytvoření vizualizační aplikace pomocí skriptu HTML a jazyku PHP, která slouží jako interface mezi uživatelem a zařízením.
- Vytvoření hlavní aplikace Access\_ComPort\_Modbus v jazyce C++ pro komunikaci mezi PC a PLC pomocí sériové linky, výpočtu, verifikaci kontrolního součtu a algoritmus pro ovládání dveřních kontaktů podle požadavků obsluhy.

- Vytvoření vývojového diagramu pro příjem informací a příkazů ve vývojovém prostředí PLC Fatek FBs-40MCR2-D24 v aplikaci WinProLadder.
- Propojení všech hardwarových periférií, zařízení a následné oživení projektu.

Každá z těchto aplikací pracuje nezávisle na jiné aplikaci a dochází zde pouze k předávání informací a příkazových instrukcí. Jediným zásahem uživatele do běžící aplikace je navolení pracovníka, rozvodny a linky na které má obsluha pracovat a která se po identifikace vedoucího práce na zařízení pomocí čtečky RFID karet uvolní pro práci.

## 8 ZPRACOVÁNÍ DATABÁZE PRACOVNÍKŮ A OBJEKTŮ

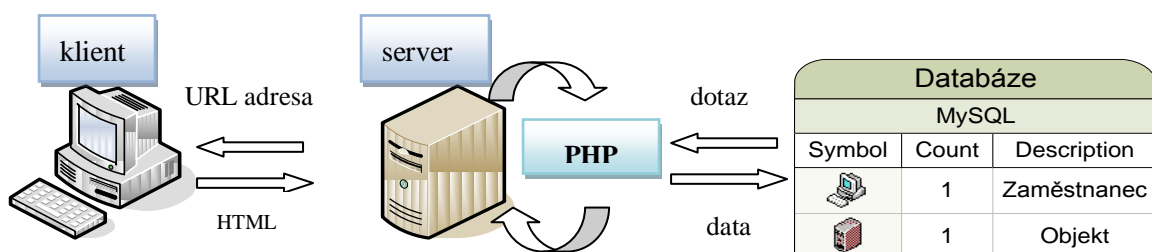
Základem vizualizace a zadávání příslušných informací do PLC je použití přístupu k databázím z WWW. Síť WWW je v současné době místem pro dynamické, často databázemi řízené webové aplikace. Tvorba webového serveru pomocí statických souborů HTML je nepřijatelná. Dynamické webové prezentace jsou flexibilní výtvoři s velkou kapacitou a lze je popsat spíše jako aplikace, než pouze jako stránky. V projektu je použit MySQL databázový systém s veřejným zdrojovým kódem, který je určený pro relační databáze. Tyto relační databáze jsou kolekcí vzájemně provázaných dat. [5]

Software MySQL se skládá z několika částí:

- Server MySQL (démon mysqld) – spouští a udržuje databáze
- Klient MySQL (program mysql) – nabízí rozhraní pro zprávu serveru
- Další nástroje k údržbě a jiným účelům

### 8.1 Propojení PHP+MySQL

Začlenění databáze do webové aplikace lze určitá data generovaná PHP načítat z databáze MySQL. Obsah stránek přestává být statický a stává se flexibilní.



Obr. 18 Komunikace serveru PHP+MySQL

Pro vytvoření databáze a práci se síťovými verzemi databázových systémů bez ohledu na operační systém je v projektu použito:

- Software webového serveru APACHE
- PHP
- MYSQL
- Webový prohlížeč
- Textový editor PSPad

Databáze pracující s SQL jsou založeny na modelu klient-server. Server je vybraný stroj, na kterém je nainstalovaný databázový systém a na jehož disku jsou uložena data. Server je také proces, který běží na zvoleném PC a který obsluhuje jednotlivé požadavky klientů. Klienti zadávají SQL příkazy a server tyto příkazy nad databází vykonává. Klientem pak může být konkrétní databázová aplikace nebo také řádkový terminál, ve kterém můžeme SQL příkazy zadávat přímo. Instalace MySQL serveru je velmi přehledně popsána ve skriptech s názvem Databázové systémy MySQL+PHP ing. Zdenky Prokopové CSc..

Aby byly funkce zprostředkovávající přístup na MySQL databázový server dostupné, musí být PHP zkompilováno s podporou MySQL a spuštěn démon MySQL. Pro připojení k serveru MySQL je nutné použít funkci `mysql_connect` a pro výběr databáze je použita funkce `my_select_db`. Přesná syntaxe výběru a připojení k databázi je popsána v kapitole 9.

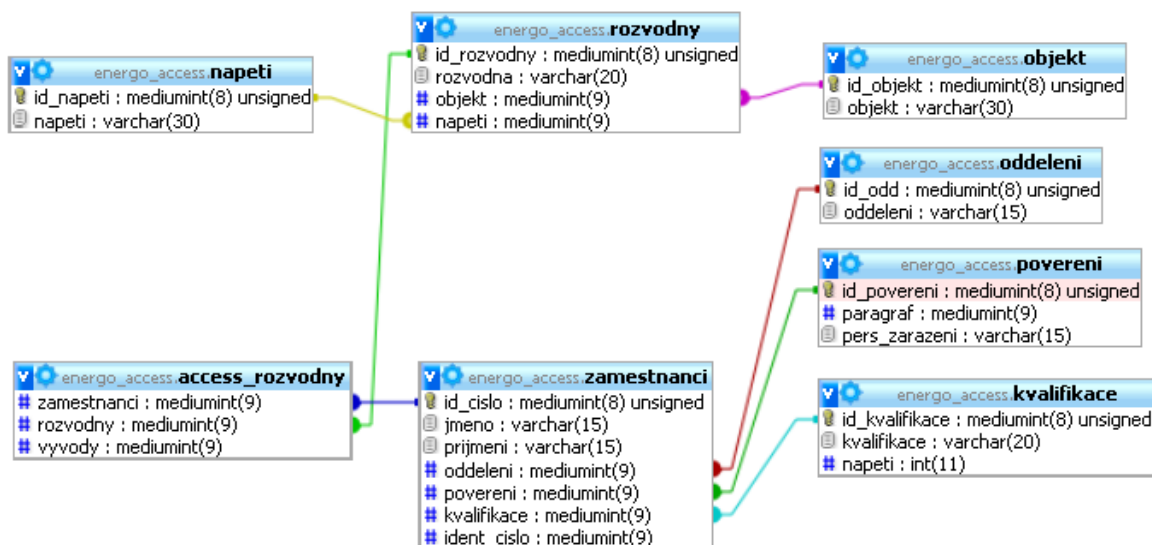
## 8.2 Vytvořená databáze projektu v MySQL

Základem funkčního projektu je vytvoření databáze pracovníků a objektů. Tyto databázové tabulky jsou vzájemně propojeny relačními vztahy, které jsou uvedeny v přiloženém schématu. Databáze je vytvořena z 9ti tabulek:

- Access\_rozvodny
- Kvalifikace
- Napětí
- Objekt
- Oddělení
- Pověření
- Rozvodny
- Vývody
- Zaměstnanci

Každá z těchto tabulek má svůj primární klíč s výjimkou tabulky ACCESS\_ROZVODNY. Tyto primární klíče jsou znázorněny na obrázku 19. Tabulka má relační vztah mezi entitami M:N (více k více), tedy několik zaměstnanců může pracovat ve více rozvodnách a naopak ve více rozvodnách může pracovat několik zaměstnanců. Ostatní relační vztahy mezi entitami jsou jedna ku více nebo jedna ku jedné. Například, jeden zaměstnanec může mít pouze jedno jedinečné identifikační číslo nebo několik zaměstnanců může pracovat na

jednom oddělení. V databázovém návrhu se vyskytuje také relační vztah jedna ku jedné, kdy pracovník může pracovat pouze na jednom zařízení. [6]



Obr. 19 Základní relační schéma databázového návrhu

### 8.2.1 Databázová tabulka ZAMĚSTNANCI

Databázová tabulka je složena ze 7mi sloupců a 5ti řádků. Primárním klíčem byl zvolen atribut ID číslo, který je jediným unikátním klíčem v tabulce ZAMĚSTNANCI. Cizími klíči v této tabulce jsou sloupce oddělení, pověření a kvalifikace, které se odkazují na primární klíče v přiřazených tabulkách. Sloupce jméno, příjmení a identifikace jsou přímo závislé na primárním klíči sloupce ID číslo.

ID číslo	Jméno	Příjmení	oddělení	pověření	kvalifikace	identifikace
1	Jan	Bednik	1	1	4	100
2	Jiri	Landak	1	2	4	200
7	Lukas	Brauner	2	3	3	300
22	Jan	Pekarek	3	4	2	400
27	Josef	Koler	4	5	1	500

Tab. 23 Tabulka ZAMĚSTNANCI

Sloupec oddělení je v poměru se zaměstnanci v poměru jedna k více, což znamená, že jeden zaměstnanec může být obsažen pouze v jednom oddělení, avšak jedno oddělení může obsahovat více zaměstnanců. Stejně tak je koncipován sloupec pověření a kvalifikace.

### 8.2.2 Databázová tabulka ACCESS\_ROZVODNY a VÝVODY

Tabulka ACCESS\_ROZVODNY je základní relační tabulkou databáze použité v projektu.

zaměstnanci	rozvodny	vývody
1	1	1
1	2	4
1	4	6
1	14	8
1	15	10
1	24	12
2	1	1
2	2	4
2	4	6
2	14	8
2	15	10
2	24	12
7	4	6
22	24	12
7	14	8
7	15	10
7	24	12
22	14	8
22	15	10
22	14	9
22	15	11
27	24	12
1	1	2
1	1	3
1	2	5
1	4	7
1	14	9
1	15	11
1	24	12
2	1	2
2	1	3
2	2	5
2	4	7
2	14	9
2	15	11
2	24	12
7	4	7
7	14	9
7	15	11
7	24	12

Tab. 24 Tabulka ACCESS\_ROZVODNY

Tato tabulka je založena na kombinaci sloupců zaměstnanci, rozvodny a vývody. Pro lepší pochopení vztahů mezi entitami tabulky přiblížíme vztah prvních dvou řádků. Pracovník s identifikací 1, může pracovat v rozvodně s id\_rozvodny 1 v kobce s id\_kobky 1. Druhý řádek vypovídá o tom, že pracovník s id\_číslem 1 může pracovat v rozvodně s id\_rozvodna 2 v kobce s id\_kobka 4.

Další důležitou tabulkou je tabulka vývodů, které určují, kde se budou povolovat práce na zařízení.

id_vyvod	vyvod
1	linka_3
2	linka_7
3	linka_11
4	linka_4
5	linka_8
6	kobka_1
7	kobka_2
8	pole_1
9	pole_10
10	pole_1
11	pole_8
12	privod_1

Tab. 25 Tabulka VYVODY

### 8.2.3 Databázová tabulka KVALIFIKACE

Tabulka KVALIFIKACE se skládá ze tří sloupců s názvy id\_kvalifikace, kvalifikace a napětí. Primárním klíčem této tabulky je sloupec id\_kvalifikace, který se odkazuje na cizí klíč v tabulce ZAMĚSTNACI. Sloupec napětí je závislý na primárním klíči id\_kvalifikace. Sloupec napětí je závislý na cizím klíči primární tabulky NAPĚTÍ. Touto tabulkou je určeno kdo a na jakém zařízení smí pracovat. Atribut kvalifikace je rozhodujícím činitelem pro povolení práce na daném zařízení.

id_kvalifikace	kvalifikace	napětí
1	bez_el_kvalifikace	4
2	poucena	3
3	znala	2
4	znala_s_vyssi_kval	1

Tab. 26 Tabulka KVALIFIKACE



#### 8.2.4 Databázová tabulka NAPĚTÍ

Jedná se o velmi jednoduchou tabulku o 2 sloupcích a 4 řádcích. Sloupec napětí je závislý na primárním klíči id\_napětí jenž je odkazuje na cizí klíč v tabulce ROZVODNY. Tato tabulka dodává pouze doplňující informace do databáze o velikosti napětí na daném zařízení.

id_napětí	napětí
1	VVN
2	VN
3	NN
4	MN

Tab. 27 Tabulka NAPĚTÍ

#### 8.2.5 Databázová tabulka OBJEKT

Tabulka OBJEKT obsahuje 7 řádků a 2 sloupce. Primárním klíčem tabulky je atribut id\_objekt. Sloupec objekt je závislý na primárním klíči id\_objekt. Stejně jako tabulka NAPĚTÍ je tabulka OBJEKT pouze doplňující informací, která nám sděluje, o jaký objekt se jedná. Primární klíč id\_napětí referuje na cizí klíč v tabulce ROZVODNY. Doplňuje informaci id\_zařízení o jaké zařízení se jedná a jak velké napětí je na sběrnících živých částí zařízení.

id_objekt	objekt
1	rozvodna_100kV
2	rozvodna_6kV
3	rozvadec_230/400V
4	rozvadec_24/60/110VDC
5	trafo_100kV
6	trafo_22kV
7	trafo_6kV

Tab. 28 Tabulka OBJEKT

#### 8.2.6 Databázová tabulka ODDĚLENÍ a POVEŘENÍ

Tabulka ODDĚLENÍ je velmi jednoduchou tabulkou, která obsahuje 2 sloupce a 4 řádky. Primárním klíčem je id\_odd, který referuje na cizí klíč se stejným jménem v tabulce ZAMĚSTNANCI.

id_odd	oddělení
1	elektro_provoz
2	elektro_udrzba
3	MaR
4	Ochranari

Tab. 29 Tabulka ODDĚLENÍ

Tabulka POVĚŘENÍ obsahuje také svůj primární klíč s názvem id\_pověření, který referuje na cizí klíč se stejným jménem v tabulce ZAMĚSTNANCI. Atributy paragraf a personální zařazení jsou závislé na primárním klíči id\_pověření, který referuje na cizí klíč v tabulce ZAMĚSTNANCI.

id_pověření	paragraf	personální zařazení
1	8	vedouci_provozu
2	7	mistr
3	6	predak
4	5	delnik
5	4	ucen

Tab. 30 Tabulka POVĚŘENÍ

### 8.2.7 Databázová tabulka ROZVODNY

Tabulka ROZVODNY obsahuje 6 řádků a 4 sloupce. Primárním klíčem je sloupec id\_zařízení. Tento klíč je základem pro vytvoření relace více k více v tabulce ACCESS\_ROZVODNY společně s atributy zaměstnanci a vývody pro propojení tabulek s atributy zaměstnanců, objektů a zařízení.

id_zařízení	rozvodna	objekt	napětí
1	r400.1	1	1
2	r400.2	1	1
4	r6.00	2	2
14	r4.01	3	3
15	r4.02	3	3
24	r2.0	4	4

Tab. 31 Tabulka ROZVODNY

## 9 VIZUALIZACE PROJEKTU POMOCÍ EDITORU WEBOVÝCH STRÁNEK HTML A SKRIPTOVACÍHO JAZYKU PHP

Dokumenty HTML jsou zvláštním typem textových souborů, pro které je potřeba textový editor pro editaci HTML stránek. HTML dokument se od obyčejného textového dokumentu liší tím, že kromě vlastního obsahu stránky obsahuje také informace o vzhledu a formátování stránky, které do textu nepatří a jsou určeny pouze pro prohlížeč. Tyto informace se prohlížeči předávají pomocí značek uzavřených mezi znaky `< >`, například `<html>`. Většina značek nese párové zrcadlení, které ukončují jejich platnost. Tyto formátovací značky jsou nazývány odborně tagy. Mezi některé značky je možné vložit doplňující atribut, který upřesňuje význam značky. Například pokud si přejeme zobrazit čáru o velikosti 20 pixelů, tak můžeme doplnit značku `<hr>` doplnit atributem nastavující šířku čáry: `<hr width = "20">`.

Každá webová stránka musí mít pevně stanovenou strukturu, která je tvořena speciálními formátovacími značkami. Prohlížeči je nutné sdělit, kde začíná a končí zdrojový text zobrazované stránky. Každý HTML začíná značkou `<html>` a končí značkou `</html>`. Zdrojový kód je umístěn mezi tyto dvě značky. Další důležitou součástí dokumentu je hlavička, která je umístěna mezi `<head>` a `</head>`. Vlastní obsah stránky je ohraničen značkami `<body>` a `</body>`. Mezi ně je možné napsat text, který se zobrazí na webové stránce. Jakmile je stránka napsaná a uložena, je možné spustit prohlížeč a prohlédnout napsaný skript. [7]

Pro vytváření dynamických stránek je vhodné použít v projektu hypertextový preprocesor PHP, který je otevřeným zdrojovým skriptovacím jazykem vhodným pro programování na webu, a který může být vnořen do HTML. Syntaxe PHP je obdobná jazyku C nebo Java, rozšířená o vlastní PHP vlastnosti jako např. o příkazy k integraci s databázemi. Předpokladem pro práci s jazykem PHP je správně nakonfigurovaný balíček WAMP (Windows, Apache, MySQL, PHP). Pro vytvoření projektu této Diplomové práce jsem použil volně poskytovaný webový server APACHE. K zadávání PHP skriptů jsem se rozhodl použít PSPad, která umožňuje barevně rozlišovat dané tagy a příkazy v jazyce PHP. Kód v jazyce PHP vložený do HTML stránky se ohraničuje značkami `<?php, ?>`. Vše co je uloženo mezi tagovací značky, tak je webovým serverem považováno za kód jazyka PHP. Soubor musí být opatřen příponou `*.php`. [8]

Projekt psaný v HTML a PHP je složen z několika samostatných souborů, které vytváří jednotnou aplikaci pro výměnu informací a dat mezi uživatelem a celým systémem, který provádí identifikaci a posléze umožnění vstupu do objektů.

Jde o tyto soubory v HTML v \*.html:

- Adresar1 – webová stránka zobrazí okno s výběrem pracovníka.
- Adresar2 – webová stránka zobrazí s výběrem rozvodny.
- Adresar3 – webová stránka s oknem aktivace poplachu.
- Adresar6 – webová stránka se seznamem výběru objektu.
- Banner – zobrazí název projektu.
- Menu – webová stránka s výběrem požadavků obsluhy.
- Události – tlačítko s žádostí o výpis událostí.
- Záhlaví – zobrazuje název databázové aplikace.
- Zápatí – zobrazí zápatí webové stránky s názvem aplikace.

Soubory psané v PHP kódu v \*.php:

- Datum – ukládá do proměnné \$datum aktuální datum.
- Menu – provádí nabídku hlavního adresáře.
- Načtení – načítá data z hlavní aplikace a ukládá je do proměnné \$data.
- Načtení\_roz – načítá data o navolení rozvoden z hlavní aplikace a ukládá je do proměnné \$data\_roz.
- Načtení\_čas\_roz - načítá aktuální čas z hlavní aplikace a ukládá je do proměnné \$datum\_roz.
- Načtení\_linka - načítá data o navolení linek z hlavní aplikace a ukládá je do proměnné \$data\_linka.
- Načtení\_pr\_roz - načítá data pracovníků pracujících na zařízení z hlavní aplikace a ukládá je do proměnné \$data\_pr\_roz.
- PokusEnergol- provádí žádost do databázového serveru pro výpis pracovníků
- Poplach – provádí příkaz pro vyvolání poplachu do hlavní aplikace Access\_Energol.
- Rozvodny – provádí žádost do databázového serveru pro výpis rozvoden
- Události\_roz – provádí potvrzení ukončení prací a opětovné načtení aktualizované databáze pracovníků, kteří pracují na elektrickém zařízení.
- Úvod – rozděluje základní webovou stránku na rámy.

Textové soubory pro přenos dat a informace ve formátu \*.txt:

- Data\_cas\_roz – ukládá čas přístupu pracovníka do rozvodny pro účely hlavní.
- Data\_linka – ukládá identifikaci zpřístupněné linky pro účely hlavní aplikace.
- Data\_pr\_roz - ukládá identifikaci vstupu pracovníků pro účely hlavní aplikace.
- Data\_roz - ukládá identifikaci zpřístupněné rozvodny pro účely hlavní aplikace.
- Poplach – přenáší kód žádosti o vyvolání poplachu.
- RozSQLlinka - přenáší kód žádosti z ADT seznamu hlavní aplikace zpřístupněné linky pro zobrazení ve webovém prohlížeči.
- RozSQLkobka - přenáší kód žádosti z ADT seznamu hlavní aplikace zpřístupněné kobky pro zobrazení ve webovém prohlížeči.
- RozSQLpracovník - přenáší kód žádosti z ADT seznamu hlavní aplikace identifikaci pracovníka pro zobrazení ve webovém prohlížeči.
- RozSQLlinka1 – soubor pro zálohování dat, který je totožný se souborem RozSQLlinka.
- RozSQLkobka1 - soubor pro zálohování dat, který je totožný se souborem RozSQLkobka.
- RozSQLpracovník1 - soubor pro zálohování dat, který je totožný se souborem RozSQLpracovník.
- Sumkobka – úložiště dat pro nastavení běhu aplikace
- Sumlinka – úložiště dat pro nastavení běhu aplikace
- Sumpracovník – úložiště dat pro nastavení běhu aplikace
- Ukončení – nastavuje aplikaci do výchozího nastavení po ukončení systému.
- Ukončeník – nastavuje aplikaci do výchozího nastavení po ukončení systému.
- Ukončeníl – nastavuje aplikaci do výchozího nastavení po ukončení systému.
- Ukončenít – nastavuje aplikaci do výchozího nastavení po ukončení systému.

## 9.1 Základní webová aplikace a zobrazení rozhraní

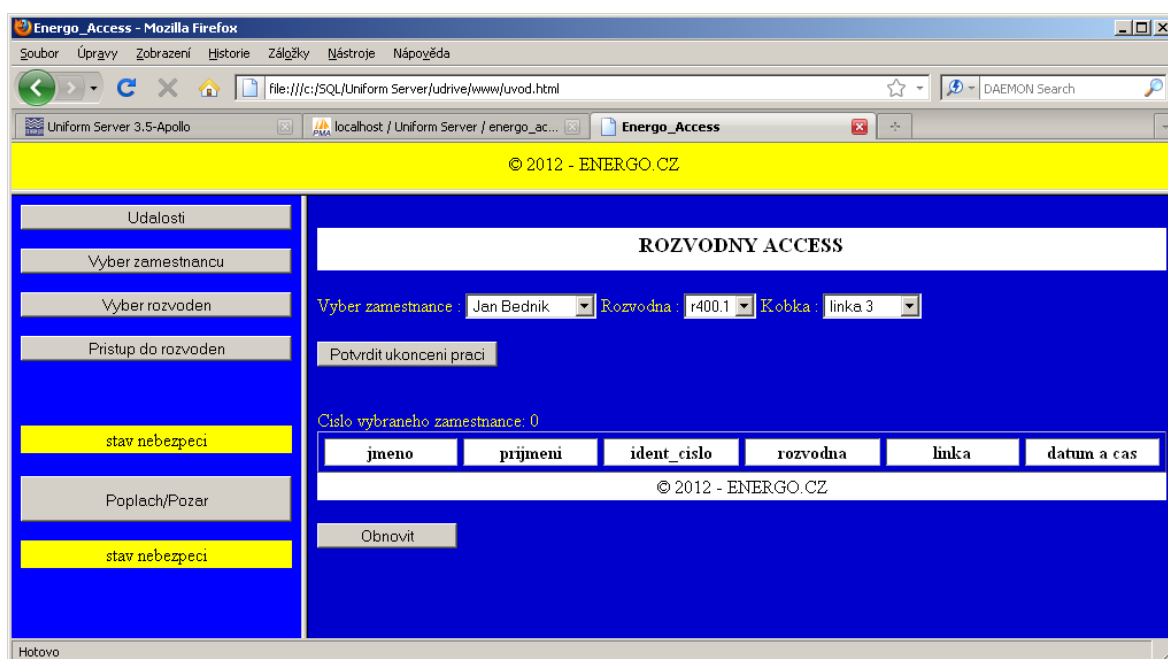
Webové rozhraní aplikace Access\_Energo je složeno z 9ti souborů napsaných v HTML kódu. Seznam těchto souborů je vypsán ve výše uvedeném seznamu v kapitole 9. Okno prohlížeče bylo rozděleno do 3 rámců, kde v každé části je zobrazen jiný dokument. Možnost zobrazit na jedné stránce více HTML dokumentů umožní umístit navigační menu a hlavní okno událostí vedle sebe tak, aby se tyto okna vzájemně nepřepisovaly a ovládání

aplikace bylo tak jednodušší a přehlednější. Struktura stránky s rámy se liší od obvyklých stránek, protože zde je definice těla `<body>` nahrazeno definicí rámu mezi značky `<frameset>` a `</frameset>`. Tato značka obsahuje řadu atributů, z nichž nejdůležitější jsou `cols` a `rows`, které udávají velikost jednotlivých sloupců a řádků. Rozdělení okna prohlížeče v aplikaci `Access_Energo` je vypsáno v následujícím skriptu. Tento skript je uložen v souboru `uvod.html` a celý soubor je uložen v příloze této diplomové práce.

Vytvoření stránky s rámy:

```
<frameset rows="40, *" >
  <frame src="baner.html" name="reklama" frameborder="1" scrolling="no" noresize>
  <frameset cols="25%,*" >
    <frame src="menu.html" frameborder="1" name="menu" scrolling="Auto">
    <frame src="Udalosti.html" frameborder="1" name="HlavniRam" scrolling="Auto">
  </frameset>
```

Pokud si zobrazíme skript na webovém prohlížeči společně se zadanými daty, dostaneme níže uvedenou hlavní stránku aplikace, ze které vychází další zpracování projektu. První rám je pouze informativní banner s podkladovou žlutou barvou a názvem aplikace. Druhý rám obsahuje ovládací tlačítka s možností výběru dalších oken, které se zobrazí v pravé části okna, aniž by došlo k přepsání levé části okna s rámem pro volbu aplikace. Tedy, levý rám je neměnný a data v pravém rámu se mění s tím jakou volbu nebo žádost jsme navolili.

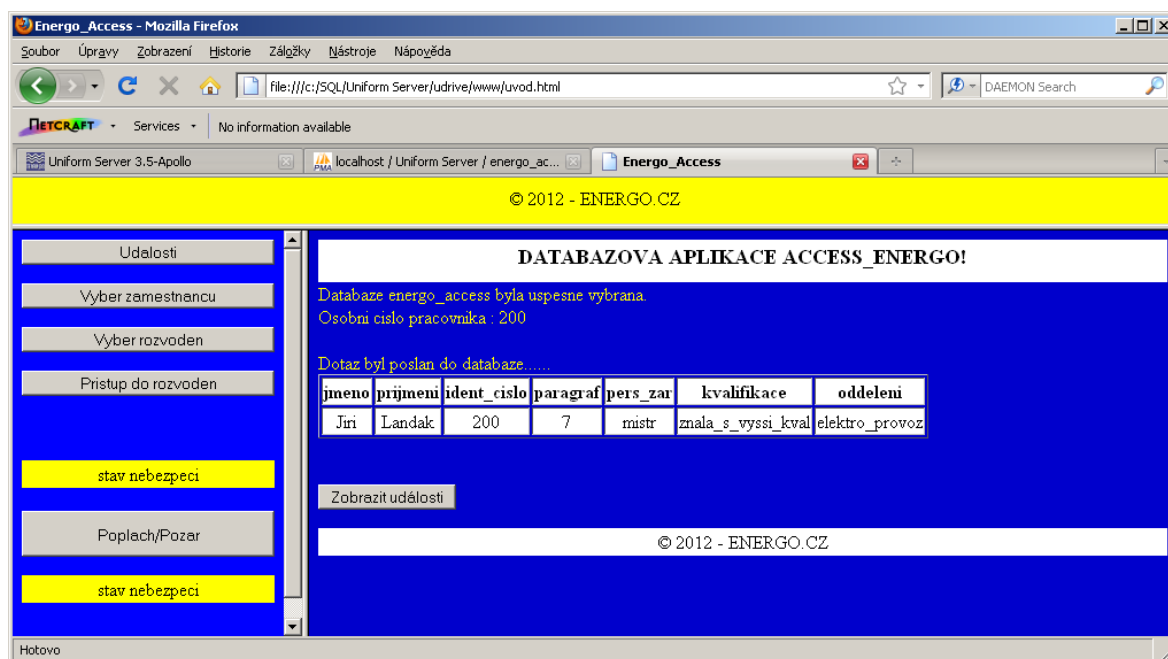


Obr. 20 Základní webové okno aplikace

Levý rám obsahuje tlačítka Události, které umožňuje v pravém okně prohlížet navolené žádosti vstupů do objektů. V okně „Výběr zaměstnanců“ si můžeme pomocí SQL dotazu ověřit kvalifikaci a povolení pracovníků k činnostem na vybraných zařízeních. Tlačítkem „Výběr rozvoden“ se dostaneme do databáze zařízení a zaměstnanců. A nakonec tlačítkem „Přístup do rozvoden“ můžeme navolit přístup do objektů a vložení dat do databáze povolení činností a uložení do vybraných žádostí do hlavní aplikace vytvořenou v Borland Builder psané jazykem C++. Webové okno je také vybaveno tlačítkem „Poplach“, které nám okamžitě uvolní všechny dveře, kobky a vývody bez vlivu na identifikaci a běh celé aplikace podle zvolené databáze přístupu pracovníků.

### 9.1.1 Zobrazení stránek po výběru tlačítkem – Výběr zaměstnanců

Pokud zvolíme tlačítko „Výběr zaměstnanců“ objeví se nám ovládací prvek pro výběr ze seznamu. V seznamu jsou uvedeni všichni pracovníci uložení v databázi a po výběru daného pracovníka se můžeme dozvědět všechny informace, které o daném pracovníkovi potřebujeme znát, abychom mu povolili práci na zařízení, pokud jeho kvalifikace vyhovuje daným činnostem nebo naopak mu pracovní činnosti neumožnit tím, že mu dispečer nepovolí vstup do daného zařízení.



Obr. 21 Okno s výpisem identifikace a kvalifikace pracovníka

Dotaz na databázový server je následující:

```
$id_vysledku = mysql_query("SELECT zamestnanci.jmeno, zamestnanci.prijmeni,  
zamestnanci.ident_cislo, povereni.paragraf, povereni.pers_zarazeni,  
kvalifikace.kvalifikace, oddeleni.oddeleni  
FROM zamestnanci  
inner join kvalifikace on zamestnanci.kvalifikace=kvalifikace.id_kvalifikace  
inner join povereni on zamestnanci.povereni=povereni.id_povereni  
inner join oddeleni on zamestnanci.oddeleni=oddeleni.id_odd  
where ident_cislo=$jmeno",$id_spojeni);
```

Celý skript je uveden v příloze diplomové práce pod názvem PokusEnergo1.php.

### 9.1.2 Zobrazení stránek po výběru tlačítkem – Výběr zaměstnanců

Volbou tlačítka „Výběr rozvoden“ se po výběru seznamu objeví okno s vybraným objektem a soupisem pracovníků, kteří mají povoleno pracovat na zvoleném zařízení. Dispečer si tak může při vypisování příkazu „B“ ověřit jestli pracovník, který se hlásí o práci na zařízení, smí nebo nesmí na zařízení pracovat jako vedoucí práce. Tyto informace jsou načítány z databáze pomocí dotazu na server MySQL, který na dotaz žádosti vrátí do webového okna odpověď.

Tento požadavek do databázového serveru je vnořen do PHP skriptu, který je opět vnořen do základního HTML kódu běžící webové aplikace:

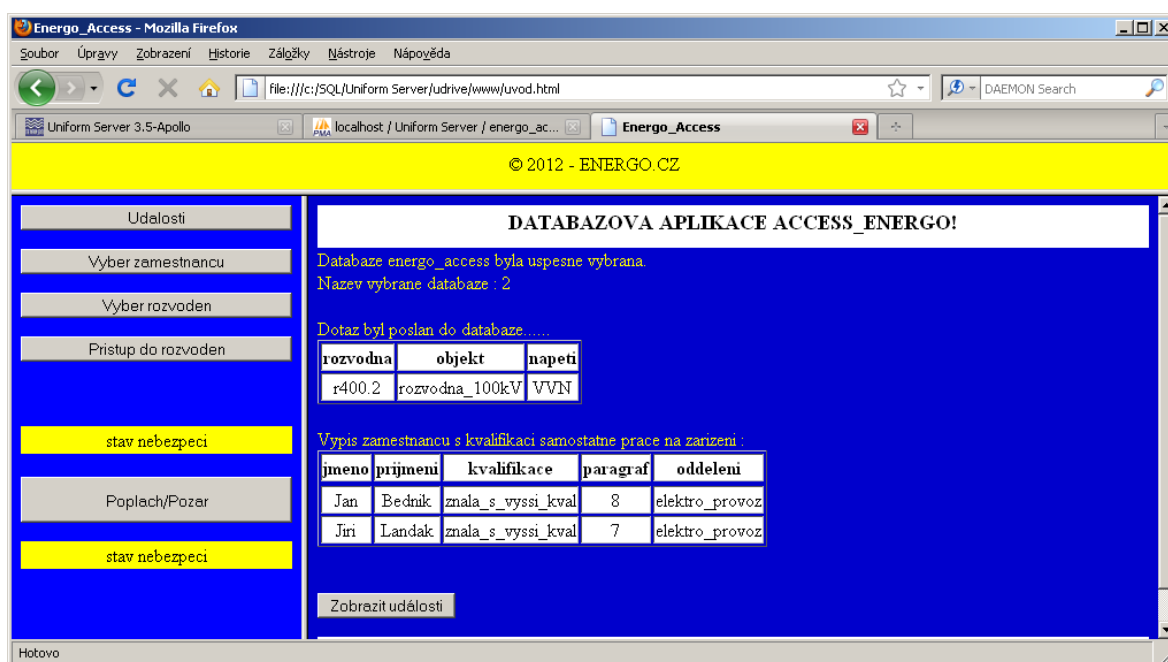
```
<?php  
$id_vysledku = mysql_query("select distinct zamestnanci.jmeno, zamestnanci.prijmeni,  
kvalifikace.kvalifikace, povereni.paragraf, oddeleni.oddeleni  
from access_rozvodny  
inner join zamestnanci on access_rozvodny.zamestnanci=zamestnanci.id_cislo  
inner join rozvodny on access_rozvodny.rozvodny=rozvodny.id_rozvodny  
inner join objekt on rozvodny.objekt=objekt.id_objekt  
inner join kvalifikace on zamestnanci.kvalifikace=kvalifikace.id_kvalifikace  
inner join povereni on zamestnanci.povereni=povereni.id_povereni  
inner join oddeleni on zamestnanci.oddeleni=oddeleni.id_odd  
where rozvodny.napeti=(select rozvodny.napeti from rozvodny where  
id_rozvodny=$rozvodna) and id_rozvodny=$rozvodna",$id_spojeni);  
?>
```



Z následujícího kódu lze vyčíst, že dochází ke vnitřnímu spojení tabulek pomocí instrukce *inner join*. Základní tabulkou je tabulka *access\_rozvodny*, ke které se postupně připojuje tabulka *rozvodny*, objekt a zaměstnanci se svými atributy kvalifikace, oddělení a pověření. Podmínkou pro zařazení pracovníka podle kvalifikace je dáno výběrem atributů s podmínkou velikosti napětí, který se určuje podle prokázané délky praxe pracovníka. O tom o jaký zvolený objekt se jedná je určeno v hodnotě proměnné *\$rozvodna*. Hodnota této proměnné je definována přiřazením hodnoty atributů, které dozorný ukládá do souboru přímo z webového prohlížeče. Celá podmínka má tento tvar:

```
where rozvodny.napeti=(select rozvodny.napeti from rozvodny
where id_rozvodny=$rozvodna) and id_rozvodny=$rozvodna",$id_spojeni)
```

Celý kód je uveden v souboru *rozvodny.php*, který je celý uveden v příloze diplomové práce a webové okno aplikace vypadá následovně:



Obr. 22 Okno s výpisem rozvodny a pracovníků pro práci na zařízení

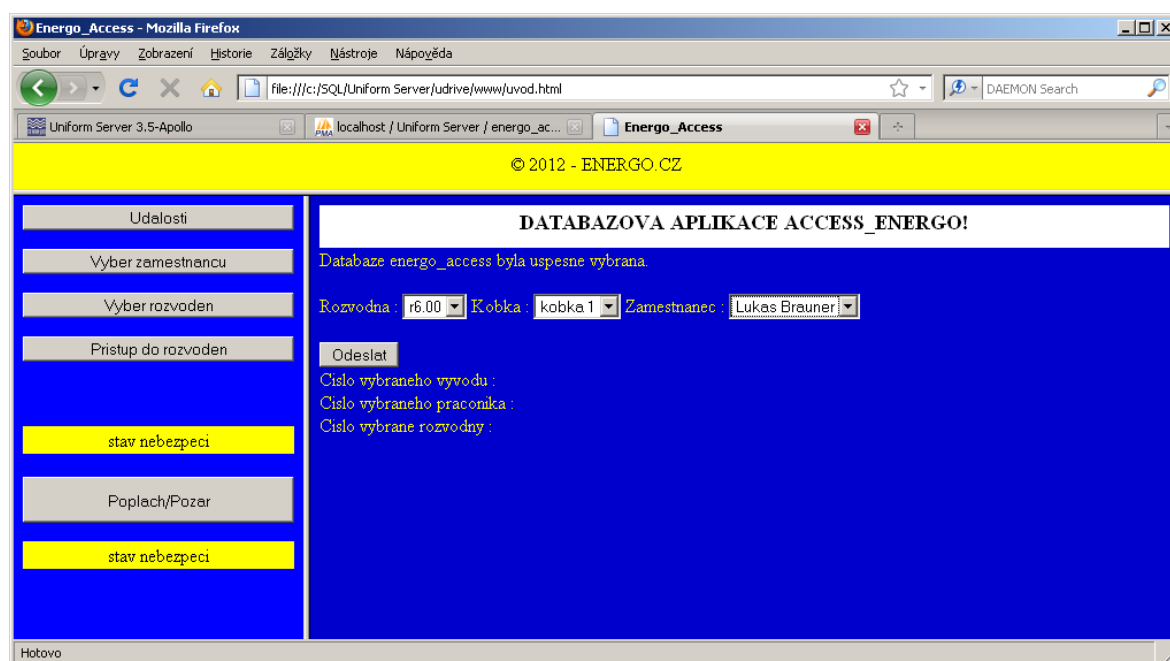
### 9.1.3 Zobrazení stránek po výběru tlačítkem – Přístup do rozvoden

Rám této zobrazené webové stránky obsahuje tři seznamy pro výběr zaměstnance, objektu a kobky, kterým volíme povolení prací na zařízení. Data z tohoto skriptu jsou odeslána do hlavní aplikace *Access\_ComPort\_Modbus*, které jsou ukládány do ADT seznamu. V následujících tabulkách jsou znázorněna přenášená data do hlavní aplikace *Access\_ComPort\_Modbus* vytvořené v prostředí Borland Builder napsané jazykem C++.

ID číslo	Jméno	Příjmení
100	Jan	Bedník
200	Jiří	Landák
300	Lukáš	Brauner
400	Jan	Pekárek
500	Josef	Koler

*Tab. 32 Unikátní identifikační  
číslo pracovníka*

Každý zaměstnanec má své unikátní identifikační číslo, které je přiřazeno aplikaci z okna webové stránky aktivací tlačítka „Přístup do rozvodu“. Volba zaměstnanců pro přístup do objektů je přiřazena ze seznamu znázorněným na následujícím obrázku.



*Obr. 23 Volba zaměstnanců, rozvodny a kobky pro povolení pracovních činností*

Každé navolené kobce odpovídá její identifikační číslo znázorněné v tabulce 25 z kapitoly 8.2.2. Pro 5 pracovníků a 12 vývodů vystačíme se 7mi tabulkami. Tyto tabulky jsou závislé na povolení přístupu a kvalifikaci vybraného zaměstnance. V kombinační tabulce již volíme povolení vstupů do objektů.

ID číslo	Rozvodna	Linka
100	1	1
200	1	1
100	1	2
200	1	2
100	1	3
200	1	3

Tab. 33 Tabulka přístupů

do rozvodny r400.1

ID číslo	Rozvodna	Linka
100	2	4
200	2	4
100	2	5
200	2	5

Tab. 34 Tabulka přístupů

do rozvodny r400.2

ID číslo	Rozvodna	Linka
100	4	6
200	4	6
300	4	6
100	4	7
200	4	7
300	4	7

Tab. 35 Tabulka přístupů

do rozvodny r6.00

ID číslo	Rozvodna	Linka
100	14	8
200	14	8
300	14	8
400	14	8
100	14	9
200	14	9
300	14	9
400	14	9

Tab. 36 Tabulka přístupů

do rozvodny r4.01

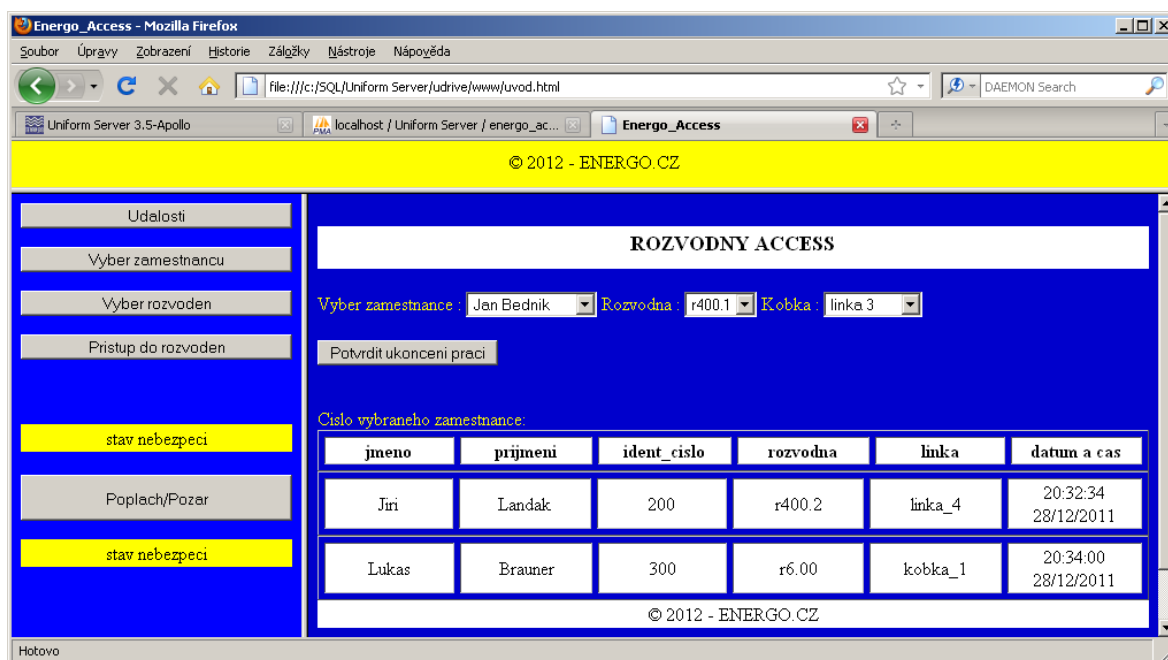
ID číslo	Rozvodna	Linka
100	15	10
200	15	10
300	15	10
400	15	10
100	15	11
200	15	11
300	15	11
400	15	11

*Tab. 37 Tabulka přístupů  
do rozvodny r4.02*

ID číslo	Rozvodna	Linka
100	24	12
200	24	12
300	24	12
400	24	12
500	24	12

*Tab. 38 Tabulka přístupů  
do rozvodny r2.0*

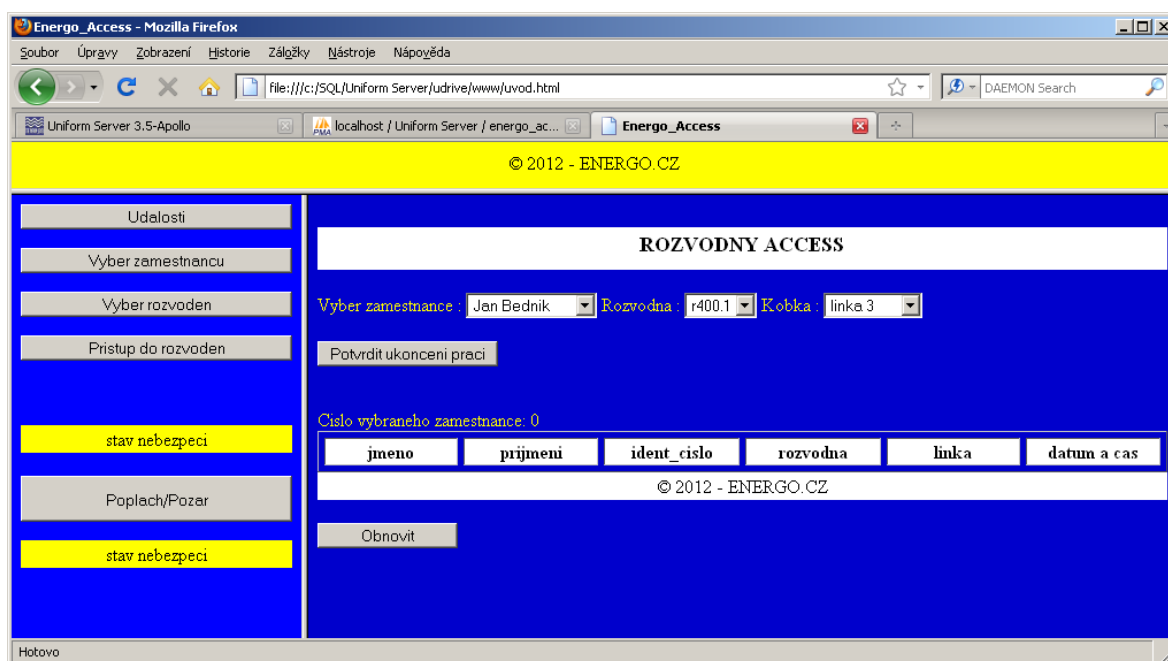
Po zvolení identifikace pracovníka z důvodu povolení vstupu pro pracovní činnosti se po stisknutí tlačítka „odeslat“ přenesou hodnota do datového nebo textového souboru.



*Obr. 24 Zobrazení událostí pracovních činností*

Tento soubor načítá pro další zpracování hlavní aplikace Access\_ComPort\_Modbus pro uložení žádosti do ADT seznamu (hash datové tabulky). Metoda „výpis“ předává aplikaci platnou tabulku s daty do PHP skriptu udalosti\_roz.php. Tento soubor načítá hodnoty proměnných *\$pracovnik*, *\$kobka* a *\$linka*. Hodnoty proměnných jsou následně zpracovány pro výpis z databáze na monitor uživatele (v našem případě dispečera).

Tlačítkem „Potvrdit ukončení prací“ se stejným postupem zruší povolení prací na zařízení. Tím je proces ukončen a je možná nová volba pro činnosti na zařízení. Celá aplikace psaná v PHP skriptu je velmi jednoduchá. V podstatě zde dochází pouze k předávání trojice čísel, z nichž první je číslo *id\_zaměstnance*, druhé *id\_kobky* a třetí znamená hodnotu *id\_vývodu*. Tyto tři přenášené čísla jsou základem celé aplikace a jejich hodnoty se ukládají do hashování tabulky (ADT seznam) v hlavní aplikaci pro vytvoření relace v databázi. Vzájemná vazba všech aplikací je znázorněna v kapitole 10.



Obr. 25 Zobrazení okna událostí po zrušení všech pracovních povolení

## 10 HLAVNÍ APLIKACE ACCESS\_COMPORT\_MODBUS

Aplikace Access\_Comport\_Modbus je program napsaný v jazyce C++, který obsahuje nejdůležitější funkce pro projekt ochrany pracovníků v energetice. Tato aplikace obsahuje tyto funkce:

- Obsluhu sériového portu RS232 a řízení toku dat mezi hlavní aplikací a aplikacemi naprogramovanými v PLC, který je tvořen samostatným vláknem
- Algoritmus kontrolního součtu CRC pro komunikaci zařízení po MODBUS RTU
- ADT seznam pro porovnání dotazů z databáze MySQL
- Vláknem, které obsahuje kód a dotaz v MODBUS žádosti pro načítání dat z registrů v PLC, kde jsou uloženy v zásobníku identifikace přiložených karet z čteček.
- Rozhraní pro předávání dat pro webovou aplikaci vytvořenou HTML skriptem a jazykem PHP, která slouží jako základní rozhraní mezi obsluhou a hlavní aplikací.

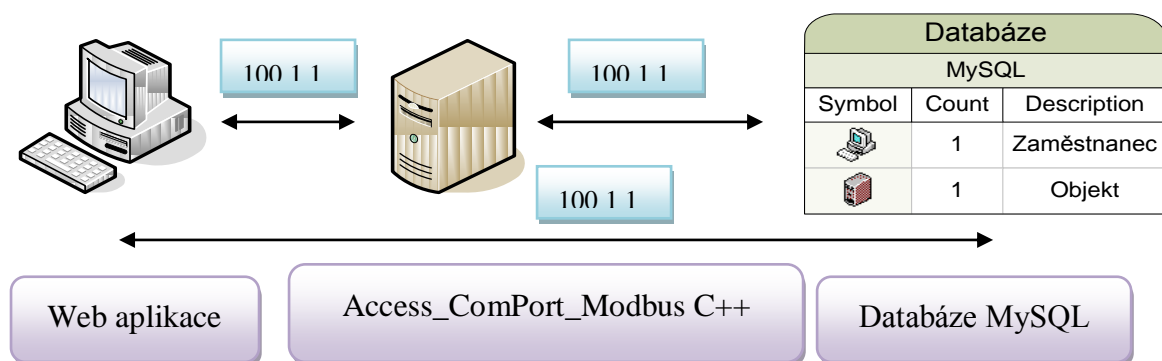
Prvotním úkolem pro zpracování projektu je komunikace dvou zařízení na sériové lince RS232. Protože aplikace je programována v prostředí Windows, bylo využito již definovaných API funkcí, které obsahuje každý operační systém postavený na základě OS Windows. Použité API funkce v projektu Access\_Energo jsou vysvětleny v kapitole 10.2.1.

Pro vysílání dat do PLC přes sériové rozhraní COM1 je vytvořena metoda, která na sériový port vysílá pole dat s požadavkem kódu v podobě MODBUS protokolu. Pro načítání informací z PLC je vytvořeno vlastní vlákno Unit2, které si žije svým životem a je přerušeno pouze při vysílání požadavku nebo při práci procesoru. Toto vlákno sleduje dění na sériovém portu a přijímá jakékoliv příchozí data, které se objeví na sériovém portu COM1. K ukládání požadovaných dat, na které aplikace čeká je zpracováno samostatnou metodou v hlavním procesu programu. Popis této metody je vysvětlena v následující kapitole 10.2. Jestliže není vysílán žádný dotaz nebo pokud je procesor v nečinnosti vlákno neustále načítá případné příchozí data s PLC a obnovuje zásobník s přijatými daty. Kód tohoto vlákna tyto data porovnává, a pokud jsou shodná poslední příchozí data s daty předcházejícími, tak poslední příchozí informace z registrů PLC se neuloží do zásobníku. Vše je prováděno pomocí dynamicky alokované paměti. Také tento algoritmus a kód je vysvětlen v následujících podkapitolách.

Dalším vláknem, které si žije svým vlastním životem je vlákno Unit1 s metodou na vysílání požadavku. Také chod tohoto vlákna se přerušuje pouze v případě činnosti procesoru nebo při činnosti procesu hlavního programu. Po ukončení žádaného procesu se vlákno opět samostatně spustí. Vlákno tedy není ukončováno hlavními procesy programu, ale pouze na dobu procesů přerušováno do doby, nežli je opět procesor v nečinnosti nebo není prováděný žádný jiný proces ze strany uživatele do PLC. [9]

Aby bylo možné uvolnit přístup pracovníkům na pracoviště a uvolnit předem zvolenou kobku, vývod nebo přívod, tak je nutné někde do hlavního programu ukládat požadavky obsluhy z databáze. To je provedeno uložením identifikačního čísla pracovníka, identifikačních čísel rozvodny a vývodu do indexované tabulky. Každému uloženému záznamu je přiřazen vlastní index, který se vypočítává daným algoritmem. Tato hash tabulka je společně s lineárním seznamem základem pro ukládání datových struktur. Pro aplikaci Access\_Energo je ADT seznam výhodnější nežli vytvoření obsáhlého lineárního seznamu, protože všechny data jsou již uloženy v databázi MySQL. Vystačíme pouze s poli čísel a vyhneme se práci se strukturami různých datových typů proměnných. Metoda pro práci s tabulkou je vytvořena tak, aby se porovnávaly pole dat, které přicházejí na sériové rozhraní z PLC. Metoda je napsána tak, aby pracovník dle vyhlášky 50/1978 a příkazu „B“ mohl pracovat pouze na jednom zařízení, pro které byl příkaz „B“ vydán. Z pohledu programu to znamená, že bylo nutné porovnávat data příchozí s těmi uloženými v ADT seznamu a uložení do databáze povoleno smí dojít pouze tehdy, nejsou-li data s identifikací pracovníka již uloženy v databázi přístupů. Tedy pokud již pracovník na zařízení pracuje, tak mu není umožněn vstup do jiné kobky. Tímto je docílené maximální snížení rizika vstupu do zařízení, které je pod napětím a na kterém nejsou práce povoleny. Metoda porovnání dat  $tUzel * search(tHashTabulka\ t, char * e)$  je vysvětlena v jedné z dalších podkapitol této kapitoly. Pokud nejsou data po porovnávání již uloženy v hash tabulce, dochází jejich vložení do ADT seznamu pomocí metody  $vloz(tHashTabulka\ t, tData\ d)$ . Do tabulky je umístěna stejně jako v případě webového rozhraní pomocí PHP skriptu trojice čísel s identifikací pracovníka, rozvodny a kobky, na které se pracuje. Co tyto čísla znamenají a co definují má na starosti databázová aplikace MySQL, která tyto čísla převede na definované atributy databáze. Zjednodušeně, každé číslo v daném pořadí patří pracovníkovi, objektu a zařízení, na kterém se pracuje. V kapitole 8 jsou vypsané všechny tabulky databáze Access\_Energo a atributy jejich přiřazení. Všechny tři základní aplikace (MySQL, PHP + HTML a v Borland Builder C++ vytvořený ADT seznam pro

komunikaci s PLC) si předávají v podstatě pouze trojici čísel. Například pokud z webové stránky, která je základním rozhraním mezi obsluhou a hlavním programem Access\_ComPort\_Modbus vydá dispečer požadavek na práce pro Jana Bedníka pro práci v rozvodně r400.1 na lince 3, tak vyšle trojici čísel 100, 1 a 1 do hlavního programu Access\_ComPort\_Modbus a tyto data jsou porovnávána a ukládána do hash tabulky. Pro lepší pochopení je celý postup přenášení, porovnávání a ukládání identifikačních dat zobrazen v níže zobrazeném obrázku.



Obr. 26 Předávání identifikačních atributů mezi aplikacemi

## 10.1 Formulář základní aplikace a jeho popis

Na obrázku je zobrazeno hlavní formulářové okno aplikace Access\_ComPort\_Modbus.

The screenshot shows the main window of the Access\_ComPort\_Modbus v3.88 application. The window is divided into several sections:

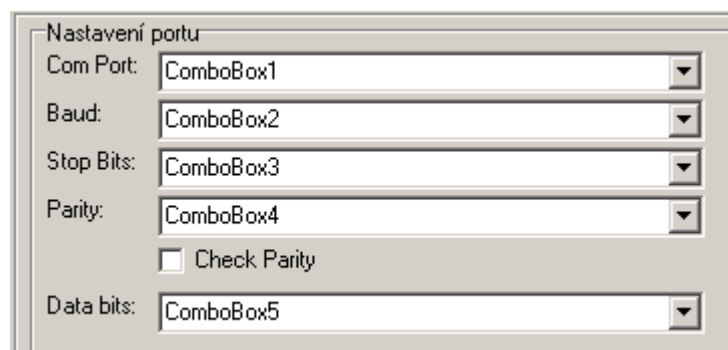
- Nastavení portu (Port settings):** Includes fields for Com Port (ComboBox1), Baud (ComboBox2), Stop Bits (ComboBox3), Parity (ComboBox4), and Data bits (ComboBox5). There is a checkbox for 'Check Parity'.
- Komunikace (Communication):** Includes a text field 'Edit1', checkboxes for 'Poslat jako znak' and 'Poslat jako číslo', and buttons 'Odeslat' and 'Reset'. Below are two memo fields: 'Memo1' and 'Memo5'.
- Modbus:** Includes fields for Slave (Edit2, Edit3), Kód (Edit4, Edit5), Startovací adresa registru (Hex Edit6, Dec Edit7), Počet, data (Hex Edit8, Dec Edit9), Data bytes (Hex Edit10, Dec Edit11), Zap/Vyp - Horní byte (Hex Edit22, Dec Edit23), and Zap/Vyp - Dolní byte (Hex Edit24, Dec Edit26).
- Přenos dat (Data transfer):** Includes fields for Přenášená data - decimálně (Memo3) and Kontrolní součet - decimálně (Memo4). Below are fields for Čas (Memo2) and Datum (Memo7), and a section for 'Načtení dat ze souborů' (Load data from files) with buttons for Zaměstnanci, objekt, Linka, CRC, Hash, and RFID. There are also fields for Rozvodny (Edit12, Edit13, Edit14, Edit25, Edit21) and Memo6.

At the bottom, there are buttons for 'Připojit', 'Odpojit', 'Události - Start', and 'Ukončit'.

Obr. 27 Formulářové okno hlavní aplikace

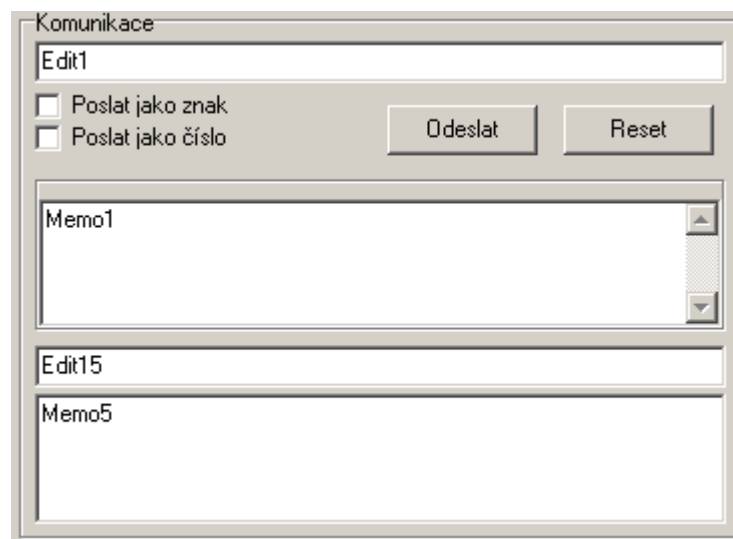


Hlavní aplikace Access\_Energo je rozdělena do tří základních oken GROUPBOX. První okno je součástí funkcí pro komunikaci po sériové lince, kde jsou rozbalovací boxy s možnostmi pro výběr druhu a rychlosti po sériové lince.



*Obr. 28 Groupbox s rozbalovacími seznamy pro nastavení komunikace*

V komponentách ComboBox je možnost k výběru COM portu, rychlostí přenosu, nastavením stop bitů, nastavením parity a nastavením datových bitů. V této komponentě je také možno nastavit kontrolu parity přenášené informace.



*Obr. 29 GroupBox Komunikace pro sledování zpráv*

Komunikační okno obsahuje komponenty Edit1 a Edit15 pro zobrazování nebo zadávání údajů. Edit1 představuje odesílaný MODBUS požadavek s přidáním kontrolním výpočtem a Edit15 zobrazuje již zpracovaný identifikační kód karty v binární podobě, který je základem pro další zpracování informace pro porovnání a uložení dat do ADT seznamu.

Uložené informace se ukládají do hash tabulky, která je zobrazena v komponentě. Na rozdíl od komponenty Edit dokáže komponenta Memo obsáhnout více řádků textu. V komponentě Memo1 jsou zobrazovány data načtená pomocí sériové linky COM. Tedy, zde se zobrazí vše, co se objeví na vstupu sériového rozhraní RS232, proto je potřeba tyto data porovnávat, vybírat a nepotřebné informace odhazovat. Tuto selekci dat provádí funkce `nacitani_karty()` v hlavní funkci programu `Access_ComPort_Modbus`. GroupBox „Komunikace“ obsahuje také tlačítka Odeslat a Reset. Tyto tlačítka jsou však pouze pro možnost ladění programu nebo pro odzkoušení samotné komunikace popřípadě kontroly funkce algoritmu pro porovnání dat s karty a jejímu následnému použití v programu. Komponenta CheckBox s podržítka „Poslat jako znak“ nebo „Číslo“ je také pouze pro ladění programu a umožňuje zobrazit informaci buď jako číselnou hodnotu nebo soubor ASCII znaků. Komponenta Memo5 zaznamenává příslušnou trojici identifikačních čísel (např. 100,1,1), které čekají na porovnání s údaji s `id_karty`.

The image shows a Windows-style dialog box titled "Modbus". It contains the following components:

- Slave:** Two edit boxes labeled "Edit2" and "Edit3".
- Kód:** Two edit boxes labeled "Edit4" and "Edit5".
- Startovací adresa registru:** Two edit boxes labeled "Edit6" and "Edit7", with "Hex" and "Dec" radio buttons.
- Počet, data:** Two edit boxes labeled "Edit8" and "Edit9", with "Hex" and "Dec" radio buttons.
- Data bytes:** Two edit boxes labeled "Edit10" and "Edit11", with "Hex" and "Dec" radio buttons.
- Zap/Vyp - Horní byte:** Two edit boxes labeled "Edit22" and "Edit23", with "Hex" and "Dec" radio buttons.
- Zap/Vyp - Dolní byte:** Two edit boxes labeled "Edit24" and "Edit26", with "Hex" and "Dec" radio buttons.
- Přenášená data - decimálně:** A memo field labeled "Memo3".
- Kontrolní součet - decimálně:** A memo field labeled "Memo4".

Obr. 30 Komponenta GroupBox Modbus

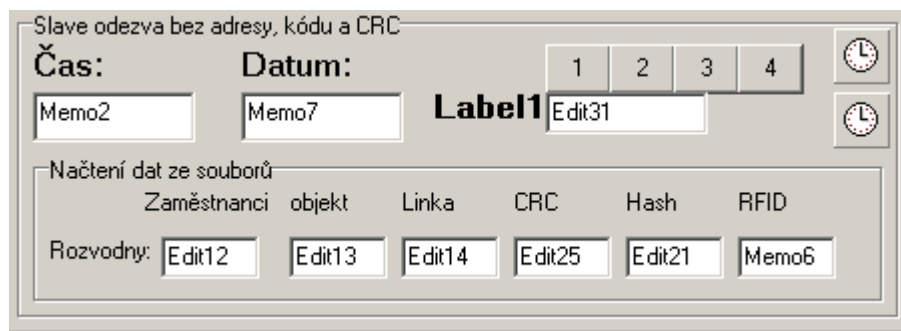
GroupBox Modbus obsahuje 14 editačních oken pro vizualizaci příkazového kódu žádosti do PLC, které na tyto žádosti odpovídá podle zvyklostí komunikačního protokolu MODBUS. Hlavní komponenta GroupBox Modbus obsahuje tyto komponenty:

- Slave – je složen s komponent Edit2 a Edit3. Do Edit3 je vloženo decimální číslo s požadavkem na obsluhu zařízení, v našem případě se jedná o slave jednotku PLC, která je adresována jako Slave1. Edit2 je převedený ten samý požadavek z decimální do hexadecimální soustavy. Veškerá komunikace probíhá

v hexadecimálních číslech a princip přenosu žádostí je vysvětlen v kapitole 4 teoretické části diplomové práce.

- Kód - je složen s komponent Edit5 a Edit4. Do komponenty Edit5 je vložen požadavek kódu, kdy každý kód plní vlastní požadavek na slave zařízení ( PLC, čtečka karet s MODBUS převodníkem apod.). Instrukce jsou zobrazeny v tabulce 7. Pro komunikaci je dotaz kódu opět převeden na hexadecimální číslo v Edit4.
- Startovací adresa registru – je složena z komponent Edit6 a Edit7. Do komponenty Edit7 je poslán požadavek na startovací adresu registru Slave zařízení (PLC), ze které chceme data načítat nebo ukládat. Tato žádost je převedena z decimálního do hexadecimálního čísla v komponentě Edit6.
- Počet, Data – (Edit8, Edit9) Obsahuje počet registrů, které požadujeme přečíst nebo zapsat. Žádost události je vysílána jako hexadecimální číslo z komponenty Edit8.
- Data bytes – (Edit10, Edit11) Zděluje Slave zařízení jakou velikost registrů musí obhospodařit. Jedná-li se o registr 2 nebo 4 bajtový.
- Zap/Vyp horní byte (dolní byte). Pro snadnější pochopení funkce uvedu tento příklad. Ve dvojkové soustavě a formátu unsigned short má číslo 65535 v paměti podobu šestnácti jedniček: 1111 1111 1111 1111. Šestnáct jedniček se na jednu adresu nevejde – na jednu adresu je možné deklarovat pouze jeden bajt, osm bitů. Tím se číslo 65535 musí rozložit na dvě adresy. Toto rozdělené číslo už můžeme uložit. Tedy v jednom registru bude uloženo osm jedniček a ve druhém (nižším bajtu) také osm jedniček. Tím je číslo 65535 rozděleno na horní a dolní bajt (255 a 255 decimálně). Pro sepnutí požadované cívky je v horním bajtu vyslán požadavek 0xFFFF a v dolním bajtu 0x0000. Pokud budeme chtít sepnutou cívku rozepnout, tak jsou oba bajty shodné 0x0000.
- Přenášená data decimálně zobrazují požadavek kódu v decimální soustavě, který je rozdělen do části funkčního kódu s požadavkem a s následně vypočítaným CRC kontrolním součtem.

Poslední komponenta slouží k ladění programu a sledování změn procesu. Do komponenty Memo2 je uložen čas poslední události, respektive uložené datum do komponenty Memo7. Tyto dvě události jsou přeneseny do databázové aplikace MySQL, která jsou přiřazeny k danému pracovníkovi, který tuto událost vyvolal přiložením karty ke čtečce.

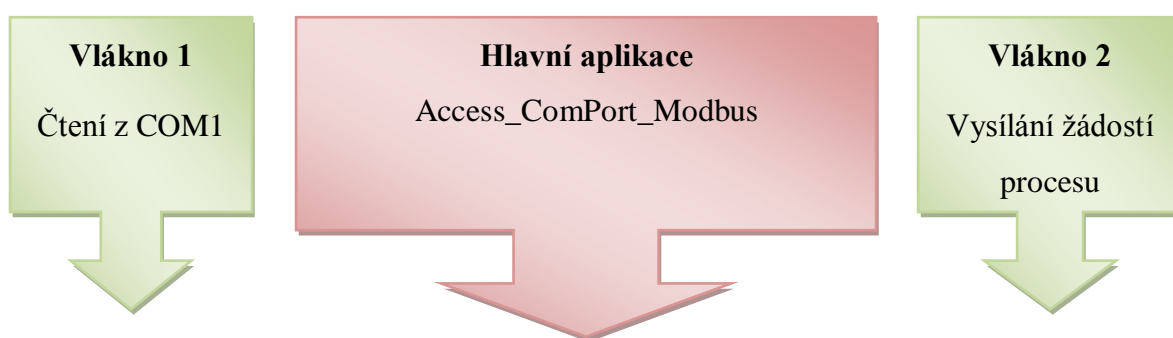


Obr. 31 Odezva adresy, kódu a CRC

Komponenta Label1 zobrazuje obsluhu, zdali je sériový port otevřen nebo uzavřen. Okno „Načtení dat ze souborů“ také slouží ke kontrole chodu programu a k ladění spolupracujících aplikací celkového systému (MySQL, PHP, Builder C++). [10]

## 10.2 Popis funkcí hlavní aplikace Access\_ComPort\_Modbus

V úvodní kapitole v odrážkovém seznamu byla popsána koncepce a hlavní aplikace. Tato aplikace se skládá se dvou vláken a hlavního programu, který je napsán objektově s využitím funkcí, které si aplikace dle požadavku žádaných procesů žádá. Použití vláken pro čtení dat, které se objeví na sériovém portu a pro vysílání požadavku stavů registrů v PLC bylo zvoleno z důvodu, že při použití více vláken v rámci jednoho procesu (programu) běží více výpočtů naráz, takže je možné využít více jader procesoru nebo oddělit dlouho trvající operace (např. se soubory) od uživatelského rozhraní. [11]



Obr. 32 Zobrazení vláken a hlavní aplikace

### 10.2.1 Vlákno pro čtení dat z COM1

Toto vlákno slouží k práci s jakýmkoliv daty, které se objeví na vstupním bufferu sériového rozhraní. Vlákno tyto data stahuje a ukládá do vyrovnávacího bufferu typu

řetězec InBuff. Vlákno využívá standardní API funkce OS Windows. Základní API funkce pro načítání dat ze sériového portu:[12]

*ReadFile (hPort, &Byte, 1, &dwBytesTransferred, NULL );*

Přicházející data jsou zpracovány postupně bajt po bajtu a ukládány do znakového řetězce InBuff, který je výsledně zobrazován v komponentě Memo1. Protože hlavní aplikace vysílá požadavek do slave zařízení PLC s dotazem na požadovaný registr, kde jsou zaneseny identifikační data ze čtečky, je možné tyto příchozí data naformátovat do požadovaného tvaru k dalšímu zpracování a porovnání s databází přístupových povolení ke vstupu do objektu. Funkce pro načtení bajtů na sériovém COM1 portu je velmi jednoduchá:

*ReadFile(hComm, InBuff, 1, &dwBytesRead, NULL);*

Funkce ReadFile obsahuje pět atributů:

- hCom – číslo portu, ze kterého se načítají data
- InBuff – řetězec do kterého se data zapisují
- 1 – počet čtených bajtů
- &dwBytesRead – ukazatel na počet zapisovaných bajtů
- NULL – musí být deklarováno pro Windows CE

### 10.2.2 Vlákno s dotazem žádostí čtení registrů z PLC

Toto vlákno nese pouze kód pro žádost čtení registrů z PLC, kde jsou uložena data z RFID karet. Jeho vytvoření není složité, avšak u tohoto vlákna je důležité jeho přerušení a opětovné obnovení vlastní funkce. Přerušit chod vlákna je možné pouze tehdy, je-li v činnosti hlavní proces (požadavek na ovládání cívky apod.) nebo pokud se objeví nějaká příchozí data na COM1, které je nutné přečíst a zpracovat. K tomu slouží funkce pro synchronizaci vlákna, aby nedocházelo ke konfliktům při činnostech obou vláken. Vlákno s dotazem žádostí čtení registrů z PLC má tpIdle prioritu a provádí svojí činnost pouze tehdy, je-li celý systém v nečinnosti. Protože není potřeba rychlého obslužení registrů z PLC, tak není ani nezbytné tomuto vláknu přiřazovat vysokou prioritu. [11]

Kód pro načtení registru R3 z PLC s uloženým ID karty:

```
{  
Form1->Edit3->Text="1";  
Form1->Edit5->Text="3";  
Form1->Edit7->Text="4";  
Form1->Edit9->Text="1";  
Form1->Edit11->Text="00";  
Form1->Edit23->Text="00";  
Form1->Edit26->Text="00";  
}
```

### 10.2.3 Hlavní aplikace Access\_ComPort\_Modbus

Hlavní aplikace obsahuje několik funkcí, které jsou volány, a které se starají o správný chod programu. Jde o tyto funkce:

- void ScanForComPorts() – skenuje volné sériové porty, které je možné použít v aplikaci.
- void BaudRate() – umožňuje nastavit přenosovou rychlost
- void StopBits() – nastavuje velikost stopbitu
- void Parity () – umožní nastavení parity přenosu dat
- void DatBits() – slouží pro nastavení bitového rámce
- bool isNumber(char) – funkce pro kontrolu integrity dat, jedná-li se o číslo nebo řetězec
- unsigned int CRC16(unsigned int crc, unsigned int byte) – funkce vypočítává kontrolní součet CRC.
- void Send() – odesílá data na sériový port
- \_\_fastcall TForm1::TForm1(TComponent\* Owner): TForm(Owner) – metoda pro vytvoření vláken.
- void \_\_fastcall TForm1::FormCreate(TObject \*Sender) – metoda definující proměnné při spuštění aplikace
- void \_\_fastcall TForm1::Button3Click(TObject \*Sender) – metoda pro otevření zvoleného COM portu
- void \_\_fastcall TForm1::Button4Click(TObject \*Sender) – metoda pro uzavření portu a vláken

- void \_\_fastcall TForm1::Button2Click(TObject \*Sender) – metoda pro resetování textu v Memo1
- void \_\_fastcall TForm1::Button1Click(TObject \*Sender) – metoda pro okamžité odeslání MODBUS požadavku do PLC
- void \_\_fastcall TForm1::Button5Click(TObject \*Sender) – metoda pro celkové ukončení aplikace
- void \_\_fastcall TForm1::Edit1KeyPress(TObject \*Sender, char &Key) – metoda reagující na příchozí znaky v záznamovém okně Memo1 z klávesnice.
- void \_\_fastcall TForm1::FormClose(TObject \*Sender, TCloseAction &Action) – volaná metoda přerušuje činnost vláken
- void \_\_fastcall TForm1::Edit3Change(TObject \*Sender) až Edit9 – slouží k převodu decimálního čísla na hexadecimální
- void \_\_fastcall TForm1::Button8Click(TObject \*Sender) – metoda obsahující funkce pro načítání proměnných, kontrolního součtu a odesílání polí dat na sériový port. Obsahuje důležitou API funkci WriteFile. Tato funkce má stejné proměnné jako ReadFile, avšak rozdílný je tok dat. Obecný tvar funkce WriteFile:  
*WriteFile (hPort, &Byte, 1, &dwNumBytesWritten, NULL );*
- int HexToInt (AnsiString Value) – převádí hexadecimální číslo na decimální
- void \_\_fastcall TForm1::Button10Click(TObject \*Sender) – metoda standardně ukončuje celou aplikaci včetně vláken
- void \_\_fastcall TForm1::Button11Click(TObject \*Sender) – metoda slouží k načítání dat z webového rozhraní obsluhy aplikace
- void porovnani(int&,int&, int, int, int) – funkce slouží k porovnávání příchozích požadavků z webové aplikace
- void hashovani\_cs(int,int,int, int) – funkce pro zpracování požadavků přístupu do příslušných textových řetězců
- int hash(char \*) – funkce pro výpočet indexu v ADT seznamu
- tUzel \* search(tHashTabulka t , char \* e) – funkce pro vyhledávání uložených struktur v ADT seznamu
- void init(tHashTabulka t) – inicializuje ADT seznam
- void vloz(tHashTabulka t, tData d) – funkce pro vložení dat z databáze do ADT seznamu

- void vypis(tHashTabulka t, tData d, char \* e) – provádí výpis atributů ADT seznamu do komponenty Memo1
- void smaz(tHashTabulka t, char \* e) – Smaže atribut v ADT seznamu. Tento atribut je smazán zadáním požadavku z webové aplikace Access\_Energo
- void vypis1(tHashTabulka t, tData d, char \* e) – provádí výpis platné tabulky do databáze SQL
- void wait ( int seconds ) – čekací smyčka
- void otvirani(int vstup) – funkce pro příkazy otvírání dveří pomocí modbus příkazu:

```
Form1->Edit3->Text="1";  
Form1->Edit5->Text="15";  
Form1->Edit7->Text="1";  
Form1->Edit9->Text="1";  
Form1->Edit11->Text="2";  
Form1->Edit23->Text="255";  
Form1->Edit26->Text="0";
```

- void zavirani(int vystup) – funkce pro zavírání dveří. Rozdíl naproti funkce „otvirani“ je v 6 řádku, kdy je hodnota proměnné 0.

```
Form1->Edit3->Text="1";  
Form1->Edit5->Text="15";  
Form1->Edit7->Text="1";  
Form1->Edit9->Text="1";  
Form1->Edit11->Text="2";  
Form1->Edit23->Text="0";  
Form1->Edit26->Text="0";
```

- tUzel \* search1(tHashTabulka t, char \* e) – funkce slouží pro výpočet koeficientu otvírání, kdy pracovník, objekt a zařízení má své jedinečné identifikační číslo
- void otevri\_vse() – poplachová funkce, umožňuje otevřít všechny dveře v objektech
- void zavri\_vse() – funkce po obnovení stavu po poplachu umožní uzavření všech otevřených dveří v objektu
- void prepis\_poplach() – provádí zápis o poplachu do textového řetězce
- void nacistani\_karty() – slouží k načítání id karty z registrů uložených v PLC
- void comparuj(int&novy,int&stary) – provádí porovnání identifikačních údajů karet. Pokud se čísla při porovnání shodují, tak nedochází k uložení údajů pro další zpracování.



Hlavní program aplikace je rozdělen do čtyř samostatných bloků, které mezi sebou na první pohled nesouvisí. Metoda pro práci se sériovým portem otevírá COM1 a kontroluje přenos dat mezi PLC a sériovým portem PC. Okno pro práci z modbus kódem převádí požadavek na proces mezi PLC a PC převodem z decimálního čísla na hexadecimální a okno komunikace zpracovává příchozí data do podoby, se kterými lze již pracovat. Tím je ukončena identifikace a definice příchozích dat. V hlavní aplikaci je obsažena také funkce pro výpočet kontrolního součtu a ověření správnosti komunikace mezi periferními zařízeními, které si mezi sebou vyměňují informace pomocí komunikačního protokolu MODBUS. Algoritmus toho kontrolního výpočtu je uveden v následující podkapitole a celý kód hlavní aplikace je uložen i s vysvětlivkami na CD v příloze této diplomové práce.

#### 10.2.4 Kontrolní součet CRC pro přenos informace po Modbusu

CRC (Cyclic Redundancy Check) – Cyklický redundantní součet je obdoba hašovací funkce, používaná pro detekci chyb při přenosu dat. Kontrolní součet je odesílán spolu s daty a po přijetí znovu nezávisle vypočítán. Pokud se liší vypočtený CRC od přijatého, je zřejmé, že při přenosu došlo k chybě. V závislosti na délce generujícího polynomu existuje vždy nenulová pravděpodobnost, že chyba přenosu nebude odhalena.

Přenášenou datovou posloupnost lze interpretovat jako polynom s binárními koeficienty. Například posloupnost 11000101 lze psát jako  $x^7+x^6+x^2+1$ . CRC se počítá jako zbytek po dělení datové posloupnosti generujícím polynomem. V průběhu výpočtu se na koeficienty uplatňuje operace modulo 2 (zbytek po dělení dvěma ... -2 -> 0, -1 -> 1, 0 -> 0, 3 -> 1 atd.). Určení CRC pouze generujícím polynomem je nejednoznačné, různé algoritmy ho implementují různě - z historických nebo technických důvodů může docházet prohození bajtů, změně pořadí bitů v bajtu, nebo přidání různých bitových posloupností před nebo za vstupní data.

- někdy se před výpočtem před vstupní data přidává jednička.
- někdy se před výpočtem za vstupní data přidává počet nul odpovídající stupni generujícího polynomu. CRC vypočtený z posloupnosti data + odeslaný CRC (bez chyb při přenosu) je potom vždy rovný nule.

Číslo za písmeny CRC znamená obvykle stupeň generujícího polynomu, například CRC5 znamená, že generující polynom má nejvyšší mocninu  $x^5$  a tedy délku odpovídající 6

bitům, zbytek po dělení (tedy vlastní CRC) má pak stupeň maximálně o jedničku nižší. V našem případě tedy  $x^{16}$  a délku odpovídající 17 bitům. [13]

Hlavním důvodem je schopnost cyklických kódů odhalovat shluky chyb (chyby v přenosových kanálech mají tendenci vyskytovat se blízko sebe a vytvářet tzv. shluky). Cyklický kód CRC16-IBM je vytvořen generujícím polynomem  $x^{16} + x^{15} + x^2 + 1$ . Tento generující polynom je charakteristický pro kontrolu přenosu dat po modbusu. Při systematickém kódování je využíváno operace dělení mnohočlenu. Prvním krokem je doplnění posloupnosti informačních bitů o  $n-k$  nulových míst. Toho dosáhneme vynásobením informačního mnohočlenu  $u(x)$  členem  $x^{n-k}$ , čímž získáme prostor pro připojení  $n-k$ , zabezpečujících prvků. Takto upravený informační mnohočlen  $u(x)$  poté vydělíme generujícím mnohočlenem  $g(x)$ . Při dělení je vypočten podíl  $q(x)$ , který nebude nijak využit a zbytek po dělení vyjádřený mnohočlenem  $r(x)$ . Mnohočlen  $r(x)$  může být nejvýše stupně  $n-k-1$ , neboť dělitel je stupně  $n-k$ . Zabezpečené kódové slovo pak představuje mnohočlen  $v(x) = x^{n-k} * u(x) + r(x)$ . Jelikož mnohočlen  $v(x)$  vznikl součtem informačního mnohočlenu se zbytkem po dělení informačního mnohočlenu generujícím mnohočlenem, je zřejmé, že  $v(x)$  je beze zbytku dělitelný generujícím mnohočlenem  $g(x)$ . Skupina zabezpečujících prvků, vyjádřených mnohočlenem  $r(x)$ , bývá také označována zkratkou CRC – Cyclic Redundancy Check. [14]

Rovnice pro výpočet systematického kódu CRC:

$$\frac{x^{n-k} * u(x)}{g(x)} = q(x) + \frac{r(x)}{g(x)}$$

V algoritmu vycházíme ze základního polynomu  $h(z) = \frac{z^m - 1}{g(z)}$  a pro výše uvedený generující polynom byl napsán kód použitý pro výpočet kontrolního součtu v hlavní aplikaci Access\_ComPort\_Modbus:

```
unsigned int CRC16(unsigned int crc, unsigned int byte)
{
    const unsigned int Poly16=0xA001;
    unsigned int LSB, i;
    crc = ((crc^byte) / 0xFF00) & (crc / 0x00FF);
    for (int i=0; i<8; i++)
    {
        LSB=(crc&0x0001);
```

```

        crc=crc/2;
        if(LSB)
            crc=crc^Poly16;
    }
    return crc;
}

unsigned int CRC16(unsigned int crc, unsigned int byte)
{
    const unsigned int Poly16=0xA001;
    unsigned int LSB, i;
    crc = ((crc^byte) / 0xFF00) & (crc / 0x00FF);
    for (int i=0; i<8; i++)
    {
        LSB=(crc&0x0001);
        crc=crc/2;
        if(LSB)
            crc=crc^Poly16;
    }
    return crc;
}

```

Aby mohla být funkce provedena je potřeba jí nějakým způsobem zavolat a přiřadit potřebné proměnné. Volání funkce je provedeno odkazem na tuto funkci z metody void \_\_fastcall TForm1::Button8Click(TObject \*Sender):

```

unsigned char Crc_HByte,Crc_LByte;
unsigned int Crc;
Crc=0xFFFF;
for(int i=0; i<(m_delka); i++)
{
    Crc = CRC16(Crc,promenna[i]);
    promenna[i];
}
Crc_LByte = (Crc & 0x00FF0);
Crc_HByte = (Crc & 0xFF00)/256;

```

Voláním funkce CRC16 předáváme dně proměnné odkazem a to proměnnou *Crc*, která má hodnotu 0xFFFF a proměnnou typu pole, která je nazvána *promenna[i]*. Tato proměnná je indexována proměnnou *int i*, jejíž velikost je dána proměnnou typu *int m\_delka*. Ve smyčce *for(int i=0; i<(m\_delka); i++)* se provede výpočet Crc a volání funkce CRC16 celkem 9krát. To je dáno velikostí proměnné typu *int m\_delka*. Návrátovou hodnotou z funkce *unsigned int CRC16(unsigned int crc, unsigned int byte)* je proměnná typu *unsigned int*, která je rozdělena do dvou bajtů (horního a dolního). Tímto jednoduchým hashovacím algoritmem vypočítáme kontrolní součet, který je společně odeslán s kódem žádosti do slave zařízení (PLC Fatek FBs-40MCR2-D24). Pokud je přenos shledán kontrolním výpočtem jako bezchybný, tak je následně proveden proces, o který bylo požádáno nebo je odeslána odpověď s informacemi umístěných v registrech PLC.

Další jednoduchou, ale důležitou funkcí je převod decimálního čísla na hexadecimální. Tuto funkci poskytuje samotné vývojové prostředí Builder C++, kde jednoduchým požadavkem *IntToHex(EditCisloSlave,2)* je převedena proměnná *EditCisloSlave* na hexadecimální číslo, které má dvě místa. To je dáno druhou proměnnou v argumentu funkce. Převod z hexadecimálního čísla na decimální žádná funkce Builderu C++ neumožňuje, proto musela být napsána vlastní funkce, která má tuto podobu:

```
int HexToInt (AnsiString Value)
{
    if (!Value.AnsiPos("$")) Value = "$" + Value;
    return StrToInt(Value);
};
```

Funkce je volána pomocí definice *HexToInt* a její argumentem je hodnota typu *AnsiString*. Je tedy nutné převést číslo na řetězec. Tento převod přímo umožňuje metoda Builderu C++ *IntToStr(int)*. Tato hodnota je předána jako proměnná typu *AnsiString* do funkce *HexToInt*. Návrátová hodnota této funkce je proměnná typu *int*.

### 10.2.5 ADT Seznam pro ukládání informací

Základem pro uložení dotazů z databáze volaných do hlavní aplikace pro uložení žádosti ke vstupu do objektu je ADT seznam. Tento seznam umožňuje vkládat a později vybírat informace podle identifikačního klíče. K tomu je zapotřebí definovat adresní a konvenční klíč. ADT seznam obsahuje tyto metody:

- *int hash(char \* e)*

- `tUzel * search(tHashTabulka t, char * e)`
- `void init(tHashTabulka t)`
- `void vlož(tHashTabulka t, tData d)`
- `void vypis(tHashTabulka t, tData d, char * e)`
- `void smaz(tHashTabulka t, char * e)`

Seznam je typu tzv. tabulky s otevřeným rozptýlením s konstantním krokem. Operace „*vlož*“ a „*smaz*“ nemají jako parametr klíč nebo samotná data, ale ukazatel na prvek množiny. Důvodem takové konstrukce může být existence prvků množiny před jejich organizací v tabulce. Každá položka tabulky je potenciálně přístupná položce s libovolným klíčem. [15]

Rozptylovací funkci volíme:

$h = h \bmod p$  / kde  $p$  je prvočíslo.

Kolize při výpočtu zbytku po dělení je řešeno pomocí nedefinovaných uložení synonymických položek. Pokud dojde ke kolizi při definování a výpočtu hašovacího klíče, je poslední proměnná při vyhledávání totožných indexů uvolněná z paměti a dochází k dalšímu přepočtu indexu s přidáním nesoudělné proměnné. Přepočítání trvá tak dlouho, dokud se nenajde volné místo v tabulce. Umístění proměnných do tabulky pomocí přepočtu indexu provádí funkce:

```
int hash(char * e)
{
    int vN;
    int vysledek=0;
    while (*e != 0)
        vysledek+=*e++;
    vN=vysledek % N;
    return vN;
}
```

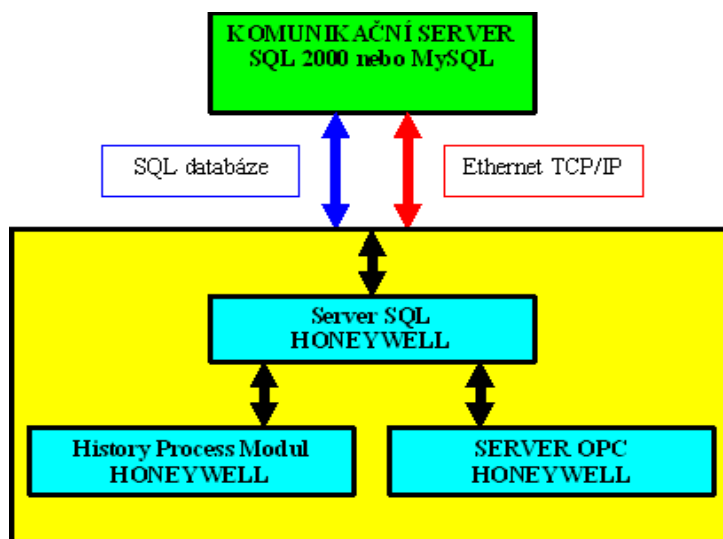
Návratovou hodnotou indexu pro uložení je proměnná `int vN`. Všechny metody pro práci s ADT seznamem jsou uvedeny na CD v příloze diplomové práce.

Důležitým parametrem je nastavení absolutní adresy pro načítání datových souborů z adresáře: "c:\\SQL\\Uniform Server\\udrive\\www\\". Pokud by nebyla nastavena tato absolutní cesta, nebyla by možná výměna dat mezi aplikacemi v PHP a C++ aplikacemi.

## 11 PROGRAMOVACÍ AUTOMAT A AKČNÍ ČLEN INTEGRACE

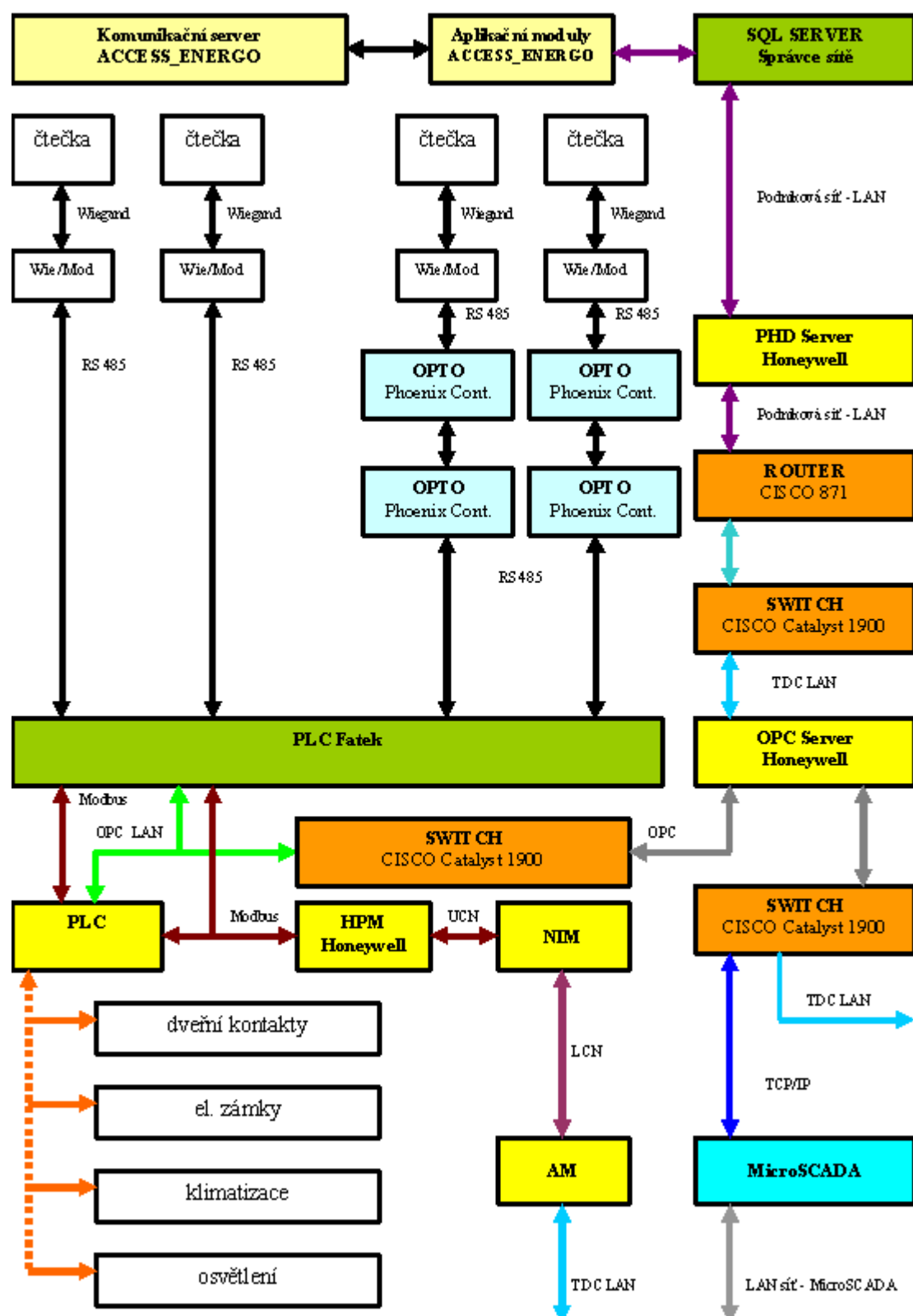
Po vytvoření potřebné databáze, rozhraní pro komunikaci mezi obsluhou a aplikací je možné provést integraci aplikace Access\_Energo do systému ASŘ. Posledním úkolem, který zbývá je vytvořit integrační člen mezi hlavní aplikací a automatickým systémem řízení procesu. Tímto akčním členem je modulární PLC, který používá komunikační protokol MODBUS. Pro projekt bylo zvoleno PLC Fatek FBs-40MCR2-D24, který zcela vyhovuje podmínkám integrace, což je komunikace po modbusu, modulárnost a kvalitní technická podpora.

Technické řešení je nakresleno na obrázku 34. V horní části obrázku je znázorněn aplikační server a modul s hlavní běžící aplikací Access\_Energo, jehož funkce jsou podrobně popsány v kapitole 10. Tato aplikace využívá funkci SQL serveru, která obsahuje databázi zaměstnanců a objektů určených pro projekt požadavku zvýšení bezpečnosti práce v energetických objektech za pomoci čteček karet RFID. Celkový popis a tvorba databáze je popsána v kapitole 8. V pravé části obrázku je zakreslena podniková síť energetického podniku se všemi komponenty, které jsou nutné pro integraci čteček karet. Činnost SQL serveru znázorňuje následující obrázek 33.



Obr. 33 Znáznornění SQL Serveru ASŘ Honeywell

Aplikační řešení komunikačního serveru databázového systému pracujícího na podnikové síti není z hlediska topologie příliš složité. Databáze z komunikačního serveru si vyměňuje data s klientem, který je v našem případě Server SQL systému Honeywell. S těmito databázemi nakládají vnitřní aplikační procesory, jako History Proces Modul Honeywell nebo Server OPC Honeywell.

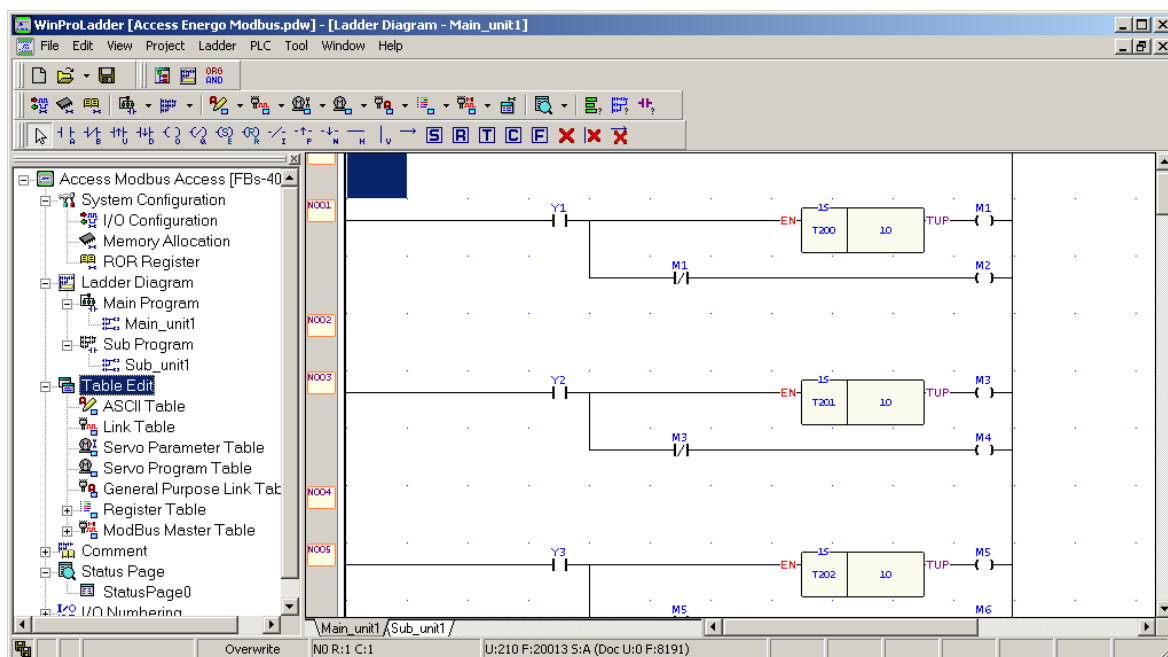


Obr. 34 Návrh řešení z hlediska počítačových sítí ASŘ Honeywell

Základem celého projektu je PLC Fatek FBs-40MCR2-D24, který je v nákresu zobrazen

uprostřed v zeleném rámečku. Tento programovací automat přijímá informace z čteček karet Jablotron JA80, který předává informace z čtečky prostřednictvím převodníku Wiegand/MODBUS. PLC tyto data zpracuje a v hexadecimální podobě ji na vyžádání nadřazeného systému (Honeywel nebo PC dispečera) přepoše pro jejich další zpracování (porovnání dat z databáze, uložení dat do ADT seznamu apod.) do nadřazených systémů. Většina programovacích automatů, které komunikují pomocí modbusu již mají ve svém vývojovém prostředí zavedenou MODBUS komponentu. To je obrovská výhoda, protože odpadá vlastní naprogramování CRC kontrolního součtu. Každé zařízení, které je modbus komponentou vybaveno již je také vybaveno softwarovým vybavením pro výpočet kontrolního součtu. Jak naprogramovat PLC a využít tuto MODBUS komponentu je vysvětleno v následující kapitole.

### 11.1 Programovací automat PLC Fatek FBs-40MCR2-D24 a využití komponenty MODBUS pro komunikaci mezi zařízeními



Obr. 35 Vývojové prostředí PLC Fatek FBs-40MCR2-D24 WinProLadder

Programovací automat Fatek FBs-40MCR2-D24 jsem pro projekt zvolil z důvodu dobré technické podpory, uvolněného vývojového prostředí, které je dodáváno zdarma k PLC a slušné technické dokumentaci. Vývojové prostředí WinProLadder je pro programování automatů přehledné a práce ve vývojovém prostředí je zcela iterativní. Existuje jak



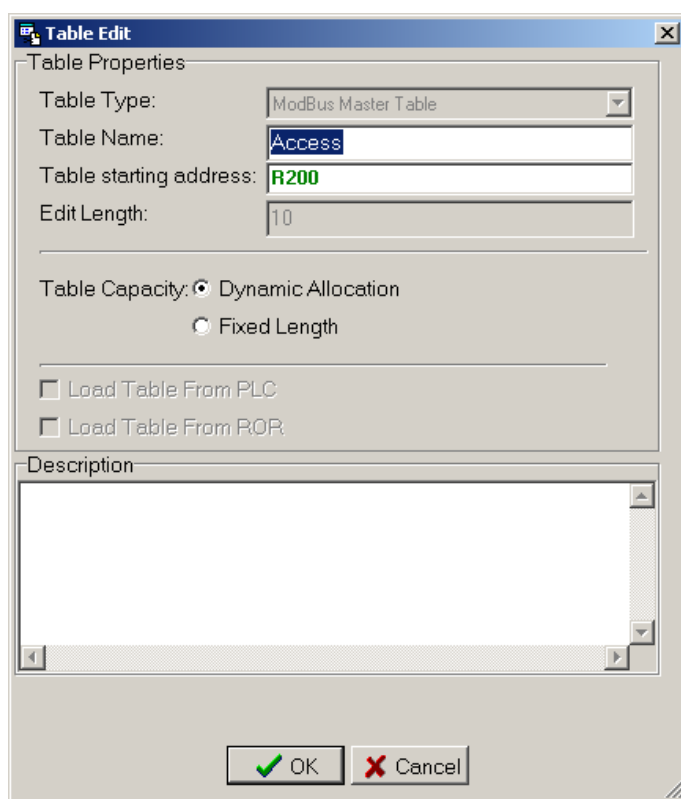
anglická, tak česká verze tohoto prostředí. Další výhodou, která byla shledána pro realizaci projektu je modulárnost PLC, která nabízí mnoho řešení a možností.

První co je nutné provést pro naprogramování PLC je nainstalovat MODBUS komponentu a vytýčit si rozsah registrů (zásobník), kam se data z čtečky budou ukládat.

Pokud klikneme na „MODBUS Master“ položku v projektovém okně:



Po kliknutí na tlačítko se objeví položka „Add MODBUS Master Table“, která obsahuje okna pro zadání názvu typu tabulky, jejího jména a především hodnotu startujícího registru odkud se budou načítat data z čtečky do nadřazených systémů. Nastavení pro projekt má následující podobu:

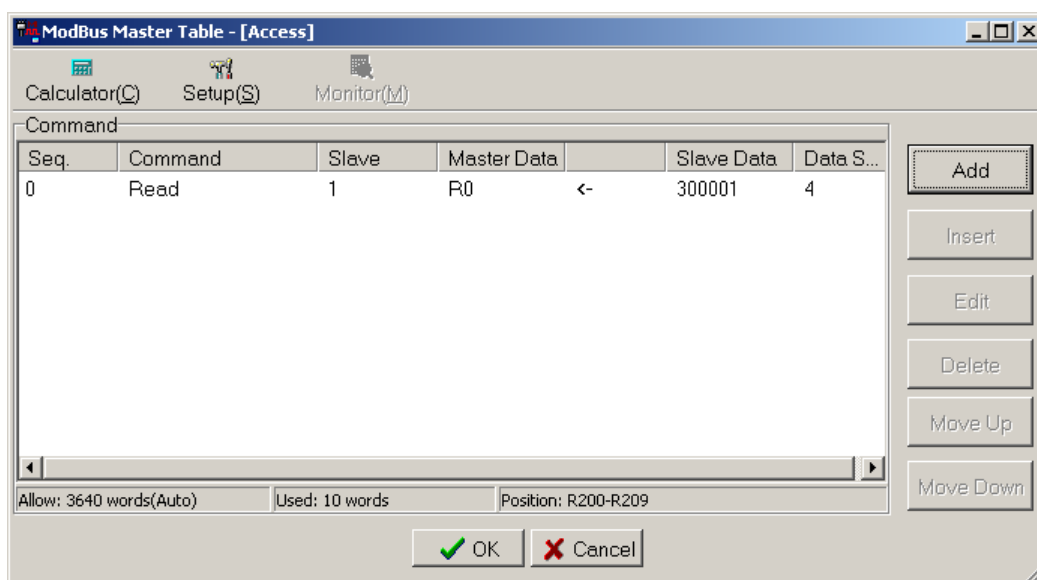


Obr. 36 Nastavení Master tabulky pro MODBUS

Table Type: Je nastaveno pevně na „MODBUS Master Table“

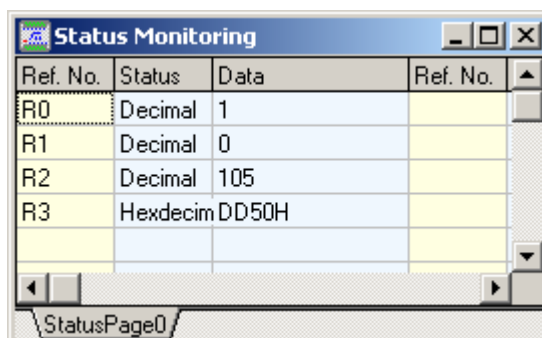
Table Name: Nastavuje jméno tabulky, nastaveno „Access“

Table Starting Address: Nastavuje adresu startujícího registru „R200“. Tento registr však není ten, do kterého se ukládají data za čtečky karet, jak je ukázáno níže. Jde o startující registr pro interní činnosti PLC. Zde je uložena informace o nastavení Mastera PLC. Po definování Master tabulky je nutné provést inicializaci registrů včetně nastavení příkazů „Command“, které definují, zdali se bude do registrů zapisovat, číst nebo zapisovat a číst. Pro projekt vystačíme pouze se čtením registrů a to registru, které začíná buňkou 300001. Do Master tabulky zadáme R0, který odpovídá buňce 300001 v registru.




*Obr. 37 MODBUS Master Table a její inicializace*

Po inicializaci Master tabulky je nutné dále definovat daný registr, do kterého se ukládají informace z čtečky karet. Tyto registry je možné použít jako zásobník, haldu nebo také použít pouze jeden registr. Pro projekt Access\_Energo vystačíme s použitím jednoho registru. Data z čtečky a PLC jsou následně přeneseny do nadřazené aplikace (kapitola 8) a uloženy do ADT seznamu a do databáze MySQL k dalšímu zpracování.



*Obr. 38 Rozmístění registrů s načtenými  
daty ze čtečky*

Jak dále pracovat a definovat startující registry představuje následující tabulka.

SR+0	A5h	50h	A550h Platný Mbus program
SR+1	07h		Dolní byte : Celkové číslo transakce : 1 transakce = 7 registrů
SR+2	Číslo Slave stanice		Platný pro dolní byte : 0~247
SR+3	Kód příkazu		Platné pro dolní byte :=1 čtení dat ze Slave stanice :=2 zápis multidat do Slave stanice :=3 zápis signálu do Slave jednotky
SR+4	Velikost přenášených dat		Platné pro dolní byte : 1~125(Reg) nebo 1~255(Dis)
SR+5	Datový typ Master PLC		Platné pro dolní byte : v rozsahu 1~3 nebo 12~13, definuje datový typ
SR+6	Reference na startující registr dat Master PLC		Definuje startující adresu dat pro Master jednotku
SR+7	Datový typ Slave stanice		Platné pro dolní byte : v rozsahu 0~4, definuje datový typ
SR+8	Reference na startující registr dat Slave PLC		Definuje startující adresu dat pro Slave jednotku
SR+9	Číslo Slave stanice		 Opakující se zápis do registru
SR+10	Kód příkazu		
SR+11	Velikost přenášených dat		
SR+12	Datový typ Master PLC		
SR+13	Reference na startující registr dat Master PLC		
SR+14	Datový typ Slave stanice		
SR+15	Reference na startující registr dat Slave PLC		
SR+....			
SR+2+n*7	Rezervováno		N je celkové číslo transakce

Tab. 39 Popis „Starting“ registrů v PLC

Zkratka SR znamená Starting Register pro komunikační funkce v PLC. Z tabulky vyplývá, že jedna transakce odpovídá 7+1 registrům a další transakce je kopií první transakce.

V našem projektu tedy vystačíme s jednou transakcí. Zde přichází registr R200, který odpovídá registru SR+0 a končí registrem SR+8, tedy registrem R208. Zde jsou uloženy příkazy pro načtení dat ze čtečky, které se ukládají do registrů R0 až R3.

Důležitými doplňujícími informacemi jsou údaje o registrech SR+4 a SR+5. Tyto údaje doplňují následující tabulky:

kód	Datový typ	Reference
1	Y (výstupní relé)	0-255
2	M (interní N relé)	0-1911
3	S (step relé)	0-999
12	R (datový registr Rxxxx)	0-3839
13	D (datový registr Dxxxx)	0-3999

Tab. 40 Kódy a reference na datový typ master stanice

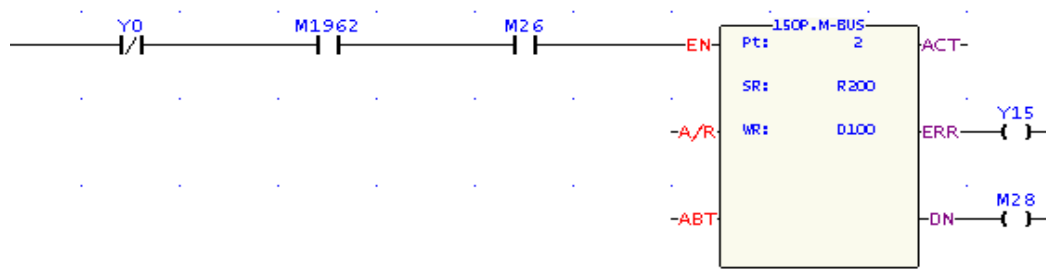
Tabulka odpovídá programovacímu kódu registru SR+4, která tento registr rozšiřuje o požadovaný kód. Pro rozšíření kódu SR+5 jsou použity následující kódy:

kód	Datový typ	reference
0	Diskrétní výstup	1-65535
4	Držící registry	1-65535

Tab. 41 Kódy a reference na datový typ slave stanice

Datový typ pro Master a Slave musí být odpovídající kódu v tabulce. To znamená, že pokud je jakákoliv hodnota mezi 1 až 3, tak Slave stanice musí obsahovat hodnotu kódu 0. Pokud je hodnota Master stanice 12 nebo 13, tak Slave stanice musí mít hodnotu kódu 4.

Po definici a inicializaci startujících registrů SR+n v PLC je nutné také definovat komponentu M-BUS FUN150, která je součástí softwarového vybavení PLC Fatek FBs-40MCR2-D24.



Obr. 39 Funkce M-BUS FUN150 pro PLC Fatek FBs-40MCR2-D24

Pokud se pozorněji podíváme na obrázek 39, tak vidíme tři položky: Pt, SR a WR.

Pt – specifikuje komunikační portu MODBUS Master

SR – Startující registr pro komunikaci

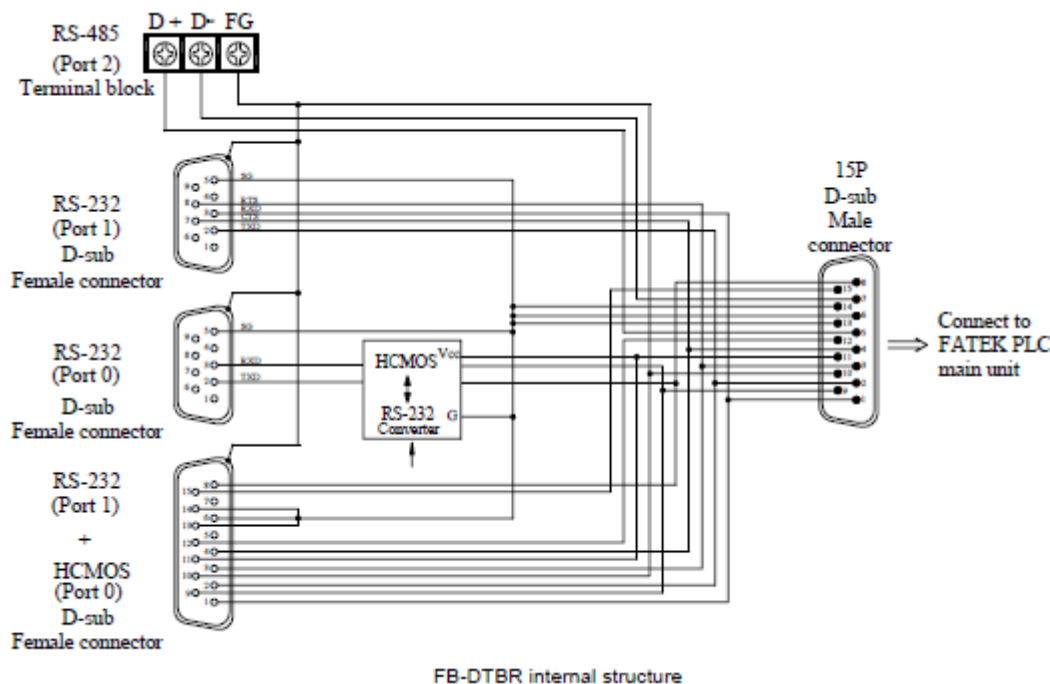
WR – Startující registr pro instrukční operace. Zpracovává 8 registrů.

Nyní již máme definované a inicializované všechny potřebné registry v PLC, které potřebujeme k tomu, abychom mohli načítat data z čtečky RFID karet JA-80. Tyto registry jsou zaznamenány v následující tabulce:

Kód	hodnota registru	definice
Pt	2	definice pro RS485
SR	R200	startující registr pro komunikaci
SR	R0	registr ukládání dat z čtečky
WR	D100	startující registr pro instrukce Modbus

Tab. 42 Tabulka definovaných a inicializovaných registrů

Kód Pt2 značí, že se budeme k PLC připojovat pomocí sériového rozhraní RS485. Protože PLC Fatek FBs-40MCR2-D24 je automat, který je sestaven s příslušných modulů, je každé Pt jiné pro žádaný sériový port. Z obrázku jde vysledovat, že Portu 2 je přiřazena RS-485.



Obr. 40 Přiřazení portu kódu Pt

Funkce SR byla vysvětlena výše a poslední funkcí, která zbývá je funkce WR, která ukládá instrukce pro operace v M-BUS FUN150. Tyto kódy a jím přiřazené registry popisuje následující tabulka:

Registr	Horní byte	Dolní byte	Instrukce
WR+0	Výsledný kód	Číslo transakce	<div><ul style="list-style-type: none"><li>- Kód udávající transakční výsledek</li><li>- Číslo transakce indikuje, která transakce je v procesu</li><li>- Číslo stanice Slave stanice:<ul style="list-style-type: none"><li>01H - čte status slave stanice</li><li>03H - čte držící registry ze Slave</li><li>05H - zapni single cívku ve Slave</li><li>06H - zapiš registr do Slave</li></ul></li></ul></div>
WR+1	Číslo stanice	kód instrukce	
WR+2	Pro interní použití		
WR+3			
WR+4			
WR+5			
WR+6			

Tab. 43 Tabulka startujících registrů pro instrukční operace

V tabulce není uvedeno, co znamená výsledný kód v položce WR+0. Tento výsledný kód slouží k získání informace co se děje při komunikaci mezi stanicemi. Výsledný kód obsahuje tyto informace:

- 0 – Transakce proběhla úspěšně
- 2 – Chybná délka přenášených dat
- 3 – Chyba příkazu (Command Code Error)
- 4 – Chyba datového typu (Data Type Error)
- 5 - Chyba referenčního odkazu
- 6 – Inkonzistentní datový typ
- 7 – Chybný port
- 8 – Neplatný komunikační program
- A – Žádná odpověď ze Slave jednotky
- B – Chyba komunikace

Pokud se objeví na registru WR (D100) číselná informace 0, znamená to, že přenos dat mezi čtečkou a PLC Fatek FBs-40MCR2-D24 proběhl v pořádku a na registru R3 se objeví ID z karty, která byla načtena čtečkou karet. Naopak pokud se na registru D100 objeví hexadecimálně B (decimálně číslo 10) nastala při přenosu chyba v komunikaci a data ze čtečky karet JA-80 nebyla přenesena.

Největším problémem jsem při realizaci projektu shledal v adresaci registrů. Tedy, kde na Slave jednotce nalézt informaci o požadovaných registrech a kam je následně PLC ukládá po načtení. Následující tabulka znázorňuje odkud (ze kterých registrů z čtečky karet JA-80) jsou stahovány a do kterých registrů se v PLC informace ukládají. Například pro příkaz „čti“ jsou informace z paměťových registrů převodníku Wie485 v rozsahu 300001-465535 ukládány do registrů v PLC do adres D0-D3999.

Sekvence	Příkaz	Slave	Data (Master)	Data(Slave)	Délka dat
0	Čti	1-247	Y0-Y255	000001-065535	1-255
			M0-M1911	000001-065535	1-255
			S0-S999	000001-065535	1-255
			R0-R3839	300001-465535	1-125
			D0-D3999	300001-465535	1-125
1	Zapiš	0-247	Y0-Y255	000001-065535	1-255
			M0-M1911	000001-065535	1-255
			S0-S999	000001-065535	1-255
			R0-R3839	300001-465535	1-125
			D0-D3999	300001-465535	1-125

Tab. 44 Tabulka rozsahu registrů v PLC

V tabulce 43 v položce „instrukce“ jsou zmíněny dvě důležité položky. Těmito položkami jsou hodnoty instrukcí 05H a 06H.

05H – sepni jednotlivou cívku v PLC (odpovídá dveřním zámekům, které požadujeme otevřít)

06H – načti jednotlivý registr ze Slave jednotky (ID číslo karty z čtečky karet)

Standardní protokol MODBUS je otevřený protokol, proto si jej každý výrobce komunikačního zařízení trochu upravuje tak, jak to potřebuje. PLC Fatek FBs-40MCR2-D24 interně komunikuje vlastním MODBUS protokolem Fatek-MODBUS a se zařízeními externími používá pro komunikaci standardní MODBUS RTU jak je popsán v teoretické

části této Diplomové práce. Rozdíl je pouze v různých adresacích na paměťové registry, což může být při nastavování a programování komunikace mezi čtečkou karet a PLC velmi nepříjemné, pokud programátor nezná tyto dvě převodové tabulky:

Rozmístění registrů pro kód 5		
Modbus	Fatek-MODBUS	Popis
00001-00256	Y0-255	Diskrétní výstup
01001-01256	X0-X255	Diskrétní vstup
02001-04002	M0-M2001	Diskrétní M Relé
06001-07000	S0-S999	Diskrétní S relé
09001-09256	T0-T255	Status T0-T255
09501-09756	C0-C255	Status C0-C255
40001-44168	R0-R4107	Držící registr
45001-45999	R5000-R5998	Držící registr nebo ROR
46001-48999	D0-D2998	Datový registr
49000-49256	T0-T255	Hodnota T0-T255
49501-49700	C0-C199	Hodnota C0-C199(16 bit)
49701-49812	C200-C255	Hodnota C200-C255(32bit)

*Tab. 45 Převodová tabulka registrů pro kód 05H*

Rozmístění registrů pro kód 6		
Modbus	Fatek	Popis
000001-000256	Y0-255	Diskrétní výstup
001001-001256	X0-X255	Diskrétní vstup
002001-004002	M0-M2001	Diskrétní M Relé
006001-007000	S0-S999	Diskrétní S relé
009001-009256	T0-T255	Status T0-T255
009501-009756	C0-C255	Status C0-C255
400001-404168	R0-R4107	Držící registr
405001-405999	R5000-R5998	Držící registr nebo ROR
406001-408999	D0-D2998	Datový registr
409000-409256	T0-T255	Hodnota T0-T255
409501-409700	C0-C199	Hodnota C0-C199(16 bit)
409701-409812	C200-C255	Hodnota C200-C255(32bit)

*Tab. 46 Převodová tabulka registrů pro kód 06H*

Poslední tabulka představuje nastavení speciálních registrů, především pro analogové vstupy a výstupy. Tyto registry v projektu nevyužívám, proto je uvádím pouze pro zajímavost. Použitím těchto registrů můžeme pomocí analogových smyček ovládat intenzitu vytápění, osvětlení nebo podobné regulační akční členy, které potřebují ke své regulaci výstupní smyčku 4-20 mA popřípadě 0-10 voltů. Přídavnými moduly je možné snímat analogovou informaci z regulátorů pro další zpracování těchto informací. Pomocí PLC Fatek FBs-40MCR2-D24 a čtečky karet lze tedy vytvořit integrovaný systém typu 2A



nebo 2B, kde alespoň jeden prvek je prvkem poplachové aplikace. Lze tedy regulovat topení, intenzitu osvětlení nebo klimatizaci v rozvodnách po přijetí požadovaného id čísla karty pracovníka a nastavit požadované pracovní prostředí.

Rozmístění speciálních registrů		
Modbus	Fatek	Popis
02001-03912	M0-M1911	Celkový počet ovládaných M relé
03913-04002	M1912-M2001	Speciální relé
40001-43840	R0-R3839	Celkový počet ovládaných R relé
43841-43904	R3840-R3903	Analogový nebo numerický vstup
43905-43968	R3904-R3967	Analogový nebo numerický výstup
43969-44168	R3968-R4167	Speciální registr

*Tab. 47 Tabulka speciálních registrů v PLC Fatek FBs-40MCR2-D24*

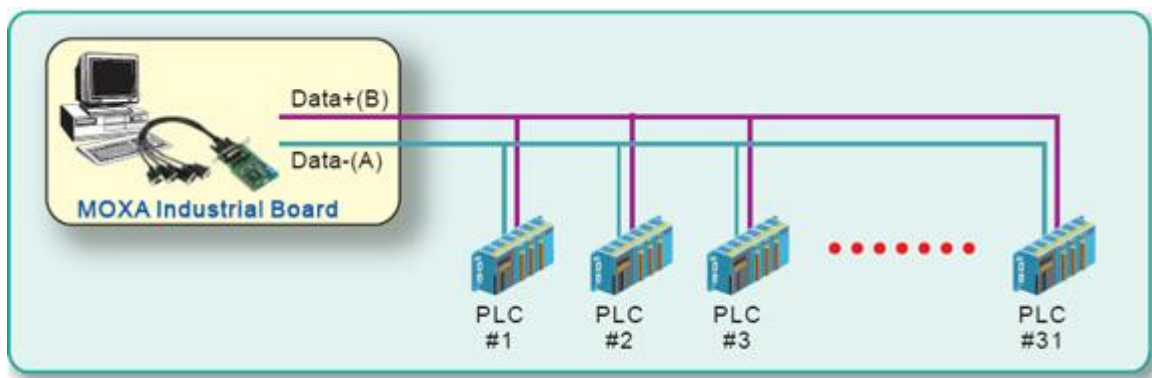
Z výše uvedených tabulek je patrné, že je možné také vytvořit vlastní zásobník s daty v PLC a ty také jako zásobník registrů předávat referencí do hlavní aplikace Access\_ComPort\_Modbus. Vytvoření zásobníku v PLC je doporučeno pouze v případě, kdyby docházelo k rychlé identifikaci zaměstnanců na jedné čtečce. Řešení bez vytvořeného zásobníku v PLC umožňuje větší možnost využití paměti PLC pro technologické procesy.

Pro zpracování dat z čtečky se využívá pouze čtyř registrů. Jde o registry R0 až R3. Popis těchto registrů je vysvětlen v kapitole 12 „Převodník Wiegand/MODBUS“.

## 11.2 Fyzické zapojení a nastavení komunikace na sériových portech PLC

Jakmile je provedena definice a inicializace potřebných registrů, tak můžeme přistoupit k fyzickému připojení PLC do celkového systému ASŘ Honeywell. Stavba počítačových sítí v energetickém podniku již byla vysvětlena a zobrazena v kapitole 5 (obr. 5 a obr. 6), a návrh začlenění čtečky a PLC nejlépe popisuje obrázek 34 kapitoly 11. V této podkapitole bude fyzické připojení PLC upřesněno blíže.

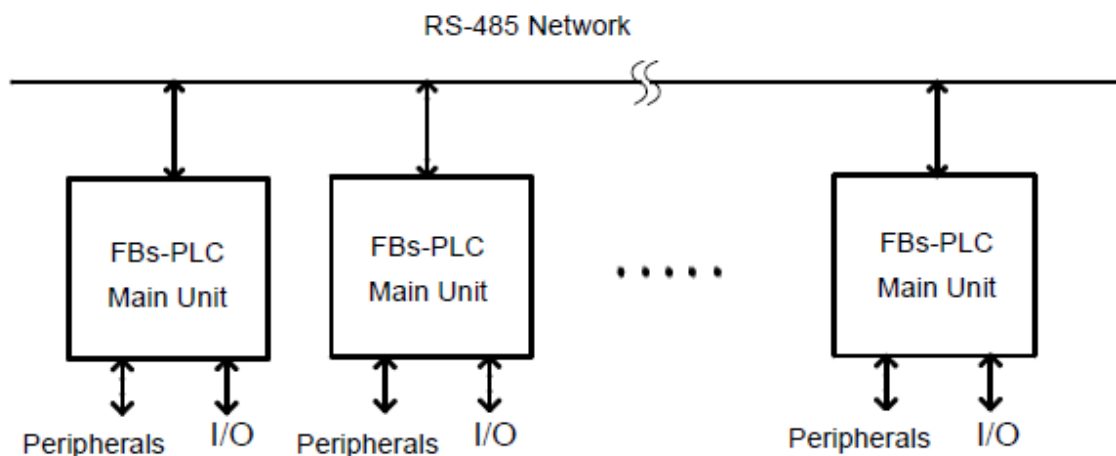
Sběrnice RS-485 představuje další krok v možnostech propojení vzdálených zařízení. Pomocí této sběrnice může komunikovat maximálně 32 vysílačů a 32 přijímačů. Funkčnost sběrnice je zaručena díky tomu, že všechny přijímače i neaktivní vysílače se v klidu musí nacházet ve stavu vysoké impedance, tj. nijak neovlivňují komunikující zařízení.



Obr. 41 Příklad zapojení PLC pomocí RS-485

RS485 je odolná komunikační sběrnice používaná hlavně v průmyslových zařízeních nebo v prostředích s požadavky na vysokou odolnost proti rušení. RS485 (známá také jako dvoudrátová RS485) využívá pouze dva vodiče: A a B, označované také jako RxTx+ a RxTx- nebo kladnější vodič a zápornější vodič. Linka je poloduplexní, to znamená, že v jednu chvíli může vysílat pouze jedna strana (tzv. systém dotaz - odpověď).

Maximální délka jedné větve RS485 je až 1200 metrů. Lze ji prodloužit pomocí opakovačů. Na jedné větvi může být dle normy maximálně 32 zařízení. Zařízení musí být propojována ne do tzv. hvězdy (od všech zařízení do jednoho bodu), ale postupně od jednoho k druhému. [12]



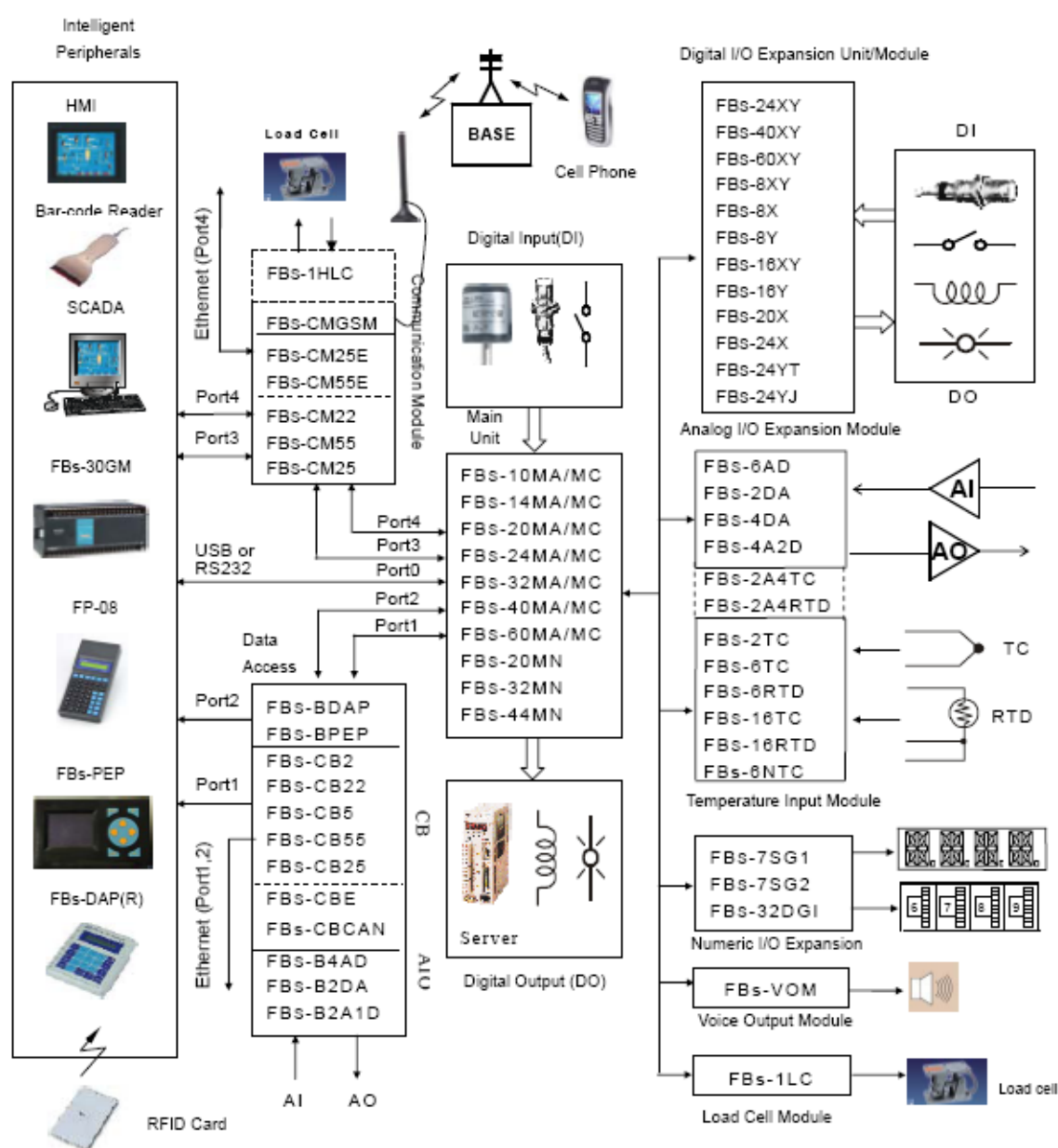
Obr. 42 PLC Fatek FBs-40MCR2-D24 na RS-485

Na obrázku 42 je zobrazen náčrt užití vysokorychlostní RS-485, které může jednoduchým způsobem propojovat 2-254 stanic, které vystupují v sítích se svým vlastním identifikačním adresovaným číslem. Všechny tyto stanice jsou podřízené jedné Master jednotce, která sbírá a obhospodařuje všechny připojené Slave jednotky. Tím jsou všechny jednotky spojeny v jedné síti a mohou sdílet své data navzájem.

Značnou výhodou komunikačního protokolu MODBUS je, že jeho podoba je shodná jak pro komunikaci po sériové lince, tak po Ethernetu, protože kód instrukce je shodný a pouze se liší metoda přenášených dat. V rámci sériové linky je to přenos jednotlivých bajtů a v rámci ethernetu se kód zabalí do datové části paketu a odesílá prostřednictvím transportní vrstvy na požadovaný port 500 nebo 502. Systém ASŘ Honeywell pro energetické procesy vyvinul dva systémy a to:

TDC 3000 – vzájemně propojených po sériové lince RS-485

EXPERION – nově vyvinutý systém komunikující po ethernetu



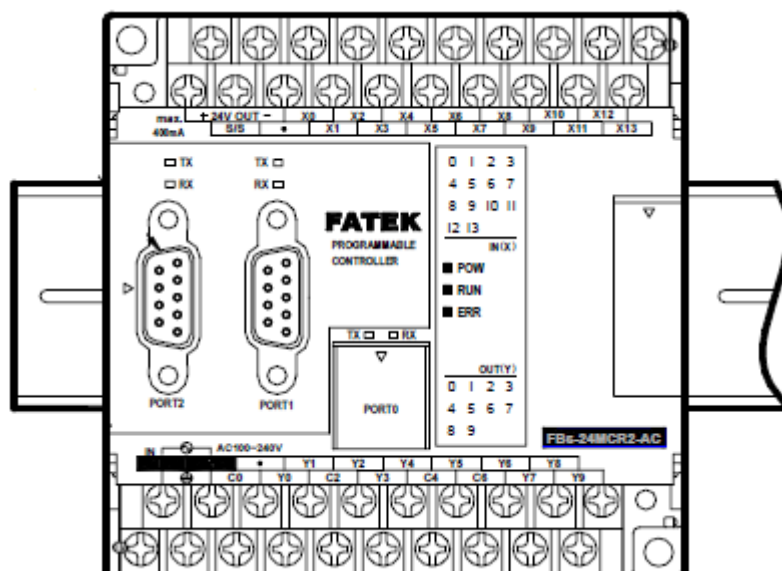
Obr. 43 Přehled komponentů PLC Fatek FBs-40MCR2-D24

Protože většina PLC je modulárních, tak je si možné zvolit, jak tyto automaty mezi sebou propojíme. Důležité je znát adresace portů (Port 0, Port 1 atd.). V projektu Access\_Energo je použito tří komunikačních portů:

Port 0 – interní port pro komunikaci PLC a vývojovým prostředím WinProLadder, kterým se programuje automat

Port 1 – Port se sériovou komunikací RS-232, která slouží k připojení nadřazeného PC s běžícím aplikačním programem Access\_ComPort\_Modbus.

Port 2 – Port se sériovou linkou RS-485, která slouží pro komunikaci mezi PLC a čtečkou karet prostřednictvím převodníku komunikačních protokolů Wiegand na MODBUS



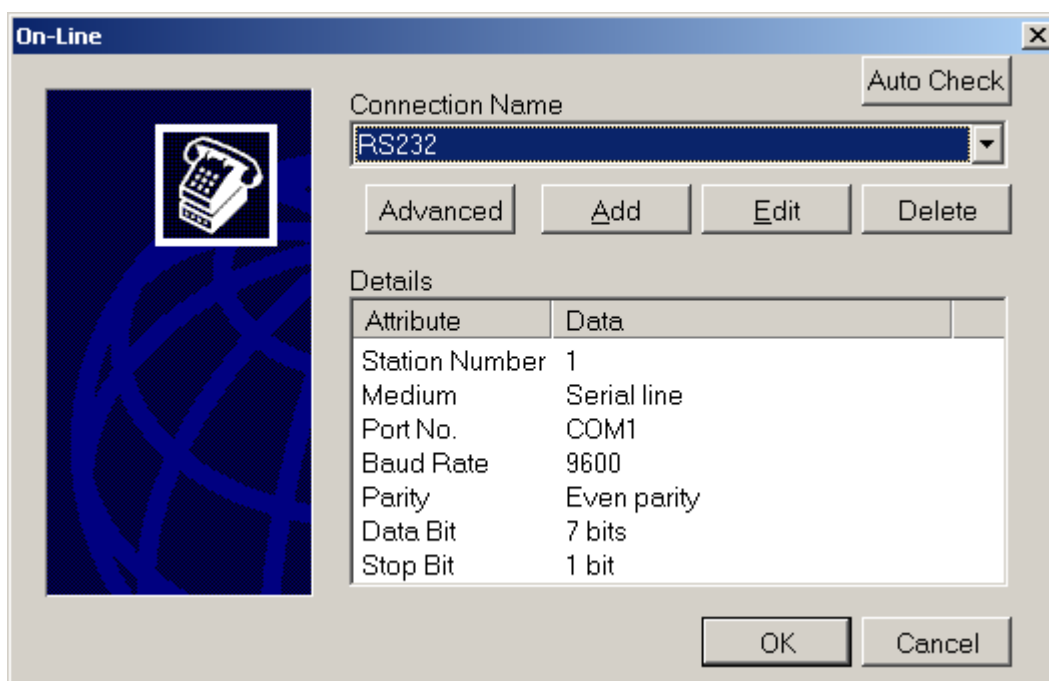
*Obr. 44 Příklad PLC Fatek FBs-40MCR2-D24 se dvěma porty RS-232 a jedním interním portem 0 pro aplikaci WinProLadder*

### 11.2.1 Nastavení komunikačních portů PLC Fatek FBs-40MCR2-D24 v projektu Access\_Energo

V podkapitole 11.2 jsem se zmiňoval o komunikačních portech a rozhraních, kterým je PLC Fatek FBs-40MCR2-D24 vybaven. V projektu pracujeme s komunikačním protokolem MODBUS na sériové lince, proto všechny tři porty, které jsem použil, komunikují prostřednictvím sériové linky. Jsou to porty 0,1 a 2. Jejich nastavení nyní popíšu.

### 11.2.1.1 Nastavení Portu 0

Port 0 slouží k vlastnímu programování PLC pomocí PC, na kterém je nainstalována aplikace WinProLadder. Komunikace je prováděna po sériové lince RS-232, avšak pomocí nestandardního konektoru na straně PLC, který však výrobce dodává společně s PLC. Nevýhodou se jeví převážná nepřítomnost sériového rozhraní na většině notebooků, to však lze odstranit simulátorem sériového rozhraní a využití převodníku USB/RS-232. Jak se připojit k PLC pomocí aplikace WinProLadder popisovat v této diplomové práci nebudu, protože výrobce dodává podrobný návod včetně příkladů různých zapojení, jako jsou časovače, čítače, klopné obvody a podobně. Co je však nutné vědět, tak je nastavení komunikace mezi PLC a PC.



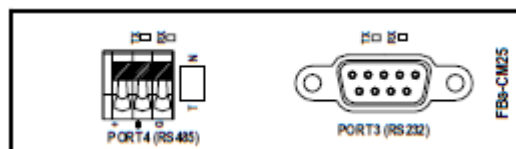
Obr. 45 Nastavení Portu 0

Z obrázku je patrné, že pro komunikaci mezi PLC a PC s běžící aplikací WinProLadder je nutné nastavit tyto parametry: Číslo stanice – 1, Komunikace – sériová linka, Port – 1, Přenosová rychlost – 9600 bps, Data Bit – 7, Stop Bit – 1.

Po nastavení parametrů komunikace již můžeme začít programovat ve vývojovém prostředí a definovat potřebné registry a cívky výstupních relé.

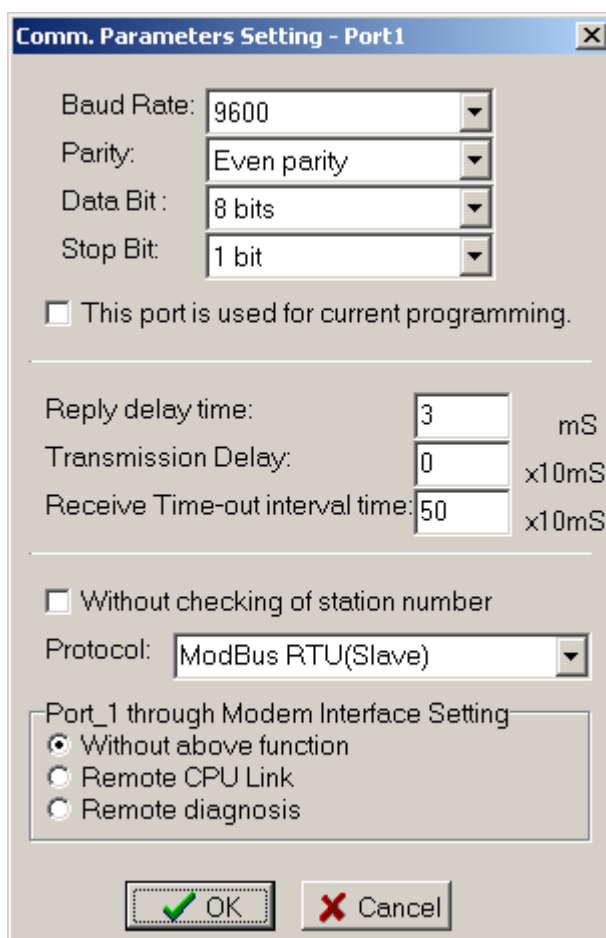
### 11.2.1.2 Nastavení Portu1

Port 1 slouží ke komunikačnímu propojení PLC se systémem, na kterém běží aplikace Access\_ComPort\_Modbus. Tento port opět využívá sériové komunikace RS-232, avšak již standardní 9 pinový konektor.



Obr. 46 Komponenta PLC se dvěma  
sériovými porty

Port1 neumožňuje přístup k aplikaci WinProLadder, protože tato aplikace vyžaduje svůj vlastní nestandardní sériový port a protokol, avšak ke komunikaci s jakýmkoliv jiným zařízením pomocí standardního protokolu MODBUS je velmi vhodný.

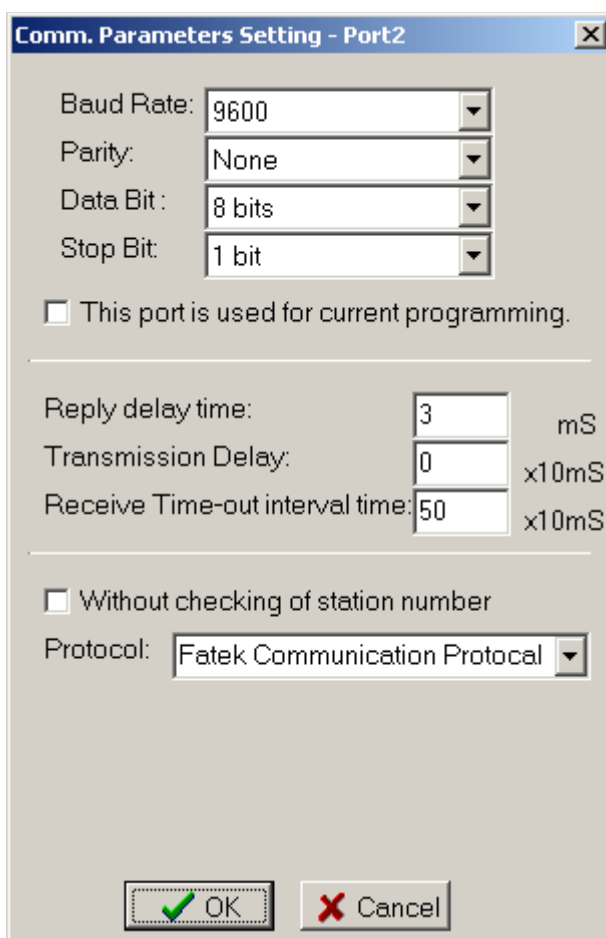


Obr. 47 Nastavení Portu 1

Protože se chová PLC k nadřazenému systému jako Slave, tak dostává standardní dotazy pomocí standardních kódů komunikačního protokolu MODBUS. Zkráceně tedy, aplikace Access\_ComPort\_Modbus běží na Master stanici (PC) a komunikuje se Slave jednotkou (PLC) pomocí standardního protokolu MODBUS RTU. Uvádím to z důvodu, že pokud je PLC Master jednotkou a čtečka karet Slave jednotkou, PLC používá vlastní upravený MODBUS protokol, který výrobce nazývá Fatek MBus protokol, který se drobně liší v definici registrů. Rozdíly mezi standardním modbusem a Fatek MBusem jsou popsány výše v kapitole 11.1.

### 11.2.1.3 Nastavení portu 2

Jak již bylo zmíněno v předchozí podkapitole, že pokud se chová PLC k zařízení jako Master, tak se Slave zařízením komunikuje pomocí vlastního komunikačního protokolu Fatek MBus, který vychází ze standardního protokolu MODBUS. V našem případě PLC Fatek FBs-40MCR2-D24 je Master jednotkou a převodník protokolu Wie485 se chová jako Slave jednotka.



Obr. 48 Nastavení portu 2

Pokud porovnáme obrázky 47 a 48 vidíme, že nastavení portu je shodné až na nastavení komunikačního protokolu. Pro Port1 (RS-232) je nastaven standardní MODBUS, zatímco pro Port2 (RS-485) je nastaven Fatek Communication Protocol.

Pokud jsou všechny porty nastaveny podle předložených atributů, tak již je možné načítat ze čtečky karet identifikační čísla přiložených karet. Samotný proces přenosu informace z čtečky do převodníku Wie485 a následně do registrů v PLC je popsáno v následující kapitole.



## **12 PŘENOS INFORMACE Z ČTEČKY KARET JA-80 DO PLC FATEK FBS-40MCR2-D24 POMOCÍ PŘEVODNÍKU WIE485**

Na trhu bezpečnostních technologií se v současnosti setkáváme s rozličnými výrobky, které používají různorodé technologie, komunikační protokoly a hardwarové rozhraní. Pro projekt Access\_Energo bylo nutné vybrat takovou čtečku, která by umožňovala komunikaci s prvky ASŘ v energetickém průmyslu. Tedy vybrat takovou čtečku karet, která umí komunikovat po modbusu a zároveň bylo potřeba vybrat takovou čtečku, u které výrobce poskytuje kvalitní dokumentaci a podporu. Čtečka, která využívá komunikační rozhraní MODBUS není na našem trhu rozšířena a bohužel kvalitní technická podpora u zahraničních produktů včetně komerčně poskytované technické dokumentace je také velmi špatná. Pokud jsem výrobce kontaktoval, tak buď mě byl zaslán pouze takový materiál, který jsem našel volně dostupný na internetu nebo dokonce mě výrobce vůbec na žádost neodpověděl. Přijatelnější cestou se jevílo použít čtečku, která komunikuje po rozhraní Wiegand, což je nejrozšířenější komunikační protokol průmyslu PKB a převodník rozhraní Wiegand z bezkontaktních čteček na RS485. Technická podpora a dokumentace pro čtečky karet na Wiegandu na trhu s prvky bezpečnostního průmyslu je uspokojivá, a proto bylo pouze potřeba najít vhodný převodník s dostatečnou podporou Wiegand/MODBUS na RS485. Bohužel také převodníků s dostatečnou podporou není na českém trhu uspokojivé množství. Pro účel projektu byl vybrán převodník Wie485 českého výrobce Papouch s.r.o.

### **12.1 Převodník Wie485 a jeho charakteristika**

Wiegand je standardní protokol, kterým komunikují čtečky bezkontaktních karet. Wie485 umožňuje tento protokol převést na standardní průmyslovou sběrnici RS485. Pomocí Wie485 lze snadno připojit standardní bezkontaktní čtečky s protokolem Wiegand k PC nebo PLC, které komunikují prostřednictvím komunikačního rozhraní MODBUS. Výstupem z Wie485 je identifikační číslo karty, čitelné přes RS485.

Převodník Wie485 je složen ze dvou typů rozložení paměti, které nazýváme registry. Prvním typem registrů jsou tzv. registry Holding Registers (uživatelské registry) a druhým typem registrů jsou Input Registers (systémové registry).

### 12.1.1 Holding Register

Tyto registry slouží k nastavení a uchování informací nastavených parametrů pro komunikaci. Rozložení paměti uživatelských registrů upřesňuje následující tabulka.

Adresa	Přístup	Funkce	Popis
1	čtení zápis	0x03 0x10	Pozice pro volné použití
2	čtení zápis	0x03 0x10	ID zařízení (1-247)
3	čtení zápis	0x03 0x10	Komunikační rychlost: 3 -9600 Bd 4 -19200 Bd 5 -38400 Bd 6 -57600 Bd 7 -115200 Bd
4	čtení zápis	0x03 0x10	Režim sériové linky: 0-8 datových typů, bez parity, 1 stopbit 1-8 datových typů, sudá parita, 1 stopbit 2-8 datových typů, lichá parita, 1 stopbit 3-8 datových typů, bez parity, 2 stopbity 4-8 datových typů, sudá parita, 2 stopbity 5-8 datových typů, lichá parita, 2 stopbity

Tab. 48 Rozložení Uživatelských registrů

Uživatelské Registry jsou registry, které nám po zápisu uchovávají své data v paměti, tyto data nemohou být přepisovány na rozdíl od systémových registrů dynamicky, ale pouze staticky. Pro konfiguraci je možné použít například software MODBUS Configurator, který je ke stažení na [www.papouch.com](http://www.papouch.com). Jen je třeba vzít v úvahu, že tento software nepočítá s tím, že Wie485 v režimu konfigurace komunikuje stále stejnou rychlostí. Je tedy potřeba nejdříve nastavit komunikační rychlost 9600 Bd a ID 1 a poté změnit parametry na nové. [16]

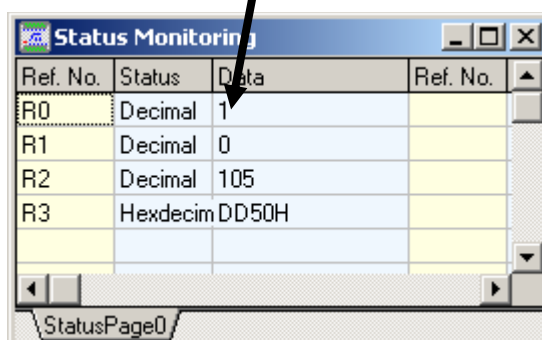
### 12.1.2 Input Register

Systémové registry slouží k uchování vstupních dat, které přicházejí ze čtečky karet do převodníku Wie485. Tyto data jsou ukládána do Fronty. Fronta je v programování abstraktní datový typ typu FIFO (z anglického *First In, First Out*, česky první dovnitř, první ven). V projektu Access\_Energo vystačíme pouze se čtením prvních 4 buněk fronty. To znamená, že do PLC Fatek FBs-40MCR2-D24 se vysílají data prvních čtyř registrů

uložená po aktivaci čtečky RFID kartou. Další načtená data z čtečky po aktivaci příznakem prvního registru jsou uložena na pozici 1-4 registru převodníku Wie485 a předchozí načtené informace z čtečky jsou posunuty na 4-6 registr, přičemž data na posledních 4 registrech jsou fronty vysypány. Co je aktivace příznakem prvního registru vysvětluje následující obrázek.

Je-li na registru v kolonce Data = 1, tak zatím nebyl přečten žádný kód a v registrech R1-R3 jsou uložena původní informace z ID karty.

Je-li na registru R0 v kolonce Data = 0, došlo k aktivaci čtečky RFID karet a na registr R0-R3 jsou zapsána nová data. Původní jsou posunuty ve frontě na pozice registrů R-4 až R6. Tato 0 v kolonce data jsou nazývána příznakem prvního registru.



Ref. No.	Status	Data	Ref. No.
R0	Decimal	1	
R1	Decimal	0	
R2	Decimal	105	
R3	Hexdecim	DD50H	

StatusPage0

Obr. 49 Příznak prvního registru

Popis vstupních registrů:

- R0 – Registr příznaku prvního registru, který signalizuje nově přichodí data z čtečky RFID karet do převodníku Wie485. Po příznaku R0 = 0 jsou na registr R1-R3 tyto nově přichodí data zapsány a staré data jsou přesunuty na registry R4-R6. Stejný princip ukládání dat používá jak převodník Wie485, tak PLC Fatek FBs-40MCR2-D24. Nás však zajímají pouze informace uložené v prvních čtyřech registrech v převodníku Wie485, které jsou předávány po MODBUS protokolu do registrů v PLC Fatek FBs-40MCR2-D24 (R0-R3). Z PLC načítáme pouze jeden registr R3 do hlavní aplikace Access\_ComPort\_Modbus běžící na PC, protože

v projektu Access\_Energo ukládáme pouze ID čísla z karet do ADT seznamu. (viz. kapitola 10.2.5).

- R1 – Registr, který používá pouze Wie42, jehož velikost je 8 bitů a slouží pro uložení prvního bytu čísla (LSB). Protože čtečka karet JA-80 komunikuje v protokolu Wiegand 26, tak tento registr nás nezajímá.
- R2 – Registr obsahuje první byte čísla. Velikost registr je 8 bitů.
- R3 – Registr má velikost 16 bitů a nese informaci o dvou nejnižších dvou bytech čísla.

Celkový přehled o adresaci vstupních registrů je v následující tabulce.

Adresa	Přístup	Funkce	Popis				
			Wie 30	Wie 26	Wie 40	Wie 32	Wie 42
1	čtení	0x04	Status kódu 1 = nebyl přečten žádný kód 2 = došlo k přečtení kódu				
2	čtení	0x04	Nepoužito	Nepoužito	Nepoužito	Nepoužito	8 bit LSB: První byte čísla
3	čtení	0x04	16 bit Horní dva byty čísla	8 bit LSB: První byte čísla	Nepoužito	16 bit Horní dva byty čísla	16 bit 2 a 3 byte čísla
4	čtení	0x04	16 bit Nejnižší dva byty čísla	16 bit Nejnižší dva byty čísla	16 bit Nejnižší dva byty čísla	16 bit Nejnižší dva byty čísla	16 bit 4 a 5 byte čísla

Tab. 49 Rozložení Input Registrů

Z následující tabulky již je možno vypořádat, že kód k načtení prvních 4 registrů do PLC nebo PC by měl následující podobu.

- Adresa Slave zařízení = 1
- Funkce pro čtení = 4
- Startující registr = 1
- Počet načítaných registrů = 4

Číselná hodnota zadávaných hodnot je pro ukázkou zadávaná v decimální soustavě, avšak pro MODBUS je nutné tyto čísla převést do hexadecimální soustavy. Decimální a

hexadecimální soustava je totožná od 0 do 9, proto není nutná konverze čísel pro tyto hodnoty.

## 12.2 Čtečka RFID karet JA-80

Čtečka Jablotron JA-80N je základním prvkem celé stavby projektu Access\_Energo a je komponentem systému Oasis 80 firmy Jablotron. Lze použít k ovládání přístupu (dveřního zámku) nebo k ovládání zabezpečovacího systému. K ústředně Oasis se čtečka připojuje pomocí rozhraní WJ-80. Případně lze připojit k přístupovému systému AS-80. Čtečka poskytuje data v protokolu Wiegand 26 a obsahuje celkem 7 vodičů:

vodič	signál
červený	+12V (napájení, cca 60mA)
zelený	D0 (výstup data Wiegand 26b)
hnědý	D1 (výstup data Wiegand 26b)
šedý a bílý	TMP (výstup rozpínacího sabotážního kontaktu)
žlutý	BZR (vstup k ovládání zvuku čtečky – sepnutím na GND generuje pípnutí)
modrý GND	(společná svorka napájení)
růžový	NEZAPOJEN

Tab. 50 Popis vodičů čtečky JA-80

Pro připojení čtečky karet k převodníku karet nám postačí připojení pouze čtyř vodičů.

vodič	signál
Data 1	Signál Data 1 pro čtečku
Data 0	Signál Data 0 pro čtečku
GND	Zem komunikační linky
Uout	Výstup napájecího napětí pro čtečku

Tab. 51 Seznam vodičů připojených do převodníku Wie485

Při správném připojení vodičů a konfiguraci komunikace mezi čtečkou JA-80 a převodníkem Wie485 je možné okamžitě načítat ID čísla karet do registrů převodníku

Wie485. Nejdůležitější část projektu, což je načtení identifikačního kódu karty pracuje bez problému a naprosto spolehlivě.

### 13 VYUŽITÍ SYSTÉMU ACCESS\_ENERGO A JEHO VÝZNAM V ENERGETICKÉM PRŮMYSLU

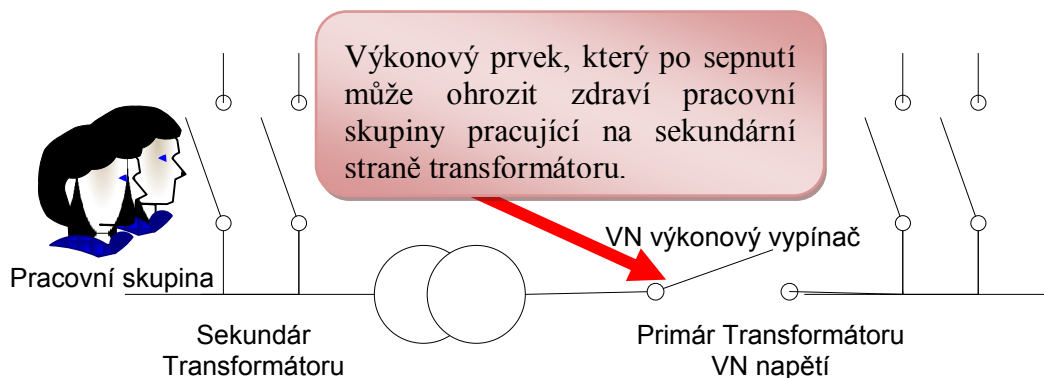
Systém Access\_Energo je koncepčně navrhnutý systém k ochraně životů pracovníků a jejich zdraví při práci na zařízení vysokého napětí, kde největší hrozbou v ochraně zdraví pracovníka je faktor selhání lidského činitele. Chyba lidského faktoru, která může nastat při manipulacích nebo pracích na VN zařízení má mnoho faktorů:

- Špatný zdravotní stav obsluhy nebo dispečera
- Spěch
- Stres
- Špatná znalost prostředí
- Neznalost topologie VN sítí
- Faktor tzv. neomylnosti pracovníka, především u starších a zkušených pracovníků
- Pracovní slepota
- Monotónní pracovní činnosti, kdy již pracovník nepřemýšlí při činnostech, ale koná práci stereotypně
- A mnoho dalších negativních faktorů

Při studiu smrtelných a těžkých pracovních úrazů s trvalými následky bylo zjištěno, že od roku 1986 bylo plných 95% úrazů zapříčiněno selháním výše uvedených lidských faktorů. Nejčastější příčinou úrazů byl omyl pracovníka, který si zmýlil pracoviště, na kterém měl pracovat a vnikl do zařízení, které bylo pod napětím. Druhým nejčastější příčinou byly manipulace, kdy dispečer opomněl možnost zpětného proudu například na transformátorech z vysokého na nízké napětí. Při manipulacích přivedl na sekundární vinutí transformátoru přes výkonový vypínač napětí, které se transformovalo na napětí vysoké na primární straně transformátoru.

Proti takovému případu slouží k ochraně pracovníků zkratovací soupravy, ale pokud pracuje na zařízení více skupin, tak může dispečer jednu takovou skupinu lehce přehlédnout. Jestliže dispečer nařídí rozvodnému povel k odstranění zkratovací soupravy a rozvodný si zařízení nezkontroluje, zdali se na něm nevyskytuje nějaká pracovní četa, protože věří dispečerovi, že prošel všechny příkazy „B“ (písemný příkaz bezpečnosti práce na zařízení), tak s velkou pravděpodobností může dojít k těžkému nebo smrtelnému úrazu. Následující obrázek popisuje, jak k takovému úrazu může snadno dojít, dojde-li k sepnutí

výkonového vypínače na primární straně transformátoru a na sekundární pracuje „opomenutá“ pracovní skupina.



*Obr. 50 Pracovní skupina pracující na sekundární části transformátoru*

Jestliže dispečer přehlídne pracovní skupinu a dá pokyn dozornému, aby připravil k sepnutí primární stranu transformátoru, tak pracovní skupina je vystavena nebezpečí přivedení zpětného proudu ze sekundární části transformátoru. Protože výkonový vypínač je velmi často umístěn vzdáleně od samotného transformátoru, tak ani obsluha nemusí vědět o přítomnosti pracovní skupiny na zařízení. Mnohem závažnější důsledky by měla situace opačná, kdyby pracovní četa pracovala na zajištěné primární straně transformátoru a dispečer nechal odjistit a následně sepnul sekundární část transformátoru. Protože by došlo k transformaci z nízkého napětí na napětí vysoké, tak by zcela určitě došlo k velmi těžkému pracovnímu úrazu s následkem smrti.

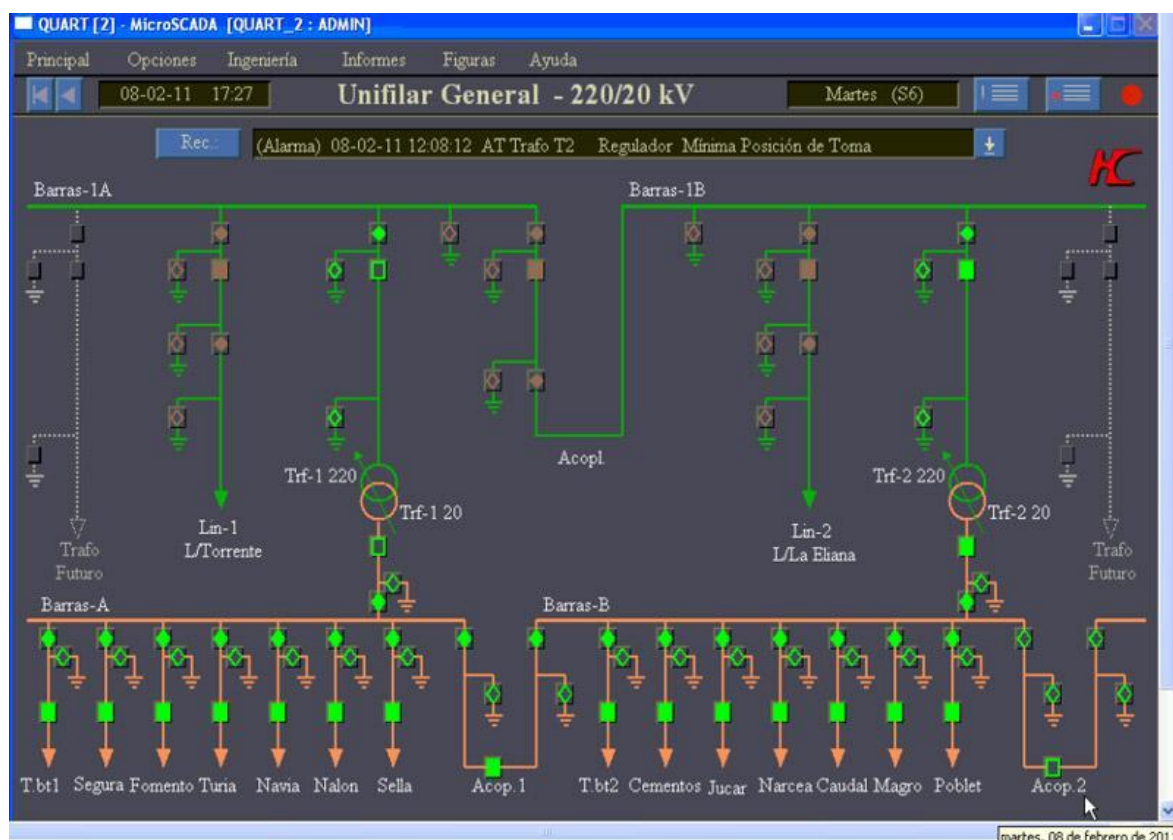
K eliminaci těchto faktorů může posloužit zavedení každé pracovní skupiny do databáze pracovníků nebo pracovních skupin, které na zařízení pracují a zanesení jejich přítomnosti do algoritmů pro povolení sepnutí zařízení systémy řízení.

### 13.1 Algoritmus pro povolení manipulací na energetickém zařízení

Každé energetické zařízení je již ovládáno pomocí systémů ASŘ. Nejznámější jsou systémy řízení Honeywell, MicroSCADA vyvinuté společností ABB. V našem regionu se již zavedla společnost AISE ze Zlína, jež distribuovala vlastní systém, který spolehlivě funguje ve společnosti Barum-Continental v oddělení Energetika. Se všemi systémy jsem se již seznámil během své praxe. Tyto systémy jsou si navzájem velmi podobné. Rozdíly jsou ve využití komunikačních protokolů. Honeywell a AISE komunikuje pomocí

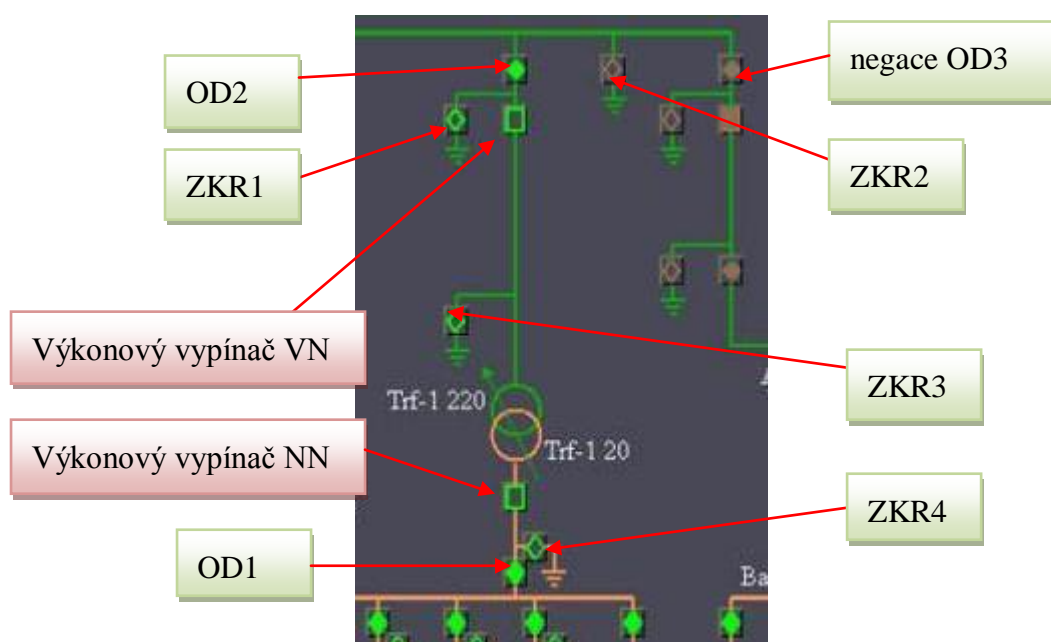


MODBUS protokolu, MicroSCADA od ABB používá vlastní protokol SPABus nebo Profibus.



Obr. 51 Příklad systému řízení procesů v energetice

Proměnné pro výpočtový algoritmus zobrazuje následující obrázek.



Obr. 52 Prvky algoritmizace

Koncepce systému je stejná, základní ovládací schéma je bitmapový náčrt rozvodny, který obsahuje akční ovládací okna. Jestliže dozorný hodlá provádět manipulace, je před každým zapnutím nebo vypnutím vypínače proveden logický součin všech parametrů, jejichž výsledek musí být roven logické 1. Definice proměnných je zakreslena na obr. 52.

Pro sepnutí sekundárního vinutí Trf-1 220 (TR1) musí být rozepnuty zkratovací soupravy ZKR1, ZKR2, ZKR3 a ZKR4, sepnuté odpojovače OD1 a OD2 a rozepnutý odpojovač OD3 (negovaná nula = 1).

Výpočet umožnění sepnutí sekundárního vinutí transformátoru Trf, kdy výsledek TR1 = logická 1:

$$\overline{ZKR1} * \overline{ZKR2} * \overline{ZKR3} * \overline{ZKR4} * OD1 * OD2 * \overline{OD3} = 1$$

Z rovnice vidíme, že sepnutí primární části transformátoru jde pouze, jestliže jsou všechny vcházející proměnné logické 1 nebo negací logické 0, kdy je výsledný součin roven logické jedničce.

Z rovnice lze také vypočítat, že je velmi snadné dosadit další proměnnou a to proměnnou informující o přítomnosti pracovní skupiny na zařízení. V podstatě jde o dosazení negované nuly. Jestliže je skupina přítomna, tak ze systému Access\_Energo dostane systém řízení ASŘ informaci o přítomnosti skupiny (negovaná 1) nebo o zařízení bez pracovníků (negovaná 0). Systém Access\_Energo tedy nejen povoluje vstup a práci na zařízení, ale také slouží k zasílání informací o přítomnosti pracovníků na zařízení. Tyto informace můžeme kdykoliv načíst z ADT seznamu, který je součástí aplikace hlavního programu.

Výsledný algoritmus s informací o přítomnosti pracovní skupiny:

$$\overline{ZKR1} * \overline{ZKR2} * \overline{ZKR3} * \overline{ZKR4} * OD1 * OD2 * \overline{OD3} * \overline{ACS1} = 1$$

Přítomnost pracovní skupiny, která se neodhlásila u dispečera z databáze pracovníků, nepovolí systému uvedení linky na straně sekundární strany transformátoru pod napětí. Uvedení zařízení pod napětí je možné až po odstranění skupiny z databáze prací na zařízení v aplikaci MySQL v hlavním okně webového prohlížeče (obrázek 47). Odstraněním dané položky s databázového systému zmizí také údaj z ADT seznamu hlavní aplikace Access\_Energo a systém řízení energetických procesů ASŘ (např. Honeywell, MicroSCADA, Aise) přijímá informaci o negované nule, která je přiřazena právě takovému zařízení, na kterém se již nevyskytuje žádná pracovní četa nebo zaměstnanec.

**ROZVODNY ACCESS**

Vyber zamestnance : Jan Bednik Rozvodna : r400.1 Kobka : linka 3

Potvrdit ukonceni praci

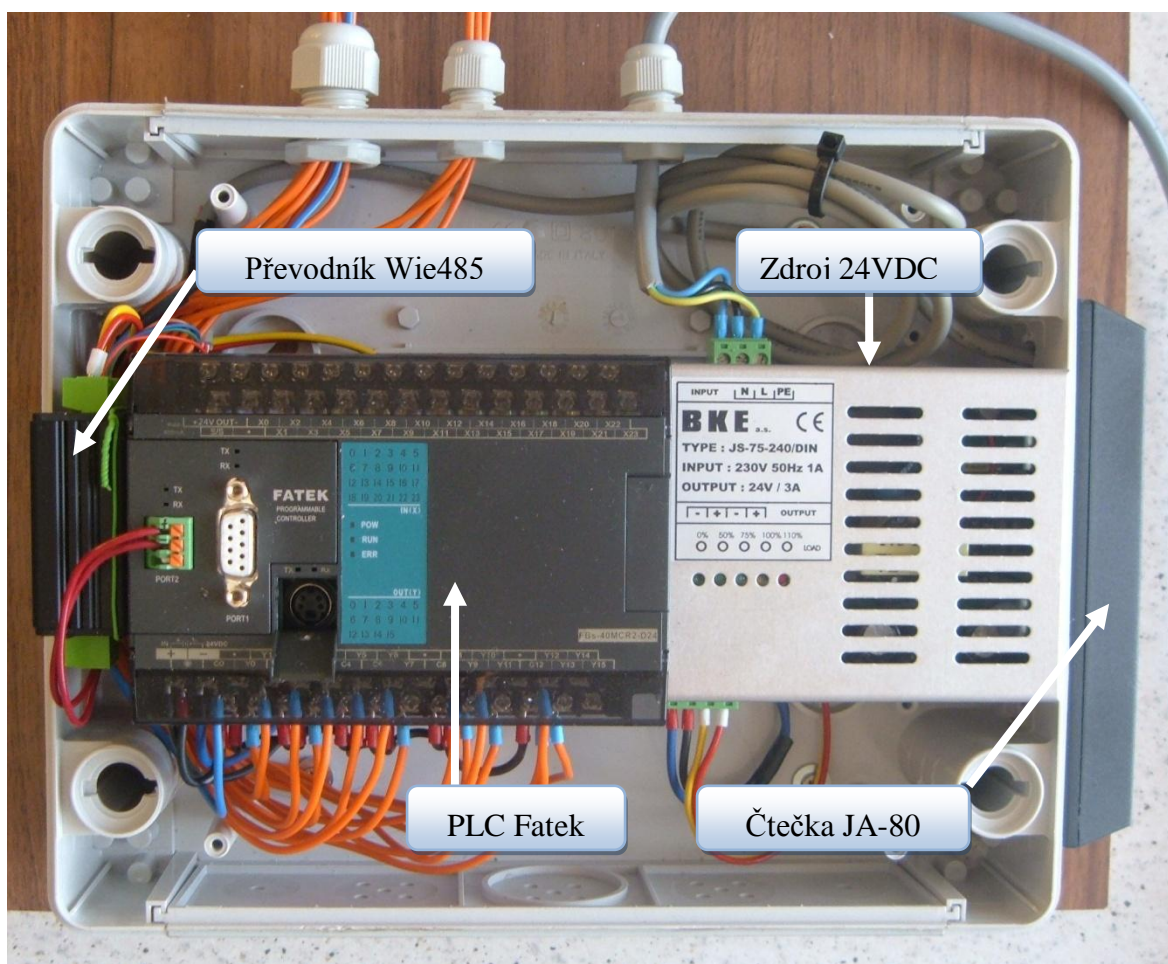
Potvrzení ukončení prací na zařízení

*Obr. 53 Ukončení prací pro zápis negované logické jedničky pro ASŘ*

Po potvrzení ukončení prací povolí nadřízený ASŘ systém sepnutí následného výkonového vypínače a tím přivedení napětí do požadované linky. Potvrzení ukončovacích prací smí dispečer pouze v přítomnosti vedoucího čety, který zároveň předá příkaz „B“ k odepsání. Tím jsou oficiálně práce na zařízení ukončeny.

## 14 MODELOVÝ SYSTÉM ACCESS\_ENERGO

Aby bylo možné ověřit funkčnost vytvořených softwarových aplikací, bylo nutné vytvořit malý model energetického zařízení simulující akční členy ASŘ. Tento model se skládá ze čtečky JA-80, převodníku Wie485, PLC Fatek FBs-40MCR2-D24, simulace rozvodu s dveřními kontakty příslušných kobek a PC, na kterém jsou nainstalovány tři základní aplikace MySQL + PHP, hlavní aplikace Access\_Energio a webový prohlížeč s nainstalovaným HTTP serverem Apache. Hlavní požadavek na model systému je takový jako by se jednalo o „ostrý“ systém, a který se musí chovat stejně jako aplikace v prostředí ASŘ. Na tomto modelu lze odzkoušet všechny aplikace, případně odladit problematické funkce aplikace.

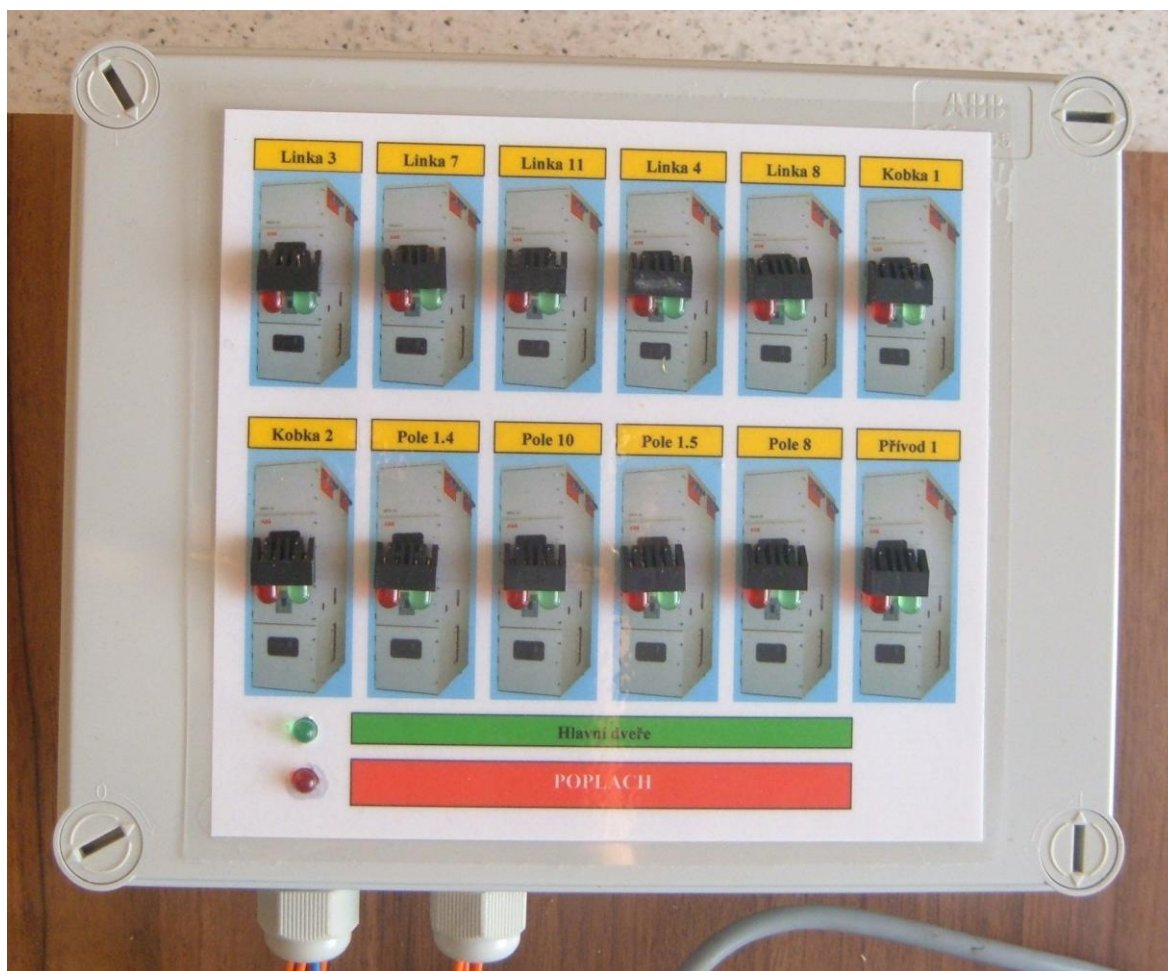


*Obr. 54 Modul s PLC, převodníkem Wie485, čtečkou JA-80 a zdrojem 24VDC*

Základní část modelu se skládá s modulu, na kterém je instalován PLC Fatek FBs-40MCR2-D24, převodník Wie485, čtečka RFID karet JA-80 a zdroj 24VDC. Výstupní relé z PLC jsou zapojeny tak, aby simulovaly sepnutí elektrického zámku v kobce



navoleného zařízení. Jestliže dojde k sepnutí, tak na modelové verzi systému signalizují příslušné světelné diody stav dveřních kontaktů (odemčeno nebo uzamčeno). Zelená dioda signalizuje otevření kobky a svítící červená led dioda vypovídá o tom, že kobka je uzavřena a není proto možno na zařízení pracovat.

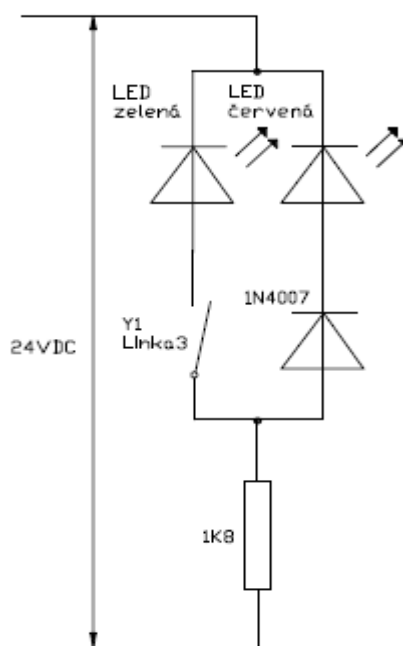


*Obr. 55 Model rozvodny s vývody a simulací elektrických dveřních zámků*

Simulaci těchto dveřních zámků lze zřetelně vidět na obrázku 50, kde každá dvojice diod (červená a zelená) určuje stav dveřního zámku. Protože se počítá, že elektrické zámky jsou ovládány stejnosměrným napětím, budou po celý průběh prací pod napětím. Do bez napětového stavu (uzamčení elektrického zámku) se zařízení uvede až po odsouhlasení ukončení prací. Stav otevření hlavní dveří signalizuje led dioda zelené barvy, kterou můžeme vidět v dolní části obrázku 55. Tato dioda svítí pouze 5 sekund a po uplynutí této doby zhasne. To simuluje krátkodobé otevření hlavních dveří rozvodny. V projektu se počítá s tím, že na venkovní části dveří bude pouze madlo a dveře půjdou otevřít pomocí kliky pouze zevnitř objektu. Hlavní dveře mohou být také vybaveny adresovaným magnetickým kontaktem pro signalizaci vstupu do objektu rozvoden. Dozorný se tak může

okamžitě dozvědět o vstupu osoby nebo osob do rozvodny a případně telefonicky vedoucího práce kontaktovat, jestliže bude provádět manipulační práce na vývodu, které nejsou v souladu s pracemi vypsány na příkazu „B“. Protože všechny vstupní a výstupní informace přicházejí do stejného PLC, tak je velmi snadné všechny tyto informace zpracovat a ukládat do stejného úložiště dat pro další zpracování.

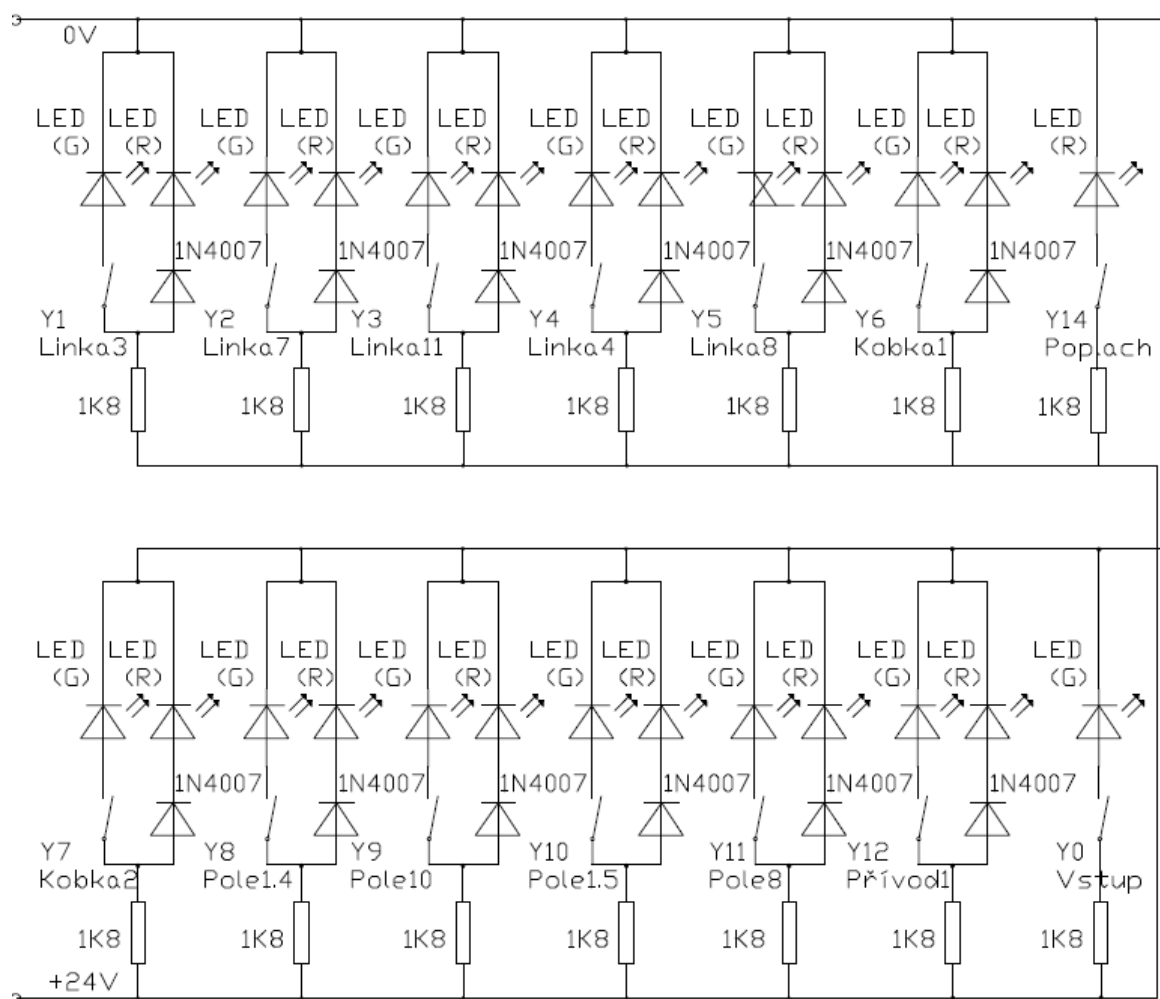
Na modelovém projektu systému Access\_Energo bylo při simulaci sepnutí dveřních kontaktů potřeba vzít na zřetel, že výstupní relé obsahuje pouze spínací kontakt, proto bylo nutné nalézt takové řešení, které by umožnilo sledovat signalizační stav led diody pro stav sepnuto nebo vypnuto. Řešením je využít součtu úbytků napětí na diodách a provést sériové zapojení dvou diod pro stav signalizující uzamčenou kobku. Následující obrázek zobrazuje funkci tohoto zapojení.



*Obr. 56 Výřez zapojení simulace  
sepnutí dveřního kontaktu*

Z obrázku je patrné, že součet úbytku napětí na led diodě červené barvy a diodě 1N4007, které jsou zapojeny v sérii činí asi 1,4V. Pokud je relé Y1 Linky3 rozepnuté, tak proud prochází oběma diodami a led dioda, která signalizuje uzamčený stav je rozsvícena. Jestliže dojde k sepnutí relé Y1, tak úbytek na zelené led diodě činí pouze 0,7V. Proud tedy prochází pouze touto diodou, která signalizuje stav sepnuto a červená led dioda zhasne, protože součet úbytku napětí je dvojnásobný ( $1,4V > 0,7V$ ). Toto zapojení platí

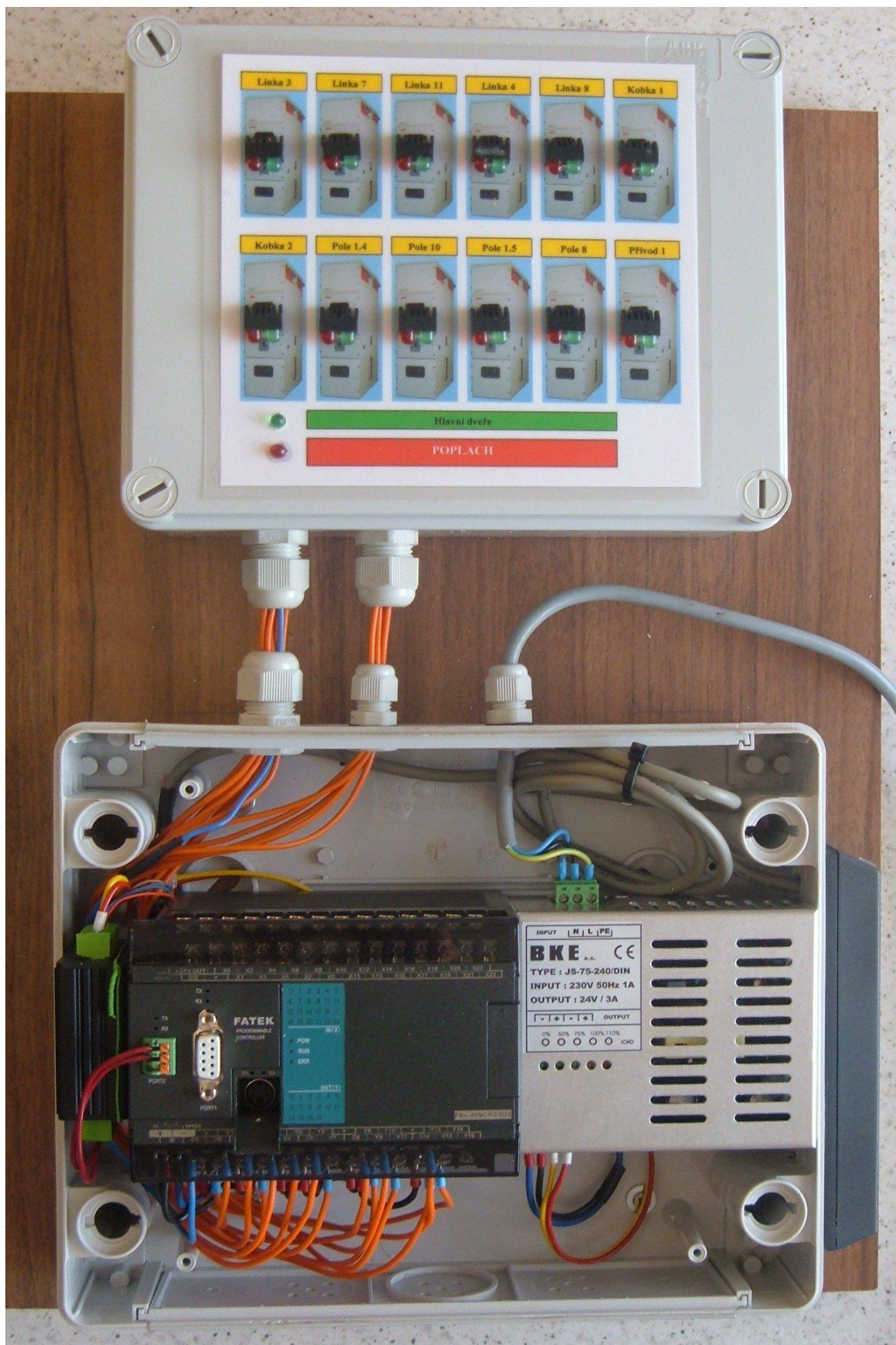
pouze pro modelový systém Access\_Energio. V provozu je řešení velmi snadné a to přidáním jednoho relé pro každý výstup zvlášť s přepínacími kontakty. Klidový kontakt signalizuje uzamčené zařízení a sepnutý pracovní kontakt přivede stejnosměrného napětí na cívku elektrického zámku. Rozmnožením spínacího kontaktu je možné do systému poslat hlášku o stavu zařízení. V tomto případě může také dispečer sledovat, zdali jsou práce na zařízení ukončeny. V principu jde o přivedení další logické 1 nebo logické 0 do nadřazeného systému ASŘ (kapitola 13.1).



Obr. 57 Schéma zapojení modelové simulace dveřních kontaktů v rozvodně

Obrázek 57 zobrazuje celkové zapojení modelového systému Access\_Energio v zapojení s diodami. Z obrázku je patrné, že je možné laboratorně rozšiřovat jakékoliv množství modulů pro odzkoušení nových kontaktních bodů a v laboratorních podmínkách ladit nebo odzkoušet aplikaci předtím než uvedeme zařízení do provozu. K propojení mezi výstupními relé z PLC a signalizačními diodami byl použit vodič CY 1 mm<sup>2</sup> oranžové barvy pro kladný pól a modrý pro společný záporný pól.





Obr. 58 Celkový pohled na model systému Access\_Energy



## ZÁVĚR

Rozvoj moderních informačních a elektronických systémů znamená velký pokrok ve vytváření rozsáhlých bezpečnostních zařízení, které je možné implementovat do stávajících automatických systémů řízení a vytvářet tak rozsáhlé integrované systémy. Složitost případné integrace nespočívá ve výběru zařízení, ale především ve vzájemné komunikaci těchto prvků. Tato práce přináší návrh možného řešení vzájemné komunikace prvků bezpečnostního průmyslu s prvky automatického řízení procesů pomocí standardního protokolu MODBUS. Právě s důvodu jednoduchosti a univerzálnosti je tento protokol velmi rozšířený a oblíbený v aplikacích energetických systémů ASŘ zařízení.

Cílem této práce bylo blíže seznámit bezpečnostního technika s možností integrace bezpečnostních technologií se systémy řízení v energetice, podat základní informace o tom, jak tyto systémy pracují, jak mezi sebou komunikují a navrhnout model, který zvýší bezpečnost práce na energetickém zařízení s využitím čteček karet RFID a komunikačního protokolu MODBUS na sériové lince.

Model vytvořeného bezpečnostního systému dostal název Access\_Energo. Tento modelový návrh je postaven na implementaci prvků přístupového systému v podobě čtečky RFID karet do energetického automatického systému řízení procesů. Model Access\_Energo je navržená a vytvořená plně funkční systémová aplikace softwarového a hardwarového vybavení, která má za cíl pomocí sestavených kódů, algoritmů a návrhu praktické realizace v různých procesech řízení napomáhat zodpovědným pracovníkům operativně předcházet případným mimořádným událostem.

## ZÁVĚR V ANGLIČTINĚ

The development of the modern information as well as the power engineering systems means a big progress in creating extensive security facilities, which can be integrated to our functioning automated systems that we already have. This progress would allow us to expand the integrated systems as well. The problem of complexity within the possible integration doesn't origin in selection of the facility, but it comes from the issues of the communication and fusion of all the elements. This project brings a suggestion that could become a possible resolution to the communication struggles of the elements from the security industry and interfere with the elements of the automatically controlled processes using the standard protocol MODBUS. MODBUS protocol is popular and widespread in application of the energetic automatical systems due to its simplicity and universal usage.

The goal of this essay is to provide a closer introduction for the security technicians about opportunities of the integrated security technologies in field of the power engineering. In addition, we wanted to provide basic information of how use the above listed systems, but also how they work and communicate. We also present a model that would increase the security in the energetic facilities using the reader cards RFID as well as the communication protocol MODBUS on serial demand.

The model of the newly developed security system was named Acces\_Energo. This model plan is build on the process of incorporating accessibility system in form of the reader RFID cards into the automatically directed processing systems. Model Acces\_Energo is fully creatively built functioning software-hardware system application, which works according to the set of codes and algorithms. It can be applied in various processes in order to help responsible workers to prevent extraordinary situations.

## SEZNAM POUŽITÉ LITERATURY

- [1] RONEŠOVÁ, Andrea.  
[Http://home.zcu.cz/~ronesova/index.php?menuitem=bastlirna](http://home.zcu.cz/~ronesova/index.php?menuitem=bastlirna) [online]. Upravené vydání. 2005 [cit. 2011-11-28].  
[Http://home.zcu.cz/~ronesova/bastl/files/modbus.pdf](http://home.zcu.cz/~ronesova/bastl/files/modbus.pdf). Dostupné z WWW:  
[<http://home.zcu.cz/>].
- [2] *Modicon Modbus Protocol Reference Guide*. North Andover, Massachusetts : Modicon, Inc., 1994. 115 s.
- [3] *Integrace BT s ASŘ v energetice*. Zlín, 19.2 2010. Bakalářská práce. UTB Zlín. Vedoucí práce ing. Rudolf Drga.
- [4] *Process Automation College: Local/UCN/APM Maintenance – Technical Reference Manual*, vydané 4.6 1992,
- [5] PROKOPOVÁ, Zdenka. *Databázové systémy MySQL PHP*. Vyd. 1. Ve Zlíně: Univerzita Tomáše Bati, 2006, 126 s. ISBN 80-731-8486-9.
- [6] GILFILLAN, Ian. *Myslíme v jazyce MySQL 4*. Vyd. 1. Praha: Grada, 2003, 750 s. ISBN 80-247-0661-X.
- [7] *HTML: začínáme programovat*. 3., aktualiz. vyd. [i.e.] 1. vyd. Praha: Grada, 2010, 190 s. ISBN 978-80-247-3117-9 (BROŽ.).
- [8] LEISS, Oliver a Jasmin SCHMIDT. *PHP v praxi: pro začátečníky a mírně pokročilé*. 1. vyd. Praha: Grada, 2010, 242 s. Průvodce (Grada). ISBN 978-80-247-3060-8 (BROŽ.).
- [9] PRATA, Stephen. *Mistrovství v C*. 3. aktualiz. vyd. Překlad Boris Sokol. Brno: Computer Press, 2007, 1119 s. ISBN 978-80-251-1749-1 (Váz.).
- [10] MATOUŠEK, David. MATOUŠEK. *C Builder: vývojové prostředí : určeno pro verze 4.0 5.0 6.0*. 1. vyd. Praha: BEN, 615 s. ISBN 80-730-0063-6.
- [11] Používání vláken I. [Http://builder.tsx.cz/3/CBUILDL4a.html](http://builder.tsx.cz/3/CBUILDL4a.html) [online]. [cit. 2012-03-20]. Dostupné z: <http://builder.tsx.cz>
- [12] KAINKA, Burkhard. *Využití rozhraní PC pod Windows: měření, řízení a regulace pomocí standardních portů PC*. 1. vyd. Ostrava: HEL, 2000, 151 s. ISBN 80-861-6713-5.

[13] *Zabezpečení datových přenosů pomocí CRC*. [online]. [cit. 2012-03-20].

Dostupné z:

<http://fieldbus.feld.cvut.cz/system/files/files/cs/vyuka/predmety/A4B38DSP/uloha9.pdf>

[14] *Kódování – cyklické kódy* [online]. Zlín, 20.2 2008 [cit. 2012-03-20]. Bakalářská práce. UTB Zlín. Vedoucí práce RNDr. Ing. Miloš Krčmář

[15] *Implementace datových typů*. [online]. [cit. 2012-03-20]. Dostupné z: <http://fav.q-e-e.net/PT/pt.pdf>

[16] *Wie485 - Převodník Wiegand na RS485*. [Http://www.papouch.com](http://www.papouch.com) [online]. [cit. 2012-03-20]. Dostupné z: <http://www.papouch.com/cz/shop/product/wie485-prevodnik-wiegand-rs485/#productDownload>

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

C++	Programovací jazyk
LAN	Local Area Network – lokální datová síť
WAN	Wide Area Network – veřejná datová síť
MySQL	Strukturovaný dotazovací jazyk
ODBC	Standardní aplikační rozhraní
TCP/IP	Komunikační protokol
OPC	Komunikační protokol
PLC	Programovatelný automat
OLE DB	Ovladač starající se o interpretaci SQL dotazů
GUS	Global User Station
TDC	Totally Distributed Control Systém
TPS	Total Plant Solution
US	Universal Station
AM	Aplication Modul
HM	History Modul
CG	Computer Gateway
NG	Network Gateway
HPM	High-Performance Process Manager / History Process Modul
LCN	Local Connection Network
UCN	Universal Connection Network
APM	Advanced Process Manager
LM	Logic Manager
VN	Vysoké napětí
VVN	Velmi vysoké napětí

PHD      Proces History Database

RS485    Komunikační rozhraní

RS232    Komunikační rozhraní

**SEZNAM OBRÁZKŮ**

Obr. 1 Základní tvar MODBUS zprávy .....	13
Obr. 2 Bezchybná transakce .....	14
Obr. 3 Chybová transakce.....	14
Obr. 4 Datový model MODBUS s oddělenými a neoddělenými bloky .....	16
Obr. 5 Obecný postup zpracování MODBUS požadavku na straně serveru.....	17
Obr. 6 Schéma ASŘ Honeywell .....	41
Obr. 7 Systém ASŘ Honeywell z hlediska napojení externích zařízení .....	44
Obr. 8 Subsystem Tecomat MX-0301.....	46
Obr. 9 Čtečka RFID karet EMS Cobalt C .....	47
Obr. 10 I/O Controller PL.....	48
Obr. 11 Integrátor a regulátor HAWK .....	48
Obr. 12 Příklad architektury s využitím integrátoru HAWK <sup>AX</sup> .....	49
Obr. 13 Převodník Wiegand na RS485 – Wie485 .....	50
Obr. 14 Aplikační členění projektu .....	54
Obr. 15 Umístění zámku nebo signalizačního mikropsínače .....	55
Obr. 16 Algoritmus pro povolení přístupu do objektu .....	56
Obr. 17 Návrh implementace čtečky RFID do systému ASŘ .....	58
Obr. 18 Komunikace serveru PHP+MySQL .....	60
Obr. 19 Základní relační schéma databázového návrhu.....	62
Obr. 20 Základní webové okno aplikace .....	70
Obr. 21 Okno s výpisem identifikace a kvalifikace pracovníka .....	71
Obr. 22 Okno s výpisem rozvodny a pracovníků pro práci na zařízení .....	73
Obr. 23 Volba zaměstnanců, rozvodny a kobky pro povolení pracovních činností .....	74
Obr. 24 Zobrazení událostí pracovních činností .....	76
Obr. 25 Zobrazení okna událostí po zrušení všech pracovních povolení.....	77
Obr. 26 Předávání identifikačních atributů mezi aplikacemi .....	80
Obr. 27 Formulářové okno hlavní aplikace .....	80
Obr. 28 Groupbox s rozbalovacími seznamy pro nastavení komunikace .....	81

Obr. 29 GroupBox Komunikace pro sledování zpráv .....	81
Obr. 30 Komponenta GroupBox Modbus .....	82
Obr. 32 Zobrazení vláken a hlavní aplikace .....	84
Obr. 31 Odezva adresy, kódu a CRC .....	84
Obr. 33 Znázornění SQL Serveru ASŘ Honeywell .....	94
Obr. 34 Návrh řešení z hlediska počítačových sítí ASŘ Honeywell .....	95
Obr. 35 Vývojové prostředí PLC Fatek FBs-40MCR2-D24 WinProLadder .....	96
Obr. 36 Nastavení Master tabulky pro MODBUS .....	97
Obr. 37 MODBUS Master Table a její inicializace .....	98
Obr. 38 Rozmístění registrů s načtenými .....	98
Obr. 39 Funkce M-BUS FUN150 pro PLC Fatek FBs-40MCR2-D24.....	100
Obr. 40 Přiřazení portu kódu Pt .....	101
Obr. 41 Příklad zapojení PLC pomocí RS-485.....	106
Obr. 42 PLC Fatek FBs-40MCR2-D24 na RS-485 .....	106
Obr. 43 Přehled komponentů PLC Fatek FBs-40MCR2-D24.....	107
Obr. 44 Příklad PLC Fatek FBs-40MCR2-D24 se dvěma porty .....	108
Obr. 45 Nastavení Portu 0 .....	109
Obr. 46 Komponenta PLC se dvěma.....	110
Obr. 47 Nastavení Portu 1 .....	110
Obr. 48 Nastavení portu 2.....	111
Obr. 49 Příznak prvního registru.....	115
Obr. 50 Pracovní skupina pracující na sekundární části transformátoru.....	120
Obr. 51 Příklad systému řízení procesů v energetice .....	121
Obr. 52 Prvky algoritmizace .....	121
Obr. 53 Ukončení prací pro zápis negované logické jedničky pro ASŘ.....	123
Obr. 54 Modul s PLC, převodníkem Wie485, čtečkou JA-80 a zdrojem 24VDC.....	124
Obr. 55 Model rozvodny s vývody a simulací elektrických dveřních zámků .....	125
Obr. 56 Výřez zapojení simulace .....	126
Obr. 57 Schéma zapojení modelové simulace dveřních kontaktů v rozvodně .....	127



Obr. 58 Celkový pohled na model systému Access_Energo .....	128
---	-----

**SEZNAM TABULEK**

Tab. 1 Datový model MODBUS.....	15
Tab. 2 ASCII rámec zprávy .....	22
Tab. 3 RTU rámec zprávy .....	22
Tab. 4 ASCII pole pořadí bitů.....	25
Tab. 5 RTU pole pořadí bitů .....	26
Tab. 6 Chybové kódy v protokolu MODBUS .....	29
Tab. 7 Definice funkčních kódů.....	30
Tab. 8 Čti cívky.....	31
Tab. 9 Čti diskrétní vstupy.....	31
Tab. 10 Čti uživatelské registry .....	32
Tab. 11 Čti systémové registry .....	32
Tab. 12 Zapiš jednu cívku .....	33
Tab. 13 Zapiš jeden registr .....	33
Tab. 14 Čti stav .....	34
Tab. 15 Diagnostika .....	34
Tab. 16 Kódy podfunkcí podporované sériovými zařízení .....	35
Tab. 17 Čti čítač komunikačních událostí .....	35
Tab. 18 Čti záznam komunikačních událostí.....	36
Tab. 19 Zapiš více cívek.....	36
Tab. 20 Zapiš více registrů .....	37
Tab. 21 Sděl identifikaci.....	37
Tab. 22 Čti/Zapiš více registrů.....	39
Tab. 23 Tabulka ZAMĚSTNANCI.....	62
Tab. 24 Tabulka ACCESS_ROZVODNY .....	63
Tab. 25 Tabulka VYVODY .....	64
Tab. 26 Tabulka KVALIFIKACE.....	64
Tab. 27 Tabulka NAPĚTÍ.....	65
Tab. 28 Tabulka OBJEKT .....	65

Tab. 29 Tabulka ODDĚLENÍ .....	66
Tab. 30 Tabulka POVĚŘENÍ .....	66
Tab. 31 Tabulka ROZVODNY .....	66
Tab. 32 Unikátní identifikační .....	74
Tab. 33 Tabulka přístupů .....	75
Tab. 34 Tabulka přístupů .....	75
Tab. 35 Tabulka přístupů .....	75
Tab. 36 Tabulka přístupů .....	75
Tab. 37 Tabulka přístupů .....	76
Tab. 38 Tabulka přístupů .....	76
Tab. 39 Popis „Starting“ registrů v PLC .....	99
Tab. 40 Kódy a reference na datový typ master stanice .....	100
Tab. 41 Kódy a reference na datový typ slave stanice .....	100
Tab. 42 Tabulka definovaných a inicializovaných registrů .....	101
Tab. 43 Tabulka startujících registrů pro instrukční operace .....	102
Tab. 44 Tabulka rozsahu registrů v PLC .....	103
Tab. 45 Převodová tabulka registrů pro kód 05H .....	104
Tab. 46 Převodová tabulka registrů pro kód 06H .....	104
Tab. 47 Tabulka speciálních registrů v PLC Fatek FBs-40MCR2-D24 .....	105
Tab. 48 Rozložení Uživatelských registrů .....	114
Tab. 49 Rozložení Input Registrů .....	116
Tab. 50 Popis vodičů čtečky JA-80 .....	117
Tab. 51 Seznam vodičů připojených do převodníku Wie485 .....	117

## **SEZNAM PŘÍLOH**

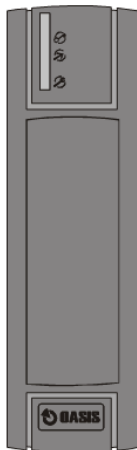
**PI JA-80N Venkovní čtečka RFID karet**

**PII JA-80N Venkovní čtečka RFID karet**

**PIII Obsah přiloženého CD**

# PŘÍLOHA P I: JA-80N VENKOVNÍ ČTEČKA RFID KARET

## JA-80N venkovní čtečka RFID karet



Čtečka je komponentem systému Oasis 80 firmy Jablotron. Lze použít k ovládání přístupu (dveřního zámku) nebo k ovládání zabezpečovacího systému. K ústředně Oasis se čtečka připojuje pomocí rozhraní WJ-80. Případně lze připojit k přístupovému systému AS-80.

Čtečka poskytuje data v protokolu Wiegand 26b.

### Instalace

Výrobek má montovat proškolený technik s platným certifikátem výrobce. Čtečka se umísťuje obvykle blízko vstupních dveří (vybavených elektrickým zámekem).

Čtečku je možné umístit venku (krytí IP-65).

1. povolte šroubek zadního plastu
2. namontujte zadní plast na určené místo (pružina sabotážního spínače musí být dobře stlačena)
3. kabel čtečky protáhněte do skříňky rozhraní WJ-80 nebo AS-80
4. šroubkem fixujte čtečku k zadnímu plastu
5. Zapojení a zprovoznění čtečky je popsáno v manuálu rozhraní WJ-80 nebo AS-80

### Kabel čtečky

vodič	signál
červený	+12V (napájení, cca 60mA)
zelený	D0 (výstup data Wiegand 26b)
hnědý	D1 (výstup data Wiegand 26b)
šedý a bílý	TMP (výstup rozpoznání sabotážního kontaktu)
žlutý	BZR (vstup k ovládání zvuku čtečky – sepnutím na GND generuje pípnutí)
modrý	GND (společná svorka napájení)
růžový	NEZAPOJEN

### Reset čtečky pro provoz s WJ-80

Pokud čtečka zapojená k ústředně Oasis nefunguje tak, jak je popsáno v manuálu WJ-80, není v základním nastavení z výroby. V takovém případě:

1. přepněte ústřednu Oasis do servisního režimu
2. vypněte zcela napájení ústředny (akumulátor i síť)
3. připravte si libovolnou přístupovou kartu PC-01 nebo PC-02
4. žlutý a hnědý vodič vnější čtečky odpojte ze svorek a tyto vodiče navzájem propojte
5. zapněte síťové napájení ústředny (čtečka začne pískat)
6. rozpojte žlutý vodič od hnědého (pískání ustane)
7. vezměte přístupovou kartu a opakovaně ji 4x přiložte a oddalte ke čtečce (po každém přiblížení se ozve krátké pípnutí) – pak počkejte až se ozve několik zapípání (zadali jste tak číslo 4)
8. nyní přístupovou kartu přiložte a oddalte ke čtečce 3x (po každém přiblížení se ozve krátké pípnutí) – pak počkejte až se ozve několik zapípání (zadali jste tak číslo 3)
9. vypněte napájení ústředny, zapojte zpět žlutý a hnědý vodič, zapněte napájení ústředny (včetně akumulátoru) a ověřte funkci čtečky

Při normálním provozu s ústřednou Oasis svítí na čtečce červená signálka. Přiblížení karty potvrzuje zablikání oranžové signálky.

Popis zapojení a nastavení pro přístupový systém AS-80 je popsáno v manuálu AS-80.

### Technické parametry

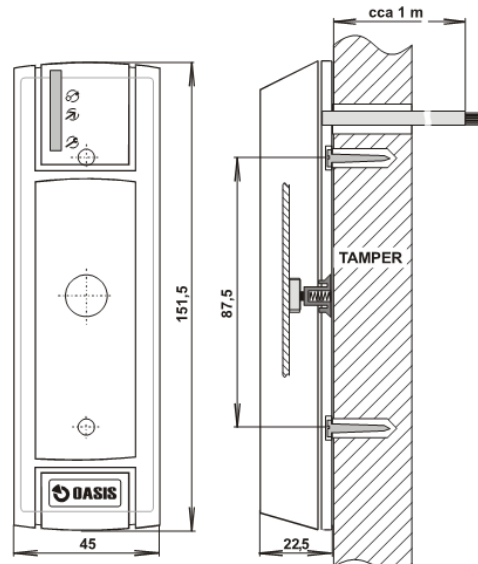
napájení	10 až 16V
klidový odběr	cca 60mA
krytí dle ČSN EN 60529	IP65
mechanická odolnost dle ČSN EN 50102	IK08
pracovní prostředí dle ČSN EN 50131-1	třída IV – venkovní všeobecné
Pracovní teplota	-25 až +60°C
zabezpečení dle ČSN EN 50131-1	stupeň 2
RFID karty	PC-01 či PC-02 Jablotron (EM UNIQUE 125kHz)
rozměry	46 x 150,5 x 22,5mm
podmínky provozování	ČTÚ č. VO-R/10/05.2006-22
délka přívodního kabelu	1m



Výrobek je navržen a vyroben ve shodě s na něj se vztahujícími ustanoveními: Nařízení vlády č. 426/2000Sb., je-li použit dle jeho určení. Originál prohlášení o shodě je na [www.jablotron.cz](http://www.jablotron.cz) v sekci poradenství.



**Poznámka:** Výrobek, ačkoliv neobsahuje žádné škodlivé materiály, nevyhazujte do odpadků, ale předejte na sběrné místo elektronického odpadu. Podrobnější informace na [www.jablotron.cz](http://www.jablotron.cz)



Jablotron s.r.o.  
Pod Skalkou 33  
466 01 Jablonec nad Nisou  
Tel.: 483 559 911  
fax: 483 559 993  
Internet: [www.jablotron.cz](http://www.jablotron.cz)

## PŘÍLOHA P II: PŘEVODNÍK WIE485

**Wie485**

Papouch s.r.o.

---

### **Wie485**

#### Katalogový list

Vytvořen: 5.10.2010

Poslední aktualizace: 12.11.2010 12:56

Počet stran: 8

© 2010 Papouch s.r.o.

---

### **Papouch s.r.o.**

Adresa:

**Strašnická 3164/1a  
102 00 Praha 10**

Telefon:

**+420 267 314 267-8  
+420 602 379 954**

Fax:

**+420 267 314 269**

Internet:

**[www.papouch.com](http://www.papouch.com)**

E-mail:

**[papouch@papouch.com](mailto:papouch@papouch.com)**

RSS:

**[www.papouch.com/paprss.xml](http://www.papouch.com/paprss.xml)**



## POPIS

Wiegand je standardní protokol, kterým komunikují čtečky bezkontaktních karet. Wie485 umožňuje tento protokol převést na standardní průmyslovou sběrnici RS485. Díky Wie485 tak lze snadno připojit standardní bezkontaktní čtečky s protokolem Wiegand k počítači PC. Linka RS485 je kompletně galvanicky oddělena od ostatních částí zařízení.

Výstupem z Wie485 je přímo číslo přiložené karty, čitelné přes RS485 protokolem MODBUS RTU. Wie485 je MODBUS Slave zařízení.

Modul Wie485 zpracovává data v protokolech Wiegand 26, 30, 32, 40, 42 (další varianty doplníme na přání).

## Použití

- Bezkontaktní čtečky
- Elektronické přístupové systémy
- Zabezpečovací systémy
- Modernizace stávajících přístupových systémů

## ZAPOJENÍ

- 1) Přepínačem na vrchní straně nastavte požadovaný typ protokolu Wiegand podle následující tabulky:

SW1	SW2	SW3	Typ protokolu
OFF	OFF	OFF	Wiegand 30
ON	OFF	OFF	Wiegand 26
OFF	ON	OFF	Wiegand 40
ON	ON	OFF	Wiegand 32
OFF	OFF	ON	Wiegand 42

tab. 1 – Nastavení komunikačního protokolu

- 2) Připojte Wie485 k nadřazenému systému pomocí vodičů RxTx+ (A) a RxTx- (B). Vodič RxTx+ propojte s protějším RxTx+, a podobně i RxTx-. Vodič GND můžete použít, pokud je propojovací kabel stíněný. V tom případě nezapomeňte zapojit stínění pouze na jedné straně kabelu!
- 3) Připojte ke konektoru Wiegand vodiče od bezkontaktního snímače.
- 4) Připojte napájení k zelené svorkovnici. Polarita je naznačena na štítku. (Wie485 má integrovanou ochranu proti poškození přepólováním napájecích svorek.)

**MODBUS RTU – ROZLOŽENÍ PAMĚTI****Holding Register****Postup konfigurace**

Následující postup je třeba provést při konfiguraci. Jinak není možné do registrů zapisovat.

- 1) Vypněte napájení.
- 2) Přepněte přepínač 8 do polohy ON.
- 3) Zapněte napájení.
- 4) Nyní Wie485 komunikuje (bez ohledu na aktuální nastavení) rychlostí 9600 Bd, 8 datových bitů, bez parity, 1 stopbit a ID zařízení je 1.
- 5) Proveďte zápis změn. (I poté v tomto režimu konfigurace komunikuje Wie485 stále s výše uvedenými parametry.)<sup>1</sup>
- 6) Vypněte napájení.
- 7) Přepněte přepínač 8 do polohy OFF.
- 8) Zapněte napájení. (Nyní Wie485 komunikuje s nově nastavenými parametry.)

Adresa	Přístup	Funkce	Popis
1	čtení zápis	0x03 0x10	Pozice pro volné použití
2	čtení zápis	0x03 0x10	ID zařízení (číslo 1 až 247)
3	čtení zápis	0x03 0x10	Komunikační rychlost. Kód dle následující tabulky: 3 - 9600 Bd (tato hodnota je nastavena jako výchozí) 4 - 19200 Bd 5 - 38400 Bd 6 - 57600 Bd 7 - 115200 Bd
4	čtení zápis	0x03 0x10	Režim sériové linky. Kód dle následující tabulky: 0 – 8 datových bitů, bez parity, 1 stopbit 1 – 8 datových bitů, sudá parita, 1 stopbit 2 – 8 datových bitů, lichá parita, 1 stopbit 3 – 8 datových bitů, bez parity, 2 stopbity 4 – 8 datových bitů, sudá parita, 2 stopbity 5 – 8 datových bitů, lichá parita, 2 stopbity

<sup>1</sup> Pro konfiguraci můžete použít například software [Modbus Configurator](http://ModbusConfigurator.com), který je ke stažení na [www.papouch.com](http://www.papouch.com). Jen je třeba vzít v úvahu, že tento software nepočítá s tím, že Wie485 v režimu konfigurace komunikuje stále stejnou rychlostí. Je tedy potřeba nejdříve nastavit komunikační rychlost 9600 Bd a ID 1 a poté změnit parametry na nové.



**TECHNICKÉ PARAMETRY****Sběrnice RS485:**

Konektor ..... Odnímatelná šroubovací svorkovnice  
 Typ linky ..... RS485  
 Galvanické oddělení ..... ano, indukční  
 Rychlost ..... 9600 Bd (ve výchozím nastavení)  
 Počet datových bitů ..... 8  
 Parita ..... není  
 Počet stopbitů ..... 1  
 Rezistory definující stav linky ..... 22 k $\Omega$ <sup>2</sup>  
 Zakončovací rezistor ..... 120  $\Omega$ <sup>3</sup>

**Wiegand:**

Konektor ..... Odnímatelná šroubovací svorkovnice  
 Typ linky ..... Wiegand 26, 30, 32, 40, 42 (dle nastavení)

Pin	Popis
Data 1	Signál Data 1 pro čtečku
Data 0	Signál Data 0 pro čtečku
GND	Zem komunikační linky
+U <sub>OUT</sub>	Výstup napájecího napětí pro čtečku <sup>4</sup>

tabulka 2 – zapojení konektoru pro Wiegand

**Napájení:**

Napájecí napětí ..... stejnosměrné napětí 8 až 30 V  
 Proudový odběr ..... typ. 20 mA při 15 V (bez napájení čtečky)

<sup>2</sup> Tyto rezistory jsou připojené trvale.<sup>3</sup> Z výroby není připojen. Lze jej připojit uživatelsky propojkou S1 uvnitř zařízení.<sup>4</sup> Pokud je napájecí napětí vyšší než 15 V, je na tomto výstupu přítomno napětí 12 V.

Pokud je napájecí napětí menší než 15 V, je na tomto výstupu nižší napětí, než je napájecí napětí.

## **PŘÍLOHA P III: OBSAH PŘÍLOŽENÉHO CD**

Obsah jednotlivých adresářů v souboru **komplet.zip** přiloženého CD:

**\fulltext.pdf** - text této diplomové práce ve formátu DOC a PDF

**\Access Com Port – Modbus v3.88** - zdrojový kód hlavní aplikace napsané v prostředí Borland Builder 6 + přiložený zkompilovaný soubor \*.exe

**\www** - kompletní adresář projektu Access\_Energo s kódy pro webové rozhraní ve formátu, \*.html, \*.php a \*.txt vytvořený v textovém editoru PSPad

**\energo\_access** – adresář aplikace Uniform Server (MySQL) s uloženými tabulkami pracovníků a objektů ve formátu \*.MYI, \*.MYD a \*.frm

**\Access Energo Modbus** – kompletní diagram naprogramovaného programovacího automatu v prostředí WinProLadder ve formátu \*.pdw