České vysoké učení technické v Praze
Fakulta elektrotechnická

# DIPLOMOVÁ PRÁCE

## Vzdálené ovládání modelu v laboratoři

**Praha, 2012** **Autor: Jaroslav Pecka**

## Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne 3.1.2012

podpis

## Poděkování

Rád bych poděkoval vedoucímu diplomové práce Pavlovi Burgetovi za jeho cenné rady a čas a také administrátorovi serveru LabLink Ondřeji Fialovi za spolupráci.

# Abstrakt

Tato práce se zabývá připojením výukového modelu do systému vzdálené laboratoře *LabLink*. Systém zajišťuje přístup k modelům umístěným v laboratoři distribuovaných řídicích systémů katedry řídicí techniky na Karlově náměstí v Praze. Tyto modely se používají především pro výuku předmětu *Řídicí systémy*. Pro vzorové připojení byl zvolen model pneumatických výtahů. Navržené síťové řešení využívá průmyslový komunikační protokol *Profinet IO*. Pro programové řízení PLC řady *SIMATIC S300*, které model řídí, je použita knihovna *Command Interface*. Pomocí této knihovny jsou v programovacím jazyce *C#* vytvořeny spustitelné skripty, které jsou automaticky spouštěny ze serveru *LabLink*. Tímto způsobem je zajištěna programová údržba PLC.

Dále je v práci řešeno rozšíření modelu o bezpečnostní prvky. V laboratoři bylo nainstalováno bezpečnostní PLC *SIMATIC 315F-2 DP/PN* a bezdrátový bezpečnostní mobilní vizualizační panel *MP277F IWLAN*. Tato zařízení byla integrována do stávajícího síťového řešení založeného na protokolu *Profinet IO* s rozšířením o bezpečnostní protokol *ProfiSAFE*. V rámci bezpečnostního systému byly v laboratoři nadefinovány efektivní oblasti a zóny pro vizualizační panel, navržen bezpečnostní řídicí program pro bezpečnostní PLC a vytvořena vizualizace procesu řízení pneumatických výtahů.

# Abstract

This thesis deals with the involvement of the educational model into the system of distant laboratory *LabLink*. This system provides access to models placed in the laboratory of distributed control systems, at the Department of Control Engineering, Charles Square in Prague. These models are primarily used at practice lessons of *Control systems* course. For the exemplary solution was chosen model of pneumatic lifts. The proposed network solution uses an industrial network communication protocol *Profinet IO*. For program control of PLC *SIMATIC S300* series, which controls the model, is used *Command Interface* library. Using this library are created executable scripts in *C#* programming language. These scripts are run automatically from the server *LabLink*. This way is ensured maintenance of configuration and programs contained in PLC.

Further is solved extension of the model of safety features. In the laboratory was installed safety PLC *SIMATIC 315F-2 DP/PN* and wireless mobile safety visualization panel *MP277F IWLAN*. This devices has been integrated into the current network solution based on *Profinet IO* protocol, with extension of safety protocol *ProfiSAFE*. Within the scope of safety system has been defined effective ranges and zones of mobile panel, the safety control program has been designed and the visualization of control proces of the pneumatic lift model has been created.

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

# ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Jaroslav Pecka**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Vzdálené ovládání modelu v laboratoři**

Pokyny pro vypracování:

1. Realizujte vzorovou aplikaci řízení výtahu včetně identifikace modelu.
2. V jazyce C# realizujte aplikaci pro nahrání HW konfigurace a programu do PLC Simatic a nasaďte ji na serveru Lablink pro správu vzdálené laboratoře.
3. Seznamte se s programováním mobilního panelu Siemens.
4. Navrhněte způsob vzdáleného ovládání modelu v laboratoři prostřednictvím sítě WiFi, zařízení Profinet IO a mobilního panelu. Navržený způsob realizujte a demonstrujte na modelu výtahu.
5. Vytvořte dokumentaci pro použití modelu v laboratoři, rovněž dbejte na důsledné průběžné dokumentování kódu a pracovních postupů.

Seznam odborné literatury:

Dodá vedoucí práce

Vedoucí: Ing. Pavel Burget, Ph.D.

Platnost zadání: do konce letního semestru 2011/2012

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

L.S.

v z. M. Demlová
prof. Ing. Boris Šimák, CSc.
děkan

V Praze dne 16. 2. 2011

# Obsah

# Seznam obrázků

# Seznam tabulek

# Kapitola 1

# Úvod

Prvním hlavním cílem této diplomové práce je vzorová integrace modelu pneumatických výtahů do systému vzdálené laboratoře LabLink. Druhým cílem je rozšíření modelu o bezpečnostní prvky a instalace bezpečnostního PLC s mobilním vizualizačním panelem.

Model pneumatických výtahů (obr. 1.1) je umístěn v laboratoři distribuovaných řídicích systémů KN:109s fakulty elektrotechnické na Karlově náměstí v Praze. Model je řízen PLC řady SIMATIC S300. Ostatní modely v laboratoři jsou řízeny stejným PLC a předpokládá se, že navržené řešení připojení modelu do systému LabLink bude po odladění upraveno i pro ostatní modely. To samé platí pro integraci bezpečnostních prvků a bezpečnostního PLC. Instalované bezpečnostní PLC bude jednou obsluhovat všechny bezpečnostní prvky v laboratoři a k němu připojený mobilní panel bude umožňovat vizualizaci a řízení všech modelů v laboratoři.

V této práci je nejdříve podrobně popsán stávající model pneumatických výtahů a jeho rozšíření. Model je identifikován a navržen vzorový řídící program. Důležitou část práce tvoří programové ovládání PLC. V této práci je vyřešeno připojení k řídícímu PLC z programovacího jazyka *C#* pomocí knihovny *Command Interface*. Výstupem této práce jsou spustitelné skripty, které umožňují vymazání programové paměti PLC a nahrání vzorového řídícího programu. První skript je spouštěn při začátku a konci studentské rezervace modelu. Díky tomu má student k dispozici vždy PLC s prázdnou pamětí. Nemůže si tedy z PLC uploadovat program studenta, který s modelem pracoval před ním. Skript pro nahrání vzorového programu je spouštěn z mobilního panelu a slouží k demonstračním účelům.

Obrázek 1.1: Model pneumatických výtahů v laboratoři s109

V druhé části práce zabývající se bezpečnostními prvky je navržen a nainstalován bezpečnostní systém řízený bezpečnostním PLC. Pomocí bezpečnostních vstupů a výstupů jsou k tomuto PLC připojeny bezpečnostní prvky modelu. V PLC běží bezpečnostní program, který tyto prvky obsluhuje. Hlavním úkolem tohoto programu je vyhodnocování stavu bezpečnostních tlačítek a na jeho základě uvádění modelu do bezpečného stavu. Tomu v našem případě odpovídá odpojování modelu od napájení pomocí relé. Druhým

úkolem je rozsvěcování světel, která při uvedení modelu do stop stavu musí zůstat svítit, a proto nemohou být připojena přímo k modelu. Poslední část této práce se zabývá vytvořením vizualizace modelu pro mobilní bezpečnostní vizualizační panel. Tato vizualizace umožňuje kromě plného ovládání modelu také již zmíněné nahrávání vzorového řídícího programu do PLC, i ovládání bezpečnostních prvků modelu.

V závěru je diskutována dosažená funkcionalita, ale také omezení realizovaného řešení.

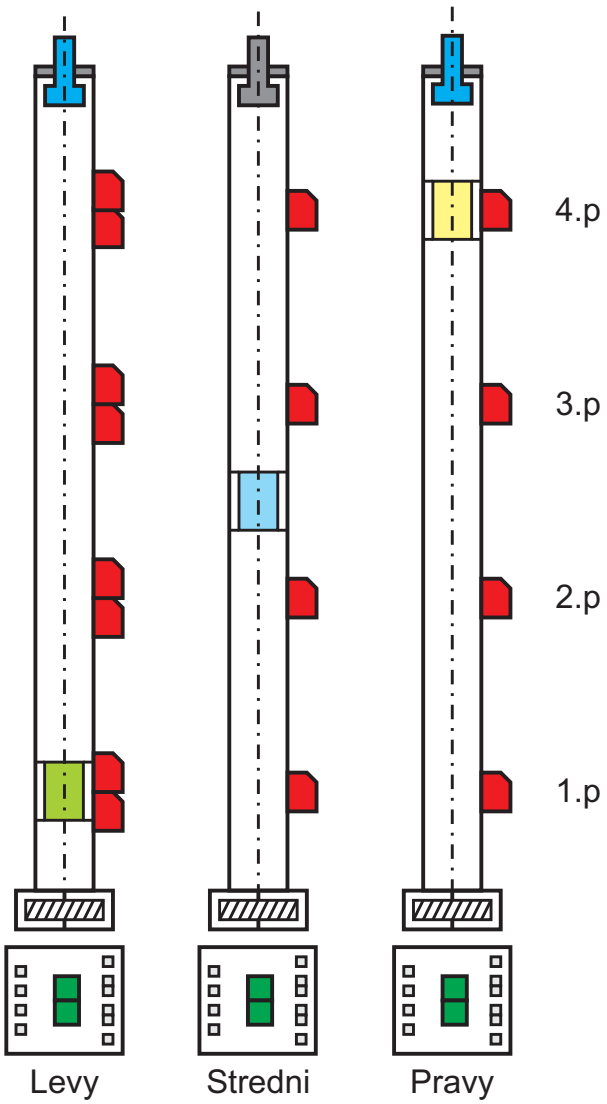
# Kapitola 2

# Popis modelu

V první části této kapitoly 2.1 je popsána konstrukce samotného modelu pneumatických výtahů, který představuje řízenou technologii. Jsou zde podrobně popsány použité senzory, tlačítka, signalizační LED diody a jejich připojení k vstupním a výstupním modulům jednotlivých periferií PLC. V druhé části 2.2 je popsáno realizované zapojení celého řídícího systému včetně bezpečnostních prvků a mobilního vizualčního panelu.

## 2.1 Popis modelu pneumatických výtahů

Model pneumatických výtahů se skládá ze tří vertikálně umístěných trubic z plexiskla, které představují výtahové šachty. Schéma modelu je na obr. 2.1. V každé trubce je umístěn papírový váleček, reprezentující kabinu výtahu. Ke spodnímu ústí každé trubky je umístěn ventilátor, který do ní vhání vzduch. Regulací otáček ventilátoru se řídí proud vzduchu, který nadnáší váleček. Tímto způsobem je možné řídit pohyb válečku v trubici.




Obrázek 2.1: Schéma modelu pneumatických výtahů

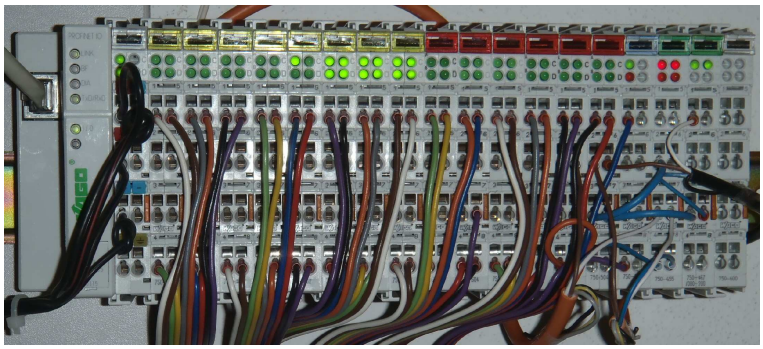
Poloha válečku je snímána ultrazvukovými senzory polohy (na schématu jsou vyznačeny modře a šedivě), které jsou připevněny k horním ústím trubic. Dále je model vybaven optickými čidly (vyznačena červeně), která detekují přítomnost kabiny výtahu (papírového válečku) v patře. Střední a pravý výtah je vybaven po jednom optickém čidle na každé patro. Levý výtah má v každém patře umístěna čidla dvě. Původní model totiž disponoval jen jedním ultrazvukovým čidlem polohy. Toto čidlo bylo umístěné na středním výtahu. Levý a pravý výtah byl na každé patro vybaven dvojicí optických čidel, tak jako teď levý výtah. Tyto výtahy byly řízeny pomocí přepínání dvou hodnot PWM modulace. Při menší hodnotě se výtah pohyboval dolů, při větší nahoru. Zastavení výtahu bylo docíleno jejich přepínáním pomocí dvojice optických čidel v patře. Tento

způsob řízení výtahu moc dobře nefunguje, proto byla přikoupena další dvě ultrazvuková čidla polohy (nová jsou na obr. 2.1 vyznačena modře, původní šedivě). Na levém výtahu byli dvojice čidel ponechány, aby si studenti mohli tento nevhodný způsob řízení zkusit. Původní konfiguraci modelu a způsob jeho řízení lze dohledat v diplomové práci (Henych, T., 2004).

Pod každým výtahem je umístěn jeden vstup/výstupní panel, na kterém jsou umístěna všechna tlačítka a LED diody signalizující aktuální stav výtahu. Tento panel je podrobněji popsán v části 2.1.1.


Obrázek 2.2: CPU 315-2 DP/PN

Celý model je řízen programovatelným logickým automatem (PLC) řady SIMATIC S300 od Simensu. Jedná se o konkrétní model CPU 315-2 PN/DP (obr. 2.2). Tento model je vybaven rozhraním Profibus DP a Profinet IO. K tomuto PLC je pomocí Profinetu IO připojena periferie WAGO 750 (obr. 2.3) se vstupnímy a výstupnímy moduly. K těmto modulům jsou připojeny všechny pracovní vstupy a výstupy řízené technologie (modelu pneumatických výtahů).



Obrázek 2.3: Periferie WAGO750 se vstupnímy a výstupnímy moduly

### 2.1.1 Vstup / výstupní panel

Každému ze tří výtahů náleží jeden vstup/výstupní panel, který je na obr. 2.4. V pravé části je umístěno šest tlačítek. Tato tlačítka reprezentují tlačítka pro přivolání výtahu umístěná v jednotlivých patrech budovy. Pro koncová patra po jednom a pro 2. a 3. patro po dvou. Jedno pro požadavek na cestu vzhůru a druhé pro požadavek na jízdu směrem dolů. Těchto šest tlačítek je zároveň i signalizačnímy LED diodami.


Obrázek 2.4: Vstup / výstupní panel výtahu

V levé části panelu se nachází čtyři tlačítka reprezentující tlačítka umístěná v kabině výtahu. Tato tlačítka opět zároveň slouží jako signalizační LED diody. Dále je na panelu umístěn přepínač zatížení výtahu. Tímto přepínačem simulujeme skutečný snímač, který je umístěn v reálném výtahu a zabraňuje rozjetí prázdného výtahu při požadavku z vnitřku kabiny. Na panelu ja také umístěn jednomístný segmentový display, na kterém lze pomocí dvou bitů rozsvítit čísla od jedné do čtyř. Hned vedle je umístěna dvojice šipek určená pro signalizaci směru pohybu kabiny výtahu.

Jednotlivé adresy všech tlačítek a přepínače zatížení kabiny jsou v tab 2.4. Adresy všech LED diod a bitů segmentového displaye jsou v tab 2.5.

Z důvodu velkého množství potřebných vstupů a výstupů nejsou všechny tři panely připojeny přímo k periferii WAGO750. Panely jsou připojeny přes elektronický přepínač a přistupujeme k nim pomocí časového multiplexu. K periferii je v konkrétní okamžik připojen vždy jen jeden panel. Který panel je právě připojen volíme prostřednictvým trojice bitů. Adresy těchto bitů jsou v tab 2.5. Každému panelu náleží jeden bit. Který

z trojice bitů má hodotu log. jedničky, ten příslušný panel je připojen. Pro čtení a zápis všech proměnných na všech panelech je tedy nutné napsat program cyklického čtení a zapisování hodnot vstupů a výstupů z paměti PLC na jednotlivé panely. Tento program musí také zaručit, že z trojice přepínaných bitů bude nabývat hodnotu log. jedničky vždy jen právě jeden bit.

### 2.1.2 Výstupy modelu

Zde musíme poznamenat, že z hlediska modelu se jedná o výstupy, zatímco z hlediska PLC o vstupy. Ze senzorů vystupují signály, které reprezentují aktualní stav modelu a vstupují do vstupního modulu PLC.

Obrázek 2.5: Ultrazvukový senzor polohy SICK UM30-214113

Ke snímání polohy levého a pravého válečku používáme ultrazvukové senzory polohy SICK UM30-214113. Na obr. 2.1 jsou vyznačeny modře. Senzor UM30-214113 je na obr. 2.5, datasheet od výrobce (SICK, 2010) se nachází na přiloženém CD. Důležité technické parametry podle kterých jsme senzor vybírali, jsou vypsány v tab 2.1. Čidlo disponuje displayem a dvojicí tlačítek jejichž pomocí lze nastavit parametry čidla. Všechny tyto funkce jsou popsány v datasheetu (SICK, 2010). My jsme nastavili analogový výstup a vzestupnou výstupní charakteristiku (při přibližování předmětu k senzoru roste výstup), která odpovídá našemu umístění senzoru u horního ústí trubice výtahu.

Polohu prostředního válečku snímá původní ultrazvukový senzor Mikrosonic, který je na obr. 2.1 vyznačen šedě. Jeho datasheet se nám nepodařilo získat, ale v tab 2.2 jsou uvedeny parametry převzaté z práce (Henych, T., 2004).

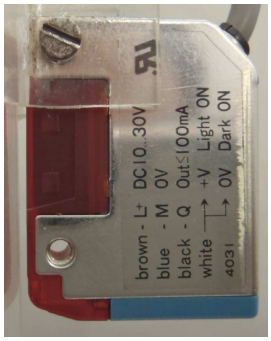
Tabulka 2.1: Parametry UZ senzoru SICK

| | |
|---|---|
| napájecí napětí | $15 - 30\,\mathrm{V}$ |
| analogový výstup | $0 - 10\,\mathrm{V}$ / $4 - 20\,\mathrm{mA}$ |
| ultrazvuková frekvence | $120\,\mathrm{kHz}$ |
| rozsah | $350 - 3400\,\mathrm{mm}$ |
| přesnost | $< 2\,\%$ |
| opakovatelnost | $\pm 0,15\,\%$ |
| časová odezva výstupu | $< 180\,\mathrm{ms}$ |

Tabulka 2.2: Parametry UZ senzoru Mikrosonic

| | |
|---|---|
| napájecí napětí | $10 - 30\,\mathrm{V}$ |
| analogový výstup | $0 - 10\,\mathrm{V}$ / $4 - 20\,\mathrm{mA}$ |
| ultrazvuková frekvence | $320\,\mathrm{kHz}$ |
| rozsah | $300 - 2500\,\mathrm{mm}$ |
| přesnost | $\pm 1\,\mathrm{mm}$ |
| časová odezva výstupu | $< 40\,\mathrm{ms}$ |

Přítomnost válečku v patře je detekována optickým senzorem polohy obr. 2.6. Tento senzor detekuje přítomnost překážky až do vzdálenosti půl metru. Tuto vzdálenost lze zkrátit pomocí žlutého trimru na senzoru. Pomocí zapojení lze dosáhnout kladné, či záporné výstupní logiky. My jsme senzor pomocí trimru nastavili tak, aby detekoval přítomnost válečku a nebyl rušen okolím. Zároveň jsme zvolili zapojení s kladnou výstupní logikou. Rozmístění těchto čidel je patrné z obr. 2.1, kde jsou tyto senzory vyznačeny červeně.




Obrázek 2.6: Optický senzor polohy

Namapování optických senzorů v jednotlivých patrech do vstupního obrazu PLC je v tab 2.3.

Tabulka 2.3: Tabulka namapování optických čidel

| **optické čidlo** | **levý výtah** | **střední výtah** | **pravý výtah** |
|---|---|---|---|
| 1. patro (horní) | I0.0 | I1.3 | I3.0 |
| 1. patro - spodní | I0.1 | - | - |
| 2. patro (horní) | I0.3 | I2.0 | I3.2 |
| 2. patro - spodní | I0.2 | - | - |
| 3. patro (horní) | I1.1 | I2.1 | I3.3 |
| 3. patro - spodní | I1.0 | - | - |
| 4. patro (horní) | I1.2 | I2.2 | I4.1 |
| 4. patro - spodní | I4.0 | - | - |

Namapování zbývajících vstupů je v tab 2.4. Jedná se o ultrazvukové senzory polohy, všechna tlačítka a detekci zatížení kabiny.

Tabulka 2.4: Tabulka namapování zbývajících vstupů PLC

| **vstup** | **typ** | **adresa** |
|---|---|---|
| tlačítko v kabině 1. patro | bool | I5.0 |
| tlačítko v kabině 2. patro | bool | I5.1 |
| tlačítko v kabině 3. patro | bool | I5.2 |
| tlačítko v kabině 4. patro | bool | I5.3 |
| tlačítko v 1. patře | bool | I6.0 |
| tlačítko v 2. patře - dolů | bool | I6.1 |
| tlačítko v 2. patře - nahoru | bool | I6.2 |
| tlačítko v 3. patře - dolů | bool | I6.3 |
| tlačítko v 3. patře - nahoru | bool | I7.0 |
| tlačítko v 4. patre | bool | I7.1 |
| zatížení kabiny | bool | I7.2 |
| uz. senzor polohy levý výtah | int | IW28 |
| uz. senzor polohy pravý výtah | int | IW30 |
| uz. senzor polohy střední výtah | int | IW20 |

### 2.1.3 Vstupy modelu

Z hlediska modelu se jedná o vstupy, zatímco z pohledu PLC o výstupy. K výstupnímu modulu PLC jsou připojeny vstupy modelu, pomocí kterých model řídíme a rozsvěcíme signalizační prvky.

Tabulka 2.5: Tabulka namapování výstupů PLC

| **výstup** | **typ** | **adresa** |
|---|---|---|
| výběr levého výtahu | bool | Q0.0 |
| výběr středního výtahu | bool | Q0.1 |
| výběr pravého výtahu | bool | Q0.2 |
| světla | bool | Q0.3 |
| LED v kabině - 1. patro | bool | Q1.3 |
| LED v kabině - 2. patro | bool | Q1.2 |
| LED v kabině - 3. patro | bool | Q1.1 |
| LED v kabině - 4. patro | bool | Q1.0 |
| LED v 1. patře | bool | Q2.0 |
| LED v 2. patře - dolů | bool | Q2.1 |
| LED v 2. patře - nahoru | bool | Q2.2 |
| LED v 3. patře - dolů | bool | Q2.3 |
| LED v 3. patře - nahoru | bool | Q3.0 |
| LED v 4. patře | bool | Q3.1 |
| LED šipka nahoru | bool | Q3.2 |
| LED šipka dolu | bool | Q3.3 |
| Display - bit 0 | bool | Q4.0 |
| Display - bit 1 | bool | Q4.1 |
| PWM levý výtah | bool | Q5.0 |
| PWM pravý výtah | bool | Q5.1 |
| PWM střední výtah | real | QD50 |

V tab 2.5 je přehled všech vstupů modelu a jejich namapování do výstupního obrazu paměti PLC. Nejdůležitějšími výstupy PLC jsou akční zásahy. Střední výtah je řízen analogově. Střední ventilátor je vybaven generátorem PWM signálu, jehož vstupem je přímo napětí reprezentující výkon ventilátoru. Tento generátor PWM je datailně popsán v práci (Henych, T., 2004). Generátor PWM signálu je připojen na adresu QD50 a je formátu

REAL. Ostatní dva ventilátory nedisponují generátorem PWM signálu, pouze elektronickým spínačem. Každému ventilátoru tedy náleží jeden bit, který připojuje ventilátor k napájení. PWM signál tedy musí být generován pomocí PLC.

Dalšími důležitými výstupy jsou tři bity Q0.0, Q0.1 a Q0.2, které slouží k přepínání jednotlivých ovládacích panelů výtahů. Při práci s modelem v noci lze výstupem Q0.3 zapnout světla. Zbylé bity slouží k rozsvěcení signalizačních prvků. Tj. LED diod a segmentového displaye.

## 2.2 Zapojení řídícího a bezpečnostního systému

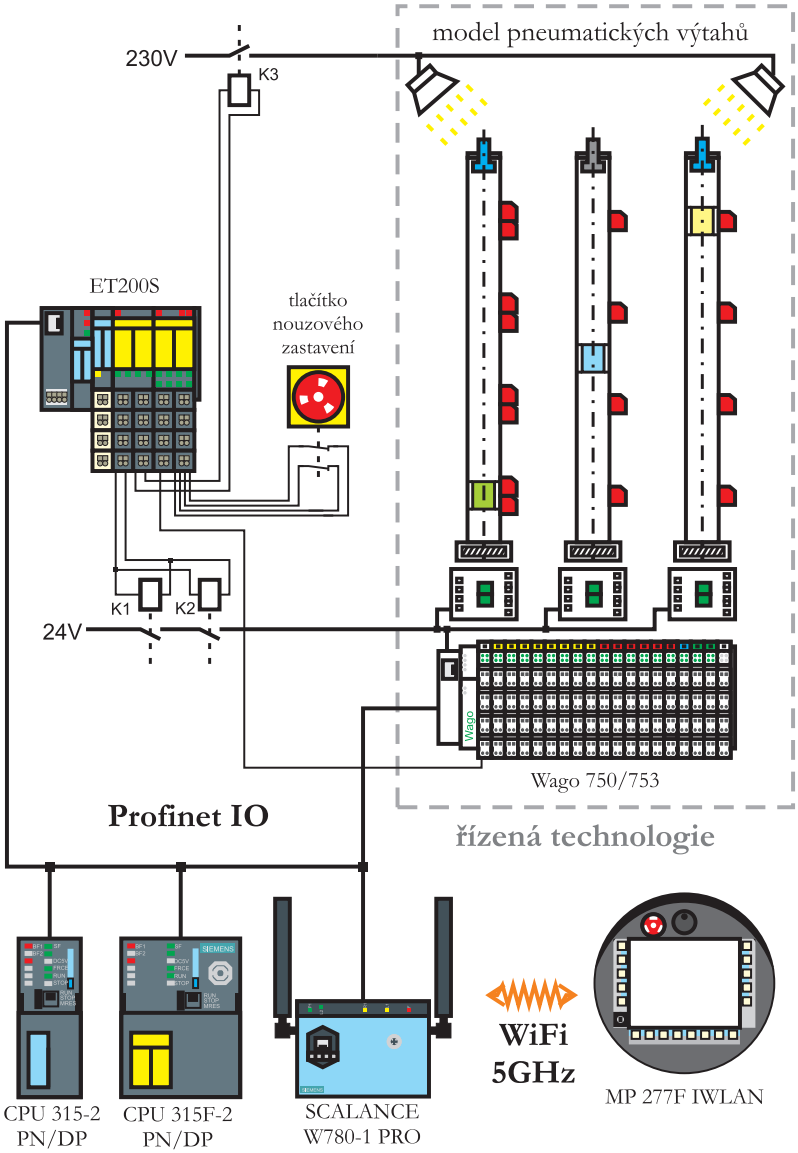
V předchozí části této kapitoly byl popsán model pneumatických výtahů, včetně použitých senzorů, aktuátorů a jejich připojení k PLC pomocí vstupních a výstupních modulů periferie WAGO750. V této částo kapitoly je popsán celý řídící systém, který je znázorněn na obr. 2.7. Všechny periferní moduly a PLC jsou propojeny pomocí průmyslové sběrnice Profinet. K řídícímu PLC *315-2 PN/DP* je připojena již zmíněná periferie WAGO750, ke které jsou připojeny pracovní vstupy a výstupy technologie. K bezpečnostnímu PLC *315F-2 PN/DP* je připojena periferie ET200S, ke ktreré jsou pomocí bezpečnostních vstupů a výstupů, připojeny bezpečnostní prvky a průmyslový přístupový bod průmyslové WiFi sítě SCALANCE, pomocí kterého s bezpečnostním PLC komunikuje bezpečnostní mobilní panel MP277F.

### 2.2.1 Bezpečnostní prvky

Na obr. 2.7 je schéma řídicího systému včetně připojení bezpečnostních prvků k periferii ET200S. První ze dvou tlačítek nouzového zastavení je umístěno přímo na modelu a k periferii ET200S je připojeno čtyřvodičově. Čtyřvodičové zapojení tlačítek nouzového zastavení a různých bezpečnostních koncových spínačů je v dnešní době naprosto běžné. Toto zapojení zvyšuje spolehlivost což je u bezpečnostních tlačítek a spínačů velmi důležitý parametr. Druhé tlačítko nouzového zastavení je umístěno na mobilním panelu. Célá řízená technologie (tím rozumíme model pneumatických výtahů, včetně periferie WAGO750) je připojována k napájení pomocí dvojice relé. Cívky relé jsou připojeny paralerně k jednomu bezpečnostnímu výstupu ET200S a jejich spínané kontakty jsou řazeny do série. Jedná se o standartní bezpečnostní zapojení relé, které zaručuje odpojení technologie od

napájení i v případě že jedno relé selže (například se mu "spečou" kontakty).

Odpojením celého modelu od napájení reprezentujeme uvedení modelu do bezpečného stavu. V praxi je nutné provést bezpečnostní analýzu řízené technologie a na jejím základě rozhodnout co to vůbec je bezpečný stav. Jeho dosažení potom bývá složitější než pouhé odpojení od napájení. Většinou se jedná o celou sekvenci opatření, jejíž vykonání nějakou dobu trvá. V našem modelovém případě ovšem žádné skutečné nebezpečí nehrozí, proto jsme žádnou rizikovou analýzu neprováděli a model odpojený od napájení považujeme za bezpečný.

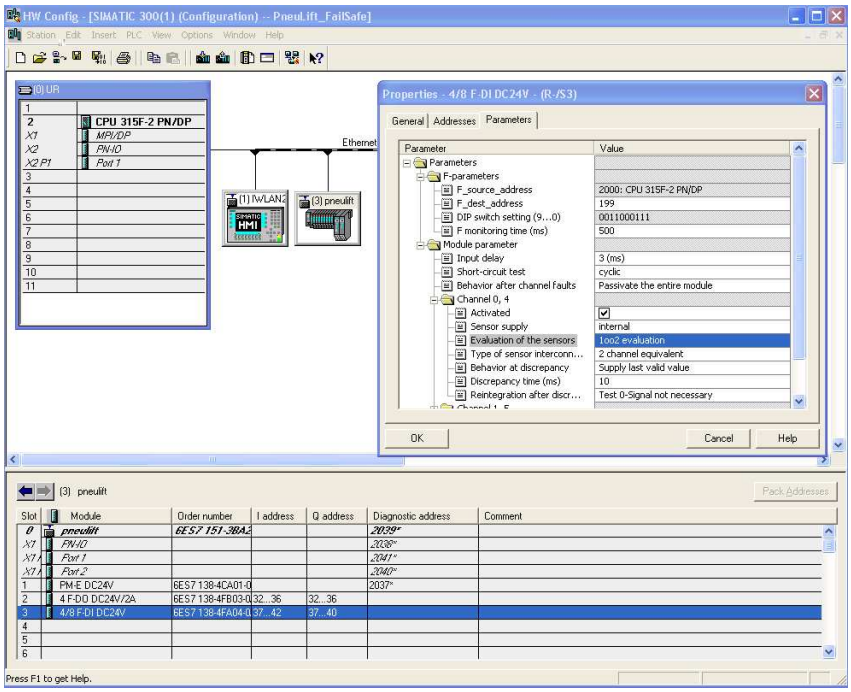



Obrázek 2.7: Schéma řídicího systému

Jelikož by při uvedení modelu do stop stavu měla zůstat svítit světla, je relé ovládající světla připojeno k bezpečnostnímu výstupu. Světla jsou tedy řízena bezpečnostním PLC. Studenti rozsvěcí a zhasínají světla z periferie WAGO750 pomocí výstupu, který je přiveden na bezpečnostní vstup modulu ET200S.

K uvádění technologie ze stop stavu zpět do provozního stavu je použita dvojice spouštěcích tlačítek na mobilním panelu.

V tab 2.6 je popsáno připojení bezpečnostních prvků k jednotlivým vstupním a výstupním modulům periferie ET200S. Konkrétně na adrese Q32.0 je paralelně připojena dvojice relé připínající celý model pneumatických výtahů k napájení. Na druhý bezpečnostní výstup je připojeno relé spínající světla modelu. Na vstupní modul je na adresy I37.0 a I37.1 čtyřvodičově připojeno tlačítko nouzového zastavení. O vyhodnocování obou vstupních kanálů se nemusíme vůbec starat. V hardwarové konfiguraci modulu jde přímo svázat dva vstupní kanály k sobě a vyhodnocovat je jako jeden. Toto nastavení je znázorněno na obr. 2.8. Tyto kanály jsou potom vyhodnocovány jako jeden a dojde-li k porušení jednoho z vodičů, je tato událost ihned detekována vyhodnocujícím bezpečnostním vstupním modulem periferie ET200S. Na základě této události je vyslán alarm a bezpečnostní tlačítko se považuje za aktivní (to standartně znamená rozepnuté).




Obrázek 2.8: HW konfigurace čtyřvodičového zapojení

Tabulka 2.6: Adresace bezpečnostních vstupů a výstupů periferie ET200S

| **typ modulu** | **výstup** | **typ** | **adresa** |
|---|---|---|---|
| výstup | relé pro připojení modelu k napájení | bool | Q32.0 |
| výstup | relé pro spínání světel | bool | Q32.1 |
| vstup | tlačítko nouzového zastavení - 1. pár vodičů | bool | I37.0 |
| vstup | tlačítko nouzového zastavení - 2. pár vodičů | bool | I37.1 |
| vstup | světla - z periferie WAGO750 | bool | I37.2 |

### 2.2.2 Profinet

Profinet je otevřený komunikační standard mezinárodní organizace Profibus International (PI), založený na standardu Ethernet. Profinet je založen na standardech informační techniky, jako je např. TCP/IP, ale pro účely provozní automatizace poskytuje také možnosti komunikace v reálném čase. Komplexní standard pro průmyslový Ethernet v průmyslové automatizaci prostřednictvím technologie RT (real-time). Celý systém pak uzavírá izochronní komunikace IRT (Isochronous Real-Time) určená pro velmi výkonné úlohy řízení pohybu, který vyžaduje přísně deterministické chování. Díky takto odstupňované komunikační architektuře je možné tyto protokoly bez jakýchkoliv omezení kombinovat. Profinet tedy nabízí otevřený standard komunikace (umožňující např. diagnostiku či připojení na síť Internet) a současně komunikaci v reálném čase. Standard definuje následující tři druhy komunikujících zařízení.

- IO-Contoller
- IO-Supervizor
- IO-Device

Jednotlivá zařízení jsou adresována pomocí IP a MAC adresy. V zařízení IO-Controller běží řídící program. Jedná se tedy většinou o PLC. IO-Supervizor je programátorská stanice (PG), PC na operátorském stanovišti, nebo stanice HMI (Human Machine Interface). Do této skupiny zařízení tedy spadá náš mobilní panel. Posledním druhem komunikujících zařízení je IO-Device. Do této skupiny spadají distribuované periferie se vstupnímy a výstupnímy moduly. Jeden IO-Device modul může komunikovat s více IO-Controller moduly. Profinet nabízí následující druhy komunikace.

- Profinet CBA
- Profinet IO RT
- Profinet IO IRT

Standard Component Based Automation (CBA) zprostředkovává komunikaci mezi zařízenímy IO-Controler. Komunikační cyklus trvá přibližně $50 - 100\,\mathrm{ms}$. Standard Profinet IO RT popisuje cyklyckou výměnu dat stanicemi mezi IO-Device a IO-Controller. Definuje speciální realtimeový kanál a VLAN prioritu rámců, čímž dosahuje délky komunikačního cyklu jen v jednotkách milisekund. Ve specifikaci *Isochronous Real Time* přidává ještě přesnou časovou synchronizaci s přesností do jedné mikrosekundy. Díky tomu je délka komunikačního cylku zkrácena na stovky mikrosekund. V této specifikaci je uvažována i známá topologie komunikační cesty a ta je předem časově rozvržena. Tato specifikace je určena především pro přesné polohování pohonů a lze s ní dosáhnout komunikačního cyklu až $250\,\mu s$.

Hlavní výhodou Profinetu je fakt, že pro připojení periferií lze použít stávající kabeláž Fast Ethernetu. Pro specifikace CBA a IO RT dokonce lze použít standartní síťové prvky (např switche). Pro specifikaci IRT už je nutno použít speciálních switchů, které podporují VLAN priority rámců a časové plánování komunikace. Podrobný popis standartu Profinet je možno dohledat v literatuře (PROFINET, 2009).

Standart Profinet samozřejmě není závislý na fyzickém médiu, proto ho lze provozovat na bezdrátové technologii WiFi. Té používáme pro připojení mobilního panelu.

### 2.2.3 PROFIsafe

PROFIsafe je bezpečnostní komunikační protokol, který zajišťuje komunikaci mezi bezpečnostními moduly. Může běžet na sběrnici společně s Profibusem, nebo Profinetem. Komunikaci těchto sběrnic nijak neovlivňuje. Komunikace probíhá ve zvláštním odděleném kanále. Zařízení nejsou adresována pomocí IP adres, ale pomocí PROFIsafe adres. Tato adresa se u řídících zařízení (PLC, HMI) nastavuje programově a na periferiích je nutné nastavit pomocí DIP přepínačů adresu vygenerovanou HW konfigurací. Více o protokolu PROFIsafe se čtenář může dozvědět v literatuře (PROFIsafe, 2010).

### 2.2.4 Mobilní panel MP277F

Mobilní panel 277F (obr. 2.9) patří do rodiny zařízení SIMATIC HMI (Human Machine Interface). Panel je vybaven osmipalcovým dotykovým TFT displayem, 18 programovatelnými tlačítky, tlačítkem nouzového zastavení, dvojicí spouštěcích tlačítek a otočným inkrementálním ovládacím prvkem. Panel je napájen z baterie. S PLC a ostatními zařízeními komunikuje prostřednictvým WiFi sítě na frekvenci 5 GHz. Ke komunikaci je použit komunikační protokol Profinet a jeho bezpečnostní nadstavba PROFIsafe. Podrobný popis všech prvků a funkcí mobilního panelu lze dohledat v manuálu (MP277F MANUAL, 2008). Tento manuál je umístěn na přiloženém CD.


Obrázek 2.9: Simatic HMI MP277F

K programování panelu je zapotřebí mít řádně nainstalované následující programy.

- STEP 7 V5.4 SP2
- SIMATIC S7 Distributed Safety V5.4 SP3
- WinCC flexible 2007

#### 2.2.4.1 Zóny a Efektivní oblasti

Mobilní panel dále podporuje Zóny (Zones) a Efektivní oblasti (Efective ranges). K mobilnímu panelu lze zakoupit transpondéry (Transponder). Každému transpondéru je pomocí trimrů přiřazeno jedinečné ID, které vysílá do svého okolí. Tyto transpondéry jsou pak umístěny u jednotlivých částí technologie. Mobilní panel přijímá vysílaná ID a podle nich detekuje v jaké části technologie se nachází.

V projektu ve WinCC flexible programátor jednotlivým transpondérům přiřadí nadefinované zóny a efektivní oblasti. Rozdíl mezi oblastmi a zónami je následující. Zóny se v mobilním panelu přepínají samy, tak jak mobilní panel detekuje ID transpondéru. Zatímco pokud operátor s panelem vkročí do efektivní zóny, musí se do ní přihlásit. Dokud tak neučiní, nejsou aktivní bezpečnostní spouštěcí tlačítka. Ta fungují jedině pokud je operátor přihlášen v efektivní oblasti. Realizované rozvržení zón a efektivních oblastí je podrobně rozebráno v kapitole o vizualizaci 6.1.

#### 2.2.4.2 Bezpečnostní stavy

Mobilní panel nepodporuje pouze stav nouzového zastavení. Mobilní panel rozlišuje až následující čtyři bezpečnostní stavy.

- Emergency stop
- Shutdown
- Local rampdown
- Global rampdown

Stav Emergency stop se spouští stiskem tlačítka nouzového zastavení. Slouží k zastavení celé technologie a není závislý na efektivní oblasti ve které se operátor s panelem nachází.

Technologie je uvedena do stavu Shutdown ve chvíli, kdy je přerušena PROFIsafe komunikace (nastane chyba komunikace) a operátor je přihlášen v efektivní oblasti. Tento scénář tedy nespouští operátor, ale bezpečnostní PLC. Stav slouží k nouzovému zastavení části technologie ve které pracuje operátor (efektivní oblasti ve které je přihlášen).

Stav Local rampdown nastává 30 vteřin po tom co operátor opustí efektivní oblast bez řádného odhlášení. Operátor je mobilním panelem několikrát upozorněn, že opouští efektivní oblast bez odhlášení. Stav tedy opět spouští bezpečnostní PLC a je závislý na

přihlášené efektivní oblasti. Tento scénář slouží k bezpečnému zastavení části technologie náležící do přihlášené efektivní oblasti.

Do stavu Global rampdown je technologie uvedena v případě že nastala chyba v PROFIsafe komunikaci a operátor není přihlášen v žádné efektivní oblasti. Stav je tedy spouštěn bezpečnstním PLC a slouží k bezpečnému zastavení celé technologie.

V tab 2.10 jsou zobrazeny všechny situace a bezpečnostní stavy které mohou nastat při jednotlivých spouštěcích událostech.

| Stav Mobilního Panelu (MP) | | Událost | | |
|---|---|---|---|---|
| | | Stisknuté STOP tlačí tko | Chyba komunikace | Timeout 30s |
| MP je přihlášen uvnitř efektivní oblasti | MP je uvnitř efektivní oblasti | Emergency stop | Shutdown | |
| | MP je vně efektivní oblasti | Emergency stop | Shutdown | Local rampdown |
| MP není přihlášen v elektivní oblasti | | Emergency stop | Global rampdown | |

Obrázek 2.10: Přehled možných bezpečnostních stavů

# Kapitola 3

# Vzorová aplikace řízení výtahu

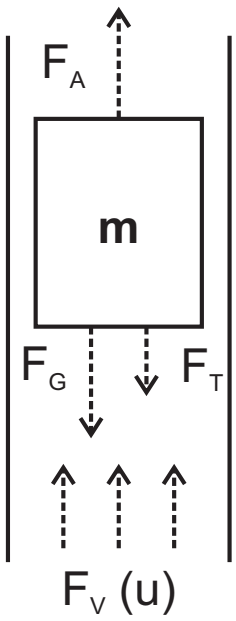
Obsahem této kapitoly je vytvoření vzorové řídící aplikace modelu pneumatických výtahů. To zahrnuje samotnou identifikaci fyzikálního systému, návrh regulátoru polohy, jeho implementace, i implementace celé logiky výtahů. V závěru této kapitoly je popsán navržený bezpečnostní program, který běží v bezpečnostním PLC a zajišťuje obsluhu bezpečnostních prvků.

## 3.1 Matematický model systému

V této části kapitoly je odvozem fyzikální popis válečku pohybujícího se ve vertikálně umístěné trubici. Dále je zde definován popis systému pomocí ARX modelu, který je následně identifikován. Výstupem této části kapitoly je konkrétní přenos systému, který je dále použit k návrhu regulátru.

### 3.1.1 Fyzikální model

Na obr. 3.1 je schéma fyzikálního modelu papírového válečku pohybujícího se ve vertikálně umístěné trubici.




Obrázek 3.1: Schéma fyzikálního modelu

Na váleček o hmotnosti $m$ působí gravitační síla $F_G$. Váleček je zároveň nadnášen silou proudu vzduchu z ventilátoru $F_v$, která je funkcí vstupního napětí ventilátoru $u$. Dále uvažujeme působení tření $F_T$, které je úměrné rychlosti válečku $v$ přes konstantu $k_T$ a působí proti směru pohybu válečku. Složením těchto sil získáme jejich výslednici $F_a$, reprezentující výslednou sílu která na váleček působí a je zodpovědná za jeho pohyb.

$$F_d + F_v(u) - F_g - F_T = 0 \tag{3.1}$$

Síla proudu vzduchu ventilátoru $F_v(u)$ je obecně funkcí vstupního napětí ventilátoru $u$. Proud vzduchu vháněný ventilátorem do trubice je přímo úměrný otáčkám ventilátoru. Správně bychom měli uvažovat mezi vstupním napětím a výstupními otáčkami ventilátoru dynamický systém minimálně prvního řádu jako ekvivalent stejnosměrného elektrického motoru ventilátoru. To ovšem v našem případě není potřeba, protože my provozujeme ventilátor v blízkém okolí rovnovážného bodu, kdy papírový váleček stojí v patře, nebo se pomalu pohybuje nahoru či dolů. Tomu odpovídá přibližně rozsah 50-70% PWM. V tomto rozsahu můžeme spolehlivě považovat sílu proudu vzduchu ventilátoru působící na papírový váleček za přímo úměrnou vstupnímu napětí ventilátoru. Dosazením za $F_v(u)$,

$F_g$, $F_a$ a $F_T$ získáme rovnici (3.2).

$$\begin{gathered} F_G = m \cdot g \quad , \quad F_a = m \cdot a \quad , \quad F_T = k_T \cdot v \quad , \quad F_v(u) = k_v \cdot u \\ m \cdot a + k_v \cdot u - m \cdot g - k_T \cdot v = 0 \end{gathered} \tag{3.2}$$

Do (3.2) dosadíme za rychlost a zrychlení derivace polohy, celou rovnici vydělíme hmotností $m$ a získáme tak výslednou pohybovou rovnici (3.3).

$$\ddot{x} - \frac{k_T}{m}\dot{x} - g + \frac{c}{m}u = 0 \tag{3.3}$$

Převedním pohybové rovnice do Laplaceových obrazů získáme přenos (3.4) s obecnými konstantamy $a_0$, $a_1$ a $b_0$.

$$G(s) = \frac{b_0}{s^2 + a_1 s + a_0} \tag{3.4}$$

### 3.1.2 ARX model

Autoregresivní model, nebo také model s chybou rovnice je definovaný následující lineární diferenční rovnicí s chybovým členem $e$.

$$y\,(t) + a_1 y\,(t-1) + \ldots + a_{n_a}\,(t - n_a) = b_0 u\,(t) + b_1\,(t-1) + \ldots + b_{n_b}\,(t - n_b) + e\,(t) \tag{3.5}$$

Rovnice (3.6) popisuje ARX model pomocí polynomů operátoru zpoždění $d$.

$$\begin{aligned} a\,(d) &= 1 + a_1 d + \ldots + a_{n_a} d^{n_a} \\ b\,(d) &= b_0 + b_1 d + \ldots + b_{n_b} d^{n_b} \\ a\,(d)\,y\,(t) &= b\,(d)\,u\,(t) + e\,(t) \end{aligned} \tag{3.6}$$

Zpoždění $n_a$ a $n_b$ nazýváme strukturní parametry modelu. Predikovaná střední hodnota výstupu je lineární funkcí měřených dat (3.7), proto lze pro odhad koeficientů modelu $\Theta$ použít lineární regresi. Definujeme tedy vektor parametrů (3.8) a regresor dat (3.9).

$$Y = Z \cdot \Theta + e \tag{3.7}$$

$$\Theta = [a_1, a_2, \ldots, a_{n_a}, b_0, b_1, \ldots, b_{n_b}]^T \tag{3.8}$$

$$z\,(t) = [y\,(t-1)\,, y\,(t-2)\,, \ldots, y\,(t - n_a)\,, u\,(t)\,, u\,(t-1)\,, \ldots, u\,(t - n_b)]^T \tag{3.9}$$

### 3.1.3 Odhad minimalizující střední kvadratickou chybu

Přehled metod odhadování včetně následujícího odvození je možné najít například v literatuře (Havlena, V.; Štecha, J., 2000).

Naměřená data jsou odezvou na vstupní signál vpuštěný do modelu. Uvažujeme aditivní chybu měření $e$ s normálním rozdělením $e \approx N(0, \sigma)$. $\Theta$ je vektor koeficientů ARX modelu (3.8) a matice $Z$ je regresor dat (3.16).

$$y = Z\Theta + e \tag{3.10}$$

$$e = \varepsilon = y - Z\Theta \tag{3.11}$$

Minimalizujeme střední kvadratickou chybu. Tomu odpovídá minimalizace kriteria $J$.

$$J = \varepsilon^T \varepsilon = (y - Z\Theta)^T (y - Z\Theta) \tag{3.12}$$

$$\frac{\partial J}{\partial \Theta} = -y^T Z - Z^T y + Z^T Z\Theta + Z^T \Theta^T Z = -2Z^T y + 2Z^T Z\Theta = 0 \tag{3.13}$$

$$Z^T Z\Theta = Z^T y \tag{3.14}$$

$$\Theta = \left(Z^T Z\right)^{-1} Z^T y \tag{3.15}$$

Lineární regrese odhadující vektor parametrů $\Theta$ s minální střední kvadratickou chybou má tedy tvar (3.15).

### 3.1.4 Identifikace konstant ARX modelu

Odvozenému přenosu (3.4) odpovídá následující ARX model.

$$y(k) = -a_1 y(k-1) - a_2 y(k-2) + b_2 u(k-2) \tag{3.16}$$

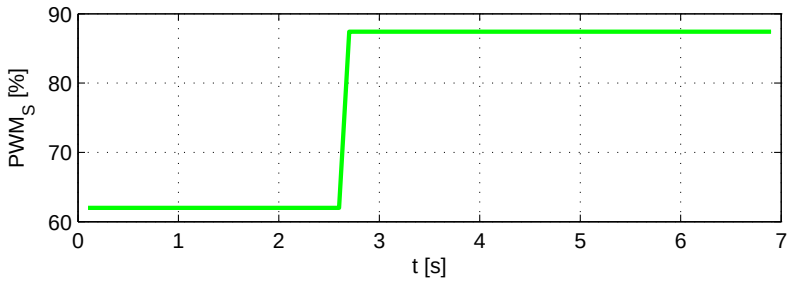
Náš vektor parametrů, které identifikujeme, má tvar.

$$\Theta = [a_1, a_2, b_2]^T \tag{3.17}$$

A regresor vypadá následovně.

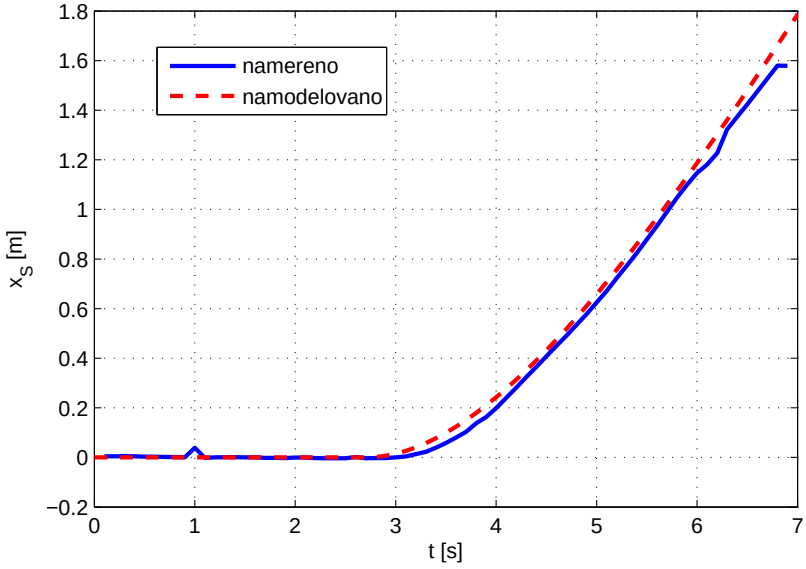
$$\begin{bmatrix} y(k) \\ y(k+1) \\ \vdots \\ y(k+2) \end{bmatrix} = \begin{bmatrix} -y(k-1) & -y(k-2) & u(k-2) \\ -y(k) & -y(k-1) & u(k-1) \\ \vdots & \vdots & \vdots \\ -y(k+n-1) & -y(k+n-2) & u(k+n-2) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ b_2 \end{bmatrix} + e \quad (3.18)$$

Po sestavení regresoru je možné provést identifikační experiment. Uvádíme zde jen jeden experiment, ale musíme poznamenat, že ve skutečnosti jsme provedli experimenty tři. Pro každý systém válečku v trubici jeden vlastní experiment. Jelikož jsou systémy prakticky identické, je jasné že postup byl ve všech případech stejný a identifikované konstanty a naměřené průběhy si jsou velmi blízké. Všechny následující obrázky popisují experiment provedený na středním výtahu (systému papírového válečku v trubici).




Obrázek 3.2: Průběh vstupího signálu identifikačního experimentu

Na obr. 3.2 je vstupní signál identifikačního experimentu. Na dalším obr. 3.3 je modrou barvou vykreslena odpovídající odezva identifikovaného systému. Z těchto naměřených dat sestavíme regresor (3.18) a provedeme lineární regresi dle předpisu (3.15).

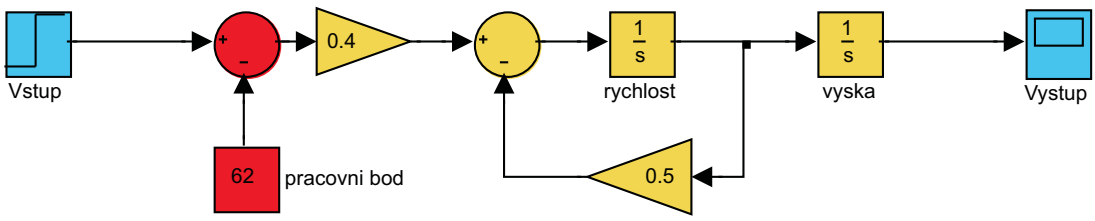



Obrázek 3.3: Průběh výšky válečku při identifikačním experimentu

Z takto získaných konstant jsme sestavili ARX model systému. Jeho převedením na spojitý přenos získáváme následující přenos modelu (3.19). Identifikované konstanty samozřejmě nevyšly takto "přesně"jako desetiná čísla. Konstanty jsou samozřejmě zaokrouhlené. Ale z dalších průběhů je patrné, že tento přibližný model pro naladění PID regulátoru postačuje. Není nutné hledat přesnější model.

$$P(s) = \frac{0.4}{s + 0.5s} \tag{3.19}$$

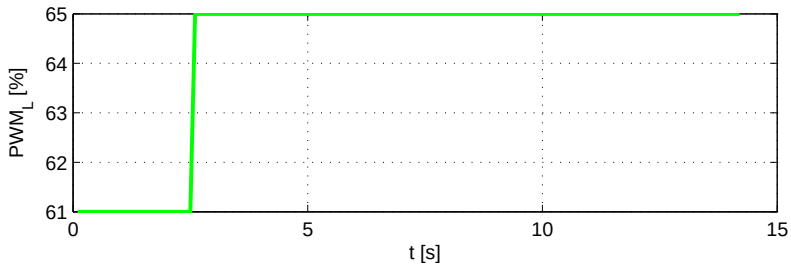
Tomuto přenosu odpovídá model systému obr. 3.4. Jelikož se jedná o zjednodušený linearizovaný model platný v okolí rovnovážného pracovního bodu, musíme jako první tento pracovní bod od vstupu odečíst. Tato operace je na schématu vyznačena červeně. Zbylá část modelu (zlutě) odpovídá přenosu (3.19). Modře je označen vstup a výstup modelu.



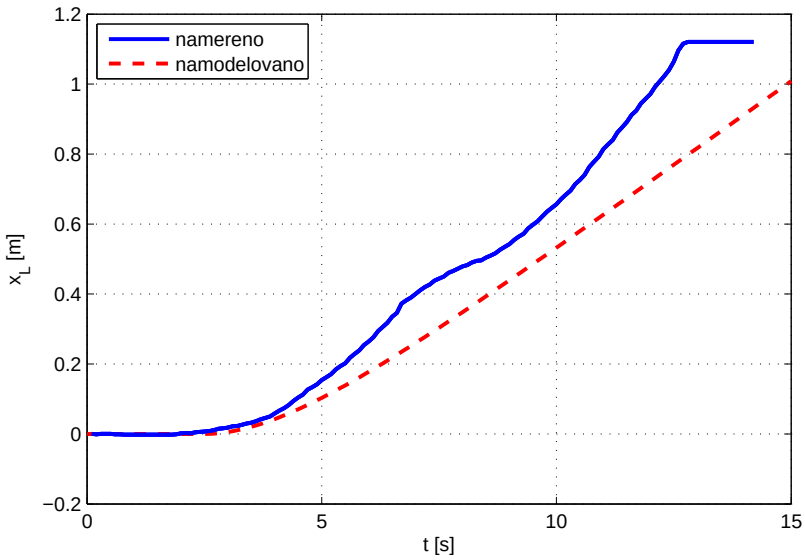

Obrázek 3.4: Schéma modelu v Simulinku

### 3.1.5 Ověření identifikovaných konstant

V předchozí části kapitoly byl identifikován systém středního výtahu. V této části jsou vykresleny odezvy levého a pravého výtahu na skok vstpního napětí ventilátoru v okolí rovnovážného pracovního bodu. Tyto odezvy jsou porovnány s modelovanými odezvami pomocí modelu identifikovaného na systému středního výtahu.

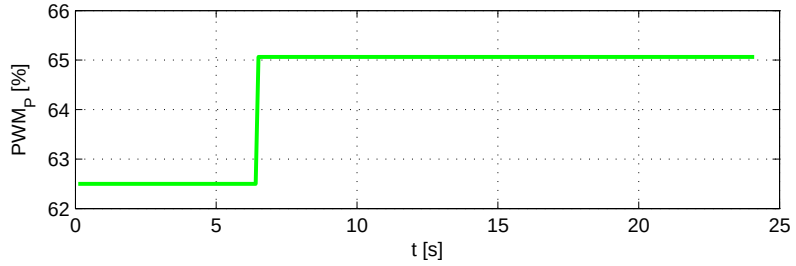



Obrázek 3.5: Skok vstupního napětí levého výtahu

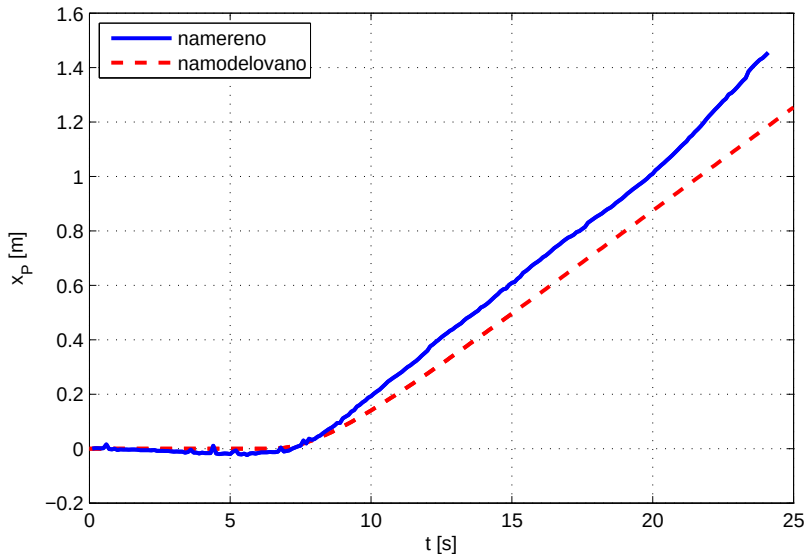



Obrázek 3.6: Odezva levého výtahu na skok vstupního napětí

Na obr. 3.5 je skok vstupního napětí vpuštěný do systému levého výtahu. Na obr. 3.6 je odezva systému porovnaná s odezvou namodelovanou pomocí modelu středního výtahu (obr. 3.4) s přenosem (3.19). Z průběhů je zřejmé, že dynamika modelu přibližně odpovídá a pro návrh jednoduchého PID regulátoru tento model lze s jistou chybou bez úprav použít.




Obrázek 3.7: Skok vstupního napětí pravého výtahu




Obrázek 3.8: Odezva pravého výtahu na skok vstupního napětí

Na obr. 3.7 je skok vstupního napětí vpuštěný do systému pravého výtahu. Na obr. 3.8 je odezva systému porovnaná s odezvou namodelovanou pomocí modelu středního výtahu (obr. 3.4) s přenosem (3.19). Z průběhů je zřejmé, že dynamika modelu přibližně odpovídá a pro návrh jednoduchého PID regulátoru tento model lze s jistou chybou bez úprav použít.

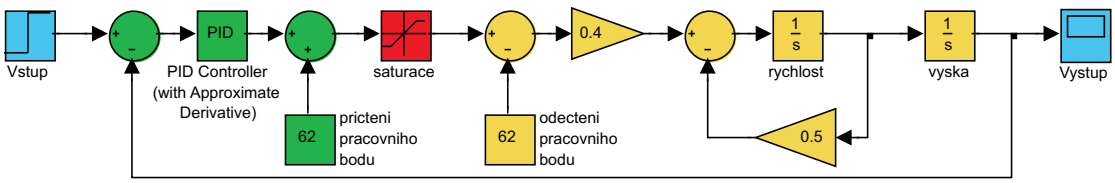
Z průběhů vstupních napětí obr. 3.2, obr. 3.5 a obr. 3.7 je vidět že rovnovážné pracovní body jednotlivých výtahů jsou různé, to nesmíme opomenout.

Závěrem této podkapitoly je, že dynamika všech tří systémů výtahů se chová podobně a proto lze s přijatelnou chybou dynamiku všech tří modelů reprezentovat stejným přenosem (3.19).

## 3.2 Regulátor polohy výtahu

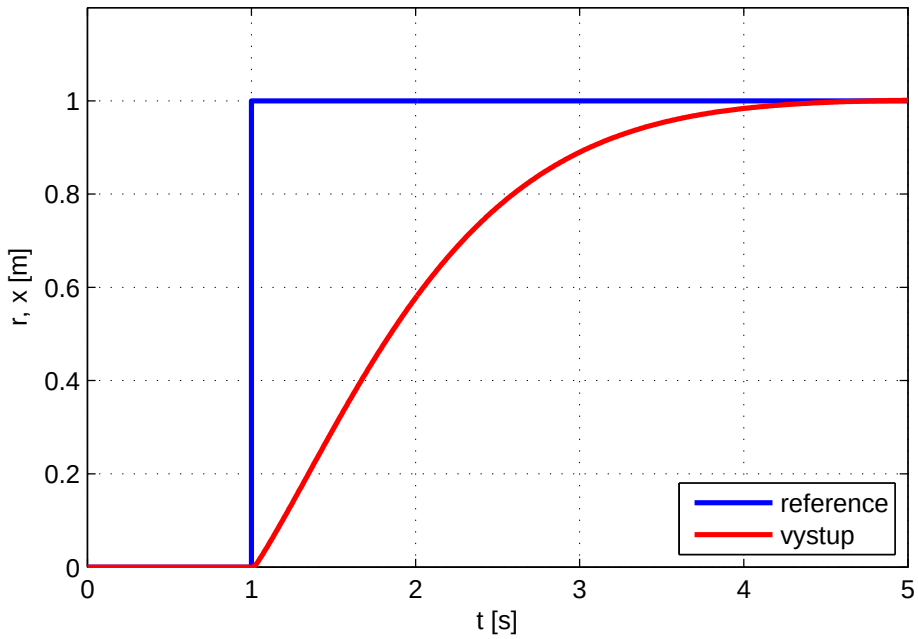
V této podkapitole je navržen PD regulátor pro model systému pneumatického výtahu s modelem obr. 3.4 a s přenosem (3.19). Tento regulátor je následně realizován jako funkční blok programu PLC v programovacím jazyce Simatic ST.

Na obr. 3.9 je schéma regulační smyčky. Model systému je opět žlutě, regulátor je označen zeleně. Červenou barvou je označena saturace, která reprezentuje skutečnost že ventilátor řídíme pomocí PWM modulace, jejíž rozsah je $0 - 100\,\%$.

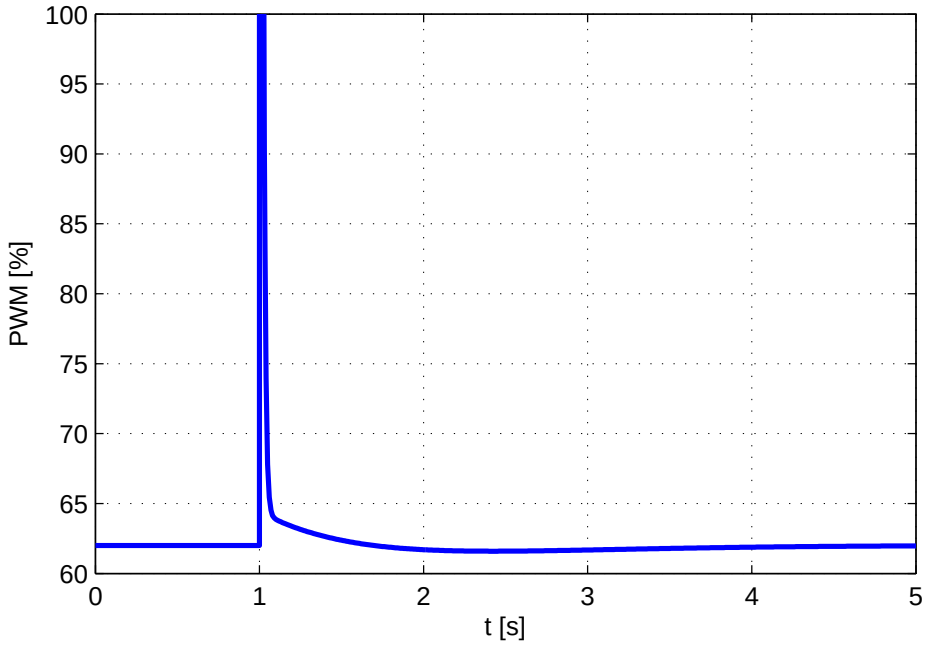



Obrázek 3.9: Schéma regulační smyčky

Od regulátoru požadujeme co nejrychleší dosažení požadované polohy bez překmitu. Tyto požadavky splňuje PD regulátor s proporcionální složkou $k_p = 5$ a derivační složkou $k_d = 5$. Při návrhu jsme použili nástroj SISOTOOL na přenos (3.19) a vytvarovali požadovanou odezvu uzavřené smyčky na jednotkový skok. Potom jsme provedli simulace s omezením akčního zásahu saturací dle schématu na obr. 3.9. Omezení akčního zásahu celou regulační smyčku zpomalilo, proto jsme museli konstanty, které jsme navrhli v SISOTOOLu, ještě zvětšit. Výsledkem je navržená odezva obr. 3.10 na jednotkový skok referenčního sinálu. Na obr. 3.11 je akční zásah vstupující do modelu.




Obrázek 3.10: Odezva uzavřené regulační smyčky na jednotkový skok reference




Obrázek 3.11: Akční zásah regulátoru

## 3.3 Řídící aplikace

V této části je podrobně rozebrána řídící aplikace modelu pneumatických výtahů. Programovatelný logický automat řady Simatic S300 je naprogramován pomocí standardního vývojového prostředí firmy Siemens Simatic Step7. Prostředí využívá programovacích jazyků dle normy IEC EN 61131-3. My jsme k programování použili žebříčkové diagramy (Ladder Diagram), assembler (Instruction List) a funkční bloky (Function Blok Diagram). Podrobnosti o těchto programovacích jazycích pro průmyslové automaty lze nalézt přímo v mezinárodní normě (IEC 1131, 1993).

### 3.3.1 Datový blok výtahu

Tabulka 3.1: Datový blok výtahu

| **Address** | **Name** | **Type** | **Initial value** |
|---|---|---|---|
| 0.0 | | STRUCT | |
| +0.0 | tlacitka | UDT1 Type_Tlacitka | |
| +2.0 | tlacitka_old | UDT1 "Type_Tlacitka" | |
| +4.0 | stop_table | UDT2 "Type_StopTable" | |
| +6.0 | kam | UDT3 "Type_Kam" | |
| +8.0 | senzory | UDT4 "Type_Senzory" | |
| +12.0 | kde | UDT5 "Type_Kde" | |
| +14.0 | konstanty | UDT6 "Type_Konstanty" | |
| +52.0 | regulator | UDT7 "Type_Regulator" | |
| +72.0 | PWM | DINT | L#50 |
| +76.0 | reference | INT | 0 |
| +78.0 | zmena | BOOL | FALSE |
| =80.0 | | END_STRUCT | |

Každému ze tří výtahů jsme vytvořili jeden datový blok. V tomto datovém bloku jsou uloženy všechny informace potřebné k řízení výtahu. Obsah tohoto datového bloku je v tab 3.1. Pro zjednodušení orientace v datovém bloku jsme použili uživatelské datové typy (User Data Types). Pomocí nich jsou informace v datovém bloku rozděleny do logických skupin.

V proměnné "tlacitka"jsou uloženy informace, která tlačítka jsou v tomto scan cyklu PLC sepnuta. Proměnná "tlacitka_old"obsahuje informaci o sepnutých tlačítkách o jeden scan cyklus dříve. Porovnáním těchto dvou proměnných lze zjistit, zda bylo nějaké tlačítko stisknuto, či uvolněno.

Proměnná "stop_table"obsahuje tabulku požadavků na zastavení výtahu v konkrétních patrech. Procházením této tabulky se rozhoduje kam bude výtah vyslán. V proměnné "kam"je uloženo následující patro ve kterém má výtah zastavit.

Uvnitř proměnné "senzory"je přechováván obraz výstupů všech senzorů náležících příslušnému výtahu. Jedná se tedy o stav čtyřech optických senzorů v jednotlivých patrech a o aktuální výšku kabiny odečtenou z ultrazvukového senzoru vzdálenosti.

Proměnná "kde"obsahuje informaci ve které části výtahové šachty se nachází kabina výtahu. Tato informace není spojitá, nýbrž připouští jen devět stavů. Kabina je buď v jednom ze čtyř pater, mezi nimi, nebo pod prvním patrem, či úplně nahoře nad čtvrtým patrem. Sektor ve kterém se kabina výtahu nachází se samozřejmě počítá z aktualní polohy kabiny z ultrazvukového senzoru polohy. Tato informace slouží k rozhodování do kterého patra bude výtah pokračovat v jízdě a k nastavování čísla aktuálního patra na segmentovém displayi.

V části datového bloku nazvané "konstanty"jsou uloženy všechny konstanty potřebné k provozu výtahu. Jedná se o referenční výšky jednotlivých pater, konstanty nastavení regulátoru polohy, atd. Tyto konstanty jsou nastaveny při restartu PLC v programovén bloku OB100 a sám program je už pak nemění, jen z nich čte.

V bloku nazvaném "regulator"jsou uloženy proměnné potřebné pro výpočet akčního zásahu regulátoru polohy kabiny výtahu. Lze se z něj číst aktuální regulační odchylku, hodnotu integračního členu, normovaný akční zásah, akční zásah v procentech PWM, atd. Tento blok není pro běžný provoz potřeba. Byl vyvořen pro potřeby ladění regulátoru.

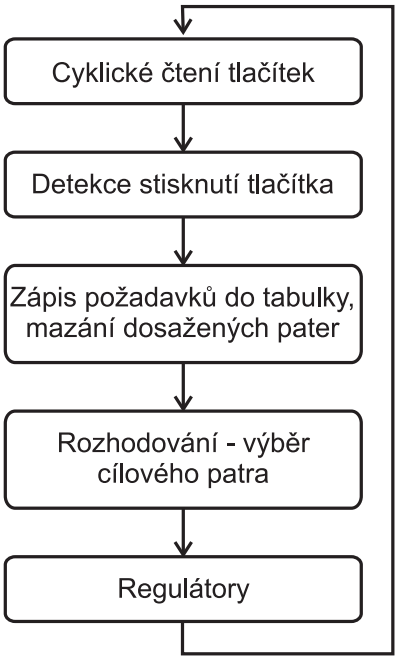
Celočíselná proměnná "reference"obsahuje výšku požadovaného patra, která je přivedena na vstup regulátoru polohy. Poslední proměnná typu boolean nazvaná "zmena"slouží ke spuštění rozhodovací logiky, která určí do kterého patra bude kabina výtahu pokračovat v jízdě. Program ji nastavuje do hodnoty logické jedničky, pokud dojde k události která vyžaduje přepočítání cílového patra (např. další požadavek na zastavení).

### 3.3.2 Hlavní programová smyčka OB1

Každému ze tří výtahů je tedy přiřazen datový blok popsaný v části 3.3.1. Základní algoritmus, který je vykonáván v každém scancyklu PLC je blokově vyjádřen na obr. 3.12.

Všechny naznačené úkony se provádějí nad všemi třemi datovými bloky výtahů. V následujících částech této kapitoly jsou jednotlivé bloky rozebrány podrobněji.

Mimo hlavní programovou smyčku OB1 je v programovém bloku cyklického přerušení OB35 s periodou 1 $ms$ spoušten program pro generování PWM signálu levého a pravého výtahu.



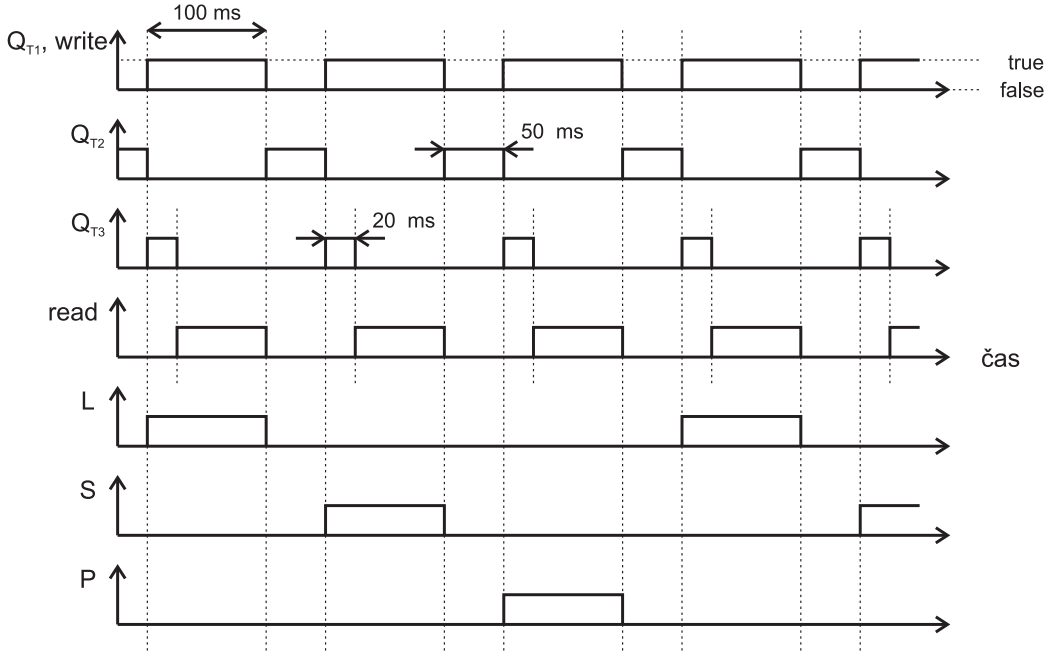

Obrázek 3.12: Kroky prováděné v programovém bloku OB1

#### 3.3.2.1 Cyklické čtení tlačítek

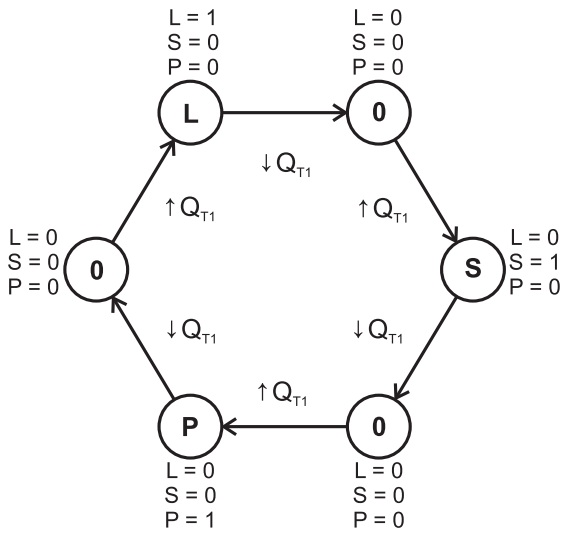
Každému z výtahů náleží jeden vstup/výstupní panel popsaný v kapitole 2.1.1. K jednotlivým panelům se přistupuje pomocí časového multiplexu. Časový průběh signálů použitých v podprogramu cyklického čtení a zápisu do vstup/výstupních panelů je na obr. 3.13.

Pomocí dvou časovačů "T1" a "T2" je generován obdélníkový průběh s periodou 150 ms

a střídou 2:1. Pomocí náběžných a spádových hran tohoto časového průběhu je řízen stavový automat (obr. 3.14), který přepíná mezi jednotlivými V/V panely. Signály "L", "S"a "P"jsou výstupními signály popsanými v tab 2.5, kterými se volí požadovaný V/V panel.




Obrázek 3.13: Časové průběhy signálů cyklického přístupu k V/V panelům




Obrázek 3.14: Stavový automat cyklického přístupu k V/V panelům

Tímto způsobem je zaručeno, že je aktivní vždy maximálně jeden ze signálů volby panelu. Mezistav (na obr. 3.14 označen "0") kdy není vybrán žádný panel byl přidán z důvodu "přeslechů"mezi panely. Bez tohoto mezistavu vždy na LED diodách panelu problikly hodnoty příslušící předchozímu panelu. Přidáním mezistavu byl tento problém při zápisu na stavové LED diody vyřešen.

Při čtení hodnot tlačítek tento problém přetrval. To bylo způsobeno tím, že aktuální hodnoty tlačítek panelu nejsou k dispozici ihned po náběžné hraně volícího signálu příslušného panelu. Takto se občas objevilo "falešné stisknutí talčítka". Proto byl do podprogramu cyklického přístupu k jednotlivým panelům zařazen další časovač "T3". Ten je spouštěn s náběžnou hranou časovače "T1"a odpočítává čas 20 ms. Teprve po uplynutí tohoto časového úseku se začne číst z příslušného panelu.

Všechny časové průběhy jsou zobrazeny na obr. 3.13. Signály "L", "S"a "P"je volen Levý, Střední a Pravý panel. Na vybraný panel se zapisují hodnoty stavových LED diod při aktivním signálu "write", který je ekvivalentní výstupu časovače "T1". Z vybraného panelu se čte při aktivním signálu "read", který je složen logickým součinem výstupu časovače "T1"a negace výstupu časovače "T3".

#### 3.3.2.2 Detekce stisknutí tlačítka, zápis požadavků do tabulky

Hodnoty tlačítek jsou cyklicky čteny z V/V panelu a zapisovány do proměnné "tlacitka"v datovém bloku výtahu. Zároveň jsou v tomto datovém bloku v proměnné "tlacitka_old"uchovávány hodnoty tlačítek z předchozího scan cyklu PLC (viz tab 3.1). Porovnáním těchto dvou proměnných je zjištěno které tlačítko bylo stisknuto.

Tabulka 3.2: Tabulka požadavků na zastavení výtahu - UDT2

| **Address** | **Name** | **Type** | **Initial value** |
|---|---|---|---|
| 0.0 | | STRUCT | |
| +0.0 | p1 | BOOL | FALSE |
| +0.1 | p2d | BOOL | FALSE |
| +0.2 | p2u | BOOL | FALSE |
| +0.3 | p3d | BOOL | FALSE |
| +0.4 | p3u | BOOL | FALSE |
| +0.5 | p4 | BOOL | FALSE |
| =2.0 | | END_STRUCT | |

Po detekci stiknutí tlačítka je nastaven příznak "zmena"v datovém bloku výtahu a příslušný bit v tabulce požadavků pro zastavení. Tato tabulka s pracovním nászvem "Stop Table"je definována uživatelským datovým typem "UDT2", který je zobrazen v tab 3.2. Jedná se o šest bitů. Každý odpovídá požadavku na zastavení v určitém patře. Koncovým patrům přísluší pouze jeden požadavek ("p1", "p4"), v druhém a třetím patře rozlišujeme požadavek na cestu nahoru ("p2n", "p3n") a dolů ("p2d", "p3d").

Tyto požadavky přímo odpovídají tlačítkům umístěným v jednotlivých patrech. Po stisku tlačítka v kabině výtahu se zapisuje dané patro s požadavkem dolů či nahoru podle aktuálního směru pohybu výtahu.

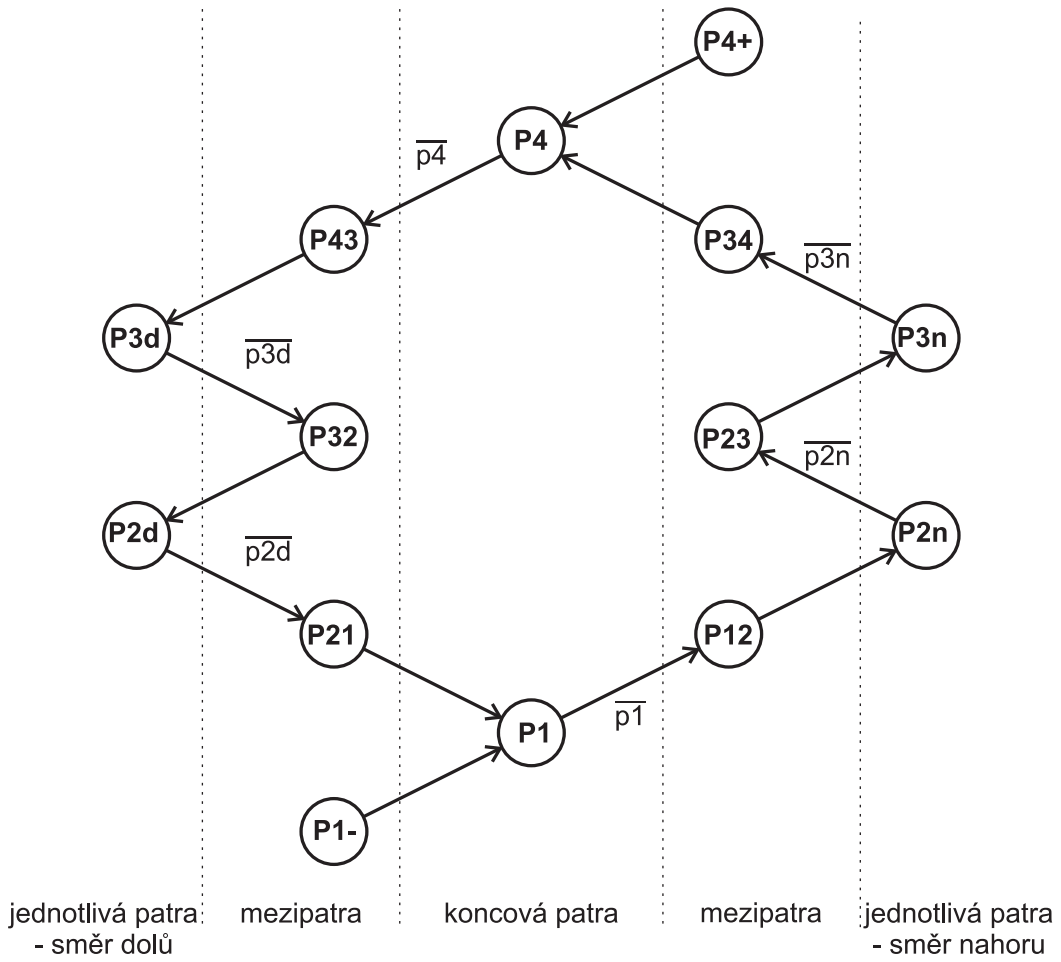
#### 3.3.2.3 Logika výběru cílového patra

Příznak "zmena"v datovém bloku výtahu je nastaven v případě že přišel nový požadavek na zastavení výtahu. To znamená, že je nutné přepočítat cílové patro výtahu. Podprogram výběru cílového patra v hlavním programovém cyklu OB1 je tedy spuštěn pouze pokud je tento příznak nastaven. Algoritmus výběru cílového patra se skládá z následujících čtyř kroků.

- výpočet části výtahové šachty ve které se kabina nachází
- nastavení počátečního patra stavového automatu
- spuštění stavového automatu výběru patra
- nastavení reference cílového patra regulátoru polohy

V prvním kroku se z údaje o poloze kabiny rozhodne ve které části výtahové šachty se kabina výtahu nachází. Chceme vědět jen zda je kabina přímo v konkrétním patře, nebo mezi kterými dvěma patry se nachází. Detekce je provedena porovnáním údaje z ultrazvukového čidla polohy s referenčními výškami jednotlivých pater, které jsou uloženy v proměnné "konstanty"v datovém bloku výtahu. Výsledek je opět uložen do datového bloku výtahu do proměnné "kde".

Na obr. 3.15 je stavový automat výběru cílového patra. Velkými písmeny jsou značeny jednotlivé stavy, malými jednotlivé bity tabulky požadavků na zastavení výtahu (viz. tab 3.2). Každému koncovému patru je přiřazen stav, druhému a třetím patru jsou přiřazeny stavy dva. Tyto stavy jsou rozlišeny směrem pohybu výtahu. Stejně tak jsou přiřazeny dva stavy každému mezipatru. Výtah se ještě může omylem vyskytnout pod prvním patrem a nad posledním patrem. Těmto možnostem jsou také přiřazeny stavy.




Obrázek 3.15: Stavový automat výběru cílového patra

Ve druhém kroku algoritmu je na základě spočtené informace o poloze výtahu a aktuálním směru jeho pohybu nastaven počáteční stav stavového auotmatu.

Před spuštěním stavového automatu ve třetím kroku je provedana kontrola neprázdnosti tabulky požadavků na zastavení výtahu. Tato kontrola zajistí konečnost stavového automatu. Z obr. 3.15 je jasné že v případě spuštění algoritmu s prázdnou tabulkou požadavků by se program zacyklil. Spuštěný stavový automat postupně prochází tabulku požadavků na zastavení a skončí v nejbližším stavu s požadavkem na zastavení.

Po skončení stavového automatu je v závislosti na koncovém stavu nastavena referenční výška cílového patra.

#### 3.3.2.4 Regulátor polohy kabiny výtahu

Realizujeme regulátor navržený v části 3.2. Regulátor je naprogramován v programovacím jazyce IL (Instruction List / assembler). Vstupem do regulátoru polohy je aktuální poloha kabiny ze snímače polohy "Y", referenční poloha kabiny "W" a proměnná "konst", která je typu UDT6 a obsahuje všechny konstanty regulátoru (tj. P,I,D složky regulátoru, pracovní bod, maximální a minimální akční zásah atd.). Výstupem je přímo hodnota PWM v procentech a proměnná "reg", která je typu UDT7. V této proměnné jsou uloženy všechny proměnné regulátoru pro potřeby ladění. Následuje výpis zdrojového kódu regulátoru.

```
L     #Y               // nacteni akt. polohy y(k)
L     #W               // nacteni pozadovane hodnoty
-I                     // e(k) = w(k) - y(k)
ITD                    // int -> double int
DTR                    // double int -> real
L     0.000000e+000    // offset
-R                     // odecteni offsetu
L     1.000000e-004    // normalizacni konstanta
*R                     // normalizace
T     #reg.E           // zapis normalizovane reg. odchylky e(k)
L     #reg.E_old       // nacteni reg. odchylky e(k-1)
-R                     // diference d(k) = e(k) - e(k-1)
T     #reg.dif         // zapis diference d(k)
L     #reg.E           // nacteni e(k)
T     #reg.E_old       // ulozeni e(k-1)
L     #konst.P         // proporcionalni slozka regulatoru
L     #reg.E           // nacteni e(k)
*R                     // k_p * e(k)
T     #temp            // ulozeni mezivypoctu
L     #konst.D         // derivacni slozka regulatoru
L     #reg.dif         // nacteni diferece d(k)
*R                     // k_d * d(k)
L     #temp            // nacteni mezivypoctu
+R                     // u(k) = k_p * e(k) + k_d * d(k)
T     #temp            // ulozeni mezivypoctu
L     #konst.I         // integracni slozka regulatoru
L     #reg.E           // nacteni reg. odchylky
*R                     // k_i * e(k)
L     #reg.intg        // nacteni integracniho clenu
+R                     // pricteni integracni casti
T     #reg.intg        // ulozeni integracniho clenu
L     #temp            // nacteni mezivypoctu P a D slozky
+R                     // pricteni I slozky regulatoru

L     #konst.az        // konstanta rozsahu akcniho zasahu
*R                     // vynasobeni rozsahem
L     #konst.prac_bod  // nacteni prac. bodu
+R                     // pricteni prac. bodu
T     #U_real          // zapis akcniho zasahu
T     #reg.u           // zapis kontroly akcniho zasahu
```

```
      L     #konst.max       // maximalni akcni zasah
      L     #U_real          // akcni zasah
      <=R                    // porovnani - konrola horni hranice a.z.
      JCN   j1               // podmineny skok na navesti j1
      L     #konst.max       // maximalni akcni zasah
j1:   T     #U_real          // navesti j1 - zapis hodnoty
      L     #konst.min       // minimalni akcni zasah
      L     #U_real          // akcni zasah
      >=R                    // porovnani - kontrola dolni hranice a.z.
      JCN   j2               // podmineny skok na navesti j2
      L     #konst.min       // minimalni akcni zasah
j2:   T     #U_real          // navesti j2 - zapis hodnoty
      RND                    // real -> double int
      T     #U               // zapis akcniho zasahu na vystup
```

Každému výtahu tedy přísluší jeden regulátor. Všechny tři regulátory jsou spouštěny pomocí časovače jednou za 100 ms. Tato vzorkovací perioda byla zvolena na základě časové odezvy ultrazvukových senzorů vzdálenosti, kterou výrobce udává $< 180$ ms.

#### 3.3.2.5 Generování PWM

Jak už bylo naznačeno o generování PWM se stará organizační blok cyklického přerušení OB35. V hardwarové konfiguraci je nastavena perioda jeho spouštění na $1\,ms$. Perioda komunikačního cyklu mezi řídícím PLC a periferií WAGO750 je $4\,ms$. Zkoušeli jsme nastavit spouštění bloku OB35 pro generování PWM signálu také na $4\,ms$. Ovšem prodloužení časové základny způsobovalo "pohupování" papírového válečku v trubici. Ukázalo se, že je lepší mít kratší časovou základnu a mít proměnné zpoždění na výstup.

V tomoto programovém bloku jsou spouštěny dva funkční bloky pro generování PWM. Jeden pro levý výtah a druhý pro pravý. Vstupem tohoto funkčního bloku je požadovaná střída typu INT v procentech. Tato porměnná tedy může nabývat 100 hodnot. Výstupem je přímo bit pro spínání PWM signálu. Uvnitř programového bloku se jednoduše inkrementuje vnitřní proměnná a porovnává se s požadovanou střídou. Dle výsledků komparace je nastavován výstupní bit.

## 3.4 Bezpečnostní program

Bezpečnostní program je spuštěn v bezpečnostním PLC 315F-2 PN/DP. Konfigurace bezpečnostního PLC je podrobně popsána v části 4.1.2. Bezpečnostní vstupy a výstupy

jsou popsány v tab 2.6.

Bezpečnostní program je spouštěn v organizačním bloku cyklického přerušení OB35. Perioda jeho spouštění je v hardwarové konfiguraci nastavena na $100\,ms$. Bezpečnostní program se skládá z následujících tří kroků.

Vyhodnocení stavu mobilního panelu a efektivních oblastí.
Vyhodnocení bezpečnostních tlačítek.
Vyhodnocení světel.

Pro každý tento krok byl vytvořen funkční blok. Tyto funkční bloky jsou podrobně rozebrány v následujících podkapitolách.

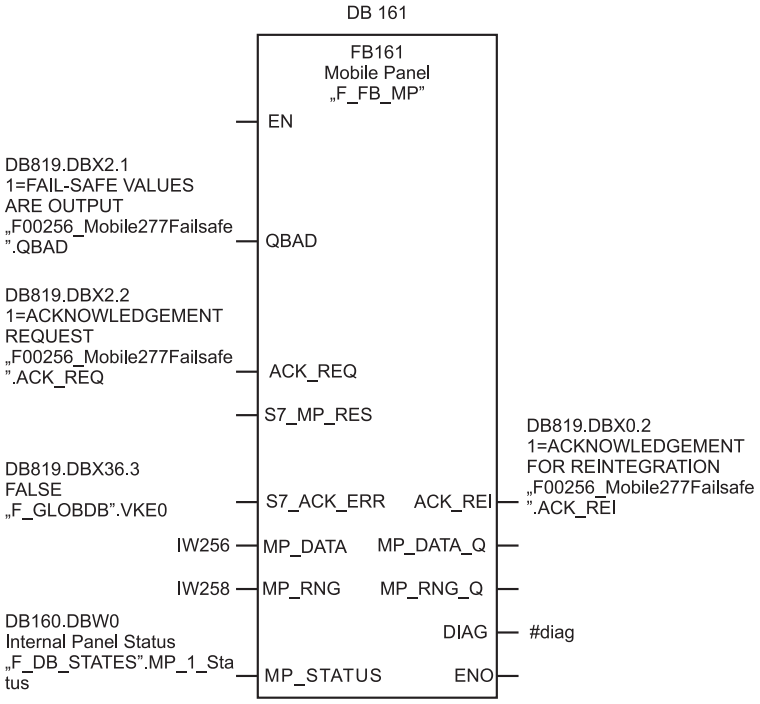
### 3.4.1 Vyhodnocení stavu mobilního panelu a efektivních oblastí

Na CD dodáném spolu s mobilním panelem MP277F popsaném v 2.2.4 je přiložená knihovna programu STEP7 s několika bezpečnostními funkčními bloky. Všechny tyto bloky jsou popsány v dokumentaci mobilního panelu (MP277F MANUAL, 2008) v kapitole 4.4. Pro naši úlohu je zapotřebí vložit do projektu dva z nich. První z nich je FB161 "Mobile panel". Tento funkční blok je zobrazen na obr. 3.16. Funkční blok představuje samotný mobilní panel. Je mu přiřazen datový blok DB161 a pomocí vstupů a výstupů tohoto funkčního bloku se nastavuje požadované chování panelu. Diskutovaná knihovna a dokumentace je na přiloženém CD.

Vstup QBAD například nastavuje výstupním datům status BAD. Vstup ACK_REQ určuje zda je požadováno potvrzení při reintegraci panelu. S7_ACK_ERR je potvrzování chyby při reintegraci. Další vstup MP_DATA určuje namapování vstupních dat panelu, MP_RNG obsahuje identifikátor efektivní oblasti ve které se mobilní panel nachází. Posledním vstupem je samotný stav mobilního panelu MP_STAT. Výstupy jsou následující. Potvrzení při reintegraci ACK_REI, výstupní data panelu MP_DATA_Q a identifikátor přihlášené efektivní oblasti MP_RNG_Q. Posledním výstupem je slovo DIAG, ze kterého se lze dozvědět stav mobilního panelu.

Použité nastavení mobilního panelu na obr. 3.16 je standardním nastavením s automatickou reintegrací mobilního panelu převzané z dokumentace (MP277F MANUAL, 2008). V zapojení mimo datového bloku mobilního panelu DB161 figurují ještě dva bezpečnostní

datové bloky. Jedná se o bloky DB818 a DB819. Tyto bezpečnostní datové bloky nevytváří uživatel, ale náleží samotnému bezpečnostnímu programu a jsou vytvářeny při kompilování hardwarové konfigurace bezpečnostního PLC.



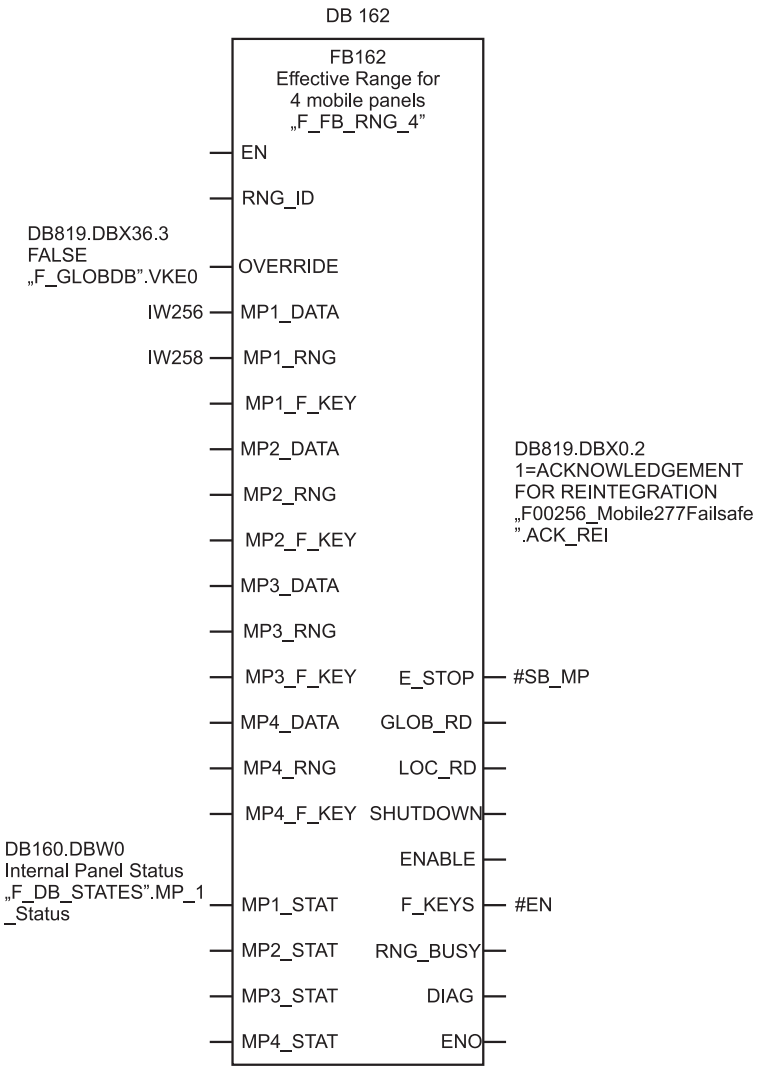

Obrázek 3.16: Funkční blok mobilního panelu FB161

Druhým použitým bezpečnostním funkčním blokem je FB162, který je znázorněn na obr. 3.17. Tento funkční blok přísluší konkrétní efektivní oblasti. V naší konfiguraci máme jen jednu efektivní oblast - oblast pneumatického výtahu (viz 6.1). Proto máme v tomto bezpečnostním programu pouze jeden funční blok FB162. Pokud budou v budoucnu do projektu přidány další efektivní oblasti příslušející dalším modelům v laboratoři, bude do tohoto místa programu nutné přidat pro každou efektivní oblast další blok FB162.

Nejdůležitějším vstupem je číslo efektivní zóny, kterou daný funkční blok reprezentuje. Toto číslo musí odpovídat číslu nastavenému ve vizualizaci ve WIN CC flexible. Pomocí vstupu OVERRIDE lze celou efektivní zónu potlačit. Dalšími vstupy jsou vstupní data mobilního panelu MP1_DATA a číslo efektivní oblasti ve které se daný panel nachází MP1_RNG. Funkčnímu bloku FB162 lze přiřadit až čtyři mobilní panely. Pro ně slouží následující nezapojené vstupy. Posledním vstupem je stav mobilního panelu MP1_STAT.

Hlavními výstupy panelu jsou všechny čtyři bezpečnostní stavy popsané v 2.2.4.2.

Jedná se o "Emergency Stop"(E_STOP), "Global RampDown"(GLOB_RD), "Local RampDown"(LOC_RD) a "ShutDown"(SHUTDOWN). Výstup ENO indikuje zda je daná efektivní oblast v pořádku vyhodnocena. Další výstupy indikují zda je v ní přihlášen mobilní panel (RNG_BUSY), zda je odemčen zámek na mobilním panelu (F_KEYS) a posledním výstupem je diagnostické slovo DIAG.

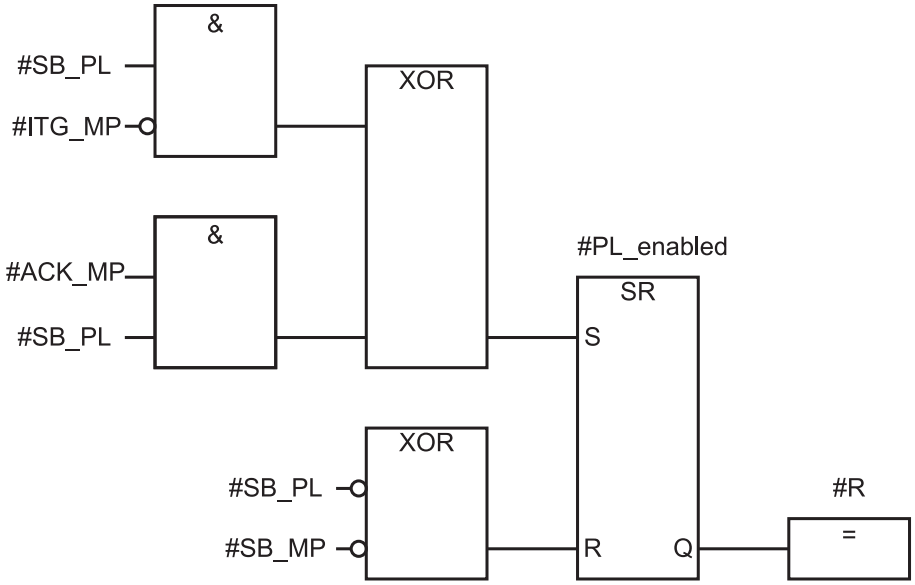



Obrázek 3.17: Funkční blok efektivní oblasti FB162

### 3.4.2 Vyhodnocení bezpečnostních tlačítek

Funkční blok pro vyhodonocení bezpečnostních tlačítek byl napsán v programovacím jazyku funkčních bloků. Důvod je při pohledu na jeho zapojení na obr. 3.18 zřejmý. Základ

tvoří RS klopný obvod jehož výstupem je spínáno relé napájející celý model pneumatických výtahů. Na vstup "R"klopného obvodu, který vypíná napájení, jsou jednoduše připojena bezpečnostní tlačítka. Tlačítko SB_PL je umístěno přímo na modelu pneumatických výtahů a SB_MP je stop tlačítko umístěné na mobilním panelu. Pro vypnutí napájení tedy stačí stisknout jedno z nich. Napájení je opět sepnuto při nastavení klopného obvodu pomocí vstupu "S". Tento případ nastane při uvolnění stop tlačítka na modelu. To ovšem jen za podmínky že mobilní panel není integrován v bezpečnostním systému (je vypnutý). V případě že mobilní panel integrovaný je, je ještě vyžadováno potvrzení spouštěcími tlačítky z mobilního panelu.

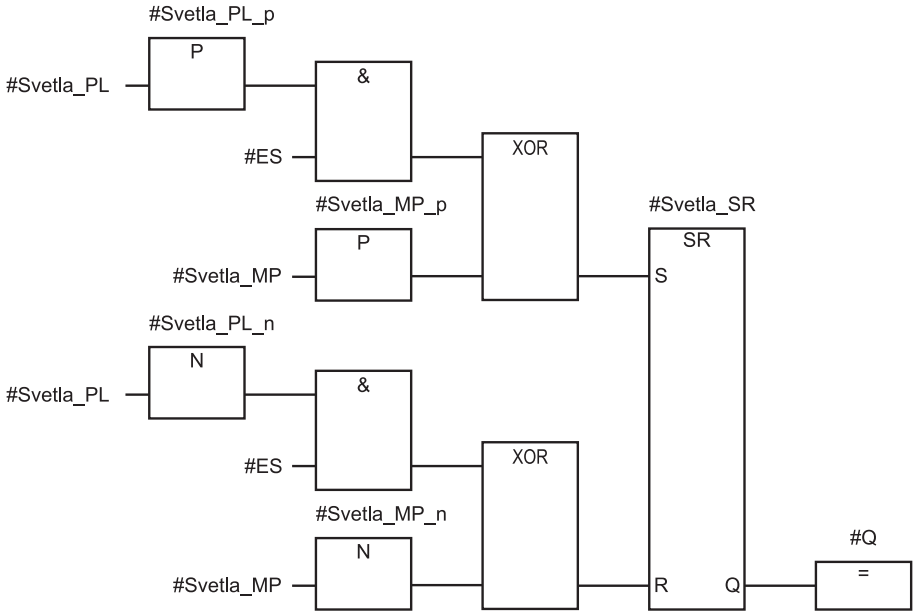



Obrázek 3.18: Funkční blok vyhodnocení bezpečnostních tlačítek

### 3.4.3 Rozsvěcení světel

Funkční blok pro rozsvěcení světel je také naprogramován v jazyce funkčních bloků. Jeho základem je opět RS obvod, který spíná relé, které připojuje světla k napájení. Světla jsou spínána buď požadavkem studenta z modulu WAGO, tato proměnná je pojmenována "Svetla_PL", nebo z vizualizace na mobilním panelu pomocí proměnné "Svetla_MP". Světla jsou spínána pomocí náběžných hran těchto proměnných a vypínána pomocí spádových hran. Zároveň je proměnná "Svetla_PL"podmíněna stavem "Emergency Stop"(ES), který je aktivní v nule. Tím je dosaženo toho, že při rozsvěcených světlech po stisku bezpečnostního tlačítka světla nezhasnou. To je základní reálný po-

žadavek stavu "Emergency Stop". Zatímco technologie (model) je odstaven, světla musí dál svítit.




Obrázek 3.19: Funkční blok pro rozsvěcení světel

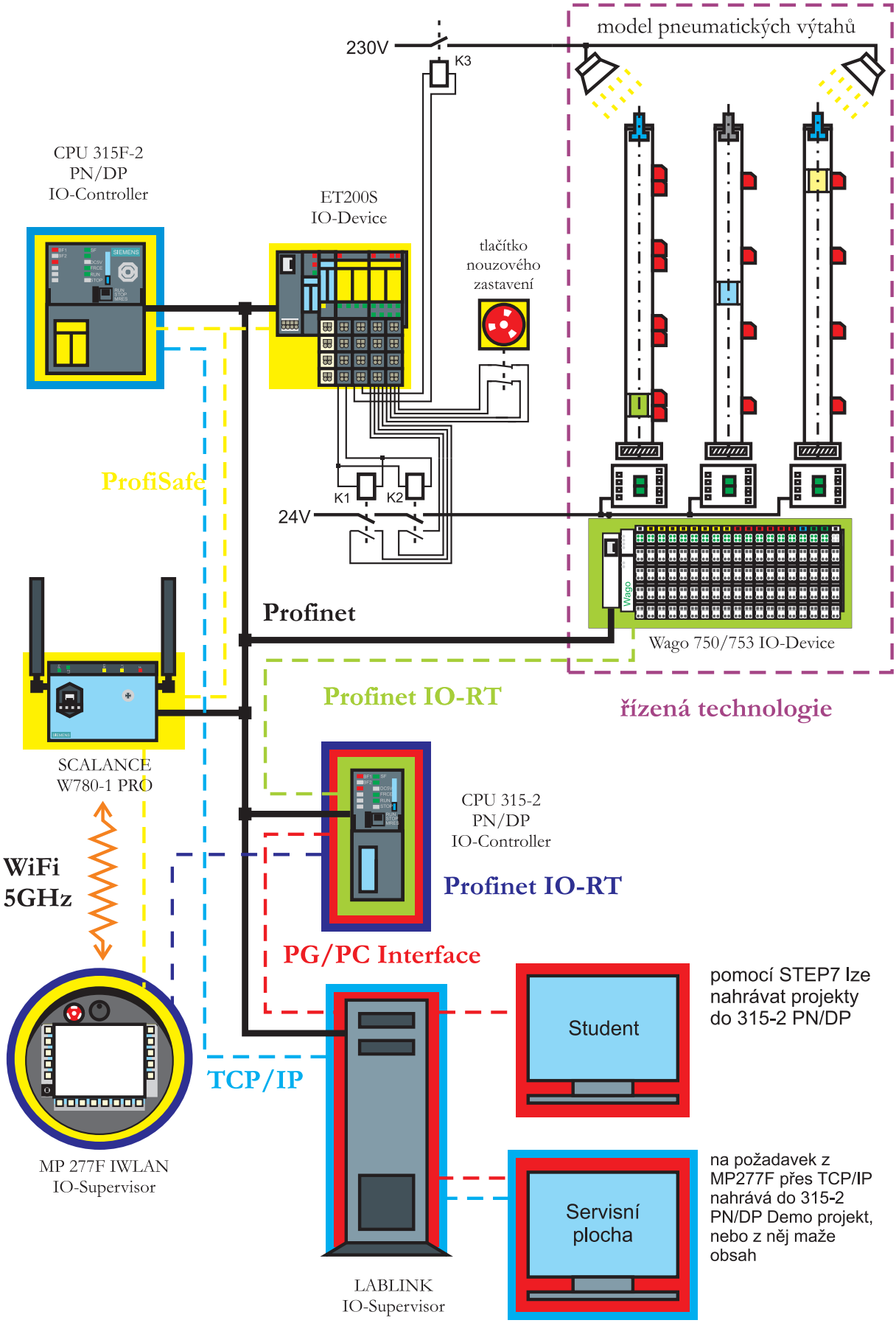
# Kapitola 4

# Vzdálené ovládání modelu

V této kapitole je popsáno realizované řešení vzdáleného přístupu k modelu. V první části kapitoly 4.1 je popsáno fyzické zapojení všech komunikujících statnic a postupně rozebrány všechny druhy navázaných spojení mezi stanicemi. Druhá část kapitoly 4.2 pojednává o výsledném přístupu k modelu skrze server pro podporu vzdálené výuky LabLink.

## 4.1 Zapojení celého systému

Na obr. 4.1 je vyobrazeno zapojení celého systému. Skoro všechny komunikující stanice jsou propojeny klasickým metalickým fast ethernetem. Pouze mobilní panel je připojen prostřednictvým průmyslového AP SCALANCE pomocí WiFi sítě na frekvenci $5\,GHz$. Komunikace na sběrnici je řízena protokolem Profinet. Tento protokol umožňuje i souběžnou komunikaci pomocí protokolu TCP/IP, který je například použit k předávání požadavku mezi mobilním panelem a serverem LabLink. Jednotlivá spojení stanic jsou na obr. 4.1 rozlišena barevně. Tato spojení jsou v následujících částích této kapitoly popsána podrobněji.




Obrázek 4.1: Schéma zapojení komunikujících stanic

### 4.1.1 Řídicí systém

Tento komunikační kanál je na obr. 4.1 vyznačen zelenou barvou. Komunikace je řízena protokoem Profinet IO-RT a účastní se jí následující zařízení.

IO-Controller - CPU 315-2 PN/DP
IO-Device - Wago 750

Toto spojení slouží k cyklické výměně dat mezi PLC řídícím model a periferií WAGO, ke které jsou připojeny všechny technologiceké vstupy a výstupy. Právě tyto dvě zařízení a jejich komunikaci konfiguruje ve svém projektu student.

### 4.1.2 Bezpečnostní systém

Bezpečnostní systém je na obr. 4.1 vyznačen barvou žlutou. Ke komunikaci je použit protokol PROFIsafe, který je nadstavbou protokolu Profinet IO-RT. Komunikace se účastní následující zařízení.

IO-Controller - CPU 315F-2 PN/DP
IO-Device - ET200S
IO-Supervisor - MP 277F IWLAN

V bezpečnostním PLC běží bezpečnostní program, který celou technologii pneumatických výtahů (samotný model a periferii WAGO750) připojuje k napájení. Jedno bezpečnostní tlačítko je připojeno k bezpečnostnímu vstupu ET200S a jedno je součástí mobilního panelu. K bezpečnostnímu výstupu jsou také připojena světla modelu. Ta musí být také obsluhována bezpečnostním PLC, protože po odstavení technologie stisknutím bezpečnostního tlačítka musí rozsvícená světla zůstat svítit.

### 4.1.3 Vizualizace

Pro účely vizualizace přistupuje mobilní panel pomocí protokolu Profinet IO-RT do paměti PLC řídícího model. Jedná se o komunikaci mezi stanicemi IO-Supervisor a IO-Controller. Na obr. 4.1 je tato komunikace vyznačena modře a účastní se jí následující stanice.

IO-Supervisor - MP 277F IWLAN
IO-Controller - CPU 315-2 PN/DP

### 4.1.4 Komunikace TCP/IP

Z vizualizace na mobilním panelu je možné poslat programu běžícímu na servisní ploše serveru LabLink požadavek na nahrání vzorového řídicího programu do řídicího PLC. Mobilní panel je v podstatě PC s operačním systémem Windows CE 5.0. Na tento systém byl nainstalován program *NetCat* pomocí kterého je jednoduše na protokolu TCP/IP odeslán požadavek na IP adresu serveru LabLink na určitém portu. Program který běží na serveru odposlouchává tento port a po detekci požadavku nahraje do řídicího PLC vzorový řídicí program. Tato komunikace je na obr. 4.1 vyznačena světle modře (cyan). Komunikace se účastní následující dvě stanice.

MP 277F IWLAN
Server LabLink - servisní plocha

### 4.1.5 Nahrávání projektů z LabLinku

Poslední druh komunikace je na obr. 4.1 vybarven červenou barvou. Jedná se o PG/PC interface. Pomocí tohoto rozhraní přistupuje aplikace STEP7 k PLC. Pomocí tohoto spojení program běžící na servisní ploše serveru LabLink vyprazdňuje paměť řídícího PLC a na požadavek z vizualizace na mobilním panelu nahrává vzorový řídící program. V neposlední řadě prostřednictvím tohoto spojení pracuje student s PLC. Komunikace se tedy účastní následující stanice.

CPU 315-2 PN/DP
Server LabLink - servisní plocha
Server LabLink - studentská pracovní plocha

## 4.2 Popis práce se serverem LabLink

Na serveru LabLink tedy běží dvě vzdálené plochy. Jedna servisní, která obstarává mazání paměti PLC a druhá studentská pracovní. Obě plochy disponují dvěmi síťovými připojeními. Jedno připojení je určeno pro přístup uživatelů z internetu ke vzdálené ploše a druhé k připojení samotného modelu.

### 4.2.1 Servisní plocha

Na servisní ploše je nainstalován program STEP7 a jsou na ní umístěny skripty *Clear PLC* a *Load DEMO* popsané v části 5.2.4. Na vzdálené ploše běží program, který sleduje následující události.

- Přihlášení studenta k pracovní ploše

  Po přihlášení studenta k pracovní ploše je spuštěn skript *Clear PLC*, který vymaže obsah paměti PLC. Student má tedy k dispozici "prázdné" PLC bez HW konfigurace a programových bloků předchozího uživatele.

- Požadavek na nahrání ukázkového programu

  Když tento program detekuje dotaz na příslušném portu, spustí skript *Load DEMO*. Mobilní panel po stisknutí příslušného tlačítka ve vizualizaci na tuto adresu odešle požadavek. Takto je zprostředkováno nahrávání vzorového řídicího programu z vizualizačního panelu.

Zde zmíněný program běžící na servisní vzdálené ploše není předmětem této diplomové práce. Jeho autorem je pan Ondřej Fiala, správce serveru LabLink.

## 4.3 Studentská pracovní plocha

Studentská pracovní plocha je hlavním prostředkem vzdálené laboratoře. Právě k této ploše se přihlašují studenti a jejím prostřednictvým pracují s modelem. Na pracovní ploše neběží žádné speciální programy, nebo skripty. O vše se starají programy běžící na servisní ploše. Na pracovní ploše je pouze nainstalováno programovací prostředí SIMATIC STEP7, které slouží k programování PLC řídícího model. Pomocí PG/PC interface nastaveného na TCP/IP se student připojí k PLC a může ho plně ovládat. Tzn. měnit provozní stavy a nahrávat, či mazat jednotlivé funkční bloky.

# Kapitola 5

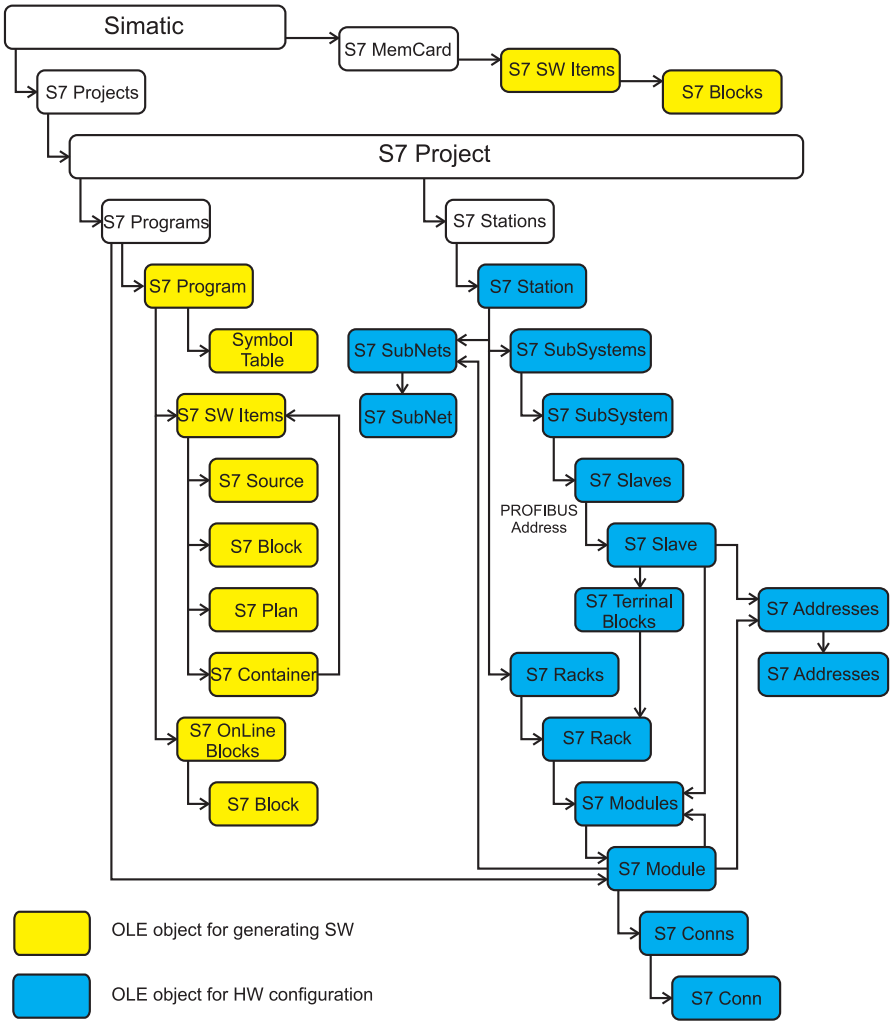
# Nahrávání HW konfigurace do PLC

Součástí programu SIMATIC STEP7 je knihovna *Command Interface*. Její součástí je i dokumentace (Command Interface Help, 2008). Prostřednictvím této knihovny je možné z vyššího programovacího jazyka přistupovat k PLC. To je možné ovládat, nahrávat do jeho paměti a mazat z jeho paměti jednotlivé části projektů. Tato knihovna je popsána v části 5.1.

Ve výjovém prostředí Microsoft Visul Studio 2010 byly vytvořeny dvě aplikace v programovacím jazyce C#. Jedna aplikace slouží k vymazání programu z PLC a druhá k nahrání vzorové řídicí aplikace do PLC. Obě tyto aplikace si jsou velmi podobné a jsou podrobně rozebrány v části 5.2 této kapitoly.

## 5.1 Knihovna Command Interface

Knihovna *Command Interface*, jak už slovo *Interface* naznačuje, pouze zprostředkovává jednotlivé funkce prostředí Siemens SIMATIC STEP7. Největší nevýhodou této knihovny je, že vedle výsledné aplikace musí být na počítači řádně nainstalován program SIMATIC STEP7. Výhodou tohoto řešení je jednoduchost práce s knihovnou. V programu se vytvoří objekt typu SIMATIC S7. Tento objekt obsahuje seznam všech projektů v programu STEP7. Z tohoto seznamu si vybereme náš projekt a ten můžeme jednoduše nahrát do PLC. Nemusíme nastavovat žádné parametry. Všechny parametry (např. nastavení PG/PC interface, atd.) jsou implicitně převzány z otevřeného projektu. Na obr. 5.1 je zobrazena základní struktura objektů knihovny *Command Interface*. Obrázek je převzán z dokumentace (Command Interface Help, 2008).




Obrázek 5.1: Struktura objektu Simatic

Žlutě jsou vyznačeny části projektu, které se nahrávají do paměti PLC. Modré bloky slouží k vytváření HW konfigurace. Jak je patrné z obrázku, rodičem celé třídy je objekt *Simatic*. Ten ukazuje na seznam všech dostupných projektů *S7 Projects*, ze kterých si vybereme námi požadovaný konkrétní projekt *S7 Project*. Projekt obsahuje seznam stanic *S7 Stations* a seznam programů *S7 Programs*. Stanice a všechny její dědicové slouží k tvorbě HW konfigurace. Každá stanice má metodu *Compile()*, která z dané stanice vytvoří datový blok HW konfigurace, který se objeví mezi programovými bloky na úrovni objektu *S7 Source*.

Na obrázku je struktura HW konfigurace znázorněná modrými bloky. Tato knihovna je postupně vytvářena spolu s programovacím prostředím STEP7, proto ve struktuře figurují stanice *S7 Slaves*. To je tím že v době vytváření této knihovny byla koncipována

hlavně pro technologii *Profibus*. Knihovna dnes samozřejmě umí vytvářet i HW konfigurace využívající *Profinet IO*. My ovšem používáme předpřipravený projekt, takže nepotřebujeme upravovat v programu HW konfiguraci. A proto se nemusíme její tvorbou dále zabívat.

Dále nás zajímá seznam programů *S7 Programs*. Z něj si vybereme náš program. Pro nás jsou zajímaví dva dědici objektu *S7 Program*. Prvním je struktura *S7 SW Items*, která obsahuje všechny části programů. V této struktuře jsou uloženy všechny jednotlivé funkční bloky, datové bloky, funkce, uživatelské datové typy a zkompilovaná HW konfigurace. Každý z těchto objektů vlastní metodu *Download()* pomocí které se nahraje do paměti PLC.

Druhá datová struktura je typu *S7 OnLine Blocks* a obsahuje všechny datové bloky nahrané uvnitř paměti PLC. Jejich procházením a voláním metody *Remove()* odstraňujeme jednotlivé datové bloky z paměti PLC. V aplikaci dále využíváme některé další metody datového typu *S7 Project*. Jedná se především o metody *NewStart()* a *Stop()*, pomocí nichž ovládáme stav PLC.

Knihovna *Command Interface* umožňuje nastavit prakticky vše, co je možné nastavit v programu STEP7. Z obrovského množství typů objektů a metod využíváme pouze nepatrný zlomek, který nám umožňuje ovládat PLC a nahrávat či mazat části programů.

## 5.2 Aplikace

Cílem je vytvořit dvě aplikace. Jedna bude sloužit k mazání paměti PLC a druhá pro nahrávání vzorového projektu do PLC. Výsledkem by měli být dva spustitelné soubory. Tyto soubory budou spouštěny ze servisní vzdálené plochy běžící na serveru LabLink. Skript pro vymazání paměti bude spouštěn vždy když se přihlásí student ke vzdálené ploše. Tak docílíme toho, že student bude mít k dispozici "prázdné PLC". Skript pro nahrání vzorové aplikace bude spouštěn na požadavek z vizualizačního panelu.

### 5.2.1 Požadavky a vytvoření projektu

Pro úspěšné vytvoření projektu s knihovnou *Command Interface* je nutné splnit následující požadavky a provést následující kroky

- Projekt ve vyšším programovacím jazyce

Projekt jsme založili ve výjovém prostředí Microsoft Visual Studio 2010 v programovacím jazyce C#. Jelikož se jedná o skript, který se spustí a po úspěšném provedení sám uzavře, založili jsme projekt jako konzolovou aplikaci.

- Nainstalovaný program STEP7

  Program máme řádně nainstalován.

- K projektu připojit knihovny *Simatic* a *S7hcom_x*

  Obě knihovny jsou umístěny v adresáři programu STEP7. Konkrétně

  $$STEP7\backslash S7bin\backslash s7abatcx.dll$$

  $$STEP7\backslash S7bin\backslash s7hcom\_x.dll.$$

  V object browseru projektu je přidáme pomocí *add references*.

- Přidat do projektu cestu k projektům programu STEP7

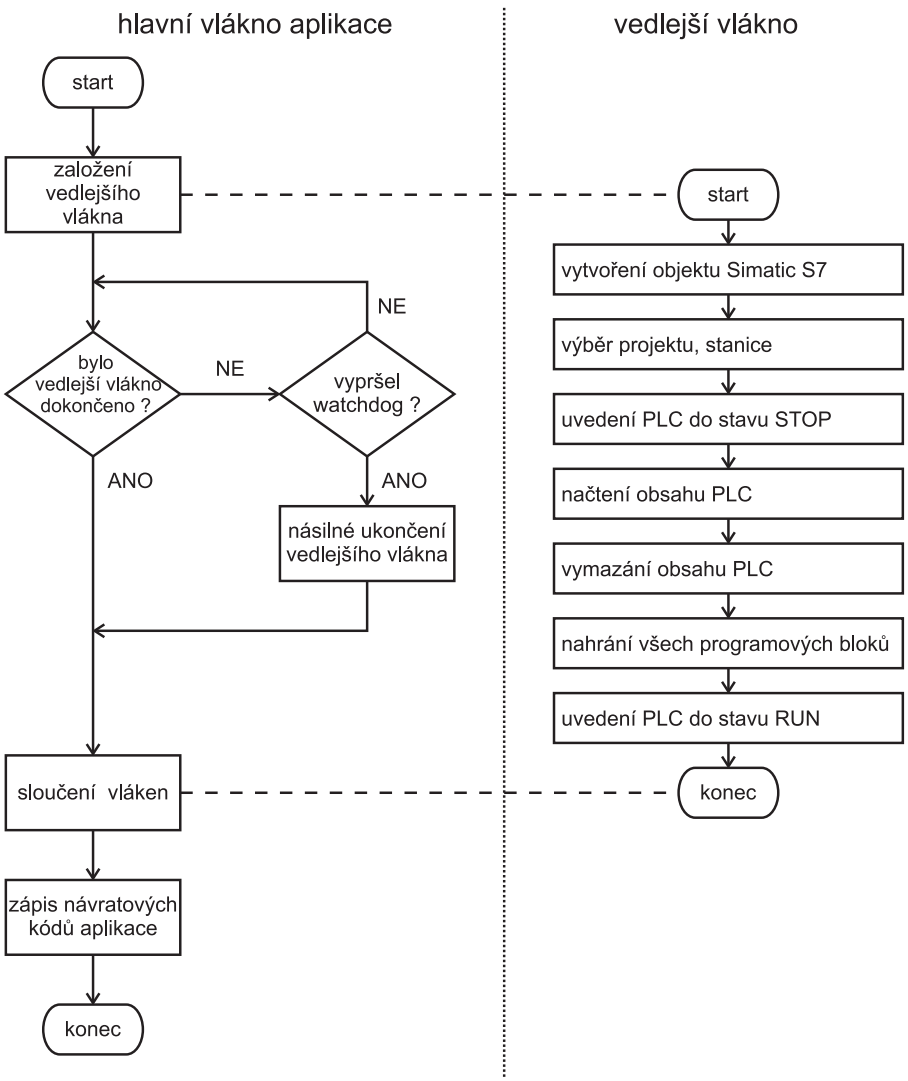
  Do projektu ve VS2010 přidáme cestu k uloženým projektům, např.

  $$C{:}\backslash Siemens\backslash Step7\backslash S7Proj.$$

  To provedeme v object browseru pomocí příkazu *add path*.

### 5.2.2 Vývojový diagram aplikace

Na obr. 5.2 je vývojový diagram aplikace. Ta se skládá ze dvou vláken. Hlavní výkonná část programu, která se připojí k PLC, smaže jeho paměť a nahraje do ní buď celý vzorový projekt, nebo jen prázdnou HW konfiguraci, běží ve vedleším vlákně. Hlavní vlákno aplikace pouze prostřednictvým časovače (watchdog) hlídá vedlejší vlákno a pokud vlákno není dokončeno před jeho vypršením, ukončí ho.




Obrázek 5.2: Vývojový diagram aplikace

### 5.2.3 Zdrojový kód aplikace

Celý projekt ve Visual Studiu 2010 je uložen na přiloženém CD. V této části práce je uvedeno jen pět úryvků kódu, které demonstrují provedení nejdůležitějších funkčních částí programu.

#### 5.2.3.1 Výběr projektu a programu

V následujícím výňatku ze zdrojového kódu je demonstrováno založení nového objektu typu *Simatic*, výběr projektu *PL_Demo* a programu *S7 Program(1)*.

```
S = new Simatic();        // Vytvoření objektu Sinamics

foreach (S7Project S7Proj in S.Projects)
            {
                if (S7Proj.Name.ToString() == "PL_Demo")
                {
                    m_projekt = S7Proj;
                }
            }                // Výběr projektu PL_Demo

foreach (IS7Program S7Prog in m_projekt.Programs)
            {
                if (S7Prog.Name == "S7 Program(1)")
                {
                    m_program = S7Prog;
                }
            }                // Výběr programu
```

#### 5.2.3.2 Načtení a vymazání datových bloků PLC

Další úryvek zdrojového kódu provede načtení datové struktury všech datových bloků v paměti PLC *m_OnLineBloky* a vymaže je z paměti PLC. Bloky, jejichž jména začínají na "SF"jsou vyfiltrovány a zachovány. Tyto bloky jsou systémové a spravuje je samo PLC.

```
m_OnLineBloky = m_program.OnlineBlocks;
                // Převzetí seznamu datových bloků v paměti PLC
try
  {
    foreach (S7Block S7OBlock in m_OnLineBloky)
      {                // Mazání všech datových bloků
        if (S7OBlock.Name.ToString()[0].Equals('S') &&
            S7OBlock.Name.ToString()[1].Equals('F'))
          {        // Bloky se jménem "SF" jsou systémové
            Console.WriteLine("Systemovy Blok: " +
            S7OBlock.Name + " - ZACHOVAN");
          }        // Nemažeme je, tyto bloky si vytváří PLC
        else
          {
            Console.Write("Blok: " + S7OBlock.Name);
            S7OBlock.Remove();
            Console.Write(" - SMAZAN\n");
          }        // Všechny ostatní datové bloky smažeme
      }
  }
catch (SystemException ex)
  {
    Console.WriteLine("Chyba: " + ex.Message);
    FAIL = true;
  }
```

#### 5.2.3.3 Nahrání funkčních bloků do PLC

Následující část kódu prochází datovou strukturu *S7 SW Items* a výbírá z ní datové, funkční a programové bloky. Tyto bloky následně nahrává do PLC metodou *Download(S7OverwriteFlags.S7OverwriteAsk)*. Parametr této metody potlačuje dotaz na přepsání existujících datových bloků. Pouze právě v této části kódu se aplikace pro nahrání vzorového projektu liší od aplikace pro vymazání paměti PLC. Aplikace pro mazání paměti PLC tuto část kódu neobsahuje.

```
foreach (S7SWItem S7Item in m_program.Next)
  {
    if (S7Item.Name == "Blocks")
      {
        foreach (S7Block S7Blok in S7Item.Next)
          {
            S7Blok.Download(S7OverwriteFlags.S7OverwriteAll);
            Console.WriteLine("Byl nahran blok '" +
            S7Blok.Name + "'");
          }
      }
   }
```

#### 5.2.3.4 Uvedení PLC do stavu STOP

Další úryvek zdrojového kódu demonstruje zjištění aktuálního stavu PLC a jeho uvedení do "Stop"stavu.

```
m_stav = m_program.ModuleState;  // Detekce aktuálního stavu PLC
Console.WriteLine("Stav PLC je: " + m_stav.ToString());

if (m_stav.ToString() != "S7Stop")
  {
    m_program.Stop();  // Uvedeni PLC do stop stavu
    m_stav = m_program.ModuleState;
    Console.WriteLine("PLC bylo uvedeno do stavu: "
    + m_stav.ToString());
  }
```

#### 5.2.3.5 WatchDog a ošetření vyjímek

Poslední úryvek kódu aplikace tvoří obsah třídy main, která implementuje watchdog. Všechny předchozí úryvky kódu pocházejí z metody *Download_Project*, která je spuštěna v novém vlákně pojmenovaném *plc*. Každou vteřinu hlavní vlákno kontroluje zda vlákno

*plc* ještě běží. Pokud už doběhlo, tak program pokračuje dále. V případě, že vlákno *plc* stále běží i po 120 vteřinách, je toto vlákno násilně ukončeno.

```
DONE = false;
FAIL = false;
string exit_text;
string exit_code;

FileStream fs = new FileStream("results.txt", FileMode.Append);
StreamWriter sw = new StreamWriter(fs);

Thread plc = new Thread(Download_Project);
plc.Start();

for (int i = 0; i != 120; i++)
  {
    Thread.Sleep(1000);
    if (!plc.IsAlive)
      {
        break;
      }
  }

if (plc.IsAlive)
  {
    plc.Abort();
    plc.Join();
  }

if (DONE)
  {
    if (!FAIL)
      {
        exit_text = "OK";
        exit_code = "OK";
      }
    else
      {
        exit_text = "ERROR : ended with exception : " + exception;
        exit_code = "ERROR:exception";
      }
  }
else
  {
    exit_text = "ERROR : WatchDog run out";
    exit_code = "ERROR:watchdog";
  }

sw.WriteLine(DateTime.Now.ToString() + "   lift_load   " + exit_text);
sw.Close();
Thread.Sleep(1000);
Console.WriteLine(exit_code);
```

Pomocí statických proměných *DONE* a *FAIL*, které nastavuje vlákno *plc*, je rozhodnuto zda aplikace proběhla v pořádku, byla ukončena vyjímkou, nebo ji násilně ukončil watchdog. Podle toho je na posledním řádku standardního výstupu, kam jsou vypisovány informace o průběhu programu, vypsán návratový kód buď *OK*, *ERROR:watchdog*, nebo *ERROR:exception*. Tato informace je použita jako zpětná vazba pro server LabLink, kde je programově kontrolováno zda skript proběhl správně. Pro účely diagnostiky se ještě informace o návratovém kódu připíše do textového souboru *results.txt*. Otevřením tohoto souboru získá administrátor informaci, kdy byl jaký skript spuštěn, její návratový kód a v případě vyjímky i podrobný text vyjímky, která aplikaci ukončila.

### 5.2.4 Výsledná aplikace

Výstupem z vývojového prostředí jsou dva spustitelné soubory

*Clear PLC* a
*Load DEMO* .

Spolu s nimi musí být ve složce umístěny tři knihovny

*Interop.S7HCOM_XLib.dll* ,
*Interop.S7NCIE_XLib.dll* a
*Interop.SimaticLib.dll* .

Tyto soubory jsou umístěny na servisní vzdálené ploše běžící na serveru LabLink. Program *Clear PLC* je spouštěn při přihlášení studenta. Tím docílíme, že student má k dispozici PLC s prázdnou pamětí. Druhý program *Load DEMO* je spouštěn na požadavek z vizualizace na mobilním panelu.

### 5.2.5 Dosažená funkčnost

Za největší nevýhodu zvoleného řešení považujeme nemožnost přístupu k PLC pomocí MAC adresy. Zde popsaný způsob totiž navazuje spojení nadefinované v projektu který je otevřen. Ten standardně přistupuje k PLC pomocí IP adresy. Knihovna *Command Interface* sice využívá funkce programu *Step7*. Ten umí nalézt PLC pomocí vestavěného DCP browseru a přiřadit mu požadovanou IP adresu. Ovšem knihovna *Command Interface* toto neumožňuje. Proto je studentům zdůrazňováno, že nesmí měnit IP adresu

PLC. Kdyby se tak stalo, tak oba vytvořené výše popsané skripty se ukončí s vyjímkou "nepodařilo se navázat spojení".

Mimo toto omezení vytvořené skripty fungují celkem spolehlivě. Kritická místa jsou ošetřena vyjímkami, takže nemohou způsobit pád programu. Při nahrávání HW konfigurace a programu do PLC je nastaven parametr *S7OverwriteFlags.S7OverwriteAll*, který potlačuje všechny dotazy na přepsání již existujících bloků. I tak se ovšem velmi zřídka může stát že při nahrávání programu vyskočí dotaz a program čeká na odpověď. Těmto situacím se snažíme předcházet. Tato situace například nastávala když student měl před ukončením rezervace aktivní diagnostické spojení a plocha byla ukončena rezervačním systémem. PLC potom vyžadovalo při dalším nahrávání potvrzení o ukončení diagnostického spojení. Jelikož server LabLink čekal na dokončení skriptu, žádná další rezervovaná vzdálená pracovní plocha nebyla vytvořena. Právě proto byl do programu přidán WatchDog, který v takovémto případě skript ukonční a rezervovaná plocha se vytvoří, ikdyž skript neproběhl správně. Díky návratové hodnotě skriptu na standartním výstupu se dá na tuto situaci reagovat. Například spustit skript znovu, nebo upozornit administrátora. Tato situace byla zároveň eliminována jednoduchým způsobem. Stačí spustit skript *Clear PLC* o pár vteřin dříve, než je ukončena studentská plocha. Pokud je PLC uvedeno do stavu STOP ještě před ukončením plochy, je dignostické spojení ukončeno a zde popisovaná situace nenastane.

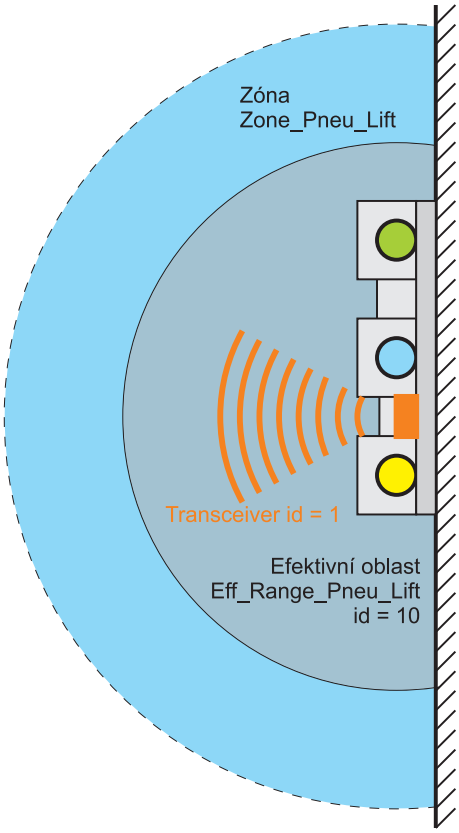
# Kapitola 6

# Vizualizace

Vizualizace běžící na mobilním bezpečnostním panelu MP 277F IWLAN popsaném v 2.2.4 je vytvořena v programu *WinCC flexible*. Při vytváření projektu je nutné nejdříve nadefinovat zóny a efektivní oblasti. Potom se nadefinují spojení s jednotlivými PLC. Pomocí těchto spojení už jednoduše přistupujeme do paměti jednotlivých PLC a pomocí tagů můžeme číst a zapisovat všechny proměnné.

## 6.1 Zóny a efektivní oblasti

Ze všeho nejdříve se v novém projektu musí nadefinovat zóny a efektivní oblasti. Na obr. 6.1 je naše uspořádání. Do těla modelu pneumatických výtahů byl nainstalován transpondér, který je na obr. 6.1 vyznačen oranžově. Na tomto transpondéru je pomocí přepínačů nastavena hodnota $id = 1$. V projektu jsme nadefinovali jednu zónu se jménem *Zone_Pneu_Lift* a svázali ji s transpondérem $id = 1$. Dále jsme nadefinovali jednu efektivní oblast *Eff_Range_Pneu_Lift* a svázali ji opět s transpondérem $id = 1$. Jak je vidět z obr. 6.1 námi nadefinovaná zóna a efektivní oblast se překrývají. To je vpořádku. Jak už bylo řečeno v 2.2.4.1 zóny přepíná podle přijímaného id transpondéru mobilní panel sám a lze jejich pomocí například přepínat obrazovky. Do efektivní oblasti se musí operátor po vkročení přihlasit jejím id. Teprve přihlášení do efektivní zóny aktivuje všechny bezpečnostní prvky panelu (až na tlačítko nouzového zastavení, to je aktivní vždy).




Obrázek 6.1: Zóny a efektivní oblasti

Každé nadefinované efektivní oblasti musí odpovídat jeden bezpečnostní funkční blok *FB162* spouštěný v bezpečnostním programu s příslušným id. Tento funkční blok je popsaný v kapitole 3.4.1. Pokud tomu tak není, mobilní panel není po zapnutí integrován do bezpečnostního systému. Také proto se při prvním spuštění vizualizace provádí kontrola efektivních oblastí. Operátor je vyzván ke vstoupení postupně do všech zón a efektivních oblastí. Z přijímaných id transpondérů vypočítá mobilní panel kontrolní součet, který je nutno zadat do projektu. Teprve po nahrání projektu se správným kontrolním součtem, který odpovídá naprojektovaným zónám a efektivním oblastem se spustí vizualizace.

Pokud se v budoucnu bude rozšiřovat vizualizace o další modely v laboratoři, bude ke každému dalšímu modelu instalován minimálně jeden transpondér a do projektu budou přidány další odpovídající zóny a efektivní oblasti, které budou svázány s vysílaným id transpondéru.

### 6.1.1 Spojení

Dále je nutné nadefinovat spojení vizualizačního panelu s jednotlivýmy PLC. Jedná se o dvě spojení. O spojení s bezpečnostním PLC 4.1.2, které je na obr. 4.1 vyznačeno žlutě. A o spojení s řídícím PLC modelu pneumatických výtahů 4.1.3, které je na obr. 4.1 vyznačeno modře. Pokud bude v budoucnu do vizualizace integrován další model, jednoduše se nadefinuje další spojení s jeho řídícím PLC.

Pomocí těchto nedefinovaných spojení už lze jednoduše vytvářet tagy a tak přímo přistupovat do pamětí jednotlivých PLC.

## 6.2 Obrazovky

Vizualizace sestává z několika obrazovek. Nejdříve je nutné vytvořit šablonu (template), která je zobrazena na všech obrazovkách. V této šabloně je řešeno přepínání obrazovek, zobrazení stavu panelu, přítomnost v zóně, přihlášená efektivní oblast, atd. Dále jsme vytvořili úvodní obrazvku, která je spuštěna po startu vizualizace, hlavní ovládací obrazovku modelu s ovládacími prvky všech tří výtahů, po jedné obrazovce s podrobnostmi o daném stavu výtahu a nakonec obrazovku pro ovládání PLC modelu pneumatických výtahů.

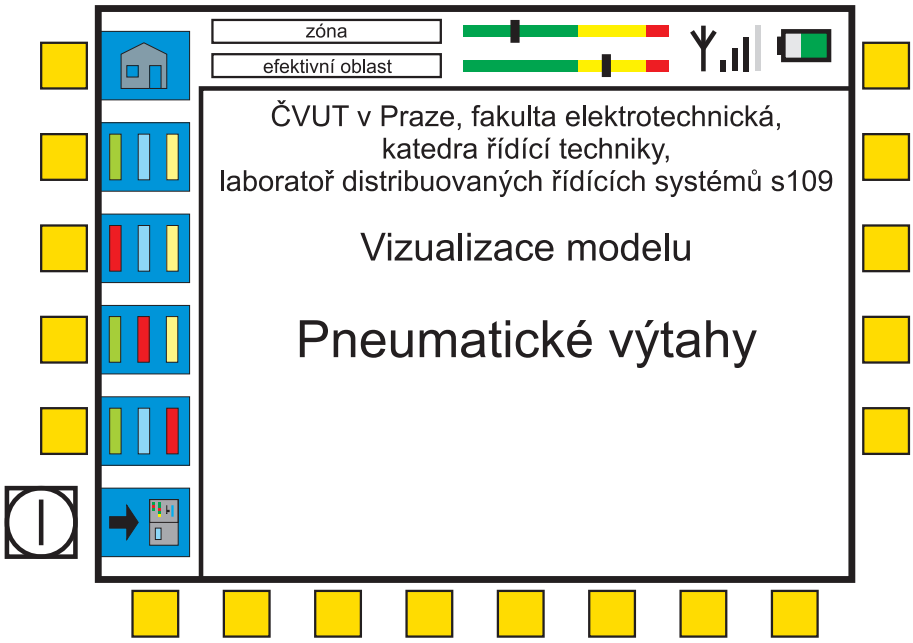
### 6.2.1 Šablona

Šablona i s tlačítky kolem obrazovky mobilního panelu je na obr. 6.2. Všech pět levých tlačítek a jedno spodní je využito k přepínání obrazovek. Obrazovky jsou naznačeny příslušnými schématickými obrázky u jednotlivých tlačítek. Pomocná tlačítka zároveň disponují LED diodou, která je použita k signalizaci aktivní obrazovky.

V horní liště šablony jsou umístěny prvky, které zobrazují jméno zóny a efektivní oblasti ve které se mobilní panel nachází. Jména zón se mění sama a lze jejich pomocí například přepínat obrazovky. Lišta efektivních oblastí zobrazuje jméno efektivní oblasti, v níž se mobilní panel nachází, ovšem pro přihlášení do efektivní oblasti je nutné kliknout na ikonku efektivní oblasti a přihlásit se k ní. Pro kontrolu je při přihlašovaní vyžadováno id efektivní oblasti (v našem případě máme v projektu jedinou efektivní blast s $id = 10$). Potom ukazatel efektivní oblasti zezelená a teprve v tuto chvíli jsou aktivní bezpečnostní prvky mobilního panelu. Pokud mobilní panel opustí přihlášenou efektivní oblast bez

odhlášení, je oprátor vyzván aby tak učinil. Pokud se do proběhnutí timeoutu neodhlásí, nastane příslušný bezpečnostní stav (Local nebo Global RampDown). Tyto stavy jsou popsány v části 2.2.4.2 a lze je detekovat v bezpečnostním programu pomocí výstupu bezpečnostního funkčního bloku FB162 příslušejícího přihlášené efektivní oblasti. Tento funkční blok je popsán v části 3.4.1 a je znázorněn na obr. 3.17, kde lze vidět výstupy *E_STOP, GLOB_RD, LOC_RD a SHUTDOWN* příslušející jednotlivým bezpečnostním stavům.

Posledními prvky na horní liště úplně v pravo jsou ukazatel síly signálu bezdrátového připojení mobilního panelu a ukazatel nabití baterie.



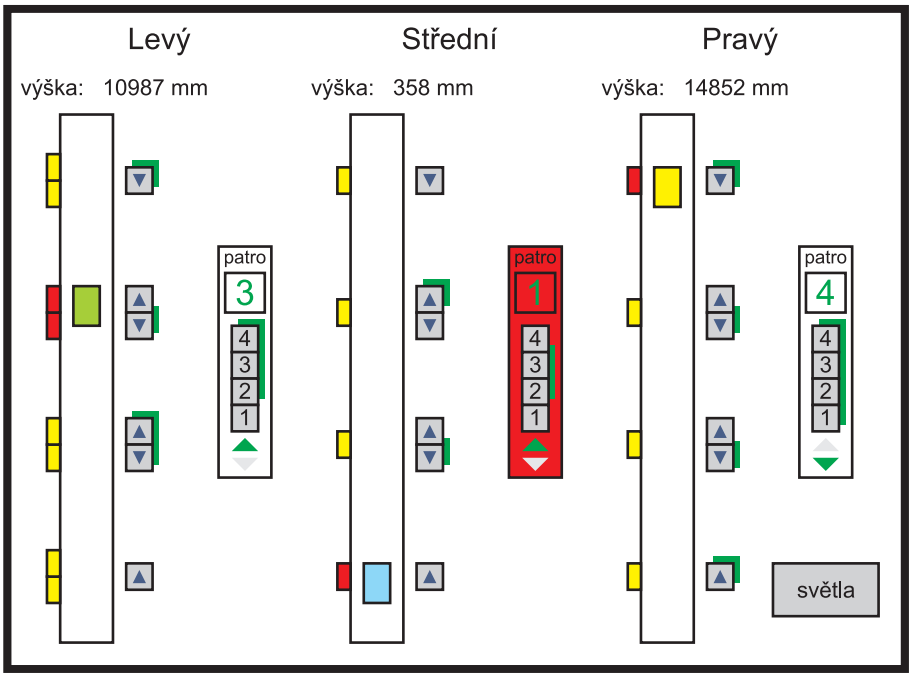

Obrázek 6.2: Šablona všech obrazovek

### 6.2.2 Úvodní obrazovka

Úvodní stránka není nijak zajímavá. Je na ní operátorovi sděleno že se nachází v laboratoři distribuovaných řídicích systémů katedry řídicí techniky na Karlově náměstí v Praze. Dále je operátor stručně seznámen s ovládáním vizualizace modelu pneumatických výtahů. K ničemu jinému tato uvítací obrazovka neslouží.

### 6.2.3 Ovládání výtahů

Toto je hlavní funkční obrazovka celé vizualizace modelu pneumatických výtahů. Obrazovka je znázorněna na obr. 6.3.




Obrázek 6.3: Obrazovka s ovládacími prvky výtahů

Jak je z předcházejícího obrázku patrné, na této obrazovce jsou znázorněny všechny tři výtahy i s pohybujícími se barevnými válečky představujícími kabiny výtahů. Lze z ní vyčíst i stavy jednotlivých optických senzorů a výšku od spodní hrany trubice ve které se kabina výtahu nachází. Vizualizace obsahuje funkční tlačítka umístěná v patrech se signalizací přijetí požadavku na zastavení. Vedle každého výtahu je umístěn panel tlačítek, který se nachází v kabině výtahu. Ten opět obsahuje funkční tlačítka se signalizací přijetí požadavku, dále také signalizaci patra ve kterém se výtah nachází a směr kterým se výtah pohybuje. Pokud je kabina výtahu zatížena, zbarvý se panel ovládacích prvků kabiny výtahu do červena.

Tato obrazovka poskytuje stejnou funkcionalitu jako tlačítka fyzicky umístěná na modelu. Navíc ještě umožňuje ovládat světla modelu.

### 6.2.4 Výpis podrobností konkrétního výtahu

Projekt obsahuje tři tyto obrazovky. Pro každý výtah jednu. Tyto obrazovky byly vytvořeny pro účely ladění PID regulátorů polohy válečku v trubici a k odečítání různých hodnot. Obsahují totiž výpis stavových proměnných výtahu, které lze interaktivně měnit. Jedná se o výšku kabiny výtahu, aktualní referenci regulátoru, akční zásah regulátoru, jednotlivé složky regulátoru a jejich vliv na akční zásah, omezení akčního zásahu, pracovní bod regulátoru atd.

Ačkoli byly tyto obrazovky vytvořeny pro účely ladění modelu, byly ve vizualizaci ponechány. Důvodem je dobrá demonstrace přímého přístupu do datových bloků s jednotlivými parametry výtahů a přímá možnost jejich změny.

### 6.2.5 Ovládání PLC

Poslední obrazovka obsahuje pouhá dvě tlačítka. Jedno se jménem *Lift Clear* a druhé se jménem *Lift Load*. Stisknutí jednoho z tlačítek pošle dotaz na IP adresu serveru LabLink, na určitý port. Pokud server detekuje tento dotaz, spustí příslušný skript *lift_clear.exe*, nebo *lift_load.exe*. Oba tyto skripty jsou popsány v kapitole 5.2.

Tato obrazovka tedy slouží k čištění programové paměti PLC a k nahrávání vzorové řídicí aplikace. Pokud se tedy někdy stane, že po startu mobilního panelu nebude vizualizace funkční, protože je v řídicím PLC modelu nahrán nějaký studentský program, je nutno přepnout na tuto obrazovku a stisknutím tlačítka *Lift Load* nahrát vzorovou řídicí aplikaci do PLC. Po dokončení nahrávání se PLC řídící model samo spustí a vizualizace bude plně funkční.

# Kapitola 7

# Závěr

Hlavním cílem této práce byla integrace modelu pneumatických výtahů do systému vzdálené laboratoře *LabLink*. Dalším cílem bylo rozšíření tohoto modelu o bezpečnostní prvky včetně instalace bezpečnostního PLC a mobilního bezpečnostního vizualizačního panelu.

V rámci seznámení s modelem byl vytvořen pospis modelu, kterému se věnuje první kapitola této práce. Jedinou podstatnou úpravou provedenou na modelu bylo přidání dvou ultrazvukových senzorů vzdálenosti *SICK UM30-214113*, posaných v části 2.1.2. Důvodem bylo, že předchozí způsob řízení výtahů nevybavených senzory vzdálenosti byl značně nepřesný a postrádal zpětnovazební informaci.

Druhá kapitola dokumentuje návrh vzorové řídící aplikace. Nejdříve byl popsaný fyzikální systém papírového válečku nadnášeného proudem vzduchu zjednodušeně odvozen pro lineární okolí pracovního bodu (část 3.1). Následně byl odhadem minimalizujícím střední kvadatickou chybu identifikován ARX model systému (část 3.1.4). Pro tento model byl navržen PID regulátor polohy válečku v trubici (část 3.2). Tento regulátor byl realizován v progromavacím jazyce Instruction List a je spouštěn s frekvencí $10\,Hz$. Tato frekvence byla zvolena jako optimální vzhledem k rychlosti fyzikálního dějě a vlastnostem použitých senzorů. Samotný řídicí program není nijak složitý. Dbali jsme spíše na přehlednost řešení. Pro každý výtah jsme vytvořili jeden datový blok, ve kterém jsou uloženy všechny jeho stavové informace. Řídicí algoritmus nejdříve čte data ze senzorů a zapisuje je právě do příslušných datových bloků jednotlivých výtahů. Toto se děje cyklicky, protože k jednotlivým vstup/výstupním panelům jednotlivých výtahů přistupujeme pomocí časového multiplexu popsaného v části 3.3.2.1. Potom je spuštěna logika výběru následujícího patra. Na základě jejího rozhodnutí je nastavena příslušná reference regulátoru polohy. Potom jsou zapsány všechny výstupy, které ovládají hlavně signalizační

LED diody modelu. Samotná generace PWM signálu probíhá nezávisle na hlavním programu. Je totiž spouštěna s periodou $1\,ms$ z bloku cyklyckého přerušení OB35. Důvodem je potřeba co nejmenší časové základny. Ukázalo se že nám nejmenší dostupná časová základna $1\,ms$ pro naši aplikaci postačuje.

Ve třetí kapitole je popsáno samotné síťové řešení problému pomocí průmyslové komunikační sítě založené na protokolu *Profinet IO*. Celé řešení je schématicky znázorněno na obr. 4.1. Všechny použité stanice jsou vzájemě propojeny do jedné podsítě. Tato podsíť je fyzicky spojena se síťovou kartou serveru LabLink, ze kterého je možné přistupovat k jednotlivým zařízením. Navržené řešení obsahuje dvě řídicí PLC. Jedno řídí samotný model pneumatických výtahů a je programováno studenty a druhé, bezpečnostní obsluhuje všechny bezpečnostní prvky systému. Ačkoli je toto PLC ve stejné podstíti a studenti se s ním mohou spojit, nemohou změnit jeho program, neboť toto PLC je zaheslováno. Všechna jednotlivá komunikační spojení jsou podrobně popsána v části 4.1.

Čtvrtá kapitola se zabývá programovým ovládáním PLC řídícího samotný model pneumatických výtahů. PLC je nutné mazat před započetím a po ukončení každé studentské rezervace. Zároveň je také potřeba mít možnost z vizualizačního mobilního panelu nahrát do PLC vzorovou řídicí aplikaci. Pro řešení byla zvolena knihovna *Command Interface*, která umožňuje přistupovat z vyššího programovacího jazyka k jednotlivým funkcím programu *Step7*, ve kterém se standardně programují PLC řady SIMATIC. Ve Visual Sudiu 2010 byly v jazyce C# napsány dva spustitelné skripty. Jeden provede vymazání PLC a nahrání prázdné HW konfigurace a druhý provede nahrání vzorové řídicí aplikace. Tyto skripty jsou psány tak aby v každém případě byly řádně dokončeny a nezpůsobily pád systému LabLink. Proto byly všechny kritické operace ošetřeny vyjímkami a na zamrznutí aplikace dohlíží WatchDog, který po určitém časovém limitu sám aplikaci ukončí.

V páte kapitole je popsána tvorba vizualizace řídicího procesu modelu pneumatických výtahů. Na jejím začátku jsou definovány zóny a efektivní oblasti, které jsou důležité nejen pro vizualizaci, ale i pro bezpečnostní řídicí program popsaný v části 3.4. Dále jsou defnována spojení s jednotlivými PLC a nakonec už přímo vizualizační tagy. Jejich pomocí byla vytvořena jednoduchá funkční vizualizace modelu. Vrcholem funkcionality mobilního panelu je možnost spouštět přímo z vizualizace zde již popsané servisní skripty pro vymazání paměti PLC a nahrání vzorové řídicí aplikace.

Navržené řešení integrace modelu do systému LabLink bylo již realizováno a spuštěno. Po spuštění rezervačního systému se vyskytla řada problémů, jež museli být ošetřeny přímo ve skriptech spouštěných servisní plochou serveru systému LabLink. Výsledná dosažená podoba skriptů je po cca dvouměsíčním provozu a ladění systému použitelná.

Skripty fungují uspokojivě. Chyby prováděných skriptů sice nebyly eliminovány zcela, ale vyskytují se v intervalu několika týdnů, což považujeme za uspokojivé řešení. Vznikající chyby mají náhodný charakter a odstraní se fyzickým restartováním PLC (odpojení od napájení). Domníváme se že tyto chyby ani naše skripty nemohou podchytit. Použitá průmyslová technologie totiž nebyla primárně vytvořena k takovémuto použití.

PLC řady SIMATIC S300 jsou vyráběna pro použití v průmyslu. Tam je toto PLC nainstalováno, je do něj nahrán program a ten je následujících několik let beze změny prováděn. Při nejvyšším vytížení je pamět našeho PLC za den přehrána třeba i dvacetkrát až třicetkrát za den a v podobném intervalu je PLC stále uváděno do STOP a RUN stavů. Nahrávané projekty nejsou vždy funkční, proto nastávají chybové stavy PLC, studenti využívají diagnostická spojení, atd. Zároveň vyžadujeme nezávislý administrátorský přístup k PLC za každé situace, i když na PLC studenti třeba právě pracují. Takovéto využití se v praxi nevyskytuje. Proto se nelze divit, že je nutné PLC jednou za cca dva týdny restartovat odpojením od zdroje. Z výše popsaných důvodů se domníváme, že jsme se přiblížili maximální možné spolehlivosti navrženého řešení.

Pokud bychom chtěli dosáhnout vyšší spolehlivosti, museli bychom řídicí PLC programově restartovat odpojením od napajení při každém provední skriptu pro mazání paměti PLC. To v podstatě ani není složité. Řídicí PLC by mohlo od napájení odpínat například bezpečnostní PLC pomocí relé. Musela by se ovšem vyřešit komunikace mezi aplikací běžící na servisní ploše systému LabLink a bezpečnostním programem bezpečnostního PLC, který by ovládal napájecí relé. Toto samozřejmě není neřešitelný problém, ale není předmětem této práce.

Závěrem ještě podotkneme, že připojení modelu pneumatických výtahů už bylo úspěšně rozšířeno na další model v laboratoři. Jedná se o model přečerpávací elektrárny. Tento model je řízen stejným PLC, takže potřebné úpravy byly minimální a spolehlivost řešení je identická.

# Literatura


Havlena, V.; Štecha, J. (2000), *Moderní teorie řízení*, Praha: Vydavatelství ČVUT.

Roubal, J., Pekař, J., Pachner, D. a Havlena, V.; (2005), *Moderní teorie řízení – Cvičení*, Praha: Vydavatelství ČVUT.

Henych, T.(2004), *Řízení a vizualizace technologického procesu pomocí PLC*

IEC 1131(1993), *IEC 1131 – 3 Programmable Controllers – part 3, Programming Languages.*

SICK(2010), *Ultrasonic sensors, UM30, UM30-214113*, www.mysick.com

Profinet(2010), *PROFINET System*, PI www.profibus.com

PROFIsafe(2010), *PROFIsafe system description*, PI www.profibus.com

MP277F manual(2008), *HMI Mobile Panel 277f function manual*, Siemens www.support.automation.siemens.com

Command Interface Help(2008), *SIMATIC Command Interface Help*, Siemens www.support.automation.siemens.com

# Příloha A

# Obsah přiloženého CD

- Tato práce ve formátu .pdf
- Zdrojové kódy této práce v programu LaTeX
- Projekty vytvořených skriptů ve Visual Studiu 2010
- Projekt vzorové řídicí apliakce modelu v programu STEP7
- Odkazované manuály
- Dokumentace knihovny Command Interface
- Knihovna bezpečnostních bloků mobilního panelu