Čestné prohlášení

Prohlašuji, že jsem svou maturitní práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Písku dne: podpis: .....

Poděkování

Děkuji Ing. Miroslavu Paulovi za možnost uskutečnit tuto práci.

Děkuji firmě Amit, za dvě školení, které jsem mohl v Praze absolvovat.

Anotace

Tato práce řeší řízení nesené brány pomocí PLC AMiNi-ES od firmy Amit. Otevírání brány je řešeno pomocí RFID karet nebo přívěsků. Každá karta nebo přívěsek je zařazen do jedné ze čtyř přístupových skupin. Časové plány jednotlivých skupin lze nastavovat v programu ViewDet. V programu lze také nastavit, v jakých časových intervalech zůstane brána otevřená a kdy se bude automaticky zavírat.

Summary

This piece of work solves control of moveable gate by PLC AMiNi-ES produced by company named Amit. The gate opening is solved by RFID cards or specific pendants. Every card or pedant is classified into four acces groups. The time plan of every group is able to be set in the program named ViewDet. This program is able to set the time intervals when the gate keeps opening and when it will automaticly close.

Klíčová slova

RFID , DetStudio, ViewDet

Obsah

1 Úvod

1.1 Popis zadané úlohy

Tato práce má vyřešit otevírání a automatické zavírání nesené brány. Asi nejjednodušší možnost je ovládání pomocí PLC (programovatelný logický automat). Po předchozích zkušenostech firmy, jsem zvolil PLC od firmy Amit. Konkrétně typ AMiNi – ES. Poté jsem musel vyřešit jak od sebe rozeznat jednotlivé uživatele. Jedna možnost je pomocí kódového zámku a druhá pomocí RFID čipů nebo přívěsků. Z cenových důvodů byla zvolena druhá možnost. K ovládání celé brány je použit motor s řídící jednotkou (Bernal SP Gatemas BASIC 230V) distribuovaný firmou AZ pohony.

Každý uživatel je zařazen do jedné ze čtyř přístupových skupin. Jednotlivé skupiny je možné nakonfigurovat pomocí ovládacího a vizualizačního prostředí ViewDet. V tomto programu dále můžeme nastavit, v jakých časových intervalech zůstane brána otevřená a kdy se začne automaticky zavírat. Lze zde přesunout uživatele z jedné přístupové skupiny do jiné přístupové skupiny (každá přístupová skupina určuje, kdy daný uživatel může otevřít bránu).

1.2 Hardwarová část

Práce zahrnuje montáž ozubeného hřebenu na bránu, montáž a zapojení řízeného motoru tj. motor Bernal SP Gatemas BASIC 230V distribuovaný firmou AZ pohony. Zabudování a zapojení RFID čteček a dvou párů fotobuněk (aby při zavírání nedošlo ke kolizi s překážkou). Propojení PLC a motoru pomocí relé, připojení RFID čteček k PLC přes sériovou komunikační linku RS 232. Pro jednodušší dekódování byl použit univerzální datový převodník PRWA2, který převede datový formát Wiegand (vysílaný z RFID čteček) na formát ASCII s komunikační rychlostí 9600 Bd, datovou délkou 8bitů, jedním stop bitem a žádnou paritou. Jedná se o poloduplexní provoz (v jednom okamžiku může probíhat komunikace pouze jedním směrem, to je v našem případě zcela postačující, jelikož zpětnou komunikaci nepotřebujeme).

1.3 Softwarová část

Softwarová část zahrnuje instalaci operačního systému NOS verze 3.58, obsluhu digitálních vstupů a výstupů, komunikaci přes sériové rozhraní RS 232. Naprogramování časových plánů, automatického zavírání v určitou dobu, dekódování přijímaného řetězce znaků a porovnávání s databází přístupových kódů. Každý přístupový kód, který je uložen v matici přístupových kódů má jednu ze čtyř priorit, která určuje, v jakých časových intervalech může uživatel s určitou prioritou otevřít bránu. Vývojové prostředí, kde se PLC programuje, se nazývá DetStudio. Jednoduché řízení a nastavování lze provádět v programu ViewDet. Oba tyto programy jsou navrhnuty speciálně pro programování a řízení PLC od firmy Amit.

2 Začátky s DetStudiem

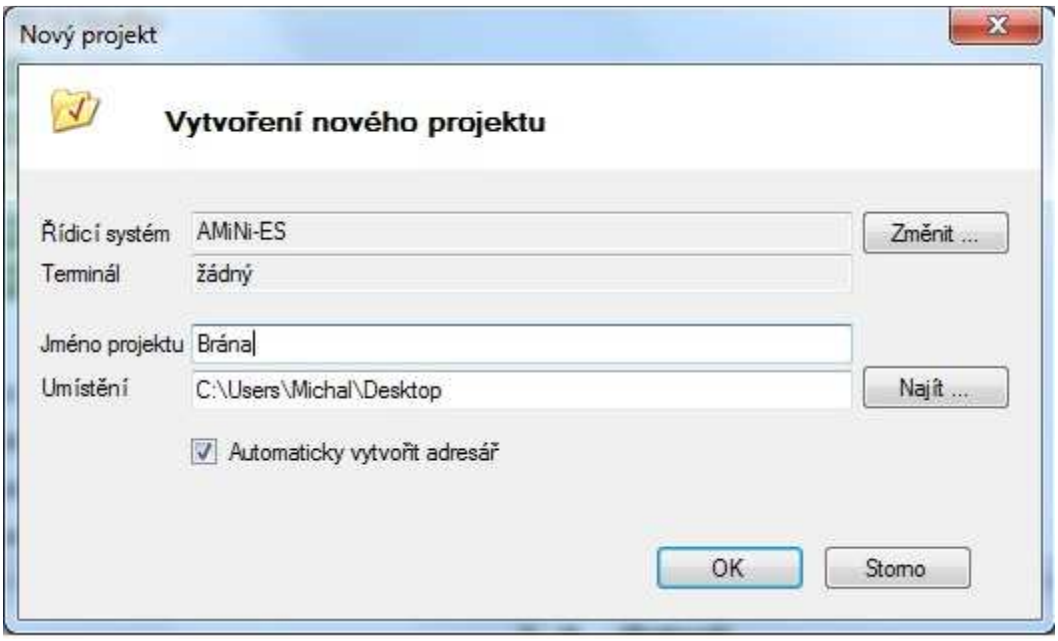
2.1 Co je DetStudio

DetStudio je vývojové prostředí určené výhradně pro řídící automaty z produkce firmy Amit. Tento program je možné si po registraci na stránkách www.amit.cz bezplatně stáhnout. Pro rychlé osvojení tohoto programu byl napsán návod na obsluhu (průvodce první aplikací). Tento návod je velice podrobný, vysvětluje základní pojmy používané v programu. Obsahuje průvodce instalací DetStudia, ovládání prostředí, založení nového projektu, nahrání operačního systému nos, nastavení komunikace mezi PC a PLC pomocí sériového rozhraní nebo Ethernetu, ukázku jednoduchého programu a to i s naprogramováním obrazovek (v našem případu je to zbytečné, jelikož AMiNi – ES patří mezi levnější PLC od firmy Amit a tudíž nemá displej). Dále uskutečňuje firma Amit každý měsíc dvě školení pro svoje zákazníky (rozděleno na základní školení a rozšířené školení). Školení je velmi rozsáhlé, podrobné a lehce pochopitelné. Obě tato školení jsem absolvoval.

2.2 Založení nového projektu

Po otevření DetStudia vybereme z hlavního menu *Soubor->Nový*. Poté se nám objeví okno, kde musíme vybrat řídicí systém, to uděláme tak, že klikneme na tlačítko Změnit a vybereme Standardní řídicí systémy -> AMiNi, ADiR. Zde vybereme AMiNi – ES s interním terminálem, následně potvrdíme tlačítkem OK. Dále nový projekt pojmenujeme

a určíme cestu na disku, kam ho chceme uložit (Obrázek 2.2). Po potvrzení se objeví okno s parametry projektu. Zde zatím nemusíme nic nastavovat.



Obrázek 2.2

2.3 Nahrání operačního systému NOS

Do každého nového PLC od firmy Amit se musí nahrát operační systém NOS. Před nahráním operačního systému je potřeba si vytvořit kabel 232RP, který má na jedné straně konektor Cannon 9 (zásuvka) a na straně druhé konektor RJ45. Návod zapojení je v příloze 1. Po vytvoření propojovacího kabelu propojíme PLC a s PC.

Dále musíme nastavit komunikaci. To provedeme tak, že ve vývojovém prostředí DetStudio v roletovém menu zmáčkneme *Přenos* a následně vybereme *Nastavení komunikace*. Podle nastavení kombinace prvních pěti pinů softwarového DIP přepínače, který nalezneme na PLC, určíme adresu stanice (0..31), jedná se o binární kombinaci těchto přepínačů (pozor na obrácené pořadí, váha není 16,8,4,2,1 ale 1,2,4,8,16). Kombinací 6 a 7 pinu softwarového DIP přepínače nastavíme jednu ze čtyř rychlostí přenosu. Stejnou rychlost nastavíme i ve vývojovém prostředí. Dále nastavíme 8 pin softwarového DIP přepínače do stavu OFF (to určuje, že budeme operační systém nahrávat přes sériové rozhraní RS 232. Poté už stačí vybrat správný komunikační port.

Problém by mohl nastat, pokud nemá váš počítač rozhraní 232 a použili bychom převodník USB => RS232. Poté se musí nastavit v komunikaci místo Sériové linky Servisní mód, protože USB nezvládá rychlost protokolu DB-Net. Dále výrobce doporučuje nastavit si převodník na Com1 – Com8. Ve Windows se to nastavuje v Ovládacích panelech -> Správce zařízení.

Dále musíme připojit PLC ke zdroji DC napětí 24 V. Propojíme kabelem 232RP a můžeme začít s nahráváním operačního systému NOS. To provedeme tak, že v roletovém menu klikneme na Přenos a vybereme Nahrát NOS. V této chvíli se spustí průvodce nahráváním operačního systému. Pokud jste vše propojili a nastavili správně, průvodce potvrdí správné propojení a začne nahrávání operačního systému. Během nahrávání po vás průvodce požaduje přepnutí jednotlivých pinů hardwarového DIP přepínače. Po úspěšném dokončení přenosu operačního systému bychom měli po vybrání *Přenos -> Indentifikace* vidět úspěšné propojení PLC s vaším počítačem.

2.4 Založení procesů, proměnných či aliasů

2.4.1 Založení procesů

Po absolvování všech předchozích kroků se už můžeme pustit do založení procesu. Proces je část programu, která se vykonává v přednastavených periodách. Nejlehčí varianta založení nového procesu je ve stromové struktuře okna projektu dvakrát kliknout na *Procesy*, poté stačí stisknout tlačítko INSERT, kterým založíme nový proces. Nový proces pojmenujeme dle libosti a vybereme v roletovém menu Proces_1 – Proces_15 (číslo určuje prioritu procesů => Proces_1 proběhne jako první). Dále vybereme typ procesu. Vybrat můžeme strukturovaný text (ST), reléové schéma (RS) nebo jazyk logických adres (LA). Pro tuto úlohu doporučuji jazyk ST, kvůli jednoduchému vkládání a konfigurování modulů. Poté už stačí vyplnit, s jakou periodou se tento proces bude opakovat (udává se v milisekundách), dopíšeme komentář a potvrdíme tlačítkem OK.

Následně vybereme ze stromové struktury v okně projektu vytvořený proces a můžeme začít programovat.

2.4.2 Založení proměnných

Zakládat proměnné můžeme několika zbůsoby. Jedna možnost je vybrat ve stromové struktuře v okně projektu složku *Databáze* a následně *Proměnné*. Poté stačí stisknout klávesu INSERT a pojmenovat proměnnou (nesmí obsahovat diakritiku, mezery a číslo na začátku), následně vybereme typ proměnné tj. Integer, Long nebo Float, stejné datové typy můžeme používat i u matic (pokud vybereme matici, musíme zadat její rozměr tj. řádky x sloupce, čísluje se od 0). WID je jedinečný číselný identifikátor proměnné, vyplňuje automaticky (výrobce nedoporučuje editovat). Pokud chceme, aby proměnná měla od začátku určitou hodnotu tak ji vyplníme do políčka Init. Poté už stačí vyplnit číslo stanice a komentář k proměnné. Pokud uživatel chce, aby proměnná měla po restartu inicializační hodnotu zaškrtne okénko Warm. Po potvrzení tlačítkem OK se námi deklarovaná proměnná vytvoří.

Pro rychlou tvorbu nové proměnné můžeme použít klávesu F12.

2.4.3 Založení aliasů

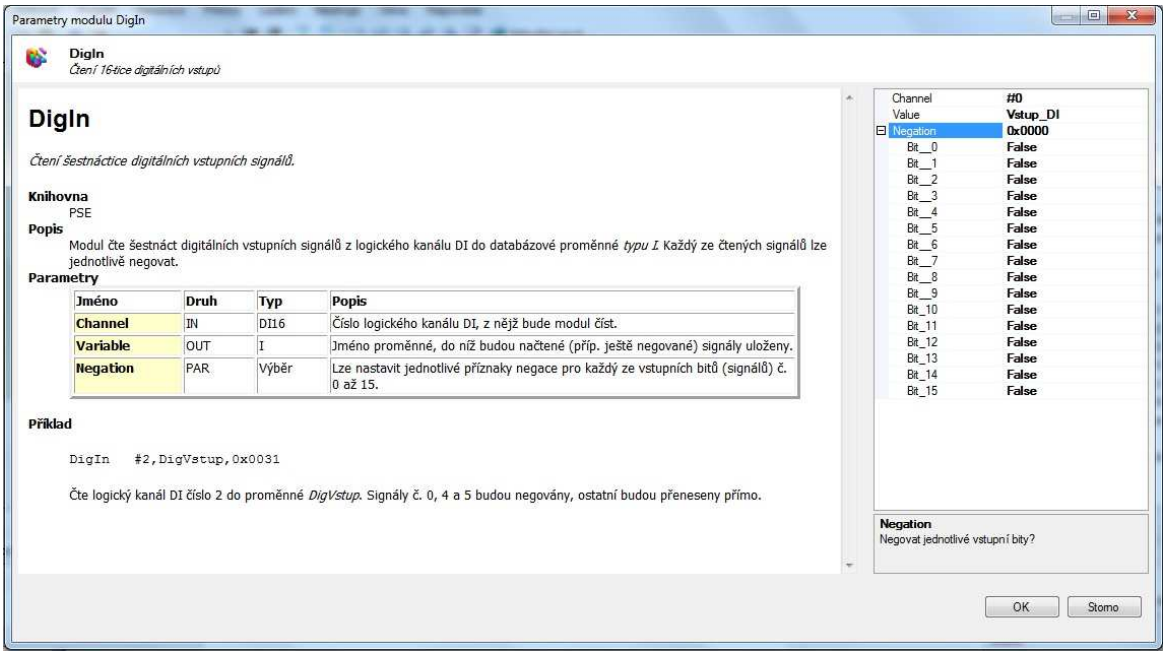
Jedna možnost založení aliasu je vybrat ve stromové struktuře v okně projektu složku *Databáze* a následně *Aliasy*. Stisknutím tlačítka INSERT vytvoříme nový alias. Aliasy se liší od proměnné tím, že mají před sebou zavináč (@). Každý alias se váže k určité proměnné, tudíž musíme vybrat určitou proměnnou, ke které budeme daný alias vázat. Poté stačí vyplnit komentář a potvrdit tlačítkem OK.

Rychlejší alternativou zakládání aliasů a klávesová zkratka ALT+F12.

2.5 Obsluha V/V, seznámení s AutoInspektorem

2.5.1 Obsluha V/V

Obsluha vstupů a výstupů je řešená pomocí modulu **DigIn** a **DigOut**. Poté, co se napíše jméno modulu, je nejjednodušší možnost, jak daný modul správně vyplnit stisknoutím CTRL+M. Poté se objeví okno, kde se zobrazí nápověda a snadno se dají vyplnit všechny potřebné parametry (viz obrázek 2.5.1).



Obrázek 2.5.1

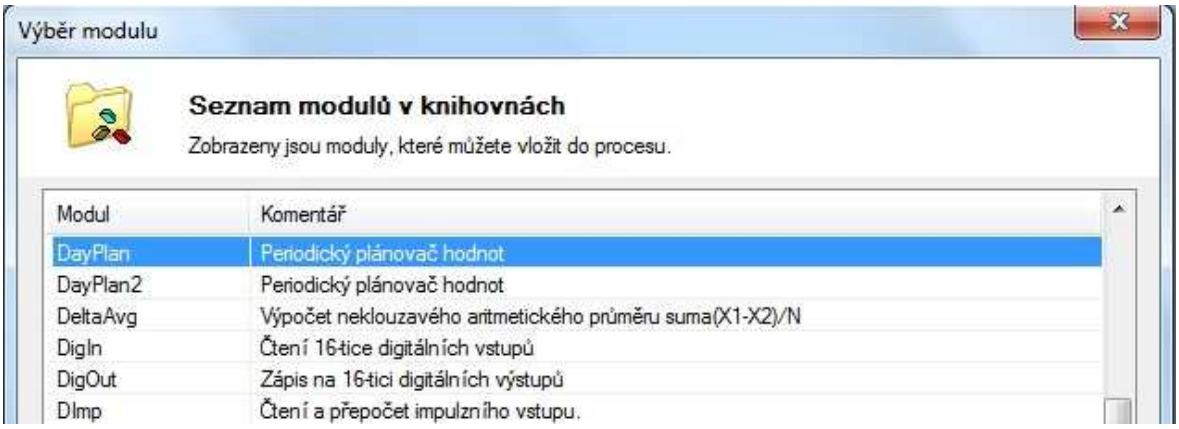
Na obrázku je vidět jak jednoduše můžeme nastavit načítání vstupních digitálních impulsů. Jednotlivé vstupy můžeme dle potřeby negovat. S jednotlivými vstupy můžeme dále pracovat dvěma způsoby. Jednodušší možnost je napsat jméno *proměnné.X* (kde X určuje s kolikátým bitem pracujeme) nebo můžeme založit alias, který se váže k proměnné, kterou jsme zadali do modulu a při zakládání aliasu určíme, s jakým bitem vstupní informace chceme pracovat. Tento alias můžeme pojmenovat libovolně.

2.5.2 AutoInspektor

AutoInspektor slouží k zobrazení aktuálních stavů proměnných, aliasů či maticí v PLC. Nejčastější použití je při odlaďování aplikace. AutoInspektor lze také použít pro editaci aktuálních hodnot proměnných, aliasů nebo matic.

2.6 Práce s moduly

Moduly jsou základním stavebním kamenem, pokud chcete programovat v jazyce ST. Vybrat modul lze pomocí klávesové zkratky CTRL + I. Poté se zobrazí seznam všech modulů s krátkým komentářem (obrázek 2.6).



Obrázek 2.6

2.7 Nahrání programu do stanice

Pokud už program vytvoříme, nahrání do stanice je jednoduché. V hlavním menu vybereme *Generace* a následně *Generuj vše*. Při častém generování programu za účelem hledání syntaktických chyb v programu je rychlejší a pohodlnější používat klávesu F6, která provede stejnou úlohu. Po chvilce se program vygeneruje. Zobrazí se nám, kolik místa daný program zaplní v paměti FLASH a v paměti RAM. Pokud chceme program přenést do stanice, zvolíme možnost přenést do stanice. Druhá možnost je přenos zrušit tlačítkem zavřít. Pokud zvolíme možnost přenést do stanice, tak se nám zobrazí okno, které popisuje stanici, do které budeme program přenášet. Dále se zobrazí cesta k programu, který budeme přenášet a můžeme si zvolit, zda chceme provádět zálohu a obnovu proměnných ve stanici a zda chceme nahrát zdrojový kód (tj. že se celý zdrojový kód i s komentáři nahraje do stanice).

Pokud zvolíme možnost nahrát zdrojový kód, tak tento zdrojový kód můžeme ze stanice kdykoliv vyčíst. Vyčtení zdrojového kódu lze chránit heslem. Vyčtení lze provést tak, že v menu vybereme *Přenos* a následně *Vyčíst zdrojový kód*. Pokud jsme při nahrávání zadali heslo tak ho musíme zadat i při vyčtení. Při neznalosti hesla není možné program vyčíst.

3 Složité programové části

3.1 Komunikace pomocí RS 232

Toto je nejsložitější část programu. Je třeba použít několik modulů - **SubInst, ComInit** a **ComRead.**

3.1.1 Ukázka programu

```
//Načtení přijatého rámce znaků
:01000 SubInst Lib0
:02000 ComInit 0x0000, 0, 9600, 8, 0, 1, :01000, :NONE,
:NONE, :NONE, RxBuf, NONE
//Podprogram obsluhující komunikaci
ComRead :02000, Buffer, IndexBuffer, IndexBuffer, NONE
Let @kontrola = if ( (IndexBuffer==17), true, false)
If @kontrola
     For I, 0.000, 256.000, 1.000
     Let Nac_hod[0,I]=Buffer[0,I]
     Endfor
     For I, 0.000, 256.000, 1.000
     Let RxBuf[0,I]=0
     Endfor
     For I, 0.000, 256.000, 1.000
     Let Buffer[0,I]=0
     Endfor
     Let IndexBuffer=0
     Let Poc_nac=Poc_nac+1
     Let @Nacteni=true
Endif
```

Obrázek 3.1.1

3.1.2 Popis Programu (obrázek 3.1.1)

Načtení přijatého rámce znaků se musí vložit do procesu ProcINIT. Modul **SubInst** slouží k vytvoření podprogramu. Číslo :01000 před modulem udává návěští tohoto modulu. Tento modul nás pošle do podprogramu jménem Lib0.

Modul **ComRead** slouží pro načtení přijatých znaků z přijímacího bufferu. Proměnná *Buffer* slouží jako Buffer, do kterého se načtou znaky z přijímacího bufferu modulu **ComInit**. [1] Proměnná *Buffer* je matice, do které se ukládá přijímaný datový rámec. Proměnná *Index_Buffer* udává místo kam se přijatý rámec znaků bude do matice ukládat. Až proměnná Index Buffer dosáhne hodnotu 17, znamená to, že už máme načtený celý

datový rámec, se kterým mužeme dál pracovat. Matici, která obsahuje datový rámec zkopírujeme do matice *nac_hod* a následně *Buffer* a *Index_Buffer* vynulujeme. Po provedení tohoto podprogramu se vrátíme na návěští :02000.

Modul **ComInit** je modul, pomocí kterého nastavujeme všechny komunikační parametry. Tyto parametry udává univerzální datový převodník PRWA2 (informace o převodníku v příloze 2). V tomto modulu musíme nastavit komunikační tychlost na 9600 Bd, datovou délku na 8 bitů, počet stop bitů na jeden a nulovou paritu.

3.2 Vytvoření a zápis do databáze

Pro databázi se musí vytvořit matice typu integer o rozměrech 100x18 (100 řádků x 18 sloupců) do prvních 17 sloupců se bude ukládat jedinečný kód RFID čipu. Do posledního sloupce se bude zapisovat priorita určitého uživatele (číslo 0 – 3). Jelikož ruční zápis do databáze by byl zdlouhavý a mohly by nastat chyby v přepsání, je nejlepší použít uživatelky nastavitelný přepínač tj. 10 pin softwarového DIP přepínače. Pro ovládání tohoto přepínače je vytvořen speciální modul **Cond**. Plní stejnou funkci jako If s tím rozdílem, že kontroluje pouze uživatelsky nastavitelný přepínač. Poté stačí napsat jednoduchý program, který bude zapisovat přijatý datový rámec do databáze kódů. Program, který bude porovnávat přijatý datový rámec s kódem v databázi, musí být také v podmínce **Cond,** ale až za příkazem **ElseCond**, aby bylo jednoznačně určeno kdy se má porovnávat načtený kód s kódem v databázi a kdy se bude načtený kód připisovat do databázové matice.

3.2.1 Algoritmus pro zapsání přijatého rámce do databáze

```
//Zápis do databáze při přepnutí uživatelského přepínače
Cond NONE.0, 1000, 0
     If @nacteni
          For x, 0.000, 16.000, 1.000
          Let Databaze[Poc_nap,x]=Nac_hod[0,x]
          Endfor
          Let Poc_nap=Poc_nap+1
     Endif
     Let @nacteni=false
EndCond
```

Obrázek 3.2.1

3.3 Porovnání přijatého datového rámce s databází

Pokud už v databázi bude zapsán jedinečný kód některého RFID čipu, můžeme přistoupit k porovnávání. Algoritmus spočívá v tom, že musíme porovnat načtený kód s kódy, které máme uložené v databázi tj. porovnání dvou matic o rozměrech 100x17 (databáze) a 1x17 (načtený kód). Cyklus pracuje tak, že porovnává stejné sloupce porovnávaných matic. Pokud nalezne shodu, tak inkrementuje proměnnou *X* (sloupce) a pokud se shoduje všech 17 sloupců, tak se nastaví alias *@Login_ok* do logické úrovně 1“. Pokud se ovšem některý ze sloupců neshoduje, tak se proměnná *X* vynuluje a inkrementuje se proměnná *Y* (řádky). Pokud načtený kód odpovídá nějakému kódu v databázi, zapíše se řádek, na kterém se tento kód nachází do proměnné *Index_data*.

Do posledního sloupce v databázi zapíšeme číslo od 0 do 3. Toto číslo určuje prioritu jednotlivých uživatelů. Podle této priority majitel určeného čipu otevře nebo neotevře v určitou hodinu a v určitý den. Nastavení časových plánu se provede v programu ViewDet.

3.3.1 Algoritmus porovnávání

```
//Porovnání přijatého rámce znaků s rámcem znaků v databázi
Cond NONE.0, 1000, 0
elsecond
     If @nacteni
          Let Index_data=0
          Let X=0
          Let Y=0
          For Y, 0.000, 99.000, 1.000
          For X, 0.000, 16.000, 1.000
          Let
@Rovnost=If(Databaze[Y,X]==Nac_hod[0,X],true,false)
          If @Rovnost
               Let @Login_ok=true
          Else
               Let @Login_ok=false
               Let X=0
               Break
          Endif
          EndFor
          If @Login_ok
               Break
          Endif
          Endfor
          If @Login_ok
               Let Index_data=Y
```

```
                Let Pristup_1=Databaze[Y,17]
            Endif
            Let @nacteni=false
        Endif
EndCond
```

Obrázek 3.3.1

3.4 Nastavení časových plánů

Nastavit časové plány jde v programu DetStudio pomocí modulu **DayPlan** nebo **DayPlan2**. V našem případě postačí modul **DayPlan**. Nejdříve nastavíme 4 časové plány pro 4 priority přístupů. Pro každý časový plán je třeba vytvořit dvě matice a jednu proměnnou typu integer. Matice *Timex_X* a *Values_X* mají rozměry 7x7. Do proměnné *Rezim_X* bude zapsánu 0 nebo 1 podle nastavení časových plánů. Poté bude potřeba nastavit ještě jeden časový plán, který bude určovat, v jakém časovém intervalu zůstane brána otevřená, pokud ji někdo otevře a kdy se brána začne automaticky zavírat.

3.4.1 Ukázka nastavení časového plánu

```
DayPlan 0x0000, 7, 0x0001, 0x0002, 0x0004, 0x0008, 0x0010, 0x0020, 0x0040, 0x0000, NONE, Times_0, Values_0, Rezim_0
```

3.5 Nastavení automatického zavírání

Jeden z časových plánů musí řešit, kdy brána zůstane otevřená a kdy se bude automaticky zavírat. Pokud bránu otevře někdo v době, kdy má být zavřená, nebylo by vhodné, aby se brána ihned začala zavírat. Proto je potřeba naprogramovat zpoždění zavírání. Pokud víme, že tento proces má periodu 1000 ms (1 sekundu) a chceme nastavit prodlevu 2 minuty, stačí inkrementovat proměnnou *pomoc_prom* v každém procesu. Poté už stačí porovnávat, jestli je proměnná větší nebo rovna 120 a pokud ano, tak se brána začne zavírat. Jakmile se brána začne zavírat, tak se proměnná vynuluje.

3.5.1 Algoritmus zpoždění automatického zavírání

```
Let @zpoz=If((Rezim_4==0)and(@otevreno==true),true,false)
If @zpoz
     Let Pomoc_prom=pomoc_prom+1
     Let @vystup_0=if((Pomoc_prom>120),true,false)
     else
     Let Pomoc_prom=0
endif
```

4 ViewDet

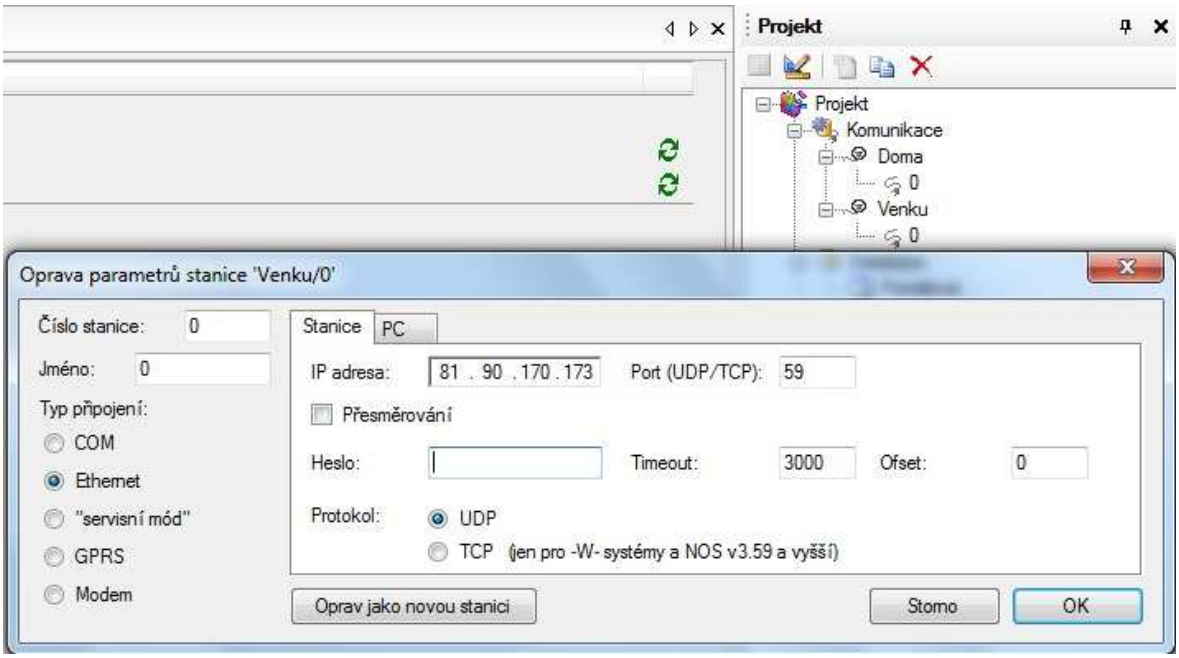
4.1 Seznámení s programem

ViewDet je servisní a vizualizační nástroj, který je určen pro monitoring a ladění aplikací běžících v řídicích systémech firmy AMiT, sběr a archivaci dat jimi vytvořených a v neposlední řadě i k vizualizaci těchto dat. Tím doplňuje a rozšiřuje možnosti vývojového prostředí DetStudio. [2]

4.2 Nastavení komunikace z domova nebo z venku

Nejjednodušší nastavení komunikace se provede tak, že v okně *Projekt* (to se nachází v pravé části programu ViewDet) otevřeme složku *Projekt*. Po otevření se zobrazí složka *Komunikace*. Na tuto složku klikneme pravým tlačítkem myši a zvolíme možnost přidat profil. Tímto se nám vytvoří profil, do kterého si musíme přidat naší stanici. Po přidání stanice můžeme začít zapisovat do tabulky IP adresu stanice. Pokud má stanice přiřazenou veřejnou IP adresu, je třeba vytvořit ještě jeden profil, který bude deklarovat nastavení připojení přes veřejnou IP adresu. Konkrétní nastavení veřejné IP adresy je vidět na obrázku 4.2. Zda stanice komunikuje, či nekomunikuje, můžeme zjistit tak, že v hlavním menu vybereme *Okno->Komunikace*. Tímto by se nám měla otevřít scéna s komunikací.

Komunikace nemusí odpovídat, pokud je zastavená. Spustit ji můžeme tak, že vybereme z hlavního menu *Projekt->Spustit komunikaci*. Alternativní možnost spuštění komunikace je stisknutí tlačítka F8. Poté musíme mít také nastavený správný komunikační profil. Podle toho jestli se do stanice připojujeme přes domácí síť nebo přes internet, zvolíme buď profil Doma nebo profil Venku.



Obrázek 4.2

4.3 Načtení proměnných

Pokud se nám povedla úspěšně nastavit komunikace, můžeme přejít k načtení proměnných. V tomto případě máme opět možnost výběru, buď načteme proměnné rovnou ze stanice, nebo můžeme načíst proměnné ze souboru tj. z programu vytvořeného ve vývojovém prostředí DetStudio. Načtení provedeme tak, že v hlavním menu programu ViewDet vybereme *Projekt->Data->* a vybereme, zda načteme proměnné ze stanice nebo ze souboru.

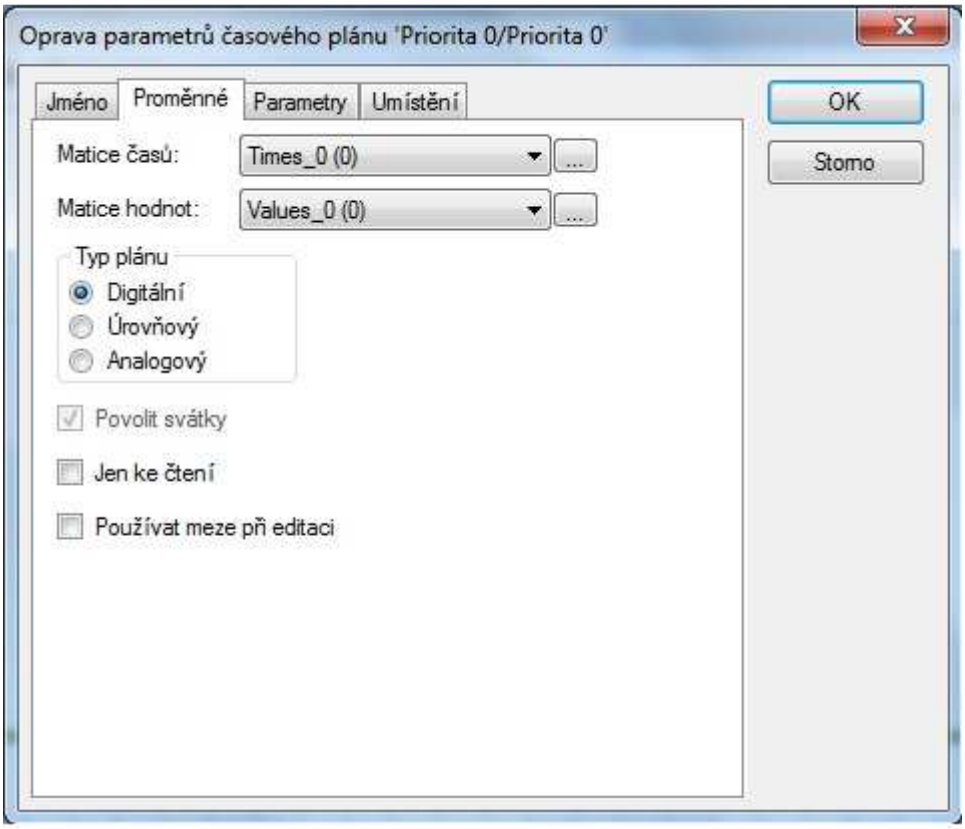
Alternativa pro načtení proměnných ze stranice je klávesová zkratka Alt+F5 a pro načtení proměnných ze souboru to je zkratka Alt+F6.

4.4 Nastavení scén

Plochy, na kterých si můžeme nechat zobrazit archiv, časový plán, matici, proměnnou, nebo jenom komentář, se nazývají scény. Scény můžeme připadat tak, že v okně projektu zmáčkneme pravým tlačítkem myši na scény a vybereme přidat scénu. Mezi jednotlivými scénami můžeme přepínat pomocí záložek, které se nacházejí v horní části programu. Na scénu nebo do pozadí scény lze vložit pouze statický obrázek. Pracovat s animacemi nebo videem tento program bohužel neumí.

Na scénu vložíme proměnnou, matici, časový plán nebo komentář tak, že nejprve vybereme scénu, na kterou chceme prvek vložit, poté klikneme někam do scény pravým tlačítkem. Zvolíme možnost přidat a následně vybereme objekt, který chceme vložit. Editaci vložených prvků se provede tak, že na prvek klikneme pravým tlačítkem myši a vybereme možnost *Editace*. V editaci lze nastavit třeba přesnou polohu prvku na scéně.

Při vkládání časových plánů je třeba vybrat správnou matici časů a matici hodnot a typ plánu je nutné přepnout na digitální (Obrázek 4.4).



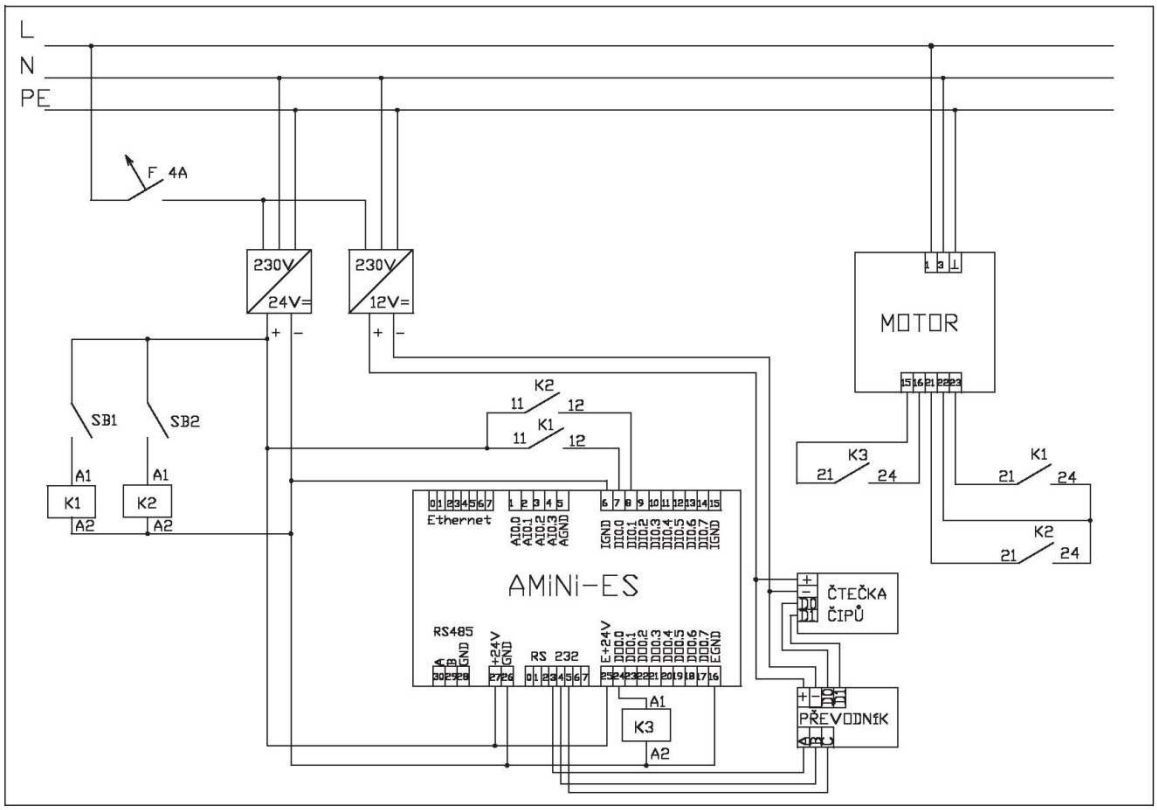
Obrázek 4.4

Při pozdějším nastavování časových plánu je třeba pamatovat na to, že změna časového plánu se projeví až při prvním časovém zlomu. Proto je dobré si nechat zobrazit výstup časového plánu jako proměnnou a dle potřeby tuto proměnnou přepsat, abychom nemuseli čekat na prvním časový zlom, který je nastavený v časovém plánu (pokud je časový plán pro celý den v logické úrovni 0“ nebo 1“ tak kontrola výstupu nastává pouze v čase 00:00).

5 Realizace

5.1 Návrh schématu

Návrh schématu obsahuje propojení PLC s motorem. Zapojení dvou RFID čteček přes převodník PRWA2 do PLC (sériová komunikace RS 232). Propojení PLC s motorem je realizováno pomocí Relé Finder typ 40.52. Pro napájení PLC, čteček a převodníku jsou potřeba dva zdroje. Použity byly spínané zdroje na DIN lištu značky MEAN WELL. Jelikož jsme použili spínač polohy, který používá řídící jednotka, je třeba vrátit informaci o otevření či zavření řídící jednotce motoru. Motor s řídící jednotkou nese označení SP Gatemas. BASIC 230 (manuál v příloze 3). Jako čtečky RFID čipů byly použity čtečky od značky Sebury konkrétně typ R2-H/EM/MF card reader. Kompletní schéma zapojení na obrázku 5.1.



Obrázek 5.1

5.2 Fotobuňky

Z důvodu bezpečnosti bylo nutné doplnit systém o jeden pár fotobuněk, jelikož k motoru s řídící jednotkou je dodáván pouze jeden pár a při pohybu vyšších nákladních

aut by mohlo nastat „podsvícení“ překážky a brána by narazila do překážky. Konkrétně byl koupen pár fotobuněk Viki 11 od firmy AZ pohony.

5.3 Montáž

Montáž byla zdlouhavá, obsahovala mnoho činností. Nejprve bylo potřeba připevnit motor. To se provedlo pomocí chemické kotvy a dvou kusů závitové tyče M10. Po připevnění motoru bylo zapotřebí přivařit na bránu ozubený hřeben, pomocí kterého se bude brána pohybovat. Dále bylo potřeba osadit dvě čtečky RFID karet a dva páry fotobuněk. Montáž dále obsahovala osazení a propojení rozvaděče, který obsahoval PLC, relé (3x), dva spínané zdroje, jistič s jmenovitým proudem 4A a univerzální PRWA2 převodník.

Po úspěšném osazení můžeme začít zapojovat. Obvod se musí zapojit dle navrženého schématu (obrázek 5.1a) a pomocí schématu, který obsahuje návod k řídící jednotce (příloha 3).

Stručná fotodokumentace z montáže v příloze 4.

5.4 Oživování

Po kompletním zapojení můžeme začít oživovat. Během oživování nastaly problémy s funkčností fotobuněk a načítání čipů. Podrobnější popis chyb a jejich odstranění bude popsán v závěru.

5.5 Nastavení brány

Po úspěšném rozpohybování brány je třeba na řídící jednotce pro motor nastavit několik parametrů. Jedny z nejdůležitějších nastavovaných parametrů jsou nastavení síly/rychlosti motoru při zavírání, zpomalení těsně před otevřením či zavřením a nastavení citlivosti na překážku tj. doba, po které se brána začne při zavírání opět otevírat, lze nastavit 0,1 – 3s.

Jak toto správně nastavit je důkladně popsáno v manuálu k řídící jednotce (příloha).

6 Závěr

Problém nastal při zapojování dvou párů fotobuněk (jeden pár fotobuněk se skládá z vysílače a přijímače, pokud přijímač s vysílačem nekomunikuje a brána se právě zavírá tak řídící jednotka okamžitě končí zavírání a nastane otevírání, aby nenastala kolize s překážkou). Po zapojení dvou párů fotobuněk se stávalo, že pokud překážka nezakrývala oba snímače, tak se brána stále zavírala. Problém byl v tom, že jsme umístili obě vysílací části fotobuněk na jeden sloupek a obě části přijímacích fotobuněk na sloupek druhý. Problém byl ve velkém vyzařovacím úhlu vysílače, který ovlivňoval oba přijímače a to i přesto že od sebe jsou vzdáleny 50cm. Asi nejjednodušší řešení tohoto problému bylo změnit umístění jednoho z vysílačů na druhý sloupek. Poté už se vysílače nemohly vzájemně ovlivňovat. Po přepojení vše fungovalo dle předpokladů.

Další problém nastal při odlaďování programu. Poté, co jsem do programu vložil modul **TimerOn** simulující časové zpoždění výstupu (které nastane, pokud někdo otevře bránu v době, kdy má být brána zavřená), přestalo správně fungovat načítání čipů. Nevím, co tuto chybu způsobilo, ale když jsem použil místo modulu algoritmus popisovaný v části 3.5.1 tak načítání čipů fungovalo správně a brána se automaticky zavírala, pokud ji někdo otevřel v době, kdy má být zavřená, nebo pokud už nastala doba, kdy se má brána automaticky zavřít.

Funkčnost této práce byla podnětem k nápadu pořídit ještě jednu čtečku RFID karet, která by se přes již používané PLC propojila s docházkovým systémem. Tím by se zvýšila kontrola příchodů a odchodů zaměstnanců.

Závěrem bych chtěl podotknout, že celá tato aplikace je plně funkční až na malý problém, který nastal při přechodu ze zimního času na letní. Tento problém se dá vyřešit pomocí aplikační poznámky 22, která je volně k dispozici na stránkách www.amit.cz

Tato práce pro mě byla velikým přínosem. Naučil jsem se pracovat v programu DetStudio a ViewDet, hlavně díky tomu, že jsem mohl absolvovat dvě školení, která mě rychle a kvalitně seznámila s tímto vývojovým prostředím. Dozvěděl jsem se nové informace o sériové komunikační lince a o protokolech, které tato komunikace používá. Získal jsem nové zkušenosti v psaní dokumentace a tvorbě schémat. Rozšířil jsem svoje programátorské znalosti. Poznal jsem, co všechno obnáší vytvářet samostatně takto

rozsáhlou práci. Všechny získané zkušenosti pro mě jistě budou při řešení dalších podobných aplikací velkým přínosem.

7 Odkazy, citace

7.1 Citace

[1]=citace z nápovědy programu DetStudio verze 1.6.17

[2]=citace z nápovědy programu ViewDet verze 1.0.0.13

8 Přílohy

Přílohy k dokumentu jsou na dalších stránkách.

Příloha 1 – Kabel 232RP

Příloha 2 – Univerzální převodník PRWA2

Příloha 3 – Manuál k řídící jednotce a motoru

Příloha 4 – Fotodokumentace

Příloha 5 – RFID čtečka, fotobuňky