



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**

BRNO UNIVERSITY OF TECHNOLOGY

**FAKULTA STROJNÍHO INŽENÝRSTVÍ**

FACULTY OF MECHANICAL ENGINEERING

**ÚSTAV VÝROBNÍCH STROJŮ, SYSTÉMŮ A ROBOTIKY**

INSTITUTE OF PRODUCTION MACHINES, SYSTEMS AND ROBOTICS

**DIGITÁLNÍ ZPROVOZNĚNÍ ROBOTIZOVANÉHO  
VÝROBNÍHO SYSTÉMU PRO BODOVÉ SVAŘOVÁNÍ**

DIGITAL COMMISSIONING OF A ROBOTIC PRODUCTION SYSTEM FOR SPOT WELDING

**DIPLOMOVÁ PRÁCE**

MASTER'S THESIS

**AUTOR PRÁCE**

AUTHOR

**Bc. Antonín Hradil**

**VEDOUCÍ PRÁCE**

SUPERVISOR

**Ing. Jan Vetiška, Ph.D.**

**BRNO 2023**



# Zadání diplomové práce

Ústav:	Ústav výrobních strojů, systémů a robotiky
Student:	<b>Bc. Antonín Hradil</b>
Studijní program:	Výrobní stroje, systémy a roboty
Studijní obor:	bez specializace
Vedoucí práce:	<b>Ing. Jan Vetiška, Ph.D.</b>
Akademický rok:	2022/23

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

## **Digitální zprovoznění robotizovaného výrobního systému pro bodové svařování**

### **Stručná charakteristika problematiky úkolu:**

Stále aktuálním cílem průmyslu je zvyšování produktivity práce a jakosti výroby. Možností jak tohoto cíle dosáhnout je robotizace rutinních činností. Tato práce se zaměřuje na virtuální zprovoznění pracoviště bodového svařování plechových dílů.

Cílem práce je zjistit aktuální stav v oblasti virtuálního zprovoznění robotizovaných svařovacích pracovišť a provést takovéto zprovoznění pro daný případ.

### **Cíle diplomové práce:**

Rešerše.

Rozbor a popis pracoviště.

Propojení simulace v Process Simulation a PLC

Simulace.

### **Seznam doporučené literatury:**

SICILIANO, Bruno a Oussama. KHATIB. Springer handbook of robotics. Berlin: Springer, c2008. ISBN 978-3-540-23957-4

HORNBERG, Alexander. Handbook of Machine Vision. 1. Weinheim: WILEY-VCH Verlag GmbH & Co. KGaA, 2006. ISBN 978-3-527-40584-8

NOF, S. Y. Springer Handbook of Automation. Springer, 2009. 1812 s. ISBN 978-3-540-78830-0

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2022/23

V Brně, dne

L. S.

---

doc. Ing. Petr Blecha, Ph.D.  
ředitel ústavu

---

doc. Ing. Jiří Hlinka, Ph.D.  
děkan fakulty

## **ABSTRAKT**

Tato práce se zabývá virtuálním zprovozněním poloautomatického robotického pracoviště, určeného k bodovému svařování. Vstupním bodem práce byl virtuální model výrobního systému, který byl využíván pouze k off-line testování robotických operací. Cílem této práce je popsat fungování zadaného pracoviště, a poté jej virtuálně oživit, jak v rovině simulační, tak z pohledu řídicího kontroléru PLC.

## **ABSTRACT**

This thesis deals with the virtual commissioning of a semi-automatic robotic workstation designed for spot welding. The starting point of the work was a virtual model of the production system, which was used only for off-line testing of robotic operations. The aim of this work was to describe the functionality of the specified workstation, and then to bring it virtually to life both in the simulation plane and from the perspective of the PLC controller.

## **KLÍČOVÁ SLOVA**

Virtuální zprovoznění, TIA Portal, Process Simulate, Tecnomatix, PLCSIM Advanced, Frekvenční měnič, Robotické pracoviště pro bodové svařování

## **KEYWORDS**

Virtual commissioning, TIA Portal, Process Simulate, Tecnomatix, PLCSIM Advanced, Variable Frequency Drive, Robotic Workstation for spot welding



## **BIBLIOGRAFICKÁ CITACE**

HRADIL, Antonín. *Digitální zprovoznění robotizovaného výrobního systému pro bodové svařování* [online]. Brno, 2023 [cit. 2023-05-24]. Dostupné z: <https://www.vut.cz/studenti/zav-prace/detail/149867>. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav výrobních strojů, systémů a robotiky. Vedoucí práce Jan Vetiška.





## **PODĚKOVÁNÍ**

Rád bych poděkoval vedoucímu diplomové práce Ing. Janu Vetiškovi, Ph.D. za veškerou pomoc a rady při zpracování této práce.

Dále bych chtěl především poděkovat rodinně za neutuchající podporu, pomoc a pevné nervy.



## **ČESTNÉ PROHLÁŠ ENÍ**

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením Ing. Jana Vetišky, PhD. a s použitím literatury uvedené v seznamu.

V Brně dne 24.05.2023

.....

Hradil Antonín



# OBSAH

<b>1</b>	<b>ÚVOD</b> .....	<b>15</b>
<b>2</b>	<b>MOTIVACE</b> .....	<b>17</b>
<b>3</b>	<b>PŘEHLED SOUČASNÉHO STAVU POZNÁNÍ</b> .....	<b>19</b>
3.1	Virtuální zprovoznění .....	19
3.2	Bodové svařování .....	19
3.3	Průmyslové roboty .....	20
3.3.1	Definování souřadného systému a pohybových instrukcí robotu.....	23
3.3.2	Programování .....	26
3.4	Tecnomatix Process Simulate.....	27
3.5	PLC (programing logic controller) .....	28
3.5.1	Programování PLC .....	31
3.5.2	Komunikace.....	32
3.5.3	Výrobci.....	32
3.6	TIA Portal.....	33
3.7	Komunikační protokoly .....	33
3.7.1	Protokoly u drátového přenosu.....	33
3.7.2	Protokoly u bezdrátového přenosu .....	33
3.8	Frekvenční měnič (Variable Frequency Drive / VFD) .....	34
<b>4</b>	<b>ROZBOR ZADANÉHO PRACOVÍŠTĚ</b> .....	<b>37</b>
4.1	Blokový diagram obecný .....	38
4.2	Blokový diagram robotické buňky .....	38
4.3	Blokový diagram upínacího zařízení/otočného stolu.....	39
4.4	Definování nebezpečných prostor .....	39
4.5	Plánovaný průběh pracovního cyklu.....	40
4.6	Roboty .....	42
4.7	Rotační upínací zařízení .....	44
4.8	Process Simulate – Object tree .....	48
4.9	Strom operaci / Operation tree .....	49
4.9.1	Zakládání/vyjímání/svařování - blatník .....	50
4.9.2	Zakládání/vyjímání/svařování - šachta .....	51
4.9.3	Kontrolní operace .....	51
4.9.4	Simulace.....	52
4.10	Bodové svařování – body svaru.....	53
4.11	Signály .....	54
4.12	Další / Sensorika.....	54
<b>5</b>	<b>VÝBĚR VYUŽITÝCH KOMPONENT</b> .....	<b>59</b>
5.1	Typ PLC .....	59
5.2	Frekvenční měniče.....	59
5.3	PLC program .....	60
<b>6</b>	<b>VIRTUÁLNÍ PRACOVÍŠTĚ</b> .....	<b>61</b>
6.1	Bezpečnostní dveře hlavní .....	61
6.2	Bezpečnostní dveře boční.....	61
6.3	Rotační upínací zařízení.....	62
6.3.1	Upínací prvky .....	62
6.3.2	Senzory upínacího prostoru.....	65

6.4	Roboty .....	65
6.4.1	Robotické operace.....	66
6.4.2	Signálová struktura .....	67
6.5	Simulace a propojení s PLC .....	69
<b>7</b>	<b>VIRTUÁLNÍ KONTROLÉR.....</b>	<b>70</b>
7.1	PLC moduly.....	71
7.2	Převod signálů do TIA Portálu .....	71
7.3	Organizace signálů.....	73
7.4	FB Převod proměnných.....	75
7.5	Řízení motorů rotačních stolů.....	76
7.5.1	Frekvenční měniče.....	76
7.5.2	Technologické objekty / Technology objects.....	77
7.5.3	Motion Control .....	78
7.6	FB Operátor .....	79
7.6.1	Šachta.....	79
7.6.2	Blatník.....	82
7.7	FB Roboty .....	83
7.7.1	Šachta.....	83
7.7.2	Blatník.....	86
7.8	FB Inicializace .....	87
7.9	FB_CONTROL.....	89
7.10	Main program PLC .....	90
7.11	HMI.....	90
7.11.1	Automat.....	91
7.11.2	Manual.....	93
7.12	Robotický program .....	94
<b>8</b>	<b>ZHODNOCENÍ A DISKUZE .....</b>	<b>99</b>
<b>9</b>	<b>ZÁVĚR.....</b>	<b>101</b>
<b>10</b>	<b>SEZNAM POUŽITÝCH ZDROJŮ .....</b>	<b>103</b>
10.1	Zdroje použité literatury .....	103
10.2	Zdroje obrázků.....	105
<b>11</b>	<b>SEZNAM ZKRATEK, SYMBOLŮ, OBRÁZKŮ A TABULEK.....</b>	<b>109</b>
11.1	Seznam zkratk .....	109
11.2	Seznam tabulek.....	110
11.3	Seznam obrázků.....	110

# 1 ÚVOD

V současné době se globálně setkáváme s nedostatkem personálu ve všech odvětvích průmyslu. Z toho důvodu se klade stále vyšší důraz na vývoj technologií umožňujících využívat plně automatizovaný provoz s minimálním množstvím obslužného personálu. Takový provoz je většinou tvořen multifunkčními roboty a manipulátory.

Kromě návrhu samotné výrobní linky je také tlak zákazníků na jeho výrobní pružnost. Jelikož se technologie rychle vyvíjí, je nutné relativně často upravovat výrobní buňky dle nového typu vstupního polotovaru. Aby byl tento proces co nejrychlejší, nejefektivnější a nejlépe předem otestovaný, je snaha vytvářet 3D virtuální modely, které téměř dokonale simulují reálné pracoviště, na nichž je možné vše vyzkoušet a doladit.

Tato práce se zabývá virtuálním zprovozněním poloautomatického robotického pracoviště, určeného k bodovému svařování. Vstupním bodem práce byl virtuální model výrobního systému, který byl využíván pouze k off-line testování robotických operací. Cílem této práce je popsat fungování zadaného pracoviště, a poté jej virtuálně oživit, jak v rovině simulační, tak z pohledu řídicího kontroléru PLC.

V rešeršní části práce proběhne seznámení se současným stavem virtuálního zprovoznění a prvků, které se k němu váží. Následně bude proveden rozbor jednotlivých článků pracoviště, a tím bude zajištěn komplexní pohled na fungování pracoviště.

Poté bude možné provést samotné virtuální zprovoznění pracoviště (všech signálových struktur a pohybových animací) i kontroléru (řídicího PLC a kontrolního HMI obsluhy). K virtuálnímu zprovoznění budou v práci využity softwary firmy Siemens, Process Simulate (pracoviště) a TIA Portal (kontrolér), které budou po vzájemném propojení simulovat chování reálného pracoviště.





## 2 MOTIVACE

S rychle se měnícími trendy v průmyslu se můžeme velmi často setkat s požadavkem na úpravu provozovaného pracoviště, určeného ke zpracování jednoho typu polotovaru tak, abychom mohli v co nejkratším čase zpracovávat polotovar jiný.

Virtuálním zprovozněním původně pouze fyzického robotického pracoviště získáváme v první řadě obrovskou výhodu flexibility pracoviště (schopnost rychle se adaptovat a měnit) a šetříme čas nutný k zavádění linky do provozu.

Při konvenční změně se můžeme potýkat se situacemi, kdy nové fyzické prvky neumožní provádět žádané operace (např. robotické) z důvodu chybně uvažovaných rozměrů součástí. Dále je také nutné ladit řídicí program přímo na místě, např. z důvodu neočekávaných vnějších vlivů na pracovišti. Při testování fyzických komponent a řídicího programu přímo ve virtuálně zprovozněném modelu jsme schopni se celé řadě běžně nepředvídatelných komplikací vyhnout, či jejich možnost výskytu eliminovat. I přesto, že virtuální zprovoznění zabere dost času, jsme díky němu schopni značně ušetřit čas zavádění do provozu, v celkovém měřítku i celkový čas realizace projektu, a tím ušetříme i značné finanční náklady.



## 3 PŘEHLED SOUČASNÉHO STAVU POZNÁNÍ

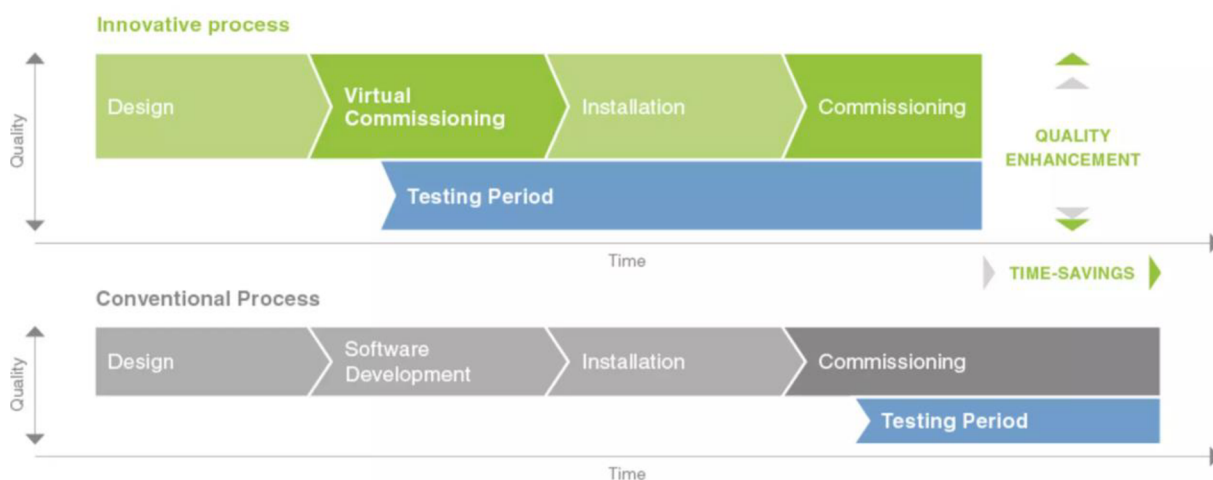
### 3.1 Virtuální zprovoznění

V současné době se setkáváme s moderními technologiemi na každém kroku a výjimkou není ani běžně stereotypní průmysl. Nedostatek personálu, nejistota a nespolehlivost nutí všechny světové firmy přecházet k plně automatizovaným výrobním linkám a k jejich využití ve svých výrobních závodech.

Proces návrhu jakéhokoliv automatizovaného pracoviště je v zásadě rozdělen na několik fází. Práce se zaměřuje na fázi poslední – uvedení do provozu. Tato část se standardně odhaduje na cca 25 % celkového času procesu, ale často se může výrazně zvýšit například z důvodu špatně navrženého softwaru. S vývojem technologií se snažíme v současné době rapidně urychlit fázi uvedení do provozu a díky virtuálnímu zprovoznění minimalizovat chybovost a rizika.

Jedná se o virtuální simulaci odpovídající reálnému fungování linky. Tradiční zprovoznění vyžaduje testování programů a prvků linky fyzicky u stroje. Nevýhodou jsou stísněné prostory, špatná viditelnost a nebezpečí škod vlivem nepozornosti. Výhodou virtuálního zprovoznění je testování a ladění programu PLC (Programmable Logic Controller), robotického programu a všech signálových struktur.

Uvedení do provozu s sebou nese také obrovskou výhodu pružnosti celého pracoviště. Při plánování malých či velkých změn linky je možné předpřipravit a doladit všechny potřebné programy a prvky pracoviště paralelně s chodem aktuální linky. Díky tomu se minimalizují prostroje všech prvků linky a tím i finanční náklady.



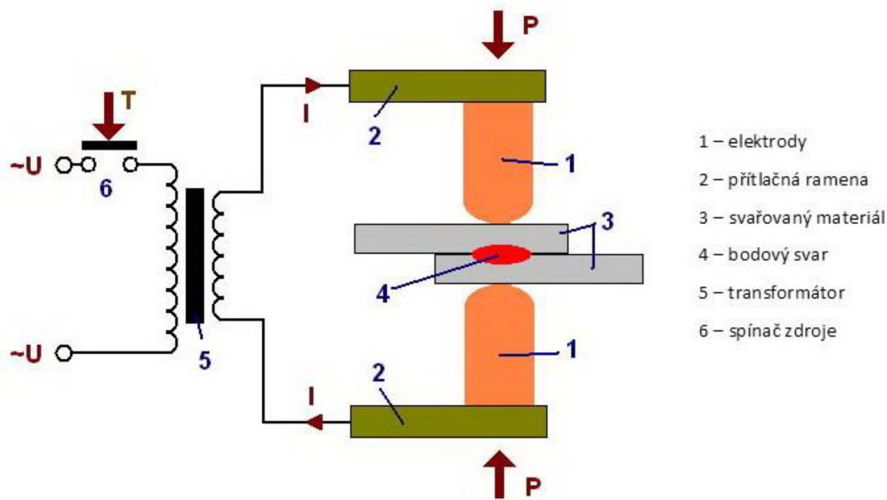
Obr. 1 - Moderní vs. konvenční proces návrhu pracoviště [1]

### 3.2 Bodové svařování

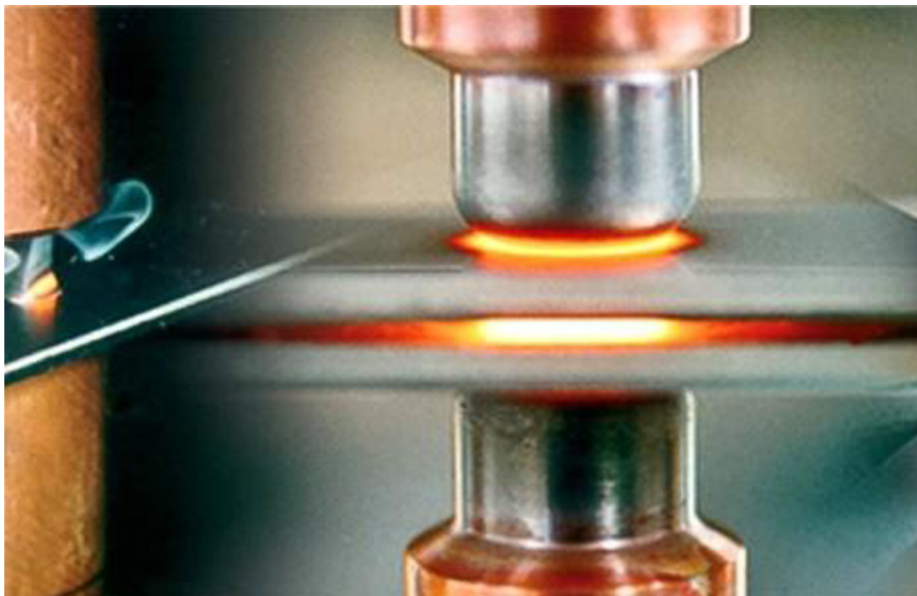
Svařování je jedna z nejdůležitějších technologií využívaných v průmyslu. V praxi se pak nejčastěji setkáváme s bodovým svařováním.

Jedná se o odporové svařování přeplátováním. Využití nachází převážně u spojování tenkých plechů a funguje velmi jednoduše. Na požadované místo svaru přimáčkneme dvě elektrody připojené k napájecímu napětí. Elektrody jsou nejčastěji tvořeny z mědi s vysokou

vodivosti elektrického proudu. Při nárazovém průchodu velkého proudu skrze plechy s menší vodivostí (vysokým odporem) dochází k okamžitému nárůstu teploty. Částečným natavením plechu a vlivem působení úchopného tlaku, dojde k lokálnímu tepelnému spojení plechů svařením.



Obr. 2 – Princip bodového svařování [2]



Obr. 3 – Detail reálného bodového svařování [3]

Výhody bodového svařování:

- Pouze lokální teplotní změna, minimální ztráty a účinky tepla v tělese
- Rychlost procesu
- Nízké náklady

### 3.3 Průmyslové roboty

Průmyslové roboty si v posledních dekádách nachází vyšší a vyšší uplatnění. Důvodem je opakovatelnost, přesnost, rychlost, stabilita, flexibilita, nedostatek operátorů, a v neposlední řadě možnost pracovat v non-stop provozu. Nevýhodou je zaváděcí cena, odbornost obsluhy, závislost na zdroji elektrické energie a odstávka při poruše.

Výběr robota závisí na typu výrobního systému, požadovaných operacích robota, potřebné nosnosti a množství požadovaných cyklů.

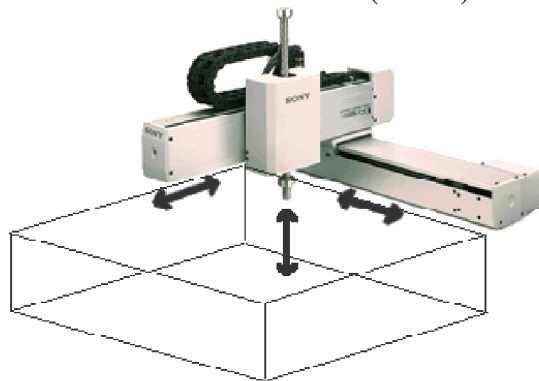
Robot je možné osadit neomezeným množstvím koncových efektorů v závislosti na typu vykonávaných operací. Také může mít tzv. dual-tool, kdy jsou na robota například umístěny dva koncové efektor, odkloněny o 45°. Takové nástroje se běžně vyskytují u operací, kdy robot zaměňuje hotový díl za polotovár v jedné operaci a urychluje tak takt výroby.



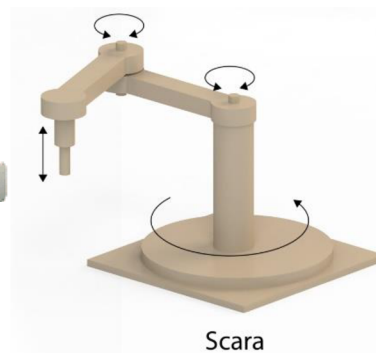
Obr. 4 – Robot osazen s dual-tool [4]

Rozdělení podle konstrukce:

- Karteziánské ( $T_x$ ,  $T_y$ ,  $T_z$ )
  - SCARA (1T 3R)



a)



b)

Obr. 5 – a) Karteziánský robot, b) SCARA robot [5] [6]

- Kloubové (0T)
- Dvouramenné
- Šestiosé (6R)



a)



b)

Obr. 6 – a) Šestiosý robot, b) Dvouramenný robot [7] [8]

- Delta (3R)



Obr. 7 – Delta robot [9]

Rozdělení dle využití:

- Svařovací
- Lakovací
- Paletizační
- Montážní

Mezi nejpoužívanější typ robotů patří šestiosý, SCARA nebo delta, a nachází uplatnění ve všech odvětvích, například v průmyslu, potravinářství, farmacii aj.

Největší výrobci průmyslových robotů:

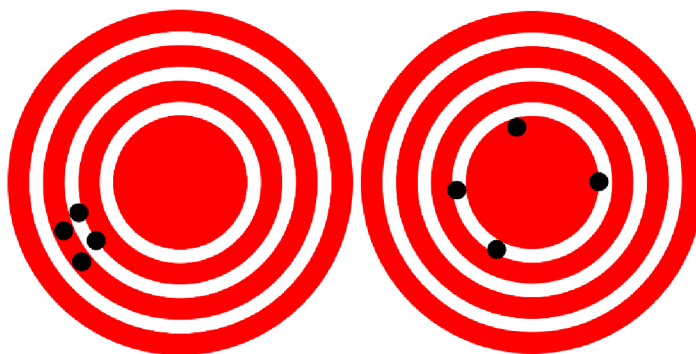
- FANUC
- ABB
- Kawasaki Robotics
- KUKA
- Yaskawa

Nejdůležitější parametry:

- Počet os

- Kinematika
- Dosah
- Nosnost
- Dynamika
- Opakovatelnost
- Přesnost

Přesnost a opakovatelnost jsou dvě rozdílné vlastnosti, ale obě jsou pro praktické využití robotů klíčové. Při vysoké opakovatelnosti a nízké přesnosti získáváme trvalou přesnou odchylku od žádané polohy. Obráceně získáváme přesnou polohu, ale s odchylkou v rádiu kolem žádané hodnoty. Obě hodnoty se u robotů měří za dodržení normy ISO 9283.



Obr. 8 – a) Vysoká přesnost, b) Vysoká opakovatelnost [10] [11]

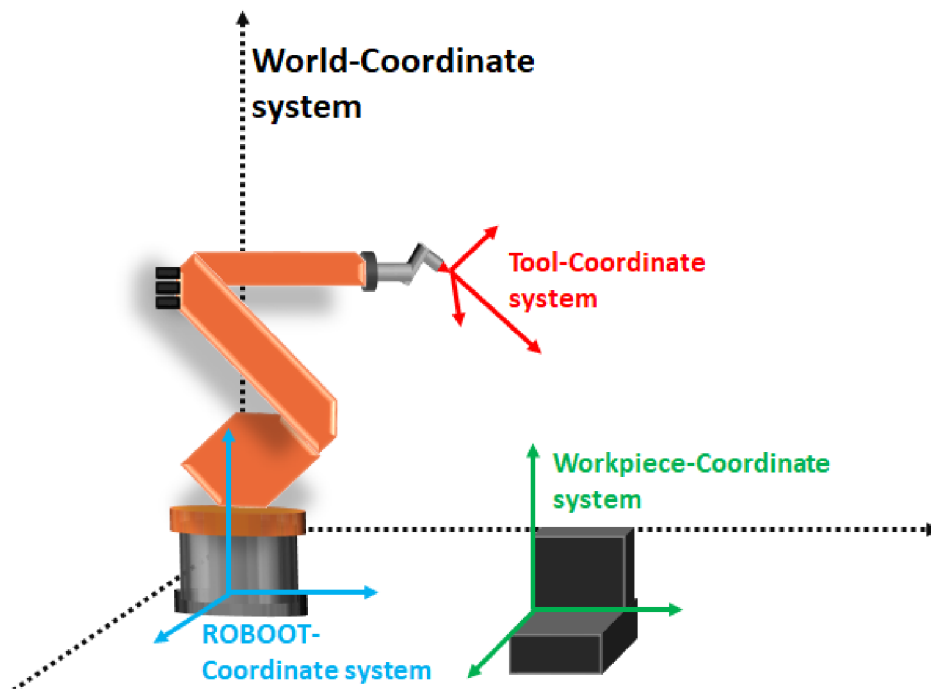
V malých jednoduchých aplikacích s cyklicky opakujícími se operacemi může robot fungovat i bez vyššího ovládání formou PLC. Při nasazení do složitějších linek jsou roboty řízeny skrze PLC, obsahující všechny potřebné signály a řídicí programy celé linky.

### 3.3.1 Definování souřadného systému a pohybových instrukcí robotu

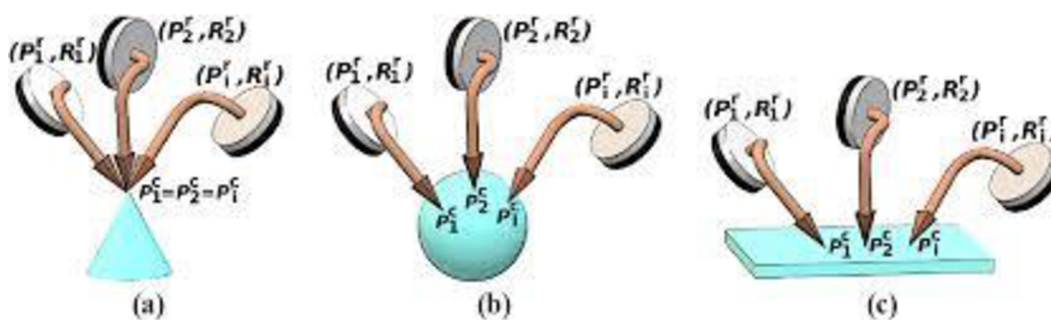
Prvním krokem před programováním je kontrola **souřadných systémů (SS)**. Nejdůležitějšími jsou Tool Center Point (TCP/Pracovní bod nástroje), World Coordinate System (WCS/Systém svět), Base Coordinate System (BCS/Systém báze robotu) a World Object Frame (WOF/Systém objektu).

WCS a BCS jsou standardně dané od výrobce, WCS je pak možné měnit. TCP je nejdůležitější souřadnicový systém pro výpočetní hardware robotu, zajišťující správné chování inverzní kinematiky. Pokud nemá robot připojený koncový efektor, nachází se TCP ve středu příruby robotu. Po mechanickém zapojení efektoru je nutné zadat údaje efektoru do kontroléru. Při použití efektoru na zakázku od externí firmy známe předběžnou hodnotu TCP, ale v každém případě se doporučuje provést zaměření TCP přímo na místě instalace pomocí jedné z ověřených metod.

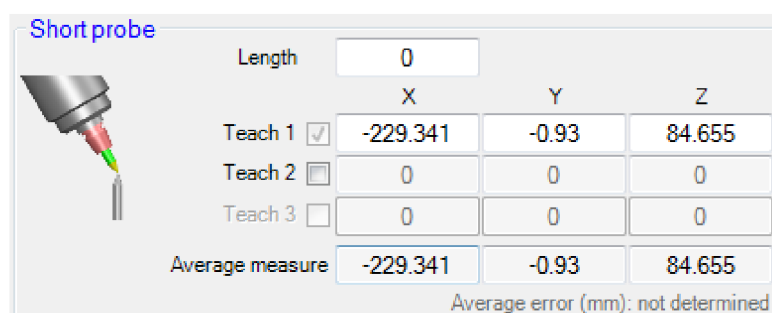
První z metod je X-bodová. Jedná se o manuální optickou metodu, kdy koncový efektor najíždí z různých uhlů k testovacímu ostnu. Minimum nutných bodů k zaměření jsou tři, čím více jich ovšem využijeme, tím získáváme přesnější hodnotu TCP. Jedná se o levnou metodu určenou k zaměření robota s nízkými požadavky na přesnost.



Obr. 9 – Souřadné systémy robotu [12]



Obr. 10 – X-bodové metody zaměření TCP [13]



Obr. 11 – X-bodová metoda v programu ABB RobotStudio [14]

Druhou metodou je kalibrace pomocí laserů. Jedná se o automatickou, velmi přesnou metodu, určenou jak k zaměřování, tak i ke kontrole aktuálního stavu TCP. Nevýhodou je vysoká cena.





a)

b)

Obr. 12 – a) Princip laserového zam. TCP, b) Reálný stojan zaměření TCP [15] [16]

Dalšími velmi užitečnými souřadnými systémy jsou WCS (SS součásti) a UCS (uživatelský SS). Každá součást má jak ve virtuálním zprovoznění, tak v reálném světě svůj SS. Díky němu můžeme propojit veškeré operace spojené s touto součástí s jejím SS, a při jakékoli změně rozložení pracoviště stačí obsluze (nebo strojovému vidění) znovu zaměřit pouze jeden bod (SS) a všechny body se přesunou s ním.

Podobným způsobem funguje UCS. Nejčastěji se zaměřuje roh stolu nebo upínací plochy, a při změně polohy dojde k přeskupení všech operací a bodů spojených s tímto centrálním bodem.

```

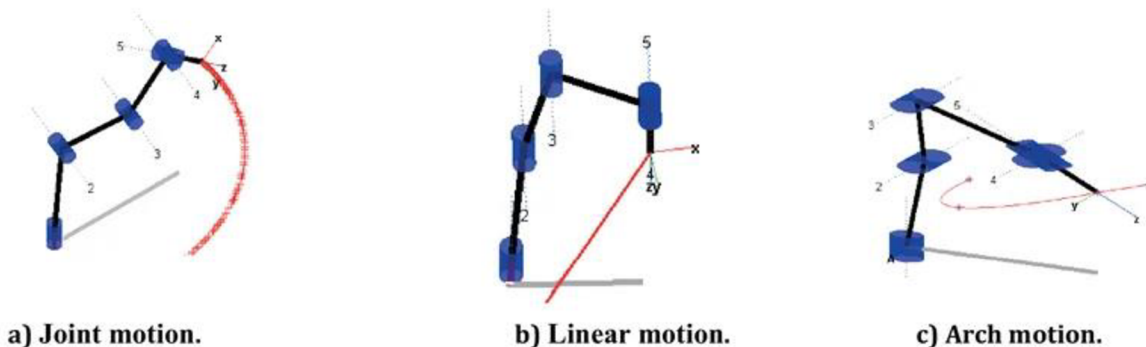
!# -----
!# ----- TOOL DATA
!# -----
PERS tooldata gRIPPER:=[TRUE,[[0,0,99],[1,0,0,0]], [5,[-24.94,-24,94.67],[1,1,0,0],0,0,0]];

!# -----
!# ----- WOBJ DATA
!# -----
PERS wobjdata Wobj_dopr_vys:=[FALSE,TRUE,"",[[ -424.69,-337.53,185.98],[0.707107,0.000006,-0.000006,0.707107]],[[0,0,0],[1,0,0,0]]];
PERS wobjdata Wobj_mer_stan:=[FALSE,TRUE,"",[[475.27,-62.99,-14],[1,0.000005,0,0]],[[0,0,0],[1,0,0,0]]];
PERS wobjdata Wobj_dopr_niz:=[FALSE,TRUE,"",[[ -359.1,134.36,48.5],[0.707107,0.000006,-0.000006,0.707107]],[[0,0,0],[1,0,0,0]]];
    
```

Obr. 13 – Definování bodů a souřadných systémů robotu ABB

Na obr. 13 můžeme vidět ukázkou zadání SS v kontroléru firmy ABB.

Při jakémkoliv plánování pohybu robotu je nutné určit, jakým způsobem se bude robot pohybovat (**Pohybové instrukce**). Mezi základní typy pohybu patří Linear (lineární dráha), Point-to-Point (logicky nejkratší cesta, ne vzdálenostně nejkratší) a Circular/Arch (kruhová interpolace).



a) Joint motion.

b) Linear motion.

c) Arch motion.

Obr. 14 – Typy pohybových instrukcí robotu [17]

### 3.3.2 Programování

Programování robotů je možné skrze přiložený **Teach pendant**, ale v případě středních až větších projektů je možné a vhodné robotický program vytvářet ve **specializovaných softwarech**. Programovací interface a struktura pracuje na stejném principu jako standardní textové programování. Je možné používat standardní struktury (IF, ELSE, CASE, WHILE, atd). Hlavním rozdílem je volání operací, které funguje na stejném principu jako u PLC. Robot vyvolává vždy program MAIN a prochází program řádek po řádku (při nesplnění podmínky jednoho řádku se zde čtení celého programu zastaví). Z toho důvodu dělíme program do podprogramů, které vyvoláváme určitým signálem v MAIN.

Softwary mohou být určeny buď jen pro určitou značku robota, jako například ABB RobotStudio či KUKA.Sim, nebo univerzální, například Process Simulate. Značkové softwary mají výhodu virtuálního zprovoznění samotného ovládacího kontroléru s vnitřní signálovou strukturou, a tím umožňují simulovat roboty kompletně. Nevýhodou je pak nemožnost kombinování dvou a více robotů jiné značky.

Samotný robotický program se liší od výrobce. Každá řídicí jednotka má svou vlastní syntax povelových instrukcí, ale odlišnost je vcelku nízká.

Poté, co jsou zadány informace o koncovém efektoru a souřadných systémech, začíná proces programování. Body, mezi kterými se má robot pohybovat, zadáváme různými způsoby: hodnotově, najetím robotu do bodu pomocí teach pendantu, nebo off-line programováním.



Obr. 15 – Teach pendant robotu KUKA [18]

```
LOCAL CONST robtarget HOME:=[[ 401,0,637],[0,0,1,0.000000],[1,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
LOCAL CONST robtarget I_PICK:=[[99,-710,70.5],[0.000001,0,1,0],[-1,0,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
LOCAL CONST robtarget PICK:=[[99,-710,40.5],[0.000001,0,1,0],[-1,0,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
LOCAL CONST robtarget PICK_I:=[[99,-710,70.5],[0.000001,0,1,0],[-1,0,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
LOCAL CONST robtarget PICK_II:=[[304.54,-929.96,36.01],[0.000001,0,1,0],[-1,0,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
LOCAL CONST robtarget I_PLACE:=[[30,30,66],[-0.000003,0.707107,-0.707107,-0.000003],[-1,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
LOCAL CONST robtarget PLACE:=[[30,30,36],[-0.000003,0.707107,-0.707107,-0.000003],[-1,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

Obr. 16 – Definování bodu v robotickém kontroléru ABB

Na obr. 16 je vidět ukázka definovaných bodů v kontroléru firmy ABB.

Definice bodu:

- Local const
  - Definice proměnné – lokální konstanta
- Robtarget
  - Definice nové pozice
- HOME

- Pozice XYZ
- Orientace XYZ
- Konfigurace OS
- Pozice externích OS

Po vytvoření bodů se určí typ pohybu, kterým se bude robot do bodu pohybovat, a tím získáváme základní operaci. Operací typu **uchop, svař, nanášeť**, poté docílíme zavoláním spouštěcího signálu a počkáním na odezvu, že operace byla/je prováděna.

Na obr. 17 vidíme ukázkou MAIN programu, volající definované podprogramy v kontroléru firmy ABB a ukázkou podprogramu OP\_2. Obr. a) obsahuje strukturu MAIN programu volajícího podprogram na základě proměnné PNI\_reqOp. Na obr b) je volaný podprogram, obsahující povelové a signálové instrukce.

U ostatních výrobců robotů se postupuje obdobně.

<pre style="font-family: monospace; font-size: 0.9em;"> PROC main()   Init;   TEST PNI_reqOp   CASE operace1:     DoOperation operace1, "OP_1";    CASE operace2:     DoOperation operace2, "OP_2";    ENDTEST  ENDPROC         </pre>	<pre style="font-family: monospace; font-size: 0.9em;"> PROC OP_2()   MoveJ HOME,v1000,fine,gRIPPER\Wobj:=wobj0;   MoveJ I_VEZMI,v200,z10,gRIPPER\Wobj:=Wobj_mer_stan;   MoveL VEZMI,v50,fine,gRIPPER\Wobj:=Wobj_mer_stan;    Set GripperClose;   WaitDI GripperClosed, high;   Reset GripperClose;    MoveL VEZMI_I,v200,z0,gRIPPER\Wobj:=Wobj_mer_stan;   MoveJ I_POLOZ,v200,z0,gRIPPER\Wobj:=Wobj_dopr_niz;   MoveJ POLOZ,v50,fine,gRIPPER\Wobj:=Wobj_dopr_niz;    Set GripperOpen;   WaitDI GripperOpened, high;   Reset GripperOpen;    MoveJ POLOZ_I,v200,z10,gRIPPER\Wobj:=Wobj_dopr_niz;   MoveJ HOME,v1000,fine,gRIPPER\Wobj:=wobj0; ENDPROC         </pre>
--	--

a)

b)

Obr. 17 – a) MAIN program robotu, b) Volaný podprogram

### 3.4 Tecnomatix Process Simulate

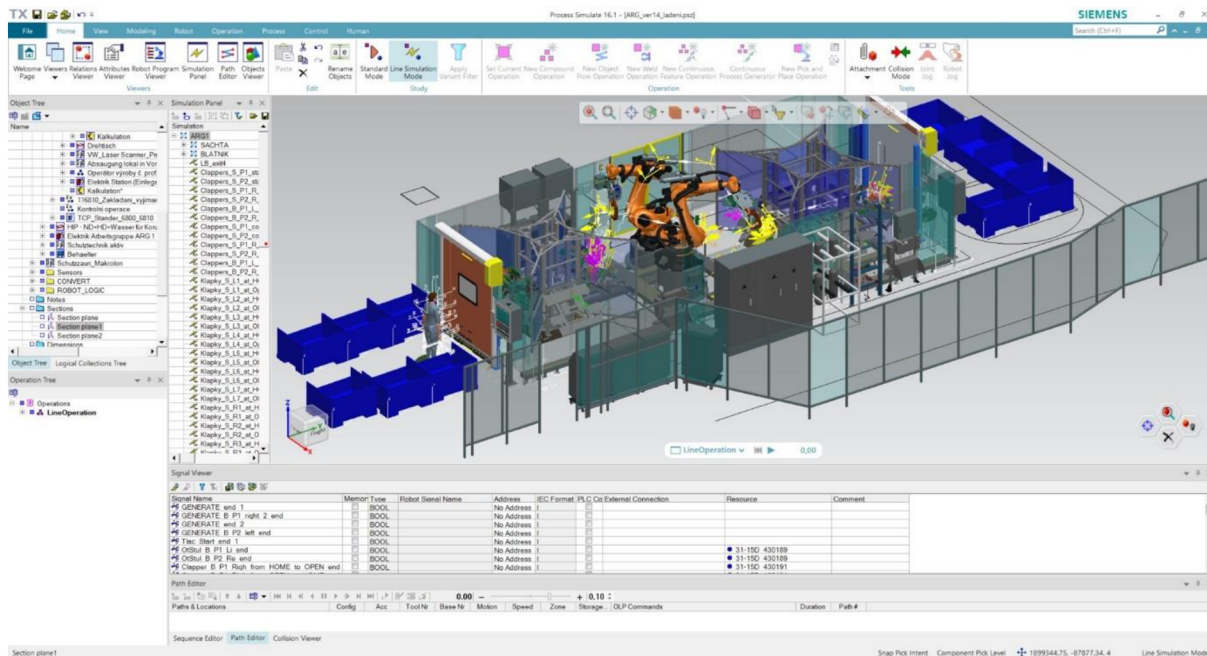
V současné době existuje celá řada softwarů, umožňujících simulaci procesů, potřebných pro virtuální zprovoznění. Velké firmy se zaměřením na automatizaci obvykle vyvíjí vlastní software. Mezi nejznámější patří software Process Simulate (zkr. PS) od firmy Siemens.

PS spolu se softwary Plant Simulation a Process Designer patří do skupiny softwarů Tecnomatix digitální továrna. Jedná se o komplexní řešení virtuální automatizace od firmy Siemens.

PS umožňuje detailně simulovat celý proces výrobní linky ve 3D prostředí. Využívá modely strojů, součástí a dodává jim potřebnou logiku k realizaci simulace, včetně vytváření potřebných signálů pro reálné fungování linky a simulování ergonomie obsluhy. Vytváří také univerzální program pro všechny typy robotů na základě definované verze kontroléru. Po vytvoření logiky je software následně možné provázat s PLC či virtuálním robotickým kontrolérem, a řídit signály externě jako v reálném prostředí.

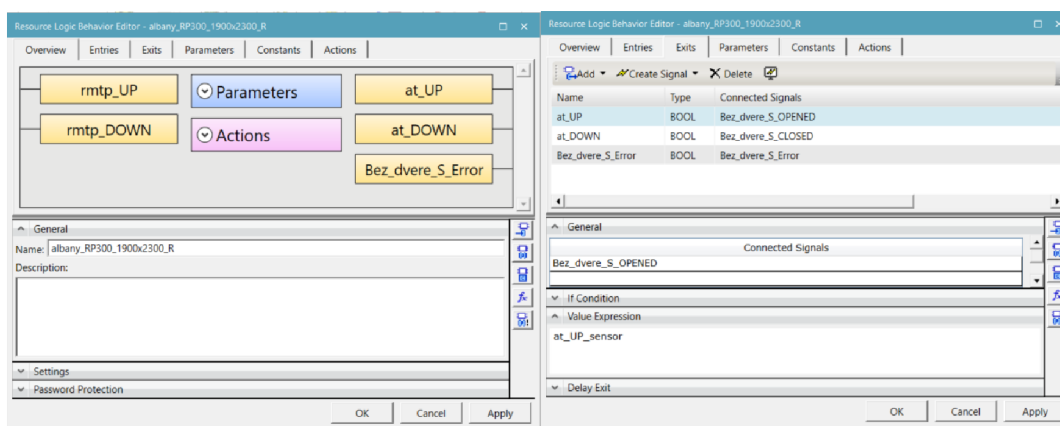
Interface softwaru PS je podobný jako u všech moderních aplikací. Horní část obrazovky je tvořena lištou s ovládacími prvky, rozdělenou do jednotlivých podskupin. Zbytek obrazovky si může uživatel uspořádat z jednotlivých oken dle vlastní potřeby. Nejdůležitějšími okny jsou v zásadě 3D pohled na tvořené pracoviště a strom využitých prvků.

Mezi další softwary určené k virtuálnímu zprovoznění dále patří například RobotStudio od firmy ABB, Kuka.Sim od firmy KUKA, DELMIA od firmy Dassault Systemes, a další.



Obr. 18 – Interface programu Process Simulate

Signály i signálovou logiku všech komponent tvoříme v tzv. **Logic Resource** editoru. Jedná se o programovací rozhraní určené k definování logiky dané součásti.



Obr. 19 - Ukázka fungování Logic Resource editoru v PS

### 3.5 PLC (programing logic controller)

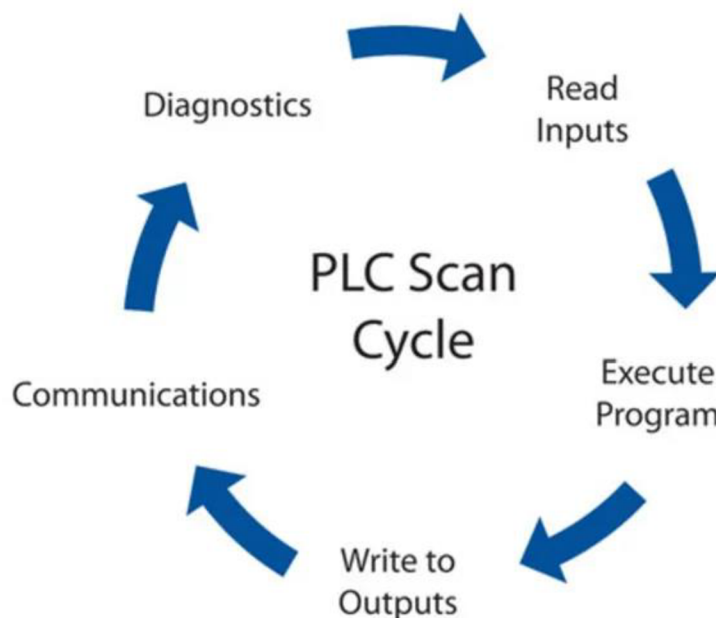
Téměř každý výrobní systém je ovládán skrze některý řídicí systém. Dříve se jednalo o jednoduché spínače relé, ale s vývojem technologií byly relé vytlačeny tranzistory s rychlejšími spínacími rychlostmi a životností. Aktuálně se na trhu můžeme převážně setkat se čtyřmi typy řídicích systémů. Jedná se o Programmable Logic Controller (PLC), Supervisor Control And

Data Acquisition (SCADA), Distributed Control System (DCS) a Human Machine Interface (HMI).

DCS vnikal paralelně s PLC a jsou si velmi podobné. Nachází uplatnění u řízení celých budov a velkých procesů. SCADA je software podobný počítačovým operačním systémům a může paralelně fungovat s PLC či DCS. Využívá se k monitorování všech dat pohybujících se procesem z mnoha zařízení a pomáhá odhalit chyby omezující proces. HMI je technologie určená ke grafickému znázornění a ovládání procesu pomocí odolné dotykové obrazovky, umístěné v blízkosti linky, kterou nejčastěji využívá obsluha. HMI je možné programovat a může fungovat samostatně. Většinou jsou ale z důvodu výpočetního výkonu vyžadovány běžící programy PLC či DCS, z kterých získáváme potřebné aktuální hodnoty proměnných, a následně posíláme rozkaz ke změně.

Nejpoužívanějším systémem průmyslové automatizace je **PLC**. Jedná se o malý počítač určený k řízení procesů v reálném čase s vysokou spolehlivostí, rychlou odezvou a bezpečností. Využití je především v průmyslu a v posledních letech i v domácí automatizaci.

Hlavním rozdílem mezi PLC a standardním stolním počítačem (PC) je forma vykonávání programu. PC funguje na bázi náhodného přístupu. To znamená, že vykonává žádaný program „paralelně“, nečeká například na dokončení FOR loopu, a podprogramy nemají pevně danou dobu k vykonání, což je v průmyslu nežádoucí (není možné čekat stovky milisekund na signál o poloze a otáčkách). Oproti tomu PLC pracuje ve smyčkách, což znamená, že program je vykonáván sekvenčně krok po kroku, a po vykonání jedné smyčky znovu načte aktuální hodnoty IO modulů a startuje od prvního řádku kódu. Nevýhodou je omezené využití smyček (program by se zasekl a nestihl by provést zbytek programu v maximálním možném čase jedné smyčky). Výhodou je možnost řízení procesů v reálném čase s minimální časovou odezvou.



Obr. 20 – Princip cyklů PLC [19]

Dalším rozdílem je uzpůsobení PLC ke zpracovávání velkého množství externích signálů z periférií a jejich řízení (stovky až tisíce). Většina PLC má základní modul pro digitální vstupy/výstupy (DI/DO) a analogové vstupy/výstupy (AI/AO) pro spojitě signály. Dále je také

možné pripojit komunikačné karty a mnoho ďalších podporných modulů (např. pro PWM - pulzní vlnovou modulaci).

Z hlediska konstrukce můžeme PLC rozdělit na dva základní typy – **kompaktní a modulární**.

Kompaktní se vyznačují omezeným množstvím přídavných modulů, po zakoupení si zákazník může dokoupit jen předem definované moduly a jejich množství. Využití těchto hardwarů je především u malých automatizací s malým množstvím I/O. Výhodou jsou pořizovací náklady.



a)



b)

Obr. 21 – Kompaktní PLC a) LOGO! firmy SIEMENS, b) firmy E.T.N [20] [21]

Modulární oproti tomu nabízí možnost přikoupení téměř neomezeného množství rozšiřujících modulů a jejich kombinací. PLC se umístí na DIN lištu a moduly se připojují vedle něj zprava. Tento typ nalezneme ve většině velkých automatizovaných procesech. Další výhodou je také možnost připojení modulů na delší vzdálenosti. Pokud se například nachází velké množství signálů v jedné části linky, je možné I/O modul umístit blíže, tím ušetřit množství tažených kabelů, a následně samotný I/O modul připojit skrze jediný kabel.



a)

b)

Obr. 22 – Modulární PLC a) bez roz. modulů, b) s moduly na DIN liště [22] [23]

Konstrukčně se PLC příliš neliší od PC, důraz je ovšem kladen na odolnost vůči prostředí, spolehlivost a stabilitu systému. Nejdůležitějšími komponentami jsou procesory.

Jádro řídicího systému (CPU) obsahuje většinou jeden hlavní procesor (master) a celou řadu sekundárních (slave). Společně zajišťují rychlé zpracování signálů a provádění programů.

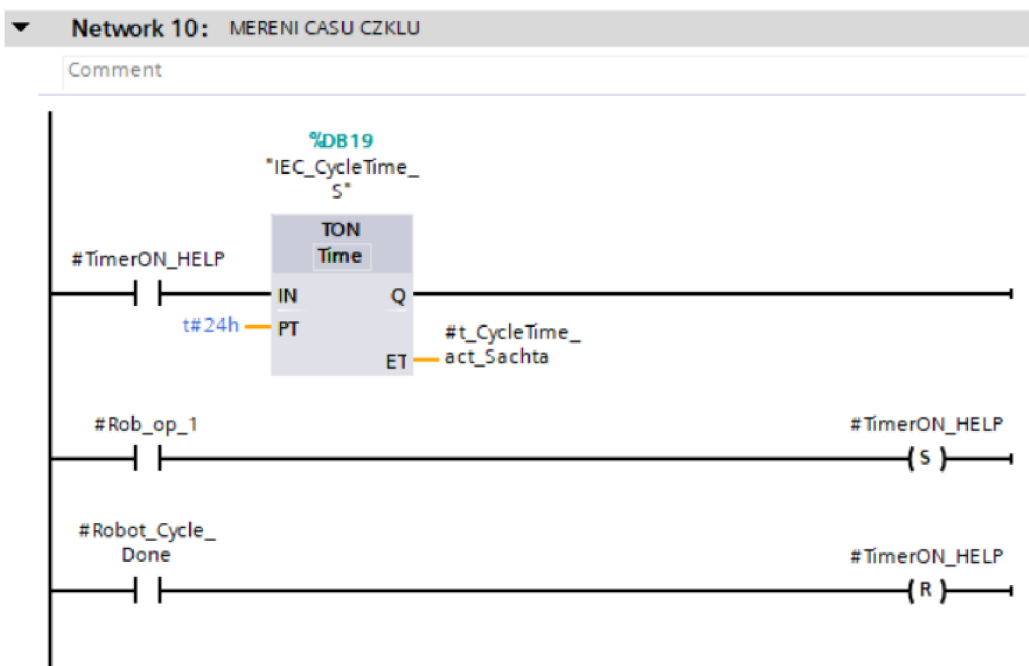
### 3.5.1 Programování PLC

V současné době většina PLC disponuje pěti programovacími jazyky, standardizovanými dle normy IEC 61131-3. Tato norma uvádí pět jazyků, z toho dva textové a tři grafické.

- Grafické
  - Ladder Diagram (LD, LAD)
  - Function Block Diagram (FBD)
  - Sequential Function (SFC)
- Textové
  - Structured Text (ST/SCL)
  - Instruction list (IL)

Při vývoji byly PLC navrhovány tak, aby je byli schopni obsluhovat a programovat i lidé bez zkušeností s tradičním programováním. Z toho důvodu byl jako první představen jazyk Ladder Diagram. Ten je spolu s FBD a SFC postaven na grafickém znázornění logiky. Výhodou je skvělá interpretace fungování řídicích programů pro ne-programátory.

**Výhodou** je přehlednost, grafické barevné znázornění simulace a jednoduchost tvoření logiky. **Nevýhodou** je složité tvoření komplikovaných funkcí, ztráta přehlednosti u programů s velkým množstvím vstupů, a tím pádem větší důraz na optimalizaci podprogramů.



Obr. 23 – Ukázka větve vytvořené v jazyku Ladder Diagram

```

18 CASE #n_rezim OF
19   #STATE_AUTO: // STATE AUTO
20     CASE #n_StateNum OF
21       #STATE_INIT: // Kontrola vstupu // reset
22         #t_timer(IN := 1,
23           PT := T#2s);
24       IF #t_timer.Q THEN
25         #n_StateNum := #STATE_AUTO_ON;
26         #t_timer.IN := 0;
27       END_IF
28     ;
29   #STATE_AUTO_ON: // Statement section case 2 to 4
30     "DB_LD_Robots_Sachta_Auto_P2"(Sensor_ll := "CONTROL_DATA"."SEN/FB".I.Up_prostor.b_senzor_S_Pl_11, ...);
64
65     "DB_LD_Robots_Blatnik_Auto"(Sensor_ll := "CONTROL_DATA"."SEN/FB".I.Up_prostor.b_senzor_B_Pl_1, ...);
86
87     "LD_Operator_Sachta_DB"(I1_Start_Aretacni := "CONTROL_DATA".CONTROL.Q.Ovladani.b_Tlac_start_Sachta, ...);
124
125     "LD_Operator_Blatnik_DB"(I1_Start_Aretacni := "CONTROL_DATA".CONTROL.Q.Ovladani.b_Tlac_start_Blatnik, ...);
148
149   ;
150 END_CASE;
151 ;
152
153 #STATE_MANUAL: // Statement section case 2 to 4
154 ;
155 ;
156 END_CASE;

```

Obr. 24 - Ukázka větve vytvořené v jazyku Structured Text (Strukturovaný text)

Druhý nejpoužívanější programovací jazyk je Strukturovaný text (ST/ Structured text). Jedná se o standardní textové programování. Výhodou je komplexní programování. Nevýhodou je nepřehlednost a špatná interpretace, programátor musí přemýšlet o čitelnosti kódu z pohledu třetí osoby, jinak se stává do budoucna jedinou osobou, která je schopna program modifikovat.

PLC se standardně programují, simulují a testují na konvenčních PC pomocí specializovaných softwarů. Po dokončení testování lze program nahrát na interní úložiště PLC (flash paměť nebo RAM) skrze síť, flash disk nebo SSD.

### 3.5.2 Komunikace

PLC běžně využívá celou řadu komunikačních **portů**, tzn. USB, Ethernet, RS – 232, RS – 485, RS – 422, či další dle požadavků zákazníka.

Porty přenáší signály dle standardizovaných **komunikačních protokolů**, například Modbus, EtherCat, PROFIBUS a další. Každý výrobce PLC upřednostňuje některý protokol jako primární pro své produkty, ale PLC je možné nakonfigurovat ke komunikaci i přes další protokoly a v případě potřeby dokoupit specifický komunikační modul. Obecně se doporučuje využívat jeden druh protokolu a využívat PLC a dodatkové moduly od jedné firmy. Díky tomu je bezpečně zajištěna kompatibilita a interní komunikace jednotlivých součástí. V reálném prostředí se ovšem většinou setkáme s celou řadou komunikačních protokolů a je nutné znát jejich princip, strukturu dat, a vědět, jak je využít a zajistit vzájemnou kompatibilitu.

### 3.5.3 Výrobci

Mezi nejznámější výrobce PLC jednotek patří:

- Siemens
  - Allen Bradley
  - Honeywell
  - Schneider Electric
  - Beckhoff
  - Mitsubishi Electric
  - ABB
- a další.



### 3.6 TIA Portal

TIA Portal (Totally Integrated Automation Portal) byl představen společností SIEMENS v roce 2011. Jedná se o multifunkční softwarové prostředí, určené k projektování, programování a testování řídicích systémů (např. PLC), operátorských rozhraní (především HMI) a elektrických pohonů.

V současné době je uživatel v TIA schopen vytvořit, simulovat a kontrolovat automatizační řetězec všech žádaných součástí pracoviště. Ať už se jedná o PLC, frekvenční měniče pohonů nebo HMI operátora. Následně lze konfigurovat komunikaci mezi všemi prvky (např. kom. protokoly, sítě a přiřazení IP adres). Každá ze zmíněných komponent má svou vlastní databázi obsahující informace o daném modelu zařízení a všech dostupných přídatných modulech, které je možné domodifikovat, jakkoli to výrobce umožňuje. Kromě interních zařízení dokáže TIA pracovat s celou řadou decentralizovaných signálů (např. sensorika).

Samotné prostředí je velmi intuitivní a přátelské obsluze.

### 3.7 Komunikační protokoly

V továrnách, civilních budovách či v domácnostech se neustále dostáváme do kontaktu s elektrotechnickými předměty. Aby bylo možné tuto techniku efektivně užívat a komunikovat s nimi navzájem, bylo nutné vytvořit určité standardy/protokoly, definující strukturu odesílaných zpráv. Díky tomu je možné odesílat drátově i bezdrátově informace veřejně (všem uživatelům sítě), popřípadě neveřejně (point-to-point/ z bodu do bodu).

Užívané protokoly můžeme rozdělit dle typu přenosu na drátové a bezdrátové.

#### 3.7.1 Protokoly u drátového přenosu

Mezi neznámější protokoly dodnes patří **MODBUS**. Jedná se o jednu z nejstarších sériových komunikací. Funguje na principu master/slave, kdy master odesílá dotazy a připojená zařízení odesílají aktuální hodnoty. Výhodou je jednoduchost, spolehlivost a open-source (otevřený zdrojový kód – může ho využívat kterýkoliv výrobce).

V současné době je MODBUS nahrazován protokoly využívající ethernet (optické kabely s kroucenou dvojlinkou). Jedním z nich je například **PROFINET**. Jedná se znovu o open-source protokol, zajišťující cyklickou a acyklickou komunikaci. Největší výhodou je rychlost přenosu dat (zajišťující stabilní komunikace v reálném čase) a zabezpečení.

Další průmyslové protokoly:

- EtherCAT
- Ethernet/IP
- IO-Link

#### 3.7.2 Protokoly u bezdrátového přenosu

Druhá kategorie se zabývá průmyslovými bezdrátovými komunikačními protokoly. V moderní době se s nimi můžeme setkat stále častěji a jsou nedílnou součástí hlasitě zmiňované průmyslové revoluce 4.0.

Největší výhodou přenosu je odstranění kabeláže, a tím i značných nákladů ve velkých výrobních závodech, čímž se zmenší úbytek pracovního prostoru.

Nevýhodou pak může být náchylnost k rušení, a tím i menší stabilita systému, vyšší nároky na bezpečnost (je nežádoucí, aby třetí strana získávala údaje z vysílacích signálů, v horším případě ovládala prvky pracoviště) a pořizovací náklady.

Protokoly na bázi **Wi-Fi standardu** se z důvodu špatné časové odezvy využívají primárně na minimálně časově závislé prvky pracoviště a na místech určených k dlouhodobému zaznamenávání hodnot. Mezi nejpoužívanější typy patří HaLow Wi-fi (frekvence 900 MHz, vysoký dosah, málo energeticky náročné) a WiGig (frekvence 60 GHz, vysoký dosah, rychlý přenos dat).

Další protokoly k bezdrátovému přenosu:

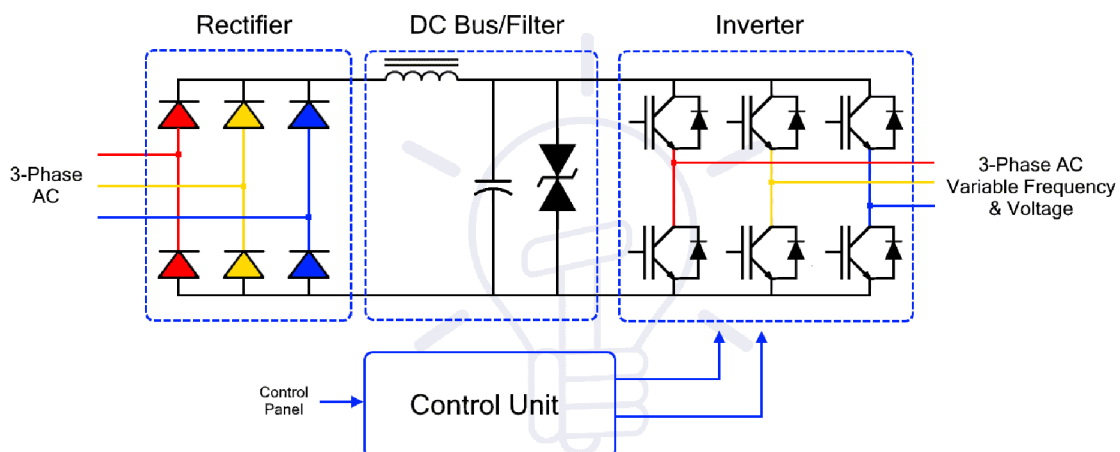
- WirelessHart
- 6LoWPAN
- ZigBee
- LoRA

### 3.8 Frekvenční měnič (Variable Frequency Drive / VFD)

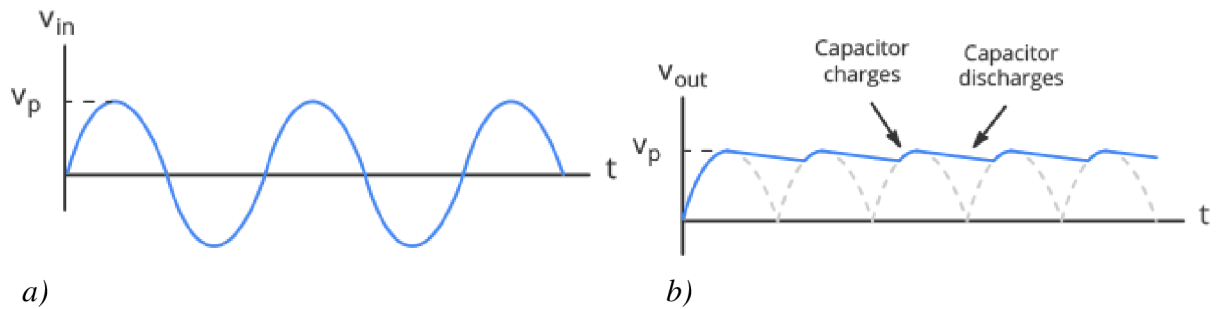
V současně době narazíme na elektrické pohony téměř na každém kroku. Výhodou je jednoduché zapojení, ovládání a finanční náklady. Ke zmíněnému ovládání využíváme frekvenční měniče. Jedná se o zařízení, které nám umožňuje měnit vstupní hodnoty frekvence elektrického napětí, a tím řídit chování elektromotoru.

VFD nachází využití u všech elektrických pohonů, u kterých požadujeme řízení otáček či přesné polohování.

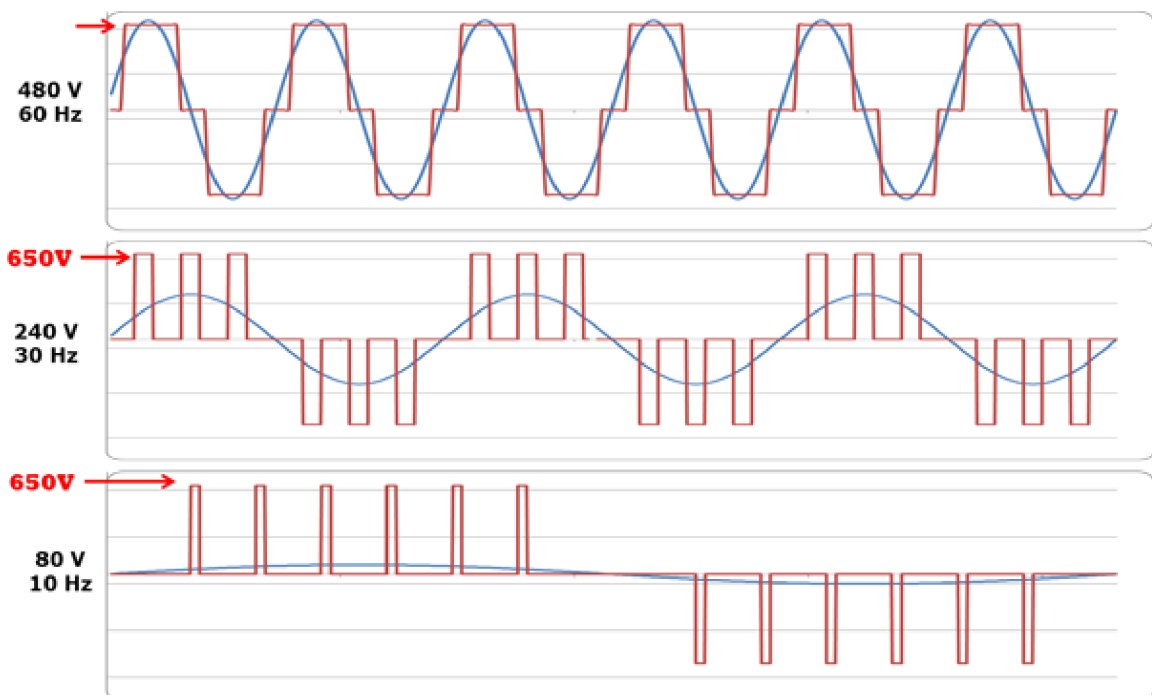
Konstrukčně obsahují měniče tři části. V první části usměrníme vstupní napětí z AC na DC za pomoci polovodičů (diod), následně prvotně usměrněné napětí vyhladíme v druhé části pomocí kondenzátorů a cívek. A v poslední části se na vytvořené DC napětí použije střídač k získání AC v žádané frekvenci. Střídač signál upravuje pomocí tzv. pulzní šířkové modulace (PWM), čímž vytváří diskrétní signály ON/OFF s definovanou střídou (časový poměr zapnuto/vypnuto). Díky této modulaci nezískáváme analogový signál jako na vstupu, ale aproximaci pomocí vln. Výhodou je efektivní ovládání výstupní frekvence.



Obr. 25 – Základní vnitřní struktura frekvenčního měniče [24]

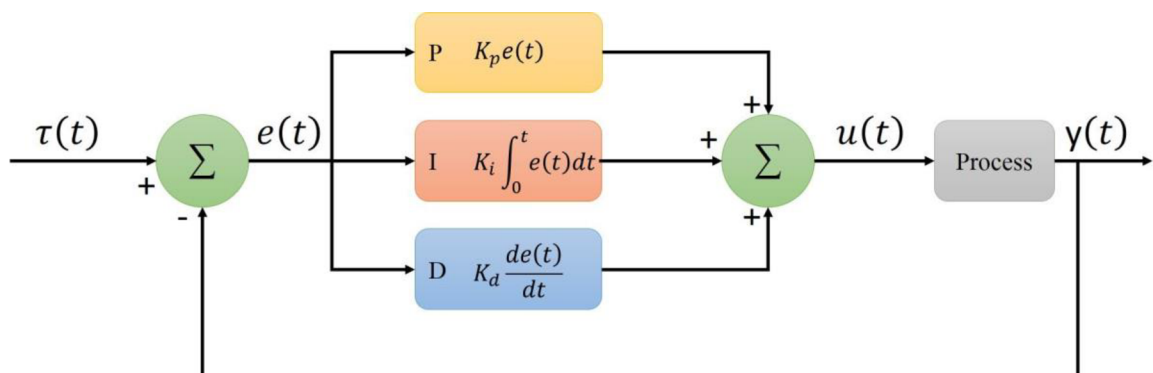


Obr. 26 – Vstupní a výstupní napětí po usměrnění a vyhlazení [25] [26]



Obr. 27 – Ukázka fungování výstupního střídače frekvenčního měniče [27]

V současné době se také můžeme stále častěji setkat s VFD měniči, které mají integrovaný PID regulátor. PID regulace zajišťuje nastavení výstupu PWM z měniče tak, aby byly dodrženy požadavky uživatele, například otáčky, a to i přes proměnlivé vnější vlivy či zátěž na regulovaný systém.



Obr. 28 – Standardní struktura PID regulátoru [28]

Na obr. 28 jde vidět schéma PID regulátoru. Systém obdrží informaci o žádané hodnotě  $\tau(t)$  a následně se z aktuální hodnoty  $y(t)$  vypočítá odchylka  $e(t)$ , kterou je cílem snížit na minimum, v ideálním případě eliminovat. Regulace poté probíhá nastavením tří složek: proporcionální, integrační a derivační (PID). Po obdržení hodnot získáváme zásah systému do procesu  $u(t)$  a cyklus se znovu opakuje.

Nastavení dílčích složek závisí na dané aplikaci, nejdůležitějšími hodnotami jsou žádaná rychlost ustálení a povolený překmit. Proces je možné nastavit mnoha způsoby, cílem je získat dostatečně vhodnou variantu pro danou problematiku. Metod pro získání hodnot PID složek je mnoho. Nejběžnější jsou například metody Ziegler-Nichols, Dvoubodová metoda a Pokus Omyl.

Využití nachází především v průmyslových aplikacích, kde je vyžadováno komplexnější ovládání zařízení. Nejběžněji se jedná o ovládání elektrických pohonů.

Výhody:

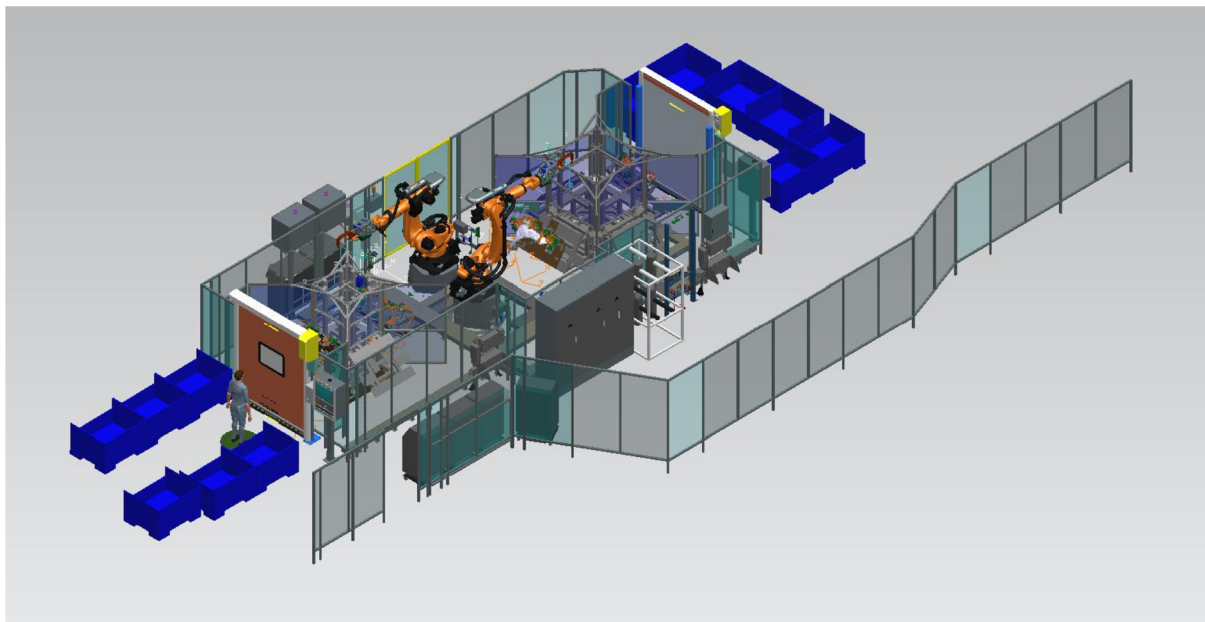
- Variabilní hodnota frekvence
- Přesné ovládání (poloha, otáčky, proud)

Nevýhody:

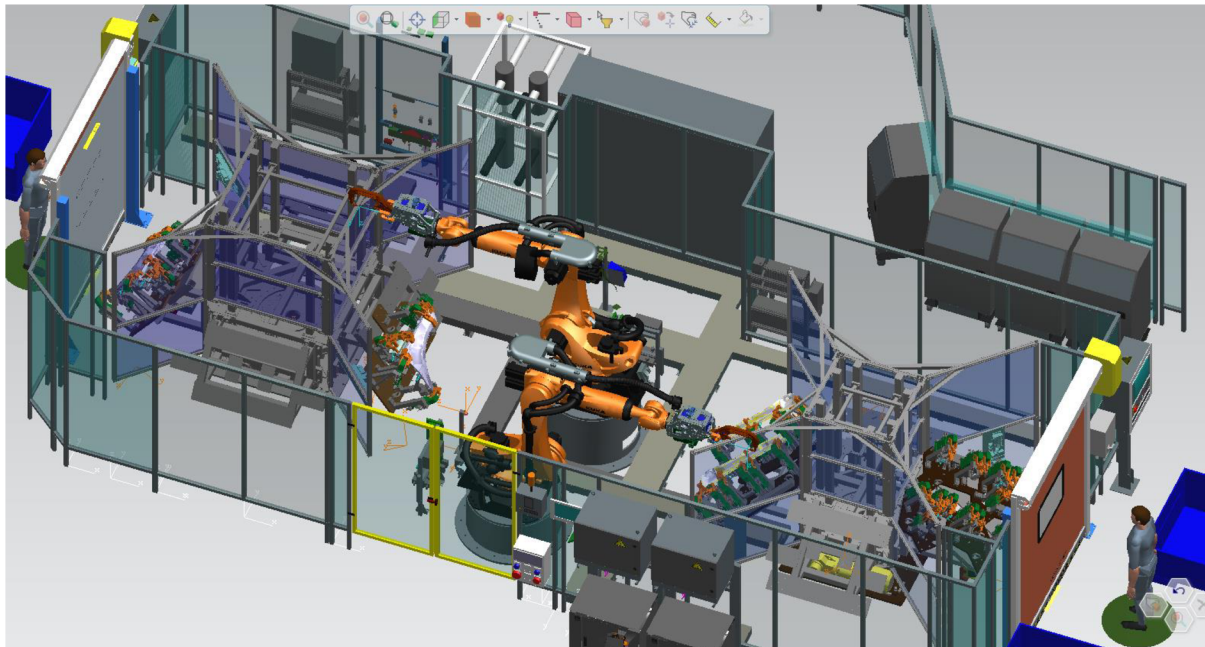
- Cena
- Bez převodovky ztrácíme výkon, jelikož výkon je roven momentu a otáčkám
  - $P = M * \omega$
- Nastavení PID / odbornost

## 4 ROZBOR ZADANÉHO PRACOVISTĚ

V práci bylo výchozím bodem robotické pracoviště pro bodové svařování. Jedná se o reálně fungující buňku určenou k bodovému svařování tenkých kovových dílů (polotovaru typu blatník a šachta).



*Obr. 29 – Zadané pracoviště*



*Obr. 30 – Detail zad aného pracoviště*

### Pracoviště tvoří:

- Ochranné krytí
- Rozvody kabelů
- 2x Otočné upínací zařízení pro upínání a vyjímání polotovaru
- 2x roboty vybaveny nástrojem pro bodové svařování

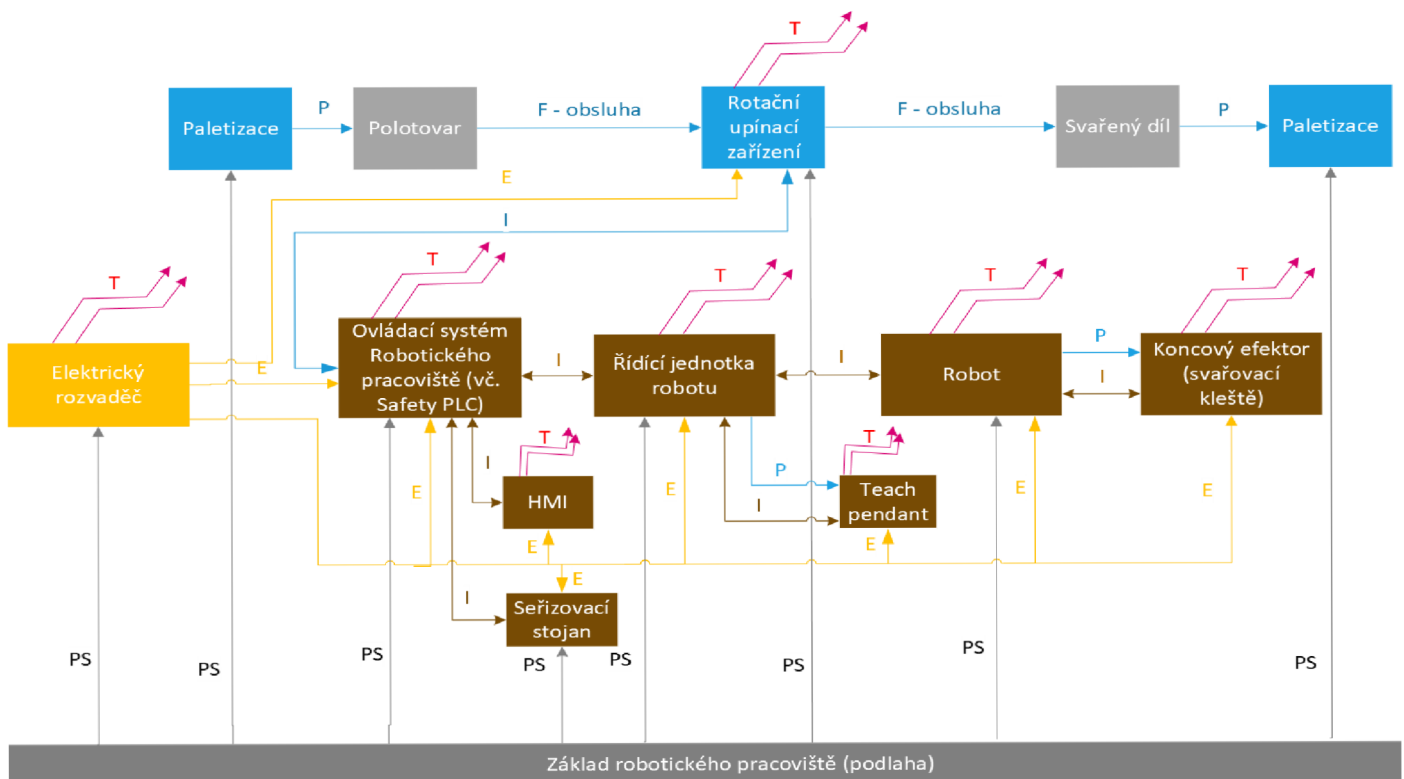
- Welding And Soldering Technology // **spot welding gun c**
- Čidla zajišťující bezpečnost a funkčnost
- Seřizovací stojany na svařovací nástroj robotu
- Rozvody a ovládací prvky elektřiny, pneumatiky
- Obsluha
- Palety

#### 4.1 Blokový diagram obecný



Obr. 31 – Obecný blokový diagram buňky

#### 4.2 Blokový diagram robotické buňky

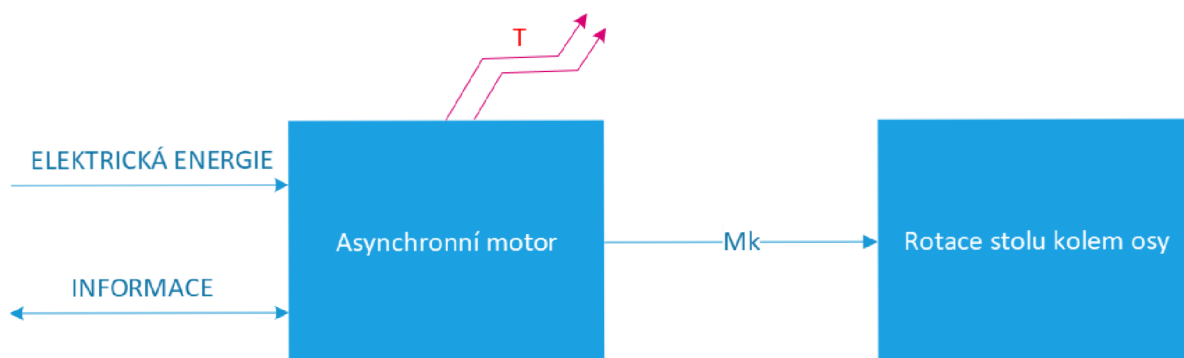


Obr. 32 – Blokový diagram robotické buňky

Legenda:

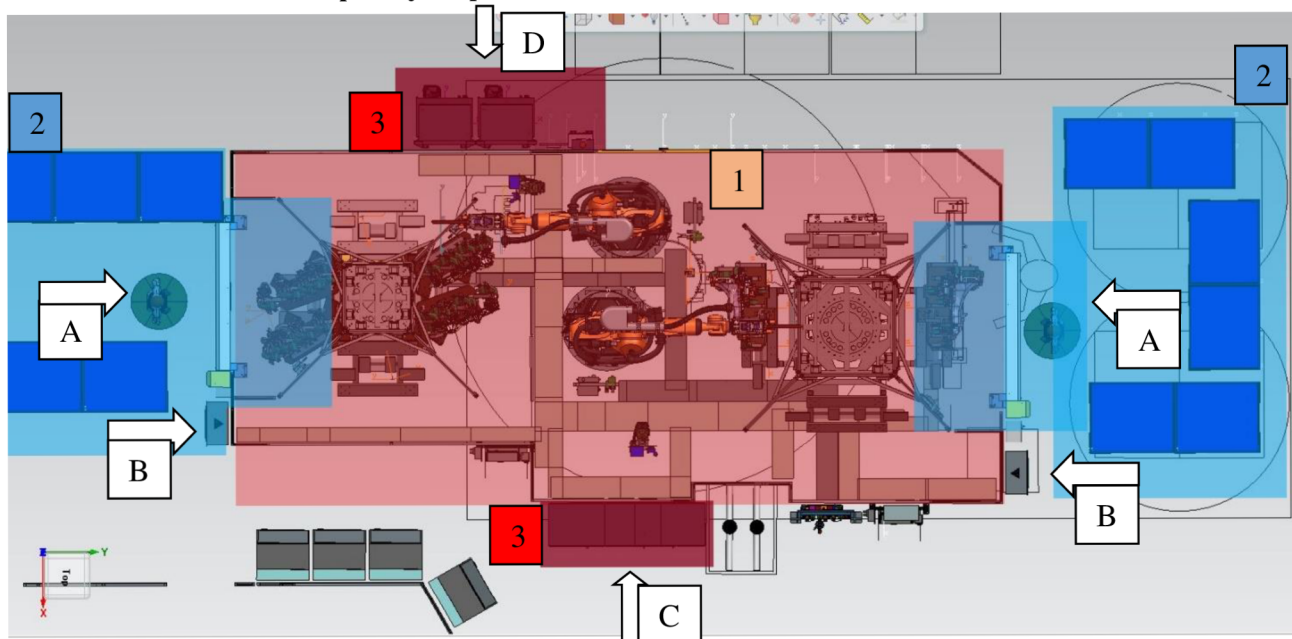
- P – polohová vazba
- F - síla
- PS – pasivní polohová vazba
- T - teplo

### 4.3 Blokový diagram upínacího zařízení/otočného stolu



Obr. 33 – Blokový diagram rotačního stolu

### 4.4 Definování nebezpečných prostor



Obr. 34 – Rozbor nebezpečných prostorů pracoviště

- |                                      |                                |
|--------------------------------------|--------------------------------|
| 1. Pracovní prostor                  | A. Vstup obsluhy               |
| 2. Prostor obsluhy + paletizace      | B. Vstup k HMI                 |
| 3. Prostory elektrorozvaděčů         | C. Vstupy k elektrorozvaděči   |
| 4. Nejbližší prostor robotické buňky | D. Vstupy k řídicí jednotce RB |

## 4.5 Plánovaný průběh pracovního cyklu

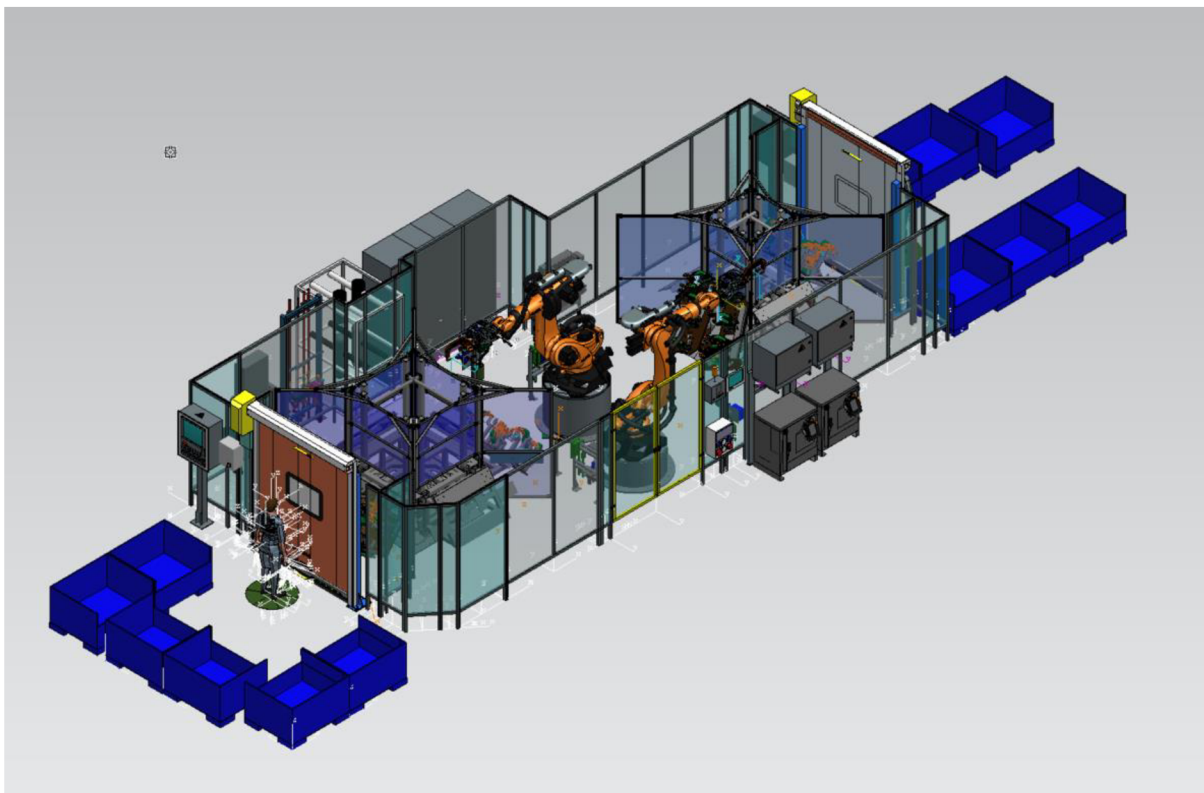
Robotizované pracoviště pro bodové svařování obsahuje dvě paralelně fungující buňky, fungující nezávisle na sobě. Obě fungují stejným způsobem. Tato podkapitola se věnuje popisu funkčního procesu jedné z nich.

### Postup obsluhy:

- Obsluha zaaretuje hříbové tlačítko, dojde k otevření bezpečnostních dveří a upínacích prvků rotačního stolu.
- Obsluha vyjme svařené díly, nebo umístí polotovary. Následně opustí pracovní prostor.
- Obsluha odaretuje hříbové tlačítko, tím dojde k zavření upínacích prvků, sjetí ochranných dveří a otočení upínacího stolu.

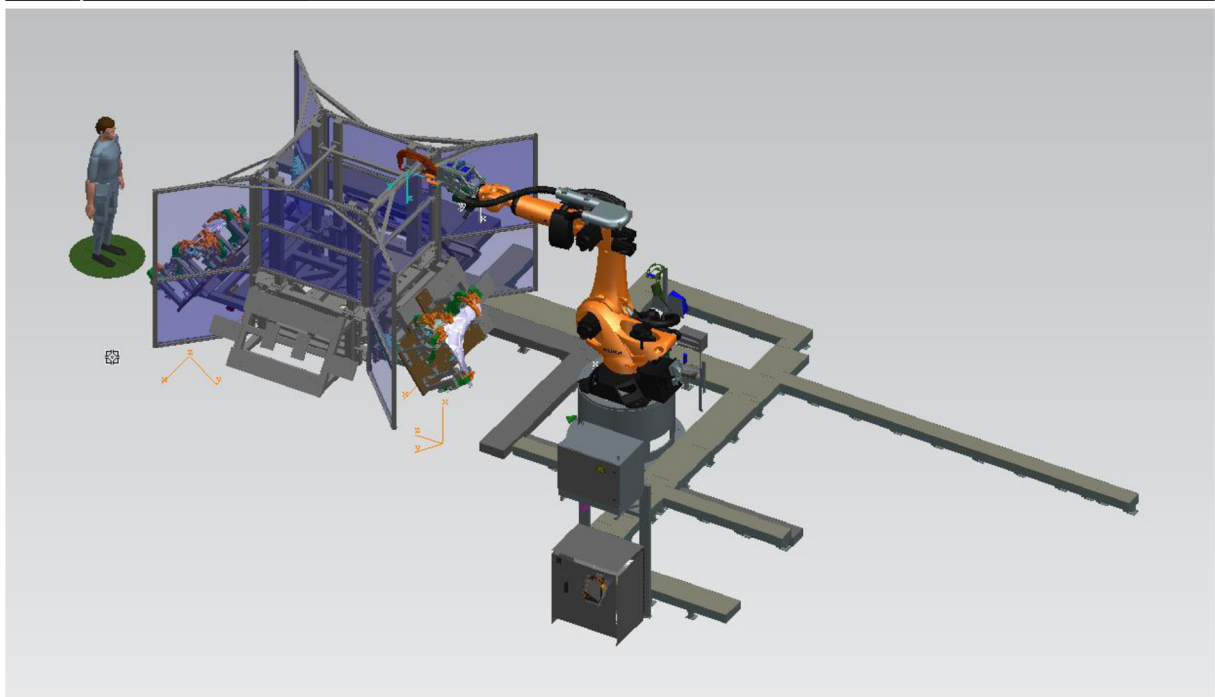
### Postup robotického pracoviště:

- Pokud je polotovar přítomen na upínací ploše rotačního stolu a stůl je natočen směrem k robotu a je zapnut AUTOMAT, provede se operace svařování.
- Po dokončení předá robot signál řídicímu systému o dokončení operace a stůl může rotovat.
- Po dokončení x svařovaných dílů robot provede seřízení u seřizovacího stojanu a následně se postaví do výchozí pozice.

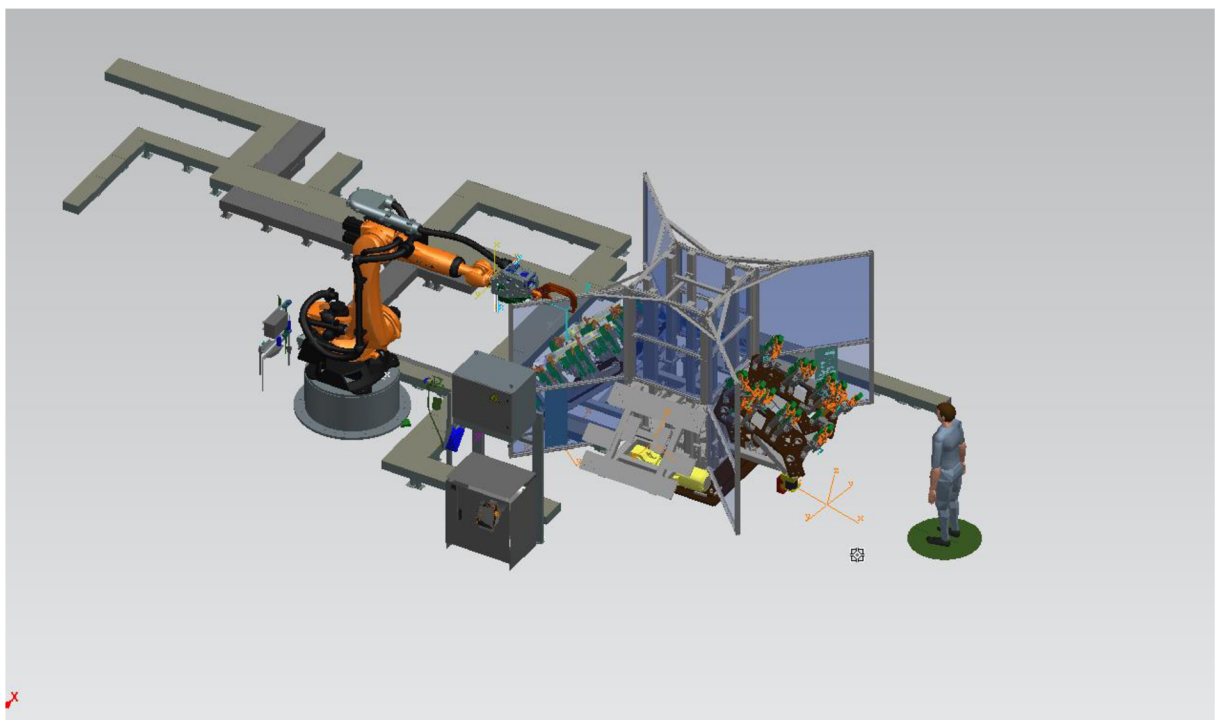


Obr. 35 – Detail pracoviště





*Obr. 36 - Detail na první část robotické buňky bez bezpečnostních prvků*



*Obr. 37 - Detail na druhou část robotické buňky bez bezpečnostních prvků*

## 4.6 Roboty

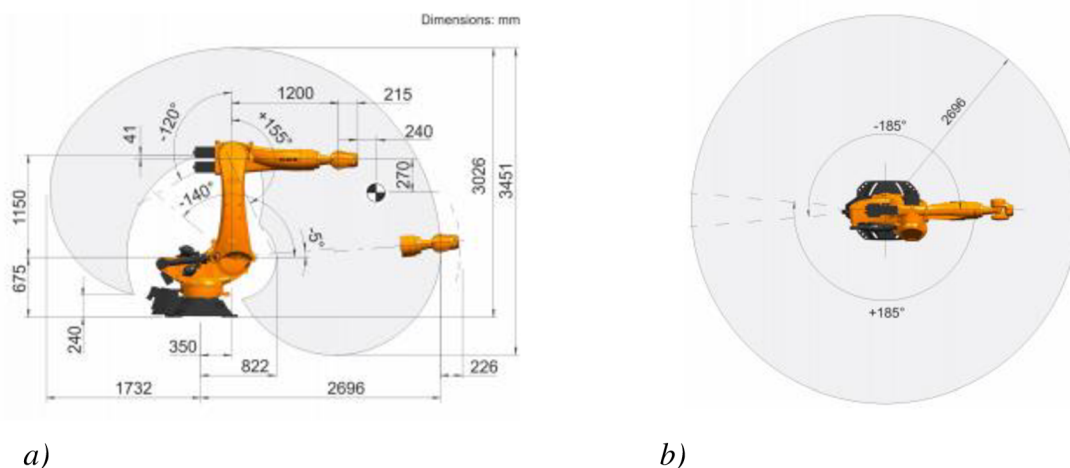
Využité roboty ve výrobní lince jsou **KUKA - kr 210 r2700 extra**. Jedná se o multifunkční šestiosé průmyslové roboty s nosností 210 kg.

V **původním stavu roboty** neobsahovali v PS žádné signály, senzory, či logiku, pouze operace sváření (viz kap. 4.9).

<b>Technické údaje</b>	
Maximální dosah	2696 mm
Jmenovitá nosnost	210 kg
Jmenovité přídavné zatížení ramene	50 kg
Jmenovité celkové zatížení	260 kg
Přesnost opakování polohy (ISO 9283)	± 0,06 mm
Počet os	6
Montážní plocha	Podlaha
Instalační plocha	830 mm x 830 mm
Hmotnost	Cca. 1068 kg
<b>Provozní podmínky</b>	
Teplota okolí při provozu	10 °C – 55 °C
<b>Druh ochrany</b>	
Druh ochrany (IEC 60529)	IP65
Druh krytí zápěstí robota (IEC 60529)	IP65
<b>Řídicí systém</b>	
Řídicí systém	KR C4

Tabulka 1 – Detaily robotu KUKA

### Obrázek pracovní zóny

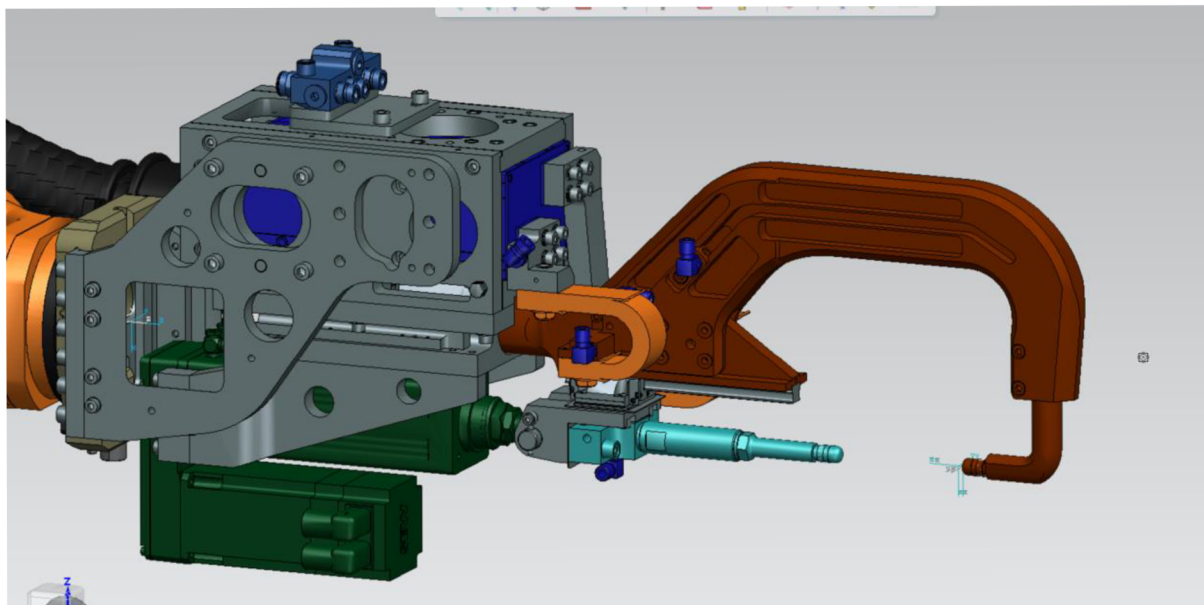


Obr. 38 – a) Pracovní zóny robotu, b) Sférický dosah robotu

Využité příslušenství:

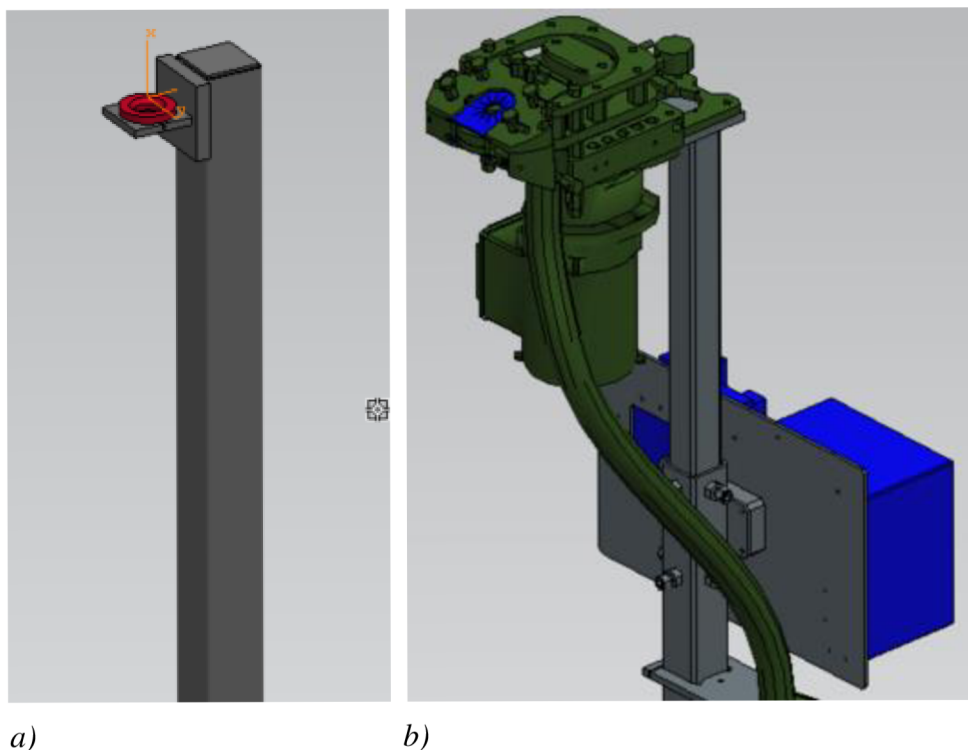
- **Bodové svařovací kleště typu C – pneumatické // spot welding gun C**

- Chladící kapalina
- Kabeláž – přívody elektrické energie k robotu a k svařovacím kleštím
- Sběrníkové moduly ke komunikaci s PLC skrze Profinet
- KUKA elektrorozvaděč robota
- Robotická konzola 500 mm vysoká pro roboty KUKA 150-240 kg Generation 2010



Obr. 39 – Svařovací kleště typu C

Svařovací kleště typu C. Jedná se o servo kleště používané k bodovému svařování. TCP je umístěno na špičce přední čelisti, kde osa Z směřuje k robotu. Druhá upínací část je pohyblivá s pomocí pneu/elektro válce. Díky tomu je zajištěna dostatečná upínací síla, nutná k vytvoření správného bodového sváru.



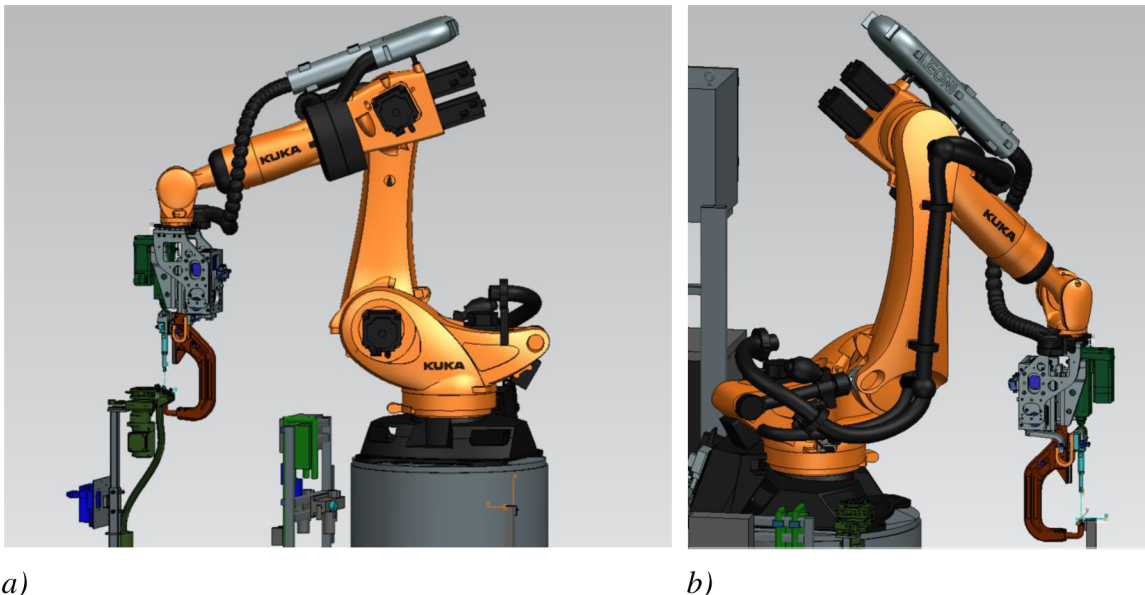
a)

b)

Obr. 40 – a) Stojan pro automatické zaměření TCP, b) Frézovací stojan

TCP a seřizovací stojan. Stojany jsou určeny k čištění/údržbě svařovacích kleští. TCP stojan má za úkol automaticky zkontrolovat a vyhodnotit aktuální pozici TCP (pracovního bodu nástroje). Svářecí stanice má za úkol ofrézovat špičku TCP, díky čemuž se odstraní nečistoty, které by mohly zabránit průchodu proudu, a tím i procesu sváření.

V **původním stavu stanice** neobsahovaly žádné signály, senzorku, či logiku.

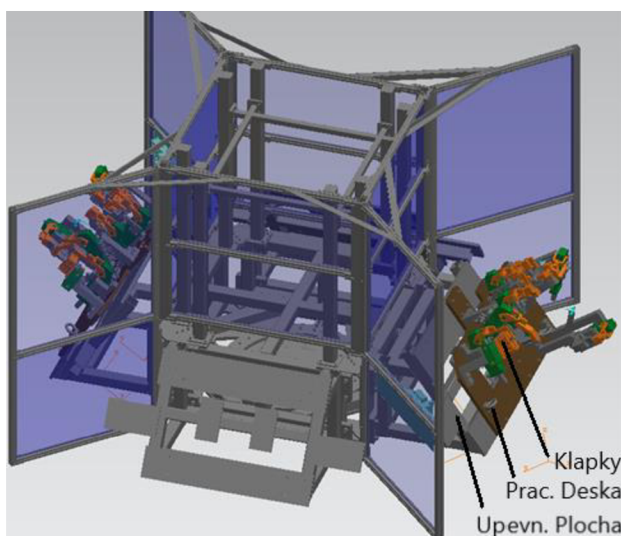


Obr. 41 – a) Operace frézování, b) Operace TCP

#### 4.7 Rotační upínací zařízení

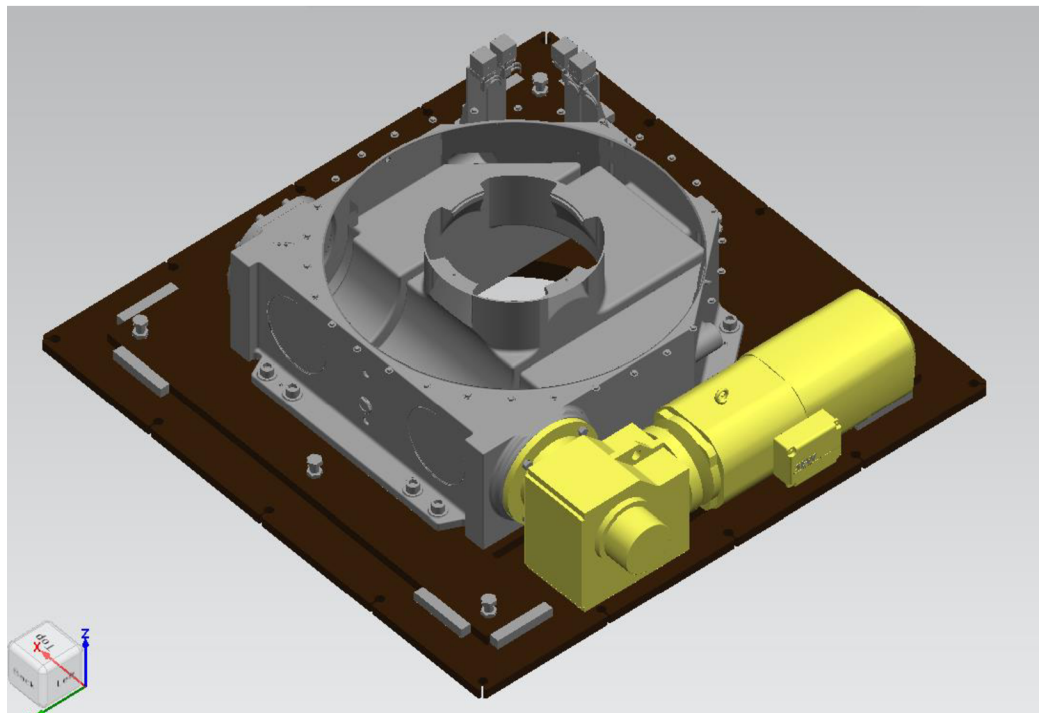
Zařízení slouží k upevnění polotovarů a následné rotaci o 180°. Tvoří ji kovová konstrukce, opatřená průhlednými ochrannými skly po bocích (ochrana před záblesky při procesu svařování) a kovovou nakloněnou plochou, na níž lze šrouby pevně upevnit pracovní deska.

Pracovní deska má již specifický tvar designovaný tak, aby na ní mohly být umístěny uchopovací prvky (clapery/“klapky“) a další pomocné prvky (podpurné, čidla atd.) dle typu používaného polotovaru.

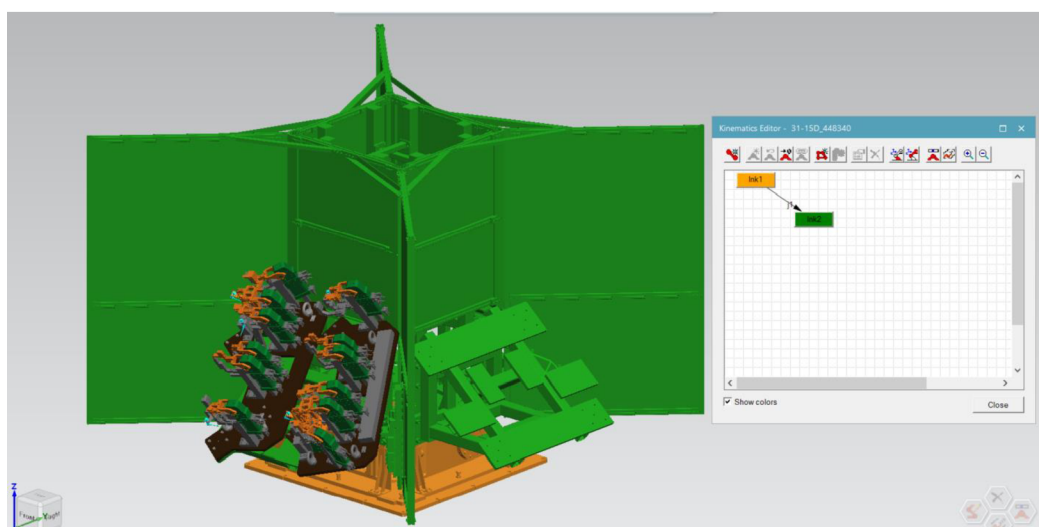


Obr. 42 – Popis rotačního stolu

Rotační mechanismus je poháněn jednoduchým servo pohonem se šnekovou převodovkou od firmy SEW. Nutné je využití enkodéru k zajištění přesného pootočení, či jiných senzorů.



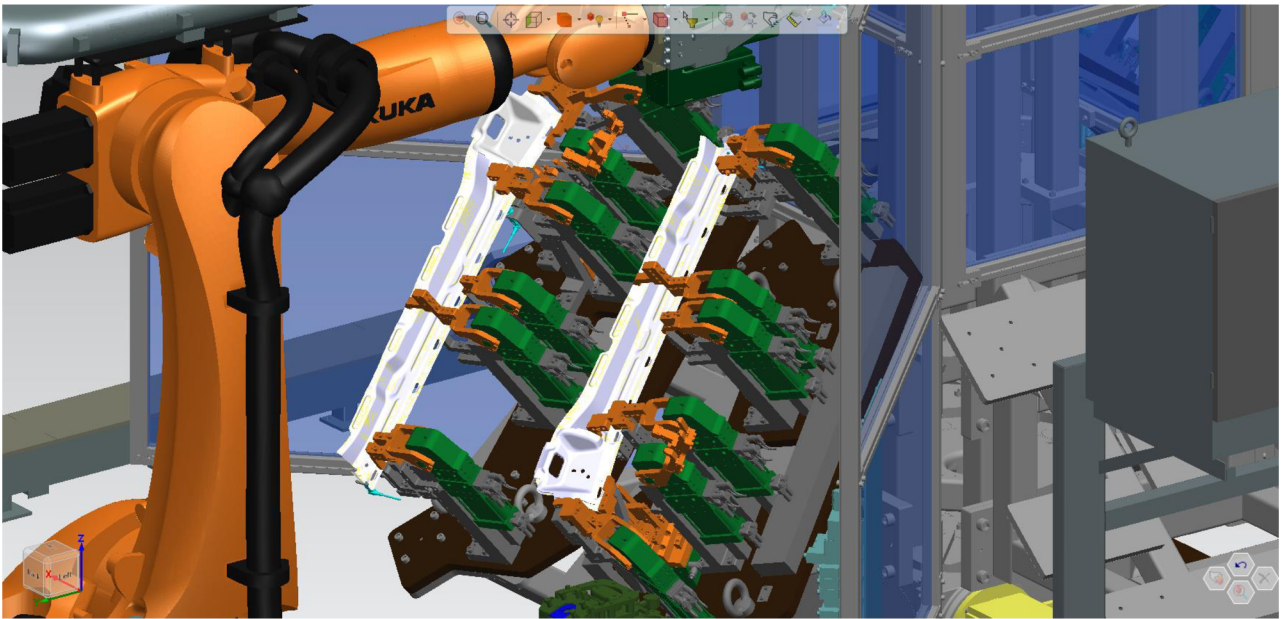
*Obr. 43 – Pohon rotačního stolu*



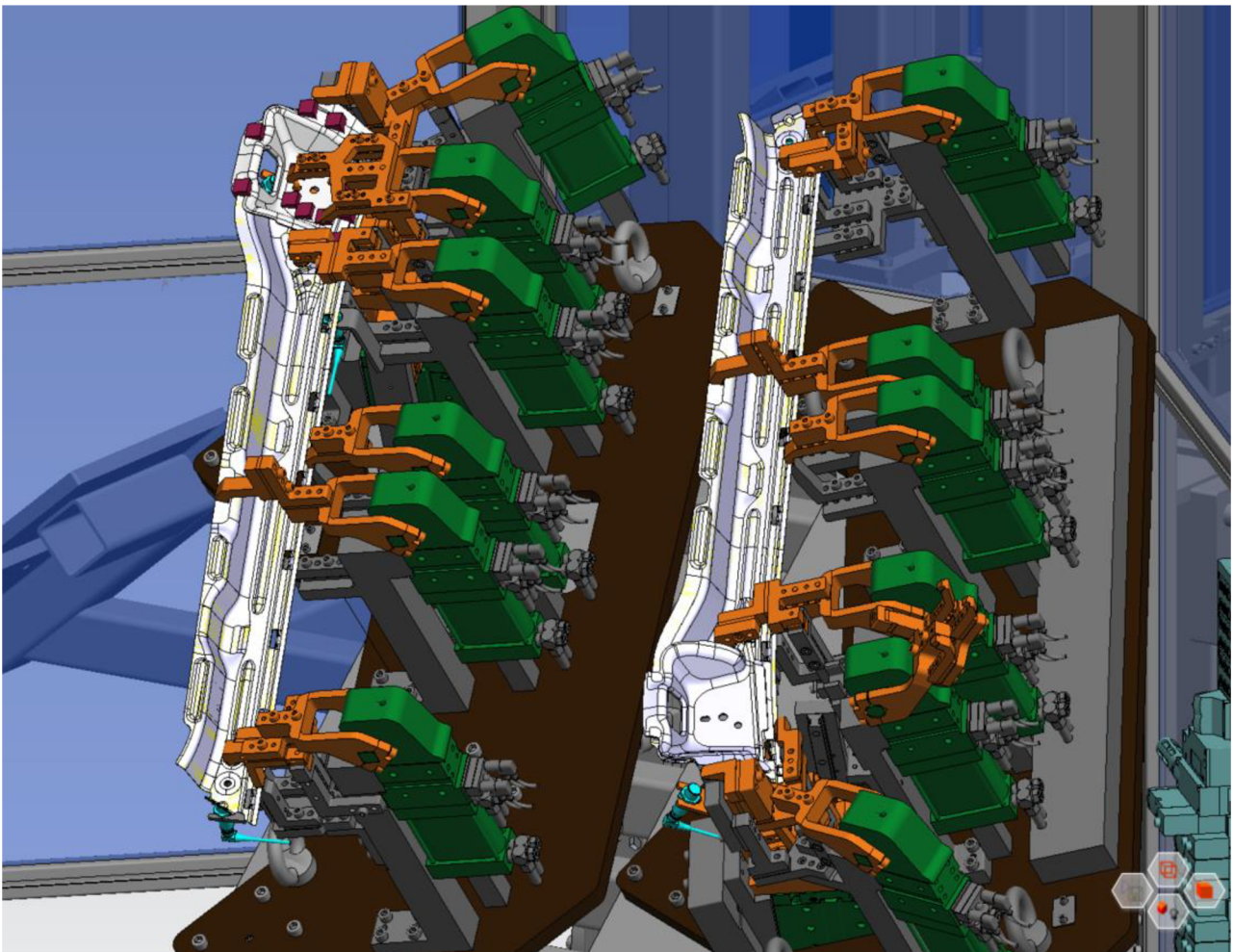
*Obr. 44 – Kinematika rotačního stolu*

Na obr. 44 lze vidět původní kinematika s definovanou HOME pozicí.

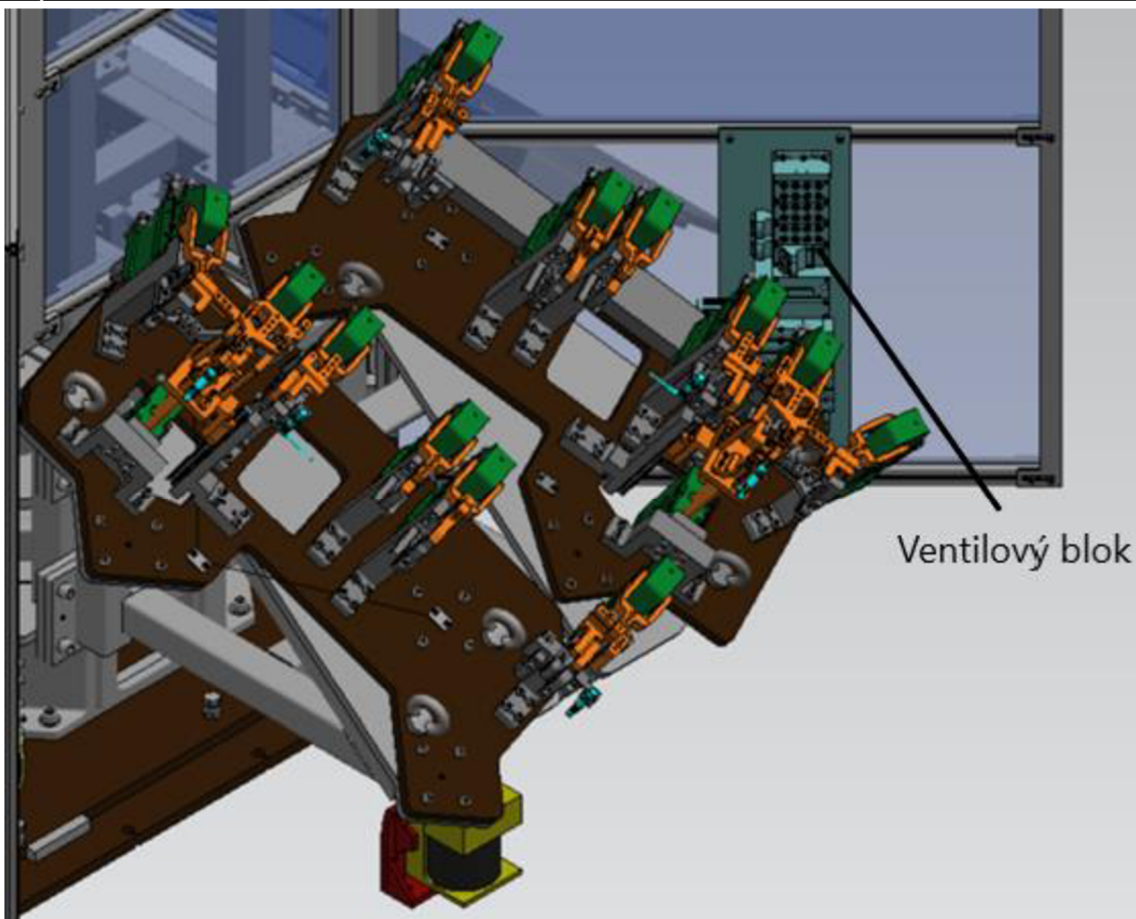
Úchopové prvky otočného stolu slouží k pevnému uchopení a vycentrování polotovaru pro zajištění správného průběhu svařovacích operací. V našem případě jsou využity pneumatické upínací prvky, zajišťující jednoduché a rychlé ovládání (obr. 45, 46, 47).



*Obr. 45 – Upinací prvky rotačního stolu polotovaru typu šachta*

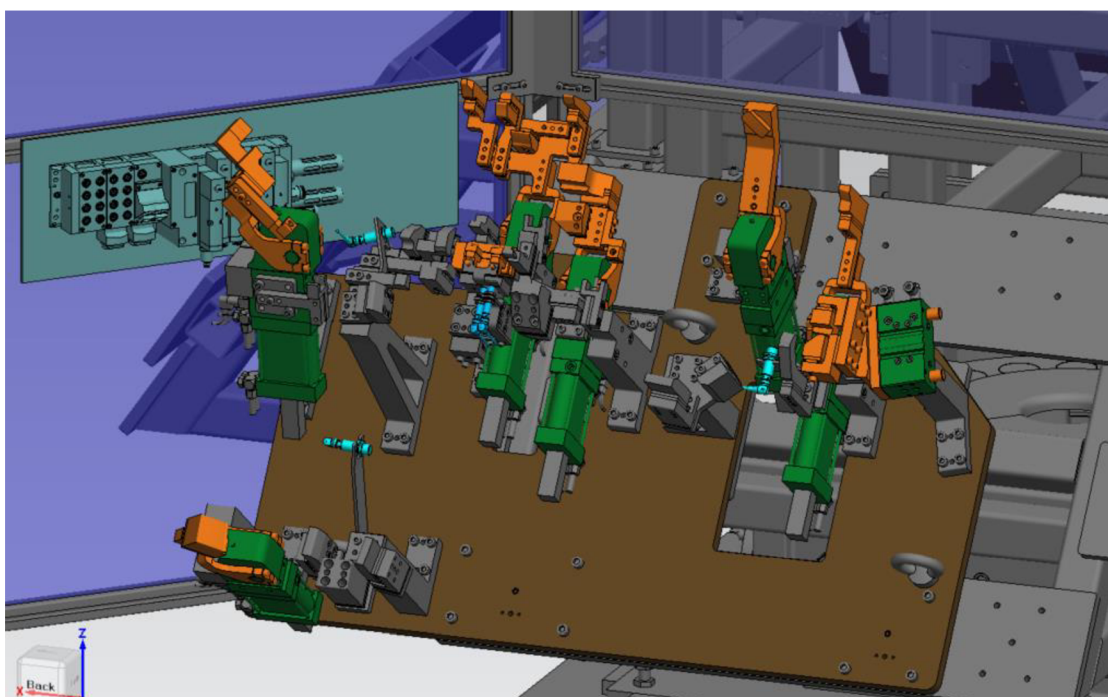


*Obr. 46 - Upinací prvky rotačního stolu polotovaru typu šachta*



Obr. 47 – Detail upinacích prvků rotačního stolu polotovaru typu šachta

Na obr. 47 je možné vidět vývody jednotlivých pneumatických ventilů a kompletní ventilový blok umístěný na bočním skle stolu. Problém tohoto uložení je napájení bloku. Potřebnou kabeláž bude nutné natáhnout skrze střed rotačního stolu do kabelového žlabu.



Obr. 48 – Detail na upinací prvky rotačního stolu polotovaru typu blatník

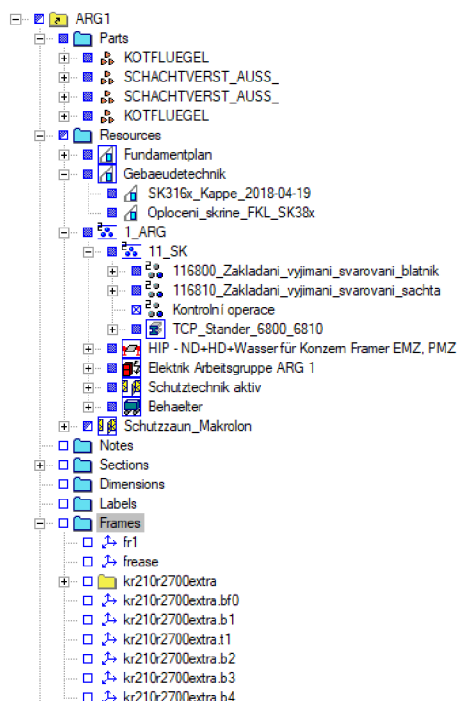
V **původním stavu** chybí u stolu i klapky veškeré operace, logika, sensorika (modely zakomponovány uvnitř modelu klapky – nedají se definovat jako senzory), signál o aktuální poloze (např. z enkodéru) a další signálová struktura.

Upínací klapky se pohybují na každé straně jako celek ve dvou pozicích OPEN/CLOSE. Chybí ještě třetí pozice WORK u jednoho z rotačních stolů, jelikož dvě klapky slouží jen jako podpora k vycentrování polotovaru při zakládání. Po založení se zavřou všechny klapky a následně se dvě klapky otevřou, aby udělaly místo pro svařovací hlavu robotu.

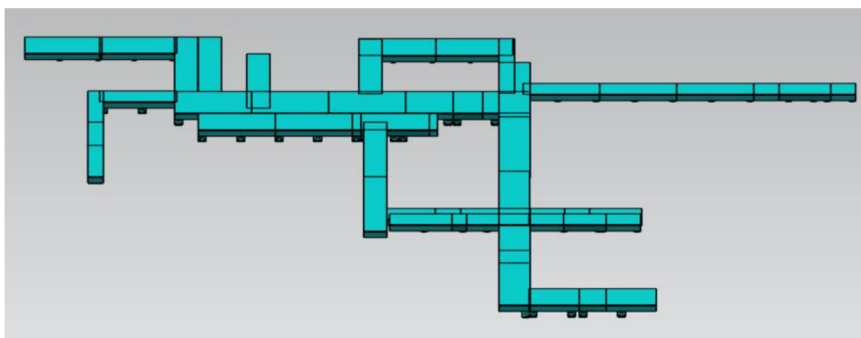
#### 4.8 Process Simulate – Object tree

Složka **parts** obsahuje jednotlivé modely polotovarů určených ke svařování. Druhá složka **Resources** obsahuje všechny ostatní komponenty v PS.

**Fundamental plan** obsahuje modely kanálů pro kabeláž.



Obr. 49 – Strom objektů projektu

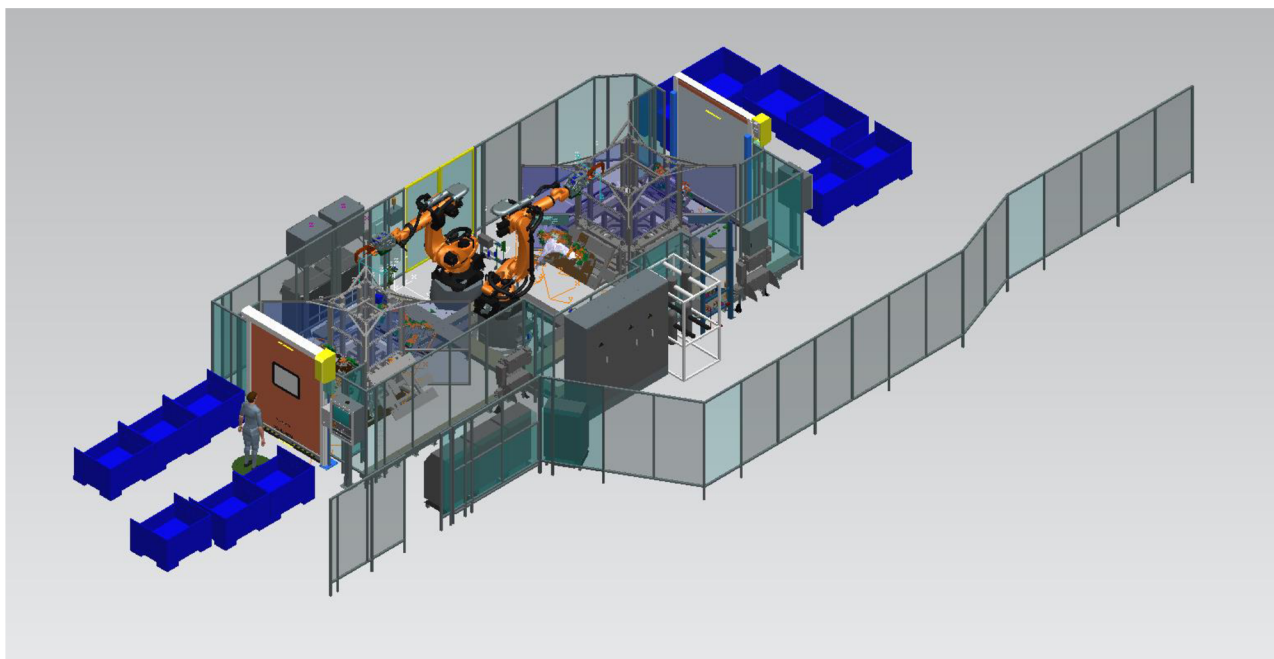


Obr. 50 – Kanálová struktura pro kabeláž po celém pracovišti

Výrobní systém je opatřen bezpečnostními prvky. Především vnějším a vnitřním oplocením. Vnější zamezuje nechtěnému vniku neoprávněných osob a odděluje systém od ostatních systémů. Vnitřní oplocení pak má za úkol plnit bezpečnostní předpisy normy ČSN



EN ISO 12100. Zajišťuje ochranu zraku osob v pracovním prostoru a zamezuje neoprávněnému vstupu na pracoviště během pracovního cyklu.



Obr. 51 – Detail na pracoviště včetně všech bezpečnostních prvků (oplocení)

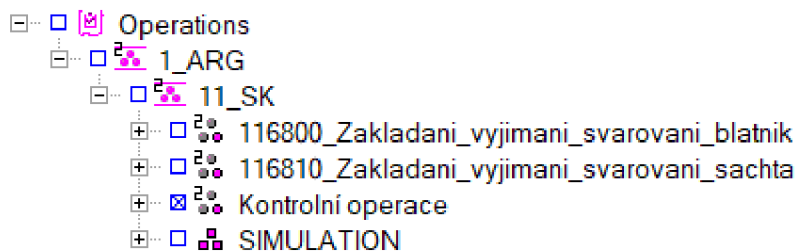
Největší složka 1\_ARG obsahuje všechny další komponenty.

- 11\_SK – roboty, svářecí kleště, rotační stoly, seřizovací stojany, elektrické rozvaděče robotů, obsluha
- Rozvody vzduchu a vzdušníky
- Hlavní elektrorozvaděč, HMI panely, bezpečnostní tlačítka u vchodů a blinkr
- Bezpečnostní dveře (vchody)
- Kontejnery polotovarů a hotových součástí

Všechny prvky jsou obsaženy bez konfigurace logiky či jakýchkoliv signálů a senzorů.

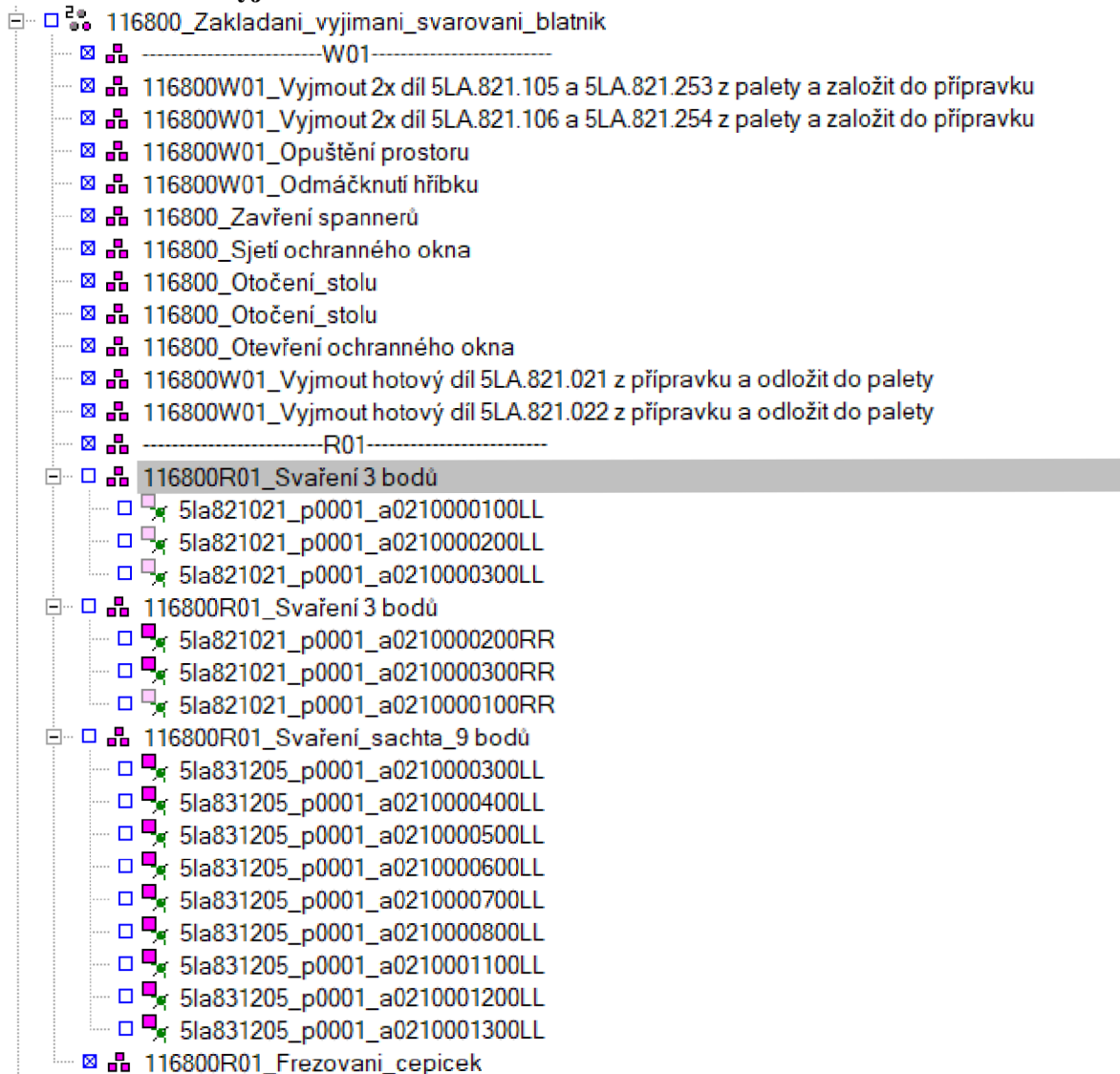
#### 4.9 Strom operací / Operation tree

Operace jsou v projektu rozděleny do čtyř základních kategorií. První dvě nám popisují fungování celého procesu, třetí upozorňuje na zohlednění požadovaného času na namátkovou kontrolu svařených dílů a poslední kategorie SIMULATION představuje dráhy robotu.



Obr. 52 – Strom operací

#### 4.9.1 Zakládání/vyjímání/svařování - blatník

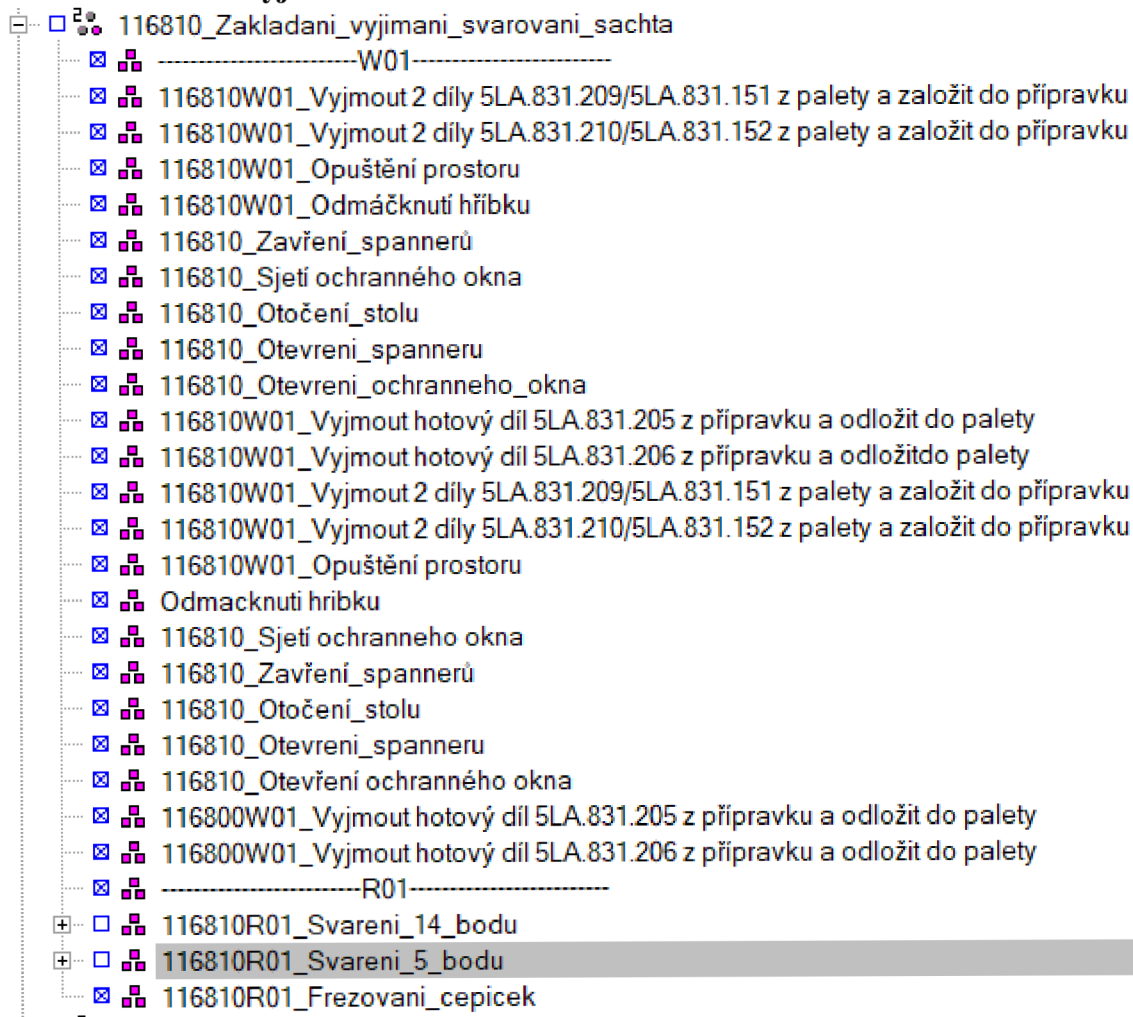


Obr. 53 – Operace typu polotovaru blatník

První část programu **W01** nám popisuje logiku fungování pracoviště z pohledu obsluhy. Ta má na starosti zakládání a vyjímání polotovarů. Pro bezpečnost obsluhy se v tomto prostoru nachází bezpečnostní dveře, bezpečnostní aretační tlačítko a laserový skener pracovního prostoru obsluhy, který zajišťuje, že se obsluha nenachází v blízkosti dveří během jejich zavírání.

Následně operace **R01** obsahuje kopie pozic svařovacích bodů prvního robotu.

## 4.9.2 Zakládání/vyjímání/svařování - šachta



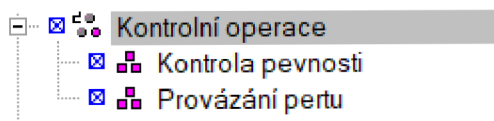
Obr. 54 - Operace typu polotovaru šachta

U druhého robotu můžeme vidět stejný sled událostí, jako u prvního. Po shrnutí pracovního cyklu zde najdeme kopie pozic svářecích bodů druhého robotu.

## 4.9.3 Kontrolní operace

Posledním krokem po svařování je kontrola pevnosti svařených dílů a provázání PERTu.

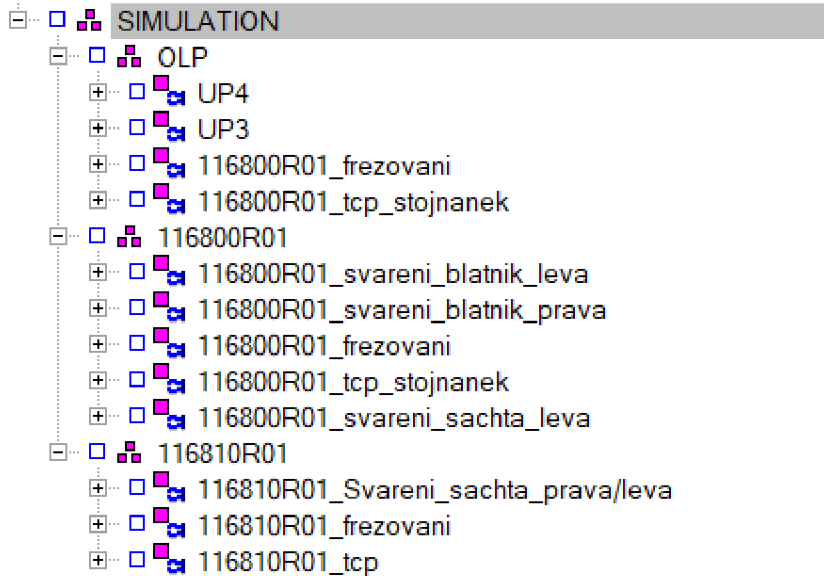
PERT (Program Evaluation and Review Technique) je metoda pro řízení projektů. Používá se pro explicitní započítání rizik do odhadu časového plánu.



Obr. 55 – Kontrolní operace

V projektu jsou tyto operace zohledněny pouze jako Compound Operation bez žádné činnosti. Slouží zde jen jako pomocné informace, jejichž časovou hodnotu lze definovat dle našich předpokladů či výpočtů. Ve virtuálním zprovoznění je důležitý především čas posledního cyklu.

#### 4.9.4 Simulace



Obr. 56 – Simulační operace

V předchozích podkapitolách byl popsán logický sled událostí. Tato podkapitola se zaměřuje na samotné svařovací simulace v PS, včetně všech potřebných svařovacích bodů a tras.

První skupina **OLP** obsahuje čtyři testovací operace ve volném prostoru. Jedná se o skupinu testovacích operací.

UP4 je operace sváření 14 bodů, odpovídajících svářecí operaci druhého robotu. Tato operace byla nevyužitelná, jelikož byla špatně umístěna v prostoru, pohybovala se na místě, kde není žádný polotovár. Jedná se o testovací operaci pro nový typ polotovaru.

UP3 obsahuje přesnou operaci určenou pro sváření 3 bodů blatníku, umístěného excentricky vlevo. Tato operace odpovídá operaci svareni\_blatnik\_leva, ale oproti ní je funkční.

Další skupina **116800R01** (strana BLATNÍK) zahrnuje operace prvního robotu (ROBOT\_B), **116810R01** (strana ŠACHTA) poté operace druhého robotu (ROBOT\_S). Obě obsahují svářecí operace, frézovací operace (seřízení svářecí hlavy), TCP operace (pro určení aktuální hodnoty TCP, eliminace chyb).

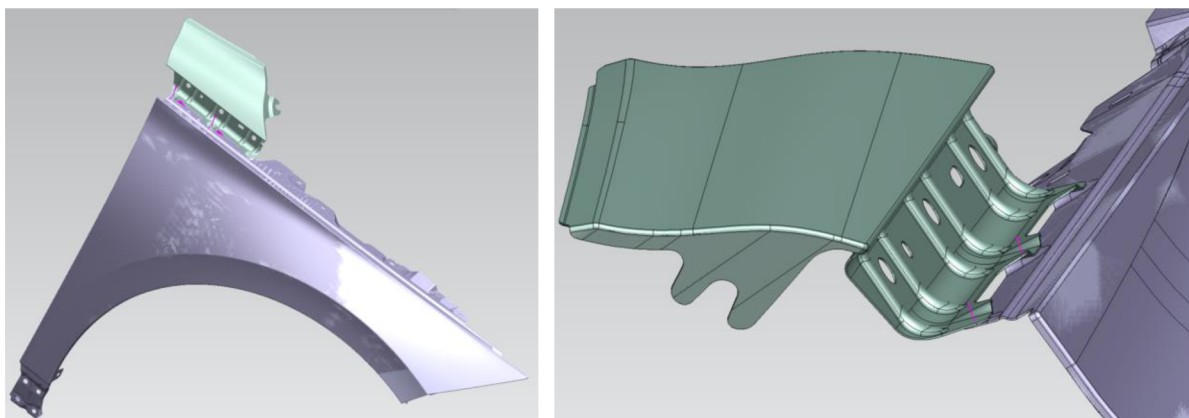
ROBOT\_S má za úkol svařit všech 14 bodů jedné šachty a 5 bodů druhé šachty, zbylých 5 bodů má z důvodu dosahu na starosti ROBOT\_B.

ROBOT\_B je určen ke sváření polotovaru typu blatník vždy ve 3 bodech. Navíc ještě vypomáhá svařit 9 bodů na polotovaru typu šachta z důvodu lepšího dosahu.

Svářecí operace robotu 116800 (ROBOT\_B), konkrétně sváření blatníku pravého (3 body) a levého (3 body), se v původně nacházelo v nefunkčním stavu. Problémem byly chybně („ustřelené“) zadané svářecí body, a také se v operacích objevovaly svářecí body určené k operaci svareni\_sachta\_leva. Poslední operace svareni\_sachta\_leva určená ke sváření 9 bodů byla již v pořádku.

Svářecí operace u druhého robotu (ROBOT\_S) se nacházely v podobném stavu, kdy musely být znovu vytvořeny některé svářecí body. U sváření polotovarů šachty by navíc docházelo ke kolizi s upínacími prvky vlivem nedefinované pracovní pozice klapek WORK.

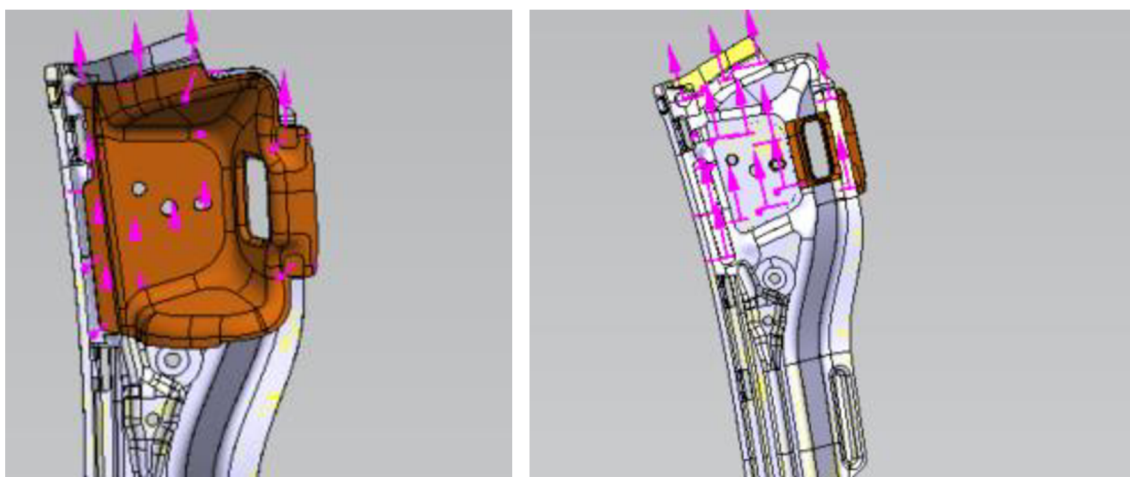
## 4.10 Bodové svařování – body svaru



a)

b)

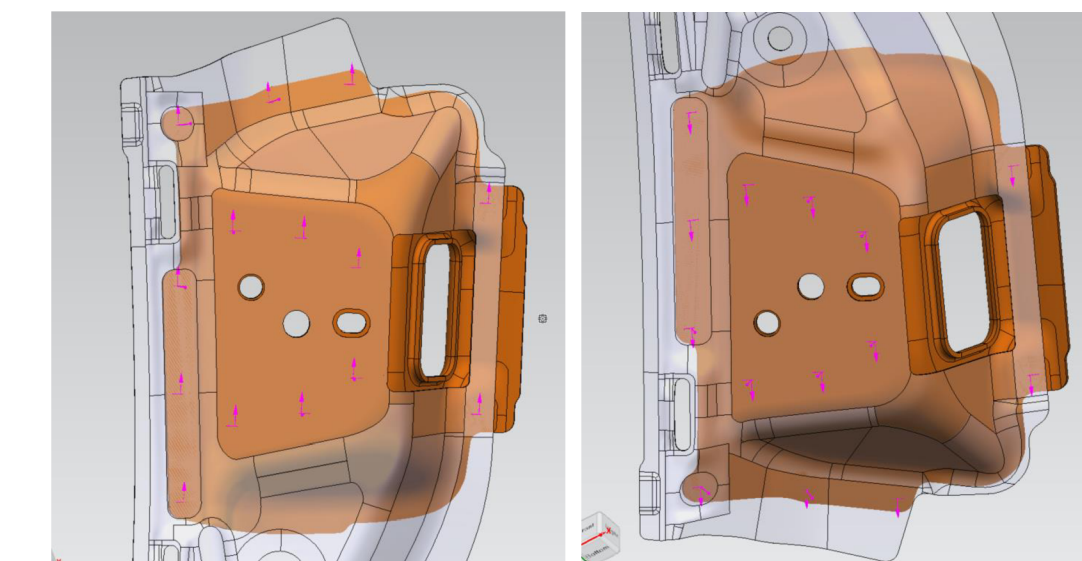
Obr. 57 - a) Svářený polotovár typu Blatník, b) Detail na svářecí body



a)

b)

Obr. 58 - a) Svářený polotovár typu šachta, b) Pohled z opačné strany



a)

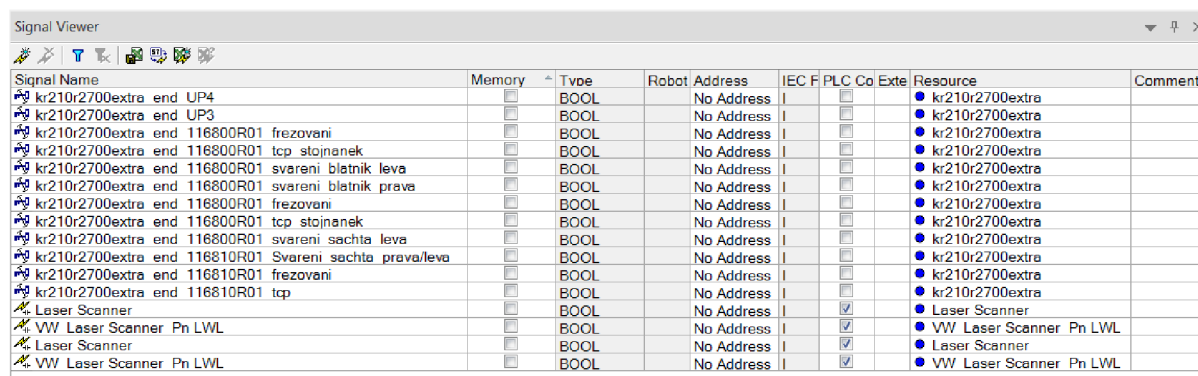
b)

Obr. 59 - a) Detail na svářecí body polotovary typu šachta, b) Detail 2

## 4.11 Signály

Z pohledu signálové struktury obsahovalo pracoviště **na počátku pouze základní KEY signály** z programu PS. Ty ovšem slouží jen k internímu fungování animací v PS, konkrétně k signalizaci o dokončení operace. Reálné signály, potřebné k řízení všech komponent procesu, včetně signálů robotu, ovšem chybí.

Nalezneme zde dva jediné užitečné signály, a to z bezpečnostního laserového skeneru pracovního prostoru obsluhy na obou stranách.



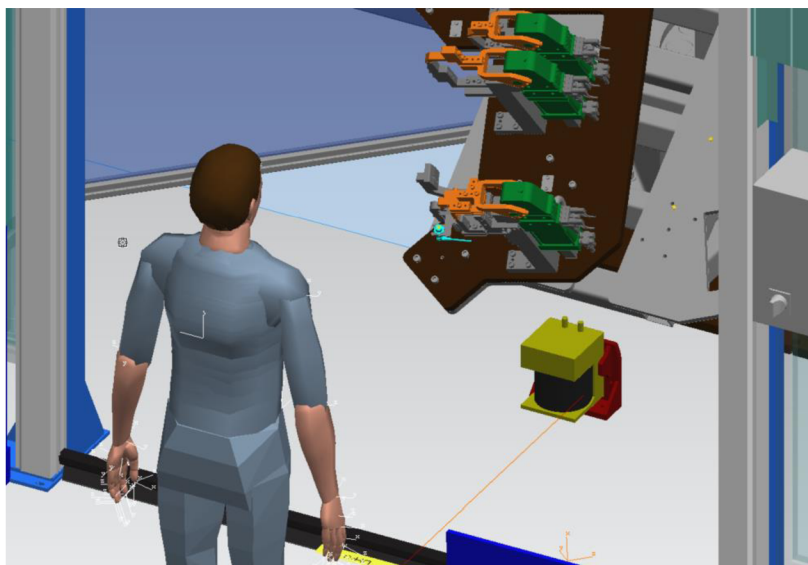
Signal Name	Memory	Tvoe	Robot	Address	IEC	F	PLC	Co	Exte	Resource	Comment
kr210r2700extra end UP4		BOOL		No Address						kr210r2700extra	
kr210r2700extra end UP3		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 frezovani		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 tcp stojanek		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 svareni blatnik leva		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 svareni blatnik prave		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 frezovani		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 tcp stojanek		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116800R01 svareni sachta leva		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116810R01 Svareni sachta prava/leva		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116810R01 frezovani		BOOL		No Address						kr210r2700extra	
kr210r2700extra end 116810R01 tcp		BOOL		No Address						kr210r2700extra	
Laser Scanner		BOOL		No Address					<input checked="" type="checkbox"/>	Laser Scanner	
VW Laser Scanner Ph LWL		BOOL		No Address					<input checked="" type="checkbox"/>	VW Laser Scanner Ph LWL	
Laser Scanner		BOOL		No Address					<input checked="" type="checkbox"/>	Laser Scanner	
VW Laser Scanner Ph LWL		BOOL		No Address					<input checked="" type="checkbox"/>	VW Laser Scanner Ph LWL	

Obr. 60 – Tabulka původní signálové struktury v PS

## 4.12 Další / Senzorika

V počátečním stavu se v PS nenacházel žádný jiný signál, než na obr. 60. V projektu se nacházely modely senzorů v upínacím prostoru, ale nebyla vytvořena žádná logika.

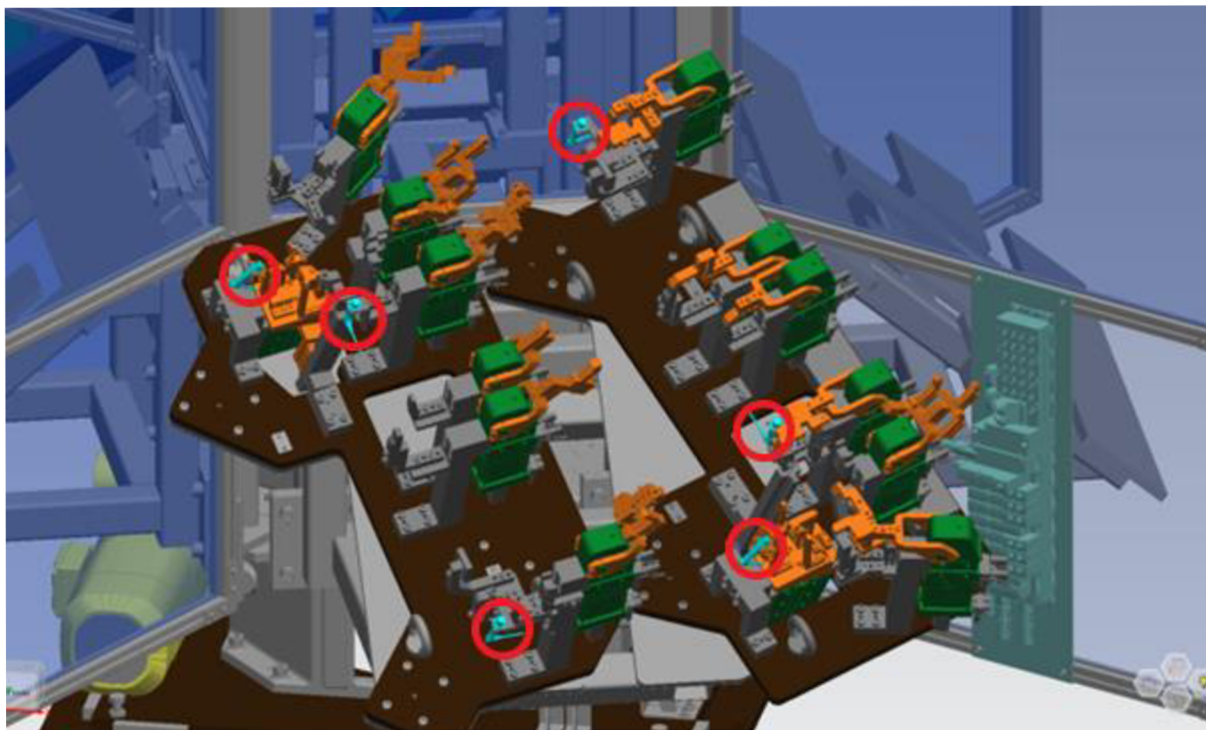
Laserový skener (obr. 61) u upínacího zařízení je bezpečnostní laserový skener. Slouží k detekci obsluhy v pracovním prostoru stolu, zabránění nechtěnému zavírání dveří a rotaci stolu.



Obr. 61 – Laserový skener pracovního prostoru obsluhy

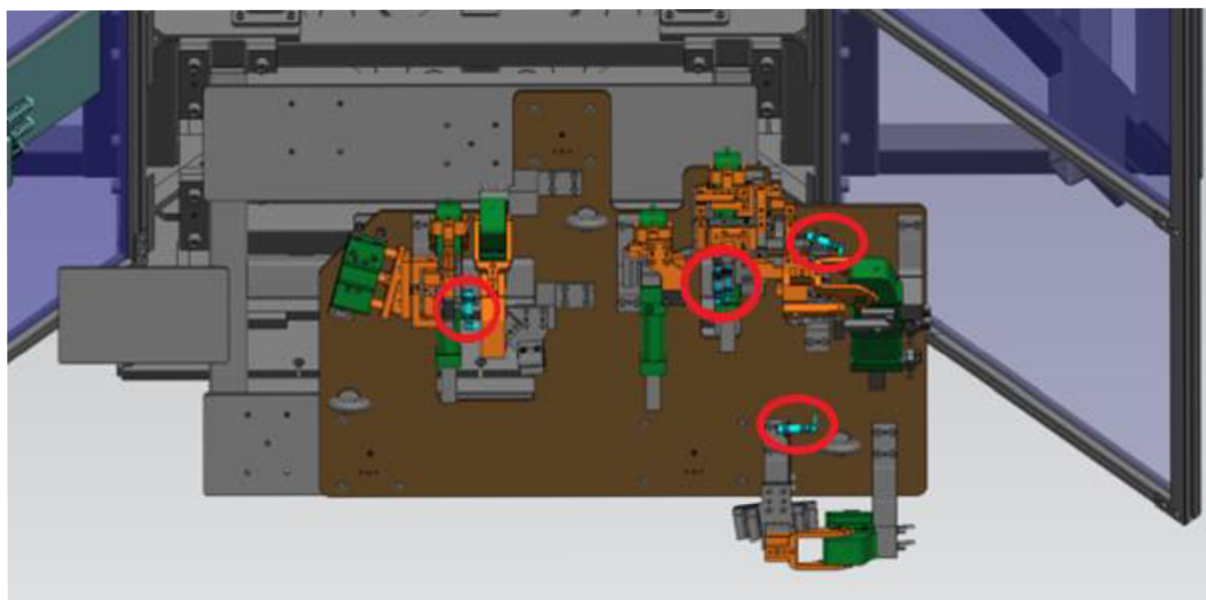
Indukční čidla jsou na rotačním stole pro polotovary typu šachta umístěna na jednom pohyblivém upínacím prvku a na dvou dalších stacionárních místech pro každý polotovar (znázorněny světle modrou barvou) (obr. 62). Pomocí jejich zpětné vazby můžeme bezpečně určit, jestli se před začátkem upínání nachází polotovar na místě, a třetí senzor potvrdí po dojetí

do polohy CLOSE přítomnost polotovaru v dalším potvrzovacím bodě. Tento postup upínání zajišťuje vysokou spolehlivost.



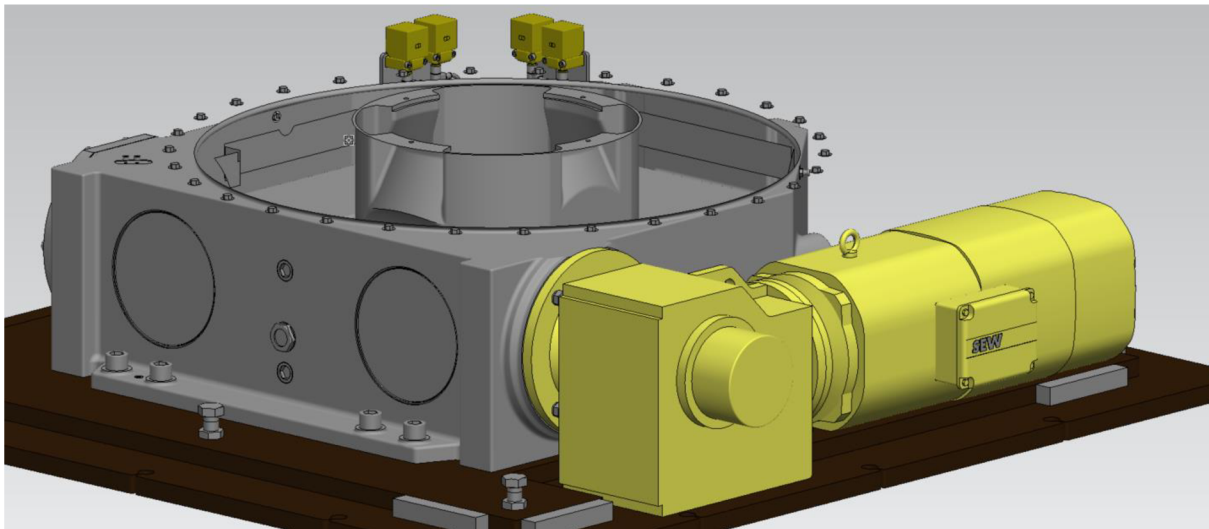
*Obr. 62 – Umístění indukčních senzorů na rotačním stole polotovaru šachta*

U druhého uchopovacího zařízení polotovaru blatník se nachází pět indukčních senzorů (obr. 63). Fungují na stejném principu jako předchozí varianta, jeden senzor je dynamický a čtyři statické. V tomto případě upínáme jen jeden polotovar.

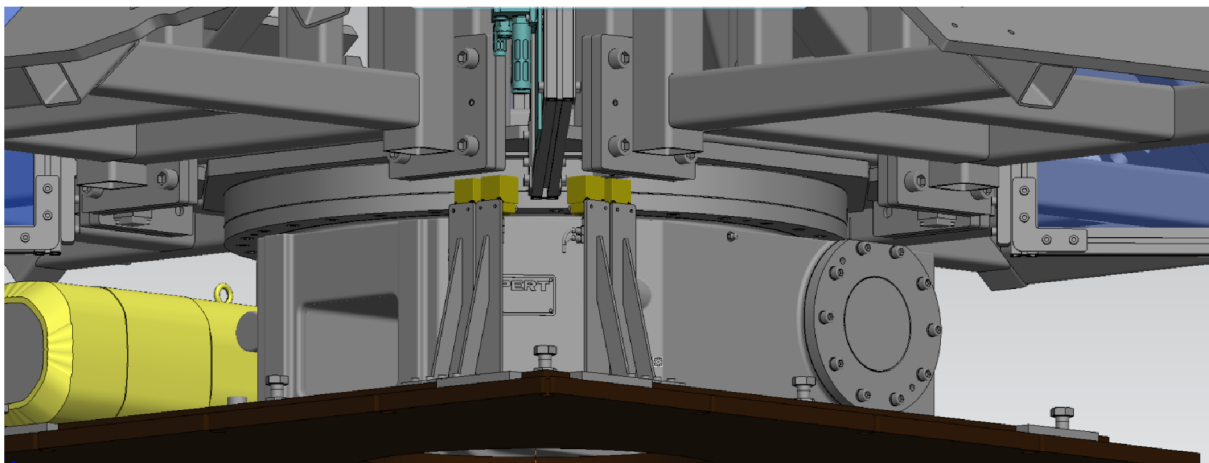


*Obr. 63 - Umístění indukčních senzorů na rotačním stole polotovaru blatník*

Senzorika u pohonu otočného stolu je tvořena čtyřmi fotoelektrickými senzory, které jsou určeny ke správnému definování koncové polohy, a tím zajišťují zpětnou vazbu nutnou k přesnému polohování stolu.

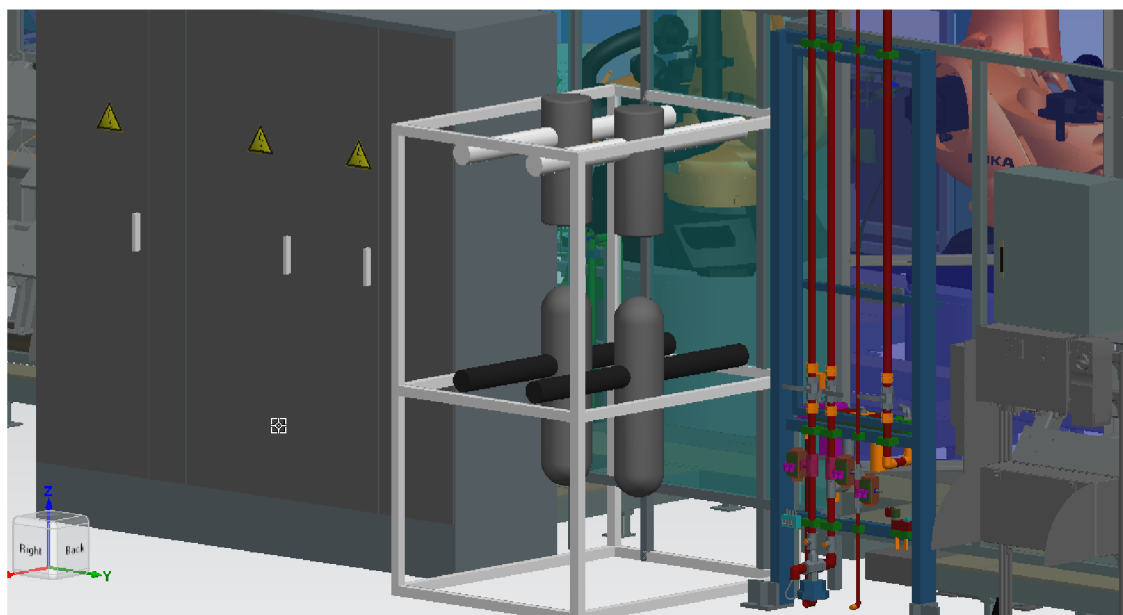


*Obr. 64 – Laserové skenery koncových poloh rotačního stolu (žlutě)*



*Obr. 65 - Laserové skenery koncových poloh rotačního stolu (žlutě)*

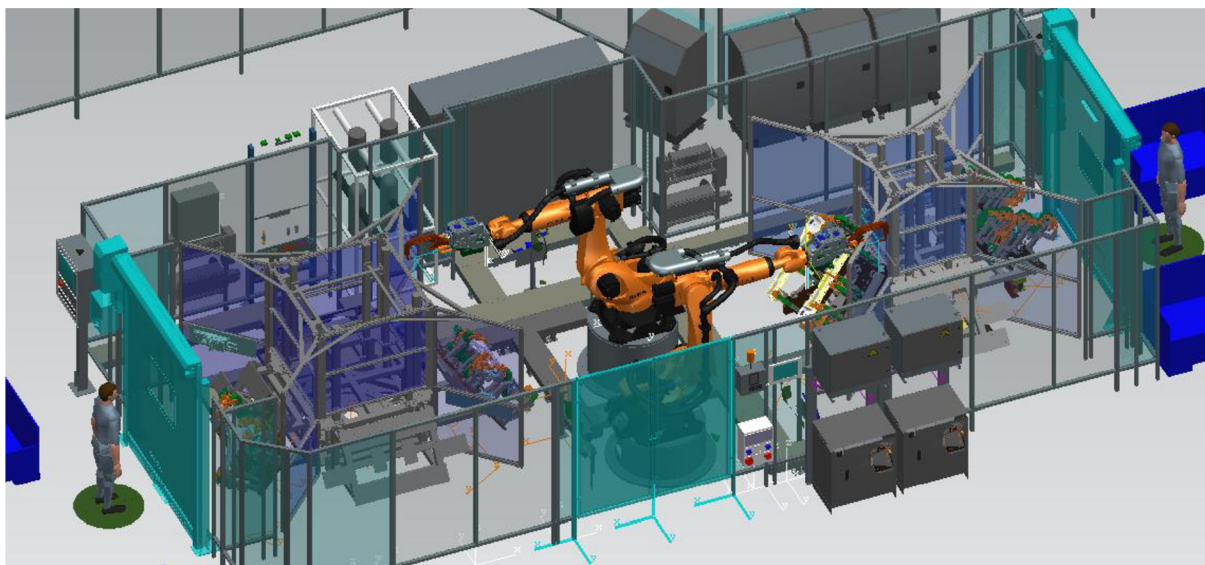
Rozvody pneumatiky, filtry a vzdušníky jsou nedílnou součástí výrobní linky.



*Obr. 66 – Rozvody pneumatiky a vzdušníky*



Vchody na pracoviště (obr. 67). Uprostřed boční, vlevo a vpravo hlavní pracovní vchody pro obsluhu.



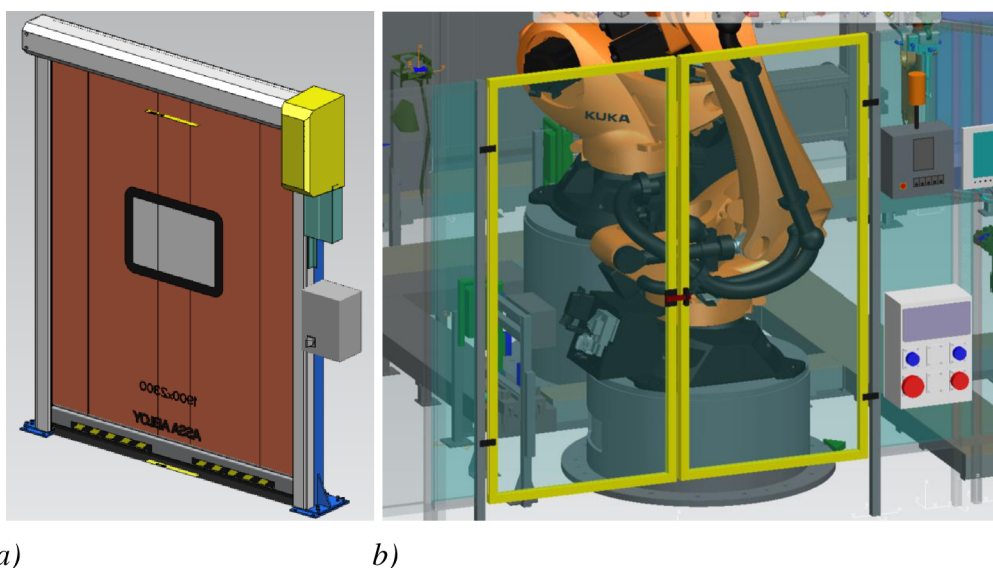
*Obr. 67- Bezpečnostní dveře pracoviště*

Bezpečnostní vysokorychlostní dveře ASSA ABLOY RR 300 PLUS (roz. 1900x2300mm). Jedná se o průmyslové rolovací dveře poháněné asynchronním motorem, řízeným pomocí integrovaného frekvenčního měniče.

Po instalaci je nutné nastavit polohy OPENED/CLOSED a žádané rychlosti pohybu. Následně je možné dveře ovládat jednoduchými příkazy OPEN/CLOSE skrze PLC a získávat zpětnou vazbu.

Dále mohou být dveře osazeny fotoelektrickými závorami a bezpečnostní lištou s protiváhou, zajišťující naprostou bezpečnost před rázovým stykem s obsluhou. V případě výpadku energie je možné ovládat dveře ručně díky integrovanému protizávaží.

Boční vchod je umožněn skrze standardní ochranné průhledné průmyslové dveře šířky 2000 mm. Bezpečnost bude řešena skrze PLC a otevření bude možné pouze po stisknutí bezpečnostního tlačítka umístěného vedle dveří.

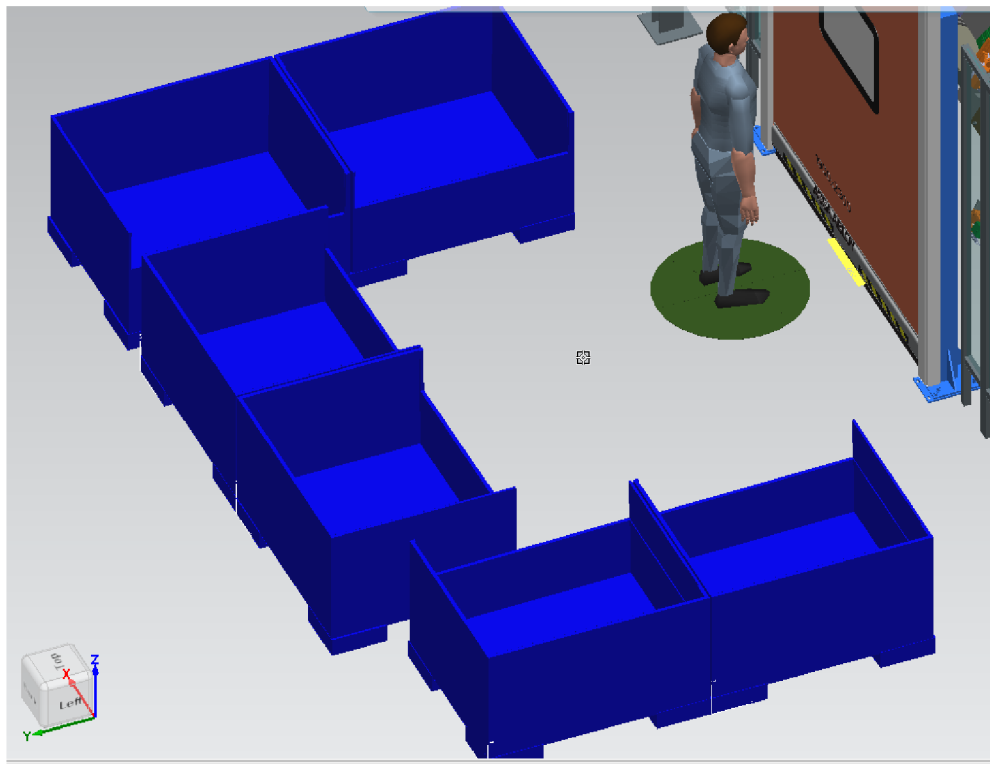


a)

b)

*Obr. 68 – a) Bezpečnostní dveře hlavní, b) Dveře boční*

Kontejnery obsahující polotovary a hotové díly. Jedná se o klasické paletové kontejnery naplněné polotovary a hotovými díly.



*Obr. 69 - Paletizace*

Analýza současného stavu robotického pracoviště přinesla detailní pochopení fungování jednotlivých prvků, z nichž se skládá, a linky jako celku. Díky těmto informacím byl také zajištěn širší pohled na celé pracoviště, což bylo nezbytné pro následné správné virtuální zprovoznění celku.

## 5 VÝBĚR VYUŽITÝCH KOMPONENT

První část práce byl věnován rozboru současného pracoviště, a byl určen počáteční bod. Dalším krokem je určení dodatečných prvků, které potřebujeme znát před započítáním virtuálního zprovoznění. Jedná se především o volbu modelu PLC, množství řízených frekvenčních měničů a způsob řízení virtuálního kontroléru.

### 5.1 Typ PLC

V této práci budou využity k virtuálnímu zprovoznění programy firmy SIEMENS (TIA Portal a Process Simulate). Z toho důvodu bylo zvoleno PLC od stejné firmy.

V současné době se můžeme setkat se třemi řadami PLC, a to: S7-300, S7-1200 a S7-1500. S modelem 300 se setkáme již jen ve starých továrnách a dále se nevyrábí. Moderní model 1500 nahradil dřívější variantu 1200 a stal se standardní vlajkovou lodí firmy SIEMENS.

V práci bude využit konkrétně model SIMATIC S7-1500 Compact CPU 1511C-1PN. Jedná se o modulové PLC střední cenové kategorie.

Model Compact obsahuje:

- Operační paměť 175 KB pro programy
- Operační paměť 1 MB pro data
- Modul 16xDI
- Modul 16xDO
- Modul 5xAI a 2xAO
- 6x vysokorychlostní čítač
- 4x vysokorychlostní výstupy pro PWM modulaci
- 2x Porty PROFINET

### 5.2 Frekvenční měniče

Robotické pracoviště obsahuje několik měničů. Obě bezpečnostní dveře obsahují měniče, které jsou ovšem řízeny interně a pro uživatele standardně programově neměnné. Komunikace s PLC zde funguje skrze jednoduché povely OPEN/CLOSE a zpětná vazba skrze OPENED/CLOSED a ERROR. Jelikož v projektu uvažujeme laserový skener pracovního prostoru obsluhy, není tedy nutné dveře osazovat laserovou bezpečnostní sítí.

Další dva měniče jsou využity k řízení asynchronních motorů pohánějících rotační stůl. Pokud bychom chtěli pouze řídit frekvenci motorů, bylo by možné využít pouze jeden, ale přišli bychom o PID regulaci polohy. Z toho důvodu byly zvoleny dva frekvenční měniče.

V rámci práce byly zvoleny měniče od firmy SIEMENS, konkrétně G120. Firma nabízí primárně měniče řady S120 a G120. Varianta G120 je ve většině případů levnější a v aplikaci řízení polohy rotačních stolů s nízkou dynamikou naprosto postačující.



a)

b)

Obr. 70 – Využití a) PLC, b) frekvenční měnič G120, c) HMI

### 5.3 PLC program

K naprogramování zvoleného PLC S7-1500 byl využit doporučený software od firmy SIEMENS TIA Portal v16. Jedná se o jedno z nejpoužívanějších programovacích prostředí pro PLC.

Kromě běžného rozhraní bylo také využito rozšíření SINAMICS Startdrive v16. Jedná se o rozšíření pro TIA Portal umožňující práci s frekvenčními měniči a technologickými objekty (TO). Kromě řízení v reálném čase také umožňuje vytvoření tzv. virtuálních os. Díky tomu jsme schopni komplexně řídit proces co nejpodobněji reálnému provedení a při aplikaci je program PLC již předpřipraven, čímž šetří čas nutný ke změně programu.

## 6 VIRTUÁLNÍ PRACOVIŠTĚ

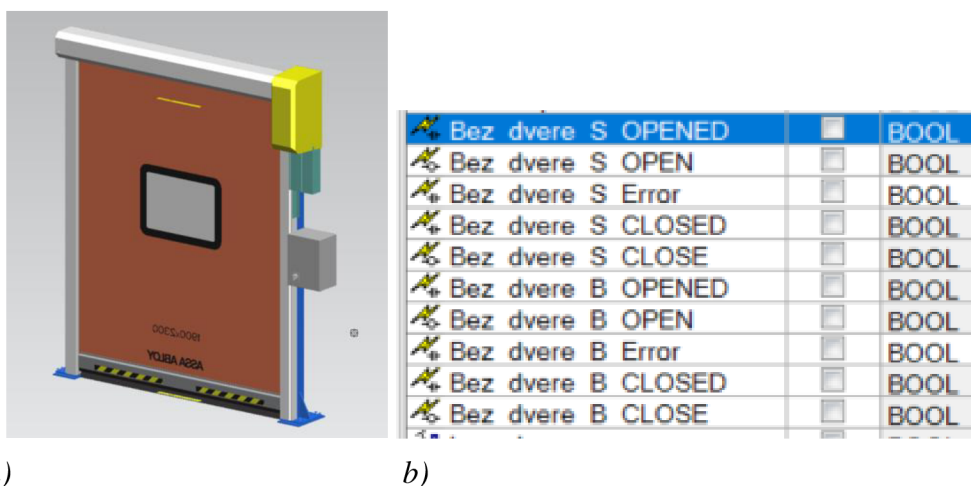
Po výběru využitých komponent, byla práce připravena k započetí virtuálního zprovoznování, které můžeme v zásadě rozdělit do dvou kategorií. Jedná se o virtuální pracoviště a virtuální kontrolér.

Pracoviště má za úkol vytvořit vizuální simulaci zadané problematiky, včetně vytvoření logiky a signálové struktury mezi jednotlivými prvky. Cílem je vytvořit simulaci, která je co nejvěrnější skutečnosti, včetně signálové struktury.

K simulaci byl použit program **Process Simulate**, jehož signálová logika je později ovládaná skrze řídicí PLC, simulované skrze software **PLC SIM Advanced v2**.

### 6.1 Bezpečnostní dveře hlavní

Před zapojením průmyslových dveří do provozu je v reálném prostředí nutné provést prvotní kalibraci. Prvním krokem byla definice pracovních poloh (standardně volíme pouze dvě) a následně volíme rychlost otevírání/zavírání, popřípadě je možné povolit/zakázat funkce typu Soft Start atd. Samotné ovládání dveří je poté limitováno pouze na povelové signály OPEN/CLOSE a stavové OPENED/CLOSED.



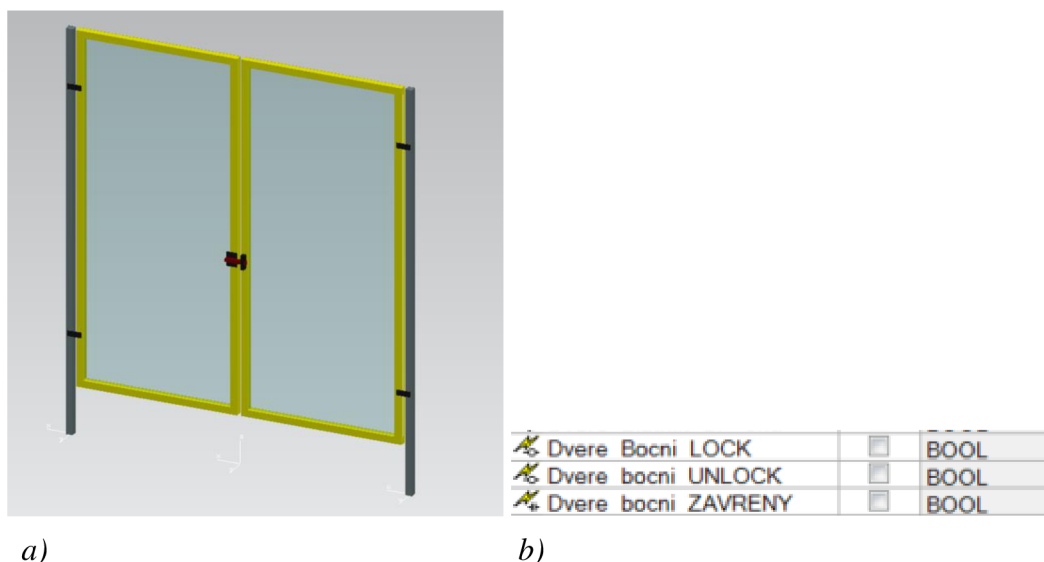
Obr. 71 – a) Bezpečnostní dveře hlavní, b) Signálová struktura

Výrobce nabízí provedení s fotoelektrickými závory pro zajištění vyšší bezpečnosti obsluhy. V pracovním prostoru uvažujeme laserový skener, zamezující neoprávněnému zavření. Z toho důvodu nejsou závory nezbytné, ale mohou fungovat jako sekundární bezpečnostní prvek ke zvýšení bezpečnosti. V práci se uvažuje s laserovým skenerem i se závory, vysílajícími chybový signál Error. Signálovou strukturu každých dveří poté představuje pět signálů.

### 6.2 Bezpečnostní dveře boční

Boční vchod k robotickému pracovišti je tvořen jednoduchými pantovými dveřmi se stínícím sklem. Z bezpečnostních důvodů je zde uvažován elektrický zámek, který se nachází v normálním stavu v pozici LOCK.

Signálová logika je tvořena ovládacími signály LOCK/UNLOCK a zpětnovazebním signálem CLOSED z indukčního snímače.



Obr. 72 - a) Bezpečnostní dveře boční, b) Signálová struktura

### 6.3 Rotační upínací zařízení

V reálném provedení by elektropohon stolu byl ovládán skrze frekvenční měnič a skrze řídicí instrukce PLC. Toto provedení bylo následně realizováno v softwaru TIA Portal.

V rámci softwaru Process Simulate uvažujeme jednoduchou signálovou logiku MOVE TO a zpětnou vazbu o dosažení polohy. Toto řešení je zcela dostačující pro řízení procesu v automatickém režimu. Využití manuálního režimu PLC je předpokládáno pouze za účelem instalace, kalibrace a servisu.

	Rot Stul B At Pos1 RVen	<input type="checkbox"/>	BOOL	129.7	I129.7	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul B At Pos2 LVen	<input type="checkbox"/>	BOOL	130.0	I130.0	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul B position	<input type="checkbox"/>	REAL	No Address	I	<input type="checkbox"/>	
	Rot Stul B ToPos1 RVen	<input type="checkbox"/>	BOOL	127.7	Q127.7	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul B ToPos2 LVen	<input type="checkbox"/>	BOOL	128.0	Q128.0	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul S At Pos1	<input type="checkbox"/>	BOOL	129.5	I129.5	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul S At Pos2	<input type="checkbox"/>	BOOL	129.6	I129.6	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul S position	<input type="checkbox"/>	REAL	132	I132	<input type="checkbox"/>	ROBOT
	Rot Stul S ToPos1	<input type="checkbox"/>	BOOL	128.2	Q128.2	<input checked="" type="checkbox"/>	ROBOT
	Rot Stul S ToPos2	<input type="checkbox"/>	BOOL	128.1	Q128.1	<input checked="" type="checkbox"/>	ROBOT

Obr. 73 – Rotační stůl – signálová struktura

Pokud bychom požadovali propojení aktuální hodnoty natočení stolu, bylo by nutné odesílat hodnotu o aktuální poloze virtuální osy v TIA portálu do softwaru Process Simulate a cyklicky vyvolávat operaci MOVE TO POSITION, což by způsobilo značné zatížení procesoru. V zájmu plynulosti simulace je uvažováno standardní řešení, kdy nastavíme rychlost animace na stejnou hodnotu jako rychlost virtuálních pohonů v TIA.

#### 6.3.1 Upínací prvky

Počet a tvar upínacích prvků se liší dle typu polotovaru. Stůl s polotvary typu šachta obsahuje dvojici upínacích ploch v každé pozici, kde každá plocha obsahuje sedm upínacích klapek. Celkově tedy každá pozice obsahuje čtrnáct klapek.

Oproti tomu stůl s polotvary typu blatník má pouze jednu upínací plochu, v každé pozici s osmi klapkami.

Jednotlivé pneumatické ventily jsou umístěny do centrálního ventilového bloku (VB), umístěného u každé upínací plochy rotačního stolu (celkem šest). V práci uvažujeme VB od firmy SMC s komunikačním modulem EX 260 (systém k sériovému přenosu dat). Komunikace mezi řídicím PLC a VB může probíhat přes celou řadu komunikačních protokolů, v práci probíhá komunikace skrze Profinet s šestnáctibitovým analogovým vstupem a výstupem.

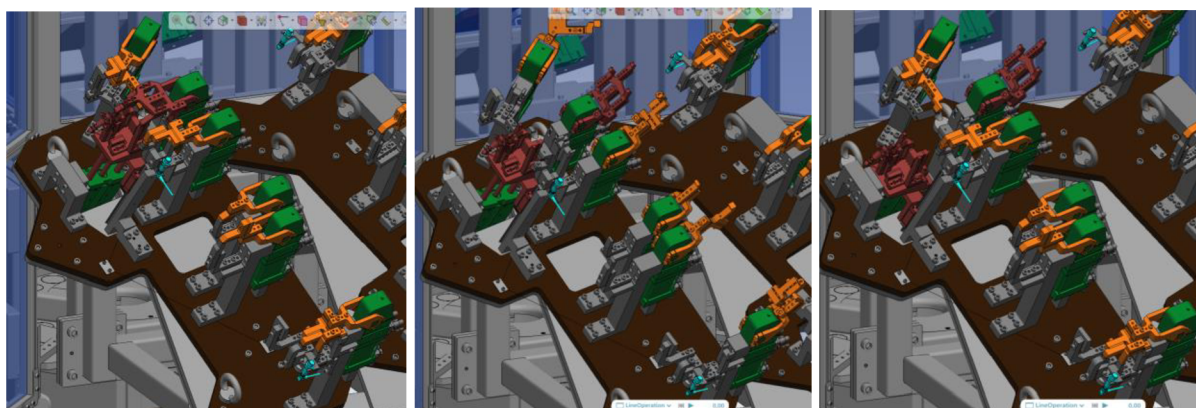
Dále existují **dvě možnosti uspořádání VB**. V první variantě jsou všechny klapky řízeny skrze jeden centrální ventil. Výhodou je cena (méně fyzických ventilů), nevýhodou pouze dvě centrální polohy a žádná zpětná vazba z jednotlivých klapek.

Druhou variantou je využití jednotlivých ventilů ke každé klapce. Výhodou je zpětná vazba z každé klapky (vyšší bezpečnost) a možnost více stavů než jen OPEN/CLOSE. Nevýhodou jsou pořizovací náklady a zástavbový prostor.

V práci byla vybrána druhá varianta. Důvodem byl také fakt, že je kromě standardní pozice OPEN/CLOSE u uchopovacích prvků typu šachta vyžadována pozice WORK. Při upínání polotovaru jsou všechny klapky v pozici OPEN, kromě klapky č. 1. Ta obsahuje tři bodce určené ke správnému prvotnímu umístění polotovaru. Po umístění polotovaru dojde k zavření všech klapek a následnému otevření klapky č. 1 a 4, díky čemuž se uvolní místo pro svářecí hlavu robotu.

POZICE/KLAPKY	OPEN	CLOSE	WORK
Klapka 1	0	0	1
Klapka 2	1	0	0
Klapka 3	1	0	0
Klapka 4	1	0	1
Klapka 5	1	0	0
Klapka 6	1	0	0
Klapka 7	1	0	0

Tabulka 2 – Tabulka stavů „klapek“ v jednotlivých pozicích. 1 – OPEN, 0 - CLOSE



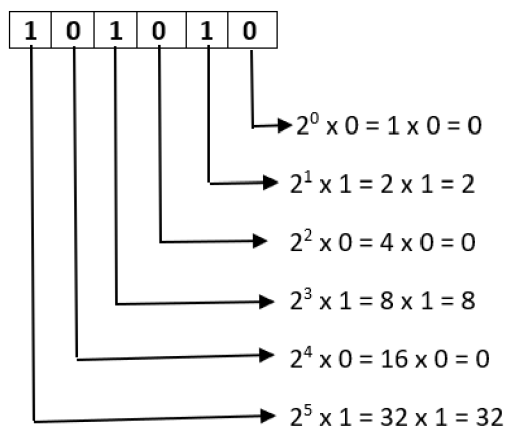
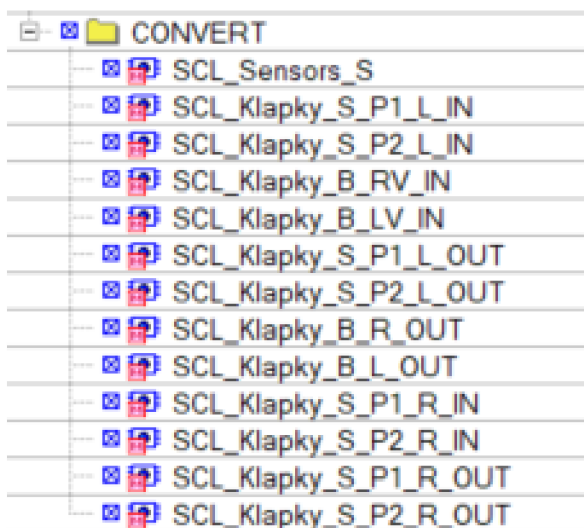
Obr. 74 – Up. Prvky v pozici a) CLOSED, b) OPENED, c) WORK

Pokud by v praxi došlo k rozhodnutí přejít k první variantě, stačí jen lehce upravit program, což by v opačném případě bylo komplikovanější.

Z pohledu PS bylo dále nutné převést zpětnovazební (STATUS) signály (OPENED/CLOSED) z jednotlivých klapek datové proměnné BOOL do centrální proměnné typu WORD (16bitová proměnná odpovídající přenosu komunikačního modulu VB), kde každý

bit představuje stavovou BOOL proměnnou jednotlivého upínacího prvku. Následně bylo nutné zrcadlově provést operaci převodu z řídicí (CONTROL) proměnné WORD na jednotlivé kontrolní (OPEN/CLOSE) bity.

Software Process Simulate nenabízí předpřipravené podprogramy k převodu BOOL proměnné na WORD a obráceně. Proto bylo nutné tyto cyklické podprogramy naprogramovat. Flexibilita PS spočívá v podpoře Strukturovaného programovacího jazyku (SCL), který je využíván například v PLC programování.



Resultant decimal number = 0+2+0+8+0+32 = 42

a)

b)

Obr. 75 – a) Prvky řídicí dodatečnou logiku signálů, b) Ukázka převodu

```

1 // START PROGRAM
2 IF #IN_0 = 1 THEN
3     #val_IN_0 := 1;
4 ELSE
5     #val_IN_0 := 0;
6 END_IF;
7 IF #IN_1 = 1 THEN ... END_IF;
12 IF #IN_2 = 1 THEN ... END_IF;
17 IF #IN_3 = 1 THEN ... END_IF;
22 IF #IN_4 = 1 THEN ... END_IF;
27 IF #IN_5 = 1 THEN ... END_IF;
32 IF #IN_6 = 1 THEN ... END_IF;
37 IF #IN_7 = 1 THEN ... END_IF;
42 IF #IN_8 = 1 THEN ... END_IF;
47 IF #IN_9 = 1 THEN ... END_IF;
52 IF #IN_10 = 1 THEN ... END_IF;
57 IF #IN_11 = 1 THEN ... END_IF;
62 IF #IN_12 = 1 THEN ... END_IF;
67 IF #IN_13 = 1 THEN ... END_IF;
72 IF #IN_14 = 1 THEN ... END_IF;
77 IF #IN_15 = 1 THEN ... END_IF;
82
83 #OUT_WORD := #val_IN_0+ #val_IN_1 +
84 #val_IN_2 + #val_IN_3 + #val_IN_4 +
85 #val_IN_5 + #val_IN_6 + #val_IN_7 +
86 #val_IN_8+ #val_IN_9+ #val_IN_10+
87 #val_IN_11+ #val_IN_12+ #val_IN_13+
88 #val_IN_14+#val_IN_15;|
89 // END PROGRAM

```

a)

```

1 // START PROGRAM
2 //
3 #temp_WORD := #IN_WORD;
4 IF #temp_WORD >= 32768 THEN
5     #OUT_15 := 1;
6     #temp_WORD := #temp_WORD - 32768;
7 ELSE
8     #OUT_15 := 0;
9 END_IF;
10 IF #temp_WORD >= 16384 THEN ... END_IF;
16 IF #temp_WORD >= 8192 THEN ... END_IF;
22 IF #temp_WORD >= 4096 THEN ... END_IF;
28 IF #temp_WORD >= 2048 THEN ... END_IF;
34 IF #temp_WORD >= 1024 THEN ... END_IF;
40 IF #temp_WORD >= 512 THEN ... END_IF;
46 IF #temp_WORD >= 256 THEN ... END_IF;
52 IF #temp_WORD >= 128 THEN ... END_IF;
58 IF #temp_WORD >= 64 THEN ... END_IF;
64 IF #temp_WORD >= 32 THEN ... END_IF;
70 IF #temp_WORD >= 16 THEN ... END_IF;
76 IF #temp_WORD >= 8 THEN ... END_IF;
82 IF #temp_WORD >= 4 THEN ... END_IF;
88 IF #temp_WORD >= 2 THEN ... END_IF;|
94 IF #temp_WORD >= 1 THEN ... END_IF;
101 // END PROGRAM

```

b)

Obr. 76 – Program pro převod a) BITS → WORD, b) WORD → BITS



Na obr. 76 a) Se nachází jednoduchý program pro převod proměnných typu BOOL na WORD. Jelikož WORD je v PS pouze v dekadickém tvaru, program funguje na principu sčítání váhových hodnot, které jednotlivé bity WORDU představují (obr. 75 b). Jedná se o spolehlivé jednoduché řešení. Sofistikovanější variantou by bylo použití cyklické funkce FOR, bohužel SCL editor v PS neefektivně pracuje s proměnnými typu Array, které jsou v tomto cyklu vhodné pro použití. Z toho důvodu byla zvolena původní primitivní a stabilní varianta.

Na obr. 76 b) se nachází další jednoduchý program pro převod proměnných typu WORD na BOOL. Program funguje stejně jako výše zmíněný převod, avšak zrcadlově. Dočasná proměnná temp\_WORD uloží aktuální hodnotu vstupního WORDu a provede porovnání s hodnotou posledního (15.) bitu. Jestliže je hodnota větší, znamená to, že bit č. 15 je TRUE, dojde k odečtení jeho hodnoty z temp\_WORD a program pokračuje v porovnávání dalších bitů. Pokud hodnota není větší, bit je rovný nule a program pokračuje dál. V softwaru TIA Portal by bylo vhodnější sofistikovanější řešení zmíněné výše, ale v PS se jedná o optimální řešení.

Celkově odesíláme z šesti VB dvanáct analogových signálů. Strana blatník obsahuje pouze dvě upínací plochy a postačují dva VB (celkem čtyři analogové signály). U strany šachta bylo nutné využít dva VB v každé poloze (celkem osm analogových signálů).

### 6.3.2 Senzory upínacího prostoru

Některé upínací prvky byly osazeny indukčními snímači, určenými ke zpětné vazbě o správně upnutém polotovaru. Každá šachta je osazena třemi senzory a blatník pěti. Signály jsou ve stavu TRUE, jakmile se v blízkosti nachází polotovary. Uvažujeme standardní zapojení každého senzoru zvlášť. Druhou variantou by bylo například využití komunikační jednotky IO-link, která by sbírala data senzorů a odesílala jejich stavy skrze jediný kabel.

Sen B P1 1	<input type="checkbox"/>	BOOL	124.0	I124.0	<input checked="" type="checkbox"/>	ROBOT
Sen B P1 2	<input type="checkbox"/>	BOOL	124.1	I124.1	<input checked="" type="checkbox"/>	ROBOT
Sen B P1 3	<input type="checkbox"/>	BOOL	124.2	I124.2	<input checked="" type="checkbox"/>	ROBOT
Sen B P1 4	<input type="checkbox"/>	BOOL	124.3	I124.3	<input checked="" type="checkbox"/>	ROBOT
Sen B P1 5	<input type="checkbox"/>	BOOL	124.4	I124.4	<input checked="" type="checkbox"/>	ROBOT
Sen B P2 1	<input type="checkbox"/>	BOOL	124.5	I124.5	<input checked="" type="checkbox"/>	ROBOT
Sen B P2 2	<input type="checkbox"/>	BOOL	124.6	I124.6	<input checked="" type="checkbox"/>	ROBOT
Sen B P2 3	<input type="checkbox"/>	BOOL	124.7	I124.7	<input checked="" type="checkbox"/>	ROBOT
Sen B P2 4	<input type="checkbox"/>	BOOL	125.0	I125.0	<input checked="" type="checkbox"/>	ROBOT
Sen B P2 5	<input type="checkbox"/>	BOOL	125.1	I125.1	<input checked="" type="checkbox"/>	ROBOT
Sen S P1 11	<input type="checkbox"/>	BOOL	125.2	I125.2	<input checked="" type="checkbox"/>	ROBOT
Sen S P1 12	<input type="checkbox"/>	BOOL	125.3	I125.3	<input checked="" type="checkbox"/>	ROBOT
Sen S P1 13	<input type="checkbox"/>	BOOL	125.4	I125.4	<input checked="" type="checkbox"/>	ROBOT
Sen S P1 21	<input type="checkbox"/>	BOOL	125.5	I125.5	<input checked="" type="checkbox"/>	ROBOT
Sen S P1 22	<input type="checkbox"/>	BOOL	125.6	I125.6	<input checked="" type="checkbox"/>	ROBOT
Sen S P1 23	<input type="checkbox"/>	BOOL	125.7	I125.7	<input checked="" type="checkbox"/>	ROBOT
Sen S P2 11	<input type="checkbox"/>	BOOL	126.0	I126.0	<input checked="" type="checkbox"/>	ROBOT
Sen S P2 12	<input type="checkbox"/>	BOOL	126.1	I126.1	<input checked="" type="checkbox"/>	ROBOT
Sen S P2 13	<input type="checkbox"/>	BOOL	126.2	I126.2	<input checked="" type="checkbox"/>	ROBOT
Sen S P2 21	<input type="checkbox"/>	BOOL	126.3	I126.3	<input checked="" type="checkbox"/>	ROBOT
Sen S P2 22	<input type="checkbox"/>	BOOL	126.4	I126.4	<input checked="" type="checkbox"/>	ROBOT
Sen S P2 23	<input type="checkbox"/>	BOOL	126.5	I126.5	<input checked="" type="checkbox"/>	ROBOT

Obr. 77 – Signály indukčních senzorů v PS

## 6.4 Roboty

Dalším neopomenutelným prvkem pracoviště jsou roboty KUKA. Virtuální zprovoznění u nich spočívá ve zprovoznění pohybových operací (vč. nastavení rychlosti, typu, zóny pohybu atd.) a vytvoření signálové struktury, která má za úkol spouštět zmíněné operace a věrně simulovat interní signálovou logiku mezi PLC a robotickým kontrolérem.

### 6.4.1 Robotické operace

Robotické operace byly v původním stavu nefunkční. Prvním krokem bylo tyto operace zprovoznit.

**První robot (ROBOT\_S)** vykonává celkem tři operace. První operace **Svareni\_Sachta\_prava/leva** spočívá ve svaření všech 14 bodů polotovaru šachta na pravé straně a 9 bodů na šachtě umístěné vlevo. Důvodem, proč pouze 5 bodů, je omezený dosah robotu. Další dvě operace jsou servisní – **Frézování** a **TCP**.

**Frézování** probíhá najetím na svařovací stojan, zde dojde k sepnutí senzoru a provede se svářecí operace. Reálná simulace sváření je nahrazena časovou prodlevou po sepnutí senzoru při dosažení frézovacího stojanu.

Poslední operace se provádí najetím robotu na měřicí stojan. Při kontaktu robot vyhodnotí aktuální pozici **TCP** a aktualizuje stávající hodnotu v kontroléru.

**Druhý robot (ROBOT\_B)** vykonává celkem pět operací. První z nich je **Svareni\_Sachta\_leva**. Jedná se o svaření zbývajících 9 bodů polotovaru šachta, které nebyl první robot schopný vykonat z důvodu dosahu.

Další dvě operace **Svareni\_Blatnik\_leva** a **Svareni\_Blatnik\_prava** se týkají polotovaru typu blatník. Dojde ke svaření 3 bodů na polotovaru. Samotné operace se liší pouze pozicí, kdy je polotovar upnut na excentricky umístěných upínacích plochách, jednou vpravo, jednou vlevo.

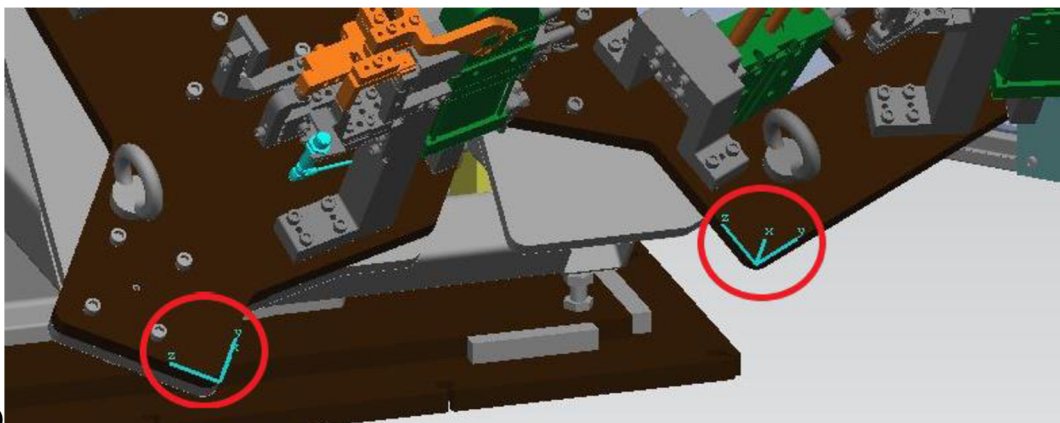
Poslední dvě operace jsou poté stejné jako u prvního robotu, tzn. **Frézování** a **TCP** a téměř se neliší. Každý robot má svůj Frézovací stojan, ale TCP stojan je pouze jeden společný.

Jednotlivé operace bylo nutné otevřít v Path Editoru a krok po kroku procházet cestu robotu. Většina chyb byla z důvodu chybně definovaných samotných svářecích operačních bodů. Ty bylo nutné odstranit a vytvořit je znovu. Po tomto postupu s každou operací byly všechny operace zprovozněny. Nakonec byla vytvořena OLP (offline programming) logika robotických operací (obr. 79) a zadány údaje o typu pohybu, rychlosti a zóně (obr. 78a).

Souřadné systémy rotačních stolů byly umístěny do spodních rohů pracovní desky z důvodu vhodné pozice k zaměření (obr. 78b). Všechny upínací prvky jsou poté vztaheny k tomuto SS.

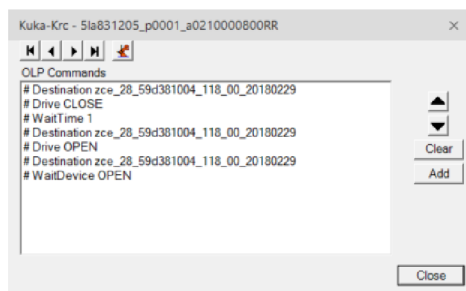
Paths & Locations	Config	Acc	Tool Nr	Base Nr	Motion	Speed	Zone	Storage...	OLP Commands	Duration
116810R01_Svare...										69.39
HOME	S2 T 10	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE			0
via	S2 T 11	100 % (d...	1 (Fr) - E...	1 - 1168...	PTP	80 %	C.DIS 5 ...			1.59
O1_P1	S2 T 10	100 % (d...	1 (Fr) - E...	1 - 1168...	LIN	0.1 m/s	C.DIS 2 ...	# Destination zce_28_59d381004_118_00_20180229 # Drive C		2.77
O1_P2	S2 T 11	100 % (d...	1 (Fr) - E...	1 - 1168...	LIN	0.1 m/s	C.DIS 2 ...	# Destination zce_28_59d381004_118_00_20180229 # Drive C		2.91

a)



b)

Obr. 78 – Definování a) bodů operací včetně OLP, b) SS up. prostoru



Obr. 79 – Využití OLP příkazy

Dalším krokem bylo ověřit, že během průchodu dráhou nedochází ke kolizím. Ověřování proběhlo pomocí funkce Collision Mode, kdy dojde k zastavení simulace a barevné výstraže kdykoliv při kontaktu robotu (včetně nástroje) s jakýmkoliv dalším objektem.

#### 6.4.2 Signálová struktura

Po zprovoznění a otestování operací bylo možné vytvořit signálovou strukturu. Stěžejní byly signály spouštěcí jednotlivé operace v PS. Dále byly předpřipraveny další signály, simulující chování reálného kontroléru robotu. Jedná se o signály Req\_Op (Required operation) a Start\_Op (Start operation).

Chování je následovné: PLC odešle signál s číslem žádané operace, robot po přijetí odešle zpětnou vazbu o obdržení hodnoty a čeká na příkaz Start\_Op = TRUE.

	Rob B OP HOME end	<input type="checkbox"/>	BOOL	No Address	I	<input type="checkbox"/>	
	Rob B OP HOME start	<input type="checkbox"/>	BOOL	131.1	Q131.1	<input checked="" type="checkbox"/>	ROBOT
	Rob S OP HOME end	<input type="checkbox"/>	BOOL	No Address	I	<input type="checkbox"/>	
	Rob S OP HOME start	<input type="checkbox"/>	BOOL	131.2	Q131.2	<input checked="" type="checkbox"/>	ROBOT
	Robot B blatnik leva start	<input type="checkbox"/>	BOOL	129.3	Q129.3	<input checked="" type="checkbox"/>	ROBOT
	Robot B blatnik prava start	<input type="checkbox"/>	BOOL	129.4	Q129.4	<input checked="" type="checkbox"/>	ROBOT
	Robot B frezovani start	<input type="checkbox"/>	BOOL	129.5	Q129.5	<input checked="" type="checkbox"/>	ROBOT
	Robot B sachta leva start	<input type="checkbox"/>	BOOL	129.6	Q129.6	<input checked="" type="checkbox"/>	ROBOT
	Robot B tcp start	<input type="checkbox"/>	BOOL	129.7	Q129.7	<input checked="" type="checkbox"/>	ROBOT
	Robot S frezovani start	<input type="checkbox"/>	BOOL	130.0	Q130.0	<input checked="" type="checkbox"/>	ROBOT
	Robot S pravaleva start	<input type="checkbox"/>	BOOL	130.1	Q130.1	<input checked="" type="checkbox"/>	ROBOT
	Robot S tcp start	<input type="checkbox"/>	BOOL	130.2	Q130.2	<input checked="" type="checkbox"/>	ROBOT

Obr. 80 – Spouštěcí signály robotických operací

Seznam předpřipravených signálů určených pro komunikaci mezi PLC a kontrolérem KUKA obr. 82 Process Simulate podporuje možnost připojení virtuálního kontroléru daného výrobce robotů, díky kterému je možné detailněji simulovat celou předpřipravenou signálovou strukturu. K práci ovšem nebyla licence k softwaru KUKA.Sim k dispozici.

V rámci co nejuvěrnější kopie programu byl vytvořen v PS SCL program, který simuluje základní logiku mezi PLC a robotickým kontrolérem (obr. 81).

```

1 IF #PNO_motorsON THEN
2     #PNI_motorsOFF := 0;
3     #PNI_motorsON := 1;
4 END_IF;
5 IF #PNO_motorsOFF THEN ... END_IF;
9
10 #PNI_ReqOp := #PNO_ReqOp;
11 #PNI_RobotWorking := 0;
12
13 IF #PNI_ReqOp = 1 AND #PNO_StartOp = 1 THEN
14     #OP_BL1 := 1;
15     #PNI_RobotWorking := 1;
16 ELSE
17     #OP_BL1 := 0;
18 END_IF;
19 IF #PNI_ReqOp = 2 AND ... THEN ... END_IF;
25 IF #PNI_ReqOp = 3 AND ... THEN ... END_IF;
31 IF #PNI_ReqOp = 4 AND ... THEN ... END_IF;
37 IF #PNI_ReqOp = 5 AND ... THEN ... END_IF;

```

Obr. 81 – Struktura podprogramu simulujícího chování robotu

Řádek	Popis
1-8	Simulace spouštění pohonů robotu ON/OFF
10	Simulace zpětné vazby robotu o obdržení signálu
11	Stavový signál RobotWorking se rovná nule. Pokud dojde dále v programu k zapnutí operace, dojde i k přepsání tohoto signálu.
13+	Pokud dojde k sepnutí signálu StartOp, dojde k zapnutí žádané operace a sepnutí stavového signálu RobotWorking.

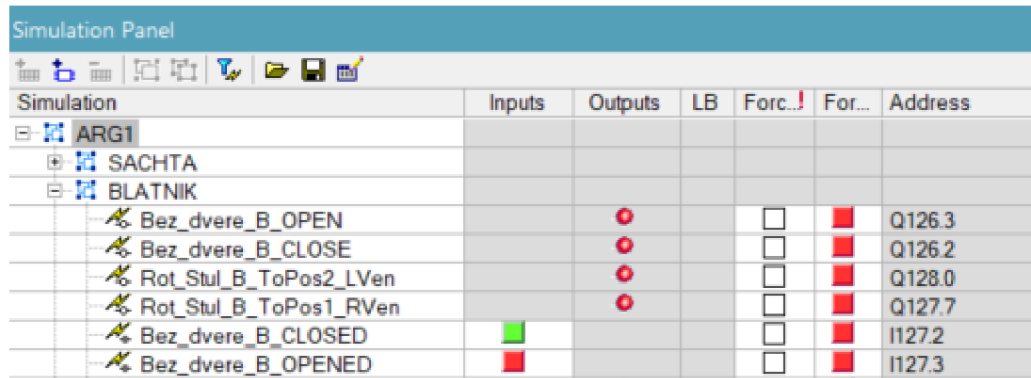
Tabulka 3 – Popis fungování podprogramu (obr. 82)

	PNO S stopAtEndOfCycle	<input type="checkbox"/>	BOOL	PNI S stopAtEndOfCycle	No Address	Q	<input type="checkbox"/>
	PNO S stop	<input type="checkbox"/>	BOOL	PNI S stop	No Address	Q	<input type="checkbox"/>
	PNO S startOp	<input type="checkbox"/>	BOOL	PNI S startOp	No Address	Q	<input type="checkbox"/>
	PNO S startAtMain	<input type="checkbox"/>	BOOL	PNI S startAtMain	No Address	Q	<input type="checkbox"/>
	PNO S start	<input type="checkbox"/>	BOOL	PNI S start	No Address	Q	<input type="checkbox"/>
	PNO S reqOp	<input type="checkbox"/>	BOOL	PNI S reqOp	No Address	Q	<input type="checkbox"/>
	PNO S motorsOn	<input type="checkbox"/>	BOOL	PNI S motorsOn	No Address	Q	<input type="checkbox"/>
	PNO S motorsOff	<input type="checkbox"/>	BOOL	PNI S motorsOff	No Address	Q	<input type="checkbox"/>
	PNO B stopAtEndOfCycle	<input type="checkbox"/>	BOOL	PNI B stopAtEndOfCycle	No Address	Q	<input type="checkbox"/>
	PNO B stop	<input type="checkbox"/>	BOOL	PNI B stop	No Address	Q	<input type="checkbox"/>
	PNO B startOp	<input type="checkbox"/>	BOOL	PNI B startOp	No Address	Q	<input type="checkbox"/>
	PNO B startAtMain	<input type="checkbox"/>	BOOL	PNI B startAtMain	No Address	Q	<input type="checkbox"/>
	PNO B start	<input type="checkbox"/>	BOOL	PNI B start	No Address	Q	<input type="checkbox"/>
	PNO B reqOp	<input type="checkbox"/>	INT	PNI B reqOp	No Address	Q	<input type="checkbox"/>
	PNO B motorsOn	<input type="checkbox"/>	BOOL	PNI B motorsOn	No Address	Q	<input type="checkbox"/>
	PNO B motorsOff	<input type="checkbox"/>	BOOL	PNI B motorsOff	No Address	Q	<input type="checkbox"/>
	PNI S runChainOk	<input type="checkbox"/>	BOOL	PNO S runChainOk	No Address	I	<input type="checkbox"/>
	PNI S robotWorking	<input type="checkbox"/>	BOOL	PNO S robotWorking	No Address	I	<input type="checkbox"/>
	PNI S reqOp	<input type="checkbox"/>	BOOL	PNO S reqOp	No Address	I	<input type="checkbox"/>
	PNI S motorsOnState	<input type="checkbox"/>	BOOL	PNO S motorsOnState	No Address	I	<input type="checkbox"/>
	PNI S motorsOffState	<input type="checkbox"/>	BOOL	PNO S motorsOffState	No Address	I	<input type="checkbox"/>
	PNI S cycleOn	<input type="checkbox"/>	BOOL	PNO S cycleOn	No Address	I	<input type="checkbox"/>
	PNI S autoOn	<input type="checkbox"/>	BOOL	PNO S autoOn	No Address	I	<input type="checkbox"/>
	PNI B runChainOk	<input type="checkbox"/>	BOOL	PNO B runChainOk	No Address	I	<input type="checkbox"/>
	PNI B robotWorking	<input type="checkbox"/>	BOOL	PNO B robotWorking	No Address	I	<input type="checkbox"/>
	PNI B reqOp	<input type="checkbox"/>	INT	PNO B reqOp	No Address	I	<input type="checkbox"/>
	PNI B motorsOnState	<input type="checkbox"/>	BOOL	PNO B motorsOnState	No Address	I	<input type="checkbox"/>
	PNI B motorsOffState	<input type="checkbox"/>	BOOL	PNO B motorsOffState	No Address	I	<input type="checkbox"/>
	PNI B cycleOn	<input type="checkbox"/>	BOOL	PNO B cycleOn	No Address	I	<input type="checkbox"/>
	PNI B autoOn	<input type="checkbox"/>	BOOL	PNO B autoOn	No Address	I	<input type="checkbox"/>

Obr. 82 – Celková signálová struktura robotů v PS

## 6.5 Simulace a propojení s PLC

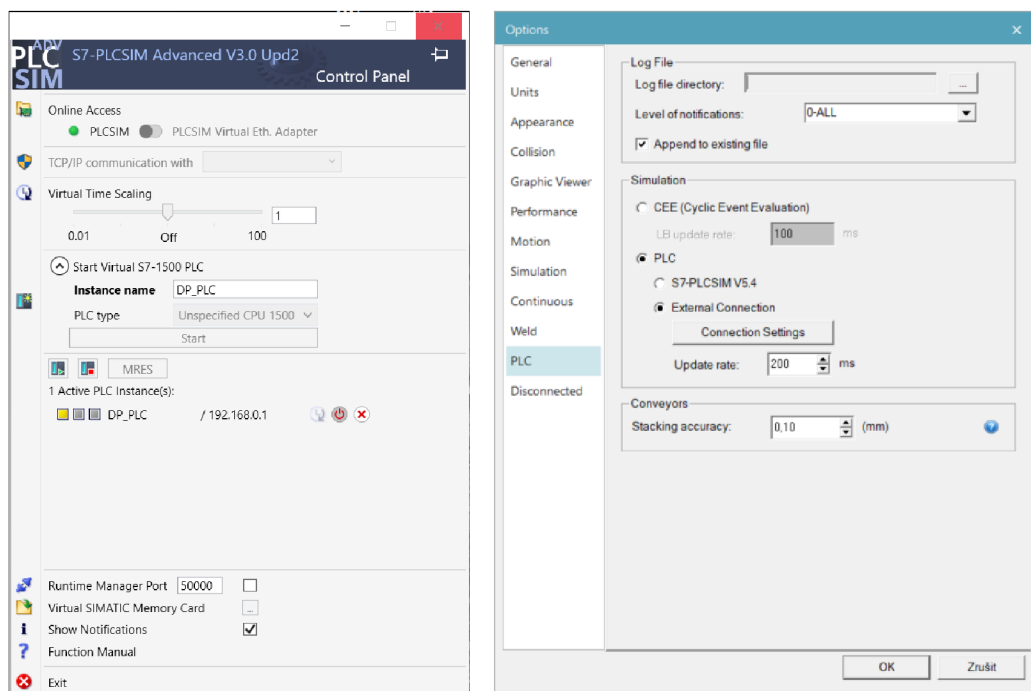
Po vytvoření signálové struktury, která je schopná ovládat každý prvek pracoviště dle požadavků, bylo vyzkoušeno jejich fungování v rozhraní PS skrze spuštění interní simulace a manuální přepínání signálů do žádaných stavů.



Simulation	Inputs	Outputs	LB	Forc..!	For...	Address
ARG1						
SACHTA						
BLATNIK						
Bez_dvere_B_OPEN			<input type="checkbox"/>	<input checked="" type="checkbox"/>		Q126.3
Bez_dvere_B_CLOSE			<input type="checkbox"/>	<input checked="" type="checkbox"/>		Q126.2
Rot_Stul_B_ToPos2_LVen			<input type="checkbox"/>	<input checked="" type="checkbox"/>		Q128.0
Rot_Stul_B_ToPos1_RVen			<input type="checkbox"/>	<input checked="" type="checkbox"/>		Q127.7
Bez_dvere_B_CLOSED			<input type="checkbox"/>	<input checked="" type="checkbox"/>		I127.2
Bez_dvere_B_OPENED			<input type="checkbox"/>	<input checked="" type="checkbox"/>		I127.3

Obr. 83 – Manuální „silové“ zapisování signálů

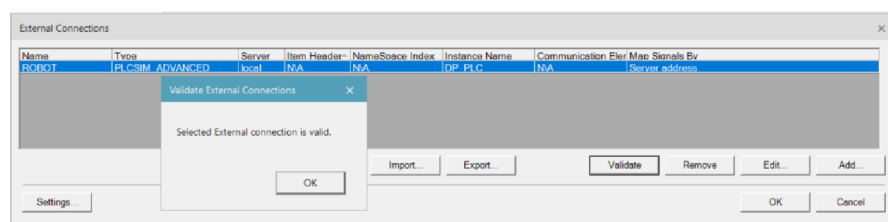
Následně bylo možné provázat PS s externím ovládáním. V našem případě se jedná o virtuální PLC v softwaru **S7-PLCSIM Advanced V3.0 Upd2**. Jakmile bylo otestováno externí připojení, PS byl dále spojen s virtuálním PLC a připraven na externí řízení.



a)

b)

Obr. 84 – a) Program PLCSIM, b) Propojení PS a TIA



Obr. 85 – Kontrola připojení mezi PS a TIA

## 7 VIRTUÁLNÍ KONTROLÉR

K virtuálnímu řízení projektu byly využity softwary od firmy SIEMENS. Nejdůležitějším byl **TIA Portal v16**, ve kterém bylo možné vytvořit všechny řídicí programy PLC, a následně software **S7-PLCSIM Advanced V3.0 Upd2**, určený k virtuálnímu spuštění naprogramovaného PLC. Díky tomu bylo možné nahrát off-line PLC program do virtuálního stroje a testovat funkčnost řídicího systému přímo skrze reakční animace v PS.

Na vývojovém diagramu je znázorněna žádaná logika fungování robotického pracoviště. Program je tvořen základní inicializací, kdy musí být splněny určité podmínky, než-li bude možné spustit automatický mód chodu pracoviště. Automatický chod je poté rozdělen na dva podprogramy. První zajišťuje logiku operátora, tzn. upínání polotovaru, rotace stolu atd. Druhý poté vytváří logiku toho, kdy může dojít ke spuštění robotické operace a jak bude sekvenčně probíhat celý automatický cyklus robotu.



Obr. 86 - Vývojový diagram fungování pracoviště

## 7.1 PLC moduly

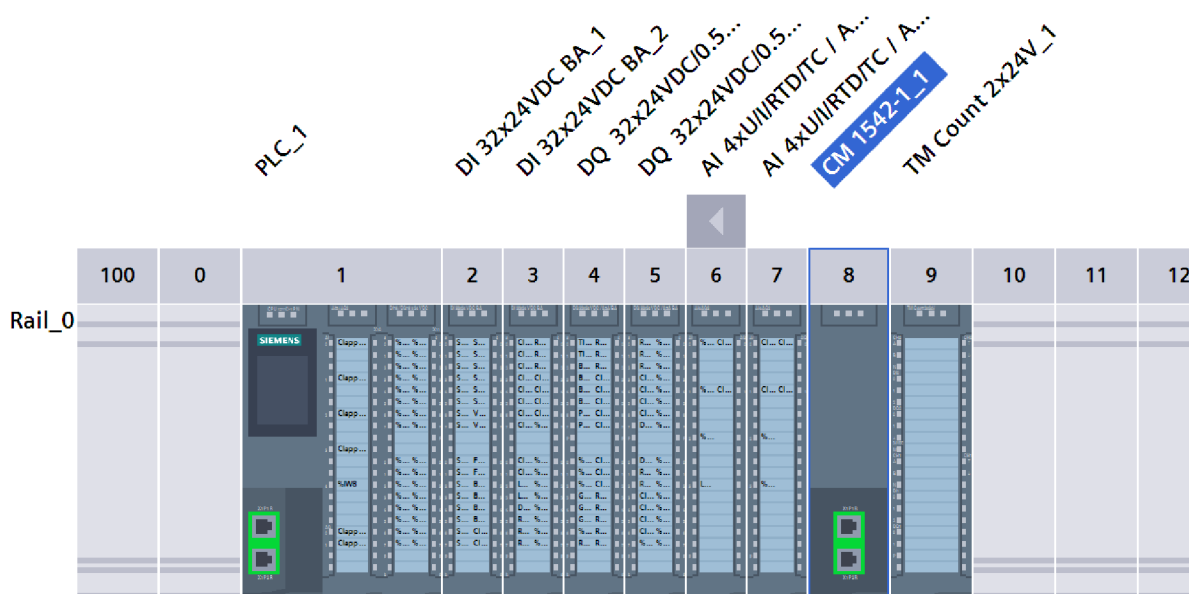
V prvním kroku bylo nutné vytvořit námi vybrané PLC v softwaru TIA a dodatečně nakonfigurovat potřebné moduly.

Vybrané PLC v původním stavu obsahovalo:

- Analogový modul (AI 5 / AO 2)
- Digitální modul (DI 16 / DQ 16)
- Profinet interface (Port 2x)

V konečném stavu bylo zapotřebí dalších modulů:

- Digitální vstupní modul DI 32 – 2x
- Digitální výstupní modul DO 32 – 2x
- Analogový modul (AI 4x / AO 2x) – 2x
- CM 1542-1
  - Komunikační modul pro Profinet (2 porty)
- TM Count 2x24V
  - Čítací modul pro vyhodnocování signálů inkrementálního enkodéru
  - dva kanály



Obr. 87 – Konfigurace PLC se všemi využitými moduly

## 7.2 Převod signálů do TIA Portálu

Po dokončení propojení mezi PLC a TIA portálem, a předpřípravou potřebných modulů k správnému fungování kontroléru, následovalo vytvoření globálních signálů PLC.

Aby bylo možné řídit signály v softwaru PS skrze signály v TIA, bylo nutné pojmenovat každý signál naprosto stejně. PS nabízí možnost exportování všech signálů do souboru .xlsx (Excelová tabulka), TIA ovšem takové rozložení tabulky nepodporuje. Z toho důvodu bylo nutné naprogramovat skript, který upraví tabulku dle požadavků TIA portálu, nebo všechny signály přepisovat ručně. K první variantě bylo využito programovací prostředí MATLAB.

	A	B	C	D	E	F	G	H	I	J
1	Signal Name	Memory	Type	Robot Signal Name	Address	IEC Format	PLC Connect	External Connection	Resource	Comment
2	a_end	NEPRAVDA	BOOL		No Address	I	NEPRAVDA			
3	albany_RP300_1900x2300_R_Op_end_2	NEPRAVDA	BOOL		No Address	I	NEPRAVDA		albany_RP300_1900x2300_R	
4	albany_RP300_1900x2300_R_Op1_end_1	NEPRAVDA	BOOL		No Address	I	NEPRAVDA		albany_RP300_1900x2300_R	
5	b_end	NEPRAVDA	BOOL		No Address	I	NEPRAVDA			
6	Bez_dvere_B_CLOSE	NEPRAVDA	BOOL		No Address	Q	PRAVDA		ToolInstance	

Obr. 88 – Náhled původní struktury tabulky z PS

	A	B	C	D	E	F	G	H	I	J
1	Name	Path	Data Type	Logical Address	Comment	Hmi Visible	Hmi Accessible	Hmi Writeable	Typeobject ID	Version ID
2	Bez_dvere_B_CLOSE		BOOL							
3	Bez_dvere_B_CLOSED		BOOL							
4	Bez_dvere_B_OPEN		BOOL							

Obr. 89 – Náhled výsledné tabulky

```

Prevod_prom_PS_na_TIA.m x +
1 clear; clc;
2 Polotovarov= importdata ("PLCTags2.xlsx"); % polotovarov kvuli prvniku radku
3 Polotovarov= Polotovarov.PLCtags;
4 Vstup = importdata ("DP_9_3_2023_2.xlsx"); % Vstup
5 Vstup = Vstup.textdata;
6 Vstup_size= size (Vstup);
7
8 Vystup = cell (Vstup_size);
9 Vystup(1,:) = Polotovarov(1,:);
10 Vystup(2:(Vstup_size(1)),1)=Vstup(2:(Vstup_size(1)),1);
11 Vystup(2:(Vstup_size(1)),3)=Vstup(2:(Vstup_size(1)),3);
12 Vystup_upraveny=Vystup;
13 VL=length(Vystup_upraveny);
14 writecell (Vystup,'Vystup.xlsx');
15 %
16 % V excelu jeste zmenit SHEET nazev na PLC Tags
17 % upravy
18 pattern= '_end';
19 pattern2= '_end_1';
20 pattern3= '_end_2';
21 for i= 0:(VL-2)
22     if contains(Vystup_upraveny(VL-i,1),pattern) == 1
23         Vystup_upraveny(VL-i,:) = [];
24     end
25     if contains(Vystup_upraveny(VL-i,1),pattern2) == 1
26         Vystup_upraveny(VL-i,:) = [];
27     end
28     if contains(Vystup_upraveny(VL-i,1),pattern3) == 1
29         Vystup_upraveny(VL-i,:) = [];
30     end
31 end
32 writecell (Vystup_upraveny,'Vystup_upraveny.xlsx');

```

Obr. 90 – Struktura podprogramu určeného k úpravě tabulky signálů z PS

Program (Obr. 90) nahraje polotovary správně formovaných vstupních dat do PLC (PLCTags2.xlsx) a hrubá data z PS (DP\_9\_3\_2023\_2.xlsx) a zjistí jejich velikost.

Následně skript vytvoří výstupní soubor (Vystup.xlsx), který má již žádané rozložení tabulky a je téměř připraven na import do TIA. Jelikož se ovšem v tabulce nachází i interní signály PS, končící **\_end**, **\_end\_1**, **\_end\_2**, které nemají žádné využití v TIA, program vyfiltruje tyto signály a vytvoří vyfiltrovaný výstupní soubor (Vystup\_upraveny.xlsx).

Řádek	Popis
1-6	Import polotovaru pro TIA portal a signálů z PS
8-14	Vytvoření výstupní matice a umístění sloupců dle našich požadavků
18-20	Definování filtrovacího klíče (název signálu jej musí obsahovat)
21-31	Prohledávání výstupního souboru a eliminace nepotřebných signálů



	Program prohledává od posledního záznamu, aby nedošlo k přeskočení signálu po vyškrtnutí aktuálního
32	Výstupní soubor bez nežádaných signálů

Tabulka 4 – Popis fungování podprogramu (obr.90)

Posledním krokem bylo otevřít výstupní soubor v programu Excel a změnit název listu na PLC\_Tags. Poté bylo možné importovat tabulku signálů do TIA Portálu (PLC tags).

Po importování signálů následovalo jejich sekundární protřídění (ne všechny signály je nutné řídit z PLC, jelikož slouží pouze k vnitřní logice PS) a přiřazení I/O svorek.

### 7.3 Organizace signálů

Z pohledu simulace a přehlednosti signálové struktury je nevhodné využívat v programu I/O signály (PLC Tagy) přímo. Z toho důvodu byly vytvořeny nové interní signály v PLC Data types, které mají za úkol reprezentovat a ovládat importované PLC tagy. Z důvodu lepší přehlednosti je vhodné tyto nové signály rozdělit do podskupin (obr. 91).

The image shows a software interface for organizing PLC data types. On the left, a tree view under 'PLC data types' shows a hierarchy: 'CONTROL' > 'Q' > 'Dvere'. Other categories include 'MotionControl', 'Ovladani', 'Robot', and 'Stul/Klapky'. On the right, a table titled 'Dvere' provides a detailed view of the signals within this category.

Dvere			
	Name	Data type	Default value
1	b_Bez_dvere_S_OPEN	Bool	false
2	b_Bez_dvere_S_CLOSE	Bool	false
3	b_Bez_dvere_B_OPEN	Bool	false
4	b_Bez_dvere_B_CLOSE	Bool	false
5	b_Dvere_bocni_LOCK	Bool	false
6	b_Dvere_bocni_UNLOCK	Bool	false

a)

b)

Obr. 91 – a) Rozdělení signálů do kategorií, b) Náhled obsahu každé kategorie

Posledním krokem bylo vytvoření centrální kontrolní databáze **CONTROL\_DATA**, jenž obsahuje přehledně všechny podložky Data types a další potřebné signály z programů PLC. Jedná se o přehlednou **Centrální databázi** (obr. 92) všech signálů této práce využívaných v PLC.

CONTROL_DATA				
	Name	Data type	Start value	Re
1	▼ Static			
2	■ ▼ CONTROL	"Control"		
3	■ ▼ Q	"Q"		
4	■ ▶ Dvere	"Dvere"		
5	■ ▶ Ovladani	"Ovladani"		
6	■ ▶ Robot	"Robot"		
7	■ ▶ Stul	"Stul/Klapky"		
8	■ ▶ Klapky_S_new	"Klapky_S_new_O"		
9	■ ▶ Klapky_B_new	"Klapky_B_new_O"		
10	■ ▶ MC_Out_B	"MC_Out_B"		
11	■ ▶ MC_Out_S	"MC_Out_S"		
12	■ ▼ SEN/FB	"Sensors/Feedback"		
13	■ ▼ I	"I"		
14	■ ▼ Dvere	"Dver"		
15	■ b_Bez_dvere_B_CLO...	Bool	false	
16	■ b_Bez_dvere_B_OPE...	Bool	false	
17	■ b_Bez_dvere_S_CLO...	Bool	false	
18	■ b_Bez_dvere_S_OPE...	Bool	false	
19	■ b_Dvere_bocni_ZAV...	Bool	false	
20	■ ▶ Stul_klapky_B_new	"Klapky_B_new"		
21	■ ▶ Stul_klapky_S_new	"Klapky_S_new"		
22	■ ▶ Robot	"Robt"		
23	■ ▶ Stul	"Stul"		
24	■ ▶ Up_prostor	"UpProstor"		
25	■ ▶ MC_IN_S	"MC_IN_S"		
26	■ ▶ MC_IN_B	"MC_IN_B"		

Obr. 92 – Centrální kontrolní databáze všech signálů v TIA

Aby bylo možné ovládat PLC tagy (signály) pomocí signálů z této databáze, bylo nutné vytvořit funkce, které propojí jejich hodnoty (logiku). Aby funkce fungovaly cyklicky, je nutné je volat v Main programu PLC.

Ukázky propisování hodnot mezi PLC tagy a CONTROL\_DATA databází.

- 1 "CONTROL\_DATA"."SEN/FB".I.Dvere.b\_Bez\_dvere\_B\_CLOSED := "Bez\_dvere\_B\_CLOSED";
- 2 "CONTROL\_DATA"."SEN/FB".I.Dvere.b\_Bez\_dvere\_B\_OPENED := "Bez\_dvere\_B\_OPENED";
- 3 "CONTROL\_DATA"."SEN/FB".I.Dvere.b\_Bez\_dvere\_S\_CLOSED := "Bez\_dvere\_S\_CLOSED";

Obr. 93 – Funkce přiřazující hodnoty vstupních signálů

- 1 "Bez\_dvere\_S\_OPEN" := "CONTROL\_DATA".CONTROL.Q.Dvere.b\_Bez\_dvere\_S\_OPEN;
- 2 "Bez\_dvere\_S\_CLOSE" := "CONTROL\_DATA".CONTROL.Q.Dvere.b\_Bez\_dvere\_S\_CLOSE;
- 3 "Bez\_dvere\_B\_OPEN" := "CONTROL\_DATA".CONTROL.Q.Dvere.b\_Bez\_dvere\_B\_OPEN;

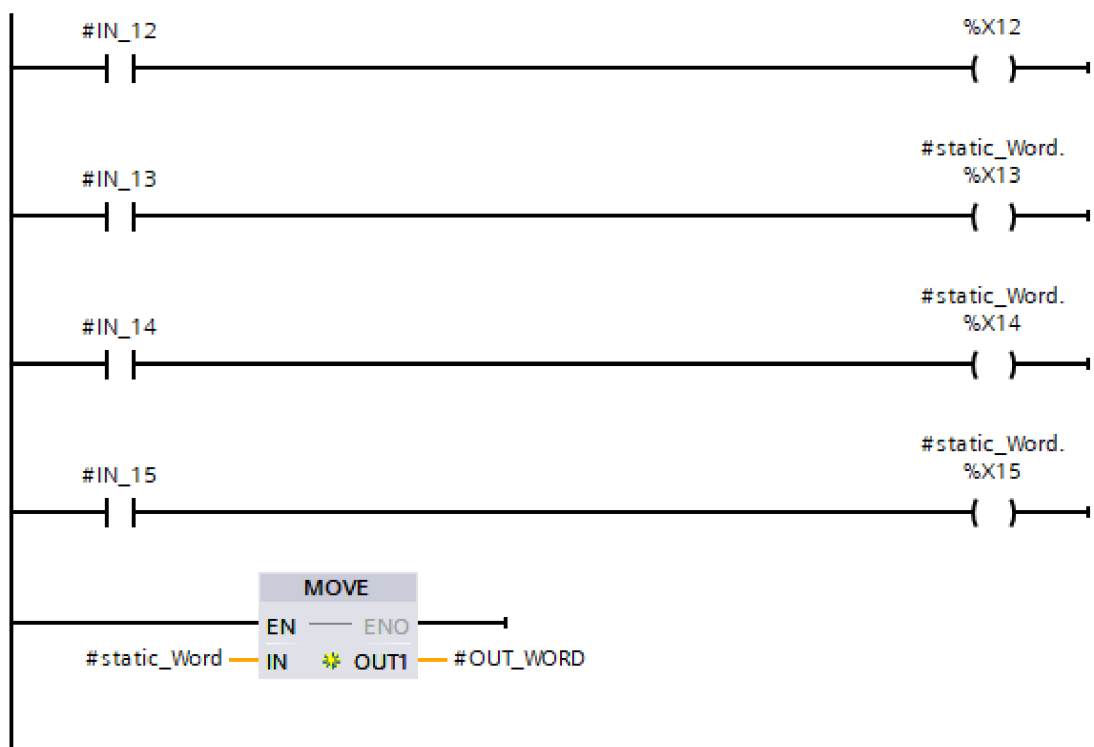
Obr. 94 - Funkce přiřazující hodnoty výstupních signálů

## 7.4 FB Převod proměnných

Stejně jako v případě Process Simulate, bylo nutné i v TIA provést konverzi proměnných z datového typu BOOL na WORD a opačně. Důvodem byla přesnější simulace reálného prostředí.

Jedná se o převod kontrolních bitů z PLC do datového typu WORD, který putuje do cílového ventilového bloku.

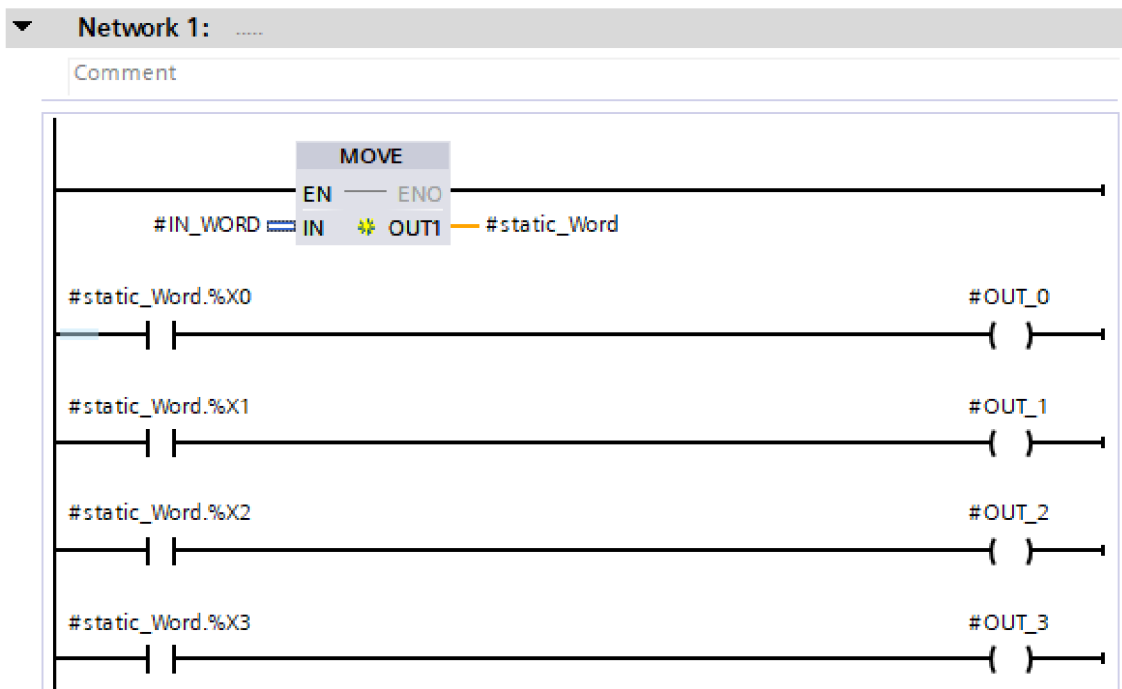
K převodu byl vytvořen funkční blok (FB) s programovacím jazykem Ladder Diagram. Struktura programu je velmi jednoduchá. Každý vstupní bit se zapíše do statického bitu datového typu WORD, a ten se v konečné podobě zapíše do výstupní proměnné. Důvodem jednoduchého programu bez využití datových typů Array (Matic) byla přehlednost proměnných a jejich relativně malý počet. Jedná se o proměnné typu OPEN/CLOSE.



Obr. 95 – Podprogram pro převod BITS → WORD

V opačném případě se jedná o převod vstupních analogových signálů z ventilového bloku na jednotlivé bity, reprezentující stavové proměnné jednotlivých předem určených signálů.

Znovu byl vytvořen FB v programovacím jazyku Ladder Logic, tak jako v předchozím případě, a logika je zrcadlově stejná. Jediným rozdílem je prvotní uložení vstupní proměnné do pomocné statické proměnné a následné přiřazení hodnoty každého jejího bitu do předpřipravených signálů řídicího PLC. Jedná se o stavové informace typu OPENED/CLOSED.



Obr. 96 - Podprogram pro převod WORD → BITS

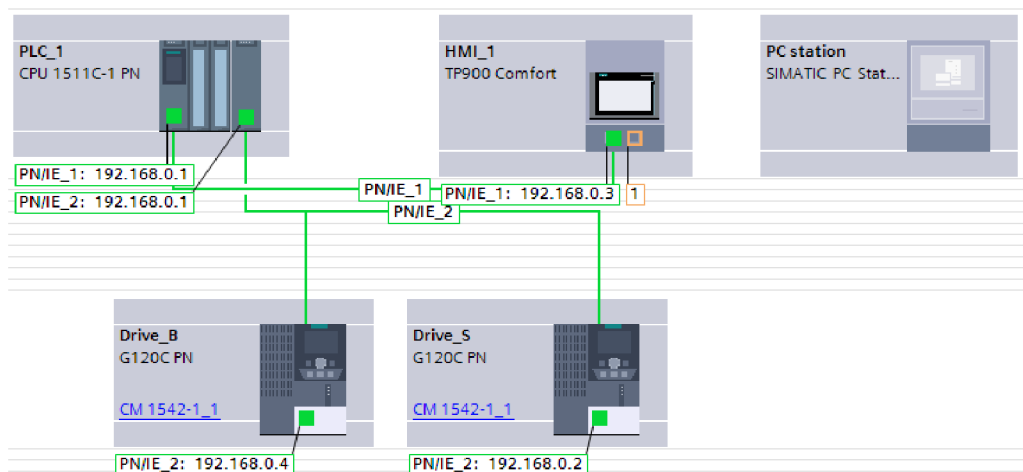
## 7.5 Řízení motorů rotačních stolů

K řízení elektromotorů pohánějících rotační stoly byly využity frekvenční měniče G120. Jedná se o standardní měniče firmy SIEMENS. Za předpokladu, že vyžadujeme pouze měnit frekvence, by bylo možné využít pouze jeden hardware G120. V našem případě ovšem vyžadujeme přesné polohování skrze PID regulaci. Z toho důvodu je žádoucí zvolit dva měniče.

V softwaru TIA s přídatným rozšířením SINAMICS DRIVE V16 bylo možné tyto měniče virtuálně zprovoznit a přichystat k reálnému použití. Kromě připravení logiky je v posledních verzích možné virtuálně zprovoznit i samotné poháněné osy, což bylo využito k dosažení co nejuvěrnější kopie reálného prostředí a fungování.

### 7.5.1 Frekvenční měniče

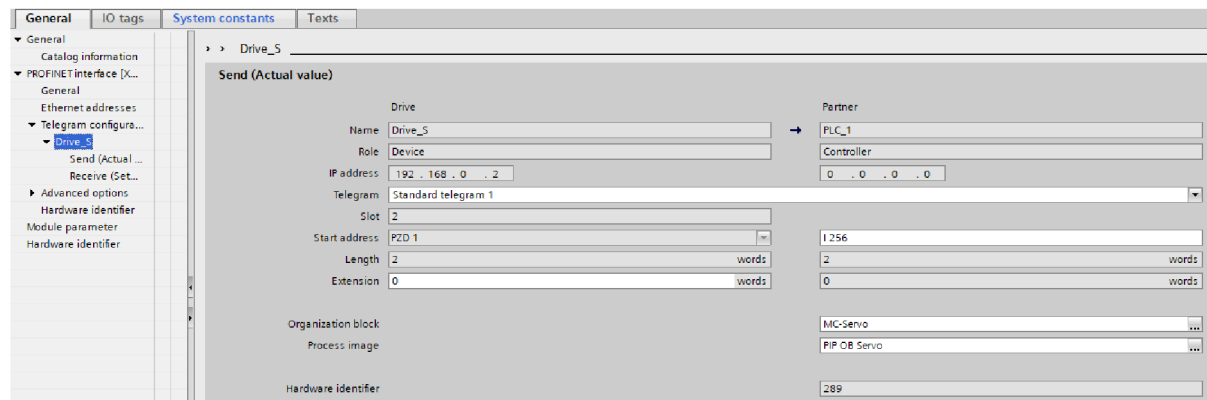
Prvním krokem bylo přidat frekvenční měniče do sítě, propojit je s nadřazenou řídicí jednotkou PLC a přiřadit jim v síti novou IP adresu.



Obr. 97 – Síťová struktura elektrických prvků pracoviště, včetně IP adres

Dalším krokem bylo nastavení použitého komunikačního protokolu s PLC, v našem případě volíme variantu Standard telegram 1. Druhou variantou byl například Standard Telegram 352, který by nám udával i aktuální hodnoty proudu, ale v tomto způsobu nasazení není potřebný.

Na obrázku můžeme vidět nastavení telegramu, adresy vstupních i výstupních slov (CONTROL WORD a STATUS WORDS) a jejich počet.



Obr. 98 – Nastavení komunikačního protokolu mezi VFD a PLC

Konfigurace frekvenčních měničů proběhla **dle přílohy č. 1**.

V první kategorii bylo jako využití vybráno Dynamic Drive Control (Dynamická kontrola pohonu), protože máme v plánu přesné polohování.

Následně byl vybrán systém G120 jako výpočetní médium řízení pohonu, jelikož nechceme PLC zatěžovat dodatečným výpočetním výkonem.

Komunikace byla nastavena skrze fieldbus (v našem případě Standard telegram 1). Vstupní napětí do pohonu uvažuje standardních 400 V.

Dále byly definovány důležité parametry pohonu od výrobce SAW a motor byl opatřen brzdou. Další parametry typu rychlost rozběhu a limit napětí byly ponechány původní.

Všechny vybrané parametry je možné kdykoliv upravit či pozměnit.

S druhým G120 bylo postupováno stejně jako s prvním.

### 7.5.2 Technologické objekty / Technology objects

Po umístění měničů do sítě a prvotním definování jejich parametrů, bylo možné samotné pohony ovládat skrze přímé ovládání kontrolních slov (CONTROL WORDS), které jsou definovány dle Standard Telegram 1 dokumentace. Druhou variantou bylo vytvořit v PLC tzv. Technology Objects (technologické objekty), které nám slouží jako prostředník mezi G120 a přímým ovládáním skrze kontrolní slova. Po jejich vytvoření bylo následně možné využívat předdefinované, garantované a standardizované funkce k ovládání pohonů. Díky tomu bylo možné vytvořit jednoduchý a srozumitelný ovládací funkční blok pohonu.

Konfigurace TO probíhala dle **přílohy č. 2**.

V Basic Parameters byl nastaven rotační typ osy, využívané jednotky a zapnut režim **Virtuální Osy**. Při zapnutí této funkce dojde k imaginárnímu zprovoznění osy uvnitř PLC, kdy ovšem osa přestane komunikovat s enkodérem a měničem (nejsme je schopni virtuálně zprovoznit). Osa se poté chová stejně jako ta reálná, ale její řízení probíhá přímo v PLC. **Využití** virtuálních os je především ve virtuálním zprovoznění, ale také se využívá ke generování tzv.

setpoints (nastavených hodnot) pro více reálných os pracujících synchronně (obdrží hodnoty ve stejnou chvíli). Po vypnutí režimu virtuální osy je osa okamžitě schopna znovu komunikovat s enkodérem a měničem dle dalšího nastavení, a tím i reálně pracovat.

Dále nastavujeme cestu využitého frekvenčního měniče a zvolený komunikační telegram. Kromě měniče zde konfiguruje i enkodér, a to využitý kanál modulu čítače PLC a komunikační telegram (ponecháváme Standard 83 pro enkodéry).

TO je poté možné hlouběji konfigurovat v kategorii Extended parameters. Pokud bychom chtěli detailněji řešit bezpečnost, je možné nastavení kdykoli upravit. V našem případě postačuje standardní nastavení.

### 7.5.3 Motion Control

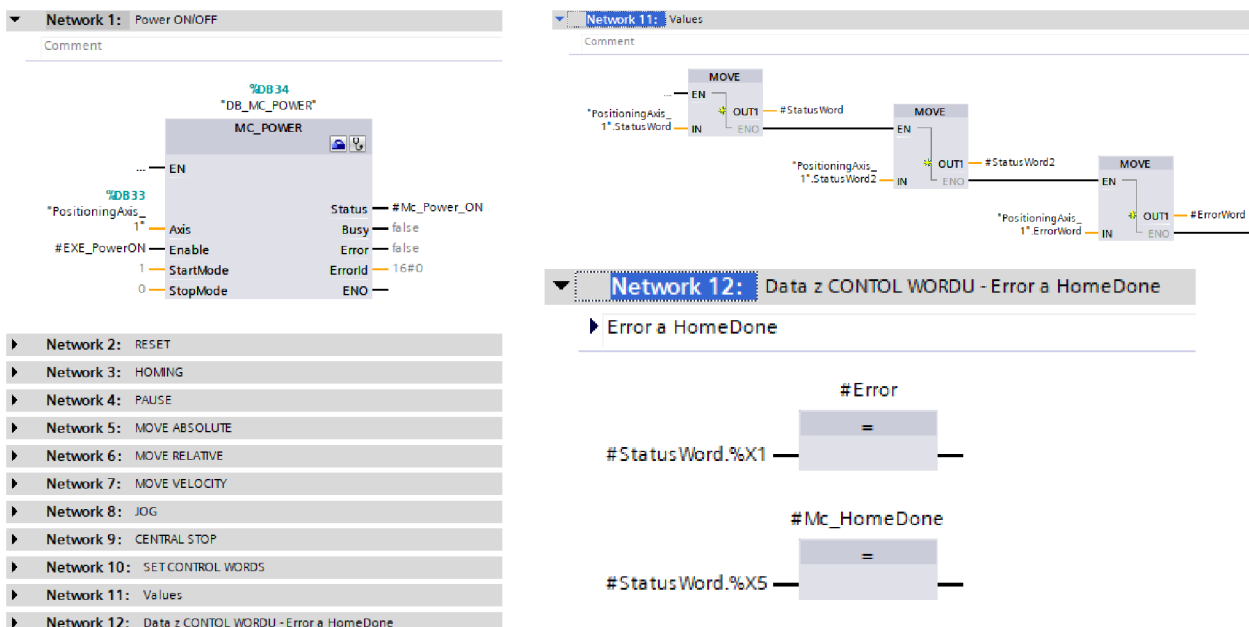
Jakmile byly vytvořeny TO, dalším krokem bylo vytvořit požadované funkční bloky, obsahující všechny ovládací funkce pohonů (virtuálních os).

Funkční blok byl vytvořen v programovacím jazyku Function Block Diagram. Program je rozdělen do dvanácti větví (Networks), kde prvních deset (obr. 99 a) obsahuje vždy jeden z hlavních předpřipravených bloků k ovládání pohonu (např. Power, MoveRelative atd.).

Větev č. 11 (obr. 99 b) obsahuje sadu MOVE instrukcí, které mají za úkol konstantně zapisovat aktuální hodnoty vstupních proměnných do lokálních statických proměnných. Díky tomu je možné sledovat stav všech jednotlivých signálů a chování funkčního bloku.

Poslední větev č. 12 (obr. 99 c) pracuje na podobném principu jako předchozí, ale slouží k vypisání hodnot jednotlivých bitů ze stavových slov (Status Word), konkrétně Error a Mc\_HomeDone.

Celkový ovládací blok je jednoduchý, přehledný a lehce upravitelný. Sledovatelných signálů je mnohem více, než které jsou vytvořeny ve větvi 12 a 13.



a)

b) – Nahore

c) Dole

Obr. 99 – a) Struktura řídícího programu, b) + c) čtení kontrolních hodnot

Posledním krokem bylo zprovoznit vytvořený ovládací blok cyklickým voláním v hlavním programu PLC – Main. K ovládání byly využity námi předpřipravené proměnné z hlavní databáze CONTROL\_DATA.

```

207 DB_MotionControll1 (EXE_Homing:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_Homing,
208 EXE_PowerON:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_PowerON,
209 EXE_Reset:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_Reset,
210 Mc_NewHomePos:="CONTROL_DATA".CONTROL.Q.MC_Out_S.r_Mc_NewHomePos,
211 EXE_MoveAbsolute:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_MoveAbsolute,
212 EXE_MoveRelative:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_MoveRelative,
213 Mc_MoveAbs_ToPosition:="CONTROL_DATA".CONTROL.Q.MC_Out_S.r_Mc_MoveAbs_ToPosition,
214 Mc_MoveRel_Distance:="CONTROL_DATA".CONTROL.Q.MC_Out_S.r_Mc_MoveRel_Distance,
215 Mc_Move_Velocity:="CONTROL_DATA".CONTROL.Q.MC_Out_S.r_Mc_Move_Velocity,
216 Mc_Reverse:="CONTROL_DATA".CONTROL.Q.MC_Out_S.n_Mc_Reverse,
217 EXE_Stop:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_Stop,
218 EXE_Pause:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_Pause,
219 Mc_Move_Done=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_Move_Done,
220 Mc_Home_Bussy=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_Home_Bussy,
221 Mc_Power_ON=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_Power_ON,
222 Mc_HomeDone=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_HomeDone,
223 ActualSpeed=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_ActualSpeed,
224 ActualVelocity=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_ActualVelocity,
225 ActualPosition=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_ActualPosition,
226 StatusWord=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_StatusWord,
227 StatusWord2=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_StatusWord2,
228 ErrorWord=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_ErrorWord,
229 Error=>"CONTROL_DATA". "SEN/FB".I.MC_IN_S.Mc_Error); //MC - Pohon Stolu SACHTA
  
```

Obr. 100 – Volání řídicího programu pohonů v MAIN

## 7.6 FB Operátor

Logika automatického ovládání pracoviště, znázorněná v navrženém logickém diagramu (obr. 86), je tvořena dvěma hlavními podprogramy. První z nich má na starosti zajišťovat logiku pracoviště z pohledu obsluhy. To znamená, že při zmáčknutí aretačního tlačítka dojde k otevření bezpečnostních dveří, upínacích prvků, a po odměčknutí ke spuštění upínacích klapek, bezpečnostních dveří a otočení stolu.

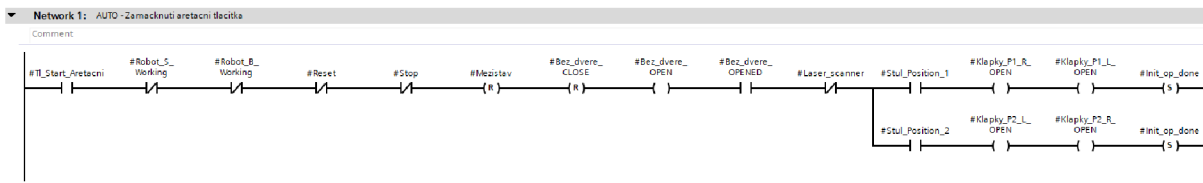
Příkaz k rotaci stolu je poté vyveden do dalšího podprogramu, který zadá hodnotu polohy MoveAbsolute a spustí operaci (kap. 7.5). V reálném nasazení by bylo nutné provést přepoččet požadovaných otáček k docílení žádané polohy v závislosti na typu využívaného převodu.

### 7.6.1 Šachta

Logika operátorů obsluhy je u polotovarů typu šachta a blatník lehce rozdílná. Prvním rozdílem je umístění dvou polotovarů místo jednoho. Dalším rozdílem jsou upínací prvky stolu, které se mohou nacházet ve třech pozicích (OPENED/CLOSED/WORK) místo obvyklých dvou (OPENED/CLOSED). Z toho důvodu bylo nutné program upravit.

Řídicí program byl vytvořen primárně v programovacím jazyku LD, ale některé ze sítí byly tvořeny ve Strukturovaném textu. Kód je rozdělen celkem do sedmi větví, z toho čtyři v LD a tři v SCL.

V první síti se čeká na stlačení hlavního aretačního tlačítka. Po jeho aktivaci dojde ke kontrole základních bezpečnostních prvků.



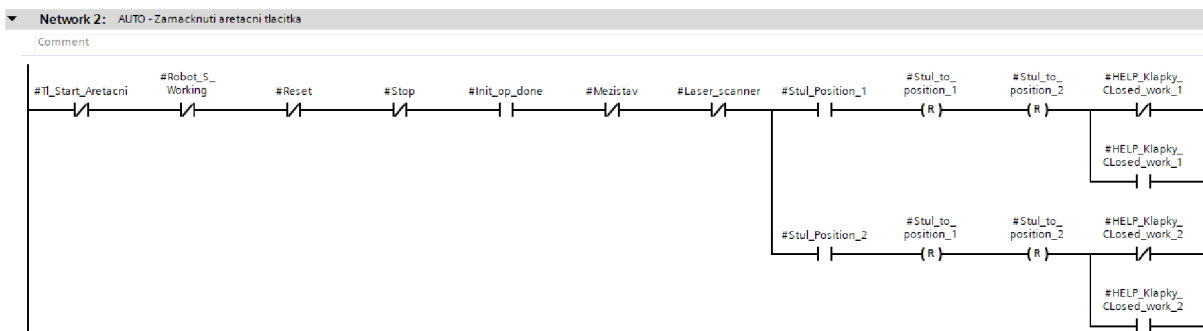
Obr. 101 – Program logiky operátora – první síť

Základní podmínkou je, že **roboti nesmí pracovat**. Bez této podmínky by mohlo při otevření bezpečnostních dveří dojít nedbalostí obsluhy k vychýlení rotačního stolu z pozice a tím ke špatnému svaření bodů. Podmínka může být hlouběji zpracována, a Robot\_B může pracovat pouze na operacích polotovarů blatník na druhém stole. V tom případě by se podmínka změnila na porovnání čísla aktuálně prováděné operace, která by se nesměla rovnat číslu prováděných operací při procesu sváření šachet. Tím by se zvýšil výrobní takt. S ohledem na bezpečnost a stabilitu byla zvolena první, ortodoxnější varianta.

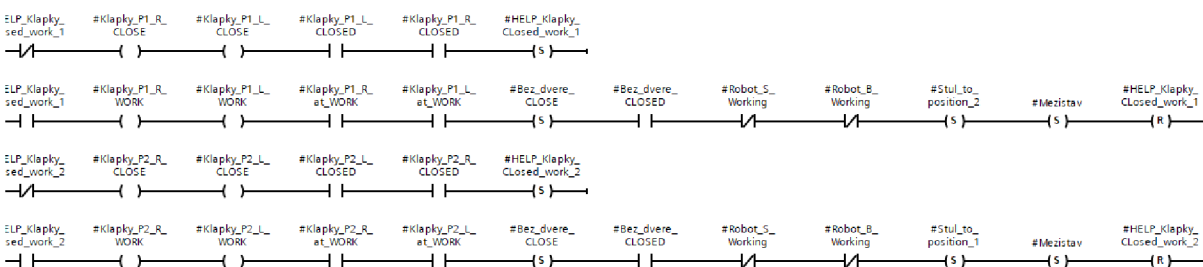
Dále nesmí být stisknuté fyzické bezpečnostní tlačítko **Stop** nebo tlačítko **Reset** na HMI. Stop tlačítko má za úkol zastavit všechny procesy, tlačítko Reset navíc spustí robotické operace pro návrat do HOME pozice.

Následně dojde k resetování hodnot signálu **Mezistav** (pomocný signál ve stavu NC k zajištění, že při prvotním zapnutí režimu AUTO nemůže dojít k aktivaci druhé sítě, jež má na starosti rotaci stolu).

Posledním krokem první části je restartování signálu k **zavření dveří** (pojistka) a sepnutí signálu k **otevření dveří**. Po otevření dojde dle aktuální pozice stolu k otevření požadovaných upínacích prvků za předpokladu, že **laserový skener** nedetekuje obsluhu uvnitř pracovního prostoru. Jedná se o bezpečnostní prvek, zajišťující bezpečnost obsluhy, který ovšem obsluze umožňuje alespoň opticky zkontrolovat proces otevírání upínacích prvků. Při jejich mechanickém poškození by obsluha díky tomu byla schopna upozornit na chybu, kterou by jinak nebylo možné bez dalších dodatečných sensorických prvků odhalit. Po provedení je sepnut signál o provedení inicializační operace (další bezpečnostní prvek spolu s Mezistav).



Obr. 102 - Program logiky operátora – druhá síť – první část



Obr. 103 - Program logiky operátora – druhá síť – druhá část



V druhé síti (obr. 102-103) se nachází program ovládající automatický chod rotačního stolu.

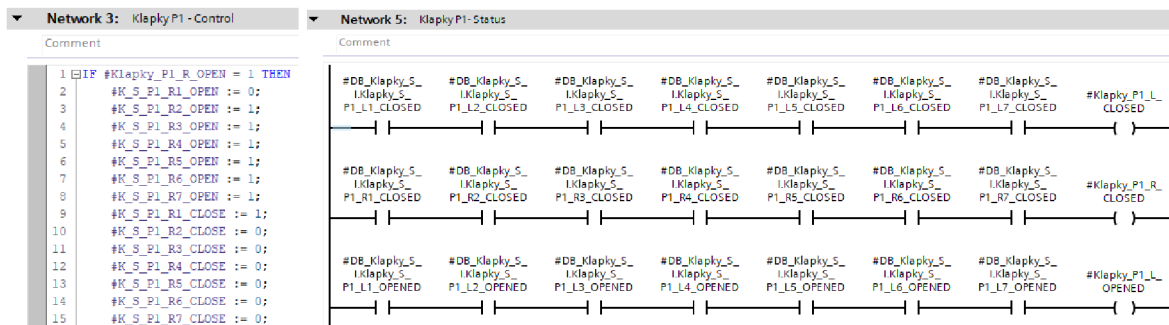
Při odaretování spouštěcího tlačítka zde znovu dojde ke kontrole, zdali robot nepracuje, jestli nebyly stisknuty Stop a Reset tlačítka a jestli se obsluha nenachází uvnitř prostoru (Laser\_scanner). Přibylly zde bezpečnostní signály o provedení inicializační operace a Mezistav musí být roven 0.

Následně se program rozděluje do dvou větví dle aktuální pozice stolu. Nejdříve dojde k resetování povelových signálů k rotaci stolu, poté kód narazí na další rozcestník, začínající pomocným signálem s názvem HELP\_\*\*\*. Pokud je pomocný signál False, dojde k sepnutí povelových signálů upínacích prvků „přejít do stavu zavřeno“ (KLAPKY\_CLOSE). Po dokončení se sepne pomocný signál HELP do True, program přepne do druhé cesty, kde vyšle příkaz upínacím prvkům „přejít do stavu WORK“ (KLAPKY\_WORK). Po dokončení následně vyšle signál k zavření bezpečnostních dveří.

Posledním krokem je znovu kontrola, že roboty nepracují, a následný příkaz k rotaci do druhé pozice a resetování pomocných signálů HELP a Mezistav. Tím je cyklus dokončen. Jakmile robot začne pracovat na daném svářecím cyklu (blatník nebo šachta), informace je obsluze znázorněna na HMI panelu. Po dokončení obsluha může znovu stisknout tlačítko k otevření bezpečnostních dveří a vyjmutí/založení polotovarů.

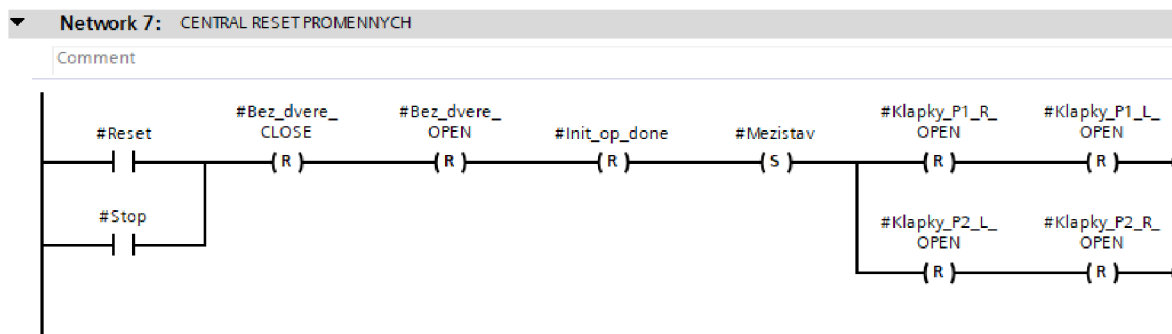
Větve číslo 3 a 4 (obr. 104) zajišťují definici pozic upínacích prvků v jazyku SCL. Každá struktura IF přehledně přiřazuje řídicí hodnotu každému z upínacích prvků.

Větve číslo 5 a 6 (obr. 104) zajišťují zpětnou vazbu o dosažení pozice upínacích prvků v jazyku LD. Každý řetězec přiřazuje určité kombinaci přehlednou centrální proměnnou.



Obr. 104 - Program logiky operátora – síť 3-6

Poslední síť zajišťuje jednoduché zastavení prvních řídicích větví a reset proměnných při zmáčknutí Stop tlačítka nebo Resetu.

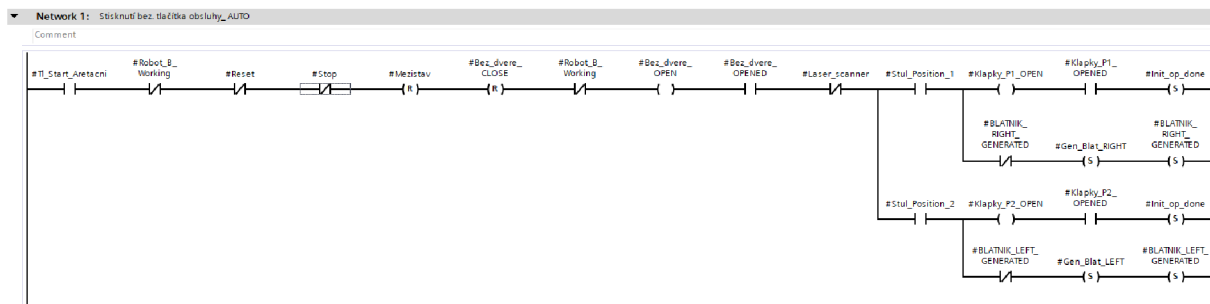


Obr. 105 - Program logiky operátora – poslední síť

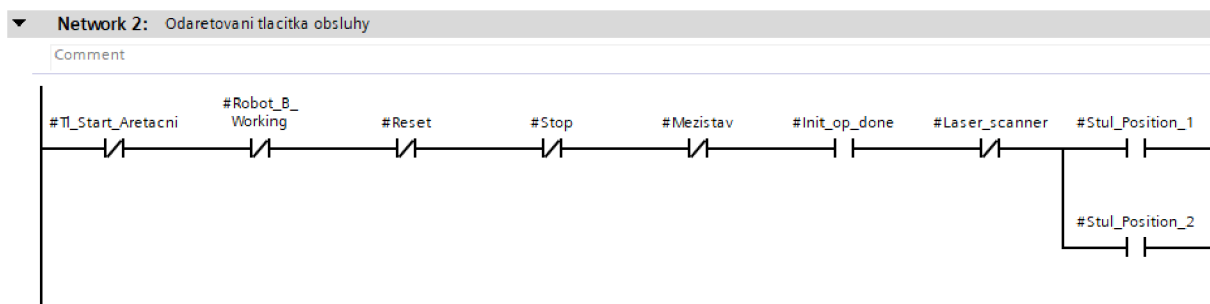
## 7.6.2 Blatník

Program pro polotovary typu blatník je téměř stejný jako pro šachtu. Prvním rozdílem je obohacení první sítě o generování vizuálního modelu blatníku. Důvodem, proč jen pro jednu větev, je přetrvávající chyba animace v PS, způsobující padání programu. V reálném nasazení by v byla větev generující polotovary odstraněna.

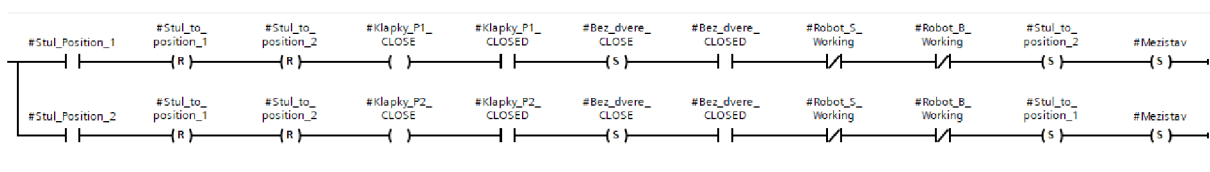
Dalším rozdílem je zjednodušení programu. U polotovarů typu blatník využíváme pouze dvě pozice (OPENED/CLOSED). Program tedy vypustil části o pozici WORK, jelikož je zde totožná s CLOSED. Byla zde možnost ponechat původní verzi a pouze definovat pozici WORK jako rovnou CLOSED, ale s ohledem na přehlednost byl vytvořen druhý program.



Obr. 106 - Program logiky operátora polotovaru blatník – první síť

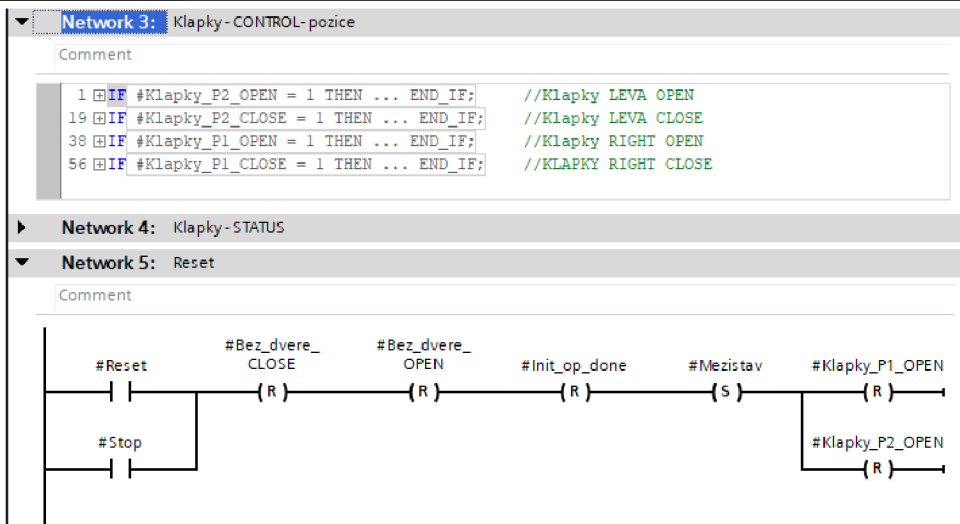


Obr. 107 - Program logiky operátora polotovaru blatník – druhá síť – první část



Obr. 108 - Program logiky operátora polotovaru blatník – druhá síť – druhá část

Původní síť 3 – 4 a 5 – 6 zde bylo vhodné sjednotit pouze do dvou, jelikož upínáme jen jeden polotovar v každé poloze a řešíme pouze dvě pozice. Struktura sítí je jinak naprosto totožná, jako u typu šachta.



Obr. 109 - Program logiky operátora polotovaru blatník

## 7.7 FB Roboty

Druhým nejdůležitějším programem je logika robotických operací. Ta má za úkol zajistit, aby po každém otočení rotačního stolu došlo nejdříve k vyhodnocení hodnot vstupních parametrů, a pokud jsou v pořádku, tak k následnému spuštění sekvence robotických operací.

Program je tvořen primárně v programovacím jazyku Ladder Diagram (LD) a jedna síť ve Strukturovaném textu (SCL).

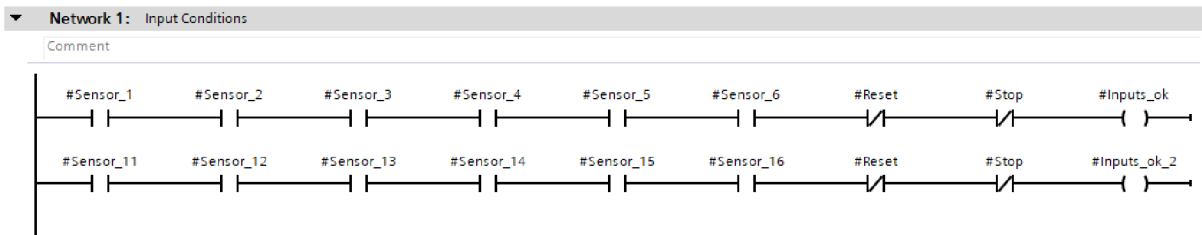
Podobně jako u programu logiky operátora jsou i zde menší rozdíly mezi řízením procesu polotovaru šachty a blatníku. Největším rozdílem je, že na svařování polotovarů šachet pracují oba roboty, kde první (Robot\_S) z nich svaří nejdříve všech 14 bodů jedné šachty a následně 5 bodů šachty druhé. Zbýlých 9 bodů svaří druhý robot (Robot\_B). Robotický program je jinak velmi podobný.

### 7.7.1 Šachta

Program je tvořen celkem z jedenácti sítí, z toho deset v LD a jedna v SCL.

**První síť** (obr. 110) má na starosti kontrolovat vstupní podmínky, které musí být splněny. V opačném případě dojde k přerušení chodu všech dalších sítí. Jedná se o signály z **indukčních senzorů**, které detekují upnutý polotovar na rotačním stole. Pokud nejsou všechny senzory v TRUE (tři pro každý polotovar), polotovar je špatně upnut, nebo došlo k poškození jednoho ze senzorů. Kromě senzorů zde nalezneme požadavek na nesepnuté bezpečnosti tlačítka Reset a Stop (při jejich sepnutí dojde k přerušení vstupních podmínek, a tím i zastavení dalších sítí). Po splnění všech podmínek dojde k sepnutí lokální pomocné proměnné Inputs\_ok a Inputs\_ok\_2 (druhá poloha).

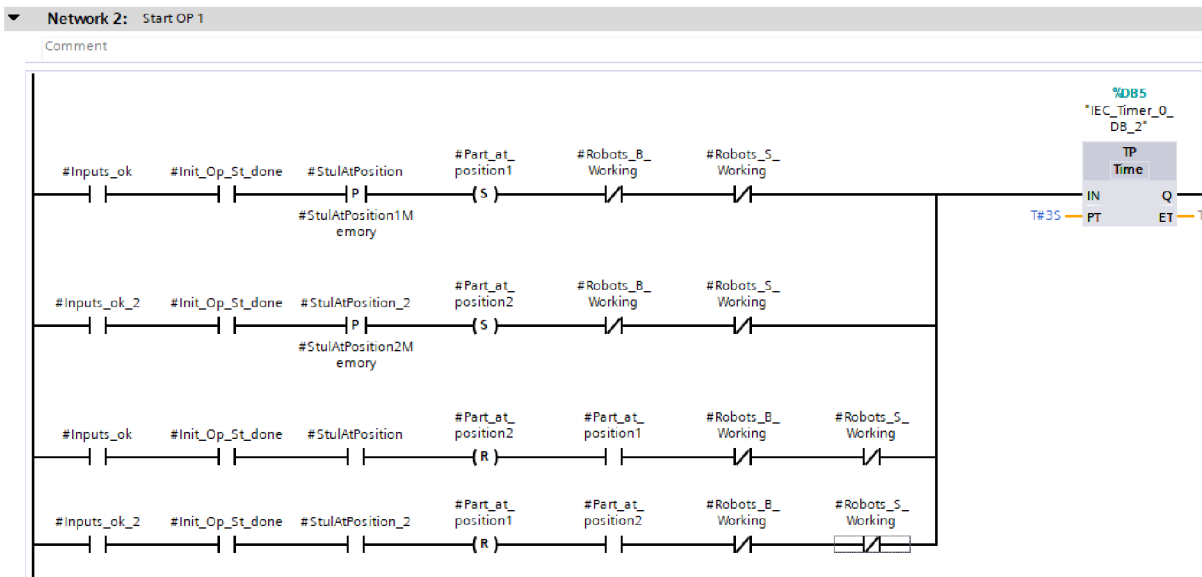
V automatickém programu uvažujeme vždy nutnost upnout oba polotovary v každé pozici, jinak nedojde ke splnění vstupních požadavků. Důvodem je struktura robotického programu, který je vytvořen pro svaření obou dílů najednou. Druhou variantou by bylo rozdělit program na svařecí operace jednoho polotovaru a druhého polotovaru. V tomto případě uvažujeme nepřetržitý provoz s dostatkem polotovarů, a proto není nutné program rozdělovat.



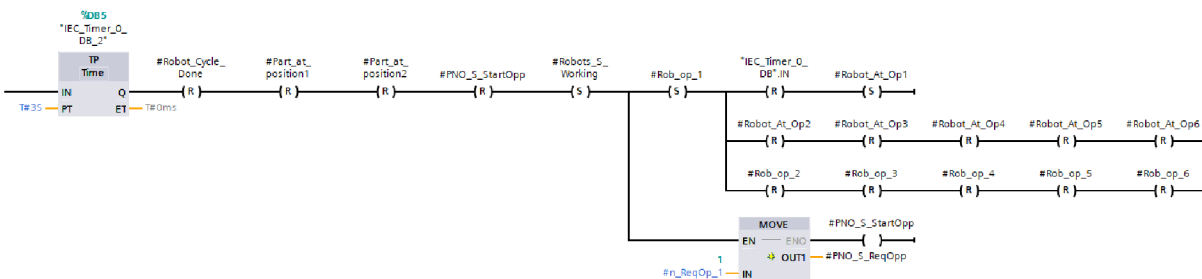
Obr. 110 - Program logiky robotu polotovaru šachta – první síť

V další síti (obr. 111) se dostáváme ke spuštění první operace automatického cyklu svařování šachet. Jakmile jsou splněny vstupní podmínky, program čeká na pozitivní hranu signálu o dojetí stolu na žádanou pozici, zapíše tuto hodnotu do dočasné proměnné (Part\_at\_position), a pokud roboty nepracují, pokračuje program dále. Pokud roboty pracují, program přejde do dalšího řádku, kde čeká na uvolnění robotů a následně spustí žádaný proces.

Dodatečná podmínka Init\_Op\_St\_done zajišťuje, aby se cyklus nespustil při zapomenutém polotovaru upnutém na rotačním stole. A vyžaduje, aby první cyklus započal po fyzickém zmáčknutí a domáčknutí aretačního tlačítka obsluhy, zmíněného výše v programu FB\_Operátor.



Obr. 111 - Program logiky robotu polotovaru šachta – druhá síť – první část



Obr. 112 - Program logiky robotu polotovaru šachta – druhá síť – druhá část

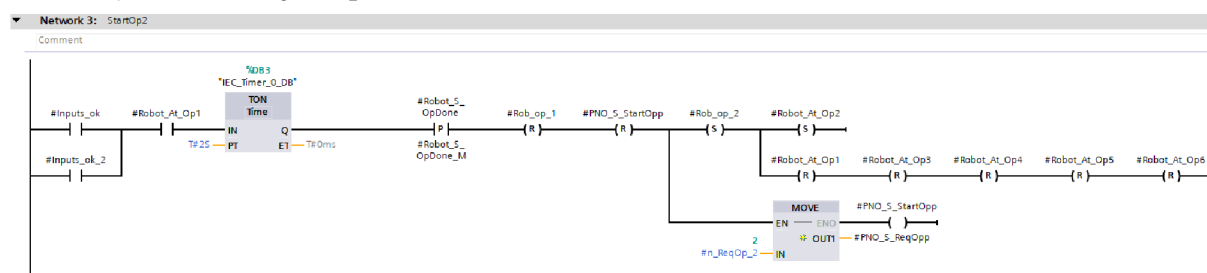
Program poté pokračuje do pomocného časovače, který po sepnutí přejde za 3 sekundy do stavu FALSE. Jde o dodatečný bezpečnostní prvek programu. Poté dojde k resetu pomocných a informačních signálů.

V posledním kroku dojde k sepnutí signálů do stavu "Robot\_S pracuje" (Robot\_Working), Robot\_At\_Op1, a samotnému spuštění operace. V našem případě animace v PS reaguje na signál z Rob\_op\_1, ale v reálném nasazení by byl využit signál ReqOp (= číslo žádané operace) a StartOp (start operace). Z toho důvodu je program nachystán na obě varianty. Nakonec dojde k resetu zbývajících pomocných proměnných a spouštěcích signálů, aby byl zajištěn správný chod pracoviště.

Jelikož došlo k sepnutí signálu do Robot\_Working, první síť přestane být funkční a přecházíme k síti číslo dvě. Časovač navíc funguje jako sekundární pojistka a zajišťuje přerušení napájení druhé části, a tím i permanentní sepnutí první robotické operace.

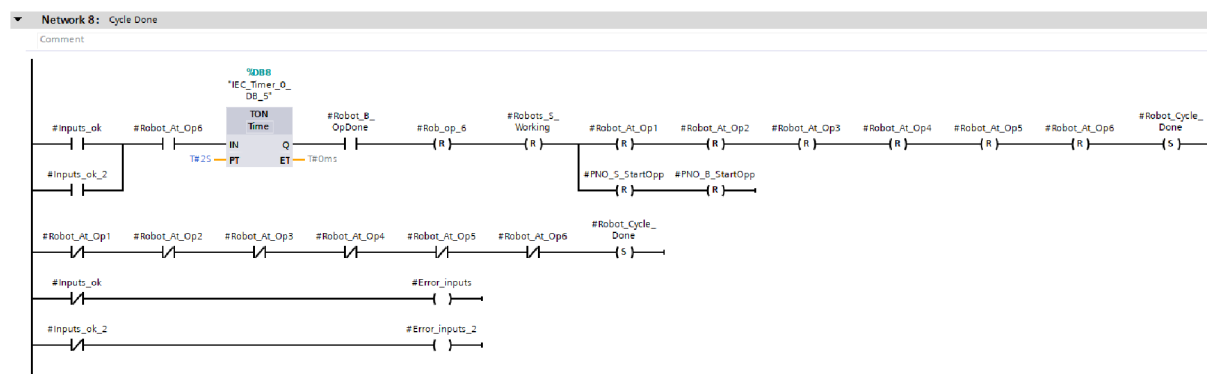
Druhá síť je již značně jednodušší. Pokud robot provádí operaci č. 1, dojde za dvě vteřiny k sepnutí výstupu časovače a program čeká na signál o dokončení operace. Při jejím dokončení dojde k sepnutí robotické operace č. 2 obdobně jako u první síti.

Signál o dokončení operace by byl v reálném nasazení takto obdržen z řídicího kontroléru robotu KUKA. V simulaci byl využit signál o najetí robotu do HOME pozice, jelikož do této pozice najíždí vždy a pouze po dokončení operace. Všechny další starty operací jsou řešeny obdobně a aby nedošlo k přeskočení jedné z nich, je v programu umístěn zmíněný TON. Ten zajistí, aby program detekoval pozitivní hranu najetí do HOME pozice až po uplynutí dvou sekund (náskok k rozjetí operace).



Obr. 113 - Program logiky robotu polotovaru šachta – třetí síť

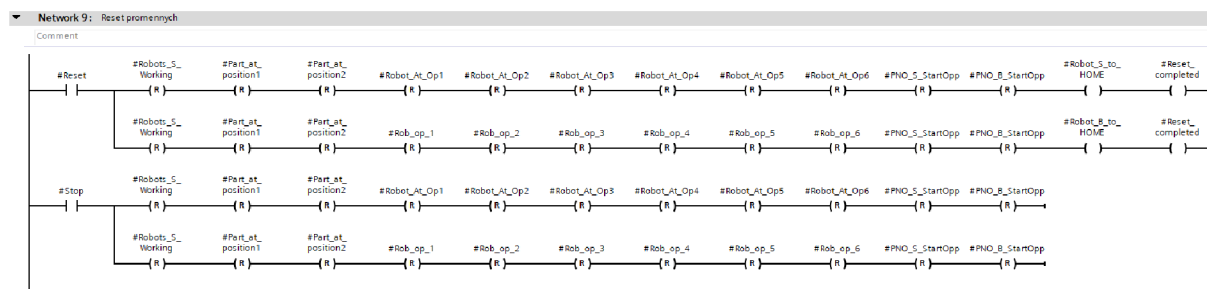
Sítě 4 – 7 vypadají totožně jako síť 3. Poslední krok cyklu je znázorněn v síti č. 8. Po dokončení poslední operace dojde k resetování všech pomocných proměnných a k sepnutí signálu o dokončení cyklu **Robot\_cycle\_Done**. Pokud není žádná operace v chodu, je tento pomocný signál také sepnut. Dále zde dochází k sepnutí signálu o chybě ve vstupech do cyklu (**Error\_inputs**).



Obr. 114 - Program logiky robotu polotovaru šachta – osmá síť

Síť 9 má na starosti reset všech proměnných na základě bezpečnostních tlačítek **Start** a **Stop**. Při zmáčknutí tlačítka Reset navíc dojde k robotickým operacím, kdy roboty najedou do své HOME pozice. Při stisku bezpečnostních tlačítek v reálném provozu by bylo nutné projít

bočním vchodem a zkontrolovat pozici robotů, aby při následném resetu nedošlo k jakékoliv kolizi, a v případě potřeby roboty vyvézt z nebezpečné pozice za použití kontroléru.

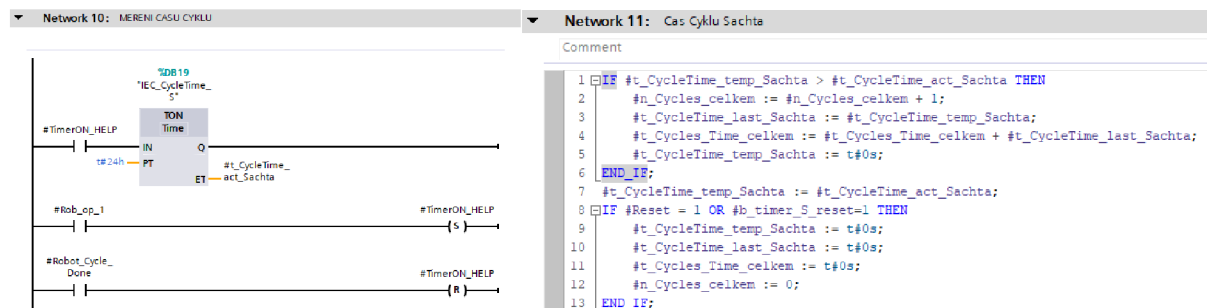


Obr. 115 - Program logiky robotu polotovaru šachta – devátá síť

Poslední dvě sítě (obr. 116) mají na starosti spínání časovače, který měří počet provedených cyklů, celkový čas cyklů, poslední čas cyklu a aktuální čas cyklu. Síť 10 zajišťuje sepnutí časovače při startu první robotické operace a resetu při dokončení operace.

Poslední síť byla vytvořena ve Strukturovaném jazyce z důvodu přehlednosti v této konkrétní aplikaci. První IF struktura posuzuje, zdali je dočasná lokální proměnná ( $t\_CycleTime\_temp\_Sachta$ ) větší než aktuální čas cyklu (uplynulý čas z časovače v síti 10). Pokud není, program pokračuje dál a dojde k zápisu aktuálního času do dočasné proměnné. Jakmile je dočasná proměnná větší než hodnota aktuálního času cyklu, cyklus se dokončí a resetuje se časovač. Tím dojde k zapsání nové celkové doby cyklů, jejich počtu a doby posledního cyklu.

Druhá IF struktura má na starosti vynulování uložených časových hodnot, pokud dojde k zmáčknutí centrálního Reset tlačítka, nebo pouze k lokálnímu  $b\_timer\_S\_reset$ .



a)

b)

Obr. 116 - Program logiky robotu pol. šachta a) zapnutí časovače b) uložení

### 7.7.2 Blatník

Ovládací program pro polotvary typu blatník je tvořen celkem z devíti sítí. Organizace všech sítí je naprosto totožná jako u polotovaru šachta.

První obsahuje vstupní požadavky. Sítě 2 – 5 zajišťují automatický chod cyklu robotických operací. Další dvě sítě mají na starost zakončení cyklu a chování při stisknutí tlačítka Stop a Reset (Reset zde již nevolá robotickou operaci HOME). Poslední dvě zajišťují zaznamenávání času.

▶ Network 1: Input Conditions
▶ Network 2: Start OP1 - svarovani LEVA
▶ Network 3: Start OP1 - svarovani PRAVA
▶ Network 4: Start OP 3 - FREZOVANI
▶ Network 5: Start OP4 - TCP
▶ Network 6: Cycle Done
▶ Network 7: Reset promennych
▶ Network 8: MERE NI CASU CYKLU
▶ Network 9: Cas: Cyklu Sachta

Obr. 117 - Program logiky robotu polotovaru blatník

## 7.8 FB Inicializace

Aby bylo možné zapnout automatický režim fungování zadaného pracoviště, bylo nutné vytvořit bezpečnostní podprogram, který zamezí zapnutí cyklu, pokud nejsou splněny určité požadavky. Funkční blok je tvořen inicializační strukturou pro jednotlivé prvky pracoviště a dvěma podprogramy pro detailnější otestování stavu robotů.

**Inicializace** má na starosti zjištění aktuálních hodnot všech prvků a přivést je do připraveného stavu pro automatický režim (ReadyToAuto). Funkční blok byl vytvořen v programovacím jazyku SCL.

Řádky 1 – 200 (obr. 119) slouží k jednoduchému řízení klapků a jejich zpětné vazbě.

**První podmínka** (řádek 258) zkontroluje, zdali se roboty nachází v HOME pozici a nevykonávají žádnou operaci. Pokud je vše v pořádku, dojde k sepnutí signálu RobotsReady. V reálném nasazení by bylo možné provést hlubší rozbor, např. MotorsOnState, RunChainOk, AutoOn atd. Tato simulace je také možná při využití dodatečného softwaru KUKA.Sim, který ovšem není k dispozici.

Další **dvě podmínky** (řádek 264-279) zkontrolují, zdali je zapnutý elektromotor a provedena operace homing (homování). Pokud ne, dojde ke spuštění obou příkazů. Po splnění podmínky se sepne signál Motor\_Ready.

**Čtvrtá podmínka** (řádek 280) má za úkol zkontrolovat pozici upínacích zařízení rotačního stolu. Žadáný počáteční stav je CLOSE, pokud se v něm prvky nenachází, dojde k sepnutí povelových signálů.

**Poslední podmínka** (řádek 289) se týká bezpečnostních dveří. Oba hlavní vchody a jeden vedlejší musí být zavřeny, jinak dojde k sepnutí signálu k jejich zavření.

Program navíc využívá pomocnou proměnnou n\_Error\_INIT, kde každý bit udává chybu, kvůli které nelze provést inicializaci. V případě, že je vše v pořádku, je hodnota rovna 0.

Pokud jsou splněny všechny základní podmínky, dojde k sepnutí pomocného signálu INICIALIZACE\_OK.

```

1 // Kontrola pozic up. prvku
2 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Klapky Blatnik CLOSED
22 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Klapky Blatnik OPENED
42 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Klapky Sachta OPENED
73 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Klapky Sachta CLOSED
104 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Klapky Sachta at WORK
135 // rizeni pozic up. prvku
136 IF #KLAPKY_B_OPEN THEN ... END_IF;
153 IF #KLAPKY_B_CLOSE THEN ... END_IF;
170 IF #KLAPKY_S_OPEN THEN ... END_IF;
200 IF #KLAPKY_S_CLOSE THEN ... END_IF;
230
231 // Kontrola vstupu // reset
232 #n_Error_INIT := 0;
233 DB_FB_INIT_ROBOT_DETAIL_S" (b_start:=#b_start_INIT, ...);
245 DB_FB_INIT_ROBOT_DETAIL_B" (b_start:=#b_start_INIT, ...);
257
258 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Podminka pro ROBOTS READY - 2
264 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Podminka pro Motor Sachta READY
272 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Podminka pro Motor Blatnik READY
280 IF #KLAPKY_B_CLOSED ... THEN ... END_IF; // Podminka pro Klapky CLOSED
289 IF "CONTROL_DATA"."SEN/... THEN ... END_IF; // Podminka Bez. Dvere CLOSED
300 // if VSE + b_reset = 0 a Laser_scanner =0
301 IF #RobotsREADY AND
302 #Motor_B_READY AND
303 #Motor_S_READY AND
304 #KLAPKY_CLOSED AND
305 #Bez_dvere_all_CLOSED_Ready THEN
306 #INICIALIZACE_OK := 1;
307 ;
308 END_IF;
309 IF #b_Robot_B_ReadyToWo... THEN ... END_IF;

```

Obr. 118 – Hlavní program inicializace

Podprogramy (řádek 233 a 245) slouží k detailnější inicializační simulaci robotů, zmíněných výše u první podmínky. Programy jsou vytvořeny ve strukturovaném jazyce a jejich struktura je tvořena primárně skrze struktury CASE s podmínkami IF. Viz **příloha č. 3**.

Řádek	Popis
1-8	Pozitivní hrana signálu Start dá příkaz ke spuštění pohonů a přepnutí do prvního stavu CASE struktury – STATE_OV_ERRORY.
10-19	STATE_OV_ERRORY Základní stav. Pokud proměnná n_Error = 0, kód přejde na další stav. Pokud ne, zapíše se číslo chyby do proměnné a program přejde do stavu STATE_ERROR.
21-28	STATE_OVERENI Kontrola RunChainOk. Pokud není splněno, přepne program do STATE_ERROR.
30-41	STATE_ERROR Pokud dojde k potvrzení chyby a stisknutí RESET tlačítka, dojde k resetu chybových informativních proměnných a program přejde znovu do prvního stavu.
44-51	STATE_INIT



	Kontrola AutoOn. Pokud není splněno, přepne program do STATE_ERROR.
<b>54-74</b>	STATE_MOTORS_ON Kontrola, jestli jsou zapnuty pohony. Pokud nejsou zapnuty, dojde k sepnutí signálu k jejich zapnutí. Pokud se pohony do dvou vteřin nenahodí, přecházíme do STATE_ERROR.
<b>77-93</b>	STATE_START_CYCLE Kontrola CycleOn. Pokud není splněno, přepne program do STATE_ERROR.
<b>96-114</b>	STATE_ROBOT_READY Pokud byly splněny všechny požadavky, dojde k sepnutí signálu b_RobotReadyToWork. Pokud ne, tak k přepnutí do STATE_ERROR. Pokud chceme inicializační proces provést znovu, je možné stisknout RESET tlačítko.

Tabulka 5 – Princip fungování programu (příloha č. 3)

Jelikož nebyl k dispozici reálný kontrolér, aby bylo možné provést simulovaný inicializační proces, bylo nutné v PS nuceně zapnout signály nutné k provedení programu. Výhodou byla možnost nanečisto odladit simulovaný program.

Pokud jsou splněny všechny základní podmínky a navíc podmínky o úspěšné inicializaci robotů, dojde k sepnutí pomocného signálu INIT\_KOMPLET\_OK.

Signal	Value	Color	Checkbox	Address
LB_exit4	63		<input type="checkbox"/>	0 I138
PNI_B_autoOn		Green	<input checked="" type="checkbox"/>	0 I181.5
PNI_B_cycleOn		Green	<input checked="" type="checkbox"/>	0 I181.0
PNI_B_motorsOffState		Red	<input type="checkbox"/>	0 I181.1
PNI_B_motorsOnState		Red	<input type="checkbox"/>	0 I181.2
PNI_B_reqOp	0		<input type="checkbox"/>	0 I
PNI_B_robotWorking		Red	<input type="checkbox"/>	0 I
PNI_B_runChainOk		Green	<input checked="" type="checkbox"/>	0 I181.4
PNI_S_autoOn		Green	<input checked="" type="checkbox"/>	0 I180.5
PNI_S_cycleOn		Green	<input checked="" type="checkbox"/>	0 I180.0

Obr. 119 – Simulace zpětné vazby robotů o jejich stavu

## 7.9 FB\_CONTROL

Všechny vytvořené řídicí programy popsané výše bylo následně možné využít k nadřazenému programu. K tomu byl vytvořen funkční blok FB\_Control (obr. 120). Jedná se o program vytvořený ve Strukturovaném textu, jehož hlavním účelem je rozdělit řízení na dva režimy, MANUAL a AUTO. V režimu AUTO jsou poté volány řídicí programy.

**Řádky 4–13** jednoduše zařizují přepínání mezi hlavními režimy. Z bezpečnostních důvodů byly k přepínání využity kladné hrany booleovských signálů b\_Automat a b\_Manual, definované v prvních dvou řádcích programu.

Case struktura v řádcích **16–213** obsahuje jednoduché rozdělení na AUTO / MANUAL. Automat se poté ještě dělí na INICIALIZACE a AUTO\_ON. Ve stavu AUTO\_ON jsou poté volány řídicí programy, popsané v předchozích kapitolách.

```

1 #re_Automat(CLK := #b_Automat);
2 #re_Manual(CLK := #b_Manual);
3
4 IF #re_Automat.Q THEN
5     #n_rezim := #STATE_AUTO;
6     #n_StateNum := #STATE_INIT;
7
8 END_IF;
9
10 IF #re_Manual.Q THEN
11     #n_rezim := #STATE_MANUAL;
12     #n_StateNum := #STATE_INIT;
13 END_IF;
14
15
16 CASE #n_rezim OF
17     #STATE_AUTO: // STATE AUTO
18     CASE #n_StateNum OF ... END_CASE;
19
200 ;
209
210     #STATE_MANUAL: // STATE MANUAL
211     ;
212     ;
213 END_CASE;
16 CASE #n_rezim OF
17     #STATE_AUTO: // STATE AUTO
18     CASE #n_StateNum OF
19         #STATE_INIT: // Kontrola vstupu // reset
20             "DB_FB_INIT_KOMPLET"();
21
22
23
24         IF "DB_FB_INIT_KOMPLET".INICIALIZACE_OK=1 AND #b_SkipINIT=1 THEN
25             #n_StateNum := #STATE_AUTO_ON;
26
27         END_IF
28     ;
29     #STATE_AUTO_ON: // Statement section case 2 to 4
30     "DB_LD_Robots_Sachta_Auto_P2"(Sensor_I1 := "CONTROL_DATA"."SEN/FB"
31     "DB_LD_Robots_Blatnik_Auto"(Sensor_I1 := "CONTROL_DATA"."SEN/FB".I
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

```

a)

b)

Obr. 120 – a) Hlavní program CONTROL, b) Detail na strukturu stavu AUTO

## 7.10 Main program PLC

Předpřipravené funkční bloky byly umístěny do programu MAIN. Zde dochází k jejich volání a realizaci.

První dva řádky slouží ke vzájemnému dosažení hodnot PLC tagů a řídicích signálů z centrální databáze CONTROL\_DATA. Řádky 4-185 pak vyvolávají převody signálů. Programy jsou popsány v kapitolách 7.2–7.4.

Následně je již volán centrální ovládací program (kap. 7.8) a nakonec řídicí programy pohybových os (kap. 7.5).

```

1 "OUTPUT_VAR_ASSIGNEMENT"();
2 "INPUT_VAR_ASSIGNEMENT"();
3 // Klapky STATUS WORD - BOOL / PS - TIA / INPUTS
4 "DB_TIA_W_to_B_1"(IN_WORD:="Clappers_S_P1_status", ...); //Klapky S Poz1 L - STATUS
20 "DB_TIA_W_to_B_2"(IN_WORD:="Clappers_S_P2_status", ...); //Klapky S Poz2 L - STATUS
36 "DB_TIA_W_to_B_3"(IN_WORD:="Clappers_B_P1_L_status", ...); //Klapky B Poz. LE - STATUS
53 "DB_TIA_W_to_B_4"(IN_WORD:="Clappers_B_P2_R_status", ...); //Klapky B Poz. PR - STATUS
71 "DB_TIA_W_to_B_5"(IN_WORD:="Clappers_S_P1_R_status", ...); //Klapky S Poz1 R - STATUS
86 "DB_TIA_W_to_B_6"(IN_WORD:="Clappers_S_P2_R_status", ...); //Klapky S Poz2 R - STATUS
101
102
103 // Klapky control BOOL - WORD / TIA - PS / OUTPUTS
104 "DB_TIA_B_to_W_1"(IN_0:="CONTROL_DATA".CONTROL.Q.Klapky_S_new.Klapky_S_P1_L1_CLOSE, ...); //Klapky S Poz1 L - CONTROL
121 "DB_TIA_B_to_W_2"(IN_0:="CONTROL_DATA".CONTROL.Q.Klapky_S_new.Klapky_S_P2_L1_CLOSE, ...); //Klapky S Poz2 L - CONTROL
136 "DB_TIA_B_to_W_3"(IN_0:="CONTROL_DATA".CONTROL.Q.Klapky_B_new.Klapky_B_L1_CLOSE, ...); //Klapky B Poz1 L - CONTROL
153 "DB_TIA_B_to_W_4"(IN_0:="CONTROL_DATA".CONTROL.Q.Klapky_B_new.Klapky_B_R1_CLOSE, ...); //Klapky B Poz1 R - CONTROL
170 "DB_TIA_B_to_W_5"(IN_0:="CONTROL_DATA".CONTROL.Q.Klapky_S_new.Klapky_S_P1_R1_CLOSE, ...); //Klapky S Poz1 R - CONTROL
185 "DB_TIA_B_to_W_6"(IN_0:="CONTROL_DATA".CONTROL.Q.Klapky_S_new.Klapky_S_P2_R1_CLOSE, ...); //Klapky S Poz2 R - CONTROL
202
203
204
205 "FB_CONTROL_DB"(); //hlavni program
206
207 "DB_MotionControl1"(EXE_Homing:="CONTROL_DATA".CONTROL.Q.MC_Out_S.b_EXE_Homing, ...); //MC - Pohon Stolu SACHTA
230 "DB_MotionControl2"(EXE_Homing:="CONTROL_DATA".CONTROL.Q.MC_Out_B.b_EXE_Homing, ...); //MC - Pohon Stolu BLATNIK

```

Obr. 121 – Struktura program MAIN

## 7.11 HMI

Posledním krokem virtuálního zprovoznění bylo vytvoření ovládacího HMI panelu, pomocí kterého by byla obsluha schopna ovládat celé pracoviště.

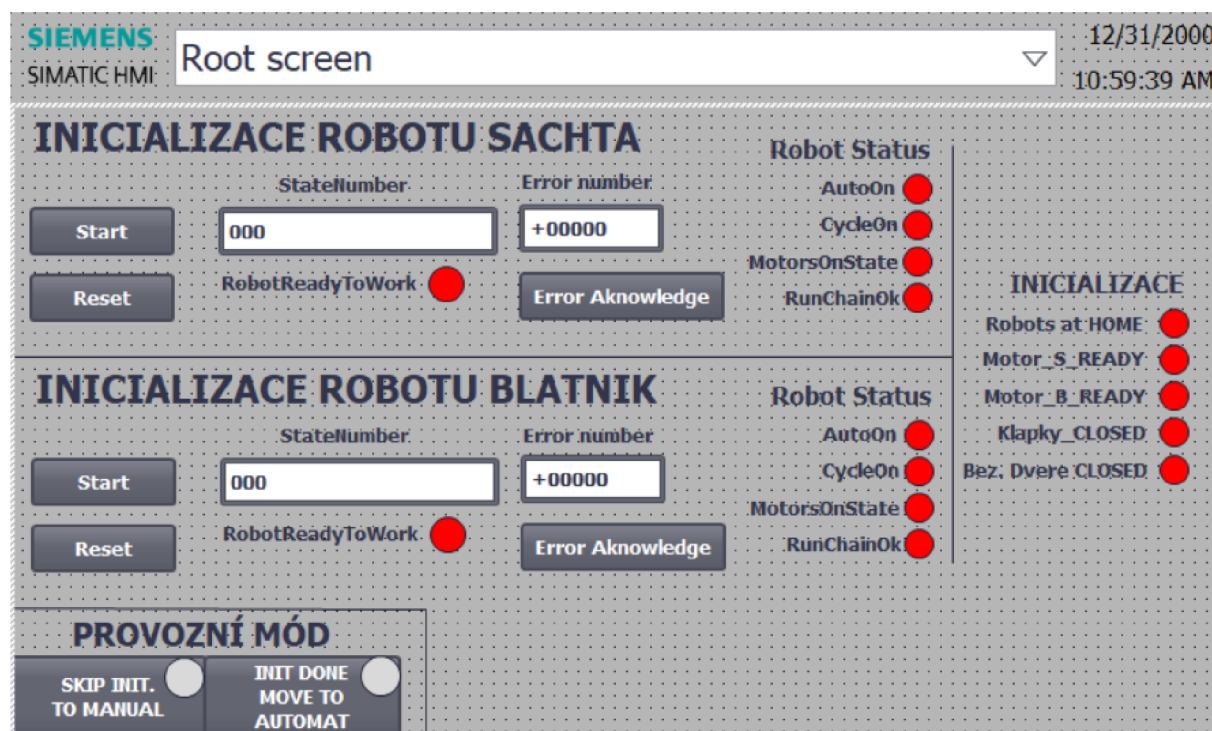
V práci bylo vytvořeno centrální ovládání celé buňky, které je možné jednoduše pozměnit a omezit funkce pouze na jednu část pracoviště, kterou právě operátor obsluhuje. Cílem bylo jednoduché, přehledné ovládání, které může fungovat ve dvou módech – MANUAL a AUTOMAT.

Úvodní obrazovka po načtení systému je tvořena pouze tlačítkem start, které po stisknutí přepne obrazovku na **inicializační** (obr. 122). Ta je určena k počátečnímu testu stavu robotů a dalších prvků pracoviště před tím, než bude možné přejít do automatického režimu.

Tlačítko start centrálně spustí inicializační podprogram obou robotů. Při chybě dojde k vypsání čísla chyby (Error number) a obsluha musí nejdříve stisknout tlačítko o potvrzení chyby a následně resetovat kontrolní podprogram. Fungování programu je popsáno v kap. 7.8.

Po splnění požadavků (Motors\_On, Cycle\_On, Auto\_On, RunChainOk) dojde k sepnutí signálu Robot\_ReadyToWork. Pokud jsou zároveň splněny další obecné požadavky pracoviště (Klapky\_CLOSED, Robots\_at\_HOME, Motors\_ON), dojde k zobrazení tlačítka přepínající obrazovku a stav na AUTO\_ON.

Pokud by obsluha chtěla přeskočit inicializační část a přejít pouze do manuálního režimu, je možné stisknout tlačítko SKIP INIT TO MANUAL. Při opětovném pokusu přejít do automatického režimu znovu dojde k přepnutí na inicializační obrazovku, kde je nutné splnit všechny požadavky.



Obr. 122 – HMI – INICIALIZACE

### 7.11.1 Automat

První mód je využíván pro automatický chod pracoviště. Režim je tvořen pouze jednou obrazovkou, která má prioritně informovat obsluhu o stavu důležitých prvků pracoviště.

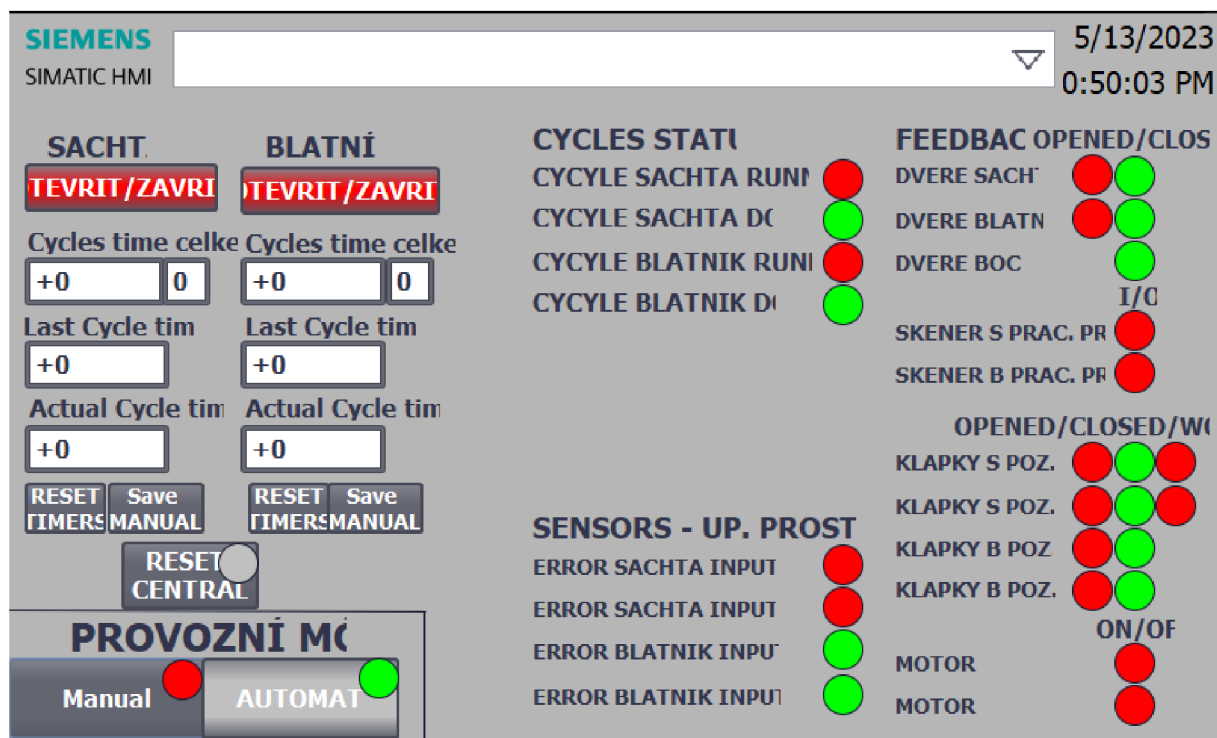
Na obr. 123 můžeme vidět rozložení obrazovky. Nejdůležitějším prvkem jsou tlačítka OTEVRIT/ZAVRIT. Jedná se o jediná tlačítka, která musí operátor obsluhovat. V reálném nasazení by tato tlačítka byla vyvedena do fyzického aretačního tlačítka spolu s bezpečnostním STOP tlačítkem.

Pravá strana obsahuje zpětnou vazbu důležitých signálů, například pozice upínacích prvků pracoviště, stav bezpečnostních dveří (OPEN/CLOSE) a samozřejmě i informační signály o tom, zda se provádí cyklus SACHTA, BLATNIK, nebo byla operace již dokončena. Signály v sekci SENSORS nám vyhodnocují, zdali je polotovar na pozici správně upevněn.

Na levé dolní straně je možné vidět hlavní tlačítka, přepínající režim robotického pracoviště. Při změně z režimu Automat na Manual dojde k zastavení všech probíhajících operací.

Na levé straně se poté nachází číselná pole, která ukazují aktuální čas cyklu, čas posledního cyklu, čas všech cyklů a počet provedených cyklů. Tyto informace jsou velmi užitečné k vizuální kontrole doby trvání cyklů. Pokud by došlo k výraznému časovému zpoždění mezi jednotlivými cykly, operátor by tak získal indikaci o problému na pracovišti.

Tlačítka umístěná pod číselnými poli slouží k resetu časových hodnot a manuálnímu uložení hodnot do souboru. Poslední RESET CENTRAL slouží k tvrdému restartu pracoviště, resetu všech proměnných do původních hodnot a roboty provedou pohyb do HOME pozice. Jedná se o tvrdý restart, kterému by obvykle předcházela fyzická tlačítka STOP.



Obr. 123 – HMI – Automat\_ON

Při každé změně stavu aretačního tlačítka poté dochází k **uložení všech časových hodnot** a aktuálního reálného času do definovaného textového souboru. Díky těmto hodnotám jsme dlouhodobě schopni statisticky vyhodnocovat, jak dlouho trval každý cyklus a jak dlouho operátorovi trvalo umístit polotovar.

Program zajišťující automatické ukládání hodnot do souboru byl vytvořen v programovacím jazyku VBSkript. Jedná se o vcelku zastaralý programovací jazyk, využívaný k vytváření webových stránek. Důvodem využití je automatická podpora tohoto programovacího jazyka v HMI v TIA.

Struktura kódu je poté velmi jednoduchá. V tab. 6 je rozepsána struktura tohoto programu (obr. 124).

Řádek	Popis
1	Název programu
3-7	Definice proměnných a následně cesty uložení, názvu a typu souboru
9, 21	Definování proměnné jako systémové funkce pro vytváření složek/souborů
13-17	Kontrola, zda existuje definovaná cesta, jinak dojde k vytvoření složky
23-37	Kontrola, zda existuje definovaný soubor, jinak dojde k vytvoření souboru Následně dojde k otevření souboru, zapsání prvního řádku (sloupců) a zavření souboru.
41-53	Otevření souboru a zapsání definovaných hodnot do volného řádku tabulky

Tabulka 6 – Princip fungování programu (obr. 124)

```

1 Sub tmpExport_S_OK()
2
3 Dim FolderPath, ObjectPath, Filename, File, FileExist, Columns, Row
4
5 FolderPath = "C:\Games\"
6
7 Filename = "Sachta_log.csv"
8
9 Set ObjectPath = CreateObject("Scripting.FileSystemObject")
10
11 'Check if folder exists
12
13 If Not ObjectPath.FolderExists(FolderPath) Then
14
15 ObjectPath.CreateFolder FolderPath
16
17 End If
18
19 ' Check if file exists
20
21 Set File = CreateObject("Scripting.FileSystemObject")
22
23 FileExist = File.FileExists(FolderPath & "\" & Filename)
24
25 If FileExist = False Then
26
27 File.CreateTextFile(FolderPath & "\" & Filename)
28
29 Set Columns = File.OpenTextFile(FolderPath & "\" & Filename, 8)
30
31 Columns.WriteLine("Time ; LAST_CYCLE [MS] ; CYCLES_CELKEM [MS] ; CYCLES_POCKET")
32
33 Columns.Close
34
35 Set File = Nothing
36
37 End If
38
39 'Write data to file
40
41 Set File = CreateObject("Scripting.FileSystemObject")
42
43 Set Row = File.OpenTextFile(FolderPath & "\" & Filename, 8)
44
45 Row.WriteLine(Time & ";" & SmartTags("DB_LD_Robots_Sachta_Auto_P2_t_CycleTime_last_Sachta") & ";" &
46 SmartTags("DB_LD_Robots_Sachta_Auto_P2_t_Cycles_Time_celkem") & ";" & SmartTags("DB_LD_Robots_Sachta_Auto_P2_n_Cycles_celkem") & ";")
47
48 Row.Close
49
50 Set File = Nothing
51 'Here we are releasing the File.
52
53 End Sub

```

Obr. 124 – HMI – Program pro automatické ukládání časů cyklů do souboru

A	B	C	D	A	B	C	D
1 Time	LAST_CYCLE [MS]	CYCLES_CELKEM [MS]	CYCLES_POCKET	1 Time	LAST_CYCLE [MS]	CYCLES_CELKEM [MS]	CYCLES_POCKET
2 9:44:36	0	0	0	2 22:26:48	10484	10484	1
3 9:44:44	0	0	0	3 22:27:52	10484	10484	1
4 9:46:59	113587	113587	1	4 9:42:46	0	0	0
5 9:47:00	113587	113587	1	5 9:42:53	0	0	0
6 9:53:48	0	0	0	6 9:43:20	13954	13954	1
7 9:53:53	0	0	0	7 9:43:25	13954	13954	1
8 10:09:34	0	0	0	8 9:44:09	35121	49075	2
9 10:09:37	0	0	0	9 9:44:14	35121	49075	2
10 10:09:44	0	0	0	10 9:45:40	40870	89945	3
11 10:09:49	0	0	0	11 9:45:45	40870	89945	3
12 13:44:44	0	0	0				
13 13:44:51	0	0	0				
14 23:45:53	0	0	0				
15 23:45:53	0	0	0				

Obr. 125 – Ukázka ukládaných hodnot (z programu na obr. 125)

## 7.11.2 Manual

Manuální režim slouží primárně k údržbě, servisu, instalaci a při dalších speciálních příležitostech. Mód je tvořen pomocí čtyř obrazovek (obr. 126) s přepínáním po levé straně uprostřed. Každá obrazovka je určena k ovládní a zpětné vazbě konkrétních skupin prvků pracoviště.

První z nich Klapky umožňuje operátorovi manuálně ovládat každý pneumatický válec (umožněno díky centrálnímu ventilovému bloku, viz kap. 6.3.1.). Levá dioda (červená) znázorňuje stav CLOSED (zavřeno) a pravá (zelená) stav OPENED (otevřeno). Pokud se upínací zařízení pohybuje mezi polohami, obě diody svítí neutrální šedou barvou.

Druhá obrazovka ROBOT obsahuje ovládní všech robotických operací a zpětnou vazbu o všech stavech signálů indukčních snímačů, detekujících upnutý polotovar na rotačním stole.

Další obrazovka (c) představuje ovládní dle názvu: POHONY STOLU. Zde jsme schopni manuálně zapínat pohon rotačních stolů, spouštět homing operaci a pohybovat stolem pomocí základních funkcí definovaných v kap. 7.5. Číselná pole umožňují nastavit žádanou rychlost pohybu a pozici, poté můžeme spustit pohyb absolutní nebo relativní.

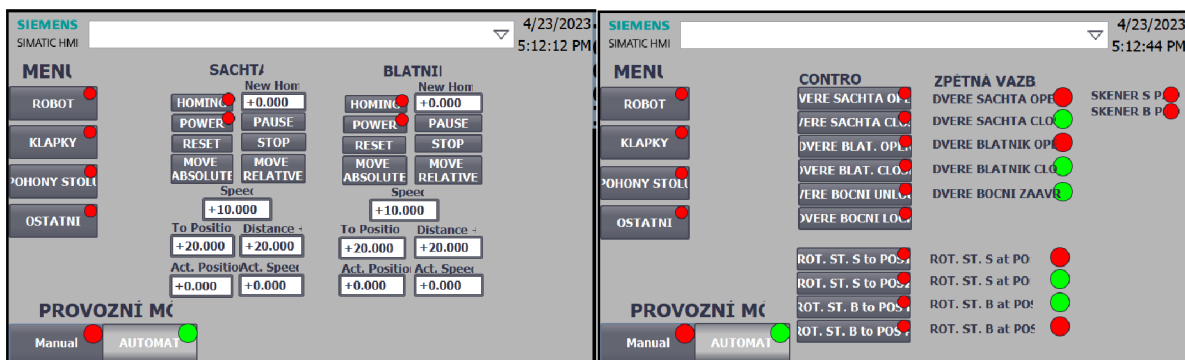
Poslední obrazovka obsahuje kontrolní tlačítka určené k ovládní bezpečnostních dveří.

Obrazovky byly vytvořeny tak, aby bylo uživateli na první pohled srozumitelné ovládní a zpětnovazební signálová struktura.



a)

b)



c)

d)

Obr. 126 – HMI – MANUAL a) Klapky, b) Robot, c) Motory, d) Ostatní

## 7.12 Robotický program

V kapitole 6.4 byly v PS vytvořeny podpůrné signály, které by byly využity v reálném nasazení robotů. Tyto signály jsme si předpřipravili následně i v TIA portálu (obr. 127) a byly by využity v reálném řízení robotů s kontrolérem.

Druhou možností by mohlo být využití softwaru KUKA.Sim, kde je možné věrně simulovat kontrolér a propojit tento software s PS. Jelikož tento software nebyl k dispozici, byl v PS vytvořen logický podprogram, který zjednodušeně simuluje robotickou logiku (kap. 6.4).

**Třetí možností simulace chování robotu** je využití plug-in TIA portálu s názvem SIMATIC Robot Library V1.1.0.0. Jedná se o simulační prostředí přímo v softwaru TIA, které podporuje celou řadu výrobců robotů. Funguje podobně jako řízení virtuálních os a frekvenčního měniče v kap. 7.5. Nevýhodou je časově omezená licence (dvě hodiny) a z toho důvodu tuto možnost využívat nebudeme.

ROBOT_CONTROL							
	Name	Data type					
1	PNI_S_cycleOn	Bool	16	PNI_B_cycleOn	Bool		
2	PNI_S_motorsOffState	Bool	17	PNI_B_motorsOffState	Bool		
3	PNI_S_motorsOnState	Bool	18	PNI_B_motorsOnState	Bool		
4	PNI_S_reqOp	DInt	19	PNI_B_reqOp	DInt		
5	PNI_S_robotWorking	Bool	20	PNI_B_robotWorking	Bool		
6	PNI_S_runChainOk	Bool	21	PNI_B_runChainOk	Bool		
7	PNO_S_motorsOff	Bool	22	PNO_B_motorsOff	Bool		
8	PNO_S_motorsOn	Bool	23	PNO_B_motorsOn	Bool		
9	PNO_S_reqOp	DInt	24	PNO_B_reqOp	DInt		
10	PNO_S_start	Bool	25	PNO_B_start	Bool		
11	PNO_S_startAtMain	Bool	26	PNO_B_startAtMain	Bool		
12	PNO_S_startOp(1)	Bool	27	PNO_B_startOp(1)	Bool		
13	PNO_S_stop	Bool	28	PNO_B_stop	Bool		
14	PNO_S_stopAtEndOfCycle	Bool	29	PNO_B_stopAtEndOfCycle	Bool		
15	PNI_S_autoOn	Bool	30	PNI_B_autoOn	Bool		

Obr. 127 – Signálová struktura robotů v TIA

Po vytvoření realistické signálové struktury a simulované logiky kontroléru, bylo posledním krokem vygenerovat robotické programy v PS ve formě, kterou jsme schopni importovat do reálného robotu.

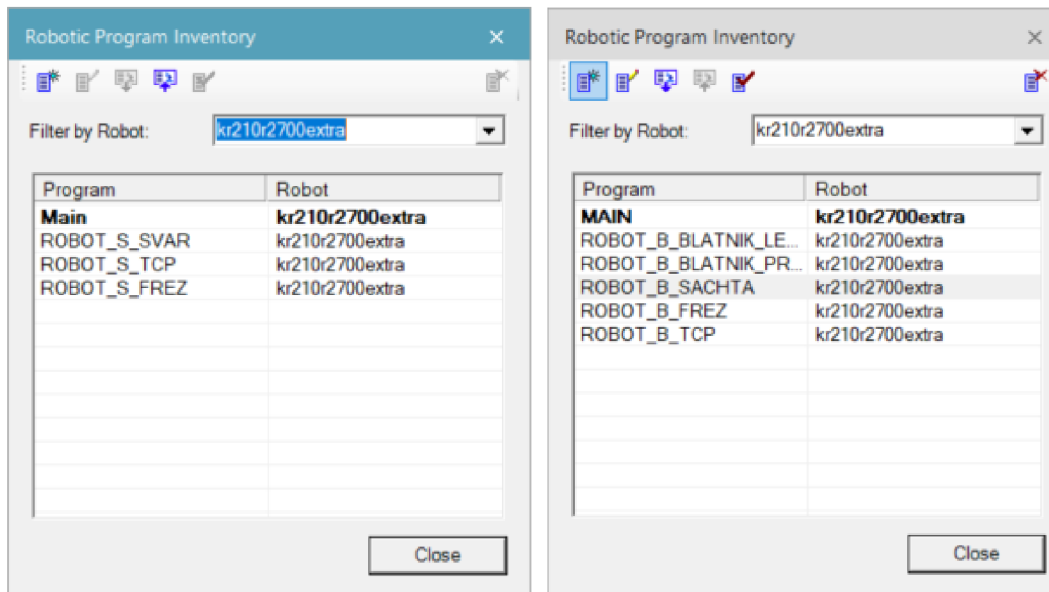
Abychom byli schopni vygenerovat soubor pro roboty, prvním krokem bylo upravit detaily jednotlivých robotických operací, tzn. rychlost, zónu, WObjecty a další. Po dokončení byly jednotlivé operace vloženy do robotického inventáře (obr. 128), kde jsme již jednoduše schopni exportovat robotický program.

Po exportu jsme obdrželi dva soubory. První z nich s příponou .DATA obsahuje údaje o všech bodech operace. Druhý typu .SRC obsahuje samotné pohybové povely.

Jelikož PS generuje kód v poněkud surovém stavu, následně by bylo nutné program upravit, například uspořádat, odstranit nadbytečné zakomentované řádky a přidat signály určené k ovládání svařovacího zdroje a nástroje. Jelikož nebyl k dispozici software KUKA, k úpravám byl zvolen software Visual Studio Code. Jedná se o univerzální editor zdrojového kódu od firmy Microsoft. Po stažení dodatečného plug-inu (zásuvný modul) bylo možné provádět úpravy robotického kódu (především hrubé syntaxe). Jelikož VS Code neumožňuje kontrolu syntaxi kódu, nebyly robotické programy dále upravovány. V reálném nasazení by totiž bylo nutné programy v každém případě doladit přímo na robotu.

Úpravy vygenerovaného kódu by spočívaly v manuálním přidání ovládacích signálů typu WAIT (čekej), WAIT FOR (čekej na), WELD (svažuj) atd. V původním stavu vygenerovaný program obsahuje návrh spouštěcích signálů ke svařování, ale u reálného robotu by bylo nutné strukturu upravit na míru pomocí nastavení v kontroléru.

Největší úpravou robotického programu bylo vytvoření **hlavního programu MAIN** ve VSC (obr. 130), který obsahuje rozřazovací program, určený k volání jednotlivých podprogramů (operací) na základě požadavku signálu ReqOp a startovacího signálu StartOp. Aby bylo možné přes tyto signály komunikovat, bylo by nutné v reálném kontroléru nakonfigurovat automatické externí signály.



a)

b)

Obr. 128 – Robotický inventář a) Robot\_S, b) Robot\_B

Paths & Locations	Config	Acc	Tool Nr	Base Nr	Motion	Speed	Zone	Storage...	OLP Commands	Duration
116810R01_Svare...										69.89
HOME	S 2 T 10	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE			0
via	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE			1.59
O1_P1	S 2 T 10	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	2.77
O1_P2	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	2.91
O1_P3	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	3.01
O1_P4	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	3.12
O1_P5	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	2.69
O1_P6	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	3.48
O1_P7	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE		# Destination zce_28_59d381004_118_00_20180229 # Drive C	2.45
via31	S 2 T 11	100 % (d...	1 (Fr) - E...	0 - Worl...	PTP	100 %	FINE			1.1

Obr. 129 – Základní konfigurace bodů robotických operací v PS



```

15  DEF CELL ( )
16      ...
17      INIT
18      BASISTECH INI
19      CHECK HOME
20      PTP HOME Vel= 100 % DEFAULT
21      AUTOEXT INI
22      LOOP
23          P00 (#EXT_PGNO,#PGNO_GET,DMY[],0 )
24          SWITCH PGNO ; Select with Programnumber
25              |
26              CASE 1
27                  P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
28                  IF #EXT_StartOp THEN
29                      ;EXAMPLE1 ( ) ; Call User-Program
30                  ENDIF
31              |
32              CASE 2
33                  P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
34                  IF #EXT_StartOp THEN
35                      ;EXAMPLE1 ( ) ; Call User-Program
36                  ENDIF
37              |
38              CASE 3
39                  P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
40                  IF #EXT_StartOp THEN
41                      ;EXAMPLE1 ( ) ; Call User-Program
42                  ENDIF
43              |
44              DEFAULT
45                  P00 (#EXT_PGNO,#PGNO_FAULT,DMY[],0 )
46          ENDSWITCH
47      ENDLOOP
48  END
    
```

Obr. 130 – Předpřipravený MAIN program robotů

Řádek	Popis
15	Název programu
23	Volání hodnoty PNO_ReqOp (z pohledu robotu input)
24	Na základě hodnoty PNO_ReqOp dojde k přepnutí na žádaný CASE (případ)
26-30	Dojde k odeslání zpětné vazby o přijaté hodnotě PNO_ReqOp a spuštění podprogramu, pokud EXT_StartOp=true
45-46	Případ DEFAULT slouží k odeslání chybové hlášky o špatné hodnotě (neexistuje pro ni případ) vstupního signálu

Tabulka 7 – Princip fungování programu (obr. 130)



## 8 ZHODNOCENÍ A DISKUZE

V práci byly splněny všechny žádané cíle s myšlenkou vytvořit virtuální zprovoznění s co možná nejpodobnější signálovou strukturou té skutečné. Zadané virtuální pracoviště obsahuje celou řadu prvků a téměř neomezené množství cest, které by umožňovaly projekt dále vylepšovat a přibližovat reálnému fungování, a tím i dokonalému virtuálnímu dvojčeti.

Z pohledu **robotů** by bylo optimální využít dodatečný simulační software (od výrobce robotů), díky kterému by bylo možné simulovat kontrolér a upravit vygenerovaný robotický program z PS tak, aby odpovídal jedna ku jedné reálnému kontroléru. Náhrada programu v podobě VS Code je z pohledu optimalizace robotického programu vhodná jen na vytváření základních struktur. V reálném nasazení by ovšem v každém případě bylo nutné projít svářeč body robotických operací a nastavit optimální podmínky sváření přímo na reálném polotovaru.

Z pohledu **PS** by dále bylo možné vytvořit uchopovací operace operátorů, provést analýzu jejich ergonomie a následně návrhy úprav pracoviště tak, aby byl systém více vstřícný a bezpečný operátorovi. Poté by bylo vhodné vytvořit dodatečnou signálovou strukturu simulující informační a bezpečnostní sensoriku (tlak pneumatických rozvodů, teplota v různých částech pracoviště, detekce nepovolaných osob v blízkém okolí PP atd.).

Z pohledu **TIA (PLC)** by bylo vhodné hlouběji řešit funkční bezpečnost pracoviště (například pomocí dodatečného safety PLC), a tím snížit pravděpodobnost vzniku nebezpečných situací a rizik. Dále by bylo možné řídicí podprogramy zjednodušit a vhodněji optimalizovat za účelem vytvoření univerzální šablony ovládacích podprogramů.

**HMI** obrazovky jsou vytvořeny bez nadbytečných animací. Důvodem je přehlednost a orientace. Dalším rozšířením by mohlo být např. vytvoření obrazovky s 3D pohledem na pracoviště. Tento pohled by byl vhodný pro nového operátora linky. Dále by zde v reálném využití bylo vhodné vytvořit samostatnou obrazovku určenou pouze k servisním pracím.



## 9 ZÁVĚR

Práce byla rozdělena do několika částí. V první rešeršní části byl přiblížen význam virtuálního zprovoznění a teoreticky přiblíženy všechny prvky, které jsou ke správnému oživení pracoviště třeba znát. Např. roboty, PLC, komunikační protokoly, softwary PS a TIA Portal.

Dalším krokem bylo rozebrat zadané neživé pracoviště, prvek po prvku. Důvodem bylo hlubší pochopení problematiky a širší rozhled nad danou problematikou. Výsledkem je přehled celé virtuální buňky, díky kterému bylo možné vytvořit logický postup, jak optimálně oživit tento výrobní systém.

Virtuální zprovoznění poté bylo rozděleno na dvě části. První z nich bylo oživení pracoviště v softwaru Process Simulate. To znamenalo vytvoření signálové struktury, logiky a animace každé jednotlivé součásti, rozebrané v předešlém kroku. Po každé interakci byl každý prvek testován pouze v PS bez vyššího řídicího systému. Teprve po kompletním (prozatímním) oživení pracoviště v PS bylo následně vytvořeno nové simulované PLC v PLCSim Advanced a softwary propojeny.

Poté bylo možné začít s druhou částí virtuálního zprovoznění, a to s virtuálním kontrolérem. V práci byl využit software TIA Portal pro programování virtuálního PLC. Prvním krokem byl export signálové struktury z PS do TIA a test vzájemné signálové komunikace mezi softwary. Po úspěšném otestování byla vytvořena centrální kontrolní databáze signálů, díky které byla zajištěna přehlednost signálové struktury celého projektu.

Dalším krokem bylo vytvořit řídicí podprogramy výrobního systému. Nejdříve bylo vhodné vytvořit myšlenkovou mapu a vývojový diagram. Výsledný program je rozdělen na dva primární podprogramy (operátor a roboty), vedlejší podprogramy, které řeší požadovanou inicializaci prvků pracoviště před zapnutím automatického režimu, a jeden hlavní program, řídicí všechny ostatní podprogramy, a tím i logiku celého pracoviště, včetně základních bezpečnostních prvků.

Posledním krokem virtuálního zprovoznění bylo vytvořit ovládací HMI a definovat jeho obrazovky, využitě k ovládání pracoviště v režimu MANUAL či AUTOMAT.

Kromě řídicích programů byl software TIA využit i pro vytvoření síťové struktury mezi všemi využitými komponentami (PLC, HMI a VFD). Díky tomu bylo možné definovat IP adresy a komunikační protokoly mezi jednotlivými komponentami, a tím detailněji virtuálně předpřipravit model k reálnému nasazení.

Mimo TIA Portal a PS, byly k práci využity další softwary. Např. Matlab (skripty upravující tabulky signálů) a VS Code (provizorní ladění vygenerovaného robotického programu a vytváření nového MAIN programu, volajícího žádané operace/podprogramy).

Výsledkem práce je virtuálně zprovozněný model zadaného pracoviště, který přináší výhodu off-line testování změn pracoviště, nových polotovarů, robotických programů a v neposlední řadě vizualizace zajišťující detailní přehled nad výrobním systémem.

Na závěr lze říci, že byly splněny zadané cíle této diplomové práce. Dodatečným osobním cílem bylo vytvořit virtuální zprovoznění co nejněpodobnější tomu reálnému, a i ten byl v rámci možností splněn. Zadané pracoviště je velmi obsáhlé, ale i nadále by bylo možné v práci hlouběji pokračovat.



# 10 SEZNAM POUŽITÝCH ZDROJŮ

## 10.1 Zdroje použité literatury

- [1] TURNBULL, CHARLOTTE. What is Virtual Commissioning. In: *Virtual Commissioning* [online]. 30-12-2022 [cit. 2023-05-22]. Dostupné z: <https://virtualcommissioning.com/what-is-virtual-commissioning/>
- [2] BRAŽINA, Jakub. Virtuální zprovoznění výrobního systému [online]. Brno, 2019 [cit. 2023-05-22]. Dostupné z: <https://www.vutbr.cz/studenti/zav-prace/detail/116785> . Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav výrobních strojů, systémů a robotiky. Vedoucí práce Jan Vetiška
- [3] Virtuální zprovoznění výrobních linek. In: *ElektroPrůmysl* [online]. Ostrava: TAURID OSTRAVA, 30-7-2022 [cit. 2023-05-22]. Dostupné z: <https://www.elektroprumysl.cz/software/virtualni-zprovozneni-vyrobnich-linek>
- [4] Bodové svařování. In: *ROCKWELD GROUP* [online]. PRAHA: ROCKWELD GROUP, 2013 [cit. 2023-05-22]. Dostupné z: <http://rockweld.cz/bodove-svarovani/>
- [5] BUCHAL, D. Virtuální model stroje Tecnomatix Process Simulate. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 65 s. Vedoucí bakalářské práce Ing. Jakub Arm.
- [6] What is robotics virtual commissioning. In: *Siemens Software* [online]. Mnichov, 2022 [cit. 2023-05-22]. Dostupné z: <https://www.plm.automation.siemens.com/global/en/our-story/glossary/what-is-robotics-virtual-commissioning/102446>
- [7] Programovatelný logický automat. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2023-05-22]. Dostupné z: [https://cs.wikipedia.org/wiki/Programovateln%C3%BD\\_logick%C3%BD\\_automat](https://cs.wikipedia.org/wiki/Programovateln%C3%BD_logick%C3%BD_automat)
- [8] Co je programovatelný logický automat. In: *ElektroPrůmysl* [online]. 2011 - 2023, 8-9-2022 [cit. 2023-05-22]. Dostupné z: <https://www.elektroprumysl.cz/automatizace/co-je-programovatelnny-logicky-automat-plc-1-cast>
- [9] Programmable logic controller. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2023 [cit. 2023-05-22]. Dostupné z: [https://en.wikipedia.org/wiki/Programmable\\_logic\\_controller](https://en.wikipedia.org/wiki/Programmable_logic_controller)
- [10] STAFF, Editorial. Top PLC Manufacturers. In: *Inst Tools* [online]. 2023 [cit. 2023-05-22]. Dostupné z: <https://instrumentationtools.com/top-plc-manufacturers/>
- [11] Top 20 PLC Manufacturers. In: *Instrumentation Blog* [online]. 2023 [cit. 2023-05-22]. Dostupné z: <https://instrumentationblog.com/plc-manufacturers-plc-brands/>
- [12] What is difference between PLC and SCADA. In: *Control Freaks* [online]. Spalding, 2020, 18-5-2020 [cit. 2023-05-22]. Dostupné z: <https://www.controlfreaksltd.co.uk/what-is-the-difference-between-plc-and-scada/>
- [13] PLC vs. HMI. In: *Software Real-Time Automatinon* [online]. Yorkville, 08-05-2022 [cit. 2023-05-22]. Dostupné z: <https://www.empoweredautomation.com/plc-vs-hmi-what-s-the-difference-between-these-two-industrial-automation-technologies>
- [14] Variable frequency drive. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2023 [cit. 2023-05-22]. Dostupné z: [https://en.wikipedia.org/wiki/Variable-frequency\\_drive](https://en.wikipedia.org/wiki/Variable-frequency_drive)

- [15] VFD PID Controller. In: *VFD - Variable Frequency Drives* [online]. 2023 [cit. 2023-05-22]. Dostupné z: <http://www.vfds.org/vfd-pid-control-997149.html>
- [16] Why using PID Control on VFD?. In: *GoHZ* [online]. Golden springs, 17-7-2016 [cit. 2023-05-22]. Dostupné z: <http://www.gohz.com/why-using-pid-control-on-vfd>
- [17] What is PID Controller and how it works. In: *PLCynergy* [online]. 16-1-2021 [cit. 2023-05-22]. Dostupné z: <https://plcynergy.com/pid-controller/>
- [18] PID tuning methods. In: *INCATools* [online]. Bosscheweg [cit. 2023-05-22]. Dostupné z: <https://www.incatools.com/pid-tuning/pid-tuning-methods/>
- [19] Industrial robot. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2023 [cit. 2023-05-22]. Dostupné z: [https://en.wikipedia.org/wiki/Industrial\\_robot](https://en.wikipedia.org/wiki/Industrial_robot)
- [20] FAIRCHILD, Mark. Types of Industrial Robots and Their Different Uses. In: *HowToRobot* [online]. 31-08-2021 [cit. 2023-05-22]. Dostupné z: <https://howtorobot.com/expert-insight/industrial-robot-types-and-their-different-uses>
- [21] Programmable Logic Controller Basics. In: *C3controls* [online]. Beaver, 2023 [cit. 2023-05-22]. Dostupné z: <https://www.c3controls.com/white-paper/back-to-plc-basics-guide-programmable-logic-controller/>
- [22] Robot YuMi. In: *Technický Deník* [online]. Praha, 29-06-2020 [cit. 2023-05-22]. Dostupné z: [https://www.technickytydenik.cz/rubriky/denni-zpravodajstvi/robot-yumi-ktery-slavi-5-let-prijima-vyzvy-i-ve-zdravotnictvi\\_50644.html](https://www.technickytydenik.cz/rubriky/denni-zpravodajstvi/robot-yumi-ktery-slavi-5-let-prijima-vyzvy-i-ve-zdravotnictvi_50644.html)
- [23] Industrial Robot Functionality and Coordinate System. In: *Inlearc* [online]. Tallinn [cit. 2023-05-22]. Dostupné z: <https://www.tthk.ee/inlearcs/industrial-robot-functionality-and-coordinate-systems/>
- [24] MUELANER, Jody. Wireless Protocols for Industrial Automation. In: *DigiKey* [online]. Minnesota, 1995, 06-05-2021 [cit. 2023-05-22]. Dostupné z: <https://www.digikey.cz/en/articles/comparing-wireless-protocols-for-industrial-automation>
- [25] IEEE\_802.11. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2023 [cit. 2023-05-22]. Dostupné z: [https://en.wikipedia.org/wiki/IEEE\\_802.11](https://en.wikipedia.org/wiki/IEEE_802.11)
- [26] Welding gun or spot welding. In: *NIMAK GmbH* [online]. Wissen [cit. 2023-05-22]. Dostupné z: <https://www.nimak.de/en/spotweldinggun/>
- [27] Nouzová vrata. In: *ASSA ABLOY* [online]. [cit. 2023-05-22]. Dostupné z: <https://www.assaabloyentrance.com/cz/cs/solutions/products/high-speed-doors/emergency-exit-doors/assa-abloy-rr300-plus-emergency-exit-door>
- [28] Industrial Communication Protocols. In: *ASSA ABLOY* [online]. Northampton [cit. 2023-05-22]. Dostupné z: <https://www.machinemetrics.com/connectivity/protocols>
- [29] CHROMČÍK, Adam. *Návrh virtuálního modelu robotického pracoviště* [online]. Brno, 2018 [cit. 2023-05-24]. Dostupné z: [https://www.vutbr.cz/www\\_base/zav\\_prace\\_soubor\\_verejne.php?file\\_id=174311](https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=174311). Diplomová práce. Vysoké učení technické v Brně. Vedoucí práce Jan Vetiška.
- [30] HAMPL, Lukáš. *Návrh a realizace robotických operací v systému Tecnomatix Process Simulate*. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně. Vedoucí práce Václav Kaczmarczyk.



- [31] *SINAMICS G: Positioning G120 with S7-1500 and HMI* [online]. In: . Munich: SIEMENS, 07-2018, s. 67 [cit. 2023-05-24]. Dostupné z: [https://cache.industry.siemens.com/dl/files/970/81666970/att\\_956174/v1/81666970\\_G120\\_at\\_S7-1500\\_TO\\_DOC\\_v1d2\\_en.pdf](https://cache.industry.siemens.com/dl/files/970/81666970/att_956174/v1/81666970_G120_at_S7-1500_TO_DOC_v1d2_en.pdf)
- [32] *KUKA System Software 8.3: Operating and Programming Instructions for System Integration* [online]. In: . Augsburg: KUKA Roboter, 14-01-2015, s. 491 [cit. 2023-05-24]. Dostupné z: <http://www.wtech.com.tw/public/download/manual/kuka/krc4/KUKA%20KSS-8.3-Programming-Manual-for-SI.pdf>
- [33] *KUKA Reference Guide v4\_1: Software - KR C1 / KR C2 / KR C3* [online]. In: . Augsburg: KUKA Roboter, s. 138 [cit. 2023-05-24]. Dostupné z: [http://robot.zaab.org/wp-content/uploads/2014/04/KRL-Reference-Guide-v4\\_1.pdf](http://robot.zaab.org/wp-content/uploads/2014/04/KRL-Reference-Guide-v4_1.pdf)
- [34] *KUKA ServoGun TC 4.1: For KUKA System Software 8.2* [online]. In: . Augsburg: KUKA Roboter, 29-07-2013, s. 125 [cit. 2023-05-24]. Dostupné z: [http://supportwop.com/IntegrationRobot/content/5-Applicatifs\\_metiers\\_KST/ServoGun/KST\\_ServoGun\\_TC\\_41\\_en.pdf](http://supportwop.com/IntegrationRobot/content/5-Applicatifs_metiers_KST/ServoGun/KST_ServoGun_TC_41_en.pdf)
- [35] *TIA Portal STEP 7 Basic V10.5* [online]. In: . NÜRNBERG: Siemens, 12-2009, s. 202 [cit. 2023-05-24]. Dostupné z: [https://cache.industry.siemens.com/dl/files/542/40263542/att\\_829827/v1/GS\\_STEP7Bas105enUS.pdf](https://cache.industry.siemens.com/dl/files/542/40263542/att_829827/v1/GS_STEP7Bas105enUS.pdf)
- [36] TIA Portal - Documents. In: *Siemens* [online]. Augsburg, 2023 [cit. 2023-05-24]. Dostupné z: <https://support.industry.siemens.com/cs/document/65601780/tia-portal-an-overview-of-the-most-important-documents-and-links-controller?dti=0&lc=en-CZ>
- [37] Siemens - General Forum. *Siemens* [online]. Augsburg, 2023 [cit. 2023-05-24]. Dostupné z: <https://community.sw.siemens.com/s/topic/0TO4O000000MilkWAC/general-forum>
- [38] *Process Simulate - Manual* [online]. Siemens, 2021 [cit. 2023-05-24]. Dostupné z: [https://docs.sw.siemens.com/en-US/doc/288782031/PL20201130102000506.tecnomatix\\_eMS\\_sc.xid1015765/xid1657141](https://docs.sw.siemens.com/en-US/doc/288782031/PL20201130102000506.tecnomatix_eMS_sc.xid1015765/xid1657141)

## 10.2 Zdroje obrázků

- [1] Digitalisierung im engineering infografik. In: *Heitec* [online]. Erlangen: Heitec, 2022 [cit. 2023-05-14]. Dostupné z: [https://automation.heitec.de/02-automatisierung/02-loesungen/01-heitec-40/en/image-thumb\\_1134\\_content-full/digitalisierung-im-engineering-infografik-en@2x.webp](https://automation.heitec.de/02-automatisierung/02-loesungen/01-heitec-40/en/image-thumb_1134_content-full/digitalisierung-im-engineering-infografik-en@2x.webp)
- [2] Odporové svařování. In: *ROCKWELD-GROUP* [online]. PRAHA: ROCKWELD-GROUP, 2013 [cit. 2023-05-14]. Dostupné z: <https://rockweld.cz/wp-content/gallery/bodove-svarovani/Odporov%C3%A9-sva%C5%99ov%C3%A1n%C3%AD.jpg>
- [3] Bodové svařování. In: *ROCKWELD-GROUP* [online]. PRAHA: ROCKWELD-GROUP, 2013 [cit. 2023-05-14]. Dostupné z: <https://rockweld.cz/wp-content/gallery/bodove-svarovani/Odporov%C3%A9-sva%C5%99ov%C3%A1n%C3%AD.jpg>

[content/gallery/bodove-svarovani/Bodov%C3%A9-  
sva%C5%99ov%C3%A1n%C3%AD\\_1.jpg](#)

- [4] Dual quick changer. In: SAMSYS [online]. Alzey: SAMSYS, 2023 [cit. 2023-05-14]. Dostupné z: <https://www.samsys.eu/wp-content/uploads/2021/09/dualquickchanger-6.jpg>
- [5] Cartesian Robot. In: *Linear Motion Tips* [online]. British Virgin Islands: RobotWorx, 2018 [cit. 2023-05-14]. Dostupné z: <https://www.linearmotiontips.com/wp-content/uploads/2015/12/CartesianRobotWorkspace-300x210.bmp>
- [6] Schematic diagram of a SCARA robot. In: *HowToRobot* [online]. HowToRobot, 2021 [cit. 2023-05-14]. Dostupné z: <https://howtorobot.com/sites/default/files/inline-images/Schematic%20diagram%20of%20a%20SCARA%20robot%20%20-%20946%20X%20751%20-%20%28from%20shutterstock%20500841208%29.jpg>
- [7] 6 Axis Robot. In: *RobotMotion* [online]. Bridgnorth: RobotMotion, 2021 [cit. 2023-05-14]. Dostupné z: <https://www.robomotion.co.uk/wp-content/uploads/2020/09/6-axis-robot-2048x1357.jpg>
- [8] ABB QuMi II. In: *Plastics Technology* [online]. Oklahoma: Plastics Technology, 2015 [cit. 2023-05-14]. Dostupné z: <https://d2n4wb9orp1vta.cloudfront.net/cms/ABB%20YuMi%20II.jpeg;maxWidth=600>
- [9] Delta Robot. In: *B&R Industrial Automation* [online]. Brno: B&R, 2023 [cit. 2023-05-14]. Dostupné z: <https://www.br-automation.com/fileadmin/1604846168210-de-html-1.0.jpg>
- [10] High precision Low accuracy. In: *Wikipedie* [online]. Brno: Wikipedie, 2018 [cit. 2023-05-14]. Dostupné z: [https://upload.wikimedia.org/wikipedia/commons/thumb/3/3a/High\\_precision\\_Low\\_accuracy.svg/1024px-High\\_precision\\_Low\\_accuracy.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/3/3a/High_precision_Low_accuracy.svg/1024px-High_precision_Low_accuracy.svg.png)
- [11] High accuracy Low precision. In: *Wikipedie* [online]. Brno: Wikipedie, 2018 [cit. 2023-05-14]. Dostupné z: [https://upload.wikimedia.org/wikipedia/commons/thumb/1/10/High\\_accuracy\\_Low\\_precision.svg/1024px-High\\_accuracy\\_Low\\_precision.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/1/10/High_accuracy_Low_precision.svg/1024px-High_accuracy_Low_precision.svg.png)
- [12] Robotics Coordinate System. In: *Learnchannel* [online]. Learnchannel, 2017 [cit. 2023-05-14]. Dostupné z: <https://learnchannel-tv.com/wp-content/uploads/2020/10/Robotics-coordinate-systems.png>
- [13] TCP calibration. In: *Emeral Insight* [online]. Bingley: Emeral Insight, 2019 [cit. 2023-05-14]. Dostupné z: [https://www.emerald.com/insight/proxy/img?link=resource/id/urn:emeraldgroup.com:asset:id:article:10\\_1108\\_IR-03-2019-0040/urn:emeraldgroup.com:asset:id:binary:IR-03-2019-004029.tif](https://www.emerald.com/insight/proxy/img?link=resource/id/urn:emeraldgroup.com:asset:id:article:10_1108_IR-03-2019-0040/urn:emeraldgroup.com:asset:id:binary:IR-03-2019-004029.tif)
- [14] TCP RobotStudio. In: *Autodesk* [online]. San Francisco: Autodesk, 2018 [cit. 2023-05-14]. Dostupné z: <https://forums.autodesk.com/t5/image/serverpage/image-id/555863iC739229F59C3B226/image-size/large?v=v2&px=999>
- [15] TraccTCP. In: *KUKA* [online]. Augsburg: KUKA, 2023 [cit. 2023-05-14]. Dostupné z: [https://www.kuka.com/-/media/kuka-corporate/images/products/software/software\\_tracctcp.jpg?rev=d49250d5ae6a4cb68bd62b1a684a5a9d&hash=5CF2EE9070FDC1504148E88D638BA2C4](https://www.kuka.com/-/media/kuka-corporate/images/products/software/software_tracctcp.jpg?rev=d49250d5ae6a4cb68bd62b1a684a5a9d&hash=5CF2EE9070FDC1504148E88D638BA2C4)

- [16] TCP. In: *Robot Welding* [online]. Robot Welding, 2001 [cit. 2023-05-14]. Dostupné z: <http://www.robot-welding.com/images/TCP.jpg>
- [17] Robot Command. In: *Control.com* [online]. Shawn Dietrich, 2022 [cit. 2023-05-14]. Dostupné z: [https://control.com/uploads/articles/robotcommand\\_2.jpeg](https://control.com/uploads/articles/robotcommand_2.jpeg)
- [18] KUKA smart pad. In: *Industrial robot book* [online]. Industrial robot book, 2019 [cit. 2023-05-14]. Dostupné z: <https://roboticsbook.com/wp-content/uploads/2019/04/kuka-smart-pad-robot-teach-pendant.jpg>
- [19] PLC scan cycle. In: *C3controls* [online]. Beaver: c3controls, 2023 [cit. 2023-05-14]. Dostupné z: [https://img.c3controls.com/image/upload/c\\_fill,dpr\\_auto,f\\_auto,fl\\_lossy,g\\_west,h\\_500,q\\_auto,w\\_auto/v1/content/c3controls-PLC-Scan-Cycle](https://img.c3controls.com/image/upload/c_fill,dpr_auto,f_auto,fl_lossy,g_west,h_500,q_auto,w_auto/v1/content/c3controls-PLC-Scan-Cycle)
- [20] LOGO!. In: *WiAutomation* [online]. Napoli: WiAutomation, 2023 [cit. 2023-05-14]. Dostupné z: <https://cz.wiautomation.com/public/images/landing/anticipa/product/6ED10521CC080BA1.jpg>
- [21] EATON Compact PLC. In: *Eaton* [online]. Pohořelice: Eaton, 2023 [cit. 2023-05-14]. Dostupné z: [https://www.eaton.com/content/dam/eaton/products/industrialcontrols-drives-automation-sensors/en-globalprime/programmable-logic-controllers-\(plc\)/ec4p-programmable-logic-controllers/eaton-ec4p-compact-plc-with-display-1000x1000.jpg](https://www.eaton.com/content/dam/eaton/products/industrialcontrols-drives-automation-sensors/en-globalprime/programmable-logic-controllers-(plc)/ec4p-programmable-logic-controllers/eaton-ec4p-compact-plc-with-display-1000x1000.jpg)
- [22] Modular PLC. In: *Wiautomation* [online]. Napoli: wiautomation, 2023 [cit. 2023-05-14]. Dostupné z: <https://cz.wiautomation.com/public/images/landing/anticipa/product/6ES75121CK010AB0.jpg>
- [23] Modular PLC. In: *Siemens* [online]. Mnichov: Siemens, 2023 [cit. 2023-05-14]. Dostupné z: <https://assets.new.siemens.com/siemens/assets/api/uuid:7d2490b0-9b33-4b02-a571-e51b8310ea1c/width:1024/quality:HIGH/7d2490b0-9b33-4b02-a571-e51b8310ea1c-high.webp>
- [24] VFD. In: *Electricla Technology* [online]. Electricla Technology, 2023 [cit. 2023-05-14]. Dostupné z: <https://www.electricaltechnology.org/wp-content/uploads/2021/10/Variable-Frequency-Drive-Circuit-Diagram.png>
- [25] Output1. In: *Whippedcreamsounds* [online]. whippedcreamsounds, 2023 [cit. 2023-05-14]. Dostupné z: [https://cdn.shortpixel.ai/spai/q\\_lossless+w\\_372+h\\_148+to\\_webp+ret\\_img/https://www.whippedcreamsounds.com/wp-content/uploads/2022/12/cap1.png](https://cdn.shortpixel.ai/spai/q_lossless+w_372+h_148+to_webp+ret_img/https://www.whippedcreamsounds.com/wp-content/uploads/2022/12/cap1.png)
- [26] Output2. In: *Whippedcreamsounds* [online]. whippedcreamsounds, 2023 [cit. 2023-05-14]. Dostupné z: [https://cdn.shortpixel.ai/spai/q\\_lossless+w\\_372+h\\_148+to\\_webp+ret\\_img/https://www.whippedcreamsounds.com/wp-content/uploads/2022/12/cap2.png](https://cdn.shortpixel.ai/spai/q_lossless+w_372+h_148+to_webp+ret_img/https://www.whippedcreamsounds.com/wp-content/uploads/2022/12/cap2.png)
- [27] VFD PWM. In: *Select Electrical* [online]. Mayerthorpe: Select Electrical, 2021 [cit. 2023-05-14]. Dostupné z: <https://selectelectricalent.com/wp-content/uploads/2021/05/Variial-Frequency-Drive-VFD-PWM-Output.png>
- [28] PID controller. In: *PLCynergy* [online]. PLCynergy, 2021 [cit. 2023-05-14]. Dostupné z: <https://plcynergy.com/wp-content/uploads/2021/01/PID-controller.jpg>



# 11 SEZNAM ZKRATEK, SYMBOLŮ, OBRÁZKŮ A TABULEK

## 11.1 Seznam zkratek

BCS	Base Coordinate Systém
CNC	Computer Numerical Control (Počítačové číslicové řízení)
CPU	Central Processing Unit
DCS	Decentralised Control System
FB	Function block (Funkční blok)
FBD	Function Blok Diagram
FC	Function (Funkce)
HMI	Human Machine Interface
IL	Instruction List
IO	Inputs/Outputs
LD, LAD	Ladder Diagram
LIN	Linear
OLP	Off-line programming (Off-line programování)
PLC	Programmable Logic Controller
PS	Process Simulate
PTP	Point-to-Point
PWM	Pulse Wave Modulation (Pulní vlnová modulace)
R	Rotace
RS	RobotStudio
SCADA	Supervisor Control And Data Acquisition
SCL	Structured Text (Strukturovaný text)
SFC	Sequential Function
SS	Souřadný systém
T	Translace
TCP	Tool Center Point
TIA	Totally Integrated Automation
TO	Technology Objects
UCS	User Coordinate Systém
VSC	Visual Studio Code
WCS	World Coordinate System
WOF	World Object Frame

## 11.2 Seznam tabulek

Tabulka 1 – Detaily robotu KUKA.....	42
Tabulka 2 – Tabulka stavů „klapek“ v jednotlivých pozicích. 1 – OPEN, 0 - CLOSE	63
Tabulka 3 – Popis fungování podprogramu (obr. 82).....	68
Tabulka 4 – Popis fungování podprogramu (obr.90).....	73
Tabulka 5 – Princip fungování programu (příloha č. 3).....	89
Tabulka 6 – Princip fungování programu (obr. 124).....	93
Tabulka 7 – Princip fungování programu (obr. 130).....	97

## 11.3 Seznam obrázků

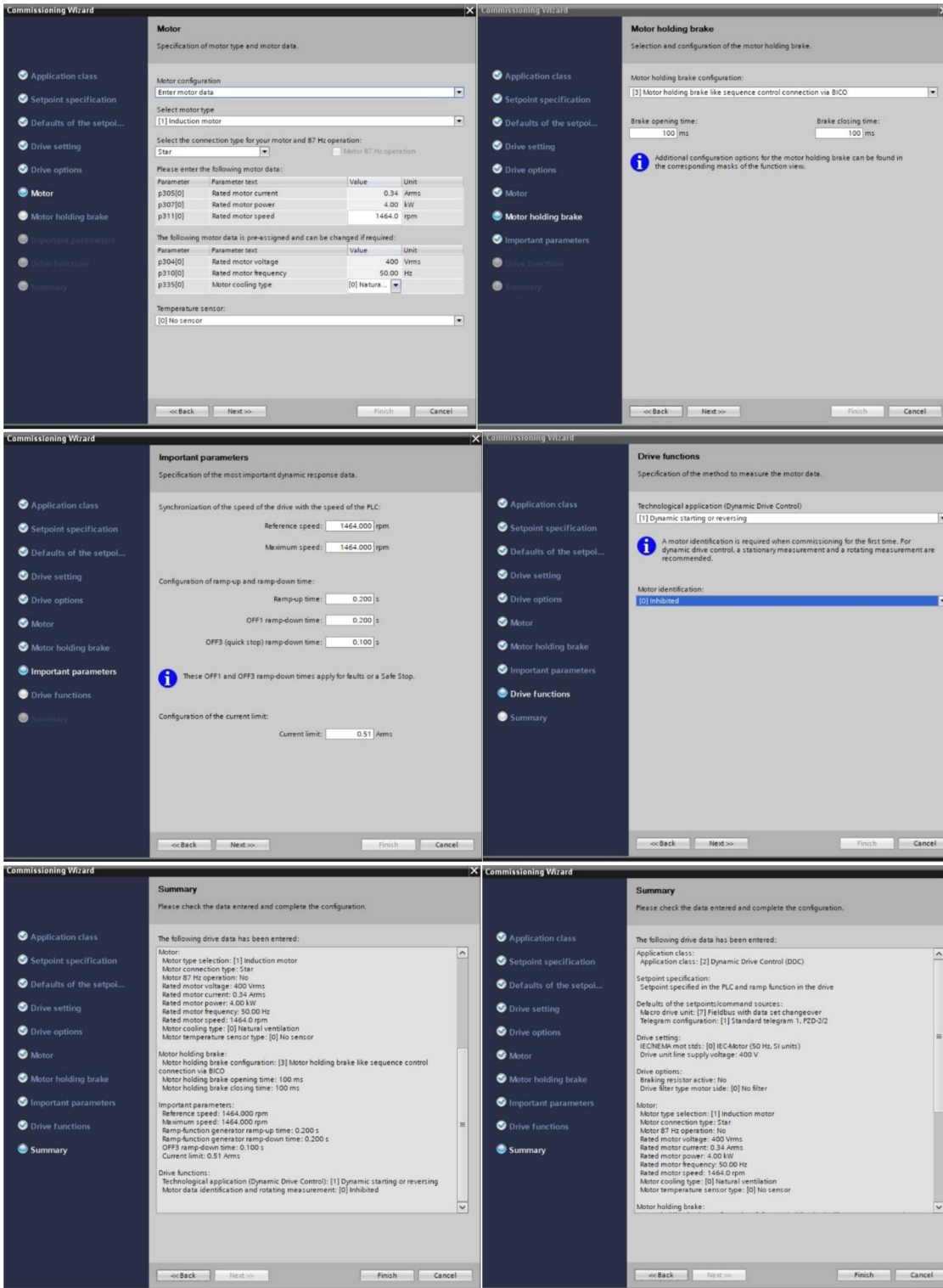
Obr. 1 - Moderní vs. konvenční proces návrhu pracoviště [1].....	19
Obr. 2 – Princip bodového svařování [2].....	20
Obr. 3 – Detail reálného bodového svařování [3].....	20
Obr. 4 – Robot osazen s dual-tool [4].....	21
Obr. 5 – a) Karteziánský robot, b) SCARA robot [5] [6].....	21
Obr. 6 – a) Šestiosý robot, b) Dvouramenný robot [7] [8].....	22
Obr. 7 – Delta robot [9].....	22
Obr. 8 – a) Vysoká přesnost, b) Vysoká opakovatelnost [10] [11].....	23
Obr. 9 – Souřadné systémy robotu [12].....	24
Obr. 10 – X-bodové metody zaměření TCP [13].....	24
Obr. 11 – X-bodová metoda v programu ABB RobotStudio [14].....	24
Obr. 12 – a) Princip laserového zam. TCP, b) Reálný stojan zaměření TCP [15] [16]	25
Obr. 13 – Definování bodů a souřadných systémů robotu ABB.....	25
Obr. 14 – Typy pohybových instrukcí robotu [17].....	25
Obr. 15 – Teach pendant robotu KUKA [18].....	26
Obr. 16 – Definování bodu v robotickém kontroléru ABB.....	26
Obr. 17 – a) MAIN program robotu, b) Volaný podprogram.....	27
Obr. 18 – Interface programu Process Simulate.....	28
Obr. 19 - Ukázka fungování Logic Resource editoru v PS.....	28
Obr. 20 – Princip cyklů PLC [19].....	29
Obr. 21 – Kompaktní PLC a) LOGO! firmy SIEMENS, b) firmy E.T.N [20] [21].....	30
Obr. 22 – Modulární PLC a) bez roz. modulů, b) s moduly na DIN liště [22] [23].....	30
Obr. 23 – Ukázka větve vytvořené v jazyku Ladder Diagram.....	31
Obr. 24 - Ukázka větve vytvořené v jazyku Structured Text (Strukturovaný text).....	32
Obr. 25 – Základní vnitřní struktura frekvenčního měniče [24].....	34
Obr. 26 – Vstupní a výstupní napětí po usměrnění a vyhlazení [25] [26].....	35
Obr. 27 – Ukázka fungování výstupního střídače frekvenčního měniče [27].....	35
Obr. 28 – Standardní struktura PID regulátoru [28].....	35
Obr. 29 – Zadané pracoviště.....	37
Obr. 30 – Detail zadání pracoviště.....	37
Obr. 31 – Obecný blokový diagram buňky.....	38
Obr. 32 – Blokový diagram robotické buňky.....	38
Obr. 33 – Blokový diagram rotačního stolu.....	39
Obr. 34 – Rozbor nebezpečných prostorů pracoviště.....	39
Obr. 35 – Detail pracoviště.....	40
Obr. 36 - Detail na první část robotické buňky bez bezpečnostních prvků.....	41

Obr. 37 - Detail na druhou část robotické buňky bez bezpečnostních prvků.....	41
Obr. 38 – a) Pracovní zóny robotu, b) Sférický dosah robotu.....	42
Obr. 39 – Svařovací kleště typu C .....	43
Obr. 40 – a) Stojan pro automatické zaměření TCP, b) Frézovací stojan.....	43
Obr. 41 – a) Operace frézování, b) Operace TCP.....	44
Obr. 42 – Popis rotačního stolu .....	44
Obr. 43 – Pohon rotačního stolu .....	45
Obr. 44 – Kinematika rotačního stolu.....	45
Obr. 45 – Upínací prvky rotačního stolu polotovaru typu šachta.....	46
Obr. 46 - Upínací prvky rotačního stolu polotovaru typu šachta .....	46
Obr. 47 – Detail upínacích prvků rotačního stolu polotovaru typu šachta.....	47
Obr. 48 – Detail na upínací prvky rotačního stolu polotovaru typu blatník.....	47
Obr. 49 – Strom objektů projektu .....	48
Obr. 50 – Kanálová struktura pro kabeláž po celém pracovišti.....	48
Obr. 51 – Detail na pracoviště včetně všech bezpečnostních prvků (oplocení).....	49
Obr. 52 – Strom operací .....	49
Obr. 53 – Operace typu polotovaru blatník.....	50
Obr. 54 - Operace typu polotovaru šachta.....	51
Obr. 55 – Kontrolní operace.....	51
Obr. 56 – Simulační operace .....	52
Obr. 57 – a) Svářený polotovar typu Blatník, b) Detail na svářecí body .....	53
Obr. 58 - a) Svářený polotovar typu šachta, b) Pohled z opačné strany .....	53
Obr. 59 - a) Detail na svářecí body polotovaru typu šachta, b) Detail 2.....	53
Obr. 60 – Tabulka původní signálové struktury v PS .....	54
Obr. 61 – Laserový skener pracovního prostoru obsluhy .....	54
Obr. 62 – Umístění indukčních senzorů na rotačním stole polotovaru šachta .....	55
Obr. 63 - Umístění indukčních senzorů na rotačním stole polotovaru blatník.....	55
Obr. 64 – Laserové skenery koncových poloh rotačního stolu (žlutě) .....	56
Obr. 65 - Laserové skenery koncových poloh rotačního stolu (žlutě).....	56
Obr. 66 – Rozvody pneumatiky a vzdušníky .....	56
Obr. 67- Bezpečnostní dveře pracoviště .....	57
Obr. 68 – a) Bezpečnostní dveře hlavní, b) Dveře boční .....	57
Obr. 69 - Paletizace.....	58
Obr. 70 – a) Využití PLC, b) Využití frekvenční měnič G120.....	60
Obr. 71 – a) Bezpečnostní dveře hlavní, b) Signálová struktura .....	61
Obr. 72 - a) Bezpečnostní dveře boční, b) Signálová struktura.....	62
Obr. 73 – Rotační stůl – signálová struktura .....	62
Obr. 74 – Up. Prvky v pozici a) CLOSED, b) OPENED, c) WORK .....	63
Obr. 75 – a) Prvky řídicí dodatečnou logiku signálů, b) Ukázka převodu.....	64
Obr. 76 – Program pro převod a) BITS → WORD, b) WORD → BITS .....	64
Obr. 77 – Signály indukčních senzorů v PS .....	65
Obr. 78 – Definování a) bodů operací včetně OLP, b) SS up. prostoru.....	66
Obr. 79 – Využití OLP příkazy .....	67
Obr. 80 – Spouštěcí signály robotických operací .....	67
Obr. 81 – Struktura podprogramu simulujícího chování robotu .....	68
Obr. 82 – Celková signálová struktura robotů v PS.....	68
Obr. 83 – Manuální „silové“ zapisování signálů .....	69

Obr. 84 – a) Program PLCSIM, b) Propojení PS a TIA.....	69
Obr. 85 – Kontrola připojení mezi PS a TIA.....	69
Obr. 86 - Vývojový diagram fungování pracoviště .....	70
Obr. 87 – Konfigurace PLC se všemi využitými moduly .....	71
Obr. 88 – Náhled původní struktury tabulky z PS .....	72
Obr. 89 – Náhled výsledné tabulky.....	72
Obr. 90 – Struktura podprogramu určeného k úpravě tabulky signálů z PS .....	72
Obr. 91 – a) Rozdělení signálů do kategorií, b) Náhled obsahu každé kategorie.....	73
Obr. 92 – Centrální kontrolní databáze všech signálů v TIA .....	74
Obr. 93 – Funkce přiřazující hodnoty vstupních signálů .....	74
Obr. 94 - Funkce přiřazující hodnoty výstupních signálů .....	74
Obr. 95 – Podprogram pro převod BITS → WORD .....	75
Obr. 96 - Podprogram pro převod WORD → BITS.....	76
Obr. 97 – Síťová struktura elektrických prvků pracoviště, včetně IP adres.....	76
Obr. 98 – Nastavení komunikačního protokolu mezi VFD a PLC .....	77
Obr. 99 – a) Struktura řídicího programu, b) + c) čtení kontrolních hodnot.....	78
Obr. 100 – Volání řídicího programu pohonů v MAIN .....	79
Obr. 101 – Program logiky operátora – první síť .....	80
Obr. 102 - Program logiky operátora – druhá síť – první část .....	80
Obr. 103 - Program logiky operátora – druhá síť – druhá část.....	80
Obr. 104 - Program logiky operátora – síť 3-6.....	81
Obr. 105 - Program logiky operátora – poslední síť.....	81
Obr. 106 - Program logiky operátora polotovaru blatník – první síť.....	82
Obr. 107 - Program logiky operátora polotovaru blatník – druhá síť – první část.....	82
Obr. 108 - Program logiky operátora polotovaru blatník – druhá síť – druhá část.....	82
Obr. 109 - Program logiky operátora polotovaru blatník.....	83
Obr. 110 - Program logiky robotu polotovaru šachta – první síť .....	84
Obr. 111 - Program logiky robotu polotovaru šachta – druhá síť – první část.....	84
Obr. 112 - Program logiky robotu polotovaru šachta – druhá síť – druhá část.....	84
Obr. 113 - Program logiky robotu polotovaru šachta – třetí síť .....	85
Obr. 114 - Program logiky robotu polotovaru šachta – osmá síť .....	85
Obr. 115 - Program logiky robotu polotovaru šachta – devátá síť .....	86
Obr. 116 - Program logiky robotu pol. šachta a) zapnutí časovače b) uložení.....	86
Obr. 117 - Program logiky robotu polotovaru blatník .....	87
Obr. 118 – Hlavní program inicializace .....	88
Obr. 119 – Simulace zpětné vazby robotů o jejich stavu .....	89
Obr. 120 – a) Hlavní program CONTROL, b) Detail na strukturu stavu AUTO.....	90
Obr. 121 – Struktura program MAIN.....	90
Obr. 122 – HMI – INICIALIZACE .....	91
Obr. 123 – HMI – Automat_ON.....	92
Obr. 124 – HMI – Program pro automatické ukládání časů cyklů do souboru.....	93
Obr. 125 – Ukázka ukládaných hodnot (z programu na obr. 125) .....	93
Obr. 126 – HMI – MANUAL a) Klapky, b) Robot, c) Motory, d) Ostatní.....	94
Obr. 127 – Signálová struktura robotů v TIA.....	95
Obr. 128 – Robotický inventář a) Robot_S, b) Robot_B .....	96
Obr. 129 – Základní konfigurace bodů robotických operací v PS.....	96
Obr. 130 – Předpřipravený MAIN program robotů .....	97







## 2) Konfigurace TO (Technology Objects)

**Basic parameters**  
 Name: PscKonfig163\_0  
 PLC → Technology object → Drive → Motor  
 Axis type:  Rotary axis  
                    Linear  
                    Rotary  
 Data of measure:  
 Limit of measure for position:   
 Limit of measure for velocity:   
 Limit of measure for torque:   
 Module:  
 Module start value:   
 Module length:   
 Simulation:  Activate simulation

**Drive**  
 Drive type: PRO Drive  
 Data connection: Drive  
 Encoder data:  
 Encoder: Encoder  
 Encoder data: Encoder data  
 Device configuration  
 Drive configuration

**Encoder**  
 Data connection: Encoder  
 Encoder: PLC\_1.TM.Count.2x24V\_1.Chan...  
 Encoder type: Incremental  
 Device configuration

**Data exchange with the drive**  
 Drive telegram: Standard telegram 1  
 Automatically apply drive values during configuration (offline)   
 Automatically apply drive values at runtime (online)   
 Reference speed: 1484.0  Limit  
 Maximum speed: 1100.0  Limit  
 Additional data:  
 Torque data

**Data exchange with encoder**  
 Encoder telegram: Standard telegram 83  
 Automatically apply encoder values during configuration (offline)   
 Automatically apply encoder values during runtime (online)   
 Measuring system: Rotary  
 Increments per revolution: 1  
 Fine resolution:  
 Bits in Gx\_NCT1: 2 bit

### 3) Podprogram INICIALIZACE

```

1  #re_start(CLK:= #b_start);
2  #re_reset(CLK := #b_reset);
3  IF #re_start.Q THEN
4      #b_Robot_MotorsOn := 1;
5      #b_Robot_MotorsOff := 0;
6      #n_stateNum := #STATE_OV_ERORY;
7
8  END_IF;
9
10 CASE #n_stateNum OF
11     #STATE_OV_ERORY:
12         IF #n_Error = 0 THEN
13             #n_stateNum := #STATE_OVERENI;
14         ELSE
15             #n_Error := 11;
16             #b_Error_ResetError := 1;
17             #n_stateNum := #STATE_ERROR;
18         END_IF;
19
20     #STATE_OVERENI:
21         IF #b_Robot_RunChainOk THEN
22             #n_stateNum := #STATE_INIT;
23         ELSE
24             #b_Error_runchain := 1;
25             #n_Error := 1;
26             #n_stateNum := #STATE_ERROR;
27         END_IF;
28

```

1)

```

30     #STATE_ERROR:
31
32     IF #re_reset.Q AND #b_ErrorAknow THEN
33         #n_Error := 0;
34         #b_Error_ResetError := 0;
35         #b_Error_NotAutoOn := 0;
36         #b_Error_runchain := 0;
37         #b_Error_NotCycleOn := 0;
38         #b_RobotReadyToWork := 0;
39         #n_stateNum := #STATE_OV_ERORY;
40         #b_ErrorAknow := 0;
41     END_IF;
42
43
44     #STATE_INIT:
45     IF #b_Robot_AutoOn THEN
46         #n_stateNum := #STATE_MOTORS_ON;
47     ELSE
48         #n_Error := 2;
49         #b_Error_NotAutoOn := 1;
50         #n_stateNum := #STATE_ERROR;
51     END_IF;
52
53

```

2)

```

54     #STATE_MOTORS_ON:
55     #t_timer(IN := 1,
56         PT := T#2s);
57     IF NOT #t_timer.Q THEN
58
59         IF #b_Robot_MotorsOnState = 1 THEN
60             #n_stateNum := #STATE_START_CYCLE;
61         ELSE
62             #b_Robot_MotorsOff := 0;
63             #b_Robot_MotorsOn := 0;
64             #b_Robot_MotorsOn := 1;
65             #n_stateNum := #STATE_START_CYCLE;
66         END_IF;
67     ELSE
68         #t_timer.IN := 0;
69         #n_Error := 3;
70         #b_Error_MotorsOn_offlimit := 1;
71         #b_Robot_MotorsOn := 1;
72         #b_Robot_MotorsOff := 0;
73         #n_stateNum := #STATE_ERROR;
74     END_IF;
75
76

```

3)

```

77     #STATE_START_CYCLE:
78     #t_timer(IN := 1,
79         PT := T#2s);
80     IF NOT #t_timer.Q THEN
81
82         IF #b_Robot_Cycle_ON THEN
83             #n_stateNum := #STATE_ROBOT_READY;
84             #t_timer.IN := 0;
85         ELSE
86             #n_Error := 4;
87             #b_Error_NotCycleOn := 1;
88             #b_Robot_MotorsOn := 0;
89             #b_Robot_MotorsOff := 1;
90             #n_stateNum := #STATE_ERROR;
91             #t_timer.IN := 0;
92         END_IF;
93     END_IF;
94
95

```

4)

```

96     #STATE_ROBOT_READY:
97     IF #b_Robot_RunChainOk AND
98         #b_Robot_AutoOn AND
99         #b_Robot_Cycle_ON AND
100        #b_Robot_MotorsOnState THEN
101
102        #b_RobotReadyToWork := 1;
103        IF #re_reset.Q THEN
104
105            #b_Robot_MotorsOn := 0;
106            #b_Robot_MotorsOff:= 1;
107            #n_stateNum := 1000;
108            #b_RobotReadyToWork := 0;
109        END_IF;
110    ELSE
111        #n_Error := 5;
112        #n_stateNum := #STATE_ERROR;
113    ;
114    END_IF;
115
116
117 END_CASE;

```

5)