VŠB TECHNICKÁ UNIVERZITA OSTRAVA | FAKULTA ELEKTROTECHNIKY A INFORMATIKY

# Vytvoření návrhu a simulace technologického procesu za pomocí využití prostředí B&R Scene Viewer

Creating a Design and Simulation of the Technological Process by using the B&R Scene Viewer Environment

**David Tomiczek**

Diplomová práce

Vedoucí práce: Ing. Martin Mikolajek

Ostrava, 2021

**Poděkování**

Tímto bych rád poděkoval mému vedoucímu práce panu Ing. Maritnu Mikolajkovi za odbornou pomoc při tvorbě mé závěrečné práce. Dále bych rád poděkoval technikům z oddělení podpory pro Bernecker & Rainer ČR za ochotu a vstřícnost při řešení technických potíží.

**Abstrakt**

Diplomová práce řeší úvod do problematiky programovatelných logických automatů a rozbor možností pro virtuální uvedení do provozu s konceptem digitálního dvojčete. Teoretická část obsahuje obecné informace o způsobech virtuálního uvedení do provozu spolu s technickými možnostmi hojně využívaných softwarů pro tyto účely. Cílem praktické části je vytvořit digitální model stávajícího technologického procesu a zhodnotit možnosti simulace v prostředí Scene Viewer spolu s vývojovým prostředím Automation Studio. Pro ověření funkčnosti simulace je vytvořen řídicí program.

**Klíčová slova**

Programovatelný logický automat (PLC); digitální dvojče; simulační model; virtuální uvedení do provozu; manipulační linka; Scene Viewer; Automation Studio; B&R X20 systém; strukturovaný text;

**Abstract**

The thesis addresses the introduction to the issue of programmable logic controllers and the analysis of the possibilities for virtual commissioning with the concept of a digital twin. The theoretical section contains general information on the modalities of virtual commissioning, together with the technical capabilities of widely used software for these purposes. The objective of the practical part is to create a digital model of the existing technological process and to evaluate simulation capabilities in the Scene Viewer environment together with the Automation Studio development environment. A control program is created to verify the functionality of the simulation.

**Keywords**

Programmable logic controller (PLC); digital twin; simulation model; virtual commissioning; handling line; Scene Viewer; Automation studio; B&R X20 systém; structured text;

**Obsah**

## Seznam ilustrací

## Seznam tabulek

## Úvod

Diplomová práce pojednává o základní problematice programovatelných logických automatů, jejich rozdělení, konstrukčních řešeních a také o programovacích jazycích, které se se používají pro implementaci řídicího kódu. Informace o této problematice jsou uvedeny stručně a okrajově, avšak dostatečně k pochopení principu činnosti programovatelných logických automatů a jejich využití na poli automatizace. Hlavním tématem této práce je virtuální uvedení do provozu, kde věnuji pozornost podrobnějšímu rozboru možností a využití těchto novodobých technologií v praxi. Cílem je podat všeobecný přehled a povědomí, k čemu virtuální uvedení do provozu slouží, jaké jsou jeho výhody, proč se využívá a jakým způsobem se virtualizuje fyzický proces. S tím také úzce souvisí pojem digitální dvojče, jenž se využívá společně v kombinaci s virtuálním uváděním do provozu. Uvádím přehled několika hojně využívaných softwarů právě pro tyto aplikace s rozborem jejich možností, výhod a vhodným zaměřením. V praktické části se zabývám samotným návrhem konceptu digitálního dvojčete v prostředí Scene Viewer 4 spolu s vývojovým prostředím Automation Studio 4.7.2.98 a X20 systémem od společnosti Bernecker & Rainer. Koncept digitálního dvojčete vychází ze stávajícího technologického procesu a to manipulační linky, která slouží pro výukové účely. Popisuji problematiku importu CAD souborů do prostředí Scene Viewer a jejich následné použití při vytváření simulačního modelu. V počáteční fázi se věnuji teoretickému rozboru stávající fyzické linky a návrhu simulačních procesů. Popisuji jednotlivé akční členy a jejich funkci v technologickém procesu. Na to navazuji návrhem a implementací simulačních funkčních bloků spolu s rozšířeným uživatelským rozhraním, které navíc otvírá nové možnosti linky pomocí nastavení dalších parametrů simulačních funkčních bloků. Tím jsou myšleny zejména poruchové a havarijní stavy technologie, které lze v simulaci do určité míry testovat, simulovat a vyhodnocovat. Pro ověření funkčnosti simulace jsem implementoval řídicí program, který třídí puky podle jejich vlastností. Celý řídicí program je naprogramován v jazyce strukturovaný text.

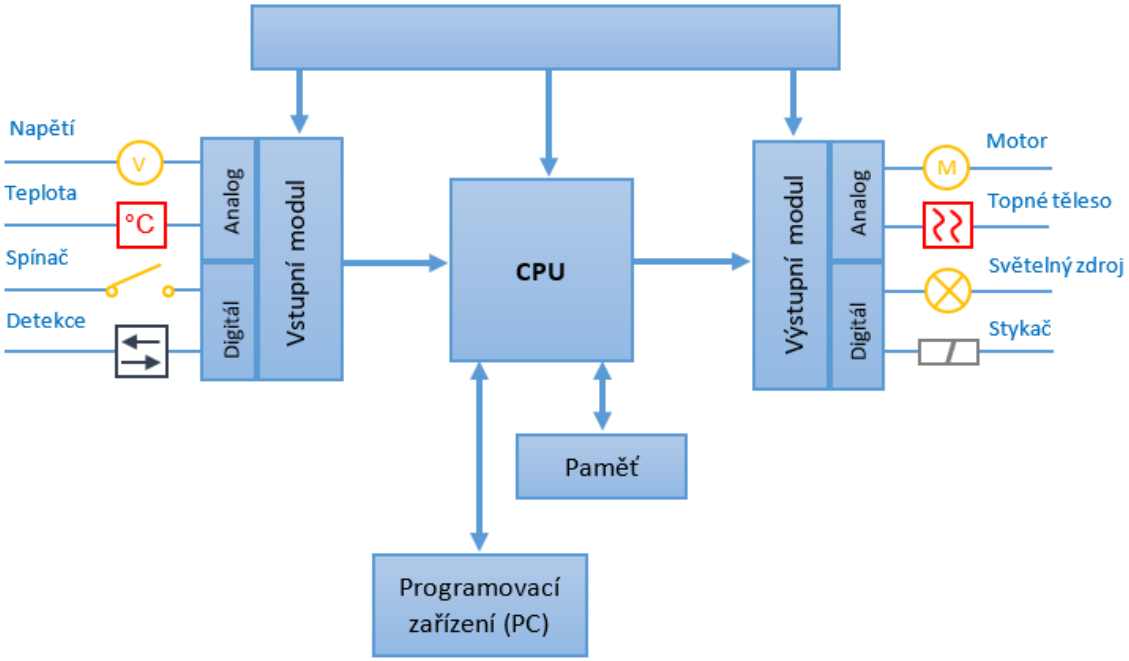
# 1. Úvod do problematiky programovatelných automatů

V současné době jsou automatizační procesy zpracovávány mnoha způsoby. Významnou část tvoří programovatelné automaty, které byly původně vyvinuty jako náhrada reléové logiky. Z anglického označení Programmable Logic Controller je odvozena velice často používaná zkratka PLC. V německé literatuře se můžete setkat s méně častým označením SPS (Speicherprogrammierbare Steuerung), v češtině pak PA jako programovatelný automat. Díky vývoje a zvyšování požadavků na řídicí systém převzal programovatelný automat velkou roli hlavně v regulačních procesech, monitorování, měření a sběru analogových dat. Používá se spíše v robustnějších průmyslových aplikacích, přičemž se považuje za velice spolehlivou platformu a s tím také souvisí cena, která dosahuje i několik stovek tisíc korun. Příklady programovatelných automatů různých výrobců je zobrazen na Obr. 1.


Obr. 1 Programovatelné automaty různých výrobců [2][3][4][5]

## 1.1. Technické parametry programovatelných automatů

Vnitřní uspořádání je téměř vždy stejné a každý programovatelný automat funguje na bázi procesoru. Dále obsahuje systémovou paměť, uživatelskou paměť, vstupní a výstupní jednotky, komunikační moduly a sběrnice. Jednoduchá struktura programovatelného automatu je znázorněna na Obr. 2.




Obr. 2 Blokové schéma programovatelného automatu

### 1.1.1. Centrální procesorová jednotka

Centrální procesorová jednotka (CPU) je jádrem celého programovatelného automatu a určuje jeho výkon. Může být jedno procesorová nebo více procesorová. U víceprocesorových jednotek mohou být použity matematické koprocesory, které jsou vhodným řešením zejména pro aplikace vyžadující složitější matematické operace. Dále vstupní a výstupní koprocesory, komunikační procesory, které disponují pokročilými funkcemi např. využívá dynamickou mezipaměť pro extrémně rychlou odezvu HMI systémů bez závislosti na skenování CPU. Pro velmi rychlé provádění bitových instrukcí je použit speciální bitový koprocesor (z angličtiny „fast operating bit“). Důležitým parametrem CPU je operační rychlost, která vyplývá z doby cyklu. Doba cyklu je časový interval (cyklus), za který PLC provede všechny potřebné operace k vykonání programu a vrátí se zpět do stejného bodu činnosti. Tento cyklus se skládá ze tří kroků:

1) Čas čtení vstupů, tedy měření analogových a digitálních dat na vstupních hardwarových modulech
2) Čas k vykonání programu. Vykonává se sekvenčně počínaje prvním řádkem.
3) Čas k zápisu na výstup, tedy nastavení analogových a digitálních hodnot na výstupních hardwarových modulech

Doba cyklu bývá obvykle v řádech jednotek až desítek milisekund, u výkonnějších CPU je možné se dostat na dobu mikrosekund. Zde tedy platí pravidlo čím kratší doba cyklu, tím výkonnější a zároveň dražší PLC. [13]

### 1.1.2. Paměť

Do paměťového prostoru se ukládají data a program PLC. Proces vkládání nových informací a dat do paměti se nazývá zápis (z angličtiny „write“). Proces načítání informací a dat z paměti se nazývá čtení (z angličtiny „read“). Běžně používané paměti jsou PROM (Programmable Read-Only Memory) nebo EEPROM (Electrically Erasable Programmable Read-Only Memory) a RAM (Random Access Memory). Paměť ROM (PROM, EEPROM) je uživatelská a slouží pouze ke čtení, nikoliv k zápisu a je použita k ukládání programů a dat, které by neměly být změněny a není závislá na elektrickém napájení (je tzv. nevolatilní). V modernějších přístrojích se namísto PROM a EEPROM paměti používá typ FLASH. Paměť RAM je systémová a slouží jak ke čtení, tak k zápisu a je závislá na elektrické napájení (je tzv. volatilní). Jsou v ní uložena programová a konfigurační data, která jsou zodpovědná za uložení hodnot interních proměnných, interních bitů a slov, časovače, čítače, posuvné registry apod.. U některých PLC se lze setkat s pamětí FRAM (Ferroelectric Random Access Memory), který funguje principiálně jako RAM s tím rozdílem, že je nevolatilní, tedy data jsou zde uchovány i bez elektrického napájení. Výhodou je rychlost zapisování (až 1400 kB/s) a velmi nízká spotřeba. K napájení využívají pouze 1,5 V a v porovnání s paměti FLASH, spotřebuje FRAM při zápisu dat rychlostí 12kB/s až 250x méně energie. [16][17]

### 1.1.3. Analogové vstupní a výstupní jednotky

Analogové vstupy a výstupy (zkráceně AO – analog output, AI – analog input) slouží k měření analogových (spojitých) technologických signálů, což znamená, že se jejich hodnota může měnit spojitě v čase v určitém rozsahu. V případě vstupů se jedná o měření tlaku, teploty, polohy, napětí, proudu apod., v případě výstupů se nejčastěji používají k ovládání akčních členů nebo regulace. Chceme-li s analogovým signálem po změření dále pracovat v programu, musí být převeden do digitální podoby, neboť analogový signál dosahuje nekonečně mnoho hodnot. Proto jsou analogové vstupy připojeny na A/D převodník, v případě analogového výstupu D/A převodník. Základní parametr převodníku určuje tzv. rozlišení převodníku, což je bitová šířka převáděné hodnoty. Rozlišení převodníku určuje kolika bity je reprezentován analogový signál. Důležitá je znalost rozsahu, protože se celý signál převádí

(kvantifikuje) na stejně velké hodnoty v podobě bitů. Běžně se používají osmi, dvanácti, šestnácti bitové převodníky.
Příklad:
Je-li změřen analogový signál dvanácti bitovým převodníkem, jeho měřící rozsah nabývá $2^{12}$ hodnot, tedy 4096. Pokud je vstupní signál v rozsahu 0 až 24 V, potom je nejmenší rozlišitelnou hodnotou 0,0059 V. Při použití šestnácti bitového převodníku je nejmenší rozlišitelná hodnota 0,00037 V, protože měřící rozsah nabývá $2^{16}$ hodnot, tedy 65 536. Tento výpočet je dán vzorcem (1):

$$(1)\quad AD_{min} = \frac{A_{max}}{2^n}$$

kde

| | |
|---|---|
| $AD_{min}$ | je minimální rozlišitelná hodnota převodníku (rozlišení) |
| $A_{max}$ | je maximální rozsah analogového signálu |
| n | je počet bitů převodníku |


Obr. 3 PLC s vstupními a výstupními moduly od firmy B&R [7]

V praxi bávají k PLC připojeny přídavné vstupní i výstupní moduly s konkrétními vlastnostmi. Mimo klasických vstupů může modul obsahovat např. vstup přímo pro teplotní senzor PT1000. Kvalitu převodu určují i jiné vlivy, které jsou však předmětem podrobnějšího zkoumání technické dokumentace daného modulu. Na Obr. 3 je zobrazen programovatelný automat s vstupními a výstupními moduly firmy B&R.

### 1.1.4. Digitální vstupní a výstupní jednotky

Digitální vstupy a výstupy (zkráceně DI – digital input, DO – digital output) pracují na velmi jednoduchém principu. Signál může nabývat pouze dvou hodnot, označovaných jako logická 0 a logická 1, nejčastěji 0 V a 24 V. Časté označení je také „true“ nebo „false“. Tyto diskrétní data jsou typické pro procesní spínače, tlačítkové spínače, koncové spínače, bezdotykové spínače, senzory přítomnosti objektu, indikace požadované hodnoty apod.. Vstupní i výstupní terminály mohou obsahovat vysokorychlostní čítače (z angličtiny „High-Speed Counter) pro extrémně rychle čtení nebo zápis digitálních hodnot v řádech desítek až stovek kHz. Moduly se běžně vyrábějí s 4, 8, 12, 16 nebo 32 kanály.

### 1.1.5. Komunikační jednotky

V drtivé většině případů se ke komunikaci používá ethernet. Slouží jak ke komunikaci mezi PLC a okolním světem, tak mezi ostatními PLC, moduly, operátorskými panely, HMI, s nadřazenými, souřadnými i podřazenými subsystémy či vzdálenými vstupně výstupními moduly. Ostatní časté komunikační standardy a protokoly jsou:

- EtherCAT
- PROFINET
- PROFIBUS DP
- CANopen
- DeviceNet
- Modbus TCP
- Modbus RTU
- RCOM
- Serial

### 1.1.6. Napájení

Napájecí zdroj není úplně součást programovatelného automatu, ale je nezbytný pro jeho chod. Napájecí zdroj převádí síťové střídavé napětí, obvykle 120 V nebo 240 V, na napětí stejnosměrné, běžně 24 V. Napájení neslouží pouze pro PLC, ale i pro ostatní komponenty v topologii, s čímž souvisí dimenzování výkonu napájecího zdroje.

## 1.2. Rozdělení programovatelných automatů

Výrobců programovatelných automatů je na trhu poměrně dost a nabídka se stále rozšiřuje a vývoj pokračuje kupředu. S tím samozřejmě přichází i vysoká nabídka a různorodost. Programovatelné automaty lze rozdělit podle různých hledisek. Princip činnosti a programování je většinou stejný nebo dosti podobný. Výrazné odlišnosti jsou však v konstrukčním pojetím a uživatelské koncepci.

## 1.3. Rozdělení podle konstrukce

**Kompaktní programovatelný automat (KPA)** - též integrovaný programovatelný automat je postaven z několika modulů v jednom pouzdře. To v praxi znamená, že o vstupně výstupní funkce rozhoduje výrobce, nikoliv uživatel. Mají pevně danou konfiguraci a montují se přímo do výrobku nebo na DIN lištu v rozvaděči. V poslední době je však snaha o to, aby kompaktní programovatelné automaty umožnily připojit další vstupně výstupní moduly a dosáhly tak určitého stupně modularity a možnosti přizpůsobit svou konfiguraci potřebám konkrétní aplikace.
**Modulární programovatelný automat (MPA)** - jak již z názvu vyplývá, jedná se o modularita celého systému. Na rozdíl od KPA umožňují MPA vícenásobné rozšíření dosavadního systému a jsou vhodné pro automatizační aplikace středního až velkého rozsahu. Hlavní výhodou škálovatelnost, tím se rozumí rozšíření o vstupně výstupních moduly, které jsou na sobě zcela nezávislé, což také umožňuje snazší detekci chyb. Pohodlně lze komponenty vyměnit či obměnit bez nutnosti obnovení jiných částí. Stejným způsobem lze také rozšířit paměť. [1]

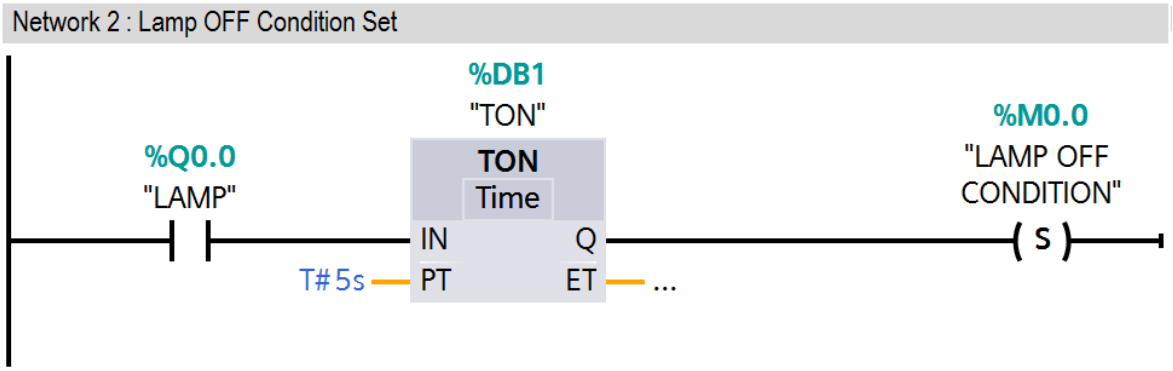
## 1.4. Rozdělení programovatelných automatů podle velikosti

**Mikro** programovatelný automat je nejlevnější a nejmenší kompaktní zapouzdřený systém, který nabízí uživateli pevně danou sestavu vstupů a výstupů. Obvykle bývají pouze binární (digitální). Mikro PLC se svou cenou v řádech několika tisíců korun slouží k méně náročným aplikacím, jako jsou jednoduché logické operace strojů a mechanismů, kde bylo původně použito reléového řízení.
**Malý** programovatelný automat je schopen zpracovat desítky vstupních a výstupních signálů. Je vhodný pro aplikace malého rozsahu a většinou poskytuje kompletní řadu funkcí pro logické a aritmetické operace. Konstrukční řešení může být jak kompaktní, tak modulární.
**Střední** programovatelný automat je schopen zpracovávat stovky vstupních a výstupních signálů. Jsou vhodné pro aplikace středního rozsahu. Konstrukční provedení bývá již většinou modulární.
**Velký** programovatelný automat je nejvyšší třída PLC. Je schopen zpracovávat stovky i tisíce vstupních a výstupních signálů. Jejich použití se uplatní v těch nejnáročnějších aplikacích a mívají výhradně modulární konstrukční řešení. [1]

## 1.5. Programovací jazyky

Programovatelné automaty mají svou specifickou skupinu programovacích jazyků. Od ostatních jazyků se liší svou jednoduchostí, což byl vlastně účel vytvoření těchto jazyků. Existují čtyři jazyky, které jsou považovány za standardní jazyky pro použití v PLC, v souladu s normou IEC 61131–3, která specifikuje jejich syntaxi a sémantiku.

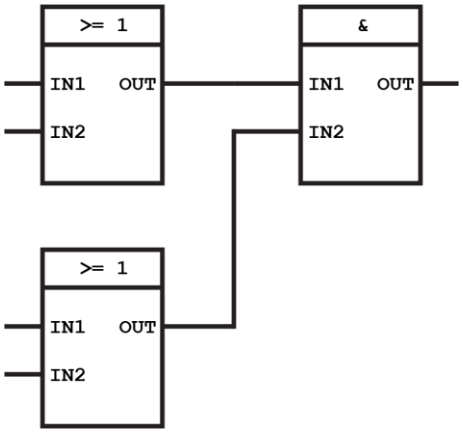
**Ladder Diagram (LD)**
Ladder Diagram je nejstarší jazyk programovatelných automatů. Tento grafický programovací jazyk byl vymodelován z reléové logiky, aby umožnil technikům a elektrikářům hladký přechod na programování. Tento programovací jazyk připomíná elektrické zapojení v reálném světě. Původně obsahoval jen několik základních elementů, programově se jim správně říká symboly, jako kontakty typu A (NO – normally open), kontakty typu B (NC – normally closed), výstupní relé, časovače a čítače. Největší rozšíření elementů přineslo zabudování mikropočítačů do PLC. Při programování se používá kombinační a sekvenční logika. Každý element má svou funkci a pomocí spojovacích čar (tzv. příček) se vytváří program. Intuitivně běží program zleva doprava, tedy vlevo je „kolejnice“ s kladným pólem a vpravo potom „kolejnice“ s nulovým pólem. Mezi nimi je uživatelská kombinace elementů a příček reprezentující kód.
Na Obr. 4 je znázorněn kus kódu v jazyce Ladder Diagram. Element „LAMP“ je typu boolean, přičemž jeho vstup (zleva) je připojen na kladný pól. Zapnutí lampy (LAMP = true) spustí časovač „TON“, který po pěti sekundách aktivního vstupu „IN“ sepne výstup „Q“, čímž aktivuje element SET („LAMP OFF CONDITION“), což způsobí vypnutí lampy (LAMP = false).




Obr. 4 Ukázka kódu v jazyce Ladder Diagram v softwaru TIA portal (Siemens) [21]

**Funkční blokový diagram (FBD)**

Blokové programovací jazyky jsou typem grafického jazyka, který minimalizuje kód do bloků, což umožňuje jednoduchý způsob vytváření spustitelných příkazů FBD zejména popisuje funkci mezi vstupy a výstupy, které jsou spojeny připojovacími linkami. Logika vstupů a výstupů je uložena v blocích. Bloky jsou naprogramovány na listy a PLC tyto listy skenuje v pořadí nebo pomocí specifických spojení mezi bloky, podobně jako procedurální jazyky. Program se opět píše zleva doprava a shora dolů, stejně jako posloupnost při kompilaci programu. Celý kód v tomto jazyce je tvořen zapojením několika bloků do sebe. Tím je myšleno, že na začátku do prvních bloků jsou zapojeny proměnné a následně výstupy těchto bloků jsou brány jako vstupy dalších bloků až dokud se program nedobere k poslednímu, kde je nastavena proměnná v paměti nebo přímo výstup z PLC.

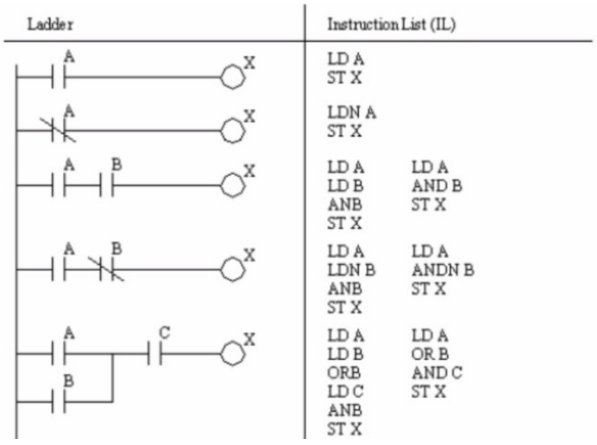



Obr. 5 Obecná ukázka FBD programování[22]

Na Obr. 5 je zobrazena obecná ukázka programování v jazyce FBD. Kód je tvořen třemi logickými funkcemi s dvěma vstupy datového formát BOOL. První dva bloky vlevo jsou typu OR, tedy logický součet. Blok vpravo je typu AND, tedy logický součin.

**Instruction list (IL)**

Tento jazyk je velice podobný assembleru. Instrukční list je také nazýván akumulátorový jazyk. Každý instrukce mění aktuální obsah akumulátoru. Skládá se především z operátorů (např. LD A, LDN B, ST B), modifikátorů (např. ANDN Y, AND X, OR Z) a časovačů. Lze jej přirovnat k Ladder Diagramu v textové podobě. Na Obr. 6 je zřetelný rozdíl mezi jazykem Ladder Diagram a Instruction List.




Obr. 6 Porovnání jazyka Ladder Diagram s jazykem Instruction List[23]

**Strukturovaný text (ST)**

V praxi se častěji stávalo, že grafické programovací jazyky plně nestačily softwarovým inženýrům, protože programy byly často rozsáhlé a tím se staly nepřehledné. Proto byl vytvořen nový jazyk, který snadno popisoval řídicí logiku. Proto byl zahrnut Struktuovaný Text do normy 61131-3 a byl navržen tak, aby odpovídal tehdy hojně používanému pascalu. Tento programovací jazyk je čistě textového formátu. Je mnohem blíže k tradičním programovacím jazykům jako jsou C, C++, Java nebo Python. Implementace ve strukturovaném textu umožňuje programátorovi vytvářet komplexní rutiny a toky, které nemusí být vždy tak snadno implementovány v logice Ladder Diagramu. Například smyčka FOR potřebuje pouze jeden řádek kódu k implementaci v ST. Kromě snazší orientace může v porovnání s ostatními jazyky ušetřit mnoho místa v paměti. Výrobci často upravují původní jazyk pro jejich vlastní programovatelné automaty.

Na Obr. 7 je znázorněn jednoduchý kód v jazyce Structuted Text. Pokud proměnná motor_start typu

```
IF motor_start AND motor_temperature > 80.0 THEN
  colling = true;
ELSE motor_stop THEN
  colling = false;
END_IF;
```

Obr. 7 Ukázka kódu v jazyce Structured Text

BOOL je v logické 1 (true) a zároveň je proměnná motor_temperature typu pravděpodobně REAL větší než 80.0, je zapnuto chlazení, coolling = true. Pokud motor stojí, tedy proměnná motor_stop = false, chlazení je vypnuto, colling = false

## 2. Virtuální uvádění do provozu

Pojmem virtuální uvádění do provozu (z angličtiny „virtual commissioning“) se rozumí použití simulační technologie k dosažení virtuálního návrhu, provozu nebo testování řídicího systému před jeho skutečným fyzickým zapojením. Reálné uvedení do provozu je zásadní součást procesu, avšak obvykle k němu dochází poměrně pozdě a během něho dochází ke zpoždění kvůli nedokonalosti programu (kódu). Cílem použití virtuálního uvedení do provozu je včasné ověření strojového kódu. Tímto je výrazně sníženo riziko nasazení programu obsahujícího chyby.
Virtuální uvedení do provozu bylo poprvé představeno na samotném konci milénia jako slibná technika na podporu vývoje strojů a zařízení. Současný stav technologie nabízí praktické řešení, která jsou automatizačnímu průmyslu přístupnější než kdy dříve. V rámci rostoucí potřeby urychlit vývoj a minimalizovat riziko při vývoji nových, inovativních produktů se digitální dvojčata a řešení virtuálního uvádění do provozu rychle stávají základními technologiemi.
Tento přístup navíc umožňuje technikům prozkoumat více možností, dávají možnost optimalizovat produkční systémy tak, aby co nejlépe odpovídaly obchodním požadavkům zákazníka, a to dokonce v širším rozsahu. Celkově je virtuální uvedení do provozu vysoce účinným nástrojem s mnoha výhodami, který pomáhá minimalizovat prostoje výrobního procesu technologie a optimalizovat celý systém.[24]

Virtuální uvedení do provozu zahrnuje tři hlavní fáze: model-in-the-loop (MIL), software-in-the-loop (SIL) a hardware-in-the-loop (HIL).

### 2.1. Model-in-the-loop

První etapa virtuálního uvedení do provozu se nazývá „Model-in-the-loop“ nebo-li testování modelu ve smyčce (MIL).
Hlavní myšlenkou je vytvořit model blokového diagramu softwarové logiky pro použití s programovatelnými automaty (PLC) a rozhraními člověk-stroj (HMI). Tento model je připojen k simulačnímu modelu, přičemž oba modely běží v reálném čase. Logický model představující PLC a HMI posílá příkazy do modelu představujícího produkční linku nebo malé celky. Odezva v podobě výsledků se vrací zpět do logického modelu a uzavírá smyčku. Pokud simulace způsobí chybu, jednoduše se upraví logický model a simulace je spuštěna znovu. Logický model se ladí tak dlouho, dokud simulace nevykazuje požadovaný výsledek.

Tento přístup umožňuje testování, simulace a ověření softwarové logiky s abstraktním chováním modelu, vyvinutého pro výrobní linky nebo menší celky. Jedná se o rychlou a snadnou metodu ověření v digitální podobě. [24]

### 2.2. Software-in-the-loop

Fáze MIL umožňuje ověřit chování logiky pro PLC a HMI. Dalším krokem je ověření, že logika v modelu je stejná, jakmile je zkompilována do softwaru. To je základní předpoklad fáze procesu virtuálního uvedení do provozu, který se nazývá „Software-the-loop“ nebo-li testování softwaru ve smyčce (SIL).
Kroky k nastavení a spuštění simulace SIL se odvíjí od simulace MIL. Nejprve je generován kompilovaný kód z logického modelu, přičemž tento kód lze automaticky generovat a spustit na emulátoru, který napodobuje reálný elektronický hardware. Software a emulátor jsou digitální ekvivalenty softwaru běžícího na fyzickém PLC nebo HMI. Tyto ekvivalenty jsou následně připojeny k modelu produkční linky a je vytvořena simulace v uzavřené smyčce v reálném čase.

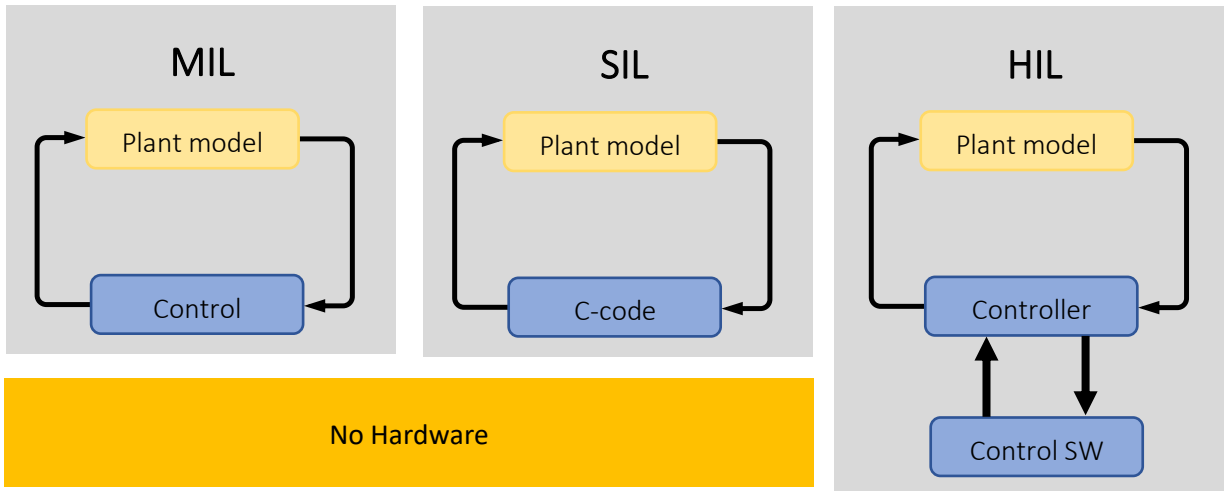
Cíl je podobný jako u simulace MIL a to odhalit chyby v kompilovaném softwaru. Logický model může fungovat za všech podmínek správně, ale po kompilaci do softwaru může dojít k dříve nezjištěným problémům. Pokud simulace MIL byla funkční a simulace SIL selhala, pak je zřejmé, že zdrojem poruchy je konverze logického modelu na kompilovaný software. [24]

### 2.3. Hardware-in-the-loop

Testování SIL umožňuje ověřit, že kompilovaný software pracuje správně na emulátoru PLC nebo HMI. Poslední fáze, hardware-in-the-loop nebo-li testování hardwaru ve smyčce (HIL), umožňuje ověřit, že kompilovaný software běží na skutečném fyzickém PLC nebo HMI.

Nastavení HIL simulace vychází ze simulace SIL. Kompilovaný software je spuštěn na skutečném PLC nebo HMI, ale přesto stále nemusí být připraven nebo správně nastaven pro konkrétní fyzickou produkční linku, proto jsou fyzické PLC a HMI připojeny k digitální simulaci produkční linky nebo celku. Tento celek běží v reálném čase.
Tímto postupem je možné ověřit, že kompilovaný software běží na skutečných fyzických PLC a HMI bez čekání na sestavení fyzické produkční linky. Selhání v tomto okamžiku naznačuje problém s kompilovaným softwarem, který byl spuštěn na fyzických kontrolérech. To znamená, že hledání po příčinách chyb se opět eliminovalo na úzký okruh, což šetři nejen čas ale i peníze.[24]
Poruchy a chyby se tak dějí stále jen ve virtuálním simulačním prostředí a jsou odstraněny dlouho předtím, než je systém uveden do fyzické produkční linky.

HIL také umožňuje zjistit stavy, které nastanou během extrémních událostí. I když je velmi nepravděpodobné, že by k takovým událostem došlo, mohly by mít zničující dopad na produkční linku. Po úspěšném testování by se mělo zamezit či předejít katastrofickým scénářům. Blokové srovnání typu simulací je znázorněno na Obr. 8.




Obr. 8 Blokové znázornění rozdílů simulací ve smyčce[26]

## 2.4. Digitální dvojče

Namísto uvedení fyzické technologie do provozu se stále více společností obrací k virtuálnímu uvedení do provozu pomocí digitálního dvojčete. Tato virtuální replika emuluje a předpovídá chování finálního fyzického produkčního systému pomocí digitálních a simulačních prostředků. Jednou z významných výhod virtuálního uvedení do provozu je, že na rozdíl od fyzického uvedení do provozu není nutné mít k dispozici veškerý hardware technologie, jako jsou, PLC, HMI, senzory, napájecí zdroje a další. Virtuální uvedení do provozu může začít dlouho předtím, než dorazí jakýkoli hardware, nutný k fyzickému uvedení do provozu. Inženýři mohou začít testovat interoperabilitu v digitální sféře, neboť všechny fyzické komponenty existují v softwarovém simulačním modelu. Tímto se transformuje fyzické uvedení do provozu na testování digitálního dvojčete.
Další výhodou je možnost prozkoumat více alternativ designu nebo konstrukčních řešení. Pomocí digitálního dvojčete lze vyzkoušet mnoho různých PLC, HMI nebo jiných zařízení. Celý proces má dopad na lepší a efektivnější výrobní systém. Fyzické uvedení do provoz znamená velice málo nebo téměř žádný čas na prozkoumání alternativ řídicího systému.
Pomocí digitálního dvojčete lze ověřit výkon a chování jednotlivých hardwarových komponent odděleně od celého systému. K digitální simulaci je rovněž možné připojit fyzické PLC nebo HMI zařízení a simulaci spouštět v reálném čase. Tomuto způsobu testování se říka Hardware in the loop (HIL). Testování HIL je technika, při které jsou reálné signály z řídicí jednotky připojeny k testovacímu systému, který simuluje realitu. Testovací iterace probíhají naprosto totožně, jako by byl používán reálný fyzický systém, viz kapitolu 2.3. Snadno lze projít tisíce možných scénářů řídicí jednotky, bez vysokých nákladů a času spojeného se skutečnými fyzickými testy.

## 2.5. Možnosti virtuálního uvádění do provozu

Před zahájením virtuálního uvádění do provozu je nutné správně zvolit typ digitálního dvojčete. Existuje mnoho úrovní a konceptů digitálních dvojčat od senzorových modelů až po fyzikální modely jednotlivých strojů. Je důležité přesně znát typ digitálního dvojčete, aby bylo možné dosáhnout požadovaných výsledků testování.
Modely vytvořené před existencí fyzického produktu jsou založeny na různých typech simulačních technologií a každý typ digitálních dvojčat poskytuje jiný pohled do různých oblastí automatizace. Bez ohledu na to, jaký typ je využit, vždy uživatel ušetří spoustu času a peněz. Ve srovnání s testováním na samotném fyzickém produktu umožňují digitální dvojčata snížit počet prototypových cyklů, otestovat stroje bez rizika poškození hardwaru a najít efektivnější řešení díky pokročilým poznatkům ze simulace. Existuje mnoho softwarů pro virtualizaci a simulaci. Každý software přináší různé výhody pro různá použití. Společnosti se snaží inovovat dosavadní software nebo vyvíjet novější, rychlejší a jednodušší. Některé z těchto softwarů jsou více popsány níže.

### 2.5.1. Digitální dvojče tovární (procesní) úrovně

Digitální dvojčata tohoto typu pomáhají modelovat výkonnost celého závodu a procesů v něm, které probíhají mezi jednotlivými stroji. Tento typ digitálních dvojčat má sklon spoléhat se méně na podrobnou fyziku každé jednotlivé součásti stroje a klade důraz na optimalizaci toku materiálů po celé lince.

Digitální dvojčata na úrovni závodu nebo procesu jsou vhodná například pro:

- **Sekvenování** - nalezení nejlepší konfigurace parametrů zařízení s cílem co nejvíc zefektivnit pohyb produktů mezi buňkami strojů
- **Propustnost** - optimalizace parametrů procesu s cílem maximalizovat výkon daného závodu

- **Rozložení závodu** - hledání způsobů, jak fyzicky navrhnout celý závod tak, aby byla maximalizována jeho efektivita

Vhodným softwarem pro tento typ digitálního dvojčete je **Emulate3D**. [28]

### 2.5.2. Digitální dvojče strojové úrovně

Strojová úroveň zužuje rozsah možností pouze na jeden stroj či zařízení dané technologie. Tato úroveň používají vysoce přesná simulační data dynamiky strojů včetně mechanického, elektrického, hydraulického a dalšího chování. Tento typ digitálního dvojčete je vhodný pro vývoj nových výrobních strojů. Poskytují celou řadu schopností pro rychlé sledování vývoje a opět jsou spojeny s nízkými náklady v porovnání s fyzickým modelem stroje. Digitální dvojčata na strojové úrovni jsou vhodná například pro:

- **Dimenzování motorů a mechanismů** - pomocí vysoce věrných simulací zatížení stroje lze najít přesné požadavky na motory a další mechanismy
- **Maximální bezpečná propustnost** – testování maximální propustnost pro daný návrh stroje a pracovní cyklus, tedy není nutné riskovat fyzické poškození stroje v reálném modelu
- **Testování řízení stroje** - hledání nejlepší možné strategie řízení pro daný stroj testováním kódu na základě realistických virtuálních modelů (virtuální uvedení do provozu).

Vhodným softwarem pro tento typ digitálního dvojčete je **MapleSim**. [28]

### 2.5.3. Digitální dvojče dílů

Tato úroveň je nejnižší stupeň analýzy. Předmětem testování jsou jednotlivé součásti samotných materiálů. Na základě vlastností materiálů se provádí komplexní simulace náročné na znalosti týkající se skutečného výkonu jednotlivých částí, kvůli reagování na provozní napětí a namáhání. Takovými simulacemi lze zjistit, jak se budou chovat určité tekuté materiály při pohybu nebo zpracovávání v různých průmyslových odvětvích.

Digitální dvojčata na úrovni dílů jsou vhodná například pro:

- **Deformace materiálu** - testování dílů pod různým napětím, s cílem simulovat jejich potenciální deformace, poruchy a nedokonalosti
- **FEA / CFD** - dvě výkonné simulační technologie pro simulaci materiálů ve 3D prostoru nebo chování tekutin
- **Návrh dílů** - hledání optimální velikosti, tloušťky a celkového vzhledu dílů s cílem splnit požadavky užití

Vhodným softwarem pro tento typ digitálního dvojčete je **SolidWorks Simulation**. [28]

## 2.6. SIMIT

Simulační software SIMIT od firmy SIEMENS je jeden z nejrozšířenějších software pro virtuální uvedení do provozu. Umožňuje simulaci a emulaci v reálném čase pro komplexní zkoumání automatizačních řešení.

SIMIT je založen na jednotné simulační platformě, která umožňuje nejen virtuální uvedení automatizačního inženýrství systémů, strojů a procesů do provozu, ale také realistické vzdělávací prostředí pro obsluhu zařízení. To lze snadno provést přímo na pracovišti, a to i bez potřeby vybavení nebo potřeby hloubkových znalostí simulace. Pro ovládání se používá skutečný nebo virtuální automatizovaný systém, například virtuální kontrolér SIMIT. Instance virtuálního kontroléru SIMIT mohou emulovat automatizační systémy SIMATIC S7-300 / S7-400 z produktové řady SIMATIC S7 a SIMATIC PCS 7 běžně používané v automatizačních procesech.

Mnoho testů pro detekci a eliminaci potenciálních poruch lze provést ještě předtím, než bude k dispozici skutečný projekt (výrobní linka, závod, elektrárna, apod).Tímto způsobem je možné

optimalizovat kvalitu konfiguračního procesu bez rizik, způsobených ve skutečném, fyzickém projektu. Jedná se zejména o aplikace správných identifikací a testování blokovací nebo propojovací logiky.

SIMIT V10.2 se používá v kombinaci s následujícími platformami:

- SIMATIC PC 7
- SIMATIC PCS neo
- TIA portal
- STEP 7

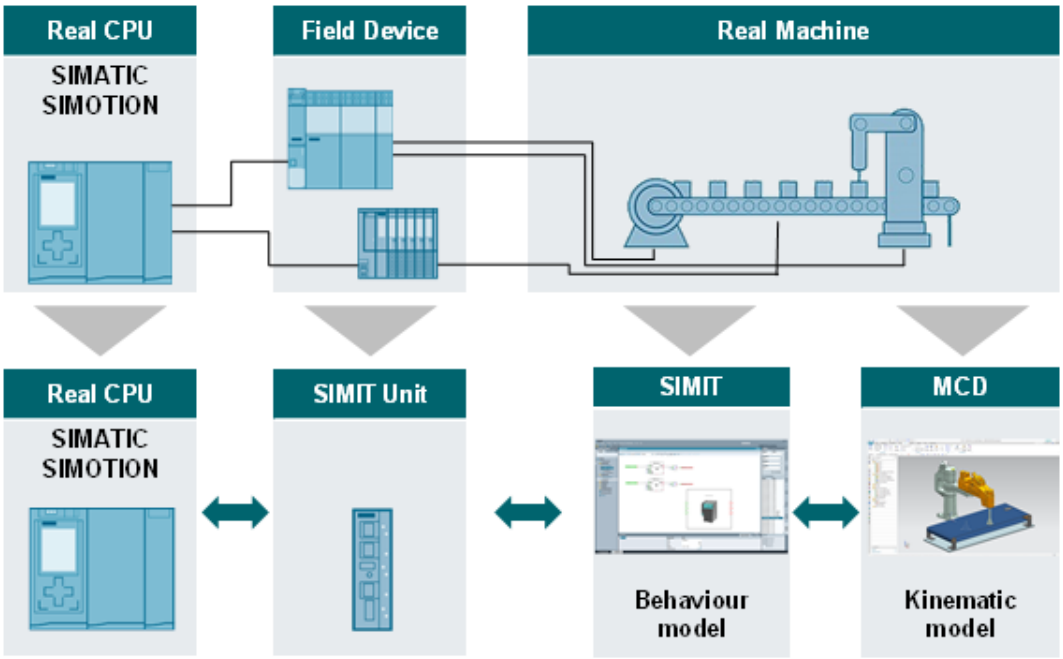
Výhody použití nástroje SIMIT:

- Testovací a tréninková prostředí bez skutečného hardwaru
- Virtuální kontroléry pro emulaci automatizačních systémů
- Flexibilní simulační a emulační prostředí pro projekty libovolného rozsahu
- Synchronizovaná simulace a emulace v reálném nebo virtuálním čase
- Testování původního automatizačního projektu
- Vyšší kvalita pro konfiguraci automatizace
- Kratší doba uvedení do provozu a riziko v důsledku předběžného testování
- Bez nutnosti konfigurace simulace v projektu automatizace

SIMIT běží na zařízeních s operačním systémem Microsoft Windows i na virtuálních systémech (VMware ESXi Server V6.7). Je možná flexibilní aplikace a integrace je možná prostřednictvím otevřených rozhraní do tovární automatizace se SIMATIC S7 a SIMATIC WinCC nebo do procesní automatizace se SIMATIC PCS 7 nebo SIMATIC PCS neo.
Protože modely lze vypočítat v reálném čase, lze SIMIT propojit se skutečným automatizačním procesem („hardware-in-the-loop“) pomocí jednotky SIMIT pro připojení přes rozhraní PROFINET nebo PROFIBUS. Test „software-in-the-loop” je možný také prostřednictvím virtualizace automatizačního systému pomocí emulačního softwaru S7 - PLCSIM nebo S7 - PLCSIM Advanced nebo integrovaného virtuálního kontroléru SIMIT.

Propojení se skutečným automatizačním systémem se obvykle provádí přes PROFIBUS DP nebo PROFINET IO s rozhraními (jednotky SIMIT), které simulují zařízení na PROFIBUS DP / PROFINET IO. Spojku PRODAVE lze také použít pro rozhraní MPI / DP nebo IE automatizačního systému pro přenos procesních dat pomocí SIMIT.
K SIMIT lze připojit další simulační modely:
Výměna dat prostřednictvím standardizovaných rozhraní, jako jsou OPC DA, OPC UA a sdílená paměť
Výměna dat prostřednictvím jednoho volně programovatelného externího propojení (uživatelem)
Synchronizace přes rozhraní dálkového ovládání
V případě propojení přes rozhraní dálkového ovládání může být SIMIT buď master nebo klient (slave) pro další simulace. Pomocí správy virtuálního času lze simulace také implementovat rychleji nebo pomaleji než v reálném čase.

Na Obr. 9 je pohled na fyzickou realizaci projektu ve srovnání s procesem simulačním.




Obr. 9 Srovnání simulačního procesu s procesem reálným[27]

**Simulační platforma SIMIT**
SIMIT lze dokonale přizpůsobit individuálním požadavkům pomocí čtyř různých softwarových balíčků, aby vyhovovaly velikosti projektu:

- SIMIT Engineering S: 2 500 simulačních značek
- SIMIT Engineering M: 15 000 simulačních značek
- SIMIT Engineering L: 200 000 simulačních značek
- SIMIT Engineering XL: 1 000 000 simulačních značek

*Pozn.: Značka = tag.*

Výše zmíněné balíčky obsahují:

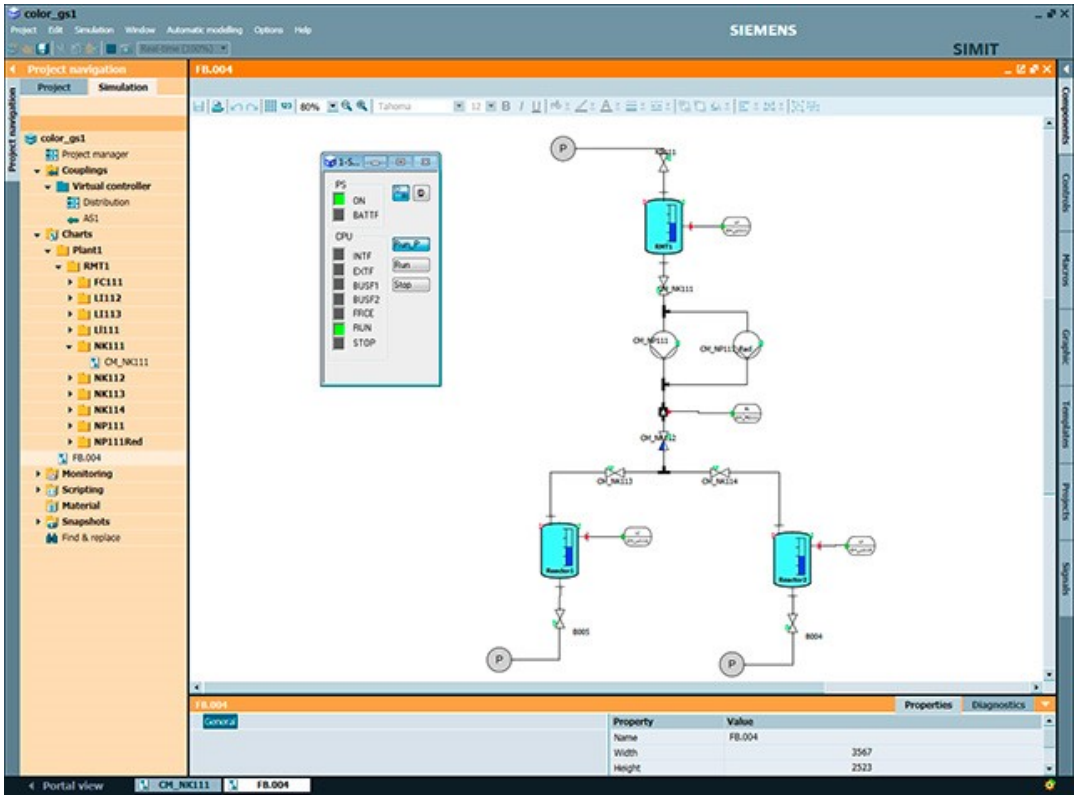
- Portálový pohled se správou pracovního toku pro vytvoření simulačního projektu
- Standardní knihovna komponent
- 3D prohlížeč založený na VRML (Virtual Reality Modeling Language)
- Rozhraní pro PROFIBUS DP, PROFINET IO a PRODAVE
- Rozhraní pro SIMIT Virtual Controller a OPC
- Trendy a zprávy (TME)
- Skriptovací prostředí
- Editor pro vytváření makro komponent (MCE)
- Editor pro vytváření dynamické grafiky a animací (DGE)
- Rozhraní automatického řízení (ACI)
- Automatické generování seznamů signálů z dat SIMATIC Manageru
- Runtime pro komponenty vyvinuté pomocí editoru typů komponent
- Rozhraní S7 – PLCSIM, S7 – PLCSIM Advanced, OPC a dálkové ovládání

- Modifikace simulačního modelu za běhu
- Simulace ve virtuálním čase
- Inženýrská účinnost pro SIMATIC PCS 7 (SMD)
- Automatické generování modelu na základě šablon
- Hromadné inženýrství
- Rozhraní sdílené paměti pro vysoce výkonné propojení
- XML rozhraní pro automatické generování modelů a připojení

SIMIT lze také rozšířit o další knihovny. Následující knihovny rozšíření zpřístupňují konkrétní technologické komponenty:

- Knihovna SIMIT FLOWNET: knihovna pro simulaci tokových sítí s homogenními médii (voda / plyny) včetně tlaků, teplot a průtoků.
- Knihovna SIMIT CONTEC: knihovna pro 2D simulaci zařízení pro manipulaci s materiálem.
- Základní knihovna SIMIT CHEM: Pro zjednodušené vytváření simulací v chemickém a farmaceutickém průmyslu. Propojením komponent těchto knihoven vzniká SIMIT model potrubní sítě (tzv. Flow network), který lze použít k simulaci termodynamických procesů v potrubní síti. Tokové sítě pak spojují komponenty s charakteristikami úložiště, např. kontejnery. Knihovna CHEM BASIC umožňuje v SIMITu použít speciální metodu řešení, která vypočítává průtoky, tlaky a specifické entalpie během simulace potrubních sítí.

V SIMIT je také možné vytvářet typy komponentů (Obr. 10) podle vlastních požadavků a funkčního očekávání s Type Component Editorem (CTE). CTE je rozšiřující modul SIMIT, který umožňuje vysoký stupeň flexibility. Nabízí možnost individuálně vyvíjet i složité simulační moduly podle vlastních potřeb nebo přizpůsobit stávající komponenty pro konkrétní typ aplikace. Chování a vzhled nových komponent je čistě na fantazii programátora.




Obr. 10 Modelování komponentů v SIMIT[35]

Komponentní modelování zařízení založené na toku signálu se provádí prostřednictvím grafického uživatelského rozhraní SIMIT podporovaného rozšiřitelnými základními knihovnami. K tomu jsou z knihovny vybrány předdefinované komponenty, umístěny na grafické rozhraní, vzájemně propojeny a nastaveny parametry. Kromě toho lze simulační model generovat exportem technických dat z COMOS. Speciální simulační znalosti nejsou u programátora vyžadovány.
Efektivní simulace je založena na abstrakci na třech různých úrovních: signálech, zařízeních (např. akční členy a snímače) a odezvě technologického procesu. Zde je technologická odezva reprezentována matematicky a logicky nebo pomocí aditivních knihoven.

Aditivní knihovny podporují simulaci technologické odezvy:

- FLOWNET: Lze použít pro rychlou a jednoduchou simulaci dynamických procesů tlaků, průtoků a teplotního rozdělení vody v potrubních sítích.
- CONTEC: Lze použít pro simulaci zařízení pro manipulaci s materiálem.
- CHEM BASIC: Lze rychle a snadno simulovat modely potrubních sítí v chemickém a farmaceutickém průmyslu. Pomocí CHEM BASIC lze modely z COMOS P&ID automaticky generovat prostřednictvím generického importu.

Uživatel může také vytvářet vlastní komponenty a šablony, které umožňují efektivní modelování podle přání zákazníka.

Informace v kapitole 2.6 jsou převzaty z oficiálních webových stránek výrobce [30].

## 2.7. WinMOD

WinMOD vznikl již v roce 1995 na základě Windows a jeho úkolem je modelování komunikací, strojů a zařízení, které jsou řízeny automatizovanými systémy. WinMOD tedy pomáhá realizovat automatizační systém ve virtuálním světě. Modulární systémová platforma WinMOD vytváří vlastní systémy WinMOD pro virtualizaci, virtuální uvedení do provozu a další.

WinMOD nahrazuje systém reálného světa virtualizovaným systémem a simuluje jeho chování v reálném čase. Vizualizace signálů, reakcí a časových průběhů pomáhá pochopit chování systému a to dokonce lépe než ve skutečnosti. Nahrazení reálného systému virtualizovanými systémy lze provést v úplném rozsahu nebo tvořit pouze část z celého automatizovaného procesu, přičemž kombinuje virtualizovaný systém s automatizačním systémem a vytváří plně funkční proces pro automatizovaný stroj nebo zařízení.

Výhody WinMOD:
- Lze připojit k jednomu nebo k více automatizovaným systémům.
- Lze se propojit s jedním nebo s více skutečnými nebo simulovanými subsystémy (např. robotické řízení).
- Lze připojit k dalším inženýrským nástrojům pro výměnu dat.
- Lze spojit s dalšími simulačními nástroji s integrovaným runtime systémem.

*Pozn.: „Runtime je doba, kdy běží program. Začíná tehdy, když se program spustí a končí, když se program vypne nebo zavře. Runtime je čistě technický termín, který se používá ve spojitosti se softwarem.“* [48]

Co umožňují WinMOD systémy:

- Ovládací software může být zcela nebo částečně zadán systémem skutečné nebo virtuální automatizace.
- Uvedení kontrolního softwaru do provozu může proběhnout na jakémkoli místě bez časového omezení
- Virtualizovaný stroj nebo proces se stává funkčním digitálním dvojčetem.
- Digitální dvojče se používá po celou dobu životnosti výrobního závodu pro optimalizaci řídicího softwaru, pro validaci migrace, pro kvalifikaci provozovatelů závodů a pracovníků údržby.

WinMOD se používá pro:

- Testování a ověření softwaru AS
- Testování převzetí systému (FAT- Factory Acceptance Test)
- Virtuální uvedení do provozu
- Digitalizace pro průmysl 4.0
- Digitální dvojče pro plánování, kvalifikaci a migraci virtuálního uvedení do provozu
- Digitální vzdělávání pro automatizaci s technologiemi

WinMOD spojuje:

- WinMOD System se systémem automatizace se skutečnou a virtuální komunikací
- WinMOD System s emulovanou technologií fieldbus
- WinMOD System s dalšími simulačními systémy, jako je např. simulace robotů nebo simulace procesů
- Různé technologie ve výrobním závodě

*Pozn.: „Fieldbus je označení pro rodinu komunikačních protokolů pro průmyslovou aplikaci, od roku 1999 normalizovaných ve standardu IEC 61158. Výraz fieldbus by se zhruba dal přeložit jako sběrnice pro nasazení v terénu, nebo aplikační sběrnice.“* [49]

WinMOD nabízí kompletní řešení simulací a knihovny pro některá hlavní odvětví a to:

**Automatizace továrny:**

- Lehké a těžké elektrické monolitické systémy
- Kontejnerové dopravní systémy
- Dopravní systémy palet
- Automatické úložné systémy
- Elektrické podlahové systémy
- Dopravní systémy SKID
- Skillet-SKID Conveyor Systems
- Obecná obsluha

**Automatizace procesů:**

- Ukládání a distribuce
- Plyn a hromadné zboží
- Chladicí systémy
- Klimatizace
- Dopravníky hromadného zboží
- Hoření a zahřívání

**WinMOD-SIMLINE**

Softwarový balíček WinMOD-SIMLINE je rozšířením systémového softwaru WinMOD. Jádrem je 3D prostředí pro simulaci dopravníkových systémů v reálném čase, manipulačních procesů a vizualizaci strojových procesů rovněž v reálném čase s využitím existujících CAD souborů. Všestranný inženýrský systém umožňuje rychle a efektivně realizovat a kombinovat všechny aspekty daných požadavků.

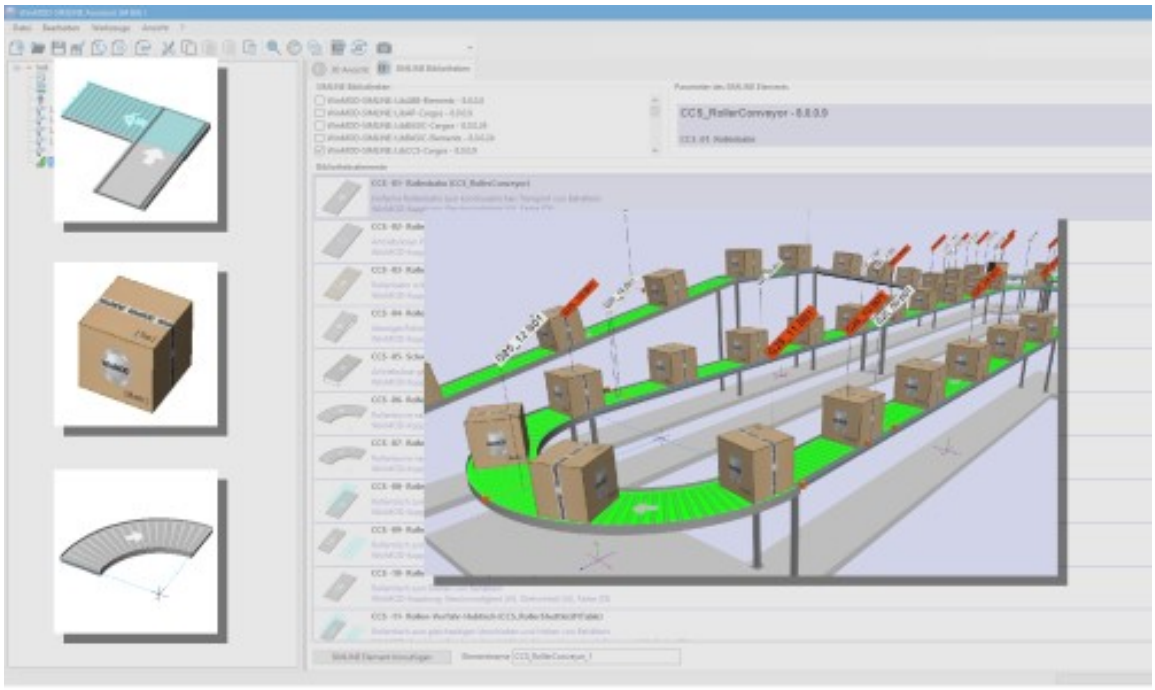
WinMOD-SIMLINE obsahuje knihovny pro různé dopravníkové systémy (například technologie paletových dopravníků, skladovací systémy). Pro vývojaře je k dispozici 3D editor WinMOD-SIMLINE Asistent pro integraci dat 3D CAD.

Nástroj WinMOD-SIMLINE Layouter slouží pro efektivní tvorbu velkých komplexních dopravníkových systémů, viz Obr. 11. Je zajištěna vysoce optimalizovaná komunikace se systémovým softwarem WinMOD a

automatizované generování všech systémových signálů potřebných pro automatizaci (například světelné bariéry, dopravní rychlosti apod.). Je možné lineární ladění parametrů systému (např. seřízení senzorů) a to přímo během provozu simulace.

Věrné zachycení geometrie a trajektorie komponentů a přepravovaného zboží v reálném čase pomocí senzorů spolu si 3D kolizemi v reálném čase pro procesy přetížení.

Na Obr. 11 je ukázka simulačního prostředí WinMOD SIMLINE spolu se simulací dopravníkové technologie.



Obr. 11 Ukázka modelace dopravníkové technologie[32]

Informace v kapitole 2.7 jsou převzaty z oficiálních webových stránek výrobce [30].

## 2.8. Tecnomatix

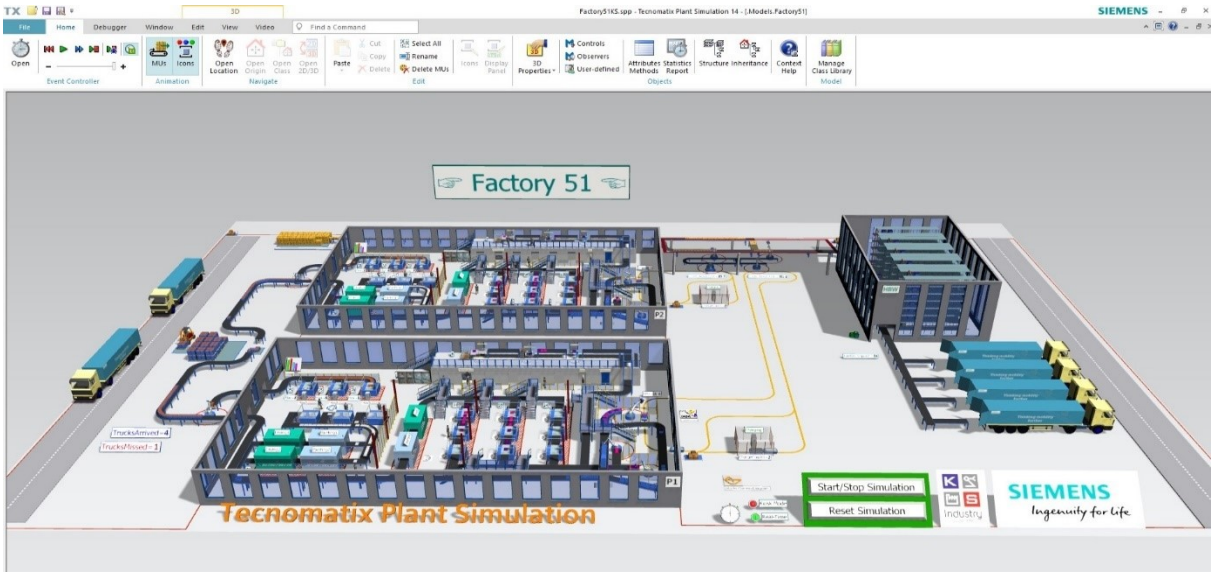
Software Tecnomatix zlepšuje efektivitu výroby v mnoha oblastech, jako je robotika a řízení automatizace, konfigurace pracoviště pro efektivitu a bezpečnost, stejně jako optimální umístění zařízení a plánování toku materiálu. Pokročilé nástroje pro plánování a ověřování vám umožňují optimalizovat výrobní linky ve fázi plánování, než se přechází do fáze implementace. To umožňuje maximalizovat efektivitu a minimalizovat investiční náklady.
Tecnomatix je kompletní sada řešení pro digitální výrobu, která umožňuje přijímat více informací a rychlejší způsoby pro identifikaci, zvýšení efektivity, snížení nákladů a dosažení požadované úrovně kvality.

**Výhody a výhody softwaru:**

- Zvýšení produktivity plánování.
- Optimální efektivita výroby.
- Zvyšování produktivity závodu.
- Optimalizace výkonu a používání postupů osvědčených postupů.
- Trojrozměrný tovární design a vizualizace.
- Analýza a optimalizace tovární logistiky.
- Simulace propustnosti tovární výroby.

**Software lze rozdělit do modulů:**

- Optimalizace výrobních linek a logistiky - simulace závodu.
- Offline programování robotů a virtuální uvedení PLC a regulátorů do provozu.
- Ergonomie lidské práce.




Obr. 12 Ukázka simulace továrny v Tecnomatix[34]

Pomocí softwaru Tecnomatix je možné optimalizovat tok materiálu a využít logistiku na všech úrovních plánování, od globální výroby přes jednotlivé továrny až po jednotlivé menší výrobní linky.
„Plant simulation“ (simulace továrny) je průmyslovým standardem pro vytváření modelů diskrétní simulace událostí (DES). Program založený na moderním, objektově orientovaném a hierarchickém přístupu patří k nejnovější 3. generaci programů pro simulaci 2D a 3D náhodných událostí.
Program je v souladu se standardy Microsoft Windows a je srozumitelným a snadno použitelným nástrojem připraveným k okamžité práci. Většina činností je prováděna na základě funkce „Drag and drop“, která významně zkracuje dobu stavby modelu a zvyšuje komfort používání.

Plant simulation (Obr. 12) zahrnuje standardní knihovny objektů pro vizualizaci materiálových procesů a toků, ale také umožňuje vytvářet a vyvíjet další objekty pro rozšíření standardních knihoven.
Komplexní analytické nástroje umožňují rychlou interpretaci dat, která jsou generová modelem a umožňují odpovědět na klíčové otázky týkající se výkonu, efektivity, úrovně nákladů, využívání zdrojů i spotřeby energie.
Otevřenost systému a řada rozhraní jako jsou: Active-X, ODBC, Excel, Socket, VRML, COM, DDE, umožňuje výměnu dat s jinými programy i integraci s dalšími systémy, např. třídou ERP (Enterprise Resource Planning).
Pokročilé genetické algoritmy a nástroje na nich založené umožňují najít optimální řešení komplexních simulačních problémů.
*Pozn.: „Drag and drop" je funkce, kdy uživatel uchopí objekt kurzorem myši a přetahuje na libovolné místo.*
*„Plánováním podnikových zdrojů (ERP) se rozumí typ softwaru, který organizace používají ke správě každodenních obchodních činností, jako je účetnictví, zadávání zakázek, řízení projektů, řízení rizik a operace dodavatelského řetězce."* [36]

Informace v kapitole 2.8 jsou převzaty z webových stránek KS industry [30].

## 2.9. KUKA.Sim

KUKA je globální automatizační korporace se sídlem v německém Augsburgu, jejíž velikost na poli automatizace podtrhuje zhruba 14 000 zaměstanců. KUKA jako jeden z předních světových dodavatelů řešení pro inteligentní automatizační systémy nabízí zákazníkům vše, co potřebují, a to z jediného zdroje: od robotů a menších celků až po plně automatizované systémy a jejich propojení na trzích, jako je automobilový průmysl, elektronika, materiálové inženýrství v oblasti kovu a plastu, spotřební zboží, elektronický obchod/maloobchod a zdravotní péče.

Inteligentní simulační software KUKA.Sim je ideální řešení pro efektivní offline programování: pomocí KUKA.Sim je možné snadno a rychle optimalizovat výrobní operace automatizačních systémů a robotů. KUKA svým simulačním softwarem zaručuje zvýšení flexibility, produktivity a konkurenceschopnosti.
Hlavním rysem využití KUKA.Sim je optimální rozvržení pro výrobní systémy, které lze tvořit již počáteční fázi projektu. KUKA.Sim nabízí inteligentní komponenty z knihoven, které lze přetahovat na potřebné místa v projektu a tím prozkoumat různé alternativy a ověřit koncepce s minimálním úsilím.

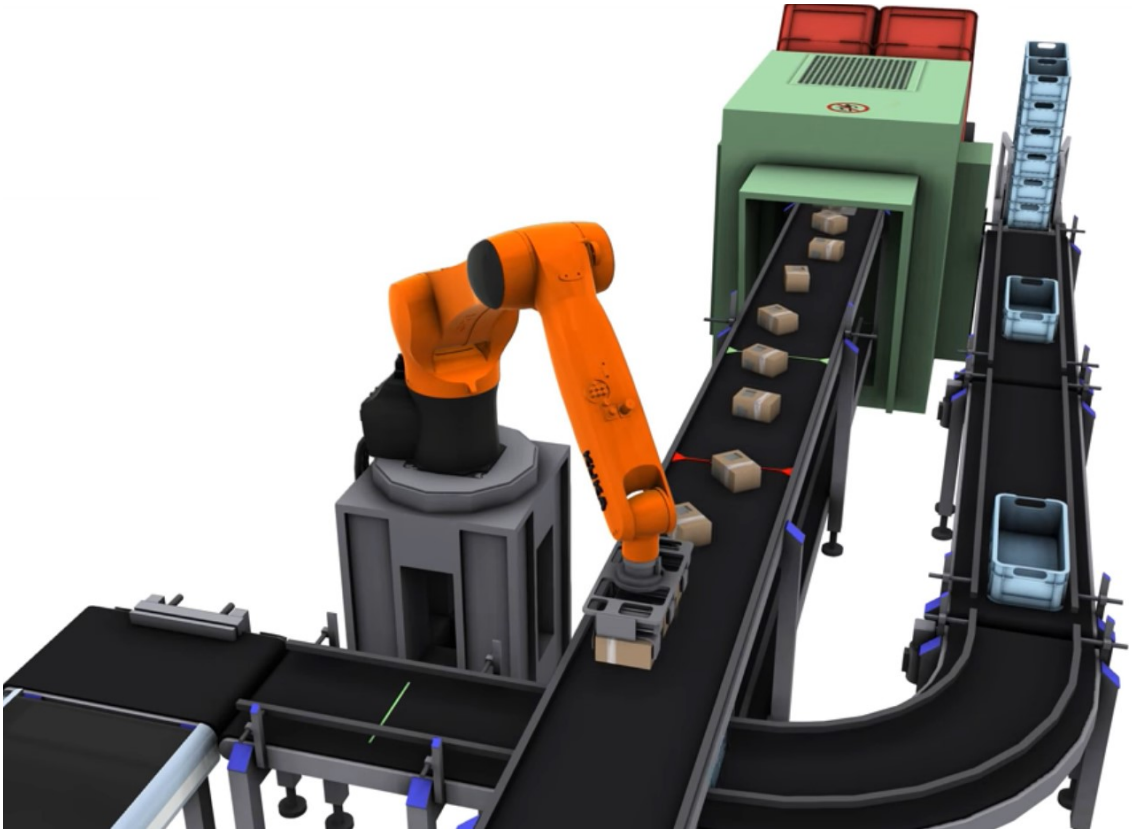
Společnost KUKA jde inovacím napřed a disponuje aplikací pro virtuální realitu nazvanou Mobile Viewer. KUKA. umožňuje vstoupit do světa virtuální reality, což je velice konkurence schopná vymoženost. Pomocí hardwaru pro virtuální realitu lze virtuálně předvést systémové koncepty. Zákazník tak dostáva naprosto věrný pohled na budoucí automatizační systém.
Pomocí aplikace Mobile Viewer lze snadno na tabletech nebo chytrých telefonech za běhu automatizačního procesu rychle a snadno zobrazovat výsledky simulace.

**Funkční rozsah verze KUKA.Sim Pro 3.1**

- 64bitová aplikace pro nejvyšší úroveň výkonu CAD
- Integrovaný import CAD (CATIA V5, V6, JT, STEP, RealDWG atd.)
- Rozsáhlá online knihovna s aktuálně dostupnými modely robotů atd.
- Konfigurovatelné testování kolizí a kontrola vzdálenosti
- 2D kresby (RealDWG) exportní funkce
- Exportní funkce AVI HD videa a 3D PDF
- Podpora 3D myši (např. 3Dconnexion)

- Rozhraní OPC UA umožňující PLC připojení pro Beckhoff TwinCAT, CodeSys, Siemens PLCSIM Advanced (Tia Portal)
- Modelování stránky pro vytváření vlastních součástí
- Podpora aplikace Prohlížeč mobilů
- Podpora virtuální reality (vyžaduje se další VR hardware)
- Určení doby cyklu
- Podpora NVIDIA PhysX

Na Obr. 13 je ukázka simulace v KUKA.Sim včetně robotu od společnosti KUKA.



Obr. 13 Ukázka simulace robotu KUKA [38]

Informace uvedené v kapitole 2.9 jsou převzaty z oficiálních stránek výrobce. [37][38]

## 3. B&R X20 systém

Systém X20 od firmy B&R stanovuje nové standardy podle hesla „Perfection in Automation“. Systém X20 se zrodil ze zkušeností získaných z aplikací po celém světě, po četných rozhovorech se zákazníky s cílem snazšího, ekonomičtějšího a bezpečnějšího používání. Je novým univerzálním řešením pro všechny automatizované aplikace v oblasti výrobních strojů a řídicích systémů. Na Obr. 14 je zobrazen charakteristický vzhled X20 systému. Tento hardware se jednoduše připojí na DIN lištu v rozvaděči.

Obr. 14 Charakteristický vzhled PLC a modulů pro X20 systém[4]

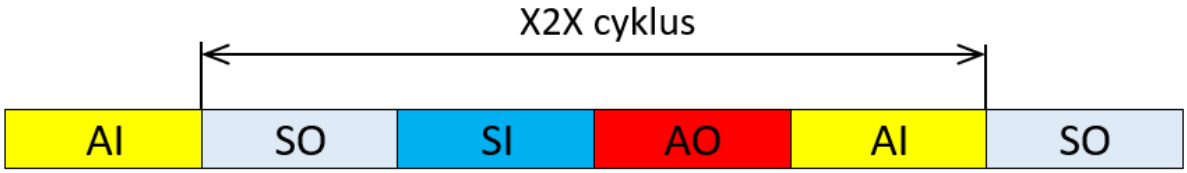
### 3.1. Komunikace X2X link

Systém X20 nastavuje nove standardy účinnosti. Nesmírnou výhodou v komunikaci je technologie X2X link, která umožňuje přesně umístit komponenty v systému X20 všude tam, kde je volné místo.
Je to decentralizovaný „backplane“ pro vstupně-výstupní moduly. Všechny moduly X2X jsou navzájem propojeny přímkou přes spojení z bodu do bodu. Každý modul má přijímač X2X a vysílač X2X. Veškerá data, která dorazí do přijímače X2X, jsou okamžitě předána vysílači X2X. Signál je pak kompletně přepracován. [44]

*Pozn.: „backplane“ se překládá jako „propojovací rovina“* [41]

**Princip činnosti**

Protokol X2X definuje cyklus s pevnou délkou. V tomto cyklu jsou odeslány čtyři snímky. Lze zvolit libovolný čas cyklu X2X (v krocích po 1 µs), přičemž nejrychlejší možný čas cyklu je 100 µs. Cyklus X2X se neustále opakuje. [39]




Obr. 15 X2X cyklus [39]

Vysvětlivky zkratek [39] Obr. 15**:**

**SO - SyncOut**

Synchronní výstupní data. Data posílá master do modulů. Tyto údaje se neustále opakují. Například stavy kanálů v digitálním výstupním modulu se přenášejí v každém cyklu X2X. Každému modulu je v tomto rámci přiřazena pevná poloha (a délka). Toto přiřazení provádí automaticky Automation Runtime.

**SI - SyncIn**

Synchronní vstupní data. Data jsou odesílána z modulů na hlavní počítač (PLC CPU nebo kontrolér sběrnice). Tyto údaje se neustále opakují. Například stavy kanálů v digitálním vstupním modulu se přenášejí v každém cyklu X2X. Každému modulu je v tomto rámci přiřazena pevná poloha (a délka). Toto přiřazení provádí automaticky Automation Runtime.

**AO – AsyncOut**

Asynchronní výstupní data. Data jsou posílána sporadicky z hlavního modulu do modulů (např. konfigurace, parametry, přístup k asynchronním kanálům). Prostor, který je k dispozici pro asynchronní komunikaci, je distribuován stávajícím modulům aplikací Automation Runtime.

**AI – AsyncIn**

Asynchronní vstupní data. Data jsou zasílána sporadicky z modulu do hlavního počítače (např. konfigurace, parametry, přístup k asynchronním kanálům, diagnostika). Prostor, který je k dispozici pro asynchronní komunikaci, je distribuován stávajícím modulům aplikací Automation Runtime.

V

Tab. 1 jsou rozepsány technické parametry X2X link.

Tab. 1 Technické parametry X2X link[39]

| Vlastnost | Hodnoty/ Funkce |
|---|---|
| Princip přenosu | Spojení bod-bod, jednosměrný ,data uspořádaná v rámečcích, detekce chyb pomocí CRC-32. |
| Topologie | Líniová struktura ( všechny moduly v řadě) |
| Minimální doba cyklu | 100 µs |
| Maximální počet modulů | 253 |
| Maximální vzdálenost mezi 2 moduly | 100 m |
| Počet snímku mezi 2 XYZ cykly | 4 |
| Maximální vstupní data | 8192 bajtů |
| Maximální výstupní data | 8192 bajtů |
| Maximální data modulu | Až 255 bajtů pro vstup a 255 bajtů pro výstup (záleží na typu modulu) |
| Nastavení číslovaní modulů | Automaticky zvýšen o 1, je možné ovlivnit pomocí čísel uzlů modulů ( příklad X67DM9321X67DM9321) |

### 3.2. Vstupní, výstupní a přídavné moduly

Firma B&R se snažila co nejvíce inovovat dosavadní produkty a nevynechala ani I/O moduly. Promyšlené detaily a sofistikovaný ergonomický design nabízí kompletní kontrolní řešení. Řada X20 umožňuje kombinovat jednotlivé komponenty mezi sebou s maximální variabilitou a kompatibilitou. Moduly X20 se skládají ze tří submodulů pro zajištění maximální snadnosti použití během celého jejich životního cyklu. Dělení na modul sběrnice, modul elektroniky a blok terminálu má několik výhod [40]:

• **Přednastavené pro různé typy strojů:** Systémové sběrnicové moduly X20 jsou základní platformou pro mnoho strojových variací. Konstrukce stroje určuje, které elektronické moduly se použijí. Software rozpozná tento návrh automaticky a zajišťuje, aby byly správné funkce poskytovány tam, kde jsou potřeba. Zacházení je tedy velice jednoduché a kompatibilita je zaručena téměř vždy

• **Konstrukce pro průmyslové kontrolní skříně:** Terminálové bloky systému X20 jsou odděleny od modulu elektroniky a samy o sobě umožňují zapojit rozvody celé skříně bez přítomnosti elektronických modulů.

• **Snadná údržba:** Moduly X20 lze snadno vyměňovat pro snadnější odstraňování problémů, neboť elektronické moduly lze vyměňovat bez přerušení i za provozu. Elektrické rozvody zůstávají díky odděleným koncovým blokům zcela totožné. Možnost rychlé výměny komponent automatizace razantně snižuje prostoje.


Obr. 16 Konstrukční řešení modulů X20 systému[40]

Popis konstrukčního řešení modulů vyobrazeném na Obr. 16.[40]:

1. **Jednoduché uchycení:** Sofistikovaný systém pro jednoduchou obsluhu a manipulaci s kolejnicovými lištami v rozvaděči nebo kontrolní skříni.
2. **Robustní modularita:** Přímé spojení mezi modulem sběrnice a blokem terminálu.
3. **Široký rozsah produktových řad**: Umožňuje optimální přizpůsobení pro každou úlohu.
4. **Elektricky izolováno**: Bezpečný provoz i za nejdrsnějších podmínek.
5. **Diagnostika**: Přímo na vrchním panelu modulu lze diagnostikovat případné potíže skrze blikající LED diody.
6. **Terminálový blok až s 16 piny:** Blok využívá maximum svých rozměrů.
7. **Terminály typu „push-in“:** Připojení je možné bez jakéhokoliv nářadí.
8. **Výsledek:** Programování podle požadavků zákazníka a bezstarostný provoz.

*Pozn.: Pojmem „pin“ se rozumí vstupní nebo výstupní kontakt, pojmem „push-in“ se rozumí pohodlný způsob připojení.*

## 3.3. Automation Studio

Vývojové prostředí Automation Studio ve verzi 4 a výše od firmy Bernecker & Rainer (B&R) je vrcholným nástrojem pro strojního a systémového inženýrství. Svým udržitelným a účinným přístupem k vývoji softwaru si dokáže udržet stoupající kvalitu, přičemž náklady na inženýrství klesají a čas na uvedení na trh se krátí – a to i se stále rostoucími nároky a složitosti produktu. Automation Studio zaručuje špičkovou kvalitu, dlouhou životnost a nízké provozní a servisní náklady [40].

Přednosti softwaru Automation Studio[40]:

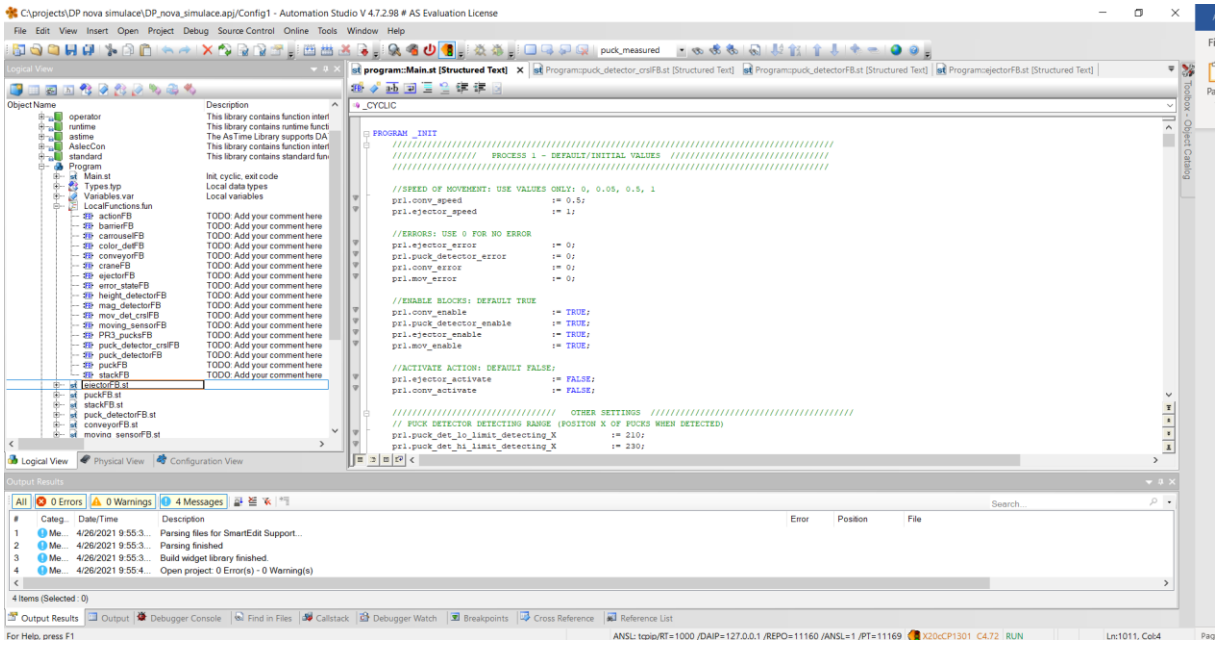
- Je jednotný programovací nástroj pro každý aspekt projektu automatizace, což minimalizuje potřeby na školení inženýrů, upevňuje celkovou integraci a odstraňuje komunikační problémy mezi inženýrskými obory.

- Urychluje čas pro uvedení na trh s automatickým generováním kódu ze strojových simulací, využívá konfiguračních dat z elektro-inženýrských softwarů a efektivně rozděluje úkoly při vývoji softwaru do modulů.

- Šetří čas a peníze zahájením vývoje softwaru před dokončením hardwaru, opakovaným použitím softwarových modulů napříč dalšími projekty a hlavně ověřováním funkčnosti pomocí simulace a uvedením do provozu modul po modulu.

- Grafická hardwarová konfigurace je plně podporována systémem a snadná softwarová modularizace jakékoliv úrovně usnadňuje vývoj strojů a systému skrze předem naprogramovanými technologickými komponenty.

Na Obr. 17 je ukázka vývojové prosředí Automation Studio ve verzi 4.7.2.98, kterou jsem použil při vývoji simulace a řídicího programu. V levém sloupci je znázorněn logický pohled na projekt, zejména funkční bloky, porměnné, knihovny, apod. V této části lze přepínat mezi fyzickým a konfiguračním pohledem. Vrchní lišta slouží k operacím jako je nahrávání programu do fyzického či virtuálního PLC, spouštění simulace, tzv. „cold“ a „warm“ restart, atd. Ve spodní části je výstupní okno s chybami, varováními a zprávami o programu. Důležitá informace se nachází v pravem dolním rohu, kde je stav jak už fyzického nebo virtálního kontroléru. K zjištění stavu hardwaru nebo softwaru slouží také „Systém Diagnostic Manager“.



Obr. 17 Ukázka vývojového prostředí Automation Studio

**System Diagnostic Manager**

Systém Diagnostic Manager (SDM) je opravdu snadný a kvalitní nástroj, který lze použít k diagnostice dění okolo kontroléru pomocí standardního webového prohlížeče. Nejsou zde vyžadovány další diagnostické nástroje. Jediným požadavkem je webový prohlížeč a připojení TCP/IP ke kontroléru [45].

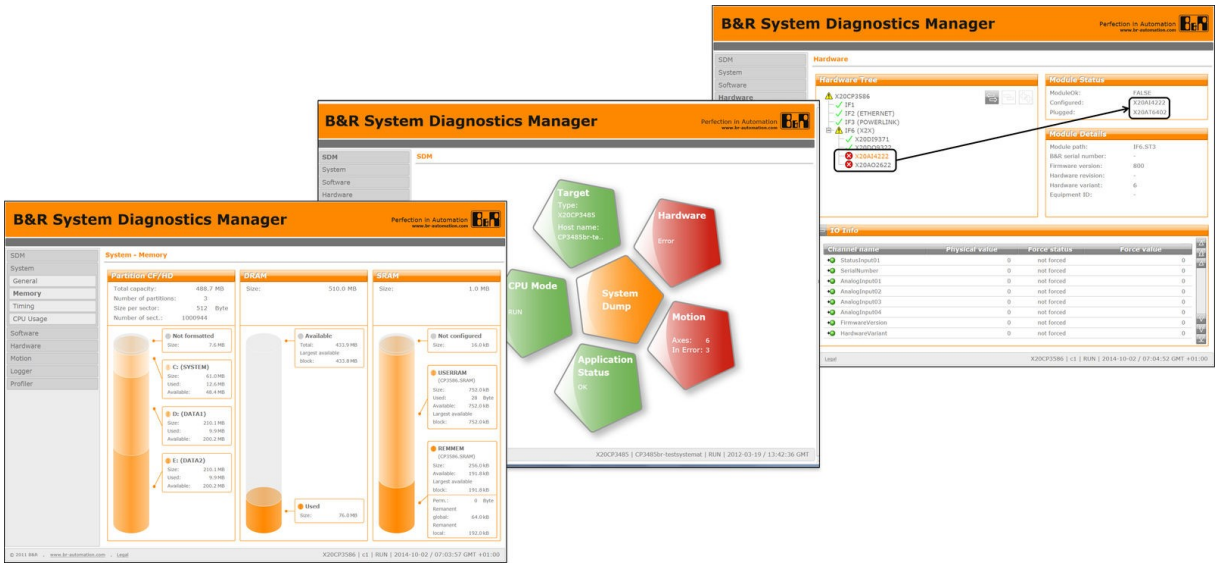
SDM poskytuje [45]:

- Hardwarovou analýzu pro zjištění potíží s konfigurací nebo hardwarem v cílovém systému
- Analýzu systémové konfigurace a runtime parametrů (např. konfigurovaná IP adresa atd.)
- Analýzu softwaru (softwarové moduly a verze v cílovém systému)
- Přístup k profileru a nahrání protokolů profileru pro další analýzu v Automation Studio

*Pozn.: Profiler je statistický nástroj, který analyzuje, kde program tráví nejvíce času*.[42]

Velmi šikovnou funkcí je detekce hardwaru do konfigurace. Stačí spojit hardware s PC skrze ethernet a v prázdném projektu, který je vytvořen v prostředí Automatio Studio, spustit funkci detektování hardwaru. Všechen připojený fyzický hardware se automaticky nakonfiguruje do projektu.

Na Obr. 18 je ukázka několika diagnostických oken, které se zobrazují přímo v prohlížeči. Jsou zde informace např. o využití CPU, paměti, teplotě, aktuálním stavu kontroléru, seznam připojených modulů a jejich funkčnost apod. Diagnostiku hardwaru i softwaru je možné průběžně ukládat do textového souboru.

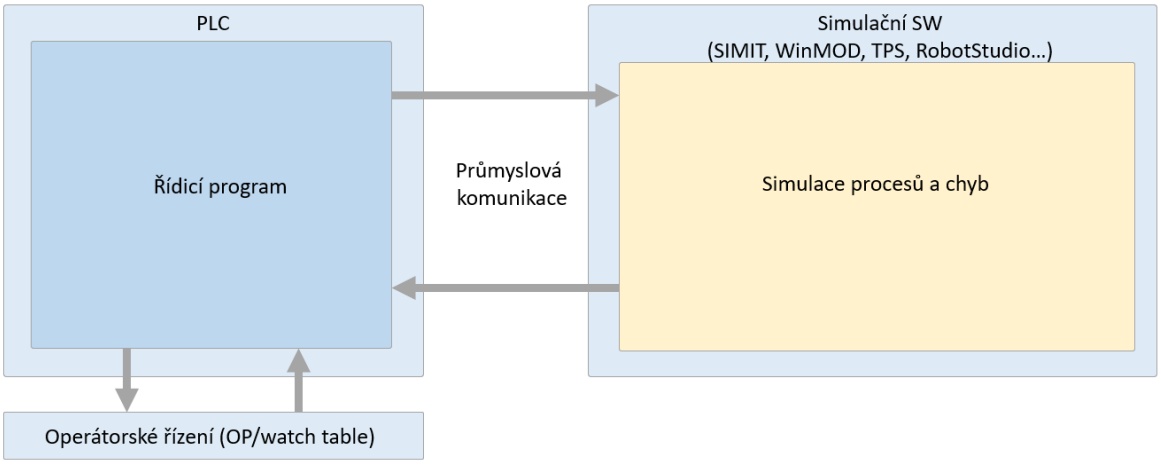



Obr. 18 Ukázka diagnostických oken SDM[43]

## 4. Možnosti simulací v prostředí Scene Viewer

Scene Viewer je vizualizační nástroj simulací od firmy B&R, který je volně dostupný bez licenčního klíče. Zde je důležité uvést, že Scene Viewer pouze vizualizuje simulační kód, který je generován jiným softwarem nebo uživatelem. To znamená, že je zajištěna pouze jednosměrná komunikace bez zpětné vazby. Jak tedy funguje Scene Viewer?

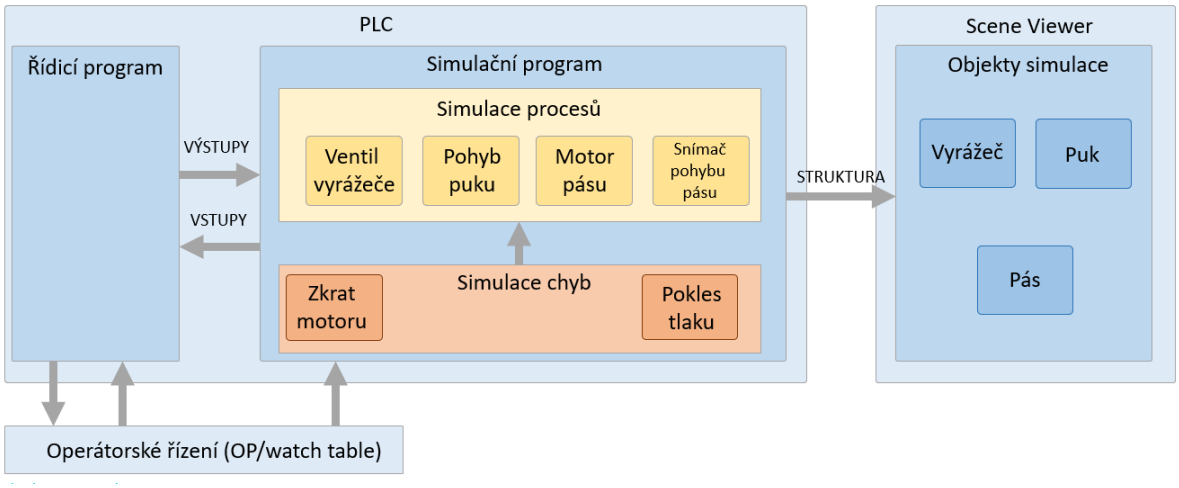
Pricnip činnosti je podněkud jednoduchý. Simulace je naprogramována na virtuálním (nebo fyzickém) PLC. Simulační blok svými výstupy reaguje na vstupy, přičemž výstupní data jsou předávána z PLC do Scene Viewer skrze OPC UA. Aby bylo možné lépe pochopit rozdíl mezi klasickou simulací a simulací v PLC s následnou vizualizací v Scene Viewer, je na Obr. 19 blokově znázorněn způsob simulace




Obr. 19 Blokové schéma simulace za využití simulačních SW

za použití simulačních SW. Skrze průmyslovou komunikaci jsou posílána data řídicím programem do simulačních SW. Z těchto simulačních SW lze čekat odezvu simulačních procesů.
Simulační SW současně slouží jako vizualizační nástroj simulace. Tímto způsobem je zajištěna virtualizace systému s externí zpětnou vazbu, což je hlavní rozdíl s porovnáním se simulací přímo v PLC.

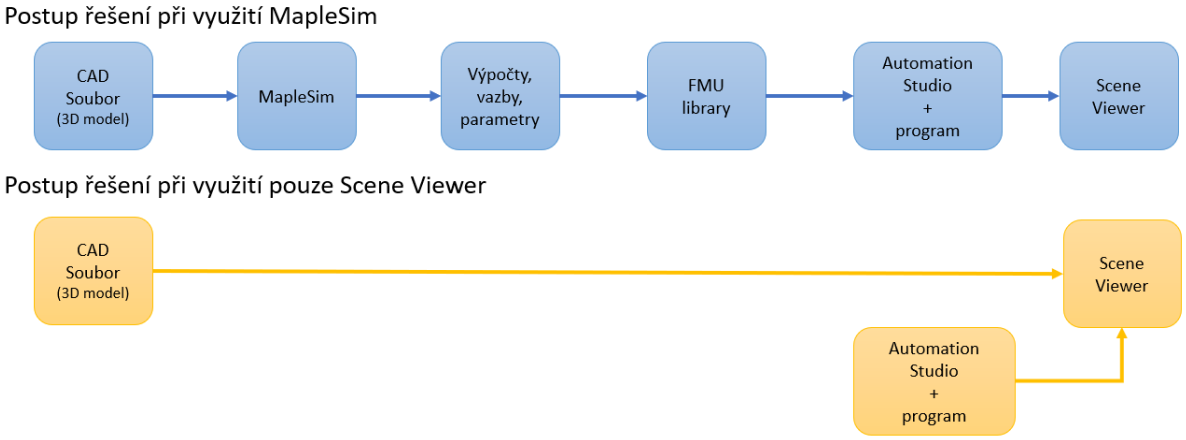
Na Obr. 20 je blokově znázorněn princip činnosti simulace, která běží přímo v PLC. Simulační program je naprogramován tak, aby výstupní data funkčních bloků jednotlivých akčních členů a procesů byla co nejvíc věrná realitě. Příkladem může být zpětná vazba motoru. Zvyšováním napětí na vstupu funkčního bloku generuje rychlost otáčení motoru na výstupu. Logika uvnitř funkčního bloku je pro programátora „black-box“, což znamená, že nezná vnitřní algoritmy a pouze sleduje výstupní data.




Obr. 20 Blokové schéma simulace v PLC

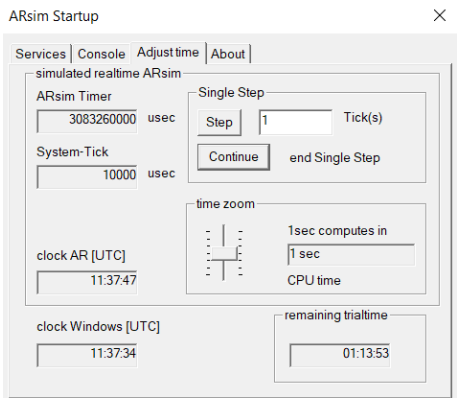
**Vytvoření simulačního programu**
Simulační program pro B&R Automation Studio nebo jeho jednotlivé části v podobě funkčních bloků se tvoří nejčastěji v softwaru **MapleSim – B&R Edition.**
MapleSim vytváří vysoce přesný dynamický model stroje vizuálně založený na datech CAD ve formátu STEP. Vzhledem k tomu, že všechny síly nebo např. točivé momenty jsou modelovány do detailu, model lze také použít jako digitální dvojče pro dimenzování součástek. MapleSim poskytuje jednoduchý přístup pro mechanické modely. Poskytuje také výběr základních komponentů v oblasti elektrotechniky, hydrauliky a termodynamiky, které **usnadňují modelování fyzikálních interakcí.**
Model itegrovaný v MapleSim lze použít jako simulační model v Automation Studio a přenést přímo do řídicího (reálného nebo simulačního) hardwaru B&R. Výsledkem je simulace hardwaru ve smyčce (HIL), kdy se chování stroje napodobuje v reálném čase, aby se vytvořilo prostředí, kde lze software stroje bezpečně testovat. V tomto prostředí lze manipulovat se všemi ovládacími parametry a také simulovat teplotní chování pohonu**. Chování digitálního dvojčete je vizualizováno živě v B&R Scene Viewer.**[46] [47]

Rozdíl mezi postupy řešení při tvorbě simulace je blokově znázorněn na Obr. 21. V případě klasického řešení, které se hojně využívá, je existující CAD soubor importován do MapleSim. Následně jsou nastaveny parametry modelu, vazby mezi jednotlivými objekty, rovnice, fyzikální zákony apod. Při tomto procesu lze využít již připravené knihovny. Jsou-li splněny všechny náležitosti modelu, přichází na řadu export tzv. FMU knihovny (functional mock-up, v překladu „funkční maketa“), která je následně importována do prostředí Automation Studio. Tato funkční maketa, mimo jiné, obsahuje simulační funkční blok vzniklého modelu a vizualizaci modelu, která je po spuštění otevřena v Scene Vieweru. Při změně vstupních parametrů simulačního funkčního bloku lze pozorovat reálné chování modelu. Při použití simulačního funkčního bloku s reálným kontrolérem vzniká digitální dvojče se simulací HIL (hardware-in-the-loop).



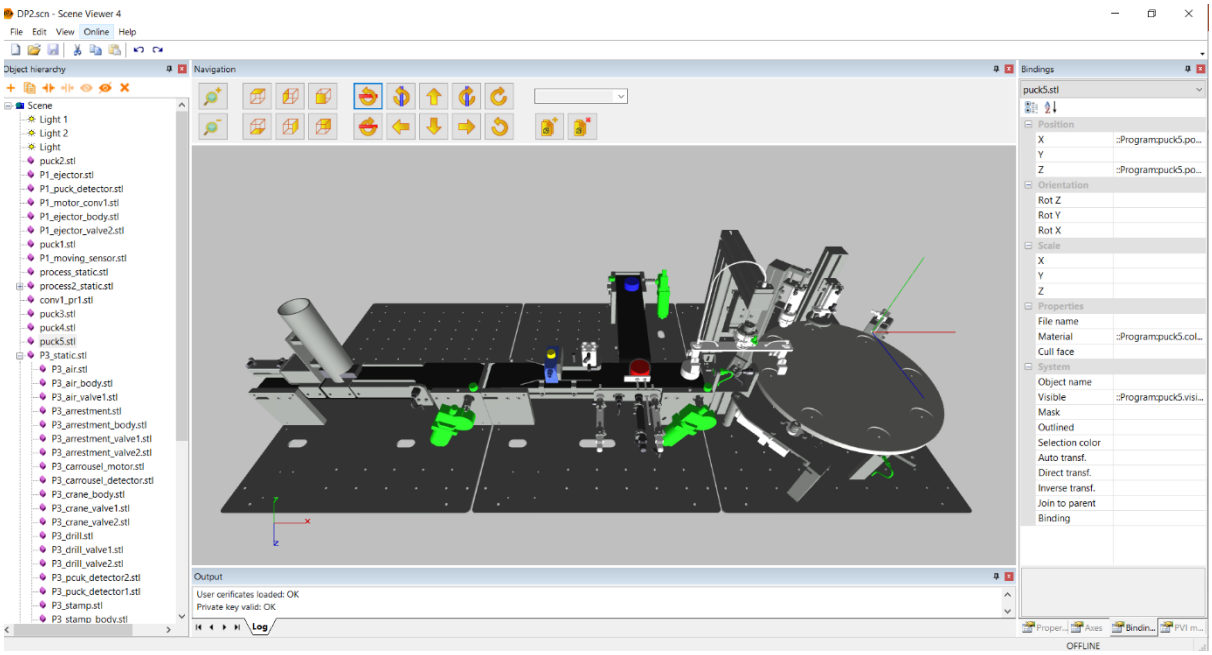

Obr. 21 Bloková schémata rozdílných postupu řešení při tvorbě simulace

Práce se SW MapleSim vyžaduje pokročilé znalosti v oboru virtuálního uvedení do provozu. Pro plnou verzi je rovněž nutná licence. Z toho důvodu jsem tento simulační nástroj nepoužil a simulační bloky v mé práci jsem tvořil sám přímo v prostředí Automation Studio, viz Obr. 21. To znamená, že mnou vytvořené simulační bloky věrně nezachycují realitu, neboť zanedbávám tření, nelinearitu, fyzikální vlastnosti apod. Při vytváření funkčních bloků jsem vycházel pouze z principu chování akčních členů a jejich potenciálních chyb. Přesný postup řešení zmiňuji dále v práci.
Plynulost simulace značně ovlivňuje velikost importovaných STL souborů. CAD soubor ve formátu STEP obsahuje mnoho zbytečných komponent, která nemají pro simulaci žádnou hodnotu, například šrouby, matky, podložky, zařízení umístěné v konstrukcích, vodiče, hadice, apod. Simulace při importu kompletního souboru neběžela plynule a byly znát trhavé pohyby. Proto jsem všechny nepotřebné komponenty smazal, včetně spodních stolů. Plynulost programu také závisí na době cyklu a na nastavení simulačního runtime (ARsim), je-li použit, viz Obr. 22.




Obr. 22 Nastavení ARsim

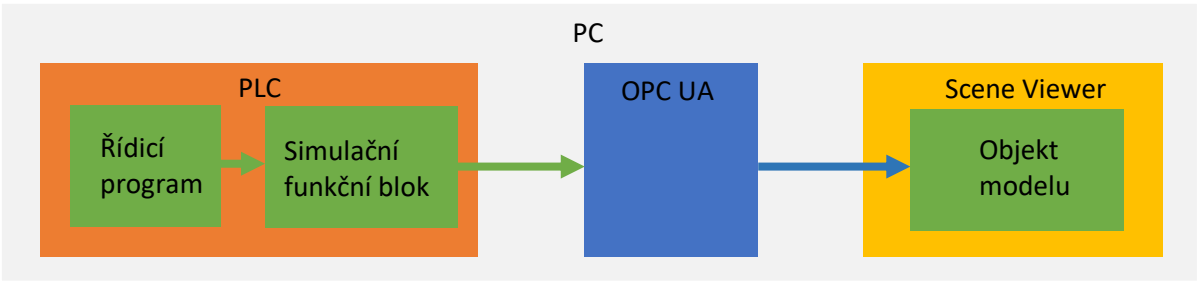
Orientace v prostředí Scene Viewer je velice jednoduchá a intuitivní. Na Obr. 23 je patrné, že se nejedná o komplexní software a disponuje pouze několika základními funkcemi. V levém sloupci je objektová hierachie. Zde jsou zobrazeny všechny objekty simulace. Mezi objekty lze vytvořit vazba (rodiče – potomek). V právem sloupci jsou zobrazeny vlastnosti konkrétního objektu. Je-li sloupec přepnutý na „binding“, lze tyto vlastnosti spojit se simulační proměnnou z Automation Studio.
Pro simulaci je využíván Scene Viewer ve verzi 4.




Obr. 23 Prostředí Scene Viewer

### 4.1.1. OPC UA

Komunikace mezi vývojovým prostředím Automation Studio a vizualizačním prostředím Scene Viewer je zajištěna pomocí OPC UA. Open Platform Communications United Architecture, zkráceně OPC UA je standard výměny dat pro průmyslovou komunikaci (komunikaci mezi stroji nebo mezi PC a stroji). Tento standard otevřeného rozhraní je nezávislý na výrobci nebo dodavateli systému aplikace, na programovacím jazyce, ve kterém byl příslušný software naprogramován ani na operačním systému, na kterém aplikace běží. [50]
Jelikož je Automation Studio i Scene Viewer od stejného výrobce, není nutné OPC UA nastavovat. Oba softwary jsou zcela připravené na spojení během několika kliknutí. Jedinou podmínkou je povolení komunikace OPC UA v konfiguraci PLC a automatické povolení proměnných, skrze které budou data přenášeny, jak je zobrazeno na Obr. 24.
Výměna dat probíha skrze proměnné datového typu struktura. K tomus slouží tzv. „binding“. Do dynamické parametrů je vepsána cesta k proměnné nebo přímo vybrána ze seznamu.



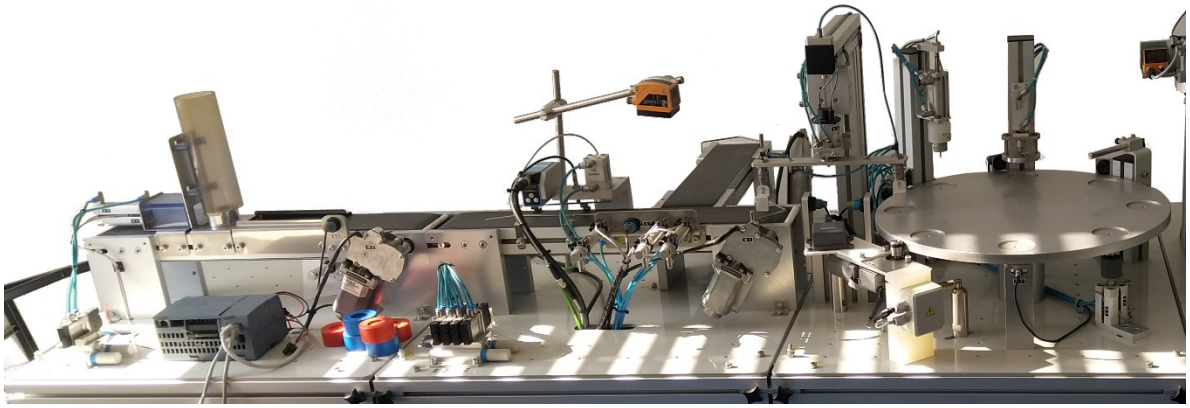

Obr. 24 Ukázka komunikace skrze OPC UA

# 5. Návrh simulace manipulační linky

Návrh simulace technologického procesu jsem tvořil na základě reálné fyzické manipulační linky, která slouží pro výukové účely na fakultě elektrotechniky a informatiky. Princip činnosti mi byl přiblížen v minulosti, kdy jsem absolvoval předmět, ve kterém studenti, včetně mne, tvořili řídicí program, ale pouze pro jedno pracoviště. Proto má analýza problematiky vycházela z výukové dokumentace, která je dostupná pro studenty FEI.

## 5.1. Stávající technologický proces

Stávající technologický proces nazvaný manipulační linka se skládá z několik zvlášť fungujících pracovišť, které na sebe navazují. Jedná se o plně funkční výukový model, který demonstruje tok materiálu a jeho testování, zpracování, třídění apod. V této práci se věnuji prvním třem pracovištím. Pracoviště jsou oddělena stoly, na kterých probíhají jednotlivé technologické procesy. K fungování akčních členů je využit mimo elektrické energie také pneumatický systém. Na Obr. 25 je fotografie stávajícího technologického procesu.



Obr. 25 Fotografie stávajícího technologického procesu

**Distribuční pracoviště**
První pracoviště je nazváno distribuční. Zde je totiž zásobník s puky, které jsou vyrážečem distribuovány na linku. Dalším komponentem je detektor přítomnosti objektu na páse (dále jen detektor puků), který se nachází zhruba v polovině dopravníkového pásu. Posledním akčním členem je motor, který pohání dopravníkový pás. Zpětná vazba pohybu pásu je zajištěna indukčním snímačem rotace.
**Testovací pracoviště**
Na dalším pracovišti probíhá testování puků. Prvním snímačem v pořadí je snímač barev, který rozezná modrou, červenou a stříbrnou barvu, přičemž disponuje také výstupním signálem pro případ, kdy nerozezná žádnou z výše jmenovaných barev. Snímač je programovatelný a lze nadefinovat i jiné barvy. Bezprostředně za snímačem barev se nachází snímač magnetických vlastností a analogový snímač výšky, který po aktivaci spouští píst až do výše puku. Aby mohl být puk v tomto místě otestován a dopravníkový pás mohl dále běžet bez omezení, je zde závora. Ta zabraňuje pukům v pohybu dále po páse. Za první závorou je opět detektor puků, který plní stejnou funkci jako detektor puků v distribučním stanovišti. V případě, že puky neprošly testy a je nutné je vyřadit z toku materiálu, je zde vyrážeč spolu s druhou závorou bezprostředně za ním. Puk je vyrážen na přilehlý dopravníkový pás, kde svůj cyklus končí a dále v lince nepokračuje. Oba dopravníkové pásy mají vlastní motor a indukční snímač rotace.

**Procesní pracoviště**
Třetím pracovištěm v pořadí je pracoviště procesní. Místo dopravníkového pásu je zde karusel s otvory pro celkem osm puků. Karusel je poháněn vlastním motorem a pohyb je sledován indukčním snímače rotace, přičemž pohyb zaznamenává při každém otočení o 45 °. Pro aretaci karuselu slouží mechanismus, jehož koncová jamka po vyražení zablokuje pohyb karuselu. Tento manévr lze provádět pouze při rotaci o 45 °C vzhledem k výchozí poloze, tedy v ten moment, kdy je aktivní indukční snímač rotace. K přemístění puků z předešlého pracoviště slouží zařízení, připomínající chapadla, která fungují obousměrně. K detekci puku na konci dopravníkového pásu testovacího pracoviště slouží opět detektor puků, který se nachází i na straně karuselu. Dále se nachází na pracovišti akční členy pro vrtání, ražbu a ofukování, které jsou rozmístěny po obvodu středů otvorů pro puky posunuty vždy o 45 °. Vrtačka a razič disponují také mechanickým zařízením pro pohyb nahoru a dolů.

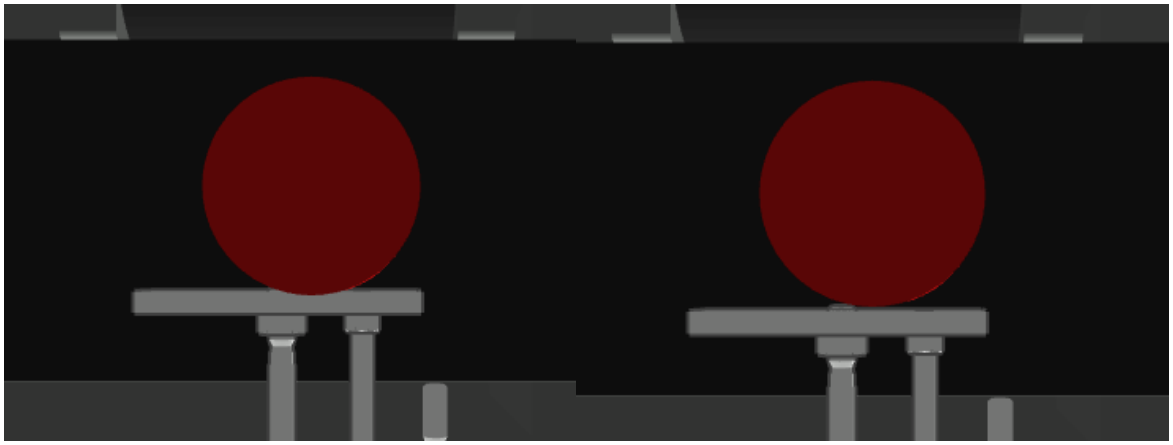
## 5.2. Simulace technologického procesu

Před samotným programováním jsem provedl jakousi analýzu, která zahrnovala závislost mezi jednotlivými simulovanými objekty a budoucí rozhraní modelu simulace. Podmínkou při vytváření rozhraní pro simulační akční členy bylo dodržení zachování principu zápisu a čtení jako u fyzické linky. To znamená, že rozhraní simulačního prostředí bude totožné s rozhraním fyzické linky, tudíž řídicí program pro simulaci musí být „teoreticky“ stejný, jako řídicí program pro fyzickou linku. Na teoretické rovině se pohybuji proto, že v simulaci zanedbávám mnoho aspektů jako například fyzikální zákony.

**Simulace směru pohybu**
Simulace pohybu je řešena diskrétním způsobem a vychází se souřadnicového systému Scene Viewer. Proměnná, která simuluje pohyb, je připojena k příslušné souřadnici objektu (pozice X,Y,Z nebo rotace podle osy X,Y,Z) právě v prostředí Scene Viewer. Inkrementací nebo dekrementací proměnných napojených na souřadnicový systém objektu lze snadno realizovat pohyb. V každém cyklu se objekt posune o určitou hodnotu. Čím nižší hodnota, tím je pomalejší, ale za to plynulejší pohyb.

**Simulace rychlosti pohybu**
Rychlost pohybu je přímo úměrná době cyklu a hodnotě, o kterou je poziční proměnná inkrementována nebo dekrementována. Aby byl model univerzálnější, simulační bloky zahrnující pohyb, jsem naprogramoval tak, že „rychlost“ je vstupním parametrem pro určitou skupinu funkčních bloků. V praxi tedy lze měnit rychlosti jednotlivých objektů linky, které vykonávají pohyb. Taková možnost u fyzické linky není, proto mají simulační objekty zahrnující pohyb nastavenou výchozí hodnotu rychlosti, která zhruba odpovídá skutečnosti. Jednotky rychlosti jsou bezrozměrné, neboť parametr „rychlost“ je ta hodnota, o kterou je poziční proměnná inkrementována nebo dekrementována. Velice důležité je zmínit, že parametr „rychlost“ nelze zvolit libovolně. Důvod je zcela prostý. Poziční parametry a podmínky ostatních objektů jsou vždy celá. Příkladem je výchozí a koncová poloha simulačního funkčního bloku vyrážeče, kde vyrážecí objekt setrvává v pozici 0 a při maximálním vyražení setrvává v pozici 80. Po aktivaci se vyrážeč dotýká puku až při pozici 20, přičemž v tento moment se začne puk pohybovat stejným směrem i rychlostí jako vyrážeč. Bude-li hodnota parametru rychlosti zvolena např. 0,74, bude pozice vyrážeče po 27 cyklech přesně 19,98 (pozice + 0,74 v každém cyklu) a po 28 cyklu se bude rovnat hodnotě 20,72. Simulační blok puku je však naprogramován tak, že se puk začne pohybovat v momentě, kdy se vyrážeč puku začne dotýkat přesně v pozici 20. Díky nespojitosti pohybu dochází k těmto problémům a simulace nebude fungovat správně. Na Obr. 26 je zřetelný rozdíl správné a chybné simulace pohybu. V pravé části je pohled shora na puk a vyrážeče. Vyrážeč se v aktuální pozici dotknul puku a v ten moment puk reaguje na jeho pohyb. V levé části je nabídnut stejný pohled na stejnou situaci, avšak s nesprávným parametrem rychlosti. Vyrážeč se překrývá s pukem, což znamená, že puk se nezačal pohybovat v místě doteku s vyrážečem, což je

právě způsobeno nesprávně zvoleným parametrem rychlosti. Jak tedy parametr rychlosti zvolit? Vždy takový, aby součet těchto hodnot obsahoval všechna celá čísla v rozsahu pohybu z výchozí do koncové polohy a vždy vyšší než -1 včetně a nižší než 1 včetně. Doporučené hodnoty pro parametr rychlosti jsou 0,05, 0,5 a 1. Nabízí se také možnost 0,25, neboť splňuje výše uvedené podmínky. Z nepochopitelného důvodu při inkrementaci nebo dekrementaci se hodnoty zobrazují s nesmyslně malými desetinnými čísly. Po čtyřech cyklech inkrementování při zvolené rychlosti 0,25 by se měla výsledná hodnota pozice rovnat 1. Automation Studio však zobrazuje hodnoty jako např. 1,032656. Kde tkví problém, jsem nezjistil.



Obr. 26 Ukázka chybné a správné simulace pohybu

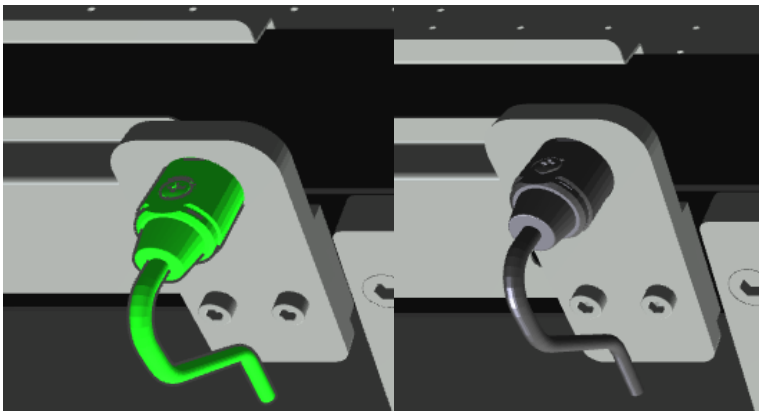
**Simulace chyb**

Vytvořit úmyslnou poruchu na fyzické lince je takřka nemožné. Do simulace jsem proto implementoval možnost vytvořit poruchy akčních členů a snímačů. Simulační bloky objektů, u kterých mohou nastat poruchy, obsahují vstup pro definici chyby. K tomu se váže možnost testování havarijních stavů. Více informací o možnostech simulací chyb a testování havarijních stavů naleznete v kapitole 6..

**Simulace funkčnosti**

Simulační objekty, které nevykazují pohyb při své činnosti, lze jen těžko hodnotit, zda-li fungují správně. Proto jsem implementoval do některých simulačních bloků změnu barvy objektů právě podle aktuálně vykonávaného procesu. Tímto je uživateli poskytnuta vizuální kontrola funkčnosti.

Na Obr. 27 je znázorněna změna barvy detektoruku puku při detekci. Tímto způsobem je kontrola funkčnosti veli snadná a přehledná a lze ji sledovaz za běhu programu. Mění se také barvy aktivních ventilů. Nefunkční komponenty svítí červeně.



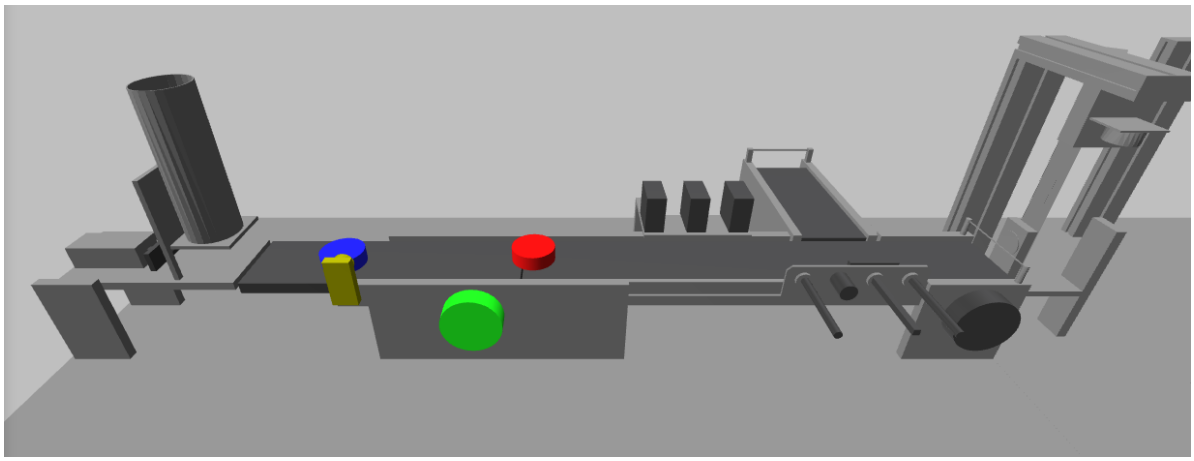
Obr. 27 Změna barvy detektoru puku při detekci

## 5.3. Digitální dvojče manipulační linky

Těžko říct, zda-li je vhodné pojmenovat můj model digitálním dvojčetem, neboť sice kopíruje vzhled, ale věrné zachycení reálného fungování zde simulováno není, jak jsem již zmínil výše. I přesto si troufám tvrdit, že můj model slouží jako plnohodnotná náhrada pro demonstrační a výukové účely. Jedná se tedy o koncept digitálního dvojčete velice nízké úrovně.

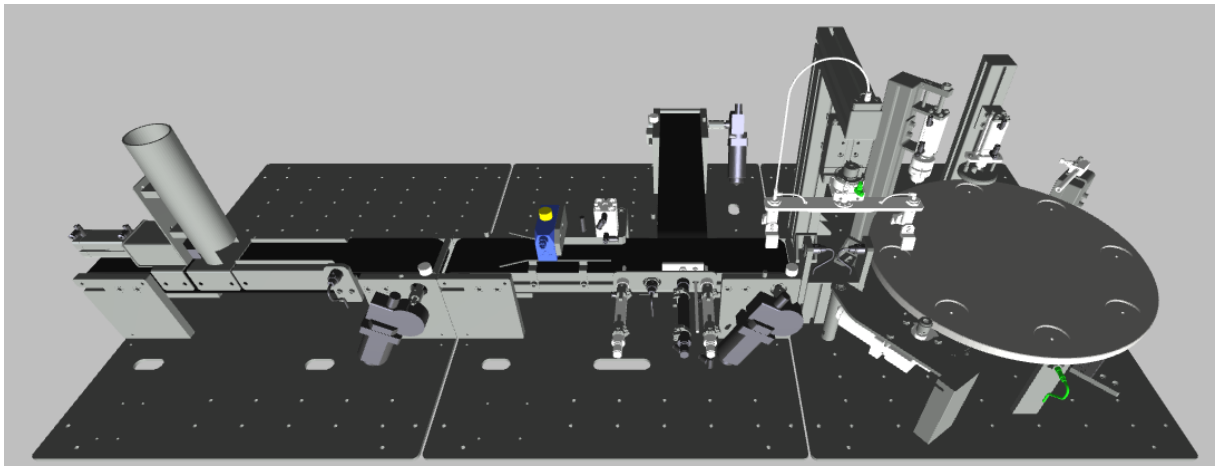
### 5.3.1. 3D model

V prvotní fázi diplomové práce jsem se snažil o import CAD souborů manipulační linky ve formátu STEP do prostředí Scene Viewer. To se mi však nedařilo a linku jsem začal modelovat sám v softwaru SketchUp, což je velice jednoduchý, primitivní a intuitivní nástroj pro 3D modelování objektů. Jednotlivé objekty jsem exportoval do formátu STL a následně importoval do prostředí Scene Viewer. Na princip činnosti simulační linky to nemělo vůbec žádný vliv, akorát můj model se vzhledově lišil od fyzické linky. Na Obr. 28 je vyobrazen nedokončený model manipulační linky, vytvořený pomocí modelovacího softwaru SketchUp.



Obr. 28 Návrh modelu manipulační linky v softwaru SketchUp

Během vypracovávání diplomové práce mi bylo technikem podpory pro B&R objasněno, jak lze importovat CAD soubory ve formátu STEP do softwaru Scene Viewer. Model manipulační linky ve fomátu STEP mi byl poskytnut z výukových materiálů školy. Soubor jako celek musí být



Obr. 29 Digitální dvojče manipulační linky v prostředí Scene Viewer

naimportován do softwaru podporující CAD soubory, v mém případě jsem použil FreeCad ve verzi 0.18, který je volně dostupný. Po importu je nutné rozdělit model na statickou a dynamickou část.

**Statická část modelu**

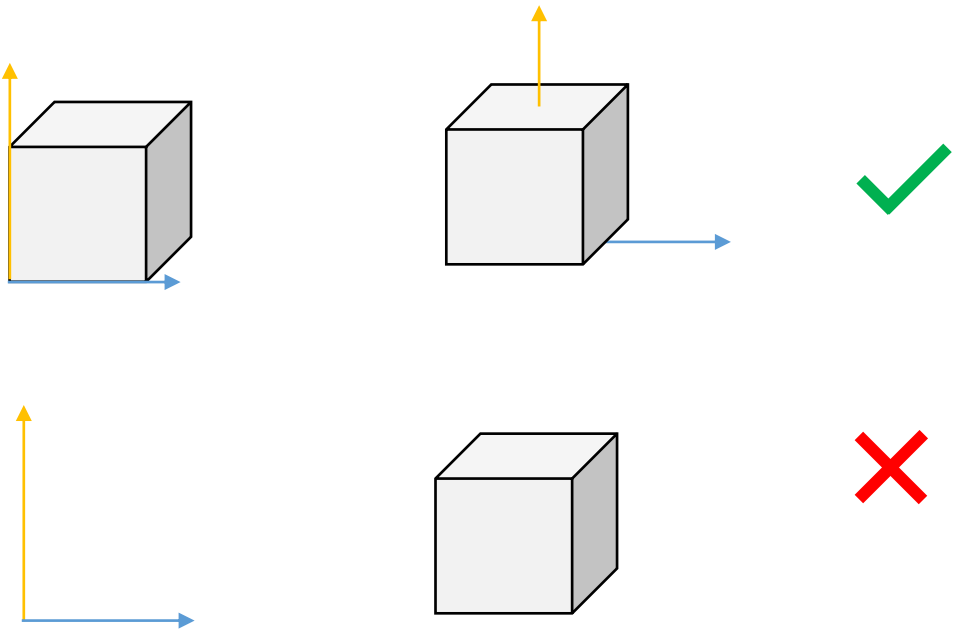
Statickou částí modelu se rozumí ty objekty, které nebudou napojeny na žádnou simulační ani řídicí proměnnou, tedy nebudou měnit své parametry po celou dobu simulace. To znamená, že jsem musel odstranit naprosto veškeré dynamické objekty. Pro každý model je jeden CAD soubor, přičemž každé pracoviště je složeno z několika stovek, možná i tisíců objektů (včetně šroubů, podložek, matek apod.). Po smazání objektů, které za běhu simulace nemění své parametry, jsem provedl export do STL, který je již kompatibilní se softwarem Scene Viewer.

**Dynamická část modelu**

Při exportu dynamických objektů jsem postupoval naprosto stejně s tím rozdílem, že statická část byla exportována jako celek, avšak dynamické objekty musejí být exportovány zvlášť, a to včetně jejich sub-objektů, jako např. ventily, aby mohla být zajištěna změna barev pouze pro daný objekt a ne dynamický celek.

**Souřadnicový systém modelu**

Pro simulovaní pohybu je nutné znát souřadnicový systém modelu a výchozí hodnoty všech dynamických objektů. Bohužel, exportovaný model v STL formátu sice má téměř všechny výchozí poziec a rotace objektů v nule, ale jako celek má počátek někde naprosto mimo relevantní čísla (v praxi stovky až tisíce bodů rozdílných od nuly, které jsem musel najít pro některé funkce). Na otázku, proč má vyexportovaný model jako celek počátek jinde než v nule i přesto, že v softwaru FreeCAD byly základní souřadnice všech objektů nastaveny na nule, mi nedokázal odpovědět ani technik podpory pro B&R. Export modelu prováděn ze softwaru SketchUp měl počátek tam, jak byl navolen



Obr. 30 Správné a chybné umístění objektu v souřadnicovém systému

před exportem. Tudíž chyba by neměla být způsobena exportem.

Pokud bych v modelu pracoval pouze s pozicemi, souřadnicový systém celku by nebyl problém, protože jeho jednotlivé objekty mají výchozí polohy v nule. Problém nastal, když jsem chtěl implementovat rotaci objektu. Konkrétně je simulace pohybu v podobě rotace implementována na karuselu, chapadlech a uchopeném puku. Aby bylo možné objekt rotovat kolem osy s počátkem v jeho středu, je nutné mít souřadnicový systém rovněž ve středu objektu.

Problém se souřadnicovým systémem se dá řešit přímo v prostředí Scene Viewer. Má několik základních objektů, které lze přidat do hiearchie objektů). Jedním z nich je „Transform coordinate systém“, který lze přeložit jako transformovaný koordinovaný systém, jinými slovy nový souřadnicový systém. Ten se vkládá přímo do scény s výchozími hodnotami v nule. Aby byla zajištěna správná simulace rotace karuselu, je nutné umístit nový souřadnicový systém tak, aby začínal přesně v jeho středu. Při modelování a hledání těchto souřadnic jsem se zhruba dostal do středu karuselu a následně i do středu chapadel a přenášeného puku. Simulační proměnná rotace je tedy připojena na nový transformační systém, s nímž mají pohybující se objekty vytvořenou vazbu.

## 5.4. Simulační funkční bloky

Jádro simulace jsou simulační funkční bloky. Při návrhu a implementace simulačních funkčních bloků jsem se snažil o to, aby jejich funkce dokázala co nejvěrněji zachytit reálné chování fyzických objektů a byla ekvivalentem hardwaru. Důležité je si uvědomit, že mezi simulačními funkčními bloky je jakási závislost a k tomu musí být přizpůsobené i jejich rozhraní. Simulačních funkčních bloků je celkem 17, přičemž řada z nich je použita opětovně. Simulační bloky obsahují jak algoritmus pro správné fungování, tak pro simulaci poruch a chyb. Testování všech funkčních bloku bylo opravdu časově náročné a nebyl v mých silách otestovat všechny možné kombinace, které na lince mohou nastat. Proto je velice pravděpodobné, že v některých situacích se simulace nebude chovat správně. Jedná se zejména o algoritmy podmínek, např. otevření chapadel během přemísťování puku z pracoviště dvě na karusel. V takové situaci není simulován pád puku.

**Vyrážeč**

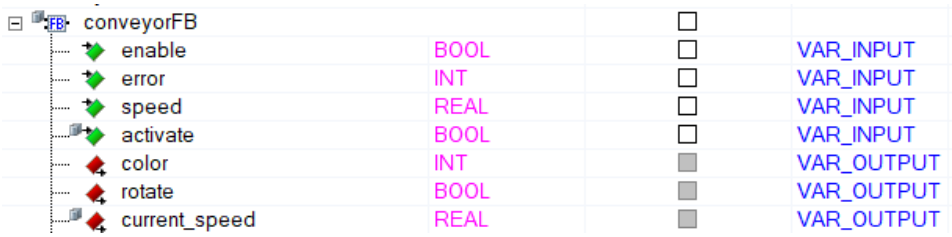
Vyrážeč, jehož hlavním úkolem je mechanický pohyb po přímce, má jednoduchou logiku. Vstupními parametry se definuje výchozí a koncová poloha, tedy poloha odkud kam bude vyrážeč svůj pohyb vykonávat. Výstupem je tedy změna pozice, barva a snímače koncových poloh (vysunut, zasunut). Na Obr. 31 je seznam vstupů a výstupů funkčního bloku pro vyrážeč. Název funkčního bloku je „ejectorFB“.

| Name | Type | & Reference | Scope |
|---|---|---|---|
| ejectorFB | | ☐ | |
| enable | BOOL | ☐ | VAR_INPUT |
| error | INT | ☐ | VAR_INPUT |
| speed | REAL | ☐ | VAR_INPUT |
| activate | BOOL | ☐ | VAR_INPUT |
| end_position | REAL | ☐ | VAR_INPUT |
| start_position | REAL | ☐ | VAR_INPUT |
| position | REAL | ☐ | VAR_OUTPUT |
| ejector_ejected | BOOL | ☐ | VAR_OUTPUT |
| ejector_rdy_to_eject | BOOL | ☐ | VAR_OUTPUT |
| current_speed | REAL | ☐ | VAR_OUTPUT |
| body_color | INT | ☐ | VAR_OUTPUT |
| valve1_color | INT | ☐ | VAR_OUTPUT |
| valve2_color | INT | ☐ | VAR_OUTPUT |

Obr. 31 Vstupy a výstupy funkčního bloku pro vyrážeč

**Dopravníkový pás**

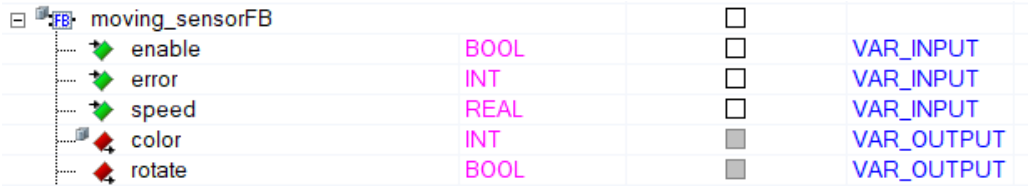
Dopravníkový nesimuluje pohyb pásů v simulaci, nýbrž jen aktivaci motoru. Je-li aktivován dopravníkový pás, výstupem je aktuální rychlost, barva pro vizualizační kontrolu a signál pro indukční snímač pohybu. Na Obr. 32 je seznam vstupů a výstupů funkčního bloku pro doprávníkový pás. Název funkčního bloku je „conveyorFB“




Obr. 32 Vstupy a výstupy funkčního bloku pro dopravníkový pás

**Indukční snímač pohybu**

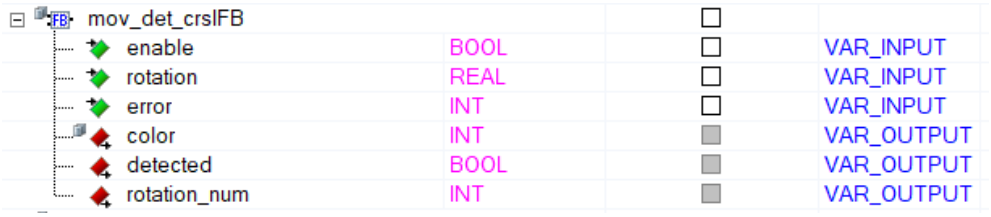
Indukční snímač pohybu, přesněji rotace hřídele motoru a pásu, je naprogramován tak, aby jeho výstup při pohybu pásu pulzoval. Děje se tomu tak i ve skutečnosti, ale mnohem rychleji. Pulzující výstup není pímo úměrný rychlosti pásu, ale je nastaven na 500 ms a to z důvodu přehldnější vizuální kontrolu pozorovatale, neboť změna barvy v tomto intervale indikuje pohyb. Kratší intervaly pulzování by mohly být pro pozorovatele matoucí a nepřehledné. Na Obr. 33 je seznam vstupů a výstupů funkčního bloku pro indukční snímač pohybu. Název funkčního bloku je „moving_sensorFB“.




Obr. 33 Vstupy a výstupy funkčního bloku pro indukční snímač pohybu

**Indukční snímač pohybu karuselu**

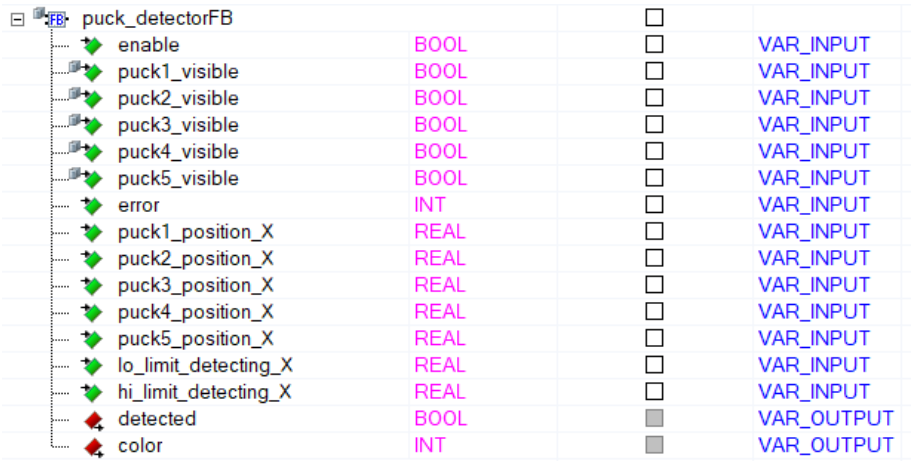
Indukční snímač pohybu dopravníkových pásu funguje reálně stejně jako indukční snímač pohybu karuselu, ale v simulaci jsem musel použít jiný funkční blok s upraveným algoritmem. Důvodem je, že indukční snímač pohybu karuselu indikuje změnu polohy vždy, když je rotace karusele vzhledem k výchozí poloze rozdílna o násobek 45 ( tedy o 45 °C). Principiálně fungují stejně. Na Obr. 34 je seznam vstupů a výstupů funkčního bloku pro indukční snimač pohybu karuselu. Název funkčního bloku je „mov_det_crslFB“.




Obr. 34 Vstupy a výstupy indukčního funkčního bloku pro indukční snímač pohybu karuselu

**Detektor puků**

Detektor přítomnsoti objektů na páse (zkráceně detektor puků) snímá pouze objekty před sebou. Vstupními parametry lze nastavit rozsah šířky snímaní. Výstupem je detekce puku a barva snímače.
Na Obr. 35 je seznam vstupů a výstupů funkčního bloku pro detektor puků. Název funkčního bloku je „puck_detectorFB“

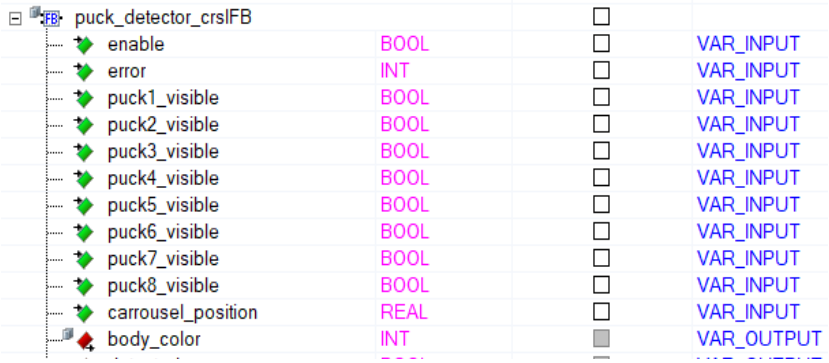


| | | | |
|---|---|---|---|
| puck_detectorFB | | | |
| enable | BOOL | | VAR_INPUT |
| puck1_visible | BOOL | | VAR_INPUT |
| puck2_visible | BOOL | | VAR_INPUT |
| puck3_visible | BOOL | | VAR_INPUT |
| puck4_visible | BOOL | | VAR_INPUT |
| puck5_visible | BOOL | | VAR_INPUT |
| error | INT | | VAR_INPUT |
| puck1_position_X | REAL | | VAR_INPUT |
| puck2_position_X | REAL | | VAR_INPUT |
| puck3_position_X | REAL | | VAR_INPUT |
| puck4_position_X | REAL | | VAR_INPUT |
| puck5_position_X | REAL | | VAR_INPUT |
| lo_limit_detecting_X | REAL | | VAR_INPUT |
| hi_limit_detecting_X | REAL | | VAR_INPUT |
| detected | BOOL | | VAR_OUTPUT |
| color | INT | | VAR_OUTPUT |

Obr. 35 Vstupy a výstupy funkčního bloku pro detektor puků

**Detektor puků karuselu**

Detektor puků karuselu reálně funguje stejně jako detektor puků na pásech, ale v simulaci jsem musel použít jiný simulační blok s upraveným algoritmem. Detektor puků na páse čte pozici puků, ale detektor puků karuselu čte rotaci puků, přesněji čte rotaci karuselu, neboť puky jsou umístěny ve stejném souřadnicovém systému jako karusel a jsou jeho potomky. Algoritmus tedy čte data z karuselu a v určitých intervalech rotace (ty intervaly, kde jsou jamky pro puky) detekuje puk, ale jen za podmínek, že jeho viditelnost v logické jedničce, tedy je přítomen v příslušné jamce. Na Obr. 36 je seznam vstupů a výstupů funkčního bloku pro detektor puků karuselu. Název funkčního bloku je „puck_detector_crslFB“.



| | | | |
|---|---|---|---|
| puck_detector_crslFB | | | |
| enable | BOOL | | VAR_INPUT |
| error | INT | | VAR_INPUT |
| puck1_visible | BOOL | | VAR_INPUT |
| puck2_visible | BOOL | | VAR_INPUT |
| puck3_visible | BOOL | | VAR_INPUT |
| puck4_visible | BOOL | | VAR_INPUT |
| puck5_visible | BOOL | | VAR_INPUT |
| puck6_visible | BOOL | | VAR_INPUT |
| puck7_visible | BOOL | | VAR_INPUT |
| puck8_visible | BOOL | | VAR_INPUT |
| carrousel_position | REAL | | VAR_INPUT |
| body_color | INT | | VAR_OUTPUT |

Obr. 36 Vstupy a výstupy funkčního bloku pro detektor puků karuselu

**Puk**

Objekt v podobě puků má také svůj funkční blok, který simuluje pohyb puků po lince. Vstupní parametry „define_hole“ a „define_height“ nejsou využity, ale jsou připraveny při případné rozšíření definice puků. Výstupem je viditelnost, výška, pozice, rotace a barva. Na Obr. 37 je seznam vstupů a výstupů funkčního bloku pro puk. Název funkčního bloku je „puckFB“.

| | | | |
|---|---|---|---|
| puckFB | | ☐ | |
| speed | REAL | ☐ | VAR_INPUT |
| ejector_pr1_position | REAL | ☐ | VAR_INPUT |
| ejector_pr2_position | REAL | ☐ | VAR_INPUT |
| visibility_enabled | BOOL | ☐ | VAR_INPUT |
| define_hole | BOOL | ☐ | VAR_INPUT |
| define_height | REAL | ☐ | VAR_INPUT |
| define_color | INT | ☐ | VAR_INPUT |
| blocked1 | BOOL | ☐ | VAR_INPUT |
| blocked2 | BOOL | ☐ | VAR_INPUT |
| visibility | BOOL | ☐ | VAR_OUTPUT |
| hole | BOOL | ☐ | VAR_OUTPUT |
| height | REAL | ☐ | VAR_OUTPUT |
| color | INT | ☐ | VAR_OUTPUT |
| position_Z | REAL | ☐ | VAR_OUTPUT |
| position_Y | REAL | ☐ | VAR_OUTPUT |
| position_X | REAL | ☐ | VAR_OUTPUT |
| rotation_Z | REAL | ☐ | VAR_OUTPUT |
| rotation_Y | REAL | ☐ | VAR_OUTPUT |
| rotation_X | REAL | ☐ | VAR_OUTPUT |

Obr. 37 Vstupy a výstupy funkčního bloku pro puk

**Snímač barev**

Příslušný výstup snímače barev je aktivován, pokud dojde k detekci dané barvy, v tomto případě modré, červené, nebo stříbrné (hliníkové). Je-li detekovaný puk jiné barvy, což lze nadefinovat ve funkčním bloku „stackFB“ , snímač barev vyhodnotí tuto barvu jako nedefinovanou. Výstup „color_informative“ vždy ponechá poslední detekovanou barvu a data jsou předána do vizualizace, kde pozorovatel zjistí, zda-li snímač funguje správně. Na Obr. 38 je seznam vstupů a výstupů funkčního bloku pro snímač barev. Název funkčního bloku je „color_detFB“.

| | | | |
|---|---|---|---|
| color_detFB | | ☐ | |
| enable | BOOL | ☐ | VAR_INPUT |
| error | INT | ☐ | VAR_INPUT |
| puck1_color | INT | ☐ | VAR_INPUT |
| puck2_color | INT | ☐ | VAR_INPUT |
| puck3_color | INT | ☐ | VAR_INPUT |
| puck4_color | INT | ☐ | VAR_INPUT |
| puck5_color | INT | ☐ | VAR_INPUT |
| puck1_pos_X | REAL | ☐ | VAR_INPUT |
| puck2_pos_X | REAL | ☐ | VAR_INPUT |
| puck3_pos_X | REAL | ☐ | VAR_INPUT |
| puck4_pos_X | REAL | ☐ | VAR_INPUT |
| puck5_pos_X | REAL | ☐ | VAR_INPUT |
| color_red | BOOL | ☐ | VAR_OUTPUT |
| color_blue | BOOL | ☐ | VAR_OUTPUT |
| color_aluminium | BOOL | ☐ | VAR_OUTPUT |
| color_undefined | BOOL | ☐ | VAR_OUTPUT |
| color | INT | ☐ | VAR_OUTPUT |
| color_informative | INT | ☐ | VAR_OUTPUT |

Obr. 38 Vstupy a výstupy funkčního bloku pro snímač barev

**Snímač magnetických vlastností**

Simulační snímač magnetických vlastností je lehce upraven oproti reálnému snímači. Nesnímá magentické vlastnosti, ale jen barvu puku. Předpokládá se, že barevné puky jsou plastové, tedy nevykazují magnetické vlastnosti a puky stříbré (hliníkové) magnetické vlastnosti vykazovat budou vždy. Proto je výstup snímače aktivní pouze tehdy, kdy snímá stříbrný (hlinikový) puk. Výstupem je také opět i barva, která detekuje aktivní výstup. Na Obr. 39 je seznam vstupů a výstupů funkčního bloku pro snímač magnetických vlastnostní. Název funkčního bloku je „mag_detectorFB“.

| | | | |
|---|---|---|---|
| mag_detectorFB | | ☐ | |
| enable | BOOL | ☐ | VAR_INPUT |
| error | INT | ☐ | VAR_INPUT |
| puck1_material | INT | ☐ | VAR_INPUT |
| puck2_material | INT | ☐ | VAR_INPUT |
| puck3_material | INT | ☐ | VAR_INPUT |
| puck4_material | INT | ☐ | VAR_INPUT |
| puck5_material | INT | ☐ | VAR_INPUT |
| puck1_position_X | REAL | ☐ | VAR_INPUT |
| puck2_position_X | REAL | ☐ | VAR_INPUT |
| puck3_position_X | REAL | ☐ | VAR_INPUT |
| puck4_position_X | REAL | ☐ | VAR_INPUT |
| puck5_position_X | REAL | ☐ | VAR_INPUT |
| mag_detected | BOOL | ☐ | VAR_OUTPUT |
| color | INT | ☐ | VAR_OUTPUT |

Obr. 39 Vstupy a výstupy funkčního bloku pro snímač magnetických vlastností

**Snímač výšky**

Snímač výšky pracuje na principu pístu, který je vyražen až do výše puku. Pro účel simulace je vynechán proces měřením zmíněným pístem a zobrazuje se pouze výsledná hodnota v rozmezí 0 až 10 V po krátkém intervalu měření, který lze definovat parametrem. Na Obr. 40 je seznam vstupů a výstupů funkčního o bloku pro snímač výšky. Název funkčního bloku je „height_detectorFB“.

| | | | |
|---|---|---|---|
| height_detectorFB | | ☐ | |
| enable | BOOL | ☐ | VAR_INPUT |
| activate | BOOL | ☐ | VAR_INPUT |
| error | INT | ☐ | VAR_INPUT |
| puck1_position_X | REAL | ☐ | VAR_INPUT |
| puck2_position_X | REAL | ☐ | VAR_INPUT |
| puck3_position_X | REAL | ☐ | VAR_INPUT |
| puck4_position_X | REAL | ☐ | VAR_INPUT |
| puck5_position_X | REAL | ☐ | VAR_INPUT |
| time_to_detect | TIME | ☐ | VAR_INPUT |
| height | REAL | ☐ | VAR_OUTPUT |
| detector_stable | BOOL | ☐ | VAR_OUTPUT |
| color | INT | ☐ | VAR_OUTPUT |
| valve1_color | INT | ☐ | VAR_OUTPUT |
| valve2_color | INT | ☐ | VAR_OUTPUT |

Obr. 40 Vstupy a výstupy funkčního bloku pro snímač výšky

**Simulace procesu**

Některé akční členy vykazují funkci, která při svém výkonu vydává zvuk. Pro tyto účely slouží funkční bloku „actionFB“. Namísto zvuku mění barvu členu. Například při ofukování puku se zbarví hadice se vzduchem do zelena. Takto je ošetřena vizuální kontrola funkčnosti. Na Obr. 41 je seznam vstupů a výstupů funkčního bloku pro simulaci procesu.

| | | | |
|---|---|---|---|
| actionFB | | ☐ | |
| enable | BOOL | ☐ | VAR_INPUT |
| error | INT | ☐ | VAR_INPUT |
| activate | BOOL | ☐ | VAR_INPUT |
| color | INT | ☐ | VAR_OUTPUT |

Obr. 41 Vstupy a výstupy funkčního bloku pro simulaci procesů

**Chapadla**

Celý mechanizmus pro manipulaci s puky z pracoviště dvě na pracoviště tři můžeme nazvat chapadla. Tento název se používá i při výukových činnostech. Výstup tohoto bloku ovládá pohyb a uchopení včetně změny barev ventilů. Vstupními parametry je nastavena výchozí a koncová poloha pozice a rotace. Na Obr. 42 je seznam vstupů a výstupů funkčního bloku pro chapadla. Název funkčního bloku je „craneFB“.

| | | | |
|---|---|---|---|
| craneFB | | ☐ | |
| activate_grip | BOOL | ☐ | VAR_INPUT |
| end_rotation | REAL | ☐ | VAR_INPUT |
| start_rotation | REAL | ☐ | VAR_INPUT |
| activate_position | BOOL | ☐ | VAR_INPUT |
| activate_rotation | BOOL | ☐ | VAR_INPUT |
| end_position | REAL | ☐ | VAR_INPUT |
| start_position | REAL | ☐ | VAR_INPUT |
| position_speed | REAL | ☐ | VAR_INPUT |
| enable | BOOL | ☐ | VAR_INPUT |
| rotation_speed | REAL | ☐ | VAR_INPUT |
| error | INT | ☐ | VAR_INPUT |
| rotation | REAL | | VAR_OUTPUT |
| position | REAL | | VAR_OUTPUT |
| rotation_end | BOOL | | VAR_OUTPUT |
| rotation_start | BOOL | | VAR_OUTPUT |
| position_end | BOOL | | VAR_OUTPUT |
| position_start | BOOL | | VAR_OUTPUT |
| position_current_speed | REAL | | VAR_OUTPUT |
| rotation_current_speed | REAL | | VAR_OUTPUT |
| body_color | INT | | VAR_OUTPUT |
| valve1_color | INT | | VAR_OUTPUT |
| valve2_color | INT | | VAR_OUTPUT |
| valve3_color | INT | | VAR_OUTPUT |
| grip_right_closed | BOOL | | VAR_OUTPUT |
| grip_left_open | BOOL | | VAR_OUTPUT |
| grip_left_closed | BOOL | | VAR_OUTPUT |
| grip_right_open | BOOL | | VAR_OUTPUT |
| grip_active | BOOL | | VAR_OUTPUT |

Obr. 42 Vstupy a výstupy funkčního bloku pro chapadla

**Zásobník puků**

Úkol zásobníku se zdá být jasný, avšak v simulaci ovládá hned několik procesů. Skrze tento funkční blok se definuje počet simulovaných puků, jejich barva a viditelnost. Viditelnost z toho důvodu, že všechny puky jsou v pozici 0 (překrývají se), ale jejich viditelnost se mění na základě jejich pořadí v zásobníku, což dělá dojem, že puky padají jeden do druhém ze zásobníku tak, jak jsou předem definovány. Viditelnost puku tedy simuluje přítomnost puku na lince. V procesu přemístění puku mezi pracovištěm dvě a tři pomáhá právě zásobník. Algoritmus pohybu puku je rozdělen zvlášť na lineární pozicování a rotaci. Toto opatření je z důvodu nevyhovujícího souřadnicového systému.

Z toho důvodu existují objekty v podobě puků zvlášť pro pracoviště jedna a dvě, dále samostatné puky v chapadlech a puky na karuselu. Na pracovištích jedna a dvě se puky pohybují pouze na základě lineárního pozicování XYZ. Dojede-li puk na konec druhého pracoviště, setrvává bez akce do doby, než je aktivováno chapadlo. V ten moment se jeho viditelnost změní do logické nuly a zároveň v ten stejný okamžik se změní viditelnost puku, umístěného v chapadlech, na logickou jedničku. Reálně se tedy jedná o dva rozdílné puky s jiným souřadnicovým systémem, ale vizualizační dojem působí tak, že je zachycen stále jeden puk. Stejný proces se opakuje při pokládání puku na karusel, kdy při otevření chapadel „zmizí“ puk v chapadlech a objeví se puk na karuselu (programově pouze změna viditelnosti objektů). Vizuální dojem tohoto procesu je věrný a pozorovatel vnímá simulaci spojitě. Barva puků v chapadlech i na karuselu je dynamická a mění se na základě barvy puku, který je první v pořadí na konci druhého pracoviště. Na Obr. 43 je seznam vstupů a výstupů funkčního bloku pro zásobník puků. Název funkčního bloku je „stackFB“.

| Name | Type | Reference | Scope |
|---|---|---|---|
| stackFB | | | |
| carrousel_rotation_num | INT | | VAR_INPUT |
| crane_grip_active | BOOL | | VAR_INPUT |
| crane_position | REAL | | VAR_INPUT |
| crane_position_end | BOOL | | VAR_INPUT |
| crane_position_start | BOOL | | VAR_INPUT |
| crane_rotation | REAL | | VAR_INPUT |
| crane_rotation_end | BOOL | | VAR_INPUT |
| crane_rotation_start | BOOL | | VAR_INPUT |
| ejector_rdy_to_eject | BOOL | | VAR_INPUT |
| puck1_position_X | REAL | | VAR_INPUT |
| puck2_position_X | REAL | | VAR_INPUT |
| puck3_position_X | REAL | | VAR_INPUT |
| puck4_position_X | REAL | | VAR_INPUT |
| puck5_position_X | REAL | | VAR_INPUT |
| quantity_of_pucks | INT | | VAR_INPUT |
| puck1_color | INT | | VAR_INPUT |
| puck2_color | INT | | VAR_INPUT |
| puck3_color | INT | | VAR_INPUT |
| puck4_color | INT | | VAR_INPUT |
| puck5_color | INT | | VAR_INPUT |
| crane_puck1_color | INT | | VAR_OUTPUT |
| crane_puck1_visible | BOOL | | VAR_OUTPUT |
| crane_puck2_color | INT | | VAR_OUTPUT |
| crane_puck2_visible | BOOL | | VAR_OUTPUT |
| crsl_puck1_color | INT | | VAR_OUTPUT |
| crsl_puck1_visible | BOOL | | VAR_OUTPUT |
| crsl_puck2_color | INT | | VAR_OUTPUT |
| crsl_puck2_visible | BOOL | | VAR_OUTPUT |
| crsl_puck3_color | INT | | VAR_OUTPUT |
| crsl_puck3_visible | BOOL | | VAR_OUTPUT |
| crsl_puck4_color | INT | | VAR_OUTPUT |
| crsl_puck4_visible | BOOL | | VAR_OUTPUT |
| crsl_puck5_color | INT | | VAR_OUTPUT |
| crsl_puck5_visible | BOOL | | VAR_OUTPUT |
| crsl_puck6_color | INT | | VAR_OUTPUT |
| crsl_puck6_visible | BOOL | | VAR_OUTPUT |
| crsl_puck7_color | INT | | VAR_OUTPUT |
| crsl_puck7_visible | BOOL | | VAR_OUTPUT |
| crsl_puck8_color | INT | | VAR_OUTPUT |
| crsl_puck8_visible | BOOL | | VAR_OUTPUT |
| puck1_visible | BOOL | | VAR_OUTPUT |
| puck2_visible | BOOL | | VAR_OUTPUT |
| puck3_visible | BOOL | | VAR_OUTPUT |
| puck4_visible | BOOL | | VAR_OUTPUT |
| puck5_visible | BOOL | | VAR_OUTPUT |

Obr. 43 Vstupy a výstupy funkčního bloku pro zásobník puků

### 5.4.1. Uživatelské rozhraní

Simulační bloky, které simulují reálný hardware, musejí mít základní uživatelské rozhraní totožné s fyzickým rozhraním. Každé pracoviště disponuje vstupy a výstupy, které jsou použity na řízení manipulační linky. Proto jsem připravil základní a rozšířené uživatelské rozhraní. U základního uživatelského rozhraní simulační linky jsou připraveny pro programátora vstupy a výstupy shodně, jako u linky fyzické. Rozšířené uživatelské rozhraní obsahuje další parametry, které lze definovat a měnit tak chování simulace. Aby byla simulace spustitelná pouze se základním uživatelským rozhraním, jsou v rozšířeném uživatelském rozhraní definovány výchozí parametry pro správný chod. Rozšířená uživatelská rozhraní a napojení simulačních proměnných na jednotlivé objekty jsou k nahlédnutí v příloze III. 2021_TOM0319_DP_Interface.

**Distribuční pracoviště** – základní uživatelské rozhraní
Struktura pr1 je společná pro celé distribuční pracoviště. V Tab. 2 je uvedeno základní uživatelské rozhraní pro toto pracoviště.

Tab. 2 Základní uživatelské rozhraní pro distribuční pracoviště

| Pracoviště 1 - fyzické X simulační rozhraní | | | | | |
|---|---|---|---|---|---|
| Název | I/O | Rozsah | Fyzikální rozsah | Simulační rozhraní | Komentář |
| Digitální vstupy | | | | | |
| B1 | I0.0 | BOOL | 24 VDC | pr1.conv_rotate | Indukční snímač rotace pásu |
| B2 | I0.1 | BOOL | 24 VDC | pr1.puck_detector_detected | Snímač přítomnosti puku |
| B3 | I0.2 | BOOL | 24 VDC | pr1.ejector_rdy_to_eject | Vyrážeč zasunut |
| B4 | I0.3 | BOOL | 24 VDC | pr1.ejector_ejected | Vyrážeč vysunut |
| Digitální výstupy | | | | | |
| Y1 | Q0.0 | BOOL | 24 VDC | pr1.ejector_activate | Aktivace vyrážeče |
| YM1 | Q0.1 | BOOL | 24 VDC | pr1.conv_activate | Aktivace pasu |

**Testovací pracoviště** – základní uživatelské rozhraní
Struktura pr2 je společná pro celé distribuční pracoviště. V Tab. 3 je uvedeno základní uživatelské rozhraní pro toto pracoviště

Tab. 3 Základní uživatelské rozhraní pro testovací pracoviště

| Pracoviště 2 - fyzické rozhraní X simulační rozhraní | | | | | |
|---|---|---|---|---|---|
| Název | I/O | Rozsah | Fyzikální rozsah | Uživatelské rozhraní | Komentář |
| Digitální vstupy | | | | | |
| B1 | I0.0 | BOOL | 24 VDC | pr2.conv1_rotate | Inkrementální snímač dopravníkového pásu č.1. |
| B2 | I0.1 | BOOL | 24 VDC | pr2.conv2_rotate | Inkrementální snímač dopravníkového pásu č.2. |
| B3 Q1 | I0.2 | BOOL | 24 VDC | pr2.color_det_red | Snímač barev – červená |
| B3 Q2 | I0.3 | BOOL | 24 VDC | pr2.color_det_blue | Snímač barev – modrá |
| B3 Q3 | I0.4 | BOOL | 24 VDC | pr2.color_det_alu | Snímač barev – hliník |
| B3 Q4 | I0.5 | BOOL | 24 VDC | pr2.color_det_undef | Snímač barev – nedefinovaná |
| B4 | I0.6 | BOOL | 24 VDC | pr2.mag_det_detected | Snímač magnetických vlastností |
| B6 | I0.7 | BOOL | 24 VDC | pr2.puck_detector_detected | Optický snímač přítomnosti na páse |
| B7 | I1.0 | BOOL | 24 VDC | pr2.barrier1_start_pos | Závora 1 zasunuta |
| B8 | I1.1 | BOOL | 24 VDC | pr2.barrier1_end_pos | Závora 1 vysunuta |
| B9 | I1.2 | BOOL | 24 VDC | pr2.ejector_rdy_to_eject | Vyrážeč zasunut |
| B10 | I1.3 | BOOL | 24 VDC | pr2.ejector_ejected | Vyrážeč vysunut |
| B11 | I1.4 | BOOL | 24 VDC | pr2.barrier2_start_pos | Závora 2 zasunuta |
| B12 | I1.5 | BOOL | 24 VDC | pr2.barrier2_end_pos | Závora 2 vysunuta |
| Digitální výstupy | | | | | |
| B3 AT | Q0.0 | BOOL | 24 VDC | pr2.color_det_activate | Vstup snímače barev – aktivace senzoru externím synchronizačním pulzem |
| B3 ET | Q0.1 | BOOL | 24 VDC | / | Vstup snímače barev – externí učící vstup (programování referenční barvy) |
| Y1 | Q0.2 | BOOL | 24 VDC | pr2.barrier1_activate | Vstup aktivující závoru 1 – zadržení puku v sekci měření |
| Y2 | Q0.3 | BOOL | 24 VDC | pr2.ejector_activate | Vstup aktivující vyrážeč – puk je přesunut na dopravníkový pás č.2. |
| Y3 | Q0.4 | BOOL | 24 VDC | pr2.barrier2_activate | Vstup aktivující závoru 2 – zadržení puku v poloze pro přesun na dopravníkový pás č.2. |
| Y4 | Q0.5 | BOOL | 24 VDC | pr2.height_sensor_activate | Vstup aktivující měření výšky – píst válce s analogovým snímáním výšky se vysune |
| YM1 | Q0.6 | BOOL | 24 VDC | pr2.conv1_activate | Vstup aktivující elektromotor dopravníkového pásu č.1. |
| YM2 | Q0.7 | BOOL | 24 VDC | pr2.conv2_activate | Vstup aktivující elektromotor dopravníkového pásu č.2. |
| Analogové vstupy | | | | | |
| B5 | PIW 272 | analog | 0-10V | pr2.height_sensor_value | Senzor výšky |

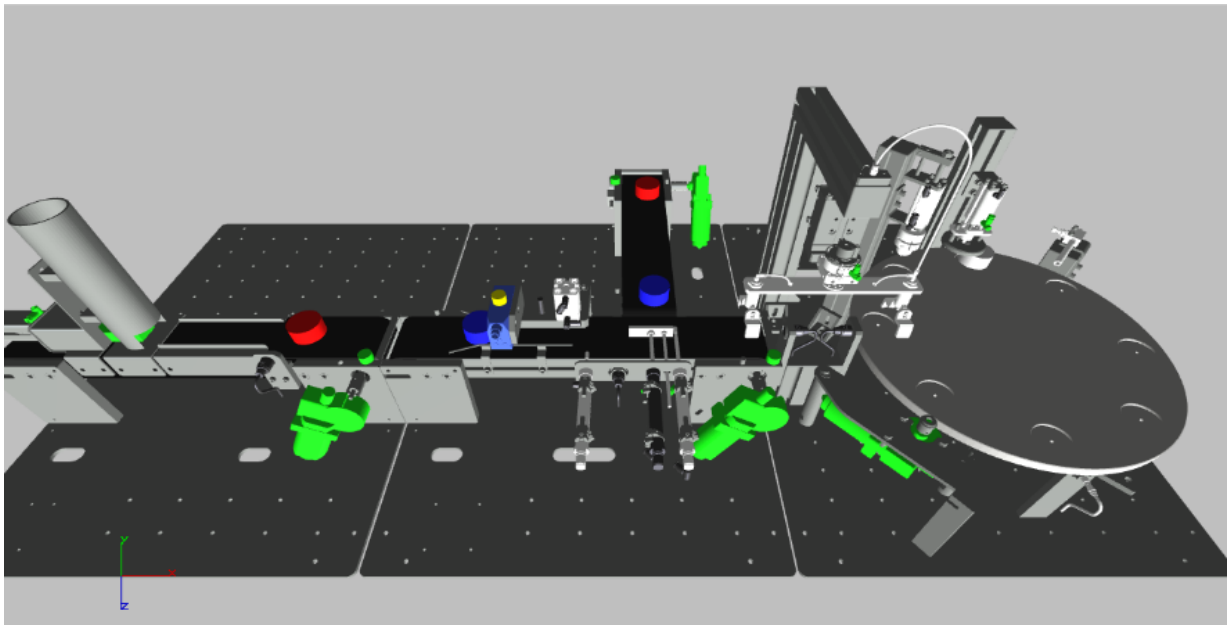
**Procesní pracoviště** – základní uživatelské rozhraní
Struktura pr3 je společná pro celé distribuční pracoviště. V Tab. 4 je uvedeno základní uživatelské rozhraní pro toto pracoviště

Tab. 4 Základní uživatelské rozhraní pro procesní pracoviště

| Pracoviště 3 - fyzické rozhraní | | | | | | |
|---|---|---|---|---|---|---|
| Název | I/O | Canon | Rozsah | Fyzikální rozsah | Uživatelské rozhraní | Komentář |
| Digitální výstupy | | | | | | |
| Y1 | Q0.0 | 1.15 | BOOL | 24 VDC | pr3.arrestment_activate | Aretace karuselu |
| Y2 | Q0.1 | 1.16 | BOOL | 24 VDC | pr3.stamp_eject | vysunutí razičky |
| Y3 | Q0.2 | 1.17 | BOOL | 24 VDC | pr3.drill_eject | vysunutí vrtačky |
| Y4 | Q0.3 | 1.18 | BOOL | 24 VDC | pr3.crane_activate_down | Chapadla dolů |
| Y5 | Q0.4 | 1.19 | BOOL | 24 VDC | pr3.crane_activate_rotate | Chapadla otočit |
| Y6 | Q0.5 | 1.20 | BOOL | 24 VDC | pr3.crane_activate_grip | Chapadla sevřena |
| Y7 | Q0.6 | 1.21 | BOOL | 24 VDC | pr3.air_activate | Aktivace ofukování |
| YM1 | Q0.7 | 1.22 | BOOL | 24 VDC | pr3.carrousel_activate | Motor karusel |
| YM2 | Q1.1 | 2.15 | BOOL | 24 VDC | pr3.drill_activate | Motor vrtačky |
| Digitální vstupy | | | | | | |
| B2 | I0.1 | 1.3 | BOOL | 24 VDC | pr3.mov_carrousel_rotate | Indukční snímač karuselu |
| B3 | I0.2 | 1.4 | BOOL | 24 VDC | pr3.carrousel_arrested | Karusel zaaretován? |
| B4 | I0.3 | 1.5 | BOOL | 24 VDC | pr3.carrousel_free | Karusel nearetován? |
| B5 | I0.4 | 1.6 | BOOL | 24 VDC | pr3.stamp_ejected | Razník vysunut? |
| B6 | I0.5 | 1.7 | BOOL | 24 VDC | pr3.drill_ejected | Vrtačka zasunuta? |
| B7 | I0.6 | 1.8 | BOOL | 24 VDC | pr3.crane_position_start | Chapadlo nahoře? |
| B8 | I0.7 | 1.9 | BOOL | 24 VDC | pr3.crane_position_end | Chapadla dole? |
| B9 | I1.0 | 2.2 | BOOL | 24 VDC | pr3.crane_rotation_start | Chapadlo uvnitř? |
| B10 | I1.1 | 2.3 | BOOL | 24 VDC | pr3.crane_rotation_end | Chapadle vně? |
| B11 | I1.2 | 2.4 | BOOL | 24 VDC | p3.pr2_puck_detected | Puk na předchozím stanovišti? |
| B12 | I1.3 | 2.5 | BOOL | 24 VDC | p3.puck_detected | Puk na karuselu pod chapadly? |

## 5.5. Řídicí program

Ověření funkčnosti simulace jsem ověřil řídicím programem. Postupoval jsem naprosto stejně, jako bych pracoval s fyzickou linkou, dle výše zmíněného uživatelského rozhraní. Simulace je spuštěna pouze impulzem náběžné hrany, přičemž další procesy jsou zcela zautomatizovány. Řídicí program funguje následovně. Prvotním impulzem je vysunut vyrážeč, který vyšle první puk na linku. Tímto se spustí všechny pásy. Po detekci prvního puku v prvním stanovišti je vyrážeč opět aktivován. Takto je zajištěna pravidelná distribuce puků ze zásobníku. V testovacím pracovišti je puk testován dle sledu snímačů. V moment, kdy je jakýkoliv výstup snímače barev aktivní, je aktivována první závora, aby mohla být změřena výška puku. Prodleva mezi detekováním barvy puku a umístěním puku pod snímačem výšky je zhruba 2 sekundy. Proto je snímač výšky aktivován s 2,5 sekundovým zpožděním. Během měření výšky je zastaven pás v distribučním stanovišti, neboť by hrozila kolize puků. Pás je opět v provozu až je první závora zasunuta. Řídicí program předpokládá, že kovový puk je vhodný pro další zpracování a bude předán na procesní stanoviště, tudíž není vyražen na přilehlý pás, kde puk svůj cyklus končí. Stejný proces se opakuje s následujícími puky s tím rozdílem, že pokud je zaznamenám plastový puk, respektive modrý nebo červený, jsou postupně vyráženy na přilehlý pás. Před vyražením je vždy aktivována druhá závora. Kovové puky dále pokračují v životním cyklu na procesním pracovišti. Postupně je každý puk podroben vrtání, ražbě a ofukování. Časové intervaly jednotlivých procesů se liší. Na Obr. 44 je ukázka simulace s řídicím programem. Celý software je v příloze I. 2021_TOM0319_DP_Program_(Automation Studio) spolu s vizualizací simulace v příloze II. 2021_TOM0319_DP_Vizualizace_(Scene Viewer).



Obr. 44 Ukázka simulace

## 6. Testování havarijních stavů

Simulace umožňuje testování havarijních stavů na základě předem definovaných chyb. Nástroj Scene Viewer nedisponuje zpětnou vazbou, tedy pouze zobrazuje chyby a poruchy v simulaci, které jsou implementovány v simulačních funkčních blocích v PLC. V Tab. 5 je přehled možných chyb, které lze simulovat v distribučním pracovišti.

Tab. 5 Havarijní stavy pro distribuční pracoviště

| Havarijní stavy | | | | |
|---|---|---|---|---|
| Distribuční pracoviště | | | | |
| Objekt | HODNOTA NA VSTUPU FB: ERROR | | | |
| | 0 | 1 | 2 | 3 |
| Vyrážeč | bez chyby | nefunkční | snížení rychlosti | nefunkční ventil pro vrácení |
| Pás | bez chyby | nefunkční | snížení rychlosti | |
| Snímač puku | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |
| Snímač pohybu pásu | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |

V Tab. 6 je přehled možných chyb, které lze simulovat v testovacím pracovišti.

Tab. 6 Havarijní stavy pro testovací pracoviště

| Havarijní stavy | | | | |
|---|---|---|---|---|
| Testovací pracoviště | | | | |
| Objekt | HODNOTA NA VSTUPU FB: ERROR | | | |
| | 0 | 1 | 2 | 3 |
| Snímač barev | bez chyby | nefunkční | přehozené barvy | |
| Snímač mag. Vlastností | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |
| Snímač výšky | bez chyby | nefunkční | | |
| Snímač puku | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |
| Závora 1 | bez chyby | nefunkční | nefunkční koncový snímač koncové polohy | nefunkční ventil pro vrácení |
| Závora 2 | bez chyby | nefunkční | nefunkční koncový snímač koncové polohy | nefunkční ventil pro vrácení |
| Vyrážeč | bez chyby | nefunkční | snížení rychlosti | nefunkční ventil pro vrácení |
| Pás 1 | bez chyby | nefunkční | snížení rychlosti | |
| Pás 2 | bez chyby | nefunkční | snížení rychlosti | |
| Snímač pohybu pásu | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |

V Tab. 6 je přehled možných chyb, které lze simulovat v procesním pracovišti.

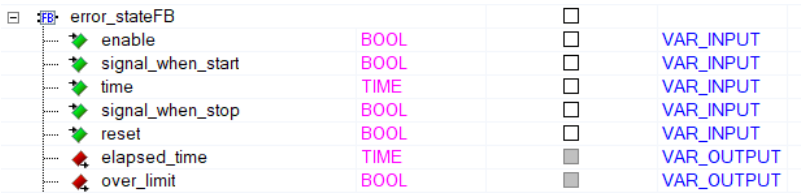
Tab. 7 Havarijní stavy pro procesní stanoviště

| Havarijní stavy | | | | |
|---|---|---|---|---|
| Procesní stanoviště | | | | |
| Objekt | HODNOTA NA VSTUPU FB: ERROR | | | |
| | 0 | 1 | 2 | 3 |
| Motor karuselu | bez chyby | nefunkční | snížení rychlosti | |
| Chapadla | bez chyby | nefunkční | rotace omezena do 150 stupňů | |
| Vrtačka | bez chyby | nefunkční | / | |
| Pohyb vrtačky | bez chyby | nefunkční | snížení rychlosti | nefunkční ventil pro vrácení |
| Ofukování | bez chyby | nefunkční | / | |
| Razič | bez chyby | nefunkční | snížení rychlosti | |
| Pohyb raziče | bez chyby | nefunkční | snížení rychlosti | nefunkční ventil pro vrácení |
| Ind. Snímač pohybu kar. | bez chyby | nefunkční | negovaná hodnota | |
| Snímač puku 1 | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |
| Snímač puku 2 | bez chyby | nefunkční | po detekci zaseknutý výstup v log. 1 | |

**Měření doby odezvy**

Do programu jsem implementoval funkční blok, který dokáže měřit, zda-li byl daný proces zvládnut v termínu. Název funkčního bloku je „error_stateFB“. Na Obr. 45 je seznam vstupů a výstupů funkčního bloku. Vstup „signal_when_start“ slouží k aktivaci časovače, vstup „signal_when_stop“ slouží k zastavení časovače, vstup „time“ slouží k nastavení deadline termínu.

*Pozn.: „Deadline“ je konečný termín, kdy musí být splněna určitá podmínka, fáze či proces.*




Obr. 45 Vstupy a výstupy funkčního bloku pro měření odezvy

Jednoduchý příkladem je měření času pohybu puku po lince. Na vstup „singal_when_start“ je přiveden koncový snímač polohy vyrážeče. Na vstup „signal_when_stop“ je přiveden výstup detektoru puků. Na vstup „time“ je přiveden čas 3 sekundy. Odpočet začne v momentě, kdy je puk vyražen a skončí tehdy, kdy je detekován detektorem puků. Pokud tento proces bude trvat déle než je nastavený čas, bude výstup „over_limit“ aktivní až do resetu. Pokud puk stihne dojet k detektoru puku dříve než nastavený čas, výstup „elapsed_time“ zobrazí, jak dlouho tento proces trval.

## 7. Závěrečné zhodnocení dosažených výsledků

Závěrem bych rád zhodnotil dosažené výsledky mé diplomové práce. Teoretická část podává stručný a ucelený obraz na problematiku programovatelných automatů a jejich využití v automatizačních procesech. Dále objasňuje pojmy virtuální uvedení do provozu a digitální dvojče spolu s výběrem několika hojně používaných softwarů pro tyto aplikace. V praktické části jsem vytvořil koncept digitálního dvojčete pomocí vývojového prostředí Automation Studio a vizualizačního nástroje Scene Viewer. Na rozdíl od jiných běžně používaných softwarů pro simulační účely, Scene Viewer nedokáže poskytovat zpětnou vazbu a slouží tedy pouze jako vizualizace simulace, nikoliv simulace samotná. Přenos dat probíhá jednosměrně skrze OPC UA. Výhodou využití softwaru Automation Studio v kombinaci Scene Viwer je maximální kompatibilita, a to v takovém rozsahu, že připojení k OPC UA zabere několik málo kliknutí, přičemž není nutné provádět žádná nastavení. Celý simulační program běží na fyzickém nebo virtuálním PLC. Scene Viewer obsahuje statický model ve formátu STL, na který lze napojit na příslušné proměnné. Na základě těchto proměnných je simulován pohyb, změna barvy, viditelnost a několik dalších parametrů. Při zvyšování počtu modelů nebo jejich grafické náročnosti se rapidně snižuje plynulost simulace, proto bylo nutné z CAD souborů odstranit vešekeré nepotřebné komponenty. Pokud bych chtěl vymodelovat koncept digitálního dvojčete, která věrně zachycuje reálné chování, musel bych daný CAD model importovat do simulačního softwaru MapleSim – B&R Edition nebo do jiných, principiálně stejných softwarů. V takovém softwaru je tvořen model na základě rovnic, vazeb a fyzikální zákonů. Tento model je exportován do FMU knihovny, která je zdrojem dat pro simulaci. V prostředí Automation Studio jsou následně využity simulační funkční bloky z této knihovny. Vytvoření takového modelu jednak vyžaduje znalosti v daném oboru, což vyžaduje určitý čas k vzdělávání a jednak je nutný licenční klíč k těmto profesionálním nástrojům. A to je podstata této práce. V mém případě jsem nevyužil žádných softwarů, ke kterým je nutné vlastnit licenční klíč. Nevýhodou tvořená vlastních simulačních funkčních bloků je však velice omezená odezva simulace, a to z důvodu zanedbání fyzikálních vlastností a nelinearity. Není v silách programátora, aby bez pomoci dalšího softwaru vytvořil simulační funkční blok, který by zvládl reagovat jako reálný fyzický hardware. Proto je výstup simulačního funkčního bloku vždy lineární a lze očekávat stejný výsledek za různých podmínek. Scene Viewer také nedisponuje kolizí předmětu, tzn. pokud se pozice dvou stejných objektů ve stejném souřadnicovém sytému budou shodovat ve všech osách, předměty se budou překrývat a simulace na to neupozorní. V takovém případě by musel být v simulačním programu implementovan funkční blok na kontrolu pozic objektů. Kromě impotu STL modelu, lze vytvářet jednoduché geometrické tvary přímo v prostředí Scene Viewer nebo využít připravených objektů, a to včetně robotických paží nebo celých robotů COMAU. Simulovaná manipulační linka je svým uživatelský rozhraním připravena stejně jako linka fyzická. Simulace tímto způsobem je naprosto nevyhovující dnešním standardům, ale pro výukové účely shledávám tento způsob jako velice inovativní. Kdokoliv si může model manipulační linky spustit na svém počítači a implementovat vlastní řídicí program. Pro fázi dalšího vývoje aplikace bych doporučil využít simulačního softwaru MapleSim pro tvorbu simulačních funkčních bloků. Tímto způsobem by bylo zajištěno věrné chování simulace s ohledem na fyzikální zákony a reálné parametry hardwaru.

## Zdroje


[1] KOZIOREK, Jiří, Antonín KUČERA, Jiří HAŠKA a Jan ŠMÍD. *PROGRAMOVATELNÉ AUTOMATY A VIZUALIZACE ŘÍDICÍCH SYSTÉMŮ*. Ostrava, 2012. Vysoká Škola Báňská - Technicka univerzita.

[2] S800 I/O. *ABB* [online]. [cit. 2021-4-28]. Dostupné z: https://webimages.imagebank.abb.com/public/default/product/9AAC207665/preview

[3] PLC Siemens CPU 1214C. *Conrad* [online]. [cit. 2021-4-28]. Dostupné z: https://asset.conrad.com/media10/isa/160267/c1/-/pl/197468_LB_00_FB/image.jpg

[4] X20 system. *B&R automation* [online]. [cit. 2021-4-28]. Dostupné z: https://www.br-automation.com/fileadmin/1363596282730-de-html-1.2.jpg

[5] Mitsubishi Electric PLC. *Indiamart* [online]. [cit. 2021-4-28]. Dostupné z: https://5.imimg.com/data5/OJ/BG/VK/SELLER-2678651/mitsubishi-electricals-500x500.jpg

[6] PLC systems. *B&R automation* [online]. [cit. 2021-4-28]. Dostupné z: https://www.br-automation.com/fileadmin/1331004070910-de-html-1.0.jpg

[7] I/O systems. *B&R automation* [online]. [cit. 2021-4-28]. Dostupné z: https://www.br-automation.com/fileadmin/1331004070861-de-html-1.1.jpg

[8] 2500P-ECC1 ETHERNET COMMUNICATION COPROCESSOR. *Napa.fr* [online]. [cit. 2021-4-28]. Dostupné z: http://www.napa.fr/en/product/2500p-ecc1

[9] KUMSTÁT, Jan. *Náhrada stávajícího reléového řídicího systému pomocí PLC*. Plzeň, 2012. Diplomová práce. ZÁPADOČESKÁ UNIVERZITA V PLZNI.

[10] *PROGRAMOVATELNÉ AUTOMATY* [online]. In: . [cit. 2021-4-28]. Dostupné z: http://isst.hys.cz/images/prezentace/PA-22.pdf

[11] DOBA CYKLU. *Program-plc* [online]. [cit. 2021-4-28]. Dostupné z: http://plc-automatizace.cz/knihovna/plc/plc-doba-cyklu.htm

[12] Operating Cycle of Programmable Logic Controller. *Program-plc* [online]. [cit. 2021-4-28]. Dostupné z: https://program-plc.blogspot.com/2011/11/operating-cycle-of-programmable-logic.html

[13] CHMIEL, Mirosław a Edward HRYNKIEWICZ. *Fast Operating Bit-Byte PLC* [online]. In: . Soul, Korea, 2008, s. 6 [cit. 2021-4-28]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1474667016413728?via%3Dihub

[14] FRANK. PLC Memory. *Automation primer* [online]. [cit. 2021-4-28]. Dostupné z: https://automationprimer.com/2016/08/28/plc-memory/

[15] *Introduction to Programmable Logic Controllers - Part 1: Introduction to PLC Hardware* [online]. In: . s. 3 [cit. 2021-4-28]. Dostupné z: https://pdhonline.com/courses/e116/PLC-module1.pdf

[16] Paměť FRAM. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-4-28]. Dostupné z: https://cs.wikipedia.org/wiki/Pam%C4%9B%C5%A5_FRAM

[17] What is the Function of RAM and EEPROM memory in Twido PLC. *Schneider Electric* [online]. [cit. 2021-4-28]. Dostupné z: https://www.se.com/se/en/faqs/FA143020/

[18] MUTHUKRISHNAN, Vidya. Programmable Logic Controllers (PLCs): Basics, Types & Applications. *Electrical 4 U* [online]. 2021 [cit. 2021-4-28]. Dostupné z: https://www.se.com/se/en/faqs/FA143020/

[19] PHILLIPS, Rick. What are the different types of PLC? *Basic PLC* [online]. 2019 [cit. 2021-4-28]. Dostupné z: https://basicplc.com/different-types-of-plc/

[20] AYINLA, Raji. Top 5 PLC programming languages. *Medium* [online]. 2020 [cit. 2021-4-30]. Dostupné z: https://medium.com/the-open-manuel/top-5-plc-programming-languages-edf4de2dc3e4 TURNBULL, CHARLOTTE. WHAT IS VIRTUAL COMMISSIONING? *Virtual Commisioning* [online]. 2019 [cit. 2021-4-28]. Dostupné z: https://virtualcommissioning.com/what-is-virtual-commissioning/

[21] PLC Program for Blinking Lamp on 5 Seconds Interval. *Instrumentation tools* [online]. [cit. 2021-4-28]. Dostupné z: https://instrumentationtools.com/wp-content/uploads/2018/06/PLC-Program-for-Blinking-Lamp-2.png

[22] Function Block Diagram (FBD) Programming Tutorial. *PLC academy* [online]. [cit. 2021-4-28]. Dostupné z: https://www.plcacademy.com/wp-content/uploads/2018/02/basic-function-block-diagram-300x280.png?ezimgfmt=ng:webp/ngcb60

[23] Why is the instruction list (IL) language for PLCs falling out of favor? *PLC academy* [online]. [cit. 2021-4-28]. Dostupné z: https://3l4sbp4ao2771ln0f54chhvm-wpengine.netdna-ssl.com/wp-content/uploads/2019/07/Ladder-diagram-vs-instruction-list-246x300.png

[24] JACKSON, CHAD. HYBRID DIGITAL TWIN FOR VIRTUAL COMMISSIONING. *Virtual Commisioning* [online]. 2020 [cit. 2021-4-28]. Dostupné z: https://virtualcommissioning.com/hybrid-digital-twin-for-virtual-commissioning/

[25] Hardware-in-the-Loop Simulation. *Speed Goat* [online]. [cit. 2021-4-28]. Dostupné z: https://www.speedgoat.com/applications-industries/applications/hardware-in-the-loop?utm_term=dspace%20hardware%20in%20the%20loop&utm_campaign=RCP+and+HIL&utm_source=adwords&utm_medium=ppc&hsa_acc=6520550235&hsa_cam=12149050276&hsa_grp=115249820045&hsa_ad=497575777415&hsa_src=g&hsa_tgt=kwd-362420883008&hsa_kw=dspace%20hardware%20in%20the%20loop&hsa_mt=b&hsa_net=adwords&hsa_ver=3&gclid=Cj0KCQjwppSEBhCGARIsANIs4p6lpPtmDt1c3oNrw6_xrxWJMgtR0VDU3nmvieYqAJwyytXyvvrofKAaAhNOEALw_wcB

[26] Relative Time Synchronization of Distributed Applications for Software-in-the-Loop Simulation. *Semantic Scholar* [online]. [cit. 2021-4-28]. Dostupné z: https://www.semanticscholar.org/paper/Relative-Time-Synchronization-of-Distributed-for-Lee-Hwang/412d1a5a02ecb7128f940aa2a55415e6dd1bab14

[27] SIMATIC/SIMOTION Virtual Commissioning with Hardware in the Loop. *SIEMENS* [online]. [cit. 2021-4-28]. Dostupné z: https://support.industry.siemens.com/cs/document/109758739/simatic-simotion-virtual-commissioning-with-hardware-in-the-loop?dti=0&lc=en-WW

[28] JACKSON, GRAHAM. TYPES OF DIGITAL TWINS – FINDING YOUR FIT. *Virtual Commisioning* [online]. [cit. 2021-4-28]. Dostupné z: https://virtualcommissioning.com/types-of-digital-twins-finding-your-fit/

[29] SIMIT Simulation Platform. *SIEMENS* [online]. [cit. 2021-4-28]. Dostupné z: https://mall.industry.siemens.com/collaterals/files/103/JPG/S_PCS7_XX_00200j.JPG

[30] SIMIT Simulation Platform. *SIEMENS* [online]. [cit. 2021-4-28]. Dostupné z: https://mall.industry.siemens.com/mall/cs/cz/Catalog/Products/10380808?tree=CatalogTree#Overview

[31] What is WinMOD? *WinMOD* [online]. [cit. 2021-4-28]. Dostupné z: https://www.winmod.de/english/winmod/

[32] WinMOD SIMLINE. *WinMOD* [online]. [cit. 2021-4-28]. Dostupné z: https://www.winmod.de/english/products/winmod-simline/winmod-simline-libraries/#basic

[33] TECNOMATIX. *KS Industry Solutions* [online]. [cit. 2021-4-28]. Dostupné z: https://ks-iss.com/produkty/tecnomatix-process-simulate-robotexpert/

[34] TECNOMATIX. *KS Industry Solutions* [online]. [cit. 2021-4-28]. Dostupné z: https://ks-iss.com/wp-content/uploads/2018/02/Wirtualna-fabryka-768x392.jpg

[35] SIMIT Simulation Platform. *SIEMENS* [online]. [cit. 2021-4-28]. Dostupné z: https://mall.industry.siemens.com/collaterals/files/103/JPG/S_PCS7_XX_00200j.JPG

[36] Definition of Enterprise Resource Planning (ERP). *Oracle* [online]. [cit. 2021-4-29]. Dostupné z: https://www.oracle.com/erp/what-is-erp/
[37] ABOUT KUKA. *KUKA* [online]. [cit. 2021-4-29]. Dostupné z: https://www.kuka.com/en-de/about-kuka
[38] KUKA.Sim. *KUKA* [online]. [cit. 2021-4-29]. Dostupné z: https://www.kuka.com/en-de/products/robot-systems/software/planning-project-engineering-service-safety/kuka_sim
[39] B&R Help Explorer – Automation Studio v. 4.7.2.98 [online]. [cit. 2021-4-29]. dostupné ke stažení z: https://www.br-automation.com/cs/soubory-ke-stazeni/software/automation-studio/automation-studio-47/automation-studio-v47/?noredirect=1
[40] X20 Systém User's Manual [online]. [cit. 2021-4-29]. Dostupné z: https://www.br-automation.com/cs/soubory-ke-stazeni/control-and-io-systems/x20-system/x20-system-users-manual/?noredirect=1
[41] BACKPLANE - překlad. *Anglicko-český slovník* [online]. [cit. 2021-4-29]. Dostupné z: https://www.anglickoceskyslovnik.cz/backplane.htm
[42] MARTINEK, David. Profilování a optimalizace programů. *VUT V BRNĚ* [online]. [cit. 2021-4-29]. Dostupné z: http://www.fit.vutbr.cz/~martinek/clang/profiling.html
[43] System Diagnostic Manager. *B&R automation* [online]. [cit. 2021-4-29]. Dostupné z: https://www.br-automation.com/fileadmin/1601828979411-en-html-1.0.jpg
[44] B&R Introduces X20 System – The New Automation Standard. *Automation* [online]. [cit. 2021-4-29]. Dostupné z: https://www.automation.com/en-us/articles/2005-2/br-introduces-x20-system-150-the-new-automation-st
[45] Diagnostics. *B&R automation* [online]. [cit. 2021-4-29]. Dostupné z: https://www.br-automation.com/cs/produkty/software/additional-information/diagnostics/
[46] MapleSim - B&R Edition. *B&R automation* [online]. [cit. 2021-4-29]. Dostupné z: https://www.br-automation.com/cs/soubory-ke-stazeni/software/simulation/maplesim-br-edition/?noredirect=1
[47] New possibilities with digital twins. *B&R automation* [online]. [cit. 2021-4-29]. Dostupné z: https://www.br-automation.com/cs/o-nas/tiskove-zpravy/new-possibilities-with-digital-twins-23-01-2019/
[48] Co je to Runtime? *IT slovník* [online]. [cit. 2021-4-29]. Dostupné z: https://it-slovnik.cz/pojem/runtime
[49] *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2021-4-29]. Dostupné z: https://cs.wikipedia.org/wiki/Fieldbus
[50] HOPPE, Stefan a Alexander STARK. IoT Basics: What is OPC UA? *Spotlight Metal* [online]. [cit. 2021-4-29]. Dostupné z: https://www.spotlightmetal.com/iot-basics-what-is-opc-ua-a-842878/

**Doporučená literatura**

[51] Martinásková Marie, Šmejkal Ladislav, PLC a automatizace 1 základní pojmy, úvod do programování BEN - technická literatura 2002 - ISBN 80-86056-58-9
[52] Šmejkal Ladislav, PLC a automatizace 2, Sekvenční logické systémy a základy fuzzy logiky, BEN - technická literatura, 2005 - ISBN 80-7300-087-3
[53] Armendia, M., Ghassempouri, M., Ozturk, E., Peysson, F., Twin-Control A Digital Twin Approach to Improve Machine Tools Lifecycle, 2019 ISBN 978-3-030-02203-7

## Seznam příloh

Přílohy jsou nahrány do IS EDISON.

| Označení | Název | Specifikace |
|---|---|---|
| I. | 2021_TOM0319_DP_Program_(Automation Studio) | Příloha v IS EDISON. |
| II. | 2021_TOM0319_DP_Vizualizace_(Scene Viewer) | Příloha v IS EDISON. |
| III. | 2021_TOM0319_DP_Interface | Příloha v IS EDISON. |