České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Pavel Šimek**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Simulace výrobních linek**

Pokyny pro vypracování:

1. Zprovozněte virtuální svařovací linku včetně reálného robota.
2. Zprovozněte bezpečnostní systém s využitím laserového scanneru, optických závor a funkce SafeOperation v řídicím systému robota.
3. Virtuální výrobní linku zprovozněte se simulačním systémem SIMIT.
4. Rozdělte linku na dílčí úlohy, které budou využity ve výuce řídicích systémů. Ověřte možnosti virtuálního zprovoznění a připravte vše pro systém LabLink. Pro každou z dílčích úlohy vytvořte odpovídající dokumentaci.

Seznam odborné literatury:

Process Simulate Reference Manual. Siemens Industry Software. 2013.
Process Designer Reference Manual. Siemens Industry Software. 2013.

Vedoucí: Ing. Pavel Burget, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

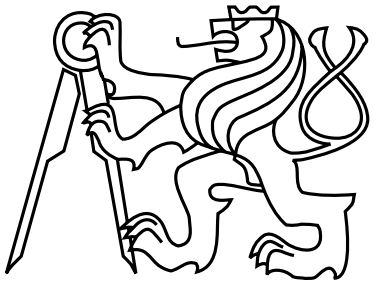
L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 12. 2. 2014

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra řídicí techniky



Diplomová práce

Simulace výrobních linek

Pavel Šimek

Vedoucí práce: Ing. Pavel Burget, Ph.D.

Studijní program: Kybernetika a robotika, navazující magisterský

Obor: Systémy a řízení

4. ledna 2015

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce panu Ing. Pavlu Burgetovi, Ph.D. za jeho cenné rady a odborné vedení. Dále bych rád poděkoval panu Jiřímu Kopencovi ze společnosti Siemens za jeho vstřícnost a ochotu pomoci mi s řešením problémů. Nakonec bych rád poděkoval přítelkyni, rodičům a všem, kteří mě během studia podporovali. Děkuji Vám všem.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Psárech dne 2. 1. 2014 .....

Abstract

This thesis deals with simulation of production processes. For this purposes manufacturing tool Tecnomatix from Siemens Industry Software is used. Tecnomatix contains several tools for planning and validation of production processes. This work is focused on Process Designer and more so on Process Simulate.

The first part of this work is about implementation and commissioning of virtual welding/painting line and commissioning of several industrial devices which are available in the laboratory (industrial robot KUKA, safety laser scanner SICK, safety light curtain SICK and HMI panel Siemens).

The second part is about using Process Simulate possibilities to extend the control systems education. The extension will lie in creation of several models in Process Simulate environment, which will be controlled through appropriate interface from PLC, or PLCSim (PLC simulator).

Abstrakt

Tato práce se zabývá simulacemi výrobních linek s využitím softwarových nástrojů Tecnomatix od společnosti Siemens Industry Software. Tecnomatix obsahuje několik nástrojů pro plánování a ověřování výrobních procesů. V této práci se zaměřuji na Process Designer a zejména pak na Process Simulate.

První část práce se věnuje tvorbě a zprovoznění virtuální svařovací/lakovací výrobní linky a zprovoznění několika průmyslových zařízení dostupných v laboratoři (průmyslový robot KUKA, bezpečnostní laserový skener SICK, bezpečnostní optický závěs SICK a vizualizační panel Siemens).

Druhá část práce se poté věnuje možnostem využití simulačního prostředí Process Simulate pro rozšíření a zatraktivnění výuky řídicích systémů. Rozšíření výuky bude spočívat ve využití simulačních modelů vytvořených v Process Simulate, které budou studenti přes vhodné rozhraní řídit z PLC, případně z PLCSim (PLC simulátor).

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

S rostoucími požadavky na efektivitu výroby, její bezporuchovost a co možná nejrychlejší uvedení do provozu se důležitou součástí při návrhu výrobních linek (a potažmo výrobních závodů celkově) stávají digitální simulace.

Digitální simulace umožňují celkové ověření konceptu výroby ještě před samotnou stavbou reálné linky. Mezi hlavní výhody (přínosy) digitálních simulací patří zejména: možnosti včasného odhalení chyb již ve fázi návrhu, optimalizace výroby (optimalizace sledu operací, layoutu linky, vytížení jednotlivých robotů a pracovišť atp.), možnost provádět analýzy proveditelnosti, vyšetření ergonomie manuálních operací atp. Pro tyto účely může být (jak je tomu například v této práci) použit softwarový balík Tecnomatix od společnosti Siemens, jehož součásti (Plant Simulation a Process Simulate) mohou být použity pro simulace na různých úrovních výroby.

1.1 Cíle, motivace

Má práce se zabývá simulací výrobních linek v prostředí Process Simulate. Prvním cílem práce je návrh, implementace a následné virtuální zprovoznění (tzv. virtual commissioning) virtuální svařovací/lakovací (dále jen svařovací) linky, jejíž podoba a činnost bude popsána v následující kapitole. To zahrnuje zejména: implementaci linky v simulačním prostředí Process Simulate, návrh a implementaci řízení této linky, prozkoumání a otestování možností propojení mezi řídicím systémem a simulací. Posledním krokem je poté zakomponování reálných zařízení, které jsou dostupné v laboratoři (průmyslový manipulátor, bezpečnostní laserový skener, bezpečnostní optický závěs a vizualizační panel), aby bylo možné průmyslový manipulátor používat v plně automatickém provozu při dodržení všech zásad bezpečného provozu.

Druhým cílem práce a zároveň motivací pro její sepsání je možnost rozšíření a zatraktivnění výuky řídicích systémů. Pro tyto účely bude vytvořeno několik menších úloh, které budou studenti v rámci příslušného předmětu řešit. Hlavní výhodou využití digitálních simulací ve výuce je, že studenti budou řídit interaktivní simulační modely, čímž se každá z úloh stane názornější a snad i zajímavější než při „pouhém“ testování funkčnosti programů s využitím PLCSim.

1.2 Struktura práce

Práce je rozdělena do 5 kapitol. Úvodní kapitola vymezuje cíle práce a popisuje její hlavní výstupy. Ve druhé kapitole je nejprve uveden stručný popis softwarových nástrojů Tecnomatix s důrazem na simulační prostředí Process Simulate, následuje specifikace svařovací linky. Hlavní část kapitoly se věnuje implementaci virtuální svařovací linky v Process Simulate a následně virtuálnímu zprovoznění linky s využitím reálného PLC. Následující kapitola se zabývá nejprve popisem a následně zprovozněním několika průmyslových komponent, které se nacházejí v laboratoři KN:E-s109 („strojovna“) na Karlově náměstí - jedná se konkrétně o průmyslový robot (manipulátor) KUKA Krc5 Arc, bezpečnostní laserový skener SICK S3000, bezpečnostní optický závěs SICK C4000 a vizualizační panel Siemens TP 700 Comfort. Tato kapitola rovněž obsahuje popis pracoviště v laboratoři. Čtvrtá kapitola se zabývá rozšířením výuky řídicích systémů. Poslední kapitola pak shrnuje výsledky dosažené v rámci mé práce, uvádí možnosti rozšíření a vylepšení.

1.3 Výstupy práce

Mezi hlavní úkoly/výstupy práce patří:

- implementace a následné virtuální zprovoznění svařovací linky v simulačním prostředí Process Simulate,
- zprovoznění a integrace laserového skeneru, optického závěsu, průmyslového robotu a vizualizačního panelu,
- návrh a tvorba úloh pro výuku řídicích systémů,
- vytvoření dokumentace k jednotlivým podúlohám.

Kapitola 2

Simulace a řízení výrobní linky

Tato kapitola se věnuje simulaci svařovací výrobní linky. Kapitolu je možno rozdělit do 4 menších částí. První část kapitoly stručně popisuje softwarový balík Tecnomatix s důrazem na simulační prostředí Process Simulate. Ve druhé části je uveden popis a základní možnosti použití simulačního prostředí SIMIT. Ve třetí části je na základě specifikace výrobní linky proveden návrh jejího layoutu a sekvence operací. Poslední část se zabývá implementací linky v simulačním prostředí Process Simulate, propojením řídicího systému se simulací a virtuálním zprovozněním (řízení z PLC).

2.1 Digitální továrna Tecnomatix

Tecnomatix je komplexní sada softwarových nástrojů pro plánování a ověřování výrobních procesů od společnosti Simemens Industry Software. Jedná se o softwarovou platformu založenou na 3 vrstvé architektuře s možností vzájemné spolupráce více uživatelů. Na vrcholu hierarchie stojí Oracle Database Server, na nejnižší vrstvě jsou takzvaní eMS klienti (např. Process Designer nebo Process Simulate), kteří se přes mezivrstvu - eMServer připojují k Oracle databázi. Data jsou ukládána na 2 místa - do Oracle databáze a na SystemRoot - obvykle se jedná o sdílený disk, na který se ukládají všechny externí objekty napojené na prvky (objekty) v projektu (například 2D, 3D modely apod[1]).

Mezi hlavní důvody pro zavádění digitálních simulací výrobních procesů patří snížení časové náročnosti uvádění do provozu a možnost odhalení chyb již v době návrhu, což má za důsledek významné šetření nákladů a času - je zřejmé, že pozdní odhalení chyb může vést k časově náročným a nákladným změnám. Simulace nám tedy umožňuje celkové ověření konceptu již ve fázi plánování ještě před stavbou reálného zařízení (linky).

2.1.1 Tecnomatix nástroje

Pro „hrubé“ plánování na vyšší úrovni slouží **Process Designer**. Jedná se o nástroj, který je určen zejména pro plánovače procesů, technology, návrháře hal a návrháře montážních linek. V prostředí Process Designer se vytvoří jakýsi celkový koncept (kostra) výrobního procesu bez konkrétní implementace jednotlivých částí (například operací). Plánovači tedy navrhují

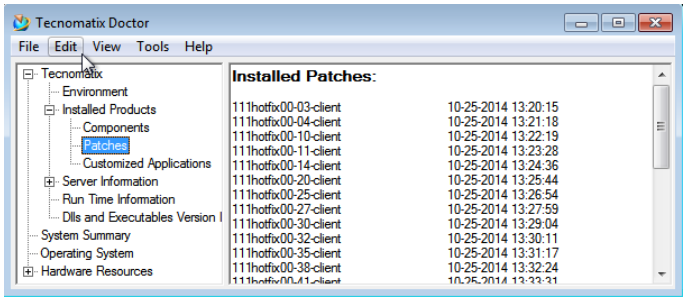
[1] v literatuře někdy označována jako „non-database files“

layout, sekvence montáže, jednotlivé procesy a definují jim přidělené časy, definují potřebné počty výrobních prostředků atp. Pro definici návazností a vztahů jednotlivých operací se používají Gant diagramy a PERT.

Process Simulate je další Tecnomatix nástroj. Jedná se o simulační prostředí se širokými možnostmi, poskytující funkčnosti od analýzy proveditelnosti manuálních operací a vyšetření ergonomie (PS Human/Jack), přes offline vytváření robotických programů (PS Robotics), až po možnost otestování PLC řízení (režim virtuálního zprovoznění, virtual commissioning). Používá se na detailní plánování, implemetaci jednotlivých operací, optimalizace a simulace (kontrola dosahů, kolizí, synchronizace robotů atp.). Process Designer a Process Simulate jsou navzájem úzce propojeny.

Dalšími důležitými součástmi Tecnomatix jsou zejména: **Plant Simulation**[2] - nástroj pro analýzu na úrovni továrny (zkoumání propustnosti systému, toku materiálu atp.), **AdminConsole** - nástroj, jehož prostřednictvím lze vykonávat administrátorské zásahy (mazání projektu, přidělování práv uživatelům atp.), **Tecnomatix Doctor** - aplikace, která umožňuje spravovat jednotlivé součásti (CAD prohlížečky, kontroléry pro roboty atp.) a nainstalované HotFixy.

Pozn: Veškeré HotFixy jsou dostupné ze stránek GTAC[3] v sekci *Download&Upload Files/ Download Files/* podsekce *Product Updates/Tecnomatix/Planning_ Applications/Hot_ Fixes / V11.1.X*. Dostupné HotFixy je nutno pravidelně instalovat nejen pro klienty (soubory ...client.zip pro 32bitové klienty, případně ...client_x64.zip pro 64bitové klienty), ale také pro server (...server.zip, případně ...server_x64.zip). Archivy obsahují běžné *.msi instalátory.



Obrázek 2.1: Tecnomatix Doctor - seznam nainstalovaných HotFixů

2.1.2 Vzájemná spolupráce více uživatelů

Funkčnost vzájemné spolupráce je založena na mechanismu *Check-in* a *Check-out*. Více uživatelů může pracovat zároveň na stejném projektu, ale ne na jeho stejné části. Pomocí příkazu Check-out si uživatel rezervuje určitou část projektu, na které chce pracovat a na které bude provádět změny. Je-li určitá část projektu zarezervována některým uživatelem (některý z uživatelů si ji pro sebe rezervoval pomocí příkazu Check-out), nemohou ostatní uživatelé tuto oblast nijak modifikovat (je jim umožněno pouze čtení). Po dokončení úprav uživatel příkazem Check-in zajistí to, že se změny trvale uloží do databáze a budou tak viditelné pro

[2] Plant Simulation se v rámci své DP věnuje T. Urban, viz [20]

[3] Global Technical Access Center, dostupné na http://support.industrysoftware.automation.siemens.com/gtac.shtml

všechny spolupracovníky. Doporučuje se pravidelné ukládání práce pomocí Check-in. Dialog umožňuje uložení do databáze a zároveň opětovné zarezervování (možnost *Keep objects checked out*).

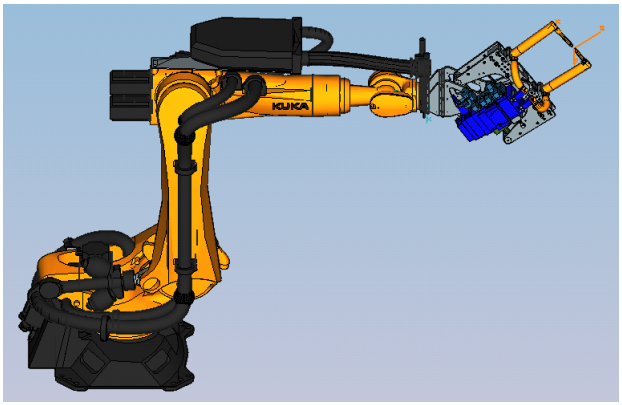
Je důležité si uvědomit, že při modifikaci ve stromovém prohlížeči (například Navigation Tree v Process Simulate) nebo při změnách prováděných v dialogu *Properties* (například změna jména PrStation v Process Simulate) není k dispozici funkčnost Zpět a Vpřed, jelikož se pracuje přímo s databází, tzn. všechny modifikace se ukládají ihned do databáze. Jedinou možností, jak se vrátit zpět (například pokud uživatel neúmyslně smaže nějaký objekt) je použít funkci *Cancel Check Out* - ta nám umožňuje vrátit se do stavu, kdy byl naposledy proveden Check-in. Kontrola data a času posledního Check-in je možná v dialogu Properties (položky *Last modified By* a *Date*).

Mechanismus vzájemné spolupráce je jednoduchý a efektivní, z mého pohledu jediným nedostatkem je nemožnost verzování. Při ukládání dat do databáze pomocí příkazu Check-in sice existuje možnost *Check-in as a new version*, nicméně využití této funkcionality pro verzování projektu není doporučováno. Pravidelné zálohy projektu lze vytvořit pomocí exportu eBOP (*Project Management / Export selected eBOP to file*) a zkopírování obsahu SystemRoot (eBOP spolu s obsahem SystemRootu obsahuje vše potřebné pro opětovný import projektu). Následný import projektu se provádí pomocí *Import new eBOP Project*.

Pozn.: U jednotlivých částí projektu je stav zarezervování přehledně graficky odlišen: zelená fajfka = část projektu, kterou mám zarezervovanou na sebe, červený křížek = část projektu, kterou má zarezervovanou jiný uživatel.

2.1.3 3D modely

V Process Designer i Process Simulate se pracuje s 3D modely ve formátu *.jt, které jsou uloženy v adresáři *.cojt v knihovně na SystemRoot. Pro prohlížení modelů můžeme použít například freewarovou JT2go prohlížečku[4].



Obrázek 2.2: 3D model KUKA robotu

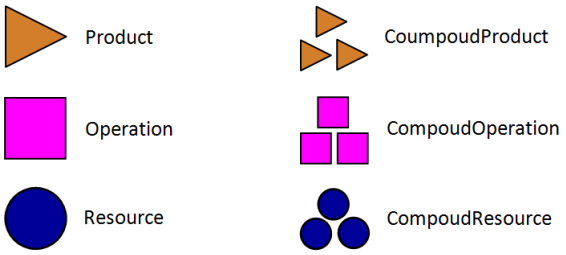
[4]po registraci dostupná z http://www.plm.automation.siemens.com/en_us/products/teamcenter/lifecycle-visualization/jt2go/

2.1.4 Process Simulate

Jak již bylo řečeno, jedná se o simulační prostředí, které obsahuje mnoho nástrojů, s jejichž pomocí můžeme vytvářet realistické simulace výrobních procesů. Mezi některé z možností patří například možnost simulace pohybu robotů, generování robotických programů, simulace a analýza pohybů člověka nebo virtuální zprovoznění linky - tzv. virtual commissioning, při kterém je linka řízena z PLC.

Každý výrobní proces je popsán propojením objektů základních typů (operace, výrobní prostředky a díly) a definicí jejich vzájemných vztahů. Další důležitou roli v simulaci hrají signály (například signál z fotoelektrického senzoru, robotické signály atp.). Kompletní popis výrobního procesu je reprezentován tzv. eBOP (Electronic Bill of Process). Ten je tvořen objekty následujících typů:

- produkt (Product) - objekt, který vznikne výrobním procesem popsaným v eBOP,
- operace (Operation) - posloupnost kroků, které je nutné provést k vyrobení produktu,
- výrobní prostředek (Resource) - jedná se o objekty, které jsou nutné k vyrobení produktu (pracovníci, nářadí, roboty atp.),
- technologické body (Manufacturing Features, MFG) - definují vztahy mezi různými díly - například svařovací body.



Obrázek 2.3: Základní prvky výrobního procesu

Process Simulate je vybaven komplexním, objektově orientovaným (jednotlivé možnosti - položky menu pravého tlačítka, tlačítka na liště nástrojů atp. se dynamicky mění / zpřístupňují dle právě vybraného objektu) uživatelským rozhraním. Mezi jeho výhody patří snadná možnost úpravy (přízpůsobení, customizace) prostřednictvím definice různých layoutů (defaultně je definováno 8 layoutů). Jednotlivé layouty se poté s výhodou používají pro určité specifické činnosti - například layout pro editace robotických cest (Robot Path Editing) obsahuje prvky používané při editaci a vytváření robotických cest.

2.2 Simulační prostředí SIMIT SCE 7

SIMIT SCE je simulační nástroj, který umožňuje vytvoření modelů reálných zařízení, které mohou být následně použity pro ověření (otestování) PLC programu. Hlavní výhodou verze

SCE (Siemens Automation Cooperates with Education) je to, že umožňuje uložení (vyexportování) vytvořeného modelu ve formě spustitelné simulace (vygenerované modely mají koncovku .simit4S), která může být jednoduše předána studentům pro testovací účely. Podmínkou pro spuštění a použití vygenerované simulace je to, že na studentském PC musí být nainstalován SIMIT4Students (Simit Runtime). Určitou nevýhodou je poté omezení maximálního počtu vstupů a výstupů a omezení maximálního množství prvků použitých pro vytvoření simulace - konkrétně je možné použít 32 digitálních vstupů, 32 digitálních výstupů, 8 analogových vstupů, 8 analogových výstupů a maximálně 250 prvků. Proces vytvoření a použití simulačního modelu je možno rozdělit do dvou kroků:

- Prvním krokem je vytvoření modelu řízeného systému (například výtah, nebo soustruh - jedná se o ukázkové modely, které jsou vytvářeny a poměrně detailně popsány v manuálech dodávaných na instalačním CD). Model je vytvářen s použitím standardní knihovny, která obsahuje bloky (aritmetické a logické operace, prvky pro simulaci pohonů atp.) s jejichž využitím definujeme chování (dynamiku) modelovaného systému. Tyto prvky jsou propojeny a umístěny na tzv. diagramy (Diagrams) takovým způsobem, aby bylo simulováno chování daného systému. Prostředí rovněž umožnuje vytvoření jednoduché vizualizace. Vizualizační obrazovky se vytváří (stejně jako logická schémata pro definování funkčnosti) na diagramech, přičemž máme k dispozici základní nastavovací a zobrazovací prvky jako tlačítka, přepínače, analogové posuvníky a zobrazovací jednotky. K dispozici je rovněž možnost kreslení základních grafických tvarů jako čar, obdélníků atp., kterým můžeme navíc definovat animaci (rotace, posuny, zvětšení/zmenšení atp.).
- Dalším krokem je využití vytvořeného modelu k otestování řídicího (PLC) programu. Zde je nutno nejprve zvolit a nastavit vhodné rozhraní (ve verzi SIMIT SCE je dostupné rozhraní PLCSim a PRODAVE) pro propojení PLC či PLCSim se simulací.

Díky dynamickému modelu řízeného systému a možnosti vytváření vizualizačních obrazovek se s použitím SIMIT stává proces vytváření a testování řídicího programu zajímavějším a realističtějším než (například) při „pouhém“ testování s použitím PLCSim a sledováním/nastavováním příslušných vstupů a výstupů.

Naše prvotní představa o využití SIMIT byla taková, že existuje možnost vyexportování modelů z Process Simulate a jejich následné použití v SIMIT. Později se ukázalo, že tato možnost exportu neexistuje. Další možností využití bylo použít SIMIT jako rozhraní mezi řídicím systémem (PLC) a simulací v Process Simulate. Hlavní výhodou použití tohoto rozhraní je (dle manuálu [1]) zejména lepší časování a synchronizace mezi Process Simulate a SIMIT. Pro naše účely však tato možnost použití nebyla příliš zajímavá (nutnost přítomnosti HW klíčenky, omezení maximálního počtu signálů) a proto jsme jako rozhraní raději zvolili OPC server. Poslední možností bylo vytvoření několika jednoduchých modelů a jejich následné použití ve výuce řídicích systémů. Vzhledem k problémům s funkčností, na které jsem narazil, však ani toto použití nebylo možné.

Vzhledem k problémům s funkčností SIMIT, které byly (dle odpovědi technické podpory) nejspíše způsobeny českou diakritikou v hardwarové klíčence, následné nutnosti její výměny (klíčenka musela být posílána do Německa) a zejména pak nepříliš velkému přínosu použití SIMIT jako rozhraní, nebyly možnosti využití SIMIT dále studovány.

2.3 Specifikace řízené linky

Základní specifikace výrobní linky, která byla několikrát konzultována s vedoucím práce a následně upravována dospěla do následující finální podoby:

Výrobní linka bude mít část svařovací a část lakovací.

*Hotový výrobek se skládá ze 2 dílů v konfiguraci - **part1A** + **part2** nebo **part1B** + **part2**. Na začátku jsou 3 palety s jednotlivými typy dílů (**part1A**, **part1B** a **part2**), ze kterých pracovník bere díly a vkládá je do přípravků na otočném stole. Jednotlivé typy dílů jsou na otočném stole identifikovány pomocí senzorů.*

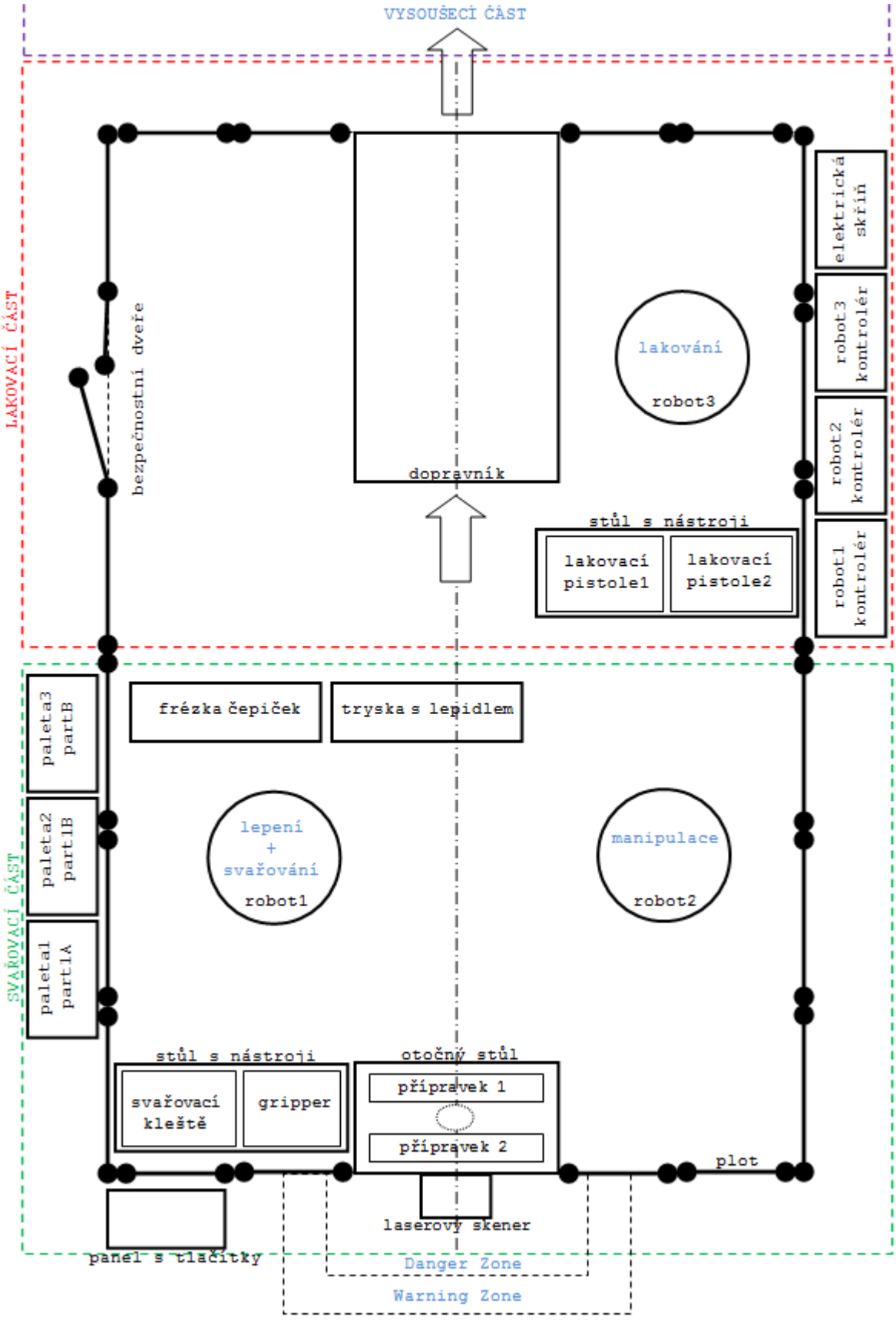
*Na začátku je otočný stůl v pozici HOME. Pracovník vezme první díl (**part1A** nebo **part1B**) a vloží ho do přípravku na otočném stole, následně opustí pracovní prostor hlídaný laserovým skenerem a stiskne tlačítko pro otočení stolu. Nyní vezme druhý kus výrobku (**part2**) a opět ho vloží do přípravku na otočném stole, který se nyní nachází v pozici FWD. Následně opět opustí pracovní prostor hlídaný laserovým skennerem a stiskne tlačítko pro spuštění práce.*

*Nejprve dojde k otočení stolu do pozice HOME, to umožní robotu č.1 uchopit druhý kus dílu (**part2**). Po uchopení dílu začnou paralelně probíhat dvě operace - otočení stolu do pozice FWD a nanášení lepidla na druhý kus výrobku (**part2**). Nanášení lepidla je prováděno stacionární tryskou s lepidlem, kolem které robot č.1 držící díl projede tak, aby bylo lepidlo rovnoměrně naneseno na příložné ploše dílu. Po nanesení lepidla robot č.1 přiloží druhý kus dílu (**part2**) na první kus dílu (**part1A** nebo **part1B**) a jde vyměnit nástroj (gripper vymění za svařovací kleště). Po výměně nástroje robot č.1 svaří oba díly k sobě, čímž vznikne výrobek. Po dokončení svařování následuje výměna nástroje u robotu č.1 a zároveň robot č.2 uchopí výrobek a umístí ho na dopravník, který díly veze do lakovací části linky. Celá tato sekvence se neustále opakuje.*

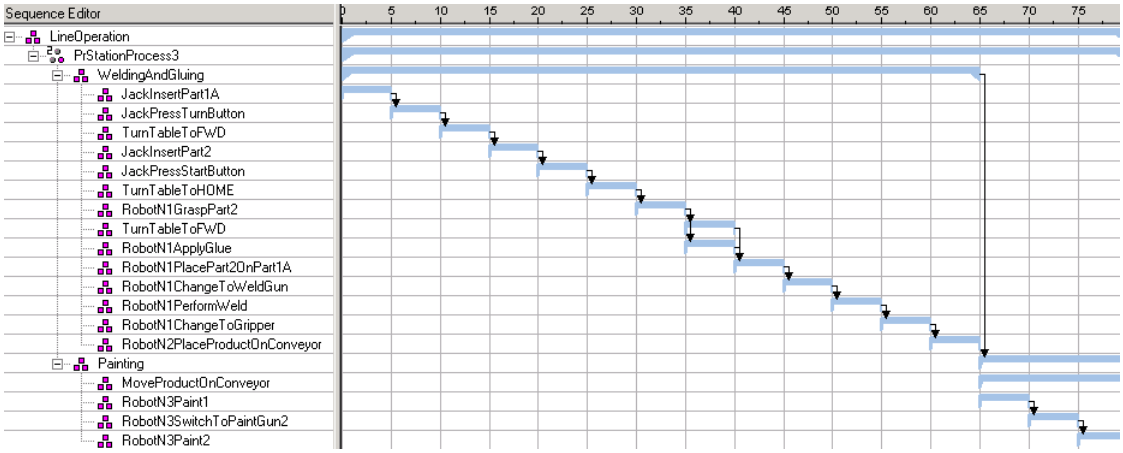
Díly se pomalu pohybují na dopravníku. Robot č.2 se pohybuje podél dílu a během pohybu lakuje první část výrobku. Potom vymění lakovací pistoli a nalakuje i druhou část dílu. Způsob lakování odpovídá typu dílu.

2.3.1 Layout a činnost linky

Na základě této specifikace byl vytvořen návrh layoutu linky. Jedná se pouze o hrubý návrh, který dává základní představu o rozmístění jednotlivých zařízení bez větších ohledů na jejich rozměry, dosahy robotů, proveditelnost operací atp. Tento layout poslouží jako základní vodítko pro pozdější rozmísťování prvků. Navíc svou grafickou reprezentací (viz Obr. 2.4) společně s přiloženým gantovým diagramem (viz Obr. 2.5), který naznačuje základní sled operací nutných pro vyrobení výrobku, který se skládá z part1A a part2 (sled operací nutných pro vyrobení výrobku, který se skládá z part1B a part2 by byl obdobný), částečně usnadňuje pochopení činnosti linky, která byla slovně specifikována výše.



Obrázek 2.4: Návrh layoutu výrobní linky



Obrázek 2.5: Sled operací linky

2.4 Implementace linky

Na základě specifikace linky, hrubého nástinu layoutu a její funkčnosti jsem začal s její celkovou implementací. Implementace se dá rozdělit do 2 hlavních částí. První částí je vytvoření modelu linky s využitím Tecnomatix nástrojů, druhou je implementace řídicího programu v PLC. Nedílnou součástí je také volba a následné zprovoznění rozhraní mezi PLC a simulací.

2.4.1 Vytvoření digitálního modelu

Implementaci linky jsem prováděl zhruba v následujících krocích:

1. **shromáždění potřebných modelů**

   Zde je příhodné zmínit to, že Tecnomatix neobsahuje žádnou knihovnu modelů. Nicméně například modely robotů některých výrobců je možné stáhnout ze stránek daného výrobce, nebo je možno využít modely dostupné ze stránek technické podpory GTAC.

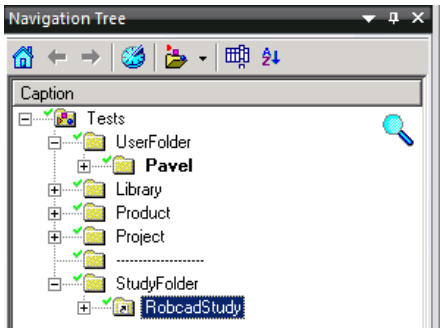
   Další možností je využít funkčnosti modelování objektů přímo v Process Simulate. To můžeme provést například pomocí *Modeling / Create Part/Resource / New Resource*, čímž dojde k otevření dialogu, ve kterém vybereme požadovaný typ modelovaného objektu (například PartPrototype). Tím dojde k vytvoření nového objektu, který můžeme vidět v *Object Tree / Resources*. To, že je tento objekt otevřen pro modelování, je naznačeno červeným M (Modeling) vykresleným v jeho ikoně. Pro účely modelování je vhodné pomocí Layout manageru přepnout na příslušný layout - Modeling, ve kterém jsou zpřístupněny položky používané při modelování (jako například Create Box, Substract Solids atp.). Po vytvoření požadovaného modelu se v Object Tree tento objekt označí a pomocí *Modeling / End modeling* ukončíme modelování. Následně budeme prostřednictvím dialogu vyzváni k uložení vytvořeného modelu. Užitečná je rovněž možnost vrácení změn k poslednímu uloženému stavu (obvykle použijeme v situaci, kdy upravujeme model nějakého objektu) pomocí *Modeling / Reload Component*.

   Po shromáždění (případně vytvoření) potřebných modelů vytvoříme na disku následující složkovou strukturu:

- PavelSimekDP - nejvýše umístěnou složku pojmenujeme stejně jako projekt (v mém případě PavelSimekDP)
- PavelSimekDP/BackUps - složka pro vytváření záloh
- PavelSimekDP/Root - emServerRoot, jedná se o složku, na kterou bude nastaven systém root v Process Designer a Process Simulate
- PavelSimekDP/Root/LIBRARIES - knihovny
- PavelSimekDP/Root/LIBRARIES/HUMAN_MODELS - obsahuje model člověka
- PavelSimekDP/Root/LIBRARIES/PARTS - modely dílů
- PavelSimekDP/Root/LIBRARIES/RESOURCES - obsahuje modely výrobních prostředků, které dále rozdělíme do podsložek (například GUNS, ROBOTS, FIXURES atp.).

2. **založení nového projektu a vytvoření jeho základní struktury v Process Designer**

   Tento krok je možno provádět buď v Process Designer, nebo přímo v Process Simulate. Vytvoření nového projektu v prostředí Process Designer probíhá klasicky přes *File / New Project*. Pro lepší strukturalizaci je projekt rozdělen do následující struktury (objekty požadovaného typu můžeme přidat například pokud klikneme pravým tlačítkem myši v Navigation Tree / New a v následujícím dialogu vybereme požadovaný objekt):



Obrázek 2.6: Základní struktura projektu pojmenovaného Tests

Nyní si stručně popíšeme význam jednotlivých složek vytvořených v rámci projektu:

- **UserFolder** (objekt typu Collection) - obsahuje pracovní složky jednotlivých uživatelů
  - Pavel (objekt typu Collection) - složka jednoho konkrétního uživatele (moje). Pro správnou funkčnost je nutno mít v projektu přítomnou a nastavenou pracovní složku, která je používána pro různé meziukládání dat (její nastavení probíhá v Process Simulate přes *File / Project management / Set as Working folder*). Pracovní složka se zobrazuje tučně.
- **Library** (objekt typu Collection) - obsahuje knihovny. Jedná se o knihovny výrobních prostředků (ResourceLibrary), knihovny dílů (PartLibrary) a knihovny technologických bodů (MfgLibrary).
- **Product** (objekt typu Collection) - obsahuje data dílů - například jednotlivé části určitého dílu.

- **Project** (objekt typu Collection) - zde se vytváří hierarchický model simulovaného objektu (v mém případě výrobní linky). Postupujeme od vyšší úrovně po nižší (linka, stanice, jednotlivá zařízení ve stanici). Plánování výrobních prostředků a procesů probíhá paralelně až do úrovně stanice, proto s výhodou používáme objekty typu dvojče (Twin[5]).

- **StudyFolder** (objekt typu StudyFolder) - slouží pro vytváření studií, které se následně používají pro simulace (jednotlivé studie jsou objekty typu RobcadStudy). Data (stanice, procesy, díly, svařovací body) nalinkovaná (pomocí jednoduchého přetažení - Drag and Drop) do RobcadStudy lze v Process Simulate nahrát pro simulaci. Přetažením dat do RobcadStudy se vytvoří jakýsi link - jedná se stále o stejná data. V důsledku toho mohou být změny provedené v simulaci jednoduše aktualizovány v projektu a naopak (ukládání dat ze studie probíhá pomocí *Selectively update the eMServer*).

Pozn: Struktura projektu není pevně daná a je možno si ji přizpůsobit a upravit dle osobních preferencí.

3. **vytvoření knihovny výrobních prostředků (ResourceLibrary) a knihovny dílů (PartLibrary)**

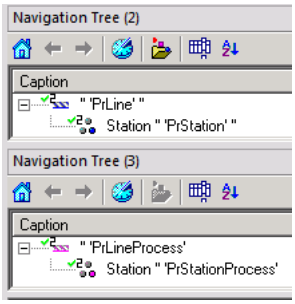
   Prvním krokem je nastavení SystemRootu tak, aby byly potřebné modely pro Process Designer / Process Simulate viditelné. Toto nastavení se provádí přes *Tools / Options*, kde v záložce eMServer nastavíme SystemRoot na požadovanou cestu na disku (PavelSimekDP / Root). Samotné vytvoření knihovny výrobních prostředků v Process Designer poté provedeme tak, že v Navigation Tree označíme složku Libraries a zvolíme možnost *Tools / Administrative Tools / Create Engineering Libraries* (v Process Simulate je proces vytvoření knihovny obdobný).

   Pozn: Knihovny výrobních prostředků obsahují prototypy výrobních prostředků (například robotů), v projektu můžeme následně z těchto prototypů vytvořit libovolné množství instancí - konkrétních objektů (ikona prototypu obsahuje písmeno P).

4. **vytvoření struktury výrobní linky**

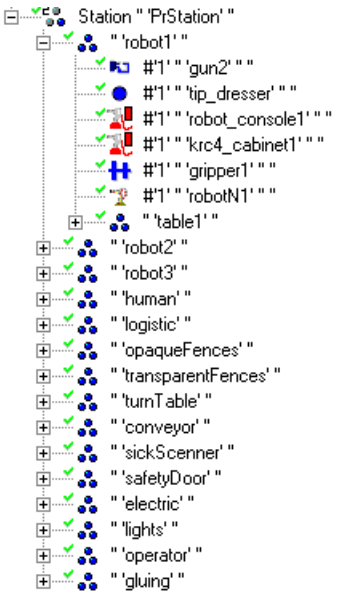
   Struktura výrobní linky se vytváří ve složce (přesněji řečeno Collection) Project. Jelikož se jedná o rozsahem a počtem zařízení jednoduchou linku, bude i odpovídající struktura poměrně jednoduchá. Na nejvyšší úrovni bude jeden objekt typu PrLine (linka), pod kterým bude jeden objekt typu PrStation (stanice). Přidání objektů se provádí stejně jako přidání kolekcí, které bylo popsáno výše. Výsledná struktura:

[5] Twin objekt poznáme podle jeho ikony, ve které je vykresleno číslo 2



Obrázek 2.7: Struktura projektu

Následně jsem pod stanici přidal potřebné výrobní prostředky (z ResourceLibray). Výrobní prostředky se přidávají jednoduchým přetažením (čímž z knihovního prototypu vytváříme konkrétní instance). Tímto způsobem jsem přidal potřebné roboty, otočný stůl, ploty, grippery, lakovací pistole atp. Následně jsem (pro lepší přehlednost) tyto objekty sdružil podle funkčnosti do několika objektů typu Compoud Resource. Výsledná struktura vypadá zhruba následovně:



Obrázek 2.8: Struktura linky

5. **vytvoření studie a nalinkování příslušných objektů**

   Nyní můžeme přistoupit k vytvoření studie - objektu typu RobcadStudy, která se přidává pod StudyFolder. Studie je objekt, který se používá při simulaci. Po vytvoření studie je nutné do ní přetáhnout (nalinkovat) objekty, se kterými budeme v rámci simulace pracovat (PrLine, PrLineProcess, díly atp.).

   Následně je možno z Process Designer studii otevřít (nahrát) pro simulaci v Process Simulate. Existují 2 simulační módy:

   - Standard Mode (možnost Open with Process Simulate in Standard Mode) - jedná se

o základní časovou (time-based) simulaci, ve které je průběh simulace určen sledem operací (Gantovým diagramem). Tento simulační mód je vhodný pro základní ověření konceptu linky - tvorba/úprava layoutu, vytvoření a otestování jednotlivých operací, otestování dosahu robotů atp.

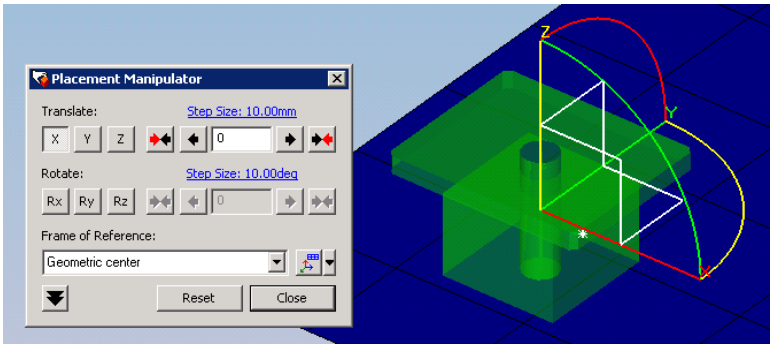
- Line Simulation Mode (možnost Open with Process Simulate in Line Simulation Mode) - je simulace řízená událostmi (event-based). Jedná se o nekonečnou simulaci, ve které je průběh simulace řízen na základě událostí (například stav PLC/OPC signálů).

6. **vytvoření layoutu** (přechod z Process Designer do Process Simulate)

   Po nahrání studie pro časovou simulaci začneme s rozmísťováním všech komponent. Během rozmísťování musíme brát v potaz proveditelnost operací, dosahy robotů a možné kolize (například při otáčení otočného stolu). Pro základní otestování dosahu robotů můžeme využít *Robot Jog*, případně *Joint Jog* (po označení příslušného robotu jsou obě možnosti dostupné přes pravé tlačítko myši). Během rozmísťování je rovněž nutné napevno spojit některá zařízení - například přípravky pro založení dílu napevno spojíme s otočným stolem tak, aby se otáčely společně se stolem (spojení se provádí pomocí *Modeling / Attach*). Pohybování objekty (tzv. Device) s definovanými pozicemi (tzv. Pose), například otočným stolem nebo lineární jednotkou, se provádí pomocí *Joint Jog*, případně pomocí *Pose Editor* (po označení příslušného objektu jsou obě možnosti dostupné přes pravé tlačítko myši).

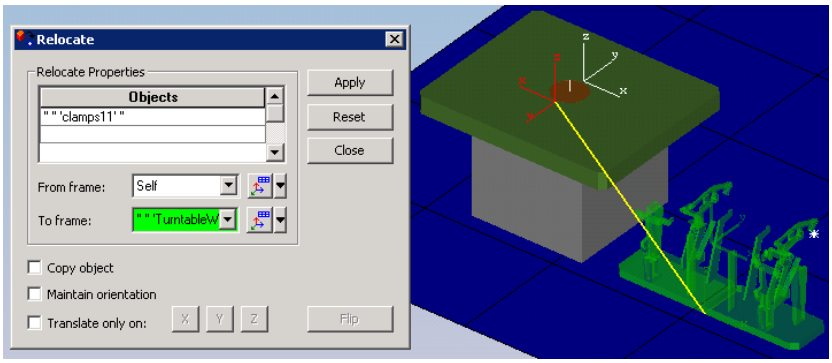
   Pro přesun objektů v prostoru jsem používal zejména:

   - Placement Manipulator - základní nástroj pro manipulaci s objekty v prostoru. Umožňuje posuny v jednotlivých osách (X,Y,Z) a rotace kolem jednolivých os (Rx,Ry a Rz) s definovaným krokem.

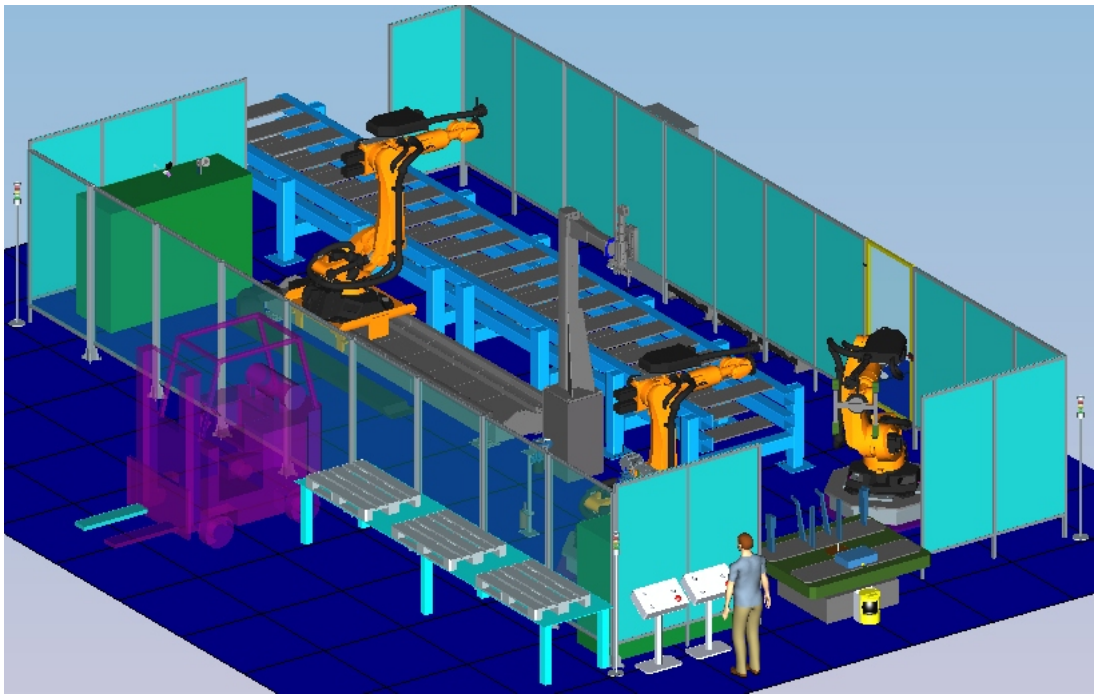


Obrázek 2.9: Manipulace s otočným stolem pomocí Placement Manipulator

   - Relocate - umožňuje přesun objektu z jednoho souřadného systému (framu) do jiného. Navíc poskytuje možnosti zachování orientace, kopírování objektů, přesunutí pouze v definovaných osách atp.



Obrázek 2.10: Přesunutí upínacího přípravku pomocí Relocate



Obrázek 2.11: Výsledná podoba linky

7. **vytvoření operací + časová simulace**

   Nyní můžeme přistoupit k vytváření jednotlivých operací (operace simulující otáčení stolu, manuální operace, činnost robotů atp.), které budeme v simulaci potřebovat. Prostředí Process Simulate nabízí několik typů operací, pro nás budou nejdůležitější operace následujícího typu:

   - **Object Flow Operation**
     Operace, která simuluje pohyb objektu po definované cestě (Path), tvořené lokacemi (Locations). Jako příklad můžeme uvést jízdu ještěrky nebo pohyb dílu po dopravníku.
   - **Device Operation**
     Operace, která simuluje pohyb zařízení (Device) z jedné polohy (Pose) do druhé.

Device je objekt s definovanými polohami (Pose), které můžeme spravovat v Pose Editoru. Jako příklad můžeme uvést otočení otočného stolu z pozice FWD do pozice HOME.

- **Generic Robotic Operation**
  Obecná robotická operace.
- **Weld Operation**
  Robotická operace pro bodové svařování.
- **Continuous Feature Operation**
  Robotická operace, která simuluje pohyb robotu po spojité křivce. Používá se například pro lakování, lepení, laserové svařování nebo leštění.
- **Non-Sim Operation**
  Operace, která není simulována, ale má definovaný určitý čas. Jako příklad si můžeme uvést čištění trysek.
- **Compoud Operation**
  Složka, která slouží pro strukturalizaci operací.
- **Jack/Human**
  Skupina operací, které umožňují simulovat činnost člověka (chůze, uchopení dílu atp.).

Pro simulaci činnosti linky bylo nutné vytvořit poměrně velké množství operací. Jedná se o: manuální operace simulující činnost člověka (chůze, uchopení dílu, založení dílu, stisknutí tlačítka atp.), operace pro otáčení otočného stolu, robotické operace (bodové svařování, lepení, lakování, manipulaci s díly, výměnu nástrojů atp.) a operace simulující pohyb výrobku po dopravníku.

Pozn: Proces vytváření jednotlivých operací zde z důvodu rozsahu není detailně popisován (postup je možno nalézt v [1], další informace případně v [3], [4], [6], [5] a [17]). Nicméně v rámci této práce byly vytvořeny návody na vytváření manuálních operací a spojitých robotických operací (z důvodu rozsahu nejsou součástí textu).

Po provedení finálních úprav layoutu, vytvoření a otestování všech operací byla činnost linky otestována v základní (časové) simulaci.

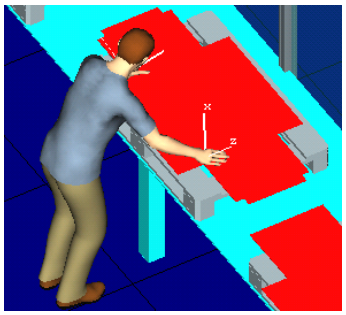
8. **simulace řízená událostmi**

Dalším krokem směrem k virtuálnímu zprovoznění linky bylo zprovoznit simulaci řízenou událostmi (Line Simulation Mode). V tomto režimu simulace bude výrobní linka řízena signály z PLC[6], které budou přes vhodné rozhraní namapovány na signály v Process Simulate. Existuje několik možných rozhraní mezi simulací a řídicím systémem (OPC, SIMIT, případně přímé napojení z PLCSim). Vzhledem k našemu požadavku na řízení z reálného PLC a dříve zmíněným problémům se SIMIT jsme jako rozhraní zvolili OPC Server (konkrétně Simatic .NET verze 8.1).

Prvním krokem bylo vytvoření fotoelektrických senzorů polohy a vygenerování signálů nutných pro řízení linky. Fotoelektrické senzory budou detekovat přítomnost dílů v přípravcích na otočném stole a budou indikovat pozici dílu na dopravníku (lakování probíhá

[6] Druhou možností je použití „interního PLC“ Process Simulate - tzv. CEE - Cyclic Event Evaluator

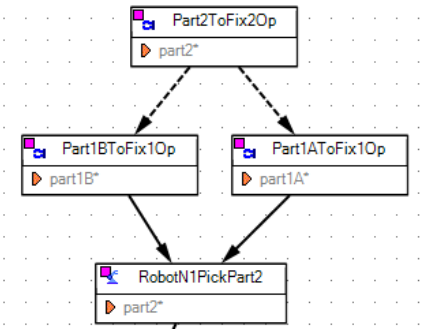
postupně na 2 pozicích na dopravníku). Dále bylo nutné vygenerovat signály potřebné pro řízení otočného stolu, signály pro řízení pohybu dílu na dopravníku a robotické signály pro řízení činnosti robotů. Posledním krokem pak bylo zprovoznění rozhraní a namapování Process Simulate signálů na signály (tagy) z PLC přes OPC rozhraní.



Obrázek 2.12: Pracovník (Jack) uchopující díl z palety

Vzhledem k tomu, že se nám (ani ve spolupráci s technickou podporou) nepodařilo nalézt způsob, jak vygenerovat signály pro spouštění operací, které simulují činnost člověka, bylo nutné upravit způsob zakládání dílů. Pro zakládání dílů do přípravků na otočném stole jsou ve finální verzi použity operace typu Object Flow, které simulují pohyby jednotlivých dílů z příslušných palet do přípravků na otočném stole. Ve finální verzi simulace řízené z PLC tedy bohužel nejsou manuální operace vůbec použity. Nicméně veškeré manuální operace (uchopení a zakládání jednotlivých dílů, chůze pracovníka, stisk jednotivých tlačítek) jsou v projektu vytvořené a byly úspešně otestované v časové simulaci. Pokud se tedy v budoucnu podaří nalézt způsob jejich spouštění na základě stavu PLC signálů, mohou být do simulace opět začleněny.

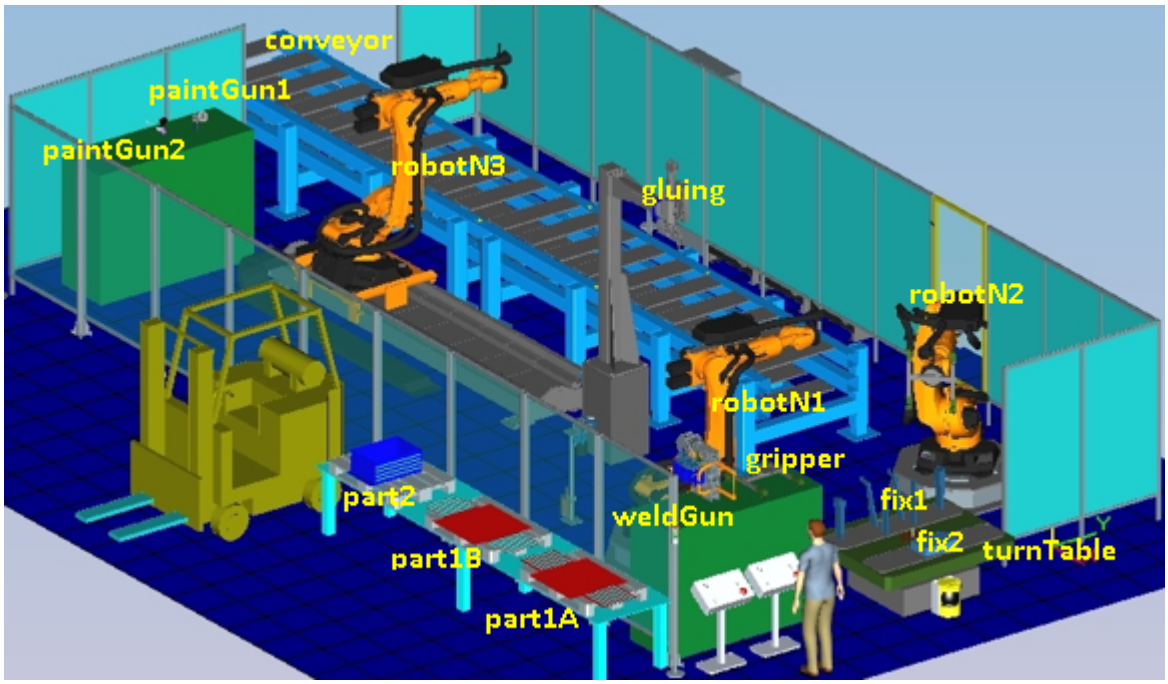
Dalším krokem nutným pro finální zprovoznění simulace řízené událostmi byla definice materiálového toku. Zde bych rád zmínil možnost definice alternativních cest, kterou jsem ve své práci využil a která byla důvodem pro změnu pořadí zakládání dílů (nejprve part2 a až poté part1A nebo part1B). Alternativní cesta je v materiálovém toku (viz Obr. 2.13) vykreslena přerušovanou čarou. Její nastavení se provádí v Material Flow Viewer pomocí *Set As Alternative Link*. Důvodem pro změnu pořadí zakládání dílů bylo to, že se mi nepodařilo vytvořit alternativní cestu při původním pořadí zakládání dílů (nejprve part1A nebo part1B a následně part2). V materiálovém toku je alternativní cesta použita na několika místech a to v situacích, kdy se provádí operace pro part1A nebo pro part1B (například svařování nebo lakování).



Obrázek 2.13: Material flow - Alternative Link

2.4.2 Virtuální zprovoznění

Finální podoba linky včetně popisu jednotlivých zařízení je uvededena na Obr. 2.14. Jedná se o linku, kterou můžeme rozdělit do dvou částí (pracovišť):



Obrázek 2.14: Svařovací linka s popisem jednotlivých zařízení

První pracoviště obsahuje otočný stůl (turnTable), robot č.1 (robotN1), robot č.2 (robotN2), stacionární trysku s lepidlem (gluing) a stůl s nástroji - na kterém jsou svařovací kleště (weldGun) a gripper pro robotN1. V této části linky probíhá zakládání dílů, jejich lepení a svařování, a následně je díl přesunut na dopravník (conveyor). Jednotlivé díly (part1A, part1B a part2) jsou umístěny na paletách poblíž otočného stolu. Otočný stůl je osazen dvěma přípravky pro založení dílu (fix1 - pro založení part1A nebo part1B a fix2 - pro založení part2). Pro zakládání jednotlivých dílů do přípravků jsou vytvořeny Object Flow operace Part1AToFix1Op, Part1BToFix1Op a Part2ToFix2Op řízené signály part1ToFix1 (založení part1A / part1B v závislosti na stisknutí tlačítka part1SWITCH v Simulation panel) a part2ToFix2 (založení part2). Přítomnost (poloha) dílů v přípravcích je indikována fotoelektrickými senzory/signály part1AInFix1, part1BInFix1 a partInFix2. Na začátku cyklu je vždy nutné nejprve založit part2. Po založení part2 dojde k otočení stolu do pozice HOME, ve které následně dojde k založení part1A/part1B. Pro řízení otočného stolu slouží 4 signály: tableToHOME - uvede stůl z aktuální pozice do pozice HOME, tableToFWD - uvede stůl z aktuální pozice do pozice FWD. Aktuální pozice stolu je poté indikována signály tableAtHOME a tableAtFWD.

Po založení part2 robotN1 uchopí gripper (Path #1), uchopí part2 z přípravku (Path #2) a provede nanesení lepidla (Path #3). Během činnosti robotu č.1 dojde k otočení stolu do pozice FWD. Následně robotN1 přiloží part2 s naneseným lepidlem na part1A/B (Path #4 pro part1A, případně Path #5 pro part1B) a jde odložit gripper (Path #6). Poté robotN1

uchopí svařovací kleště (Path #7) a svaří díly k sobě (Path #9 pro part1A, případně Path #8 pro part1B). Po dokončení svařování jde robotN1 odložit svařovací kleště (Path #10), zároveň s tím robotN2 (osazený gripperem) uchopí svařený výrobek a položí ho na dopravník (Path #1 pro part1A, případně Path #2 pro part1B). Vždy po 10 svařovacích cyklech je navíc nutné pro robotN1 provést frézování čepiček kleští (Path #11).

Druhé pracoviště obsahuje robot č.3 (robotN3), který je (pro zvětšení dosahu) umístěn na lineární jednotce[7]), dopravník (conveyor) a stůl s lakovacími pistolemi (paintGun1 a paintGun2) pro robotN3. Nejprve jsou svařené a slepené výrobky (svařování a lepení probíhá v pracovišti č.1) umístěny robotem č.2 na dopravník. Lakování robotem č.3 na dopravníku probíhá ve 2 pozicích. Přítomnost výrobku v první pozici je detekována fotoelektrickými senzory/signály productOnPostion1Front a productOnPosition1Back, přítomnost dílu v druhé pozici je detekována fotoelektrickými senzory/signály productOnPostion2Front a productOnPosition2Back. Pohyb výrobků na dopravníku je simulován operacemi typu Object Flow:

- MoveProductOnConveyor1Op - přesune výrobek part1A+part2 do 1. pozice. Operace je řízena signálem moveProductOnConveyor1.
- TurnProductOnConveyorOp - provede otočení výrobku part1A+part2 v 1. pozici. Operace je řízena signálem turnProductOnConveyor.
- MoveProductOnConveyor2Op - přesune výrobek part1A+part2 do 1. pozice. Operace je řízena signálem moveProductOnConveyor2.
- MoveProductOutOfLineOp - přesune výrobek part1A+part2 pryč z linky - například do vysoušecí části. Operace je řízena signálem moveProductOutOfLine.
- MoveProduct2OnConveyor1Op - přesune výrobek part1B+part2 do 1. pozice. Operace je řízena signálem moveProduct2OnConveyor1.
- TurnProduct2OnConveyorOp - provede otočení výrobku part1B+part2 v 1. pozici. Operace je řízena signálem turnProduct2OnConveyor.
- MoveProduct2OnConveyor2Op - přesune výrobek part1B+part2 do 1. pozice. Operace je řízena signálem moveProduct2OnConveyor2.
- MoveProduct2OutOfLineOp - přesune výrobek part1B+part2 pryč z linky - například do vysoušecí části. Operace je řízena signálem moveProduct2OutOfLine.

Během pohybu výrobku do 1. pozice (případně i dříve) robotN3 uchopí lakovací pistoli č.1 (Path #1). Jakmile je výrobek posunut do 1. pozice (indikované signály productOnPosition1Front a productOnPosition1Back), provede robotN3 lakování první části výrobku (Path #2 pro part1A, případně Path #9 pro part1B), poté jde odložit lakovací pistoli č.1 (Path #3). Následně uchopí lakovací pistoli č.2 (Path #4) a provede lakování druhé části dílu (Path #5). Po dokončení lakování dojde nejprve k otočení výrobku na dopravníku a následně se díl po dopravníku posouvá do 2. pozice. Zároveň s posunem dílu robotN3 znovu uchopí lakovací pistoli č.1 (Path #1) a jde lakovat první část dílu z druhé strany (Path #7

[7]Lineární jednotka je jednotka, která slouží k lineárnímu pohybu robotu a je řízena řídicím systémem robotu jako externí osa. Pomocí lineární jednotky můžeme zvětšit pracovní rozsah robotu.

pro part1A, případně Path #10 pro part1B). Po dokončení lakování robot odloží lakovací pistoli č.1 (Path #3), uchopí lakovací pistoli č.2 (Path #4) a provede lakování druhé části dílu z druhé strany (Path #8). Po dokončení lakování výrobek po dopravníku pomalu opouští lakovací část linky. Zároveň s tím robotN3 odkládá lakovací pistoli č.2 (Path #6).

Pozn: Výsledný řídicí program zde není uveden z toho důvodu, že implemetace řízení této linky bude (pravděpodobně) v budoucnu součástí semestrální práce řešené studenty.

Pozn: Další informace o virtuálním zprovoznění (proces vytváření senzorů, generování signálů pro spouštění operací, práce s robotickými signály, mapování signálů atp.) je možno nalézt v Kapitole 4.

Signal Name = Electrical Name	DATA TYPE	IN/OUT	TYPE
isRunning	BOOL	I	level
tableToHOME	BOOL	Q	level
tableToFWD	BOOL	Q	level
tableAtHOME	BOOL	I	level
tableAtFWD	BOOL	I	level
part1AInFix1Sensor	BOOL	I	edge
part1BInFix1Sensor	BOOL	I	edge
partInFix2Sensor	BOOL	I	edge
productOnPosition1Front	BOOL	I	edge
productOnPosition1Back	BOOL	I	edge
productOnPosition2Front	BOOL	I	edge
productOnPosition2Back	BOOL	I	edge
robotN1StartProrgam	BOOL	Q	edge
robotN1ProgramNumber	BYTE	Q	level
robotN1ProgramEnded	BOOL	I	level
robotN2StartProrgam	BOOL	Q	edge
robotN2ProgramNumber	BYTE	Q	level
robotN2ProgramEnded	BOOL	I	level
robotN3StartProrgam	BOOL	Q	edge
robotN3ProgramNumber	BYTE	Q	level
robotN3ProgramEnded	BOOL	I	level
moveProductOnConveyor1	BOOL	Q	level
moveProductOnConveyor2	BOOL	Q	level
moveProduct2OnConveyor1	BOOL	Q	level
moveProduct2OnConveyor2	BOOL	Q	level
turnProductOnConveyor	BOOL	Q	level
turnProduct2OnConveyor	BOOL	Q	level
moveProductOutOfLine	BOOL	Q	level
moveProduct2OutOfLine	BOOL	Q	level
part1ToFix1	BOOL	I	level
part2ToFix2	BOOL	I	level

Tabulka 2.1: Seznam (OPC) signálů

Operation Name	Operation Type	Path #
RobotN1MountGripper	Generic Robotic operation	1
RobotN1PickPart2	Generic Robotic operation	2
RobotN1ApplyGlueOnPart2	Continuous Feature Operation	3
RobotN1PlacePart2OnPart1A	Generic Robotic operation	4
RobotN1PlacePart2OnPart1B	Generic Robotic operation	5
RobotN1UmnoutnGripper	Generic Robotic operation	6
RobotN1MountGun	Generic Robotic operation	7
RobotN1WeldPart1BWithPart2	Weld Operation	8
RobotN1WeldPart1AWithPart2	Weld Operation	9
RobotN1UnmountGun	Generic Robotic operation	10
RobotN1TipDressing	Generic Robotic operation	11

Tabulka 2.2: RobotN1 - seznam robotických operací

Operation Name	Operation Type	Path #
RobotN2PlaceProductOnConveyor	Generic Robotic operation	1
RobotN2PlaceProduct2OnConveyor	Generic Robotic operation	2

Tabulka 2.3: RobotN2 - seznam robotických operací

Operation Name	Operation Type	Path #
RobotN3MountPaintGun1	Generic Robotic Operation	1
RobotN3PaintPart1A	Continuous Feature Operation	2
RobotN3UnMountPaintGun1	Generic Robotic Operation	3
RobotN3MountPaintGun2	Generic Robotic Operation	4
RobotN3PaintPart2	Continuous Feature Operation	5
RobotN3UnMountPaintGun2	Generic Robotic Operation	6
RobotN3PaintPart1A_2	Continuous Feature Operation	7
RobotN3PaintPart2_2	Continuous Feature Operation	8
RobotN3PaintPart1B	Continuous Feature Operation	9
RobotN3PaintPart1B_2	Continuous Feature Operation	10

Tabulka 2.4: RobotN3 - seznam robotických operací

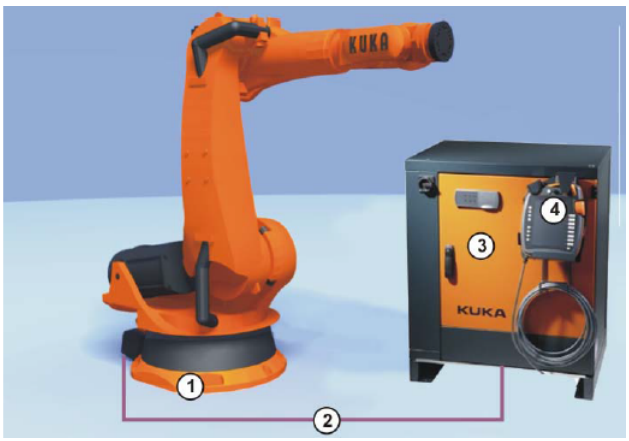
Kapitola 3

Robot, bezpečnost a vizualizace

Tato kapitola se věnuje stručnému popisu a zprovoznění několika průmyslových zařízení. Jedná se konkrétně o průmyslový manipulátor KUKA Kr5 Arc, bezpečnostní skener SICK S3000, bezpečnostní optický závěs SICK C4000 a vizualizační panel Siemens TP700 Comfort. Všechna tato zařízení jsou umístěna v laboratoři KN-E:s109 na Karlově náměstí. Tato kapitola rovněž poskytuje základní popis pracoviště v laboratoři.

3.1 KUKA Kr5 Arc

KUKA Kr5 Arc je nízkozáťěžový 6-osý průmyslový robot vhodný pro standardní obloukové svařování, manipulaci, lepení a jiné aplikace od firmy KUKA. Průmyslový robot (viz Obr. 3.1) se skládá (v dokumentaci je občas používán pojem komponenty průmyslového robotu) z robotu (manipulátoru) jako takového [1], řídicího systému (KRC) [3], propojovacích kabelů [2], ručního programovacího přístroje (Kuka Control Panel, KCP, někdy také označován jako Kuka SmartPad nebo Teach pendant) [4] a příslušného software. Konfigurace může eventuelně obsahovat také nástroj dle dané aplikace - například svařovací kleště nebo gripper (robot v laboratoři je vybaven gripperem), případně může být použita externí osa (například lineární jednotka).



Obrázek 3.1: Součásti průmyslového robotu

Software se sestává z Windows XPe (XP Embedded) a Systémového Software KUKA (KUKA System Software, KSS). KSS zajišťuje základní funkčnosti nutné pro činnost robotu tzn.: plánování trajektorií, správa vstupů/výstupů, správa dat a souborů atp. KSS je vybaven uživatelským rozhraním smartHMI. Mezi jeho základní funkčnost patří například možnost editace programů, správa uživatelů nebo inline formuláře pro programování.

typ	Kr5 Arc
počet os	6
opakovatelnost pozic	$\pm 0.04mm$
hmotnost	přibližně 127kg

Tabulka 3.1: Základní technické údaje

Jednotlivé osy manipulátoru jsou omezeny nastavitelnými softwarovými koncovými spínači (nastavují se při uvádění do provozu). Ty slouží jako ochrana zařízení a musí být nastaveny tak, aby manipulátor nemohl dojet na mechanické koncové dorazy (v případě najetí na mechanické dorazy může dojít k poškození manipulátoru). Mechanické koncové dorazy jsou přítomny na osách A1, A2, A3 a A5.

3.1.1 Druhy provozu

Robot může být provozován ve 4 režimech provozu. Jejich přehled a typické použití je uvedeno v Tab. 3.2. Postup pro vytvoření a zprovoznění programu je zhruba takový, že robot naučíme a naprogramujeme v režimu T1, poté program otestujeme v režimu T2 (je-li nutno otestovat program s vyšší rychlostí, než dovoluje režim T1) a následně (v našem případě) budeme vytvořený program volat z nadřazeného řídicího systému (v našem případě bezpečnostní PLC Simatic) v režimu AUT EXT.

Vytváříme-li program pomocí simulačního software (například Procces Simulate), musíme brát v potaz, že simulace neodpovídá přesně realitě. Z tohoto důvodu je nutné vždy program nejprve otestovat v režimu T1, v případně nutnosti provést potřebné změny a až poté spouštět program z nadřazeného PLC.

Pozn.: V testovacích režimech T1 a T2 je při provádění programu (případně při manuálním pohybu) nutné držet stisknutý spínač souhlasu a tlačítko Start na KCP. V testovacích režimech T1 a T2 není ochrana obsluhy aktivní.

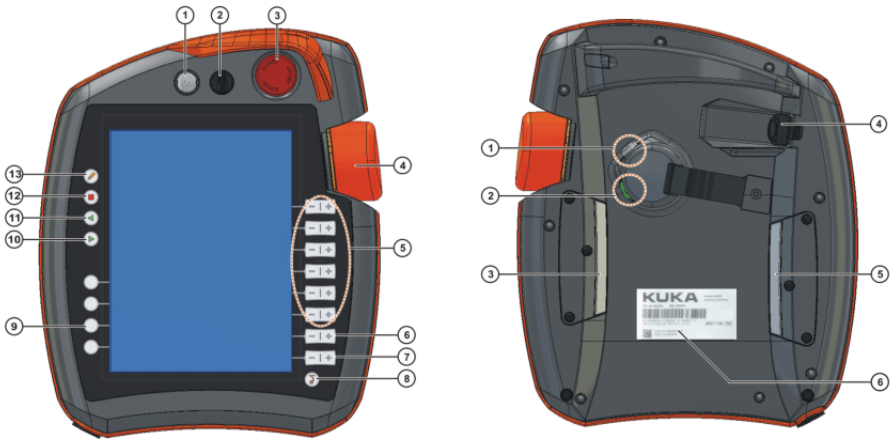
T1	pro testování programů, programování a učení	rychlost omezena na 250mm/s, umožňuje manuální pohybování robotem (Jog Mode neboli Jogging)
T2	pro testování programů	manuální pohybování robotem není možné, verifikace programu s naprogramovanou rychlostí
AUT	pro průmyslové roboty bez nadřazeného řídicího systému	manuální pohybování robotem není možné
AUT EXT	pro průmyslové roboty s nadřazeným řídicím systémem	manuální pohybování robotem není možné

Tabulka 3.2: Druhy provozu

3.1.2 Kuka Control Panel

KCP je ruční programovací přístroj sloužící k obsluze a programování robotu. Je vybaven dotykovým displejem s uživatelským rozhraním smartHMI.
Mezi hlavní prvky na přední straně patří:

- **2** - klíčový spínač k vyvolání manažera spojení - používá se pro změnu druhu provozu,
- **3** - tlačítko nouzového zastavení robotu v nebezpečných situacích,
- **5** - pohybové klávesy pro manuální pohybování robotem,
- **8** - klávesa pro zobrazení hlavní nabídky.



Obrázek 3.2: KCP

Důležité prvky na zadní straně KCP:

- **1**, **3** a **5** - spínače souhlasu - všechny spínače souhlasu mají 3 polohy: nestisknutý, střední poloha (pohony uvolněny) a stisknutý, spínač souhlasu uvolňuje pohony v režimu T1 a T2 (v ostatních režimech neplní žádnou funkci),
- **2** - tlačítko start - slouží pro vykonávání programů.

3.1.3 Manuální pohybování robotem

Při manuálním pohybování robotem můžeme využít pohybu v kartézské souřadné soustavě - TCP se pohybuje v kladném nebo záporném směru podél os zvoleného souřadnicového systému tzn.: X, Y, Z, pro natáčení kolem os zvoleného souřadnicového systému slouží pohybové klávesy A (rotace kolem Z), B (rotace kolem Y) a C (rotace kolem X). Nebo použijeme osově specifický pohyb, ve kterém pohybujeme jednotlivými osami $A1$ až $A6$ v kladném nebo záporném směru.

Při manuálním pohybování robotem se využívá pohybových kláves na KCP (případně je možno použít 6D myš na KCP). Užitečnou možnost (zejména při vytváření programů a

učení) poskytuje inkrementální ruční pohyb, který umožňuje pohybovat manipulátorem v krocích s definovanou velikostí - například o 10mm (případně ve stupních při použití osově specifického pohybu).

Obrázek 3.3: Osy robotu s vyznačenou orientací

Pokud potřebujeme manuálně (prostřednictvím pohybových kláves na KCP) pohnout robotem v situaci, kdy „neběží“ bezpečnost - typicky při uvádění do provozu, při poruše PLC atp., je možno využít režimu uvádění do provozu. Abychom mohli spustit (aktivovat) tento režim, musíme být přihlášeni jako SAFETY MAINTENANCE (změna uživatele se provádí přes *Main menu / Configuration / User Group*). Po přihlášení aktivujeme Režim uvádění do provozu přes *Main menu / Start-up / Service / Start-up mode*.

3.1.4 Programování

Existují tři možnosti pro vytváření robotických programů. První možnost je vytváření programů pomocí inline formulářů, druhou možností je vytváření programů v KRL (KUKA Robot Language), poslední možností je poté vygenerování (vyexportování) programů ze simulačního software (například Process Simulate) a následný download programu do robotu. První možnost je vhodná pro méně zkušené uživatele, druhá z možností (psaní programů přímo v KRL) je vhodná pro zkušené uživatele. Následuje ukázka struktury robotického programu:

```
1 DEF my_program( )
2 INI
3
4 PTP HOME Vel= 100 % DEFAULT
...
8 LIN point_5 CONT Vel= 2 m/s CPDAT1 Tool[3] Base[4]
...
14 PTP point_1 CONT Vel= 100 % PDAT1 Tool[3] Base[4]
...
20 PTP HOME Vel= 100 % DEFAULT
21
```

```
22 END
```

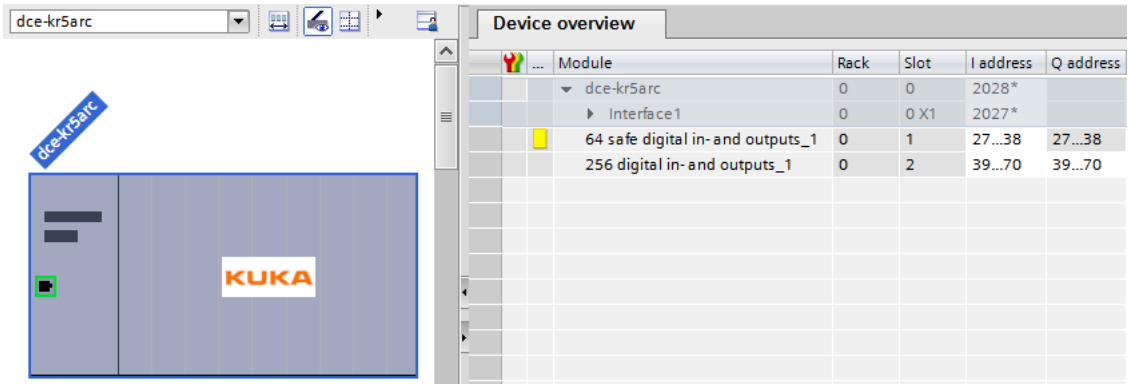
Každý robotický program začíná příkazem DEF, kde je definován název programu, následuje příkaz INI, který slouží k inicializaci interních proměnných a parametrů. První instrukce pohybu robotu musí jednoznačně definovat výchozí polohu, proto je zde použita pozice HOME (pozice HOME je globálně platná pozice uložena v řídicím systému robotu). Zpravidla bývá pozice HOME i poslední pozicí v programu. Hlavní část programu zde představují řádky 8 a 14 ve kterých probíhá nejprve lineární pohyb (instrukce LIN) a poté point-to-point pohyb (istrukce PTP).

3.1.5 Režim AUT EXT

V režimu AUT EXT je robot řízen z nadřazeného řídicího systému - v našem případě se jedná o bezpečnostní PLC Simatic. Mezi PLC a robotem je v tomto módu přenášeno několik signálů. Bezpečnostní signály jsou přenášeny přes PROFINET/PROFIsafe, ostatní poté přes PROFINET.

Konfigurace ze strany PLC

Veškerou konfiguraci ze strany PLC jsem prováděl ve vývojovém prostředí Step7 V13. Nejprve bylo nutno rozšířit hardwarový katalog o příslušné zařízení, následně bylo potřeba nastavit ip adresu a PROFINET jméno robotu (v mém případě 192.168.136.206 a dce-kr5arc). Následně bylo nutné přidat robot do hardwarové konfigurace (*Other field devices / PROFINET IO / I/O / KUKA Roboter GmbH / KUKA / KRC nexxt Device V8.2*) a provést příslušné propojení s PLC. Dalším krokem bylo přidání bezpečnostních vstupů a výstupů (*Module / 64 safe digital in- and outputs*) a normálních vstupů a výstupů (*Modules / 256 digital in-and outputs*). Výslednou konfiguraci včetně přiděleného adresového prostoru pro bezpečnostní i normální I/O vidíme na Obr. 3.4.



Obrázek 3.4: HW konfigurace robotu

Posledním krokem pak bylo nastavení parametrů PROFIsafe komunikace - v Device overview označíme 64 safe digital in- and otuputs a v záložce *Properties / PROFISafe* nastavíme potřebné parametry, konkrétně F_Source_Add, F_Dest_Add[1] a F_WD_Time.

[1] ProfiSafe adresu robotu je možno přečíst z KCP - *Main menu / Configuration / Safety Configuration / Communication Parameters* kde vidíme Profinet device a **Current safety ID**, které musí korespondovat s nastavením F_Dest_Add ve Step7. Aktuálně je nastavena na hodnotu 10.

Pozn: Během činnosti robotu je na obrazovce KCP zobrazováno poměrně velké množství hlášení (alarmy a eventy). Při zprovozňování robotu se mi (při problémech s funkčností) osvědčilo kvitovat všechna hlášení pomocí Confirm All, čímž jsem se „zbavil“ nepodstatných hlášení. Hlášení, která zůstala zobrazena, pak obvykle obsahovala relevantní informace o aktuálním problému.

Spouštění programu z PLC

Po provedení veškeré potřebné konfigurace a zprovoznění bezpečnosti můžeme přistoupit k řízení robotu (resp. k volání programů) z PLC. Postup pro volání programů z PLC můžeme rozdělit do následujicích kroků:

1. po spuštění robotu a PLC přepneme robot do režimu T1 (přepnutí se provádí pomocí klíčového přepínače, režim T1 je ze strany robotu indikován nastavením signálu **RI_ROBOT_T1**) a provedeme manuální kvitaci
2. navolíme (tlačítkem *Select* na KCP) program cell.src
3. zvolíme bázi a nástroj a aktivujeme Submit interpreter (*Submit interpreter/Start/Select*)
4. následně stiskneme spínač souhlasu a tlačítko start na zadní straně KCP a počkáme, dokud robot nedojede do HOME
5. následně se na KCP zobrazí hláška *Programmed path reached (BCO)*
6. nyní přepneme do režimu AUT EXT (režim AUT EXT je ze strany robotu idikován nastavením signálu **RI_ROBOT_EXT**)
7. v PLC nastavíme výstupy **RQ_EXT_START** a **RQ_DRIVES_ON**
8. v případě potřeby (po studeném startu řídicího systému) se provede test brzd
9. po dokončení testu brzd (byl-li prováděn) je vše potřebné připraveno (na KCP se zobrazí hláška *Wait for PGNO_VALID = true*) a můžeme začít volat programy z PLC
10. nastavíme požadované číslo programu (výstup **RQ_PGNO**)
11. vyšleme puls na **PGNO_VALID** (kontrolér reaguje na náběžnou hranu)
12. robot začne vykonávat program
13. o dokončení programu nás robot informuje nastavením **RI_PROGRAM_ENDED**
14. dokončení programu potvrdíme nastavením **RQ_PROGRAM_ENDED_ACK** (nastavením tohoto PLC výstupu resetujeme RI_PROGRAM_ENEDED)
15. v případě potřeby opakujeme sekvenci pro zavolání programu od kroku 10

V případě, že robot nedokončí vykonávaný program - například při porušení bezpečnosti, musíme vykonat následujicí sekvenci:

1. při přerušení vykonávání programu robot nastaví PLC vstup **RI_STOP_MESS**
2. po odstranění původce problému (např. po opuštění ochranné zóny hlídané laserovým skenerem) je nutná manuální kvitace tlačítkem
3. následně resetujeme PLC výstupy **RQ_EXT_START** a **RQ_DRIVES_ON**
4. vyšleme puls na **RQ_CONF_MESS** (kvitace chyby)
5. nastavíme **RQ_DRIVES_ON** - tím dojde k resetování **RI_CONF_MESS**
6. následně nastavením **RQ_EXT_START** řekneme robotu, aby dokončil program; o dokončení nás opět informuje nastavením vstupu **RI_PROGRAM_ENDED**
7. pokračujeme krokem č.14

3.2 Bezpečnost

Důležitou roli při provozu (nejen) mechanických zařízení hraje bezpečnost resp. zařízení, která mají pomoci bezpečnost zajistit. V našem případě je nutno monitorovat okolí robotu a v případě jeho narušení zastavit potenciálně nebezpečné pohyby. Hlavními prvky, které slouží k zajištění bezpečnosti jsou: bezpečnostní PLC Simatic, bezpečnostní laserový skener SICK a bezpečnostní optický závěs SICK. Tato podkapitola poskytuje stručný popis těchto zařízení a základní informace o jejich konfiguraci.

3.2.1 Bezpečnostní PLC

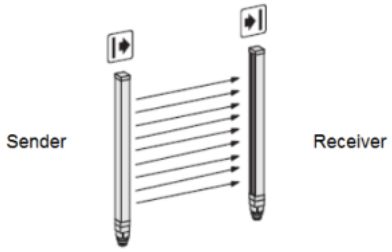
Základním prvkem bezpečnostního systému je bezpečnostní PLC (Safety PLC) Simatic, v našem případě konkrétně CPU 315F-2 PN/DP. Bezpečnostní PLC vykonává jak standardní uživatelský program, tak i bezpečnostní program (ten je vykonáván v organizačním bloku OB35, který je prováděn vždy po uplynutí určité časové prodlevy). Pro více informací o bezpečnostních PLC a bezpečnosti obecně mohu odkázat na [19].

3.2.2 Optický závěs SICK

K zajištění bezpečnosti v okolí robotu a zamezení přístupu do jeho pracovního prostoru je použit optický závěs (light curtain) SICK C4000 Standard. Optický závěs (viz Obr. 3.5) se skládá z vysílače (sender), přijímače (receiver), v případě potřeby (jako například v laboratoři na Karlově náměstí) je mezi vysílač a přijímač možno umístit zrcadlo(a) a vytvořit tak požadované ochranné pole (protective field). Princip funkce bepečnostního závěsu je zřejmý: vysílač vysílá optický signál, který je přijímán přijímačem, v případě přerušení signálu (způsobeno například vstupem člověka) dojde k resetování OSSD[2] výstupů (OSSD výstup je zdvojený) a následně k zastavení robotu.

[2]OSSD - Output Signal Switching Device - výstupní signál bezpečnostního zařízení, který je použit k zastavení nebezpečných pohybů.

Synchronizace vysílače a přijímače probíhá automaticky, jejich vzájemné propojení tedy není nutné. Jak přijímač, tak i vysílač jsou vybaveny optickými indikátory (4 LED a na přijímači navíc 7 segmentový displej), z jejichž stavu je možné určit aktuální stav systému (význam jednotlivých symbolů a indikátorů můžeme nalézt v [13]). K zajištění bezpečnosti slouží OSSD výstupy (tzn. vstupy PLC) přijímače (OSSD1 a OSSD2). Jakmile je ochranné pole narušeno, oba tyto výstupy jsou resetovány (logická 0). Výstupy senzoru (OSSD1 a OSSD2) jsou připojeny do distribuovaných bezpečnostních vstupů WAGO, jejichž hodnoty jsou následně přes PROFINET/PROFIsafe dostupné v PLC.



Obrázek 3.5: Bezpečnostní optický závěs - vysílač a přijímač

Konfigurace senzoru probíhá prostřednictvím PC s využitím Sick CDS (SICK Configuration & Diagnostic Software), jehož nejnovější verzi je možno stáhnout ze stránek výrobce. V našem případě pracuje senzor v režimu without restart interlock (tzn. po přerušení a opětovném obnovení optického signálu jsou OSSD výstupy opět nastaveny) - interlock (v podstatě nutnost kvitování tlačítkem po přerušení a opětovném obnovení ochranné zóny) poté realizujeme softwarově v rámci bezpečnostního PLC programu. Následující tabulky uvádí seznam použitých vodičů.

Barva	Popis
hnědá	+24 VDC
modrá	0 VDC
růžová	OSSD2
šedivá	OSSD1

Tabulka 3.3: Přijímač

Barva	Popis
hnědá	+24 VDC
modrá	0 VDC

Tabulka 3.4: Vysílač

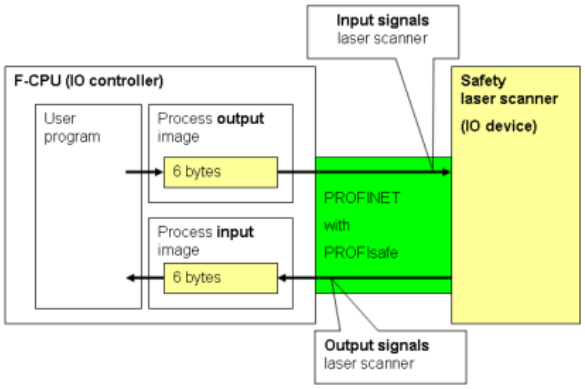
3.2.3 Laserový skener SICK

SICK S3000 je označení řady bezpečnostních senzorů od firmy SICK. Tyto laserové skenery mohou sloužit k ochraně osob a věcí. V našem případě bude senzor použit k monitorování okolí průmyslového robotu KUKA za účelem detekce osob vstupujících do jeho blízkosti a potažmo tak k zajištění bezpečnosti.

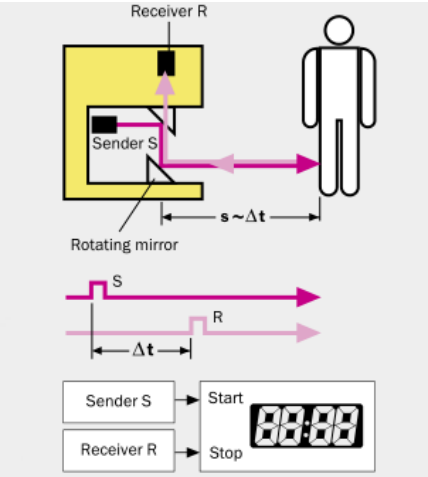
Obrázek 3.6: Bezpečnostní laserový skener SICK S3000

Senzory řady S3000 jsou koncipovány jako modulární - senzor se skládá ze 3 částí: **senzorové hlavy**, **I/O modulu** a **systémové přípojky** (v dokumentaci označována jako system plug), která obsahuje konfigurační paměť. Součástí systémové přípojky je napájecí kabel (hnědá/červená +24 VDC, modrá 0 VDC). Existují 3 typy senzorových hlav, které se liší maximálním skenovacím dosahem, čímž určují maximální možnou velikost ochranného pole (protective field). K dispozici jsou hlavy s dosahem 4m, 5,5m a 7m. Maximální velikost varovného pole (warning field) je pro všechny typy stejná a to 49m.

V laboratoři je instalován laserový skener SICK S3000 Profinet IO Advanced, model S30A-4111CP. Tento senzor disponuje 4 metrovým skenovacím dosahem, umožňuje vytvoření až 4 polí, každé pole (field set) se skládá z ochranného (protective field) a varovného pole (warning field). Skener umožňuje současné monitorování 2 polí. Díky použitému I/O modulu je umožněna přímá integrace do sítě PROFINET a následná komunikace přes bezpečnostní profil (specifikaci, standard) PROFIsafe. Pro tyto účely je senzor vybaven 2 portovým switchem. V rámci PROFINET sítě je senzor tzv. IO Device, který očekává konfiguraci z nadřazeného systému - tzv. IO Contoller (v našem případě bezpečnostní PLC Simatic), se kterým si cyklicky posílá IO data.



Obrázek 3.7: S3000 - PROFINET/PROFIsafe



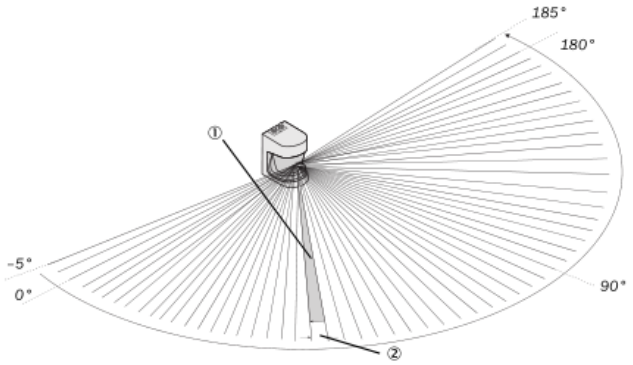
Obrázek 3.8: Princip funkce

3.2.3.1 Princip funkce

Princip funkce senzoru (viz. Obr. 3.8) je založen na metodě měření doby letu (time-of-flight). Senzor skenuje své okolí s použitím infračerveného laserového paprsku. Senzor vyšle velmi krátký světelný pulz (S) a zároveň začne eletronickými stopkami měřit čas. V okamžiku, kdy světelný pulz narazí na překážku, dojde k jeho odrazu a následnému přijetí. Z doby (Δt), která uběhla mezi vysláním a přijetím světelného pulzu pak senzor vypočte příslušnou vzdálenost objektu od senzoru (s).

K dosažení daného monitorovacího rozsahu je senzor vybaven zrcadlem, které se otáčí konstantní rychlostí, čímž rozmítá laserový paprsek takovým způsobem, aby výsledný monitorovaný rozsah byl právě 190°. Jednotlivé světelné pulzy jsou vysílány s úhlovou přesností (v

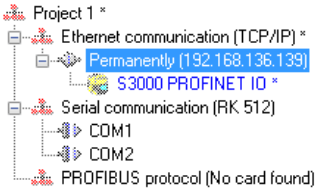
dokumentaci označováno jako angular resolution) 0.5° nebo 0.25° dle konfigurace (viz Obr. 3.9). Díky tomu senzor dosahuje schopnosti rozlišit objekty o velikosti 30mm až 150mm.



Obrázek 3.9: Rozmítání paprsku

3.2.3.2 Konfigurace senzoru

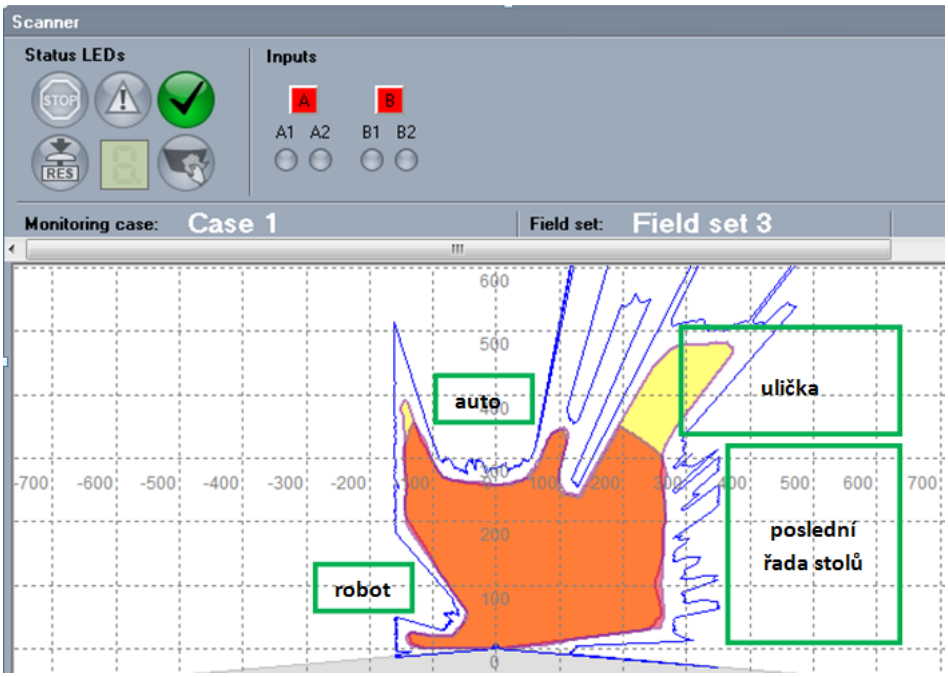
Konfigurace senzoru probíhá prostřednictvím PC s využitím Sick CDS. Nejprve připojíme skener síťovým kabelem k PC, následně otevřeme Sick CDS a založíme nový projekt. Ve struktuře nového projektu se automaticky vytvoří uzel *Ethernet communication (TCP/IP)*, klikneme pravým tlačítkem na tento uzel a stiskneme položku *Settings...* tím dojde k otevření dialogu, ve kterém nastavíme ip adresu skeneru (v mém případě 192.168.136.139). Následně klikneme pravým tlačítkem na *Permanently (192.168.136.139)* a stiskneme *Identify*, čímž se software pokusí identifikovat zařízení připojené přes síťové rozhraní a o výsledku nás informuje dialogem. V případě úspěšného nalezení dojde k přidání skeneru do projektové struktury. Následně vybereme skener v projektové struktuře a stikneme tlačítko *Receive*, čímž vyčteme aktuální konfiguraci ze skeneru. Následně dvojklikem na skener otevřeme Device Window, ve kterém se provádí veškeré nastavení.



Obrázek 3.10: Úspěšná identifikace zařízení

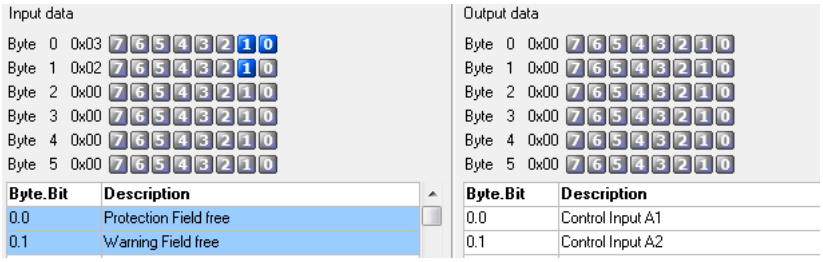
V záložce *System parameters* nastavujeme název aplikace a PROFIsafe adresu, která se musí shodovat s nastavením adresy (F_Dest_Add) ve Step7. V záložce *Inputs* volíme vstupy, které chceme použít pro přepínání monitorovacích případů, v naší aplikaci máme definován pouze jeden monitorovací případ (monitoring case), proto zvolíme možnost *No inputs*. V záložce *Restart* je možné nastavit způsob reakce na detekci objektu v ochranné zóně. Pokud chceme, aby skener po detekci objektu v ochranné zóně a následném odstranění objektu ze zóny stále hlásil narušení prostoru až do doby kvitace, zvolíme možnost *With restart interlock*. Druhou možností je *Without restart interlock*, kdy po opuštění ochranné zóny senzor opět nastaví OSSD. V našem případě používáme nastavení Without restart interlock (stejně jako v případě bezpečnostního závěsu, restart je poté řešen v bezpečnostním programu PLC). Záložka *PROFINET alarm* obsahuje seznam všech alarmů (červeně označené jsou aktivní).

Skenovací zóny je možné nastavit v záložce *Field sets*. Každé pole se skládá z ochranné a varovné zóny a může být použité ve více monitorovacích případech. Po vytvoření pole je nutné toto pole přiřadit monitorovacímu případu. K tomu je určena záložka *Cases*. V levé části je možné zvolit požadovaný případ. V pravé části pak k němu můžete přiřadit primární pole (Field set) a sekundární pole (Simultaneous Field set). Sekundární pole není nutno nastavovat (v naší aplikaci použito není). Po dokončení veškerého nastavení nahrajeme konfiguraci do skeneru pomocí *Transfer Configuration to the device*.



Obrázek 3.11: Diagnostika - online monitorování okolí

Jakmile je ve skeneru nahrán projekt s konfigurací, můžeme provádět online diagnostiku zařízení. Důležitá je zejména možnost online monitorování prostoru v záložce *Data recorder* - umožňuje nám zkontrolovat navrhnuté pole, skutečnou pozici překážek atp. Navíc zde vidíme stav LED indikátorů a stav 7 segmentového displeje. Zobrazení online stavu všech vstupů a výstupů je poté dostupné v záložce *PROFINET online monitor* - modře označené položky jsou nastaveny (true). Seznam vstupů a výstupů je uveden v Tab 3.5 a 3.6. Signály důležité pro naši konkrétní aplikaci jsou v tabulkách tučně zvýrazněny.



Obrázek 3.12: Diagnostika - online monitorování vstupů a výstupů

Bit	Description
0.0	Protective Field unoccupied – nastaven (TRUE), pokud v primární ochranné zóně není detekován žádný objekt
0.1	Warning field unoccupied - nastaven, pokud v primární varovné zóně není detekován žádný objekt
0.2	Simultaneous protective field unoccupied - nastaven, pokud v sekundární ochranné zóně není detekován žádný objekt
0.3	Simultaneous warning field unoccupied - nastaven, pokud v sekundární varovné zóně není detekován žádný objekt
0.4	Reset required (protective field) - nastaven, pokud skener detekoval objekt v primární ochranné zóně a je potřeba restartovat pole (používáno v režimu with restart interlock)
0.6	Reset required (simultaneous protective field) - nastaven, pokud skener detekoval objekt v sekundární ochranné zóně a je potřeba restartovat pole
1.0	Contamination - nastaven, pokud je přední kryt senzoru znečištěn
1.1	Monitoring case valid - nastaven, pokud je číslo monitorovacího případu platné
1.2	Monitoring case number – Bit 0 - číslo aktivního monitorovacího případu 0. bit
1.3	Monitoring case number – Bit 1 - číslo aktivního monitorovacího případu 1. bit

Tabulka 3.5: Výstupy skeneru (vstupy PLC)

Bit	Description
0.0	Monitoring case switching A1 – přepínání monitorovacích případů
0.1	Monitoring case switching A2 - přepínání monitorovacích případů
0.2	Monitoring case switching B1 - přepínání monitorovacích případů
0.3	Monitoring case switching B2 - přepínání monitorovacích případů
1.0	Reset the protective field - resetování primární ochranné zóny pokud došlo k detekci objektu v této zóně (používáno v režimu with restart interlock)
1.2	Reset the simultaneous protective field - resetování sekundární ochranné zóny pokud došlo k detekci objektu v této zóně (používáno v režimu with restart interlock)
1.4	Stand-by - přepnutí do režimu Stand by
1.5	Initialising - inicializace zařízení

Tabulka 3.6: Vstupy skeneru (výstupy PLC)

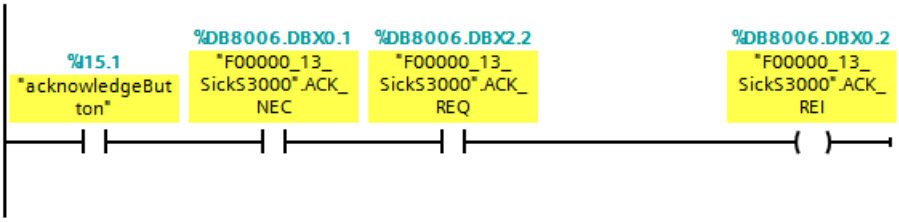
3.2.3.3 Konfigurace ze strany PLC

Veškerá konfigurace ze strany PLC byla provedena ve vývojovém prostředí Step7 V13. Po instalaci obsahuje hardwarový katalog pouze zařízení od firmy Siemens. Hardwarový katalog použitelných zařízení lze rozšířit pomocí importu GSD souboru (Generic Station Desription based on XML). Pro účely naší práce bylo nutné katalog rozšířit nejen o SICK skener, ale také o KUKA robot. Import GSDML souboru se provádí přes *Options / Install general station description file (GSD)*, v následujicím dialogu vybereme cestu k požadovanému *.XML souboru a stiskneme tlačítko *Install*. Po instalaci je nutné restartovat vývojové prostředí Step7.

Po instalaci GSDML souboru můžeme do stávající konfigurace přidat SICK skener, který by nyní měl být dostupný v hardwarovém katalogu v sekci Other field devices. Po přidání skeneru provedeme jeho připojení do PROFINET sítě, přiřadíme mu ip adresu (v mém případě

192.168.136.139) a Profinet device name (v mém případě Sick-S3000). Následně dvojklikem otevřeme skener, v Device overview vybereme S3000 In/Out_1 a zobrazíme Properties. Zde jsou důležité zejména záložky PROFIsafe - zde je nutné nastavit F_Dest_Add tak, aby korespondovala s nastavením provedeným prostřednictvím SICK CDS a I/O addresses - zde vidíme, jaký adresový prostor byl skeneru přidělen (kam se mapují vstupy a výstupy skeneru).

Přestože je v rámci bezpečnostního programu prováděna reintegrace všech pasivních F-I/O (dle [19] se zařízení dostane do pasivního stavu například po ztrátě napájení, případně po chybě v komunikaci) pomocí knihovního funkčního bloku ACK_GL (aktivovaného kvitovacím tlačítkem), docházelo k problémům s reintegrací bezpečnostního skeneru (skener prostřednictvím alarmu požadoval reintegraci). Tento problém jsem vyřešil přidáním reintegrace převzaté z [11] na konec bezpečnostního programu - viz Obr. 3.13.



Obrázek 3.13: Reintegrace

Pozn: Kompletní bezpečnostní program pro PLC, včetně stručného popisu PROFIsafe rozhraní robotu je uveden v příloze této práce.

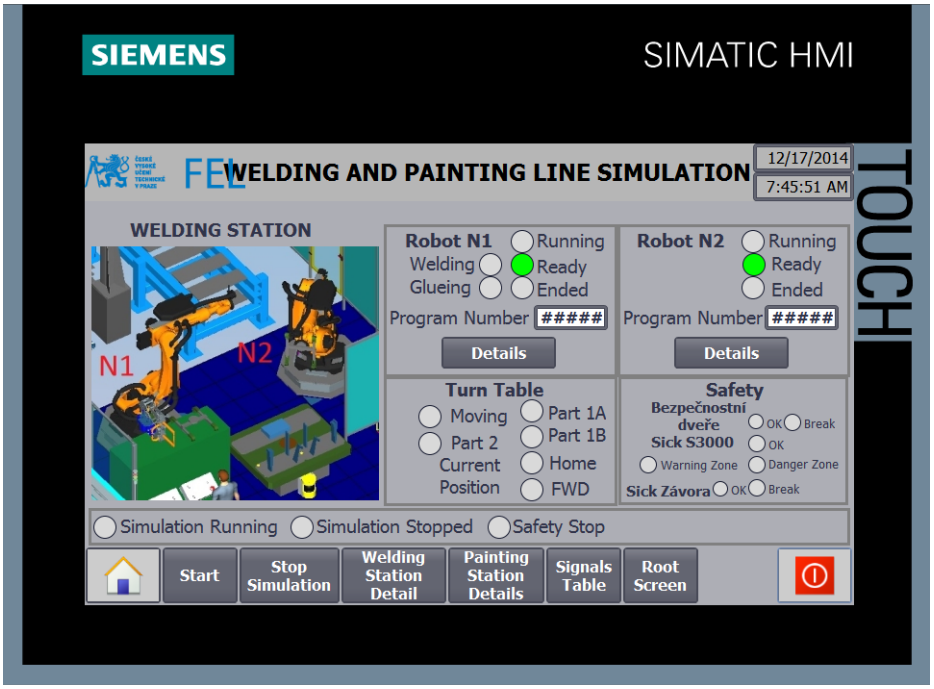
3.3 Vizualizace na operátorském panelu

Další součástí této práce bylo vytvoření vizualizace k svařovací lince, jejíž činnost je simulována v simulačním prostředí Process Simulate. Pro účely vizualizace byl použit 7 palcový dotykový vizualizační panel Siemens TP700 comfort. Tento panel je k PLC připojen přes průmyslovou síť PROFINET. Pro tyto účely je panel vybaven 2 portovým switchem. Více informací o panelu (montáž, připojení, konfigurace atp.) je možno nalézt v [26].

Vizualizace pro svařovací linku byla rozdělena do několika obrazovek (Screens). Každá z obrazovek je rozdělena do 3 částí. Horní část zobrazuje aktuální čas, datum a nadpis, spodní část obsahuje lištu s tlačítky. Spodní i horní část jsou společné pro více obrazovek a jsou vytvořeny jako Template. Template můžeme využít k definici společných prvků pro více obrazovek. Template se v projektu vytváří pod *Screen management / Templates*, přiřazení Template ke konkrétní obrazovce nebo obrazovkám pak provedeme v Dialogu Properties příslušné obrazovky. Samotná vizualizace je poté vytvořena v prostřední části obrazovky. Hlavní obrazovka (Root Screen) zobrazuje aktuální stav celé linky (stav simulace, stav bezpečnosti, stav jednotlivých pracovišť, počet vyrobených kusů atp.) s možností přepnutí na detailnější zobrazení jednotlivých pracovišť. Detailní zobrazení poskytuje další informace - stavy fotoelektrických senzorů detekujících polohu dílů, číslo aktuálně vykonávaného robotického programu, robotické signály atp.

Prostředí Step7 V13 umožňuje simulovat činnost HMI panelu. To je výhodné zejména z toho důvodu, že nemusíme pokaždé přehrávat projekt ve fyzickém zařízení (na které nemusíme mít od PC přímý dohled). Během spuštěné simulace funguje nejen samotné vykreslování

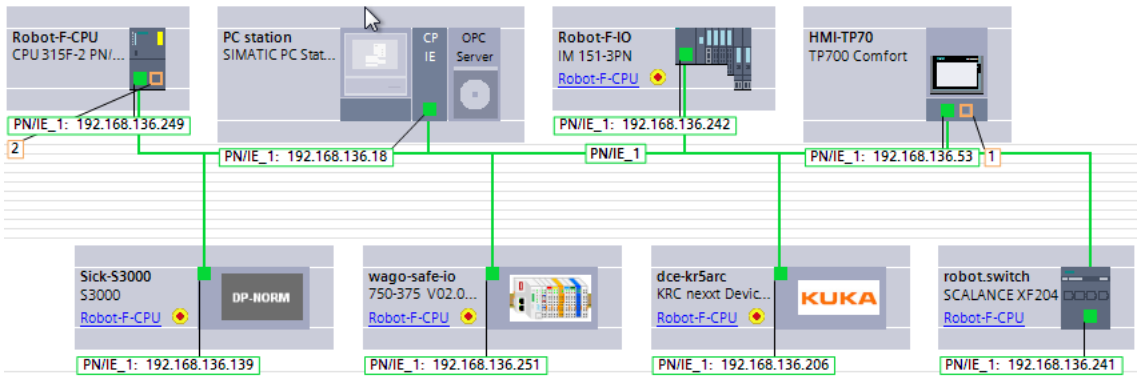
(layout), ale také činnost tlačítek atp. Simulace (položka *Start Simulation*) je dostupná prostřednictvím kontextového menu (vyvolané při stisku pravého tlačítka myši) příslušného HMI zařízení.



Obrázek 3.14: Ukázka vizualizace (simulace)

3.4 Popis pracoviště

Pracoviště v laboratoři na Karlově náměstí obsahuje všechny dříve zmíněné prvky. Výsledná konfigurace je patrná z Obr. 3.15. Konfigurace obsahuje kromě dříve zmíněných komponent (robotu, vizualizačního panelu, bezpečnostního skeneru a bezpečnostního optického závěsu) také síťový switch (Scalance) umístěný v kontroléru robotu a jednotku WAGO. Z obrázku jsou patrné jak ip adresy, tak i Profinet jména všech zařízení.



Obrázek 3.15: Výsledná konfigurace

Pro účely mechanické montáže byl pořízen stojanový systém od firmy Rittal. Čelní panel tohoto systému byl vyříznut a následně do něj byla uchycena jak tlačítka, tak i vizualizační

panel. Panel obsahuje tlačítko pro nouzové zastavení (červené, dvojitý rozpínací kontakt), kvitovací tlačítko (bílé, dvojitý spínací kontakt se signálkou), klíčový přepínač (jednoduchý spínací konkakt) a ještě jedno tlačítko (modré, jednoduchý spínací kontakt). Na vrchní část byl rovněž přichycen světelný maják.

Obrázek 3.16: Uspořádání čelního panelu

Signalizační maják je osazen 3 signalizačními prvky (shora červená, oranžová a zelená). Červená svítí, pokud je narušen bezpečnostní prostor - tzn. OSSD výstup bezpečnostního skeneru je nastaven do nuly nebo jsou do nuly nastaveny OSSD výstupy bezpečnostního světelného závěsu (případně jsou do nuly nastaveny všechny). Oranžová svítí, pokud je nutná manuální kvitace tlačítkem. Ve zbylých situacích svítí zelená.

Dovnitř skříně bylo umístěno PLC spolu s distribuovanými I/O. Konkrétně se jedná o jednotku IM 151-3 PN obsahující následující moduly:

- **PM-E 24VDC** - zdroj č.1,
- **4 F-DO DC24V/2A**[3] - bezpečnostní výstupy pro připojení signálky kvitovacího tlačítka,
- **4/8 F-DI DC24V**[4] - bezpečnostní vstupy pro připojení tlačítka nouzového zastavení a kvitovacího tlačítka,

[3]PROFIsafe adresa (F_Dest_Add) modulu se nastavuje pomocí DIP switchů umístěných ze strany modulu, aktuálně je nastavena na hodnotu 1 (0000000001)

[4]PROFIsafe adresa (F_Dest_Add) modulu se nastavuje pomocí DIP switchů umístěných ze strany modulu, aktuálně je nastavena na hodnotu 2 (0000000010)

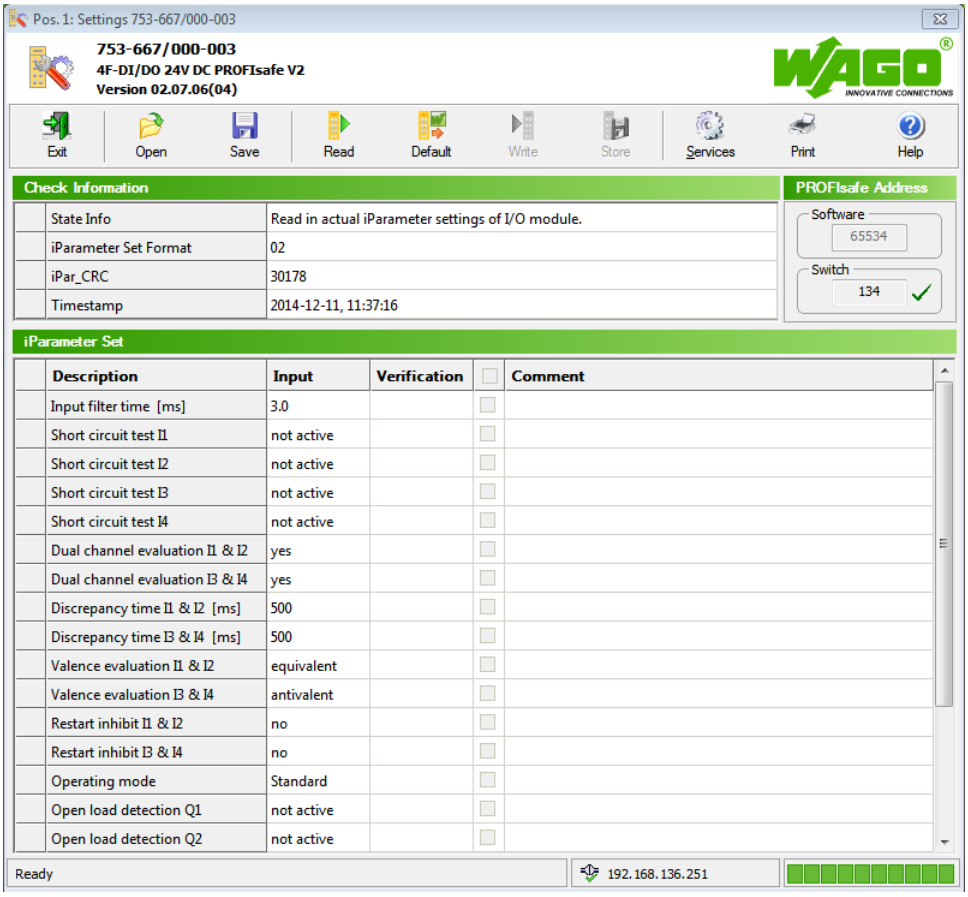
- **PM-E DC24V/8A RO** - zdroj č.2,
- **4DO x 24VDC/0.5A HF** - výstupy pro připojení signalizačního majáku,
- **4 DI x 24VDC HF** - vstupy pro připojení klíčového spínače a jednoduchého tlačítka.

Obrázek 3.17: Mechanická a elektrická instalace

Výstupy z optického závěsu SICK (tzn. vstupy PLC) a 2 kontakty referenční polohy pro robot (jeden spínací a jeden rozpínací) jsou připojeny do WAGO karty bezpečnostních vstupů a výstupů (**4FDI/4FDO iPar**), která je umístěna v jednotce WAGO **750-375** V02.01.xx (FW03), PE-DAP a k PCL připojena přes PROFINET. Veškerá konfigurace se provádí ve WAGO IO Check, který je ze stránek výrobce volně dostupný ke stažení.

Konfigurace WAGO

Stejně, jako v případě KUKA robotu a SICK skeneru je nutné začít tím, že pomocí importu GSD souboru rozšíříme hardwarový katalog vývojového prostředí Step7 V13. Následně přidáme WAGO zařízení (Profinet IO Device) do HW konfigurace - konkrétně přidáme zařízení 750-375 V02.01.xx, PE-DAP, nastavíme jeho ip adresu a Profinet device name. Poté přidáme kartu bezpečnostních vstupů/výstupů (konkrétně 75x-667/000-003 4FDI/4FDO 24V/2A). Další konfiguraci provedeme pomocí WAGO IO Check (stejně jako SICK CDS můžeme IO Check otevřít přímo ze Step7) - v Device overview vybereme kartu 75x-667/000-003 4FDI/4FDO 24V/2A_1, klikneme pravým tlačítkem a zvolíme možnost *Start device tool...* v následujícím dialogu vybereme (jedinou) možnost WAGO-I/O-Check 3 a stiskneme Start.



Obrázek 3.18: WAGO IO Check - konfigurace bezpečnostních IO

Tím dojde k otevření WAGO IO Check a dialogu pro nastavení parametrů karty bezpečnostních vstupů a výstupů - viz Obr. 3.18. V tomto dialogu je pro nás důležitých několik prvků a tlačítek. Ve spodní části dialogu (sekce iParameter Set) nastavujeme parametry vstupů a výstupů. Důležité je nastavit:

- Dual channel evaluation pro I1 & I2 na yes a stejně tak pro I3 & I4 (máme zdvojené OSSD vstupy i referenční tlačítka pro robot)
- Dicrepancy time pro I1 & I2 a pro I3 & a I4 (neshoda vstupů) - v manuálu k bezpečnostnímu závěsu není tento parametr uveden, použitá hodnota (500ms) byla převzata z [15]
- Valence evaluation - equivalent pro I1 & I2 (bezpečnostní závěs - OSSD1 a OSSD2) a antivalent pro I1 & I2 (referenční tlačítka robotu - spínací a rozpínací kontakt).

V prostřední části dialogu je pro nás důležitá hodnota iPar_CRC (po kliknutí pravým tlačítkem můžeme zkopírovat v šestnáctkové soustavě, kterou následně použijeme jako hodnotu ve Step7 V13) a PROFIsafe adresa, která se nastavuje pomocí DIP switchů (aktuálně nastavena na 134, opět musí korespondovat s nastavením ve Step7).

Po provedení veškeré potřebné konfigurace a nastavení stikneme tlačítko *Write*, čímž se upravené hodnoty zkopírují do sloupce Verification. Po kontrole údajů zaškrtneme check box ve sloupci Verification a stiskneme tlačítko *Store*, čímž nahrajeme konfiguraci do zařízení.

Kapitola 4

Využití ve výuce

Jedním z důležitých úkolů této diplomové práce byla tvorba úloh pro předmět Řídicí systémy (A3M35RIS). V rámci laboratorních cvičení tohoto předmětu řeší studenti během semestru několik menších, prakticky zaměřených úloh (typicky 3). Tyto úlohy si kladou za cíl seznámit studenty s programováním programovatelných logických automatů Simatic od firmy Siemens. V rámci řešení první úlohy se studenti seznámí s vývojovým prostředím Step7 (proces vytváření projektu, hardwarová konfigurace atp.), základním čtením vstupů, nastavováním výstupů a detekcí hran. Druhá úloha je poté zaměřena na praktické použití čítačů a časovačů a poslední úloha se zabývá funkcemi a funkčními bloky.

Doposud probíhalo testování řídicích programů (vytvořených studenty na základě zadání úlohy) s využitím PLCSim (softwarový simulátor PLC). PLC vstupy (senzory) byly simulovány klikáním na příslušné vstupy v PLCSim, výstupy poté generoval řídicí program. Toto řešení umožňovalo poměrně jednoduché testování funkčnosti, postrádalo však realističnost a přehlednou ilustraci funkčnosti. Z tohoto důvodu byly v prostředí Process Simulate vytvořeny 2 jednoduché linky (projekt RIS-Project1 a projekt RIS-Project2) a k nim příslušná zadání (zadání může být do budoucna poměrně snadno pozměněno). Pro každou linku byly vytvořeny 3 úlohy (celkem tedy 6 úloh).

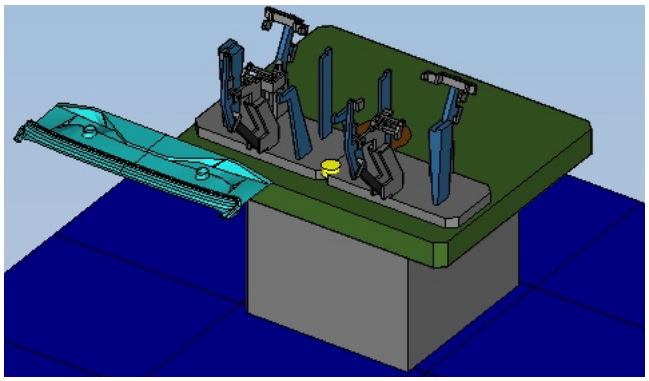
Při návrhu konkrétních úloh jsem vycházel ze zadání, které bylo zadáno studentům v rámci výuky v zimním semestru akademického roku 2014/2015 (i na zadání těchto úloh jsem se částečně podílel). Kompletní zadání všech vytvořených úloh je umístěno v příloze této práce. Tato kapitola se spíše než popisu jednotlivých úloh věnuje popisu tvorby linek a následně jejich virtuálnímu zprovoznění.

4.1 Linka č.1

Tato podkapitola poměrně podrobně popisuje proces vytváření a zprovoznění jednoduché linky. Takto podrobný popis je zde uveden proto, že linka je velmi jednoduchá a výsledný popis může do budoucna sloužit jako reference (návod) k virtuálnímu zprovoznění.

4.1.1 Popis linky

Jedná se o jednoduchou linku obsahující otočný stůl (turnTable) a přípravek pro upnutí dílu (clamps). Základní činnost linky se dá popsat velice jednoduše - do upínacího přípravku je založen díl (například pracovníkem nebo robotem), následně dojde k upnutí dílu v upínacím přípravku, otočení stolu, uvolnění dílu z upínacího přípravku a jeho následnému odebrání. Činnost linky může být popsána gantovým diagramem - viz Obr. 4.2.



Obrázek 4.1: Celkový pohled na linku

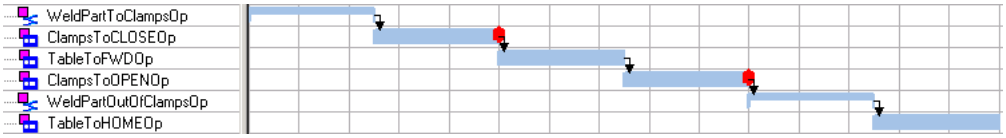
Otočný stůl (turnTable) se může nacházet ve dvou pozicích: v pozici HOME - pozice, ve které stůl přijímá díl (indikována signálem tableAtHOME) a v pozici FWD - pozice, ve které díl opouští linku (indikována signálem tableAtFWD). Upínací přípravek se může nacházet buď v pozici OPEN - stav, kdy je možno založit díl (indikováno signálem clampsAtOPEN), nebo v pozici CLOSE - stav, kdy je díl v přípravku pevně upnut (indikováno signálem clampsAtCLOSE) a může dojít k otočení stolu. Pro řízení otočného stolu slouží signály tableToFWD (uvede stůl z aktuální pozice do pozice FWD) a tableToHOME (uvede stůl z aktuální pozice do pozice HOME). Pro řízení upínacího přípravku slouží rovněž dva signály - clampsToOPEN (uvedení z aktuální pozice do pozice OPEN) a clampsToCLOSE (uvedení z aktuální pozice do pozice CLOSE). Dalším důležitým prvkem je fotoelektrický senzor, který indikuje přítomnost (pozici) dílu v upínacím přípravku (signál partInClamps). Tento senzor je v obrázku reprezentován žlutým válečkem. Zakládání a odebrání dílu do/z upínacího přípravku je řízeno signály weldPartToClamps (založení) a weldPartOutOfClamps (odebrání).

Pro simulaci činnosti linky potřebujeme 6 operací (jednotlivé operace můžeme vidět v Gantově diagramu):

- založení dílu do upínacího přípravku (operace typu Object Flow Operation)
- odebrání dílu z upínacího přípravku (operace typu Object Flow Operation)
- otočení stolu do pozice HOME (operace typu Device Operation)
- otočení stolu do pozice FWD (operace typu Device Operation)
- otevření upínacího přípravku (operace typu Device Operation)

- zavření upínacího přípravku (operace typu Device Operation).

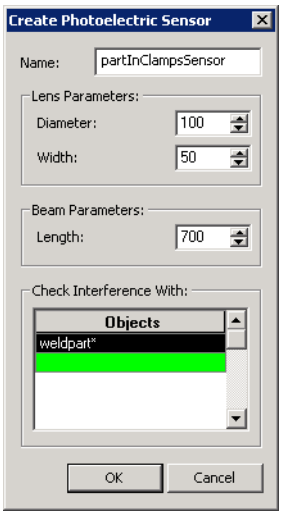
Po vytvoření těchto operací byla činnost linky nejprve otestována v časové simulaci - jak již bylo řečeno, jedná se o simulaci, kdy je sled operací (SOP = Sequence Of Operations) definovaný Gantovým diagramem. V Gantově diagramu můžeme vidět nejen operace, ale i události (Events) různých typů (nejčastěji používané jsou Attach/Dettach Event, a Signal Event). Zde vidíme Attach Event pro uchycení dílu v upínacím přípravku (na konci operace ClampsToCLOSEOp) a Dettach Event pro následné uvolnění dílu (na konci operace ClampsToOpenOp).



Obrázek 4.2: Sled operací - Gantův diagram

4.1.2 Senzory, signály a tok materiálu

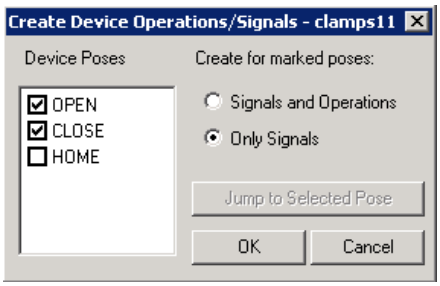
Následně bylo nutné připravit tuto linku tak, aby bylo možno řídit její činnost z PLC (v tzv. režimu virtuálního zprovoznění). Pro tyto účely je nutné zprovoznit simulaci řízenou událostmi (Line Simulation Mode). Zde začneme například přidáním fotoelektrického senzoru, který bude indikovat polohu dílu v upínacím přípravku. Senzor přidáme pomocí *CEE / Sensors / Create Photoelectic Sensor*, v následujícím dialogu zadáme jméno senzoru (což je zároveň jméno signálu příslušného k danému senzoru), jeho geometrické rozměry, délku (detekčního) paprsku a seznam objektů, jejichž polohu chceme senzorem detekovat. Tím dojde k přidání senzoru, který je nyní otevřen pro modelování.



Obrázek 4.3: Vytváření fotoelektrického senzoru

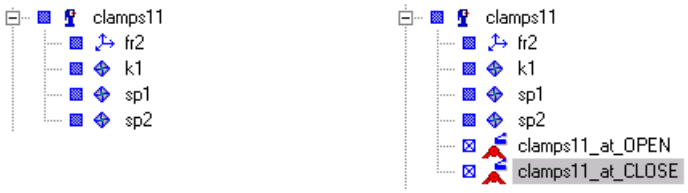
Následně senzor přesuneme do požadované polohy - na upínací přípravek a napevno ho s ním spojíme (pomocí *Modeling / Attach*). Nyní můžeme ukončit modelování senzoru

(*Modeling / End Modeling*). Následně můžeme upravit délku detekční zóny (paprsku) tak, abychom detekovali díl v požadované pozici. Umístíme tedy díl (weldPart) do pozice, ve které ho chceme pomocí senzoru detekovat (zde například využijime *Relocate* do poslední lokace Object Flow operace WeldPartToClampsOp) a upravíme délku paprsku (*CEE / Sensors / Edit Sensor*). Zbývá aktivovat sensor (*CEE / Sensors / Activate Sensor*), případně zobrazit/skýt detekční zónu (*CEE / Sensor / Display/Hide Detection Zone*). Nově vytvořený signál indikující stav senzoru a tím i přítomnost dílu v upínacím přípravku můžeme vidět v Signal Vieweru - jedná se o Resource Input Signal (bereme z pohledu PLC - senzor = PLC vstup) typu BOOL - pojmenovaný partInClampsSensor.



Obrázek 4.4: Vytvoření signálů pro řízení upínacího přípravku

Dalším krokem je vygenerování signálů potřebných pro řízení otočného stolu a upínacího přípravku. Pro tyto účely byly vytvořeny operace typu Device Operation - TableToFWDOp (uvede otočný stůl z aktuální pozice do pozice FWD), TableToHOMEOp (uvede otočný stůl z aktuální pozice do pozice HOME), ClampsToOPENOp (uvede upínací přípravek z aktuální pozice do pozice OPEN) a ClampsToCLOSEOp (uvede upínací přípravek z aktuální pozice do pozice CLOSE). Nyní je potřeba vytvořit signály pro spuštění těchto operací a signály pro indikaci aktuální polohy - to uděláme pomocí *CEE / Signal Generation / Create Device Operation/Signals*, kdy v následujícím dialogu (viz Obr. 4.4) vybereme pozice (tzv. POSE), pro které chceme signály vygenerovat - v případě upínek vybereme pozice OPEN a CLOSE, zvolíme možnost *Only Signals* (operace již máme vytvořené, pokud bychom neměli můžeme zvolit možnost *Signals and Operations*, čímž bychom si usnadnili práci). Tím dojde k vytvoření 4 důležitých signálů (které můžeme vidět v Signal Vieweru) - clamps_at_OPEN, clamps_at_CLOSE, clamps_to_OPEN a clamps_to_CLOSE. Význam signálů je patrný z jejich názvů. Stejný postup zopakujeme pro vytvoření potřebných signálů (a případně i operací) pro řízení otočného stolu - výstupem by měly být 4 signály: table_at_HOME, table_at_FWD, table_to_HOME a table_to_FWD. Veškeré takto vygenerované signály je možno přejmenovat (například pomocí F2) v Signal Vieweru.



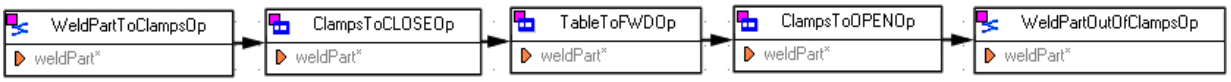
Obrázek 4.5: Struktura clamps před (vlevo) a po (vpravo) vygenerování signálů

Nyní máme připraveny všechny potřebné signály pro řízení upínacího přípravku a otočného stolu, zbývá vytvořit signály pro spuštění operace založení a odebrání dílu. Jedná se o operace WeldPartToClampsOp a WeldPartOutOfClampsOp - operace typu Object Flow Operation. Vygenerování těcho signálů provedeme pomocí *CEE / Signal Generation / Create All Flow Start Signals*. Tím dojde k vygenerování dvou signálů - WeldPartToClamps_start a WeldPartOutOfClamps_start, které slouží pro spouštění příslušných operací (jedná se Resource Output signály, tzn. PLC výstupy).

Signal Name	Data Type	Input/Output	Type
tableAtHOME	BOOL	input (I)	level
tableAtFWD	BOOL	input (I)	level
clampsAtOPEN	BOOL	input (I)	level
clampsAtCLOSE	BOOL	input (I)	level
partInFixSensor	BOOL	input (I)	edge
tableToHOME	BOOL	output (Q)	level
tableToFWD	BOOL	output (Q)	level
clampsToOPEN	BOOL	output (Q)	level
clampsToCLOSE	BOOL	output (Q)	level
weldPartToClamps	BOOL	output (Q)	level
weldPartOutOfClamps	BOOL	output (Q)	level

Tabulka 4.1: Tabulka signálů (po přejmenování)

Dalším důležitým krokem je definice toku materiálu. Jak již bylo řečeno, je simulace řízená událostmi nekonečnou simulací, ve které se (na rozdíl od časové simulace) nepracuje přímo s díly (Parts). Namísto toho se v této simulaci pracuje s tzv. Appearances, které jsou produkovány a konzumovány dle toho, jak je definován materiálový tok. Tok materiálu se definuje v Material Flow Vieweru. Pro tuto konkrétní linku je tok materiálu poměrně jednoduchý a vypadá následovně:



Obrázek 4.6: Tok materiálu

Nyní si (pro pozdější účely testování a monitorování) vytvoříme *.spss (Simulation Panel Signal Setting) soubor. Zobrazíme si Simulation Panel a Signal Viewer, následně vybereme v Signal Vieweru signály, jejichž hodnotu budeme chtít v Simulačním panelu monitorovat (všechny signály z Tab. 4.1) a v Simulation Panelu stiskneme *Add Signal To Viewer*. Následně si v případě potřeby vytvoříme skupiny signálů (osobně jsem si například vytvořil dvě skupiny signálů - PLC INPUTS a PLC OUTPUTS), a uložíme *.spss soubor.

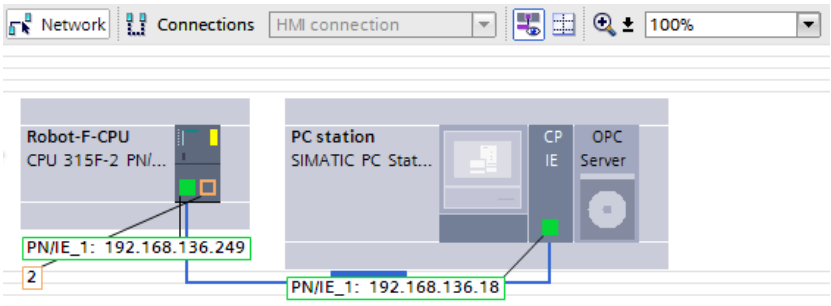
4.1.3 Virtuální zprovoznění s reálným PLC

Dalším krokem při virtuálním zprovozňování linky je konfigurace rozhraní mezi řídicím systémem (PLC, případně PLCSim) a simulačním prostředím (Process Simulate). Jak již bylo řečeno, existuje několik možností propojení (PLCSim, Simit a OPC rozhraní). Zde se zaměřím na možnost řízení linky z reálného PLC - konkrétně se bude jednat o bezpečnostní PLC Simatic 315F-2 PN/DP. V této části bude poměrně detailně popsána konfigurace nutná k tomu, aby linku vytvořenou v Process Simulate bylo možno řídit z PLC nastavováním a čtením signálů přes OPC server. Veškerá konfigurace a programování ze strany PLC byla prováděna ve vývojovém prostředí Step7 V13.

Konfigurace ze strany PLC

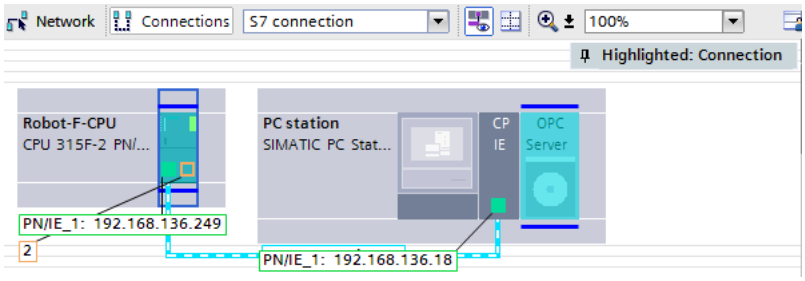
Začneme vytvořením projektu v prostředí Step7, následně přidáme PLC - *Add New Device / Controllers / SIMATIC S7-300 / CPU / CPU 315F-2 PN/DP / ES7 315-2FJ14-0AB0* (důležité je správně nastavit verzi - v našem případě V3.2) a PC stanici - *Add New Device / PC Sytems / PC general / PC Station*. Nyní přidáme z hardwarového katalogu do slotu 1 PC stanice komunikační modul - *Communications modules / Profinet/Ethernet / IE general* - zde je nutné zvolit správnou verzi dle nainstalované verze Simatic .NET (v našem případě 8.1) a do slotu 2 OPC server - *User applications / OPC Server* - i zde zvolíme verzi 8.1.

Nastavíme generování konfiguračního souboru OPC serveru - *PC station / Properties / XDB configuration*, kde zaškrtneme možnost *Generate XDB File* a zvolíme požadovanou cestu, kam se bude soubor generovat. Následuje nastavení ip adres - to se provádí v záložce *Properties / PROFINET interface / Ethernet addresses* a následně propojíme PLC s PC stanicí (viz Obr. 4.7).



Obrázek 4.7: Výsledná konfigurace

Nyní vytvoříme spojení mezi PLC a OPC serverem, po volbě typu propojení (záložka Connections, kde vybereme S7 Connection) nám vývojové prostředí zvýrazní zařízení, která tento typ propojení podporují (viz Obr. 4.8). Důležité je rovněž nastavit, které PLC tagy budou dostupné přes OPC rozhraní - nastavení probíhá v *OPC Server / Properties / S7 / OPC tags*.

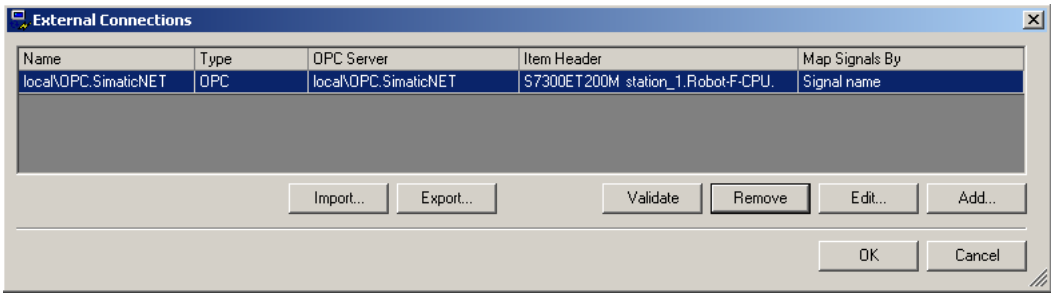


Obrázek 4.8: Vytvoření spojení mezi PLC a OPC serverem

Nyní vytvoříme v PLC všechny tagy z Tab. 4.1 (umístíme je do segmentu M). Posledním krokem je zkompilování projektu, jeho nahrání do PLC, vygenerování konfiguračního souboru pro PC stanici (*.XDB soubor) a jeho následný import, který se provádí v *Station Configuration / Import Station*. Užitečným nástrojem pro monitorování stavu OPC tagů, forcování jejich hodnot atp. je OPC Scout, který je součástí standardní instalace OPC serveru Simatic .NET. Tím jsme dokončili veškerou konfiguraci ze strany PLC (a OPC), můžeme tedy přistoupit ke konfiguraci ze strany Process Simulate.

Konfigurace ze strany Process Simulate

Konfigurace rozhraní ze strany Process Simulate probíhá v záložce *PLC* v dialogu *Tools / Options*. Zde zaškrtneme možnost *PLC* a *External Connection*. Následně kliknutím na *Connection Settings...* otevřeme dialog *External Connetions*, kde pomocí *Add / OPC Connection* přidáme novou položku. Případně naimportujeme existující konfiguraci z *.xlsx souboru.



Obrázek 4.9: Nastavení OPC rozhraní ze strany Process Simulate

Dalším důležitým krokem je namapování PLC tagů na odpovídající signály v Process Simulate. Zde je příhodné připomenout, že směr Process Simulate signálů se bere z pohledu PLC. Existuje několik možností mapování, které jsou dostupné v Signal Vieweru. První možností je mapování pomocí External Mapping Tool s využitím symbolické tabulky (*.sdf soubor) vygenerované ze Step7. Při používání tohoto způsobu mapování jsem však narazil na dva problémy: mapování PLC tagů typu BYTE (typicky číslo programu pro robot) a mapování PLC tagů ze segmentu M. Další nevýhodou je to, že Step7 V13 neumožňuje (na rozdíl od starší verze Step7) vyexportování symbolické tabulky (namísto toho je umožněn export do excelovské *.xlsx tabulky). Z toho důvodu jsem zvolil možnost mapování pomocí Signal Mapping Tool. Při tomto způsobu mapování vytvoříme mapovací tabulku, kterou použijeme jako vstup pro Signal Mapping Tool, následné mapování proběhne víceméně automaticky a

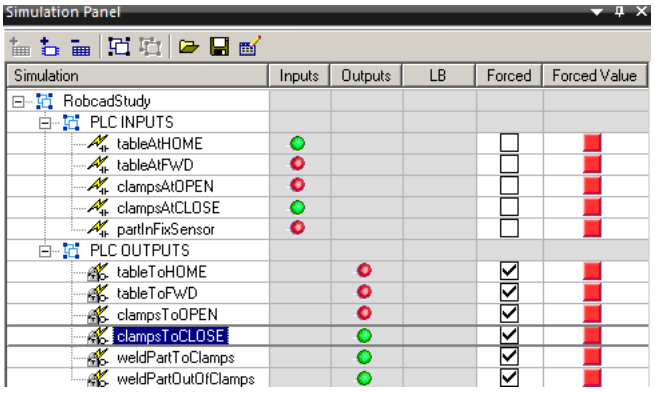
o jeho výsledku jsme informováni logem ve formě textového souboru. Mapovací tabulka má následující tvar:

	Signal Name	Electrical Name	Type	Address	comment
1	tableAtHOME	tableAtHOME	BOOL	I 0.0	
2	tableAtFWD	tableAtFWD	BOOL	I 0.1	
3	clampsAtOPEN	clampsAtOPEN	BOOL	I 0.2	
4	clampsAtCLOSE	clampsAtCLOSE	BOOL	I 0.3	
5	partInFixSensor	partInFixSensor	BOOL	I 0.4	
6	tableToHOME	tableToHOME	BOOL	Q 0.0	
7	tableToFWD	tableToFWD	BOOL	Q 0.1	
8	clampsToOPEN	clampsToOPEN	BOOL	Q 0.2	
9	clampsToCLOSE	clampsToCLOSE	BOOL	Q 0.3	
10	weldPartToClamps	weldPartToClamps	BOOL	Q 0.4	
11	weldPartOutOfClamps	weldPartOutOfClamps	BOOL	Q 0.5	

Obrázek 4.10: Mapování signálů - excelovská tabulka

4.1.4 Spuštění simulace

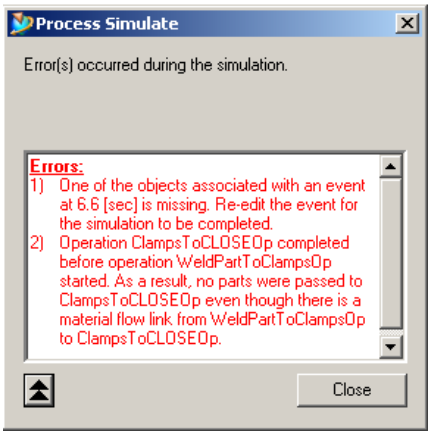
Nyní máme vše potřebné nakonfigurované a můžeme tedy přistoupit k testování funkčnosti linky (a následně případně k implementaci řídicího programu v PLC). Pro spuštění simulace v režimu virtuálního zprovoznění je nutné mít studii nahranou v Line Simulation Mode a mít aktuální operaci nastavenou na LineOperation. Následně je možno simulaci spustit, sledovat hodnoty signálů v simulačním panelu, který umožnuje forcování, případně můžeme hodnoty forcovat přímo ze Step7.



Obrázek 4.11: Simulační panel

Vzhledem k tomu, že v simulaci řízené událostmi musíme mít Definovaný tok materiálu, je nutné dodržovat sekvenci operací, které pracují s díly a jsou definovány v materiálovém toku. Pokud se vrátíme k Obr. 4.6, vidíme, že je zde 5 operací, které pracují s dílem (resp. s Appearance dílu). Pořadí vykonávání těchto operací (vykonávaných na základě nastavování/resetování signálů z PLC) musí být dodrženo. Jedině tak může být zaručena správná funkčnost simulace. Například operace ClampsToCLOSEOp musí být vykonána až poté, co byla vykonána operace WeldPartToClamps, jelikož prvně zmíněná operace pracuje (konzumuje) s Appearance dílu, který vyprodukovala operace WeldPartToClampsOp. Dalším problémem při nedodržení sekvence operací je nemožnost provedení Attach/Dettach Eventů,

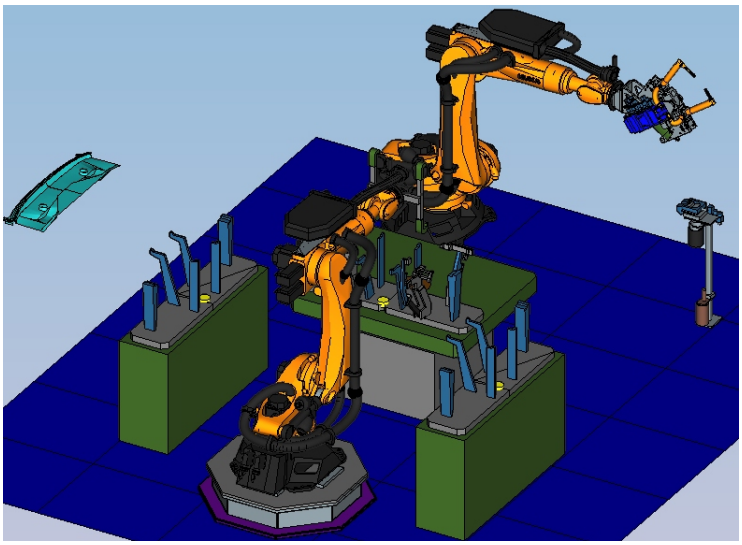
které pracují s Appearance. O nedodržení/porušení sekvence operací jsme informováni (stačí pozastavit simulaci tlačítkem Pause Simulation) dialogem (viz Obr. 4.12).



Obrázek 4.12: Nedodržení sekvence operací

4.2 Linka č.2

Linka č.2 částečně vychází z linky č.1, rozšířené o několik zařízení. Jedná se o linku, která obsahuje: otočný stůl, na kterém je umístěn upínací prípravek, 2 KUKA roboty - robot č.1, který je vybaven gripperem a robot č.2, který je vybaven svařovacími kleštěmi. Dále linka obsahuje frézku čepiček a dva pomocné stoly, na kterých jsou umístěny 2 přípravky pro založení dílu.



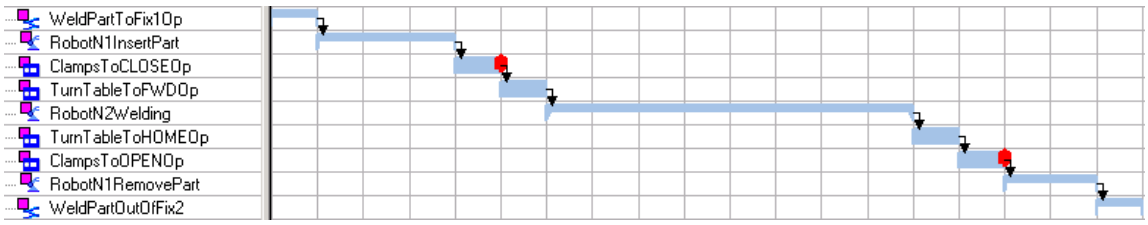
Obrázek 4.13: Celkový pohled na linku

Popis vytvoření a zprovoznění této linky již není tak podrobný jako popis linky č.1, namísto toho jsem se zde soustředil na předvedení dalších možností, které Process Simulate poskytuje, a které se u linky č.1 nevyužívaly (například řízení robotů a vykonávání robotických programů pomocí robotických signálů, propojení simulace a potažmo řízení linky s využitím PLCSim atp.).

Základní činnost linky můžeme popsat následujícím způsobem: Na začátku simulace je otočný stůl (turnTable) v pozici HOME a upínací přípravek (clamps) na otočném stole se nachází v pozici OPEN. Nejprve dojde k založení dílu (weldPart) do přípravku pro založení dílu (fix1) na stole č.1. Následně robotN1 uchopí díl a umístí ho do upínacího prípravku na otočném stole. Poté dojde k zavření upínacího přípravku (do pozice CLOSE) a otočný stůl se otočí do pozice FWD, kde provede robotN2 svaření dílu. Po dokončení svařování se stůl otočí zpět do pozice HOME a dojde k otevření upínacího přípravku. Následně robotN1 uchopí díl z upínacího přípravku a umístí ho do přípravku pro založení dílu (fix2) na stole č.2. Nakonec díl opuští linku a celá sekvence se opakuje.

Pro simulaci činnosti linky potřebujeme 10 operací (jednotlivé operace můžeme vidět v Gantově diagramu - viz Obr. 4.14):

- založení dílu do přípravku pro založení dílu na stole č.1 (operace typu Object Flow operation)
- odebrání dílu z přípravku pro založení dílu na stole č.2 (operace typu Object Flow operataion)
- otočení stolu do pozice HOME (operace typu Device Operation)
- otočení stolu do pozice FWD (operace typu Device Operation)
- otevření upínacího přípravku (operace typu Device Operation)
- zavření upínacího přípravku (operace typu Device Operation).
- odebrání dílu z přípravku pro založení dílu na stole č.1 a následné založení dílu do upínacího přípravku na otočném stole - operace prováděná robotem č.1 (operace typu Generic Robotic Operation)
- odebrání dílu z upínacího přípravku na otočném stole a následné založení dílu do přípravku pro založení dílu na stole č.2 - operace prováděna robotem č.1 (operace typy Generic Robotic Operation)
- svaření dílu robotem č.2 (operace typu Weld Operation)
- frézování čepiček svařovacích kleští - robot č.2 (operace typu Generic Robotic Operation).

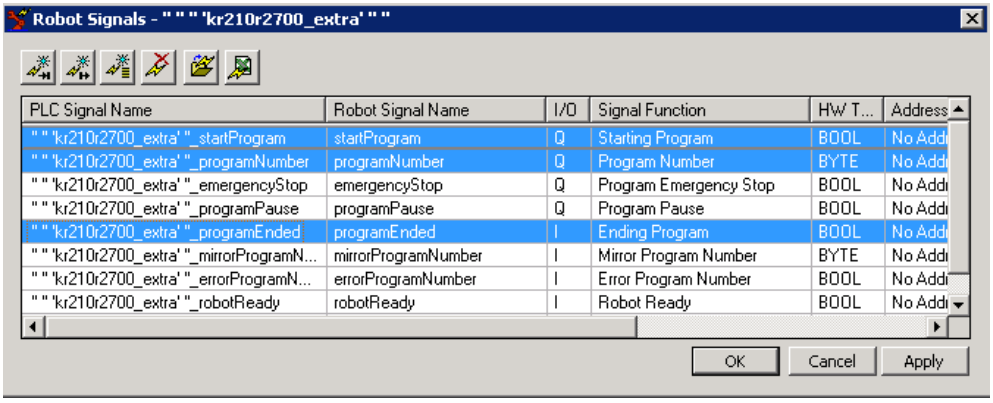


Obrázek 4.14: Sled operací - Gantův diagram

Nejprve byla činnost linky otestována v časové simulaci. Poté bylo nutno linku zprovoznit v režimu virtuálního zprovoznění. Zde jsem začal přidáním fotoelektrických senzorů pro

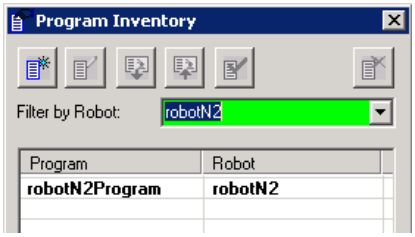
indikaci polohy dílu (partInFix1Sensor, partInClampsSensor a partInFix2Sensor). Následně byly vygenerovány signály a operace pro řízení otočného stolu a upínacího přípravku a signály pro spuštění operací založení dílu a odebrání dílu (operace typu Object Flow Operation).

Poté bylo nutné vygenerovat signály pro řízení robotů. Vybereme příslušný robot, zvolíme *Robotics / Robot Signals*, kde v následujícím dialogu můžeme signály vytvořit ručně, nebo je možno využít automatického vygenerování signálů pomocí *Create Default Signals*. Pro účely této linky jsou pro nás (pro každý robot) důležité 3 signály: startProgram, programNumber a programEnded (zvýrazněné v Obr. 4.15).



Obrázek 4.15: Robotické signály

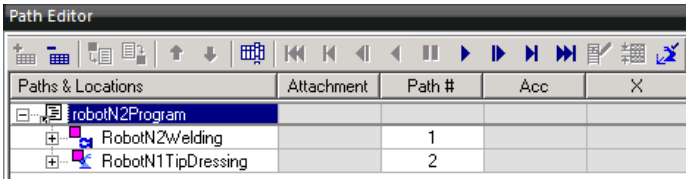
Následně musíme pro oba roboty vytvořit robotické programy. Robotické programy se skládají (obsahují) z jednotlivých cest (Path), reprezentovaných robotickými operacemi (např: operace typu Generic Robotic Operation nebo Weld Operation). Pro tyto účely otevřeme dialog Program Inventory (*Robotics / Robot Programs / Robotic Program Inventory*), ve kterém zvolíme robot, pro který chceme vytvořit program - například robot č.2 (robotN2). Pro tento robot máme vytvořeny 2 operace - operaci svařování a operaci frézování čepiček vykonávanou vždy po vykonání 10 operací svařování. Nový robotický program vytvoříme pomocí *Create New Program*, následně vytvořený program nastavíme pomocí *Set As Default Program* jako defaultní (defaultní program je v seznamu zobrazen tučně) a dvojklikem ho otevřeme v Path Editoru.



Obrázek 4.16: Robotic Program Inventory

Takto vytvořený program zatím neobsahuje žádné cesty (operace). Do tohoto robotického programu chceme přidat 2 operace: RobotN2Welding (Weld Operation) a RobotN2TipDressing (Generic Robotic Operation). Přidání robotické operace do robotického programu se provádí

tak, že vybereme příslušnou robotickou operaci v Operation Tree a následně v Path Editoru zvolíme *Add Operation To Program*. Po přidání robotických operací do robotického programu je rovněž nutné přiřadit jednotlivým cestám (Path) číslo (programNumber) - to se provádí ve sloupci Path # v Path Editoru (viz Obr. 4.17).



Obrázek 4.17: Robotický program složený z cest (robotických operací)

Pak je možné spouštět robotické programy (přesněji řečeno cesty definované v robotickém programu) na základě robotických signálů. Sekvence pro zavolání programu je velmi jednoduchá: nastavíme požadované číslo programu (signál programNumber) a následně nastavíme signál startProgram. Robot začne vykonávat příslušný program (reaguje na náběžnou hranu signálu startProgram) a o dokončení vykonávání programu nás informuje nastavením signálu programEnded. Celá sekvence se může opakovat.

4.2.1 Virtuální zprovoznění s PCLSim

Obr. 4.18 zobrazuje mapovací tabulku. Oproti předchozí lince zde vidíme signály dalšího datového typu - BYTE. Syntaxe pro mapování těchto signálů je patrná z obrázku.

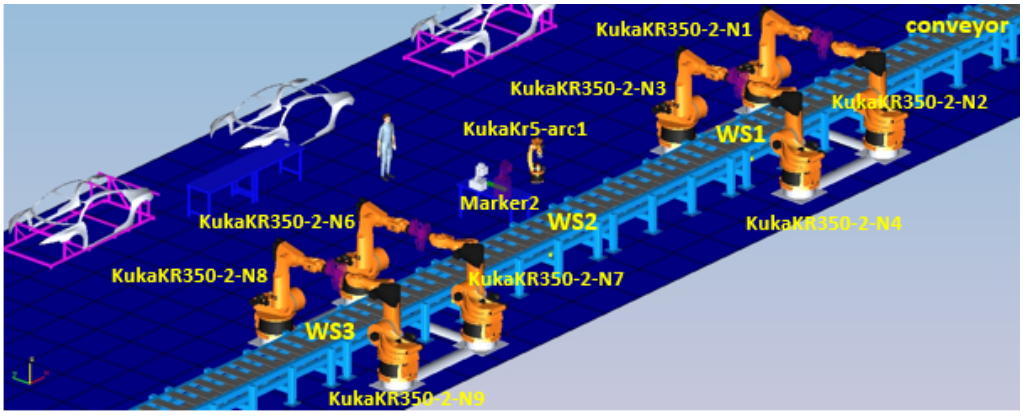
Ani po přechodu na 32 bitovou verzi Tecnomatix, která PLCSim podporuje se mi nepodařilo propojit simulaci v Process Simulate s PLCSim.

	A	B	C	D	
1	Signal Name	Electrical Name	Type	Address	comment
2	turnTableAtFWD	turnTableAtFWD	BOOL	I 0.0	
3	turnTableAtHOME	turntableAtHOME	BOOL	I 0.1	
4	clampsAtOPEN	clampsAtOPEN	BOOL	I 0.2	
5	clampsAtCLOSE	clampsAtCLOSE	BOOL	I 0.3	
6	partInFix1Sensor	partInFix1Sensor	BOOL	I 0.4	
7	partInClampsSensor	partInClampsSensor	BOOL	I 0.5	
8	partInFix2Sensor	partInFix2Sensor	BOOL	I 0.6	
9	robotN1ProgramEnded	robotN1ProgramEnded	BOOL	I 0.7	
10	robotN2ProgramEnded	robotN2ProgramEnded	BOOL	I 1.0	
11	turnTableToFWD	turnTableToFWD	BOOL	Q 0.0	
12	turnTableToHOME	turnTableToHOME	BOOL	Q 0.1	
13	clampsToCLOSE	clampsToCLOSE	BOOL	Q 0.2	
14	clampsToOPEN	clampsToOPEN	BOOL	Q 0.3	
15	partToFix1	partToFix1	BOOL	Q 0.4	
16	partOutOfFix2	partOutOfFix2	BOOL	Q 0.5	
17	robotN1StartProgram	robotN1StartProgram	BOOL	Q 0.6	
18	robotN2StartProgram	robotN2StartProgram	BOOL	Q 0.7	
19	robotN1ProgramNumber	robotN1ProgramNumber	BYTE	Q 100	
20	robotN2ProgramNumber	robotN2ProgramNumber	BYTE	Q 101	

Obrázek 4.18: Mapování signálů - excelovská tabulka

4.3 Svařovací linka

V rámci této práce došlo k opětovnému zprovoznění svařovací linky vytvořené v rámci diplomové práce Miroslava Konopy ([17]). Vzhledem k přechodu na novější verzi Tecnomatix (z verze 10 na verzi 11.1) a následným problémům s importem finální verze projektu svařovací linky muselo dojít k importu starší (ne zcela dokončené) verze. Následně bylo provedeno několik úprav (odebrání světelné indikace a příslušných logických bloků, odebrání robotu N5 z pracoviště č.2, menší úpravy materiálového toku atp.), nutných pro opětovné zprovoznění linky. Celkový koncept linky a hlavní část její funkčnosti však zůstala zachována.



Obrázek 4.19: Svařovací linka vytvořená Miroslavem Konopou (mnou upravená verze)

Dalším krokem byla implementace nebo spíše modifikace řídicího programu s ohledem na úpravy modelu linky. Řízení linky je založeno na řídicím programu, který byl vytvořen (a poměrně detailně popsán) v rámci diplomové práce M. Konopy. Řízení bylo doplněno jednoduchou vizualizací (ukázka vizualizace viz Obr. 4.20) s využitím vizualizačního panelu Siemens TP700 Comfort. Posledním krokem bylo vytvoření řádné dokumentace pro účely pozdějšího využití (ať už ve výuce nebo pro účely ukázek).

Jedná se o svařovací linku, která obsahuje 3 pracoviště - WorkSpace1 (WS1), WorkSpace2 (WS2) a WorkSpace3 (WS3). První (ve směru toku materiálu / pohybu dopravníku spojujícího jednotlivá pracoviště) pracoviště obsahuje 4 svařovací roboty typu Kuka Kr350, druhé pracoviště obsahuje jeden robot typu KUKA Kr5 Arc a marker pro označení dílu, třetí pracoviště obsahuje, stejně jako první, 4 roboty typu KUKA Kr 350. Činnost linky můžeme popsat zhruba následovně: karoserie automobilu je po dopravníku postupně posouvána mezi pracovišti, v prvním se provádí svařování karoserie, ve druhém dochází k označení dílu v markeru a jeho následnému přiložení na karoserii (manipulaci s dílem obstarává robot), ve třetím se, stejně jako v prvním, provádí bodové svařování.

Pro (případné) účely návrhu řízení uvedeme ještě podrobnější popis, vysvětlující význam jednotlivých signálů, operací, definující čísla robotických programů atp.:

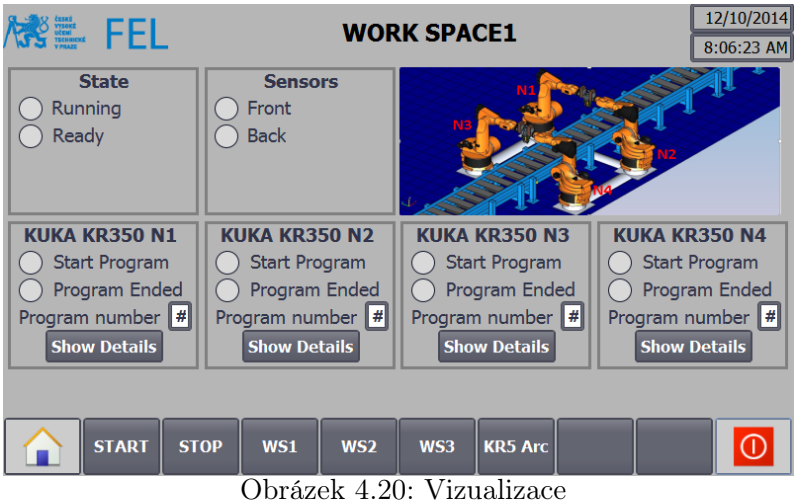
- Spuštění simulace je ze strany Process Simulate indikováno signálem StartSimulation (nastaven, pokud běží simulace v Process Siumlate).
- Karoserie automobilu je pomocí dopravníku posouvána mezi jednotlivými pracovišti ve směru od pracoviště č.1 přes pracoviště č.2 do pracoviště č.3. Následně svařená karoserie

(s přiloženým označeným dílem) opouští linku. Pro tyto účely jsou vytvořeny operace typu Object Flow: CarOnFrame_to_WorkSpace1, CarOnFrame_to_WorkSpace2, AssembledCarParts_to_Workspace3 a AssembledCarParts_to_Out, které simulují pohyb karoserie po dopravníku. Tyto operace jsou řízeny signály CarOnFrame_to_WorkSpace1_start (karoserie se po dopravníku posune do 1. pracoviště), CarOnFrame_to_WorkSpace2_start (karoserie se posune do 2. pracoviště), AssembledCarParts_to_Workspace3_start (karoserie se posune do 3. pracoviště) a AssembledCarParts_to_Out_start (karoserie opouští linku). Pozice karoserie na dopravníku resp. na pozicích v jednotlivých pracovišťích je snímána fotoelektrickými senzory - na příslušných pozicích jsou vytvořeny 2 senzory (celkem tedy 6 senzorů) detekující přední a zadní část karoserie. Poloha karoserie v prvním pracovišti je indikována signály LS_at_ReadyPosititonWS1_Back a LS_at_ReadyPositionWS1_Front (obdobně pro pracoviště č.2 a č.3).

- V prvním pracovišti provádí roboty N1 až N4 svařování karoserie. Podmínkou pro zahájení svařování je to, že karoserie auta je v příslušné pozici (indikováné signály z příslušných senzorů). Pro řízení činnosti každého robotu slouží 3 signály. Například pro robot N1 jsou to signály: KukaKR350-2-N1_startProgram, KukaKR350-2-N1_programNumber a KukaKr350-2-N1_programEnded. Číslo svařovacího programu (respektive robotické cesty definované v rámci robotického programu) je pro všechny roboty stejné a je rovno jedné (Path #1).

- Ve druhém pracovišti nejprve robot uchopí díl a umístí ho do markeru, který se musí nacházet v pozici HOME. Číslo programu pro uchopení dílu a jeho umístění do markeru je rovno jedné (Path #1). Pro označení dílu musíme marker uvést do pozice RUN (pro řízení markru jsou vytvořeny signály Marker2_at_HOME, Marker2_at_Run, Marker2_to_HOME a Marker2_to_RUN). Následně se marker vrací do pozice HOME, robot uchopuje označený díl a přikládá ho ke karoserii (Path #2).

- Pracoviště č.3 obsahuje roboty N6 až N9 a dochází zde ke svařování obdobně jako v pracovišti č.1. Číslo svařovacího programu (respektive robotické cesty definované v rámci robotického programu) je opět pro všechny roboty stejné a je rovno jedné.



Obrázek 4.20: Vizualizace

Signal Name = Electrical Name	DATA TYPE	IN/OUT	TYPE
LS_at_ReadyPositionWS1_Back	BOOL	I	level
LS_at_ReadyPositionWS1_Front	BOOL	I	level
LS_at_ReadyPositionWS2_Back	BOOL	I	level
LS_at_ReadyPositionWS2_Front	BOOL	I	level
LS_at_ReadyPositionWS3_Back	BOOL	I	level
LS_at_ReadyPositionWS3_Front	BOOL	I	level
LS_at_ReadyPositionOut_Front	BOOL	I	level
LS_at_ReadyPositionOut_Back	BOOL	I	level
StartSimunlation	BOOL	I	level
CarOnFrame_to_WorkSpace1_start	BOOL	Q	level
CarOnFrame_to_WorkSpace2_start	BOOL	Q	level
AssembledCarParts_to_Workspace3_start	BOOL	Q	level
AssembledCarParts_to_Out_start	BOOL	Q	level
Marker2_to_HOME	BOOL	Q	level
Marker2_to_RUN	BOOL	Q	level
Marker2_at_HOME	BOOL	I	level
Marker2_at_RUN	BOOL	I	level
KukaKR350-2N1_startProgram	BOOL	Q	edge
KukaKR350-2N1_programNumber	BYTE	I	level
KukaKR350-2N1_programEnded	BOOL	I	level
KukaKR350-2N2_startProgram	BOOL	Q	edge
KukaKR350-2N2_programNumber	BYTE	I	level
KukaKR350-2N2_programEnded	BOOL	I	level
KukaKR350-2N3_startProgram	BOOL	Q	edge
KukaKR350-2N3_programNumber	BYTE	I	level
KukaKR350-2N3_programEnded	BOOL	I	level
KukaKR350-2N4_startProgram	BOOL	Q	edge
KukaKR350-2N4_programNumber	BYTE	I	level
KukaKR350-2N4_programEnded	BOOL	I	level
KukaKR350-2N5_startProgram	BOOL	Q	edge
KukaKR350-2N5_programNumber	BYTE	I	level
KukaKR350-2N5_programEnded	BOOL	I	level
KukaKR350-2N6_startProgram	BOOL	Q	edge
KukaKR350-2N6_programNumber	BYTE	I	level
KukaKR350-2N6_programEnded	BOOL	I	level
KukaKR350-2N7_startProgram	BOOL	Q	edge
KukaKR350-2N7_programNumber	BYTE	I	level
KukaKR350-2N7_programEnded	BOOL	I	level
KukaKR350-2N8_startProgram	BOOL	Q	edge
KukaKR350-2N8_programNumber	BYTE	I	level
KukaKR350-2N8_programEnded	BOOL	I	level
KukaKR350-2N9_startProgram	BOOL	Q	edge
KukaKR350-2N9_programNumber	BYTE	I	level
KukaKR350-2N9_programEnded	BOOL	I	level
KukaKR5arc1_startProgram	BOOL	Q	edge
KukaKR5arc1_programNumber	BYTE	I	level
KukaKR5arc1_programEnded	BOOL	I	level

Tabulka 4.2: Seznam (OPC) signálů

Kapitola 5

Závěr

Tato práce měla dva hlavní cíle. Prvním cílem byla tvorba a následné zprovoznění virtuální svařovací/lakovací linky. Druhým cílem pak bylo rozšíření a zatraktivnění výuky řídicích systémů (předmět vyučovaný vedoucím této diplomové práce panem Ing. P. Burgetem, Ph.D. v magisterském studijním oboru Systémy a řízení).

V první části práce jsem v simulačním prostředí Process Simulate úspěšně implementoval a virtuálně zprovoznil svařovací linku. Tato linka simuluje různé typy operací, z našeho pohledu jsou zajímavé zejména robotické operace několika typů - konkrétně se jedná o robotické bodové svařování, spojité robotické operace lepení a lakování (při lakování se využívá lineární jednotka pro zvýšení rozsahu robotu) a výměnu robotických nástrojů (jeden z robotů provádí jak svařování, tak lepení, další z robotů pak postupně lakuje dvěma různými lakovacími pistolemi). Linka vytvořená v simulačním prostředí byla následně přes OPC rozhraní řízena z PLC a doplněna o jednoduchou vizualizaci. Dalším krokem bylo seznámení se, zprovoznění a následné začlenění do simulace několika průmyslových zařízení dostupných v laboratoři. V této části jsem úspěšně zprovoznil laserový bezpečnostní skener SICK, opětovně zprovoznil bezpečnostní optický závěs SICK, KUKA robot, jednotku WAGO a modifikoval bezpečnostní program pro PLC. Kromě samotné linky a jejího řízení (implementovaného v PLC v jazyce S7-Graph) jsem navíc vytvořil do budoucna užitečnou dokumentaci k jednotlivým podúlohám (robot, skener, atp.).

Druhým cílem pak bylo využít možností simulačního prostředí Process Simulate pro rozšíření výuky řídicích systémů. Pro tyto účely bylo úspěšně vytvořeno několik jednoduchých linek a 6 konkrétních (jednoduše modifikovatelných a rozšiřitelných) zadání, která mohou být ve výuce využity. Vzhledem k problémům s instalací nové verze Tecnomatix (instalaci jsem ve spolupráci s technickou podporou řešil několik týdnů) jsme bohužel nestihli nasadit simulační modely do výuky v tomto akademickém roce. Pro další běh předmětu a eventuelní použití ve výuce je však z mé strany vše připraveno (vytvořená zadání a projekty, připraven virtuální stroj). Namísto použití systému Lablink (vzdálená laboratoř usnadňující výuku řídicí techniky) jsme nakonec (zejména z důvodu poměrně velké náročnosti grafiky), provedli instalaci do virtuálního PC (s využitím VmWare). Virtuální stroj, který obsahuje vše potřebné pro práci - klienty Process Designer a Process Simulate (které se licencují k licenčnímu serveru), Step7, PLCSim a OPC server (Simatic .NET), bude následně nasazen na počítačích v laboratoři na Karlově náměstí.

Během řešení této práce jsem narazil na poměrně velké množství překážek a problémů, ale i přesto se mi podařilo cíle mé práce splnit. Díky výsledkům této práce, jako je integrace prvků funkční bezpečnosti, je nyní možné používat robot v laboratoři v plně automatickém režimu a napojit ho na simulační úlohy řešené studenty v rámci běžné laboratorní výuky. Do budoucna je možné propojit robot s výsledky dalších prací, které se na katedře řídicí techniky už řešily, jako například s měřením aktuálního příkonu a zobrazování historických dat na vizualizačním panelu, uspávání robotu v době nečinnosti apod. Rovněž je vhodné pokusit se v budoucnu vyřešit (ve spolupráci s technickou podporou) propojení simulace v Process Simulate s PLCSim.

Literatura

[1] *Siemens Industry Software.* Process Simulate Reference Manual. 2013.

[2] *Siemens Industry Software.* Process Designer Reference Manual. 2013.

[3] *M. Baumruk.* Process Simulate Advanced CEE 2013.

[4] *M. Baumruk.* Process Simulate Robotic Spot 2014.

[5] *M. Baumruk.* Process Simulate Robotic KUKA (VKRC) OLP Basic 2014.

[6] *M. Baumruk.* Process Simulate Basic Modeling & Kinematic 2014.

[7] *KUKA Roboter GmbH.* KR C4 PROFINET 2.2. 2012.

[8] *KUKA Roboter GmbH.* KR 5 ARC - Operating Instructions 2012.

[9] *KUKA Roboter GmbH.* KUKA System Software 8.2 - Operating and Programming Instructions for System Integrators 2012.

[10] *KUKA Roboter GmbH.* WorkVisual 2.4 - For KUKA System Software 8.1 and 8.2 2012.

[11] *Siemens.* Distributed Use of a Safety Laser Scanner on a SIMATIC F-CPU, with Monitoring Case Switching Using an F-CPU 2012.

[12] *SICK.* S3000 Safety laser scanner - Operating instructions 2012.

[13] *SICK.* S3000 PROFINET IO and S3000 PROFINET IO-OF Safety laser scanner - Addendum operating instructions 2014.

[14] *SICK.* C4000 Standard and C4000 Advanced Safety Light Curtain - Operating instructions 2014.

[15] *Siemens.* Distributed Use of a Safety Light Curtain on a SIMATIC F-CPU 2012.

[16] *Siemens.* Simatic HMI - Comfort Panels Operating instructions 2012.

[17] *M. Konopa.* Simulation of Production Processes. Diplomová práce, ČVUT, Fakulta elektrotechnická, 2013.

[18] *P. Burget.* Studijní materiály k předmětu A3M35RIS. ČVUT, Fakulta elektrotechnická, 2014.

[19] *P. Polák.* Bezpečnostní funkce pro Simatic S7. Diplomová práce, ČVUT, Fakulta elektrotechnická, 2009.

[20] *T. Urban.* Simulace rozsáhlejších výrobních celků. Diplomová práce, ČVUT, Fakulta elektrotechnická, 2014.

[21] *Siemens.* SIMIT SCE - Basic Library 2009.

[22] *Siemens.* SIMIT 7 - Getting Started 2013.

[23] *Siemens.* SIMIT SCE - User Manual 2009.

[24] *Siemens.* Training Document for SIMIT SCE - Module G1 2009.

[25] *Siemens.* Training Document for SIMIT SCE - Module G2 2009.

[26] *Siemens.* HMI devices - Comfort Panels Operating Instructions 2012.

[27] *WAGO.* WAGO-I/O-SYSTEM 750 - PROFINET IO advanced Fieldbus Coupler 750-375(/xxx-xxx) 2014.

[28] *WAGO.* WAGO-I/O-SYSTEM 750 - 4FDI/4FDO 24V/2A PROFIsafe V2 iPar 750-667/000-003 2013.

[29] *Siemens.* Distributed I/O System Fail-Safe Engineering - Installation and Operating Manual 2013.

[30] *Siemens.* ET 200S Distributed I/O System - Fail-Safe Modules 2005.

Příloha A

Úlohy pro předmět RIS

Následuje několik možných variant zadání pro předmět Řídicí systémy (A3M35RIS) vyučovaný na ČVUT FEL v magisterském studijním oboru Systémy a řízení panem Ing. Pavlem Burgetem, Ph.D. Všechna zadání je díky možnostem Process Simulate a poměrně detailnímu popisu/dokumentaci linek v Kapitole č.[4] možno poměrně snadno upravit/rozšířit.

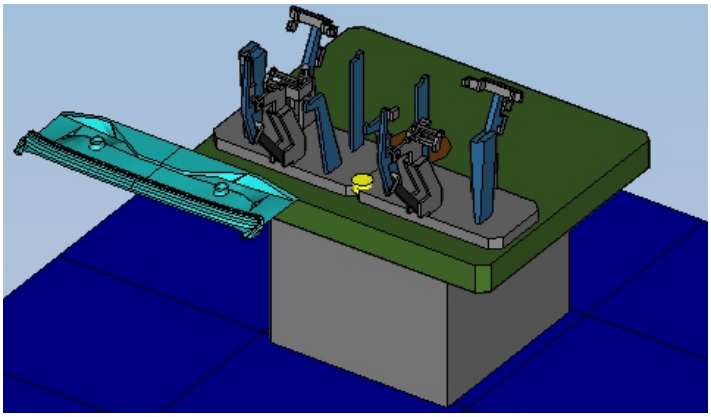
Přiložená zadání vychází ze zadání použitých v zimním semestru tohoto akademického roku (2014/2015) vytvořených panem Ing. Pavle Burgetem, Ph.D.

A.1 Task 1

Motivation

This is the first of three small tasks that aim at controlling a turn table, which receives a welding part from a human operator or from a robot, attaches it and turns with it to release it for another robot. The purpose of these three tasks is to get acquainted with the basics of Siemens Step7 programming environment, to work with the PLC Simulator and to learn the principles of logic programming in ladder diagram and later also in structured text (SCL environment).

The turn table is at position HOME and receives a welding part. The presence of the part is signalled with a sensor. After the part is present the clamps move to fix it. The clamps must be at CLOSE position to allow the turn table turning to position FWD, where the clamps are opened (moved to position OPEN) and the part moves away. Figure [A.1] shows the turn table being at HOME position. The sensor is marked in yellow and the welding part is indicated as being inserted on the turn table.



Obrázek A.1: Turn table with clamps

The behaviour of the process is simulated in Process Simulate, i.e. the input signals from the controlled process are generated by Process Simulate and the outputs are generated by your PLC program.

Assignment

The behaviour for Task 1 is as follows. Simulation state is indicated by simulationRunning PLC input signal (true - simulation in Process Simulate is being executed, false otherwise). The table movements are controlled by output signals tableToFWD (drives table to FWD position) and tableToHOME (drives table to HOME position), the current position is signalled by tableAtFWD and tableAtHOME input signals. The table can turn to FWD position only when part is fixed in the clamps (insertion of the part is controlled by weldPartToClamps output signal which must bet set until part presence is indicated by signal from a sensor). Part presence in the clamps is indicated by partInFixSensor input signal - this is the only edge-controlled signal, which means that edge detection must be performed on it. Similarly as for the table, the clamps movements are controlled by output signals clampsToOPEN and clampsToCLOSE, the current position is signalled by clampsAtOPEN and clampAtCLOSE

input signals. After the turn table reaches the FWD position (indicated by tableAtFWD) the clamps are released and the part is removed (removal of the part is controlled by weldPartOutOfClamps output signal). After that the turn table turns back to its HOME position and is prepared to receive next part.

The list of signals is shown in Table A.1. There is the symbolic name for each signal, its description, the fact if it is level or edge based, and the direction to or from the PLC.

Individual points of the assignment

1. **Create basic HW configuration and connect to PLCSim.**
   You need to enter just the CPU in the HW configuration. Use CPU 315-2 PN/DP and put it into slot 2 of the rack in HW configuration.

2. **Design a state machine that describes the behaviour of your program.**
   Draw the state machine in a graphical way, name the states and transitions. Use the same names in your program.

3. **Create the symbolic names in Symbol Table and import them to PLCSim.**
   The symbolic names created in Step7 must be imported to PLCSim. After that you can see the symbols during PLC simulation at respective addresses.

4. **Implement the state machine in ladder diagram.**
   More details about implementing state machines in PLC programs are given at the first lecture.

5. **Connect PLCSim to Process Simulate and test your program functionality.**

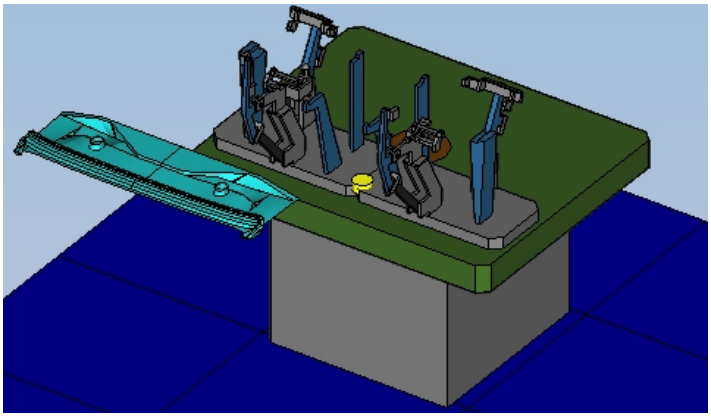
Signal Name	Data Type	Input/Output	Type
simulationRunning	BOOL	input (I)	level
tableAtHOME	BOOL	input (I)	level
tableAtFWD	BOOL	input (I)	level
clampsAtOPEN	BOOL	input (I)	level
clampsAtCLOSE	BOOL	input (I)	level
partInFixSensor	BOOL	input (I)	edge
tableToHOME	BOOL	output (Q)	level
tableToFWD	BOOL	output (Q)	level
clampsToOPEN	BOOL	output (Q)	level
clampsToCLOSE	BOOL	output (Q)	level
weldPartToClamps	BOOL	output (Q)	level
weldPartOutOfClamps	BOOL	output (Q)	level

Tabulka A.1: Table of signals

A.2 Task 2

Motivation

In this second task the focus is put on using timers and counters with the turn table, for which you created basic manipulation program in the first task



Obrázek A.2: Turn table with clamps

Similarly as with the first task, the behaviour of the process is simulated in Process Simulate, i.e. the input signals from the controlled process are generated by Process Simulate and the outputs are generated by your PLC program.

Assignment

The behaviour for Task2 is as follows. If the system is started (running simulation is indicated by simulationRunning input signal) and the turn table is at HOME position a part may arrive (insertion of the part is controlled by weldPartToClamps output signal which must bet set until part presence is indicated by signal from a sensor). Part fixing is initiated by a rising edge on ACKButton[1] input. After that the table turns to FWD position, where the clamps are released (by setting clampsToOPEN output signal). The table must wait in FWD position for at least 10 seconds (counting after falling edge on partInFixSensor input occurs) during which the weldPartOutOfClamps output signal must stay set (true) before going to HOME position.

After 10 parts are processed, the table goes to a maintenance state, where it stays for 20 seconds. During maintenance an appropriate output signal (executeMaintenance) must be set. After that the system is returned back to Operational state by a rising edge on ACKMaintenance[2].

When the STOPButton[3] is pressed the production will finish current cycle and then will be stopped until STOPButton button is released.

The list of signals is shown in Table A.4 that must be added to the symbol table from Task 1. You also have to create variable in the user space that contain initial values for the timers and counters.

[1] this PLC input signal is mapped to Key signal, which is displayed in Simulation Panel and can be pressed as a normal button, in Process Simulate

[2] again this PLC input signal is mapped to Key signal ACKMaintenance

[3] again this PLC input signal is mapped to Key signal STOPButton

Do not forget to copy the Step7 project of Task 1 to another one, which you start with for Task 2.

Signal Name	Data Type	Input/Output	Type
ACKButton	BOOL	input (I)	edge
ACKMaintentance	BOOL	input (I)	edge
STOPButton	BOOL	input (I)	level
executeMaintenance	BOOL	output (Q)	level
partsCounter	INT	output (Q)	

Tabulka A.2: Table of signals

Individual points of the assignment

1. **Design a state machine that describes the behaviour of your program**
   Draw a time diagram for the selected variables. Enhance the state machine from the first task with the functionality according to the assignment in this task. Use the same names in your program.

2. **Create the symbolic names in Symbol Table and import them to PLCSim.**
   Do not forget to create symbols for the timers and counters. Add the necessary timers and counters to the visible panels in PLCSim.

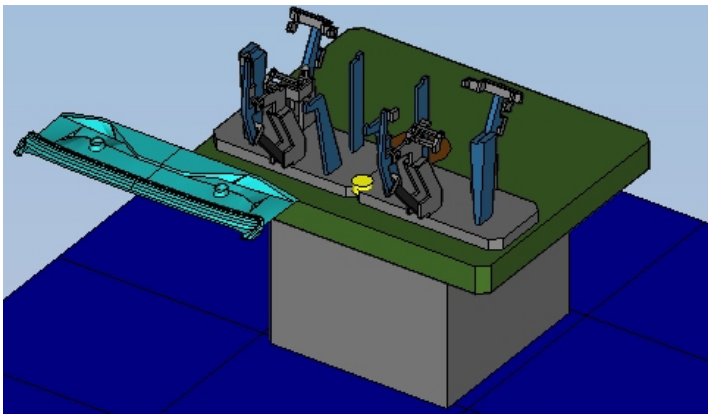
3. **Implement the state machine in ladder diagram.**
   More details about timers and counters and their use in PLC programs are given at the second lecture.

4. **Connect PLCSim to Process Simulate and test your program functionality.**

A.3 Task 3

Motivation

In this third task we extend the up-to-now functionality and concentrate on using functions and function blocks. Thanks to this, the program gets better structured. Moreover, another programming language will be used, which is SCL.



Obrázek A.3: Turn table with clamps

Similarly as with the previous tasks, the behaviour of the process is simulated in Process Simulate, i.e. the input signals from the controlled process are generated by Process Simulate and the outputs are generated by your PLC program.

Assignment

The behaviour from Task 2 remains and is extended in the following way. Function block called FB_maintenance must be added that watches the utilisation of certain parts of the system. In our case, the utilisation of the clamps and of the turn table is watched. Thus, the FB will be called several times in the PLC scan cycle, i.e. there will be one instance for each set of the clamps and one instance for the turn table. Similarly, this FB can be used for counting the parts that you did in Task 2[4]. The FB counts the number of operations and after reaching the given number, which in case of clamps operations is 10 (this number counts both closing and opening), in case of turn table movements is 8 (movement in both directions) and for counting parts the number of 10 parts remains as in Task 2, there is a maintenance period that must elapse before the system gets back to normal operation. Again - during maintenance the appropriate output signal (executeMaintenance) must be set and the system is returned back to Operational state by a rising edge on ACKMaintenance.

Design FC called FC_clampsHandling to handle the clamps, i.e. move the functionality from OB1 in Task 2 to this new FC.

Do not forget to copy the Step7 project of Task 2 to another one, which you start with for Task 3.

[4]Remember that each instance needs its own instance data block

Individual points of the assignment

1. **Design the interfaces of the FC and FB.**
   Describe inputs, outputs, static and other types of parameters where necessary. Write it in a document that will be part of the final assessment.

2. **Update the list of signals if necessary.**
   Write the list of signals in a document that will be part of the final assessment.

3. **Create the symbolic names in Symbol Table and import them to PLCSim.**
   Do not forget to create symbolic names for the FCs and FBs.

4. **Implement the blocks in the PLC and integrate them in OB1**
   Use Ladder diagram to implement the FC and SCL to implement the FB. OB1 remains in Ladder.

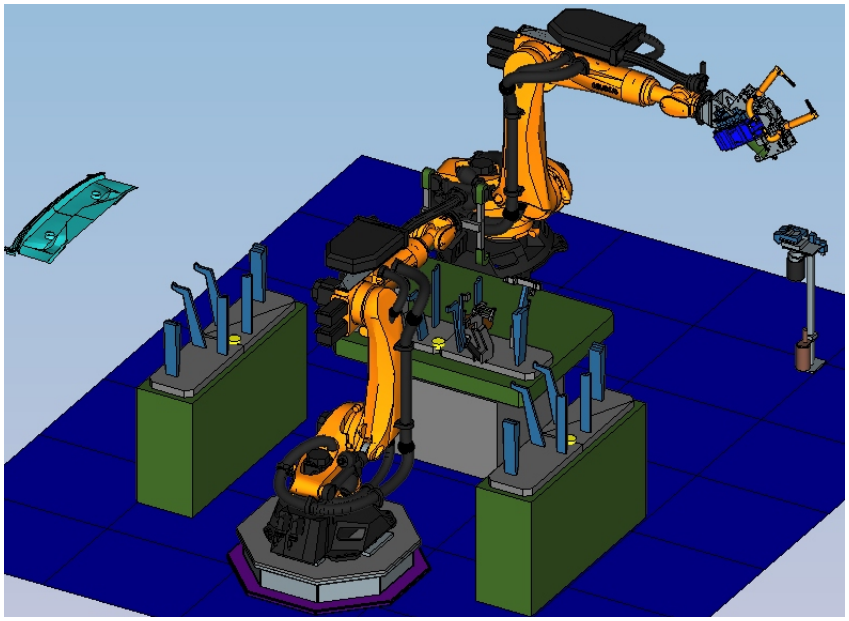
5. **Connect PLCSim to Process Simulate and test your program functionality.**

A.4 Task 1B

Motivation

This is the first of three small tasks that aim at controlling a simple robotic cell. The cell contains a turn table, which receives a welding part from a robot, attaches it and turns with it to make it available for second robot which performs welding. The purpose of these three tasks is to get acquainted with the basics of Siemens Step7 programming environment, to work with the PLC Simulator and to learn the principles of logic programming in ladder diagram and later also in structured text (SCL environment).

The cycle begins with part insertion - the part is inserted to fix1 on table1. After the part is present in fix1 a robotN1 takes the part and inserts it into the clamps, which must be in OPEN position, after the part is present in clamps the clamps move to fix it. The clamps must be at CLOSE position to alow the turn table turning to position FWD, where robotN2 performs welding. After welding the turn table moves to HOME position where the part is released then the first robot takes the part and inserts it into fix2 on table2. Finally the part can be moved out of cell. Figure [A.1] shows the turn table being at position HOME and both robots at HOME position. The sensors are marked in yellow and the part is indicated as being inserted to fix1 on table1.



Obrázek A.4: Robotic cell

The behaviour of the process is simulated in Process Simulate, i.e. the input signals from the controlled process are generated by Process Simulate and the outputs are generated by your PLC program.

Assignment

The behaviour for Task 1 is as follows. Simulation state is indicated by simulationRunning input signal (true - simulation in Process Simulate is being executed, false otherwise). The cycle begins with part insertion - the part is inserted to fix1 on table1 (insertion of the part is controlled by partToFix1 output signal which must bet set until part presence is indicated by signal from a sensor). After the part is present in fix1 (indicated by partInFix1Sensor input signal), the first robot takes it and places it to the clamps on turn table (Path #1). The table can turn to FWD position only when part is fixed in the clamps. The table movements are controlled by output signals turnTableToFWD (drives table to FWD position) and turnTableToHOME (drives table to HOME position), the current position is signalled by turnTableAtFWD and turnTableAtHOME input signals. Part presence in the clamps is indicated by partInClampsSensor input signal. Similarly as for the table, the clamps movements are controlled by output signals clampsToOPEN and clampsToCLOSE, the current position is signalled by clampsAtOPEN and clampAtCLOSE input signals. After the turn table reaches the FWD position (indicated by tableAtFWD), robotN2 performs welding (Path #1). After welding the table goes to HOME position, where the clamps are released and the part is moved to fix2 on table2 by the robotN1 (Path #2). Finally the part is leaving the cell (removal of the part is controlled by partOutOfFix2 output signal).

The list of signals is shown in Table A.3. There is the symbolic name for each signal, its description, the fact if it is level or edge based, and the direction to or from the PLC.

Individual points of the assignment

1. **Create basic HW configuration and connect to PLCSim.**
   You need to enter just the CPU in the HW configuration. Use CPU 315-2 PN/DP and put it into slot 2 of the rack in HW configuration.

2. **Design a state machine that describes the behaviour of your program.**
   Draw the state machine in a graphical way, name the states and transitions. Use the same names in your program.

3. **Create the symbolic names in Symbol Table and import them to PLCSim.**
   The symbolic names created in Step7 must be imported to PLCSim. After that you can see the symbols during PLC simulation at respective addresses.

4. **Implement the state machine in ladder diagram.**
   More details about implementing state machines in PLC programs are given at the first lecture.

5. **Connect PLCSim to Process Simulate and test your program functionality.**

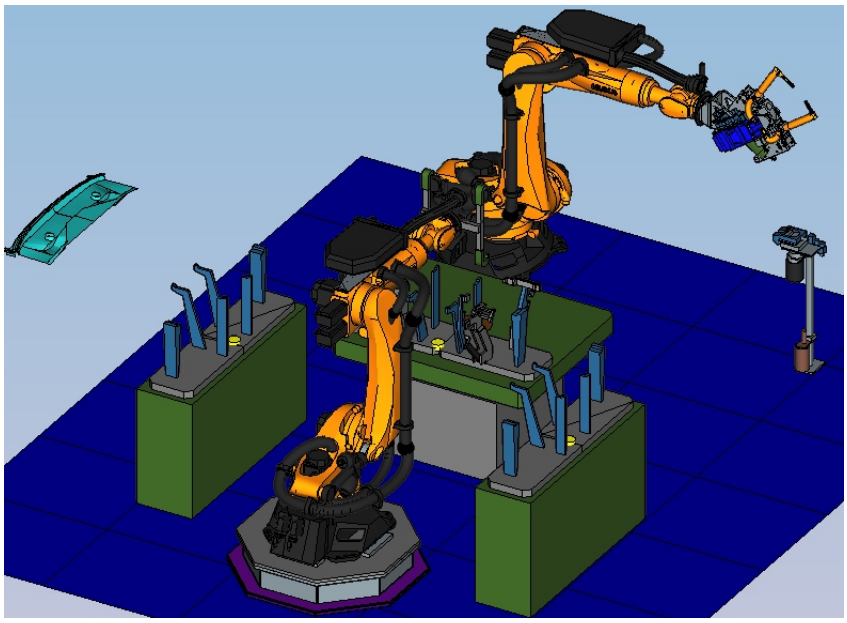
Signal Name	Data Type	Input/Output	Type
simulationRunning	BOOL	input (I)	level
turnTableAtHOME	BOOL	input (I)	level
turnTableAtFWD	BOOL	input (I)	level
clampsAtOPEN	BOOL	input (I)	level
clampsAtCLOSE	BOOL	input (I)	level
partInFix1Sensor	BOOL	input (I)	level
partInClampsSensor	BOOL	input (I)	edge
partInFix2Sensor	BOOL	input (I)	level
robotN1ProgramEnded	BOOL	input (I)	edge
robotN2ProgramEnded	BOOL	input (I)	edge
turnTableToHOME	BOOL	output (Q)	level
turnTableToFWD	BOOL	output (Q)	level
clampsToOPEN	BOOL	output (Q)	level
clampsToCLOSE	BOOL	output (Q)	level
robotN1StartProgram	BOOL	output (Q)	edge
robotN2StartProgram	BOOL	output (Q)	edge
robotN1ProgramNumber	BYTE	output (Q)	
robotN2ProgramNumber	BYTE	output (Q)	
partToFix1	BOOL	output (Q)	level
partOutOfFix2	BOOL	output (Q)	level

Tabulka A.3: Table of signals

A.5 Task 2B

Motivation

In this second task the focus is put on using timers and counters with the turn table, for which you created basic manipulation program in the first task



Obrázek A.5: Robotic cell

Similarly as with the first task, the behaviour of the process is simulated in Process Simulate, i.e. the input signals from the controlled process are generated by Process Simulate and the outputs are generated by your PLC program.

Assignment

The behaviour for Task2 is as follows. If the system is started (running simulation is indicated by simulationRunning input signal) the part may arrive. When the part is present in fix1 (insertion of the part is controlled by partToFix1 output signal which must bet set until part presence is indicated by partInFix1Sensor input signal) on table1 and the turn table is in HOME position (indicated by tableAtHOME input signal), the robotN2 takes the part and places it to the clamps (Path #1) which moves to fix it. Part presence in the clamps is indicated by partInClampsSensor input signal. After that the table turns to FWD position, where the robotN2 performs welding (Path #1). Welding is initiated by a rising edge on ACKButton[5]. After that the turn table goes back to HOME position, where the clamps are released. After the part is released the first robot takes it and places it to fix2 on table2 (Path #2). Finally the part can be moved out of cell - removal of the part is controlled by partOutOfFix2 output signal, which must be set at least 10 seconds after falling edge on partInFix2Sensor occurs.

[5]this PLC input signal is mapped to Key signal, which is displayed in Simulation Panel and can be pressed as a normal button, in Process Simulate

After 10 parts are processed, the system goes to a maintenance state, where it stays for at least 20 seconds. During maintenance an appropriate output signal (executeMaintenance) must be set and robotN1 must perform tip dressing (Path #2). The system is returned back to operational state by a rising edge on ACKMaintenance[6].

When the STOPButton[7] is pressed the production will finish current cycle and then will be stopped until STOPButton button is released.

The list of signals is shown in Table A.4 that must be added to the symbol table from Task 1. You also have to create variable in the user space that contain initial values for the timers and counters.

Signal Name	Data Type	Input/Output	Type
simulationRunning	BOOL	input (I)	level
ACKButton	BOOL	input (I)	edge
ACKMaintentance	BOOL	input (I)	edge
STOPButton	BOOL	input (I)	level
executeMaintenance	BOOL	output (Q)	edge
partsCounter	INT	output (Q)	

Tabulka A.4: Table of signals

Individual points of the assignment

1. **Design a state machine that describes the behaviour of your program**
   Draw a time diagram for the selected variables. Enhance the state machine from the first task with the functionality according to the assignment in this task. Use the same names in your program.

2. **Create the symbolic names in Symbol Table and import them to PLCSim.**
   Do not forget to create symbols for the timers and counters. Add the necessary timers and counters to the visible panels in PLCSim.

3. **Implement the state machine in ladder diagram.**
   More details about timers and counters and their use in PLC programs are given at the second lecture.

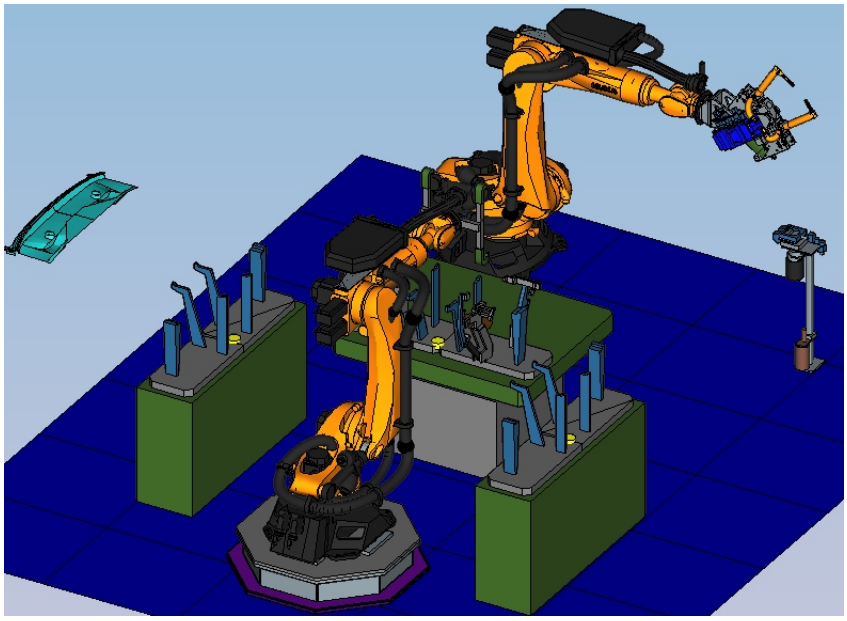
4. **Connect PLCSim to Process Simulate and test your program functionality.**

[6] again this PLC input signal is mapped to Key signal ACKMaintenance
[7] again this PLC input signal is mapped to Key signal STOPButton

A.6 Task 3B

Motivation

In this third task we extend the up-to-now functionality and concentrate on using functions and function blocks. Thanks to this, the program gets better structured. Moreover, another programming language will be used, which is SCL.



Obrázek A.6: Robotic cell

Similarly as with the previous tasks, the behaviour of the process is simulated in Process Simulate, i.e. the input signals from the controlled process are generated by Process Simulate and the outputs are generated by your PLC program.

Assignment

The behaviour from Task 2 remains and is extended in the following way. Function block called FB_maintenance must be added that watches the utilisation of certain parts of the system. In our case, the utilisation of clamps, of the turn table and of robots is watched. Thus, the FB will be called several times in the PLC scan cycle. Similarly, this FB can be used for counting the parts that you did in Task 2[8]. The FB counts the number of operations and after reaching the given number, which in case of clamps operations is 10 (this number counts both closing and opening), in case of turn table movements is 8 (movement in both directions) and for counting parts, the number of 10 parts remains as in Task 2, there is a maintenance period that must elapse before the system gets back to normal operation. Again - during maintenance the appropriate output signal (executeMaintenance) must be set and robotN1 must perform tip dressing (Path #2). The system is returned back to Operational state by a rising edge on ACKMaintenance.

Design FC called FC_robotHandling to handle the robots, i.e. move the functionality from OB1 in Task 2 to this new FC.

[8]Remember that each instance needs its own instance data block

Do not forget to copy the Step7 project of Task 2 to another one, which you start with for Task 3.

Individual points of the assignment

1. **Design the interfaces of the FC and FB.**
   Describe inputs, outputs, static and other types of parameters where necessary. Write it in a document that will be part of the final assessment.

2. **Update the list of signals if necessary.**
   Write the list of signals in a document that will be part of the final assessment.

3. **Create the symbolic names in Symbol Table and import them to PLCSim.**
   Do not forget to create symbolic names for the FCs and FBs.

4. **Implement the blocks in the PLC and integrate them in OB1**
   Use Ladder diagram to implement the FC and SCL to implement the FB. OB1 remains in Ladder.

5. **Connect PLCSim to Process Simulate and test your program functionality.**

Příloha B

Bezpečnostní program pro PLC

Tato příloha obsahuje finální verzi bezpečnostního programu a je zde přiložena zejména pro účely budoucí reference (funkčnost programu zde není detailně popisována). Bezpečnostní program je vytvořen v rámci funkčního bloku (FB), který je volán v systémovém bloku OB35.

Hlavním úkolem bezpečnostního programu je zejména bezpečné zastavení robotu na základě stisknutí tlačítka pro nouzové zastavení (Network 3) nebo na základě narušení bezpečnosti (ať už přerušení bezpečnostního optického závěsu SICK nebo narušení ochranného pole bezpečnostního laserového skeneru SICK, Network 4). Dále pak reintegrace pasivních F I/O (Network 8 a 9) a signalizace nutnosti kvitace (Network 7). Pro více informací ohledně bezpečnosti mohu odkázat na dokumentaci k Step7 Distibuted Safety, případně na diplomovu práci P. Poláka - viz [19].

Zajímavým rozšířením bezpečnostního programu by bylo využití informace o narušení varovné zóny skeneru (bitový signál dostupný přes PROFIsafe). Varovná zóna disponuje větším dosahem, než zóna ochranná a její narušení by mohlo vést ke zpomalení robotu. Tato možnost však v rámci této práce nebyla studována.

Bezpečnostní rozhraní robotu

Abychom porozuměli bezpečnostnímu programu, musíme se nejprve seznámit s bezpečnostním rozhraním KUKA robotu. Jak již bylo řečeno, robot komunikuje s nadřazeným bezpečnostním PLC přes PROFIsafe. Dochází k výměně několika bitových signálů, které jsou uvedeny v Tab. B.1 a v Tab. B.2. Adresový prostor přidělený robotu a použitý v rámci bezpečnostního programu můžeme vidět na Obr. 3.4 - bezpečnostní vstupy PLC (výstupy robotu) začínají na adrese 27 (důležité jsou první dva bajty), bezpečnostní výstupy PLC (vstupy robotu) na adrese 27 (důležité jsou rovněž první dva bajty). Tzn. například vstup robotu (výstup PLC) **NHE** (externí bezpečné zastavení) je namapován na adresu Q 27.1, výstup robotu (vstup PLC) **AUT** (indikace AUT EXT nebo AUT režimu) je namapován na adresu I 27.5 atp.

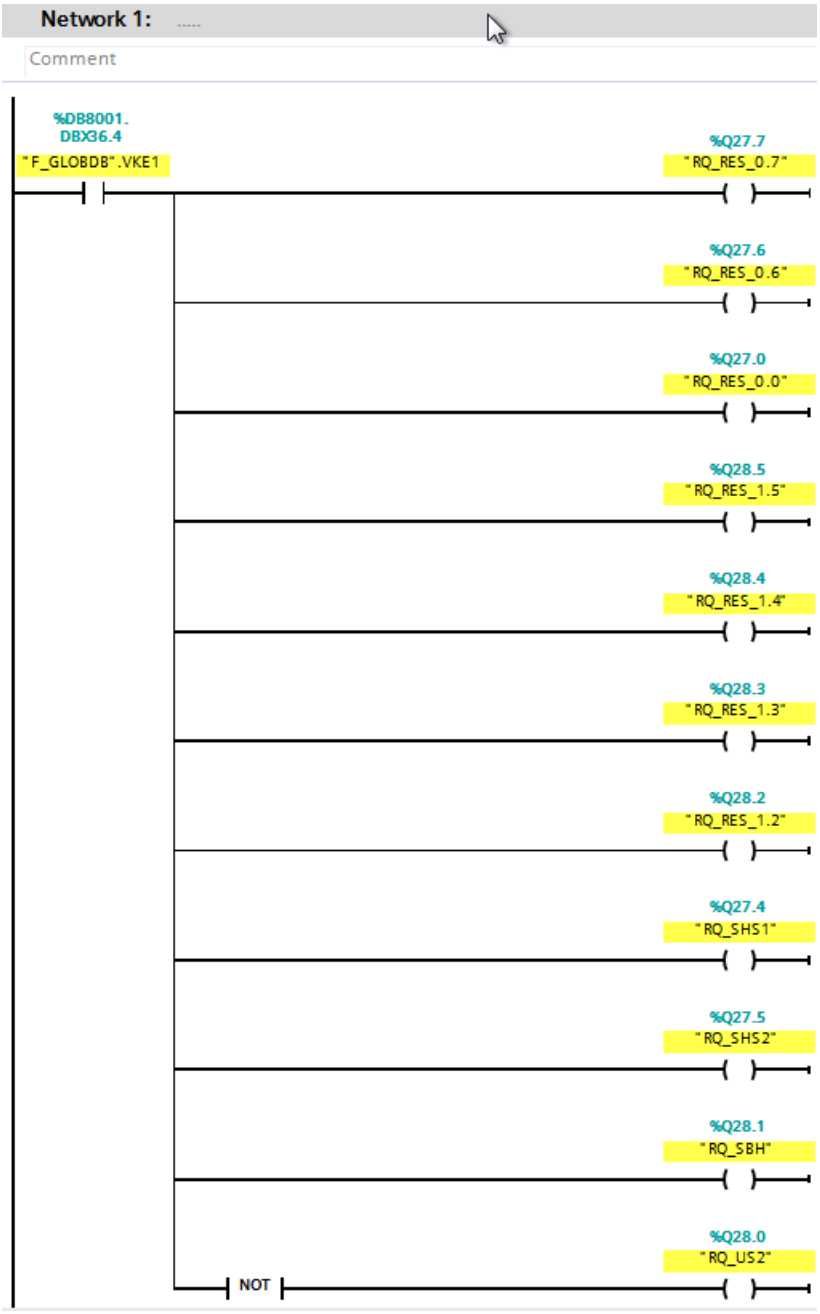
Pozn: Podrobnější informace je možno nalézt v [7].

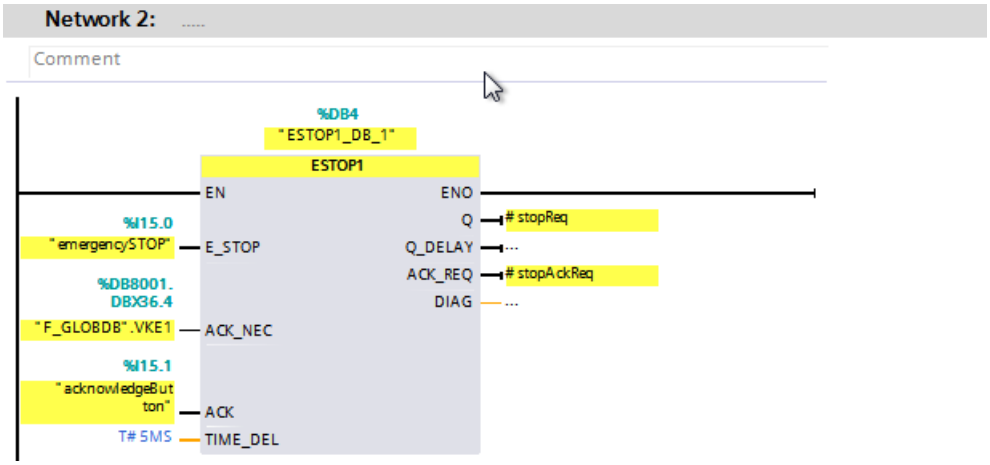
Bit	Signal	Description
0.0	RES	Reserved 1 (musí být nastaven do jedničky)
0.1	NHE	Input for external Emergency Stop (vstup pro externí bezpečné zastavení, 0 = externí E-STOP je aktivní, 1 = externí E-STOP není aktivní)
0.2	BS	Operator safety (0 = bezpečnost operátora není aktivní, 1 = bezpečnost operátora je aktivní)
0.3	QBS	Ackowledgement of operator safety (je-li nastaven signál BS, pak náběžnou hranou na tomto vstupu dojde k potvrzení bezpečnosti operátora)
0.4	SHS1	Safety STOP 1 (0 = bezpečné zastavení je aktivní, 1 = bezpečné zastavení není aktivní, **nesmí být použito pro E-STOP**)
0.5	SHS2	Safety STOP 2 (0 = bezpečné zastavení je aktivní, 1 = bezpečné zastavení není aktivní, **nesmí být použito pro E-STOP**)
0.6	RES	-
0.7	RES	-
1.0	US2	Supply voltage US2 (není-li tento vstup použit, musí být nastaven do nuly)
1.1	SBH	Safe operational stop
1.2	RES	Reserved11 (musí být nastaven do jedničky)
1.3	RES	Reserved12 (musí být nastaven do jedničky)
1.4	RES	Reserved13 (musí být nastaven do jedničky)
1.5	RES	Reserved14 (musí být nastaven do jedničky)
1.6	RES	Reserved15 (musí být nastaven do jedničky)
1.7	SPA	Shutdown PROFIsafe Acknowledge

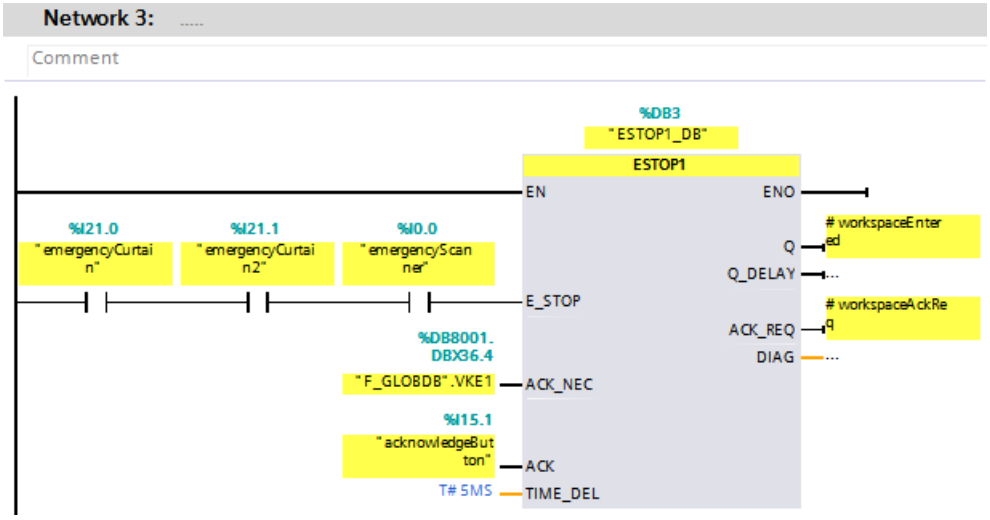
Tabulka B.1: Vstupy robotu (výstupy PLC)

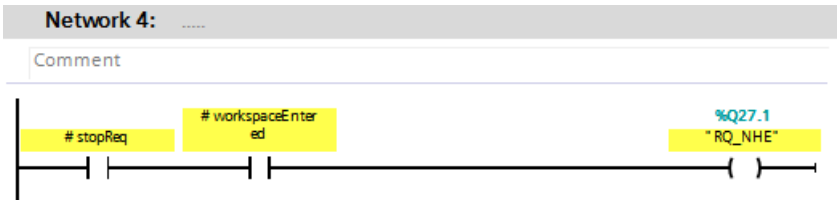
Bit	Signal	Description
0.0	NHL	Local E-STOP (0 = lokální E-STOP je aktivní, 1 = lokální E-STOP není aktivní)
0.1	AF	Drives enable (0 = drives enable není aktivní, 1 = drives enable je aktivní)
0.2	FF	Motion enable (interní bezpečnostní kontrolér v KRC povolil pohyb motorů, 0 = motion enable není aktivní, 1 = motion enable je aktivní)
0.3	ZS	One of the enabling switches is in the center position
0.4	PE	Peri Enabled
0.5	AUT	The manipulator is in AUT or AUT EXT mode (0 = AUT nebo AUT EXT režim je aktivní, 1 = AUT nebo AUT EXT režim není aktivní)
0.6	T1	The manipulator is in Manual Reduced Velocity mode (0 = T1 režim je aktivní, 1 = T1 režim není aktivní)
0.7	T2	The manipulator is in Manual High Velocity mode. (0 = T2 režim je aktivní, 1 = T2 režim není aktivní)
1.0	NHE	External E-STOP has been triggered (0 = externí E-STOP je aktivní, 1 = externí E-STOP není aktivní)
1.1	BS	Operator safety (0 = bezpečnost operátora není zajištěna, 1 = bezpečnost operátora je zajištěna)
1.2	SHS1	Safety stop 1 (0 = bezpečné zastavení stop 1 není aktivní, 1 = bezpečné zastavení stop 1 je aktivní)
1.3	SHS2	Safety stop 2 (0 = bezpečné zastavení stop 2 není aktivní, 1 = bezpečné zastavení stop 2 je aktivní)
1.4	RES	Reserved 13
1.5	RES	Reserved 14
1.6	PSA	PROFIsafe active (0 = PROFIsafe není aktivní, 1 = PROFIsafe je aktivní)
1.7	SP	Shutdown PROFIsafe

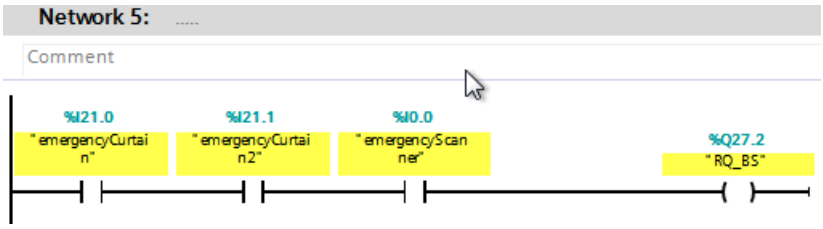
Tabulka B.2: Výstupy robotu (vstupy PLC)

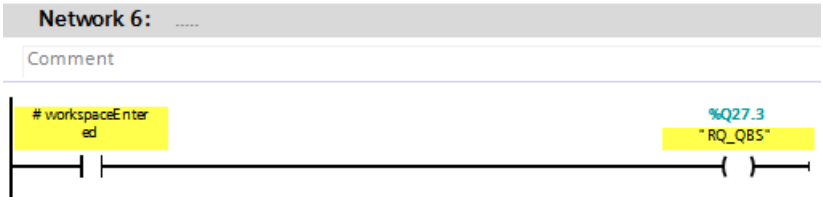

Network 1:
Comment
%DB8001.
DBX36.4
"F_GLOBDB".VKE1
%Q27.7
"RQ_RES_0.7"
%Q27.6
"RQ_RES_0.6"
%Q27.0
"RQ_RES_0.0"
%Q28.5
"RQ_RES_1.5"
%Q28.4
"RQ_RES_1.4"
%Q28.3
"RQ_RES_1.3"
%Q28.2
"RQ_RES_1.2"
%Q27.4
"RQ_SHS1"
%Q27.5
"RQ_SHS2"
%Q28.1
"RQ_SBH"
%Q28.0
"RQ_US2"
NOT

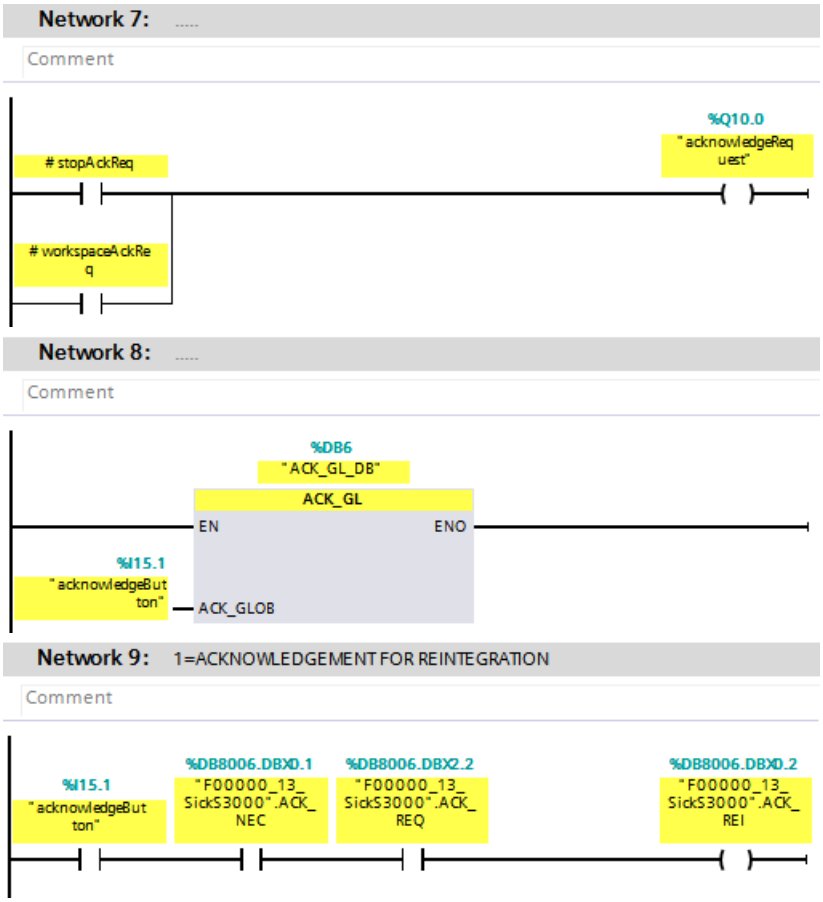

Network 2:
Comment
%DB4
"ESTOP1_DB_1"
ESTOP1
EN
ENO
%I5.0
"emergencySTOP"
E_STOP
Q
stopReq
Q_DELAY
ACK_REQ
stopAckReq
%DB8001.
DBX36.4
"F_GLOBDB".VKE1
ACK_NEC
DIAG
%I5.1
"acknowledgeBut
ton"
ACK
T# 5MS
TIME_DEL


Network 3:
Comment
%DB3
"ESTOP1_DB"
ESTOP1
EN
ENO
%I1.0
"emergencyCurtai
n"
%I1.1
"emergencyCurtai
n2"
%I0.0
"emergencyScan
ner"
Q
workspaceEnter
ed
Q_DELAY
E_STOP
ACK_REQ
workspaceAckRe
q
%DB8001.
DBX36.4
DIAG
"F_GLOBDB".VKE1
ACK_NEC
%I5.1
"acknowledgeBut
ton"
ACK
T# 5MS
TIME_DEL


Network 4:
Comment
stopReq
workspaceEnter
ed
%Q27.1
"RQ_NHE"


Network 5:
Comment
%I1.0
"emergencyCurtai
n"
%I1.1
"emergencyCurtai
n2"
%I0.0
"emergencyScan
ner"
%Q27.2
"RQ_BS"


Network 6:
Comment
workspaceEnter
ed
%Q27.3
"RQ_QBS"


Network 7:
Comment
%Q10.0
"acknowledgeRequest"
stopAckReq
workspaceAckReq
Network 8:
Comment
%DB6
"ACK_GL_DB"
ACK_GL
EN
ENO
%I15.1
"acknowledgeButton"
ACK_GLOB
Network 9: 1=ACKNOWLEDGEMENT FOR REINTEGRATION
Comment
%I15.1
"acknowledgeButton"
%DB8006.DBX0.1
"F00000_13_SickS3000".ACK_NEC
%DB8006.DBX2.2
"F00000_13_SickS3000".ACK_REQ
%DB8006.DBX0.2
"F00000_13_SickS3000".ACK_REI

Příloha C

Obsah přiloženého CD

simekpa5DP.pdf - text diplomové práce