ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra telekomunikační techniky

Demonstrační model průmyslového řízení

květen 2024

Bakalant: Tomáš Janča

Vedoucí práce: doc. Ing. Jiří Vodrážka, Ph.D.

Čestné prohlášení

Prohlašuji, že jsem zadanou bakalářskou práci zpracoval sám s přispěním vedoucího práce a konzultanta a používal jsem pouze literaturu v práci uvedenou. Dále prohlašuji, že nemám námitek proti půjčování nebo zveřejňování mé bakalářské práce nebo její části se souhlasem katedry.

Datum: 20. května 2024 .....

Podpis bakalanta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Janča** Jméno: **Tomáš** Osobní číslo: **507399**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra telekomunikační techniky**

Studijní program: **Elektronika a komunikace**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Demonstrační model průmyslového řízení

Název bakalářské práce anglicky:

Industrial Control Demonstration Model

Pokyny pro vypracování:

Navrhněte a realizujte demonstrátor vybraného systému, např. úseku výrobní linky. K realizaci použijte polytechnickou stavebnicí MERKUR, případně doplněnou o další vyrobené díly (např. na 3D tiskárně). K řízení použijte platformu IPLOG společnosti METEL. Vypracujte metodiku a návod, který poslouží dalším studentům pro realizaci dalších projektů s danou platformou.

Seznam doporučené literatury:

[1] IPLOG-GAMA - průmyslové PLC s OS Linux a IO moduly. METEL. On-line: https://www.metel.eu
[2] Tomáš Tomašica - Demonstrační model průmyslového řízení, bakalářská práce, ČVUT v Praze, 2020.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Jiří Vodrážka, Ph.D. katedra telekomunikační techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **01.02.2024** Termín odevzdání bakalářské práce: **24.05.2024**

Platnost zadání bakalářské práce: **21.09.2025**

doc. Ing. Jiří Vodrážka, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. | Datum převzetí zadání

Podpis studenta

Anotace:

Tato práce se zabývá problematikou průmyslového řízení pomocí jednotek PLC. Konkrétně PLC jednotkou IPLOG od české společnosti Metel. V práci je ukázáno jednak prvotní nastavení jednotky včetně úvodní orientace ve způsobech jejího programování. Funkčnost jednotky je poté v práci demonstrována na modelu dopravního pásu, a to včetně zdůraznění důležitých kroků z procesu návrhu tohoto modelu.

Klíčová slova:

Metel, IPLOG, PLC, Pásový dopravník, Demonstrační model

Summary:

This thesis deals with the issue of industrial control using PLC units, specifically, the IPLOG PLC unit from the Czech company Metel. It shows the initial setup of the unit including the initial orientation in its programming. The functionality of the unit is illustrated on a model of a conveyor belt. Thesis then focuses on highlighting important steps in the design process of this model.

Index Terms:

Metel, IPLOG, PLC, Conveyor belt, Demonstration model

OBSAH

1. Úvod 8
2. Způsob využití řídící jednotky 9
2.1. Seznámení s fyzickými vlastnostmi řídící jednotky 9
2.2. Prvotní nastavení IPLOG 11
2.2.1. Nastavení Ethernetového rozhraní 11
2.2.2. Nastavení generování IO pro modul 12
2.2.3. Získání IO souboru 14
2.3. Instalace a úvodní orientace v programovacím studiu 15
2.3.1. Instalace studia 15
2.3.2. Programování ve studiu – norma IEC 61131-3 15
2.3.3. Založení projektu 15
2.3.4. Založení programu a import IO_variables 16
2.3.5. Nastavení „Run“ a „Debug“ 17
2.4. Základní programování v METEL-IDE 19
2.4.1. Naprogramování hlavního programu 19
2.4.2. Simulace a spuštění programu 22
3. Vlastní konstrukce 24
3.1. Návrh modelu a jeho konstrukce 24
3.1.1. Výběr vhodného zařízení pro realizaci 24
3.1.2. Fyzické vlastnosti modelu 24
3.1.3. Výběr vhodného způsobu konstrukce 25
3.1.4. Vytvoření podpůrné konstrukce pro elektrické prvky 25
3.2. Elektronicko-mechanická část modelu 27
3.2.1. Konstrukce a pohon pásu, signalizace jeho spuštění 27
3.2.2. Realizace stanic pásové výroby 29
3.2.3. Realizace shazovače 29
3.2.4. Realizace nakládacího sila 31
3.2.5. Prvky pro interakci s uživatelem 33
3.2.6. Další použité elektronické prvky 33
3.3. Řídící program modelu 34
3.3.1. Funkční blok Paměť 35
3.3.2. Funkční blok Analog 36
3.3.3. Funkční blok Ping 37
3.3.4. Fukční blok Průjezd 37
3.3.5. Funkční blok Píst 38
3.3.6. Hlavní program 39
4. Závěr 43
Reference 44

SEZNAM OBRÁZKŮ

Obrázek 1 - Horní strana řídicí jednotky IPLOG 9
Obrázek 2 - Boční strana řídící jednotky IPLOG [1] 11
Obrázek 3 - Nastavení IP adresy 12
Obrázek 4 - Výpis z PuTTY 13
Obrázek 5 - Ověření správnosti výpisu PUTTY 13
Obrázek 6 - Přihlašovací obrazovka internetového rozhraní 14
Obrázek 7 - IO_variables ve webovém prohlížeči 14
Obrázek 8 - Okno s možnostmi POU 16
Obrázek 9 - Vytvoření nové konfigurace 18
Obrázek 10 - Vyplněná tabulka konfigurace 18
Obrázek 11 - Lokalizace periferií v IO_variables 19
Obrázek 12 - Program main s vloženým vstupem a výstupem 20
Obrázek 13 - Přidání nové variable 21
Obrázek 14 - Vyplněná tabulka variables 21
Obrázek 15 - Výsledný program main 22
Obrázek 16 - Ukázka prostředí simulátoru 23
Obrázek 17 - Schéma modelu 24
Obrázek 18 - Návrh Stojanu na IR senzor 26
Obrázek 19 - Stojan na IR senzor v praxi 26
Obrázek 20 - Detail na konstrukci pásu 27
Obrázek 21 - Ukázka systému poháněj ící pás 28
Obrázek 22 - IR senzor překážky 28
Obrázek 23 - Senzory tvořící IR bránu 28
Obrázek 24 - Náčrt funkčnosti shazovače – model pumpa 29
Obrázek 25 - Mechanická konstrukce shazovače v praxi 29
Obrázek 26 - Detekce dokončení otáčky pomocí dorazového senzoru 30
Obrázek 27 - Použitý pulzní modulátor pro zpomalení otáček shazovače 31
Obrázek 28 - Nákladové silo 32
Obrázek 29 - Nákladové kolo 32
Obrázek 30 - Ovládací tlačítka a LED diody 33
Obrázek 31 - Semafor pro indikaci chybových stavů 33
Obrázek 32 - Napájecí zdroj 34
Obrázek 33 - Step-Down měnič 34
Obrázek 34 - Svorkovnice 34
Obrázek 35 - Funkční blok Paměť 35
Obrázek 36 - Funkční blok Analog 36
Obrázek 37 - Funkční blok Ping 37
Obrázek 38 - Funkční blok Průjezd 38
Obrázek 39 - Funkční blok Píst 39
Obrázek 40 - Hlavní program 39
Obrázek 41 - Sekce ovládající pás 41
Obrázek 42 - Sekce ovládající silo a uživatelské vstupy 41

Seznam tabulek

Tabulka 1 - Seznam vstupních a výstupních periferií jednotky 10
Tabulka 2 - Výhody a nevýhody uvažovaných způsobů konstrukce 25
Tabulka 3 - Seznam připojení komponent na periferie řídicí jednotky 40

Seznam zkratek

CAD - Computer-aided design – neboli proces návrhu objektu pomocí počítače, v našem případě je použit při vytváření 3D objektů určených pro následný 3D tisk

FBD - Function Block Diagram – způsob programování pomocí kreslení funkčních bloků a hradel

IR - Infrared – technologie některých použitých senzorů, které pracují s infračerveným světlem

PLC - Programmable logic controller – označuje druh průmyslové řídicí jednotky

POU - program organization unit – segment programu definovaný v normě IEC 61131-3

ST - Structured Text – způsob programování pomocí psaní kódu

1. Úvod

Ať už pracujeme s malými domácími spotřebiči či velkými výrobními linkami, vše kolem nás je dnes řízeno stroji. Počítače a mikrokontrolery dokážou zastat většinu těchto procesů. Proč je tomu ale v průmyslovém řízení jinak? Řízení průmyslových linek má totiž svá vlastní specifika. Téměř vše se v tomto odvětví odvíjí od jednoho klíčového parametru – spolehlivosti. Velká výrobní linka si nemůže dovolit žádné chyby ani výpadky, ty by totiž mohly provozovatele stát mnohdy až astronomické částky. Proto se v průmyslů používají velmi specifické řídicí jednotky – PLC (Programmable logic controller), a právě jimi se má práce bude zaobírat. Konkrétně budu pracovat s PLC jednotkou IPLOG od české společnosti Metel [1].

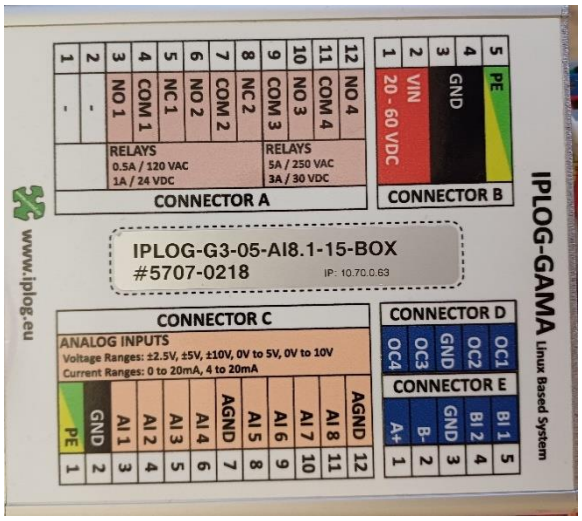
Cílem této práce je vytvořit metodický postup pro budoucí uživatele této řídicí jednotky. Díky své specifičnosti nejsou totiž jejich procesy ve volně dostupných pramenech zdaleka tak dobře popsány, jako třeba právě u mikrokontrolerů nebo počítačů. V prvním velkém celku práce projdu jednak prvotní nastavení IPLOG a také způsob jeho programování. V druhé části práce se poté zaměřím na mnou vytvořený model, který je touto jednotkou řízen. Vysvětlím jednotlivé prvky jeho mechanické konstrukce, použitou elektroniku a popíši vytvořený program, pomocí kterého jednotka IPLOG můj model řídí.

2. Způsob využití řídící jednotky

V této kapitole projdu metodický postup popisující dle vlastních zkušeností prvotní nastavení jednotky a způsob jejího využití, respektive jejího programování.

2.1. Seznámení s fyzickými vlastnostmi řídící jednotky

Nejdříve se seznámíme s důležitými fyzickými vlastnosti použité řídící jednotky, jejichž znalost je nezbytná pro její snadnou obsluhu a provoz. Na obrázku 1 vidíme horní stranu naší řídící jednotky.



Obrázek 1 - Horní strana řídicí jednotky IPLOG

Zde najdeme všechny důležité informace pro konfiguraci řídící jednotky jako výchozí IP adresa – v našem případě 10.70.0.63, přesné produktové číslo jednotky – AI8.1-15 a popis jednotlivých portů. Porty a jejich popis jsou pro přehlednost uvedeny v tabulce 1.K jednotce IPLOG je v případě potřeby možné dokoupit rozšiřující moduly s dalšími periferiemi různého typu. V našem případě se však budeme snažit vystačit s porty, které jsou dostupné přímo na samotné řídicí jednotce. Na boční straně jednotky (Obr.2) poté můžeme vidět ethernetový komunikační port, porty pro možnou instalaci antén a také signalizační LED diody jednotlivých periferií řídicí jednotky.

Tabulka 1 - Seznam vstupních a výstupních periferií jednotky

Connector	Port	Název	Funkce
A	3	NO 1	Výstup pro sepnuté relé 1
A	4	COM 1	Vstup pro relé 1
A	5	NC 1	Výstup pro nesepnuté relé 1
A	6	NO 2	Výstup pro sepnuté relé 2
A	7	COM 2	Vstup pro relé 2
A	8	NC 2	Výstup pro nesepnuté relé 2
A	9	COM 3	Vstup pro relé 3
A	10	NO 3	Výstup pro sepnuté relé 3
A	11	COM 4	Vstup pro relé 4
A	12	NO 4	Výstup pro sepnuté relé 4
B	1-2	VIN	Napájecí port pro řídící jednotku
B	3-4	GND	Zemnící port
B	5	PE	Uzemnění vodivého obalu řídící jednotky
C	1	PE	Uzemnění
C	2	GND	Zemnící port
C	3-6	AI 1-4	Analogové vstupy 1-4
C	7	AGND	Referenční zem pro analogové vstupy 1-4
C	8-11	AI 5-8	Analogové vstupy 5-8
C	12	AGND	Referenční zem pro analogové vstupy 5-8
D	1,2,4,5	OC 1-4	Výstupy typu otevřený kolektor 1-4
D	3	GND	Zem pro výstupy OC
E	1,2	A+, B-	Napájecí porty pro logické výstupy
E	3	GND	Zem logických výstupů
E	4,5	BI 1,2	Logické výstupy

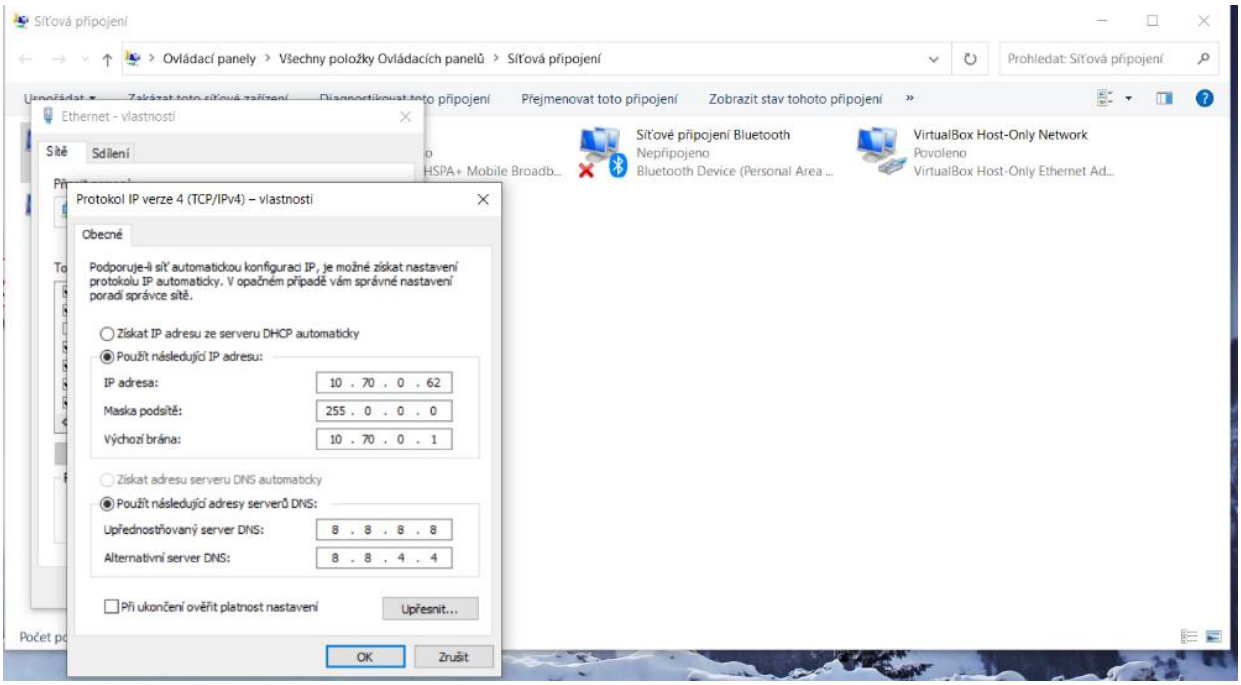
Obrázek 2 - Boční strana řídící jednotky IPLOG [1]

2.2. Prvotní nastavení IPLOG

V této kapitole se budu věnovat krokům, které je třeba provést před samotným programováním řídící jednotky.

2.2.1. Nastavení Ethernetového rozhraní

Pro komunikaci s počítačem používá IPLOG propojení přes ethernetový kabel. Aby bylo toto připojení úspěšné, je potřeba nastavit statickou IPv4 adresu ethernetového portu na našem počítači do stejného rozsahu, jako je IPv4 IPLOG. Nejjednodušší tak bude zvolit IP adresu o 1 vyšší nebo nižší. Z předchozí kapitoly víme, že IPLOG má výchozí IPv4 10.70.0.63. Tato adresa se dá v případě potřeby i libovolně měnit. Pro naše potřeby však výchozí IP adresa zcela postačí. Já jsem zvolil IP adresu o 1 nižší tedy 10.70.0.62, viz obrázek 3.

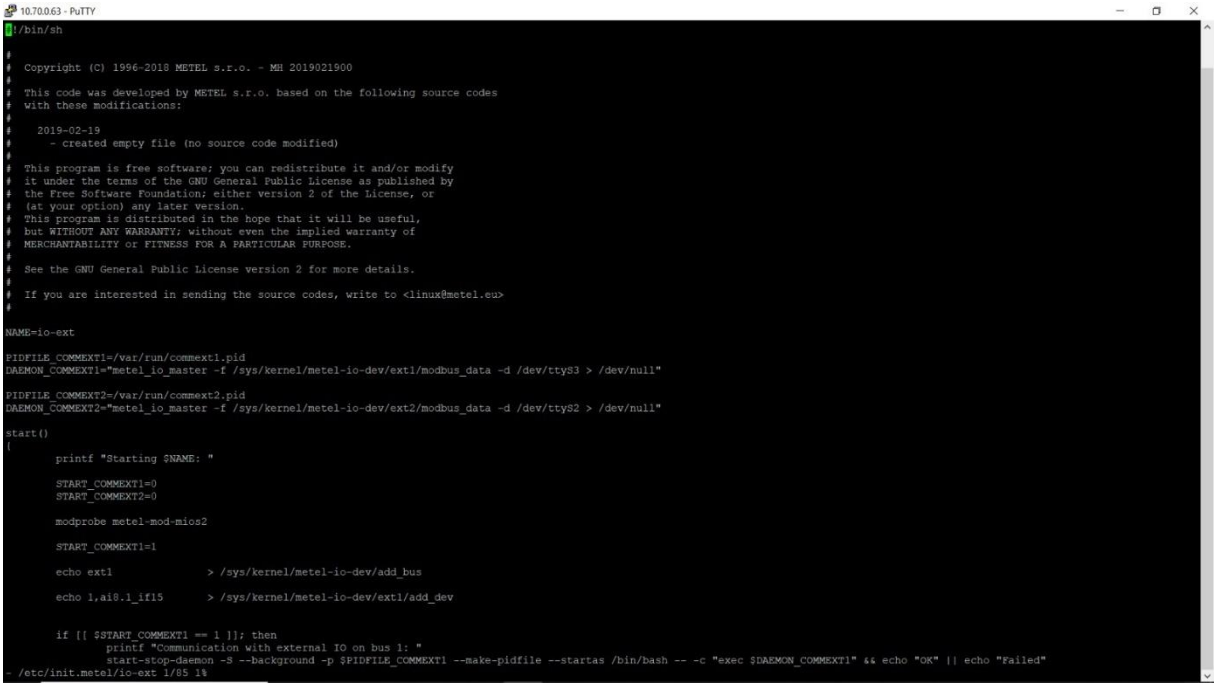


Obrázek 3 - Nastavení IP adresy

2.2.2. Nastavení generování IO pro modul

Než budeme jednotku moci programovat, musíme získat takzvaný IO soubor. Ten programovacímu studiu předá popis jednotlivých portů a jejich fungování. Pro správné vygenerování tohoto souboru je potřeba zkontrolovat a popřípadě upravit inicializační program IPLOG. K tomu budeme potřebovat produktové číslo IPLOG. V našem případě to je – viz kapitola 1 – AI8.1-15.

Nejprve je potřeba se k IPLOG vzdáleně připojit a ověřit správnost nahraného firmwaru. K tomuto účelu nám poslouží klient PuTTY, který můžeme zdarma stáhnout z [2]. Po úspěšné instalaci se díky znalosti IP adresy naší jednotky jednoduše připojíme do její vzdálené zprávy. Po připojení nás systém požádá o přihlášení, pro naše účely je na zařízení vytvořen účet „root“ bez hesla. Dalším krokem je zadání příkazu „vi/etc/init.metel/io-ext“. Zadání tohoto příkazu vidíme na obrázku 4.



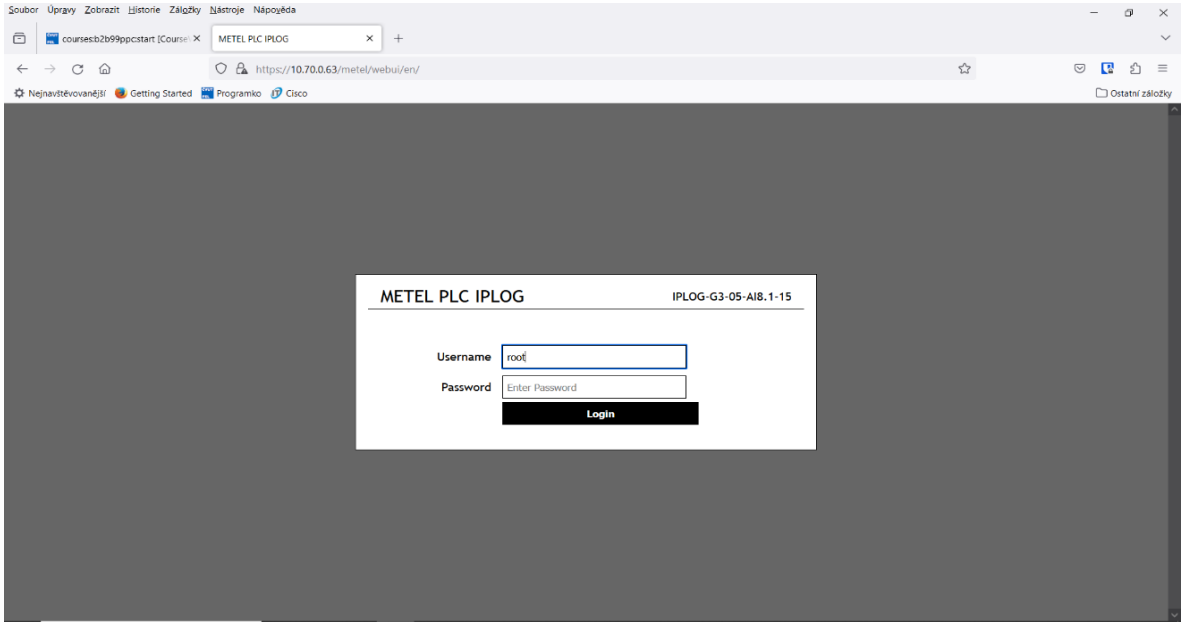
Obrázek 4 - Výpis z PuTTY

Po vypsání nás zajímá výpis na řádku začínajícím „echo 1“. Zde musíme ověřit, zda se výpis shoduje s produktovým číslem našeho zařízení. Na obrázku 5 vidíme, že v našem případě se výpis shoduje s produktovým číslem AI8.1-15. [3]

Obrázek 5 - Ověření správnosti výpisu PUTTY

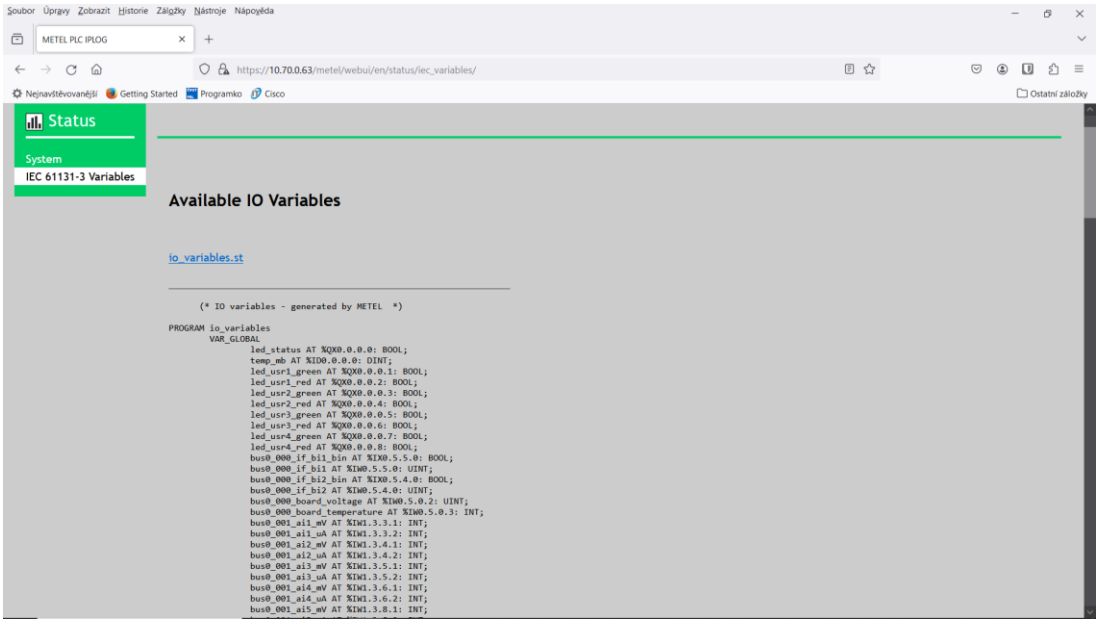
2.2.3. Získání IO souboru

Po kontrole inicializačního programu již stačí získat příslušný IO soubor. Do vyhledávače v prohlížeči zadáme IP adresu IPLOG, čímž se dostaneme na přihlašovací stránku internetového rozhraní naší řídící jednotky, kde zadáme opět jméno „root“ bez hesla. Průběh přihlášení vidíme na obrázku 6.



Obrázek 6 - Přihlašovací obrazovka internetového rozhraní

Po úspěšném přihlášení budeme mít na obrazovce 2 tlačítka. Klikneme na „Status“ a poté na „IEC 61131-3 – variables“. Zobrazí se nám výpis s hlavičkou „PROGRAM io_variables“ (Obr.7). Ten si zkopírujeme a uložíme jako textový soubor, jeho obsah budeme potřebovat v pozdější fázi. [3]



Obrázek 7 - IO_variables ve webovém prohlížeči

2.3. Instalace a úvodní orientace v programovacím studiu

V této kapitole se budu věnovat programovacímu studiu „METEL-IDE“, ve kterém se IPLOG programuje a je doporučováno a poskytnuto výrobcem. Projdu veškeré potřebné kroky od instalace programu po vysvětlení všech důležitých funkcí, které budou využity při programování mého modelu.

2.3.1. Instalace studia

Programovací studio je dostupné zdarma na stránkách výrobce [4].

2.3.2. Programování ve studiu – norma IEC 61131-3

Konkrétnímu programování se budu věnovat v následujících kapitolách. Na úvod je však dobré zmínit, že programovací jazyk studia spadá pod normu IEC 61131-3. Hlavním cílem této normy je sjednocení syntaxe jednotlivých programovacích jazyků. *Norma sjednocuje syntaxi (formální pravidla, gramatiku) i sémantiku jazyků pro programování řídicích jednotek (PLC). Lze říci, že jazyky specifikované v této normě jsou jakýmsi esperantem, které nahrazuje dosavadní množství různorodých jazyků. Uživatelé a programátoři PLC zde najdou typy jazyků, které jsou zvyklí používat, které vycházejí z jazyků používaných světovými výrobci, a bez větších problémů jim porozumějí.* [5] Tato norma rovněž popisuje, jakými způsoby můžeme programovat:

- LD (Ladder Diagram) - reléové schéma
- FBD (Function Block Diagram) - jazyk funkčních bloků
- IL (Instruction List) - jazyk mnemokódů
- ST (Structured Text) - strukturovaný text
- SFC (Sequential Function Chart) - jazyk sekvenčního programování

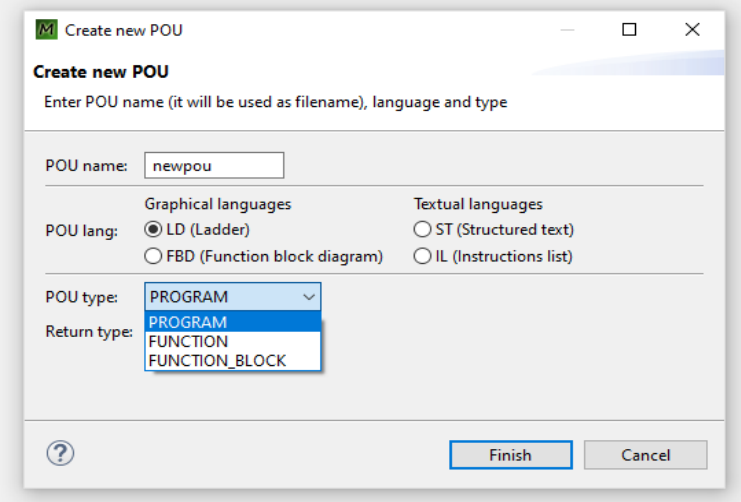
Následující kapitoly se tedy budou věnovat programování v konkrétním studiu od společnosti Metel, ale díky této normě jsou základní programovací principy přenositelné i k dalším výrobcům.

2.3.3. Založení projektu

Po úspěšné instalaci a otevření studia je potřeba založit nový projekt. To provedeme najetím myši na lištu s označením „File“ v levém horním rohu obrazovky a následným zvolením možnosti „New“ a poté „New GEB 61131-3 project“. Projekt si pojmenujeme a uložíme.

2.3.4. Založení programu a import IO_variables

Samotný projekt však ještě není program. Podobně jako v ostatních programovacích jazycích se náš program bude skládat z hlavní části „main“ a přidružených souborů, které budou sdruženy pod naším projektem. Těmto souborům se v tomto programovacím studiu říká POU. Pravým klikem myši na náš projekt se nám zobrazí nabídka, ze které zvolíme „Create new POU“. Zobrazí se nám okno možností, které je vidět na následujícím obrázku – Obr. 8.



Obrázek 8 - Okno s možnostmi POU

Než budeme pokračovat v tvoření našeho prvního programu, upřesním vlastnosti jednotlivých možností, které nám studio nabízí při tvoření POU.

- POU name – V tomto bodě máme největší benevolenci a jméno POU je zcela na nás. Jedinou výjimku tvoří POU „main“ a POU IO_variables – oběma těmto výjimkám se budu věnovat později v konkrétním popisu založení programu.
- POU lang – Jde o vybraní možnosti, jakým způsobem chceme programovat. V mém projektu se omezím na použití ST (Structured text) a FBD (Function block diagram).
    - ST – Při založení POU v této formě budeme programovat pomocí psaní kódu
    - FBD – Při zvolení této možnosti budeme POU programovat graficky pomocí propojování hradel a dalších funkčních bloků. Můj projekt bude z velké části realizován právě tímto provedením.
- POU type – Tento parametr nám určuje, o jaký typ POU se bude jednat. V mém projektu se omezím na „PROGRAM“ a „FUNCTION_BLOCK“
    - PROGRAM – Tvoří program, který se bude přímo nahrávat do řídicí jednotky.

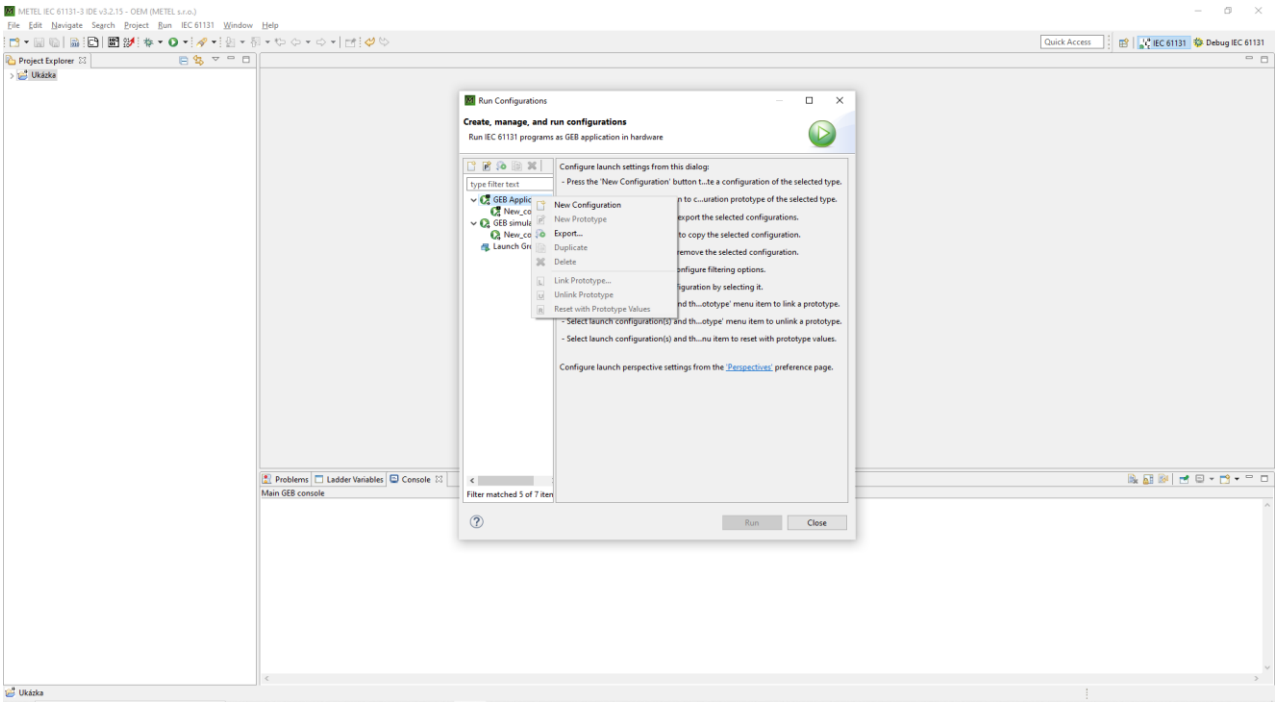
- FUNCTION_BLOCK – Tvoří část programu (funkci), která poté bude v hlavním programu zobrazena jako název tohoto POU. Velice podobné jako když vytváříme funkce v jazyce C a poté je v hlavním programu pouze voláme.

Poté co jsme prošli základní typy a charakteristiky POU, můžeme pokračovat v zakládání našeho prvního programu. Jako první chceme vytvořit právě hlavní program. Název POU tedy bude „main“. POU lang záleží na preferenci konkrétního uživatele, v našem případě ale zvolíme možnost FBD. Typ POU pak u hlavního programu bude „PROGRAM“. Tlačítkem „Finish“ poté POU vytvoříme.

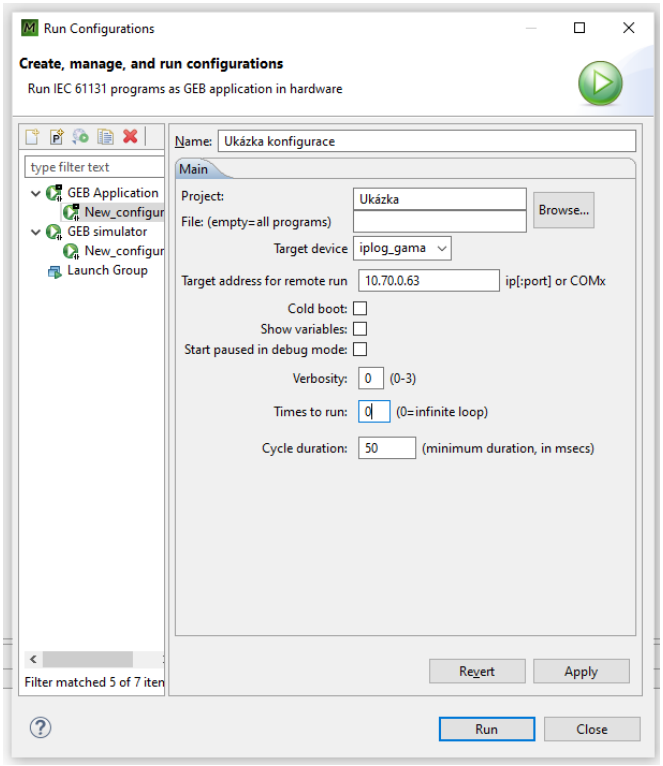
Po založení hlavního programu je ještě potřeba importovat IO_Variables, které jsme získali v kapitole 2.2.3. Pro tento účel založíme v našem projektu další POU stejným způsobem jak hlavní program, tentokrát se však bude jmenovat „IO_variables“, lang bude typu ST a typ POU bude opět „PROGRAM“. Po založení nakopírujeme obsah textového souboru, který jsme získali ve webovém rozhraní jednotky do tohoto POU a změny uložíme. POU IO_variables nám definuje jednotlivé vstupy a výstupy, kterými naše jednotka disponuje.

2.3.5. Nastavení „Run“ a „Debug“

Posledním krokem nastavující naše programovací studio je nastavení správného odesílání programu do naší jednotky. Na horní liště zvolíme „Run“, poté „Run configuration“. Pravým kliknutím na „GEB Application" můžeme vybrat možnost „New configuration“ (Obr. 9). Otevře se nám tabulka, kterou musíme správně vyplnit. Kliknutím na „Browse“ vybereme námi založený projekt, položku „File“ necháme prázdnou. Jako „Target device“ vybereme „Iplog_gama“ a do položky „Target address for remote run“ vyplníme IP adresu naší jednotky – v našem případě 10.70.0.63. Posledním bodem je nastavit položku „Times to run“ na hodnotu 0 čímž docílíme nekonečného běhu našeho programu po nahrání do jednotky. Vytvořenou konfiguraci potvrdíme tlačítkem „Apply“. Tím jsme dokončili prvotní konfiguraci našeho studia.



Obrázek 9 - Vytvoření nové konfigurace



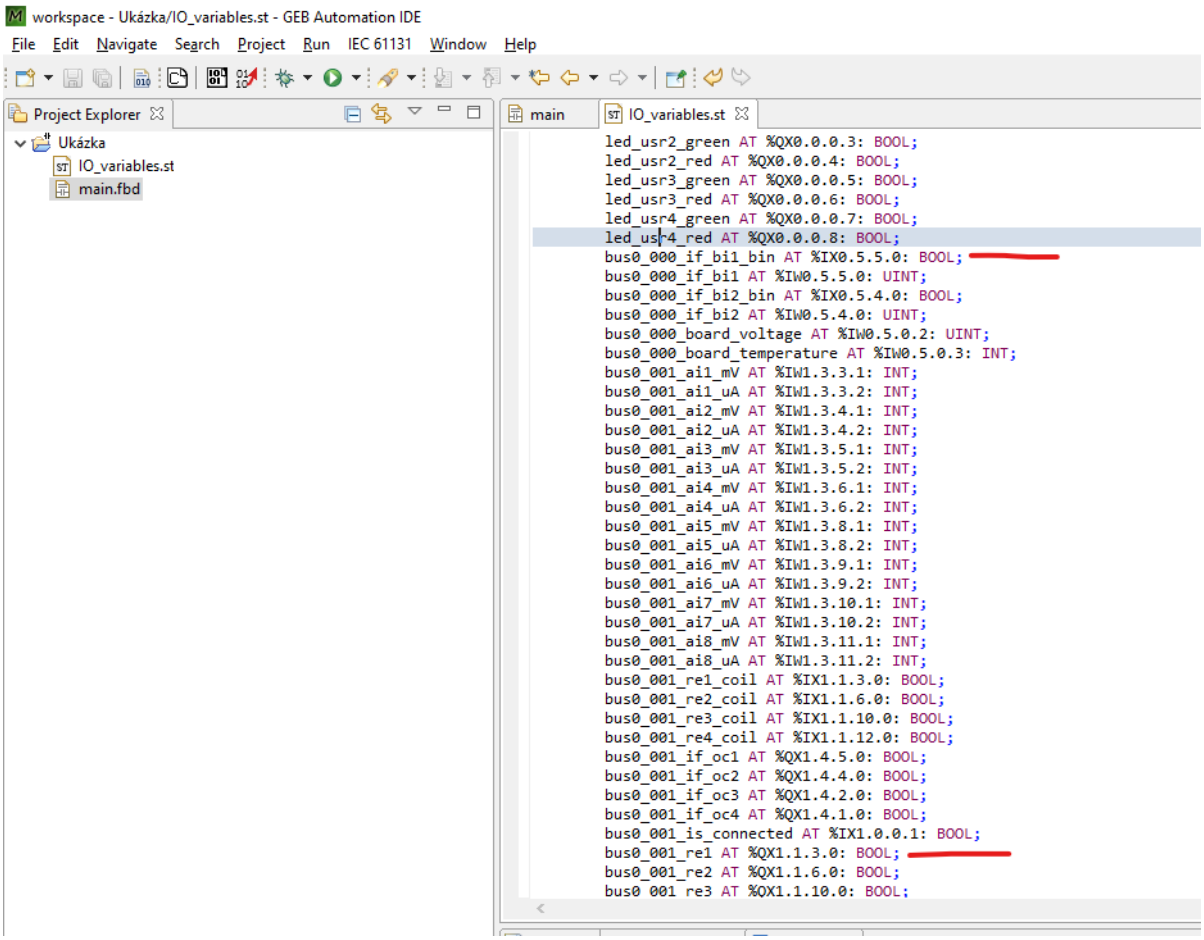
Obrázek 10 - Vyplněná tabulka konfigurace

2.4. Základní programování v METEL-IDE

V poslední kapitole teoretické části bych chtěl přiblížit proces programování v programovacím studiu. Vytvořím zde jednoduchý program převádějící signál z jednoho digitálního vstupu jednotky na jeden z jejích digitálních výstupů, na kterém budu demonstrovat proces programování v tomto studiu.

2.4.1. Naprogramování hlavního programu

V minulé kapitole jsme si založili projekt a program, který využijeme k naší demonstraci. V hlavním programu typu FBD vlastně kreslíme obvod, který dává vstupy a výstupy jednotky do vzájemného vztahu, který potřebujeme. Prvním krokem je definovat s jakými periferiemi jednotky budeme pracovat – pro naše účely si vybereme digitální vstup BI 1 a výstup relé RE 1. Tyto dvě periferie musíme najít v seznamu IO_variables, abychom zjistili přesnou syntax jejich označení. Jejich nalezení je možné vidět na obrázku 11, pro naše další potřeby je důležitá část před klíčovým slovem „AT“. Klíčové slovo „BOOL“ nám říká, že se jedná o periferie digitálního typu.

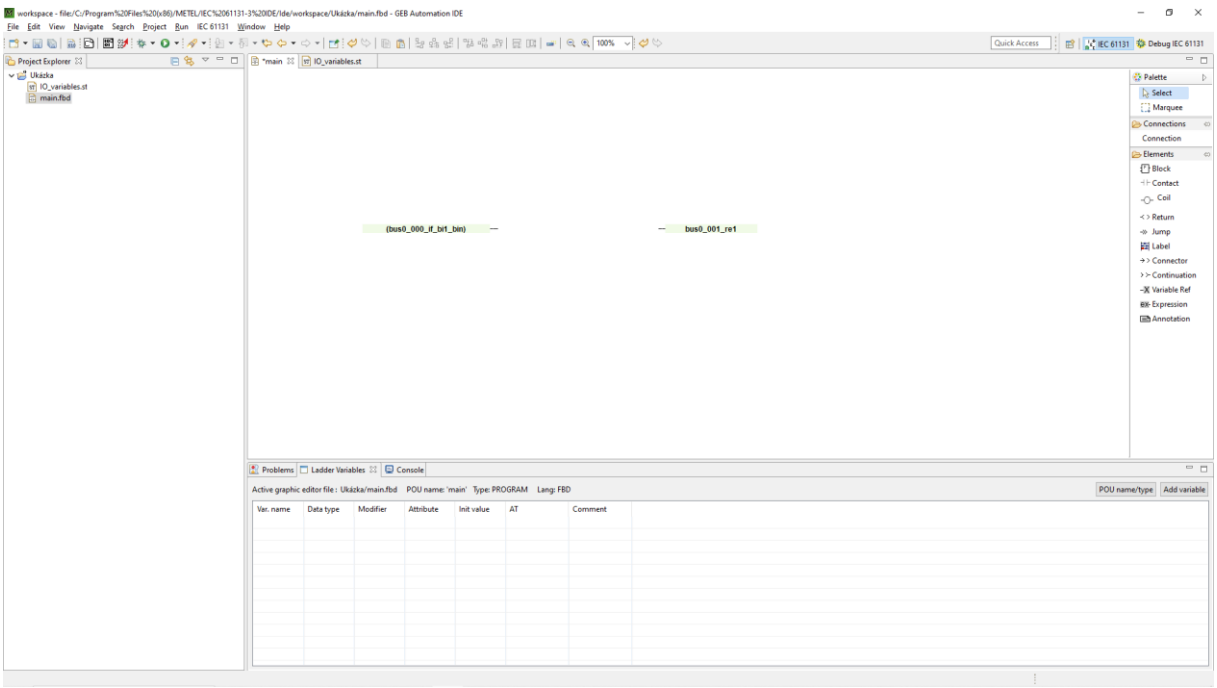


Obrázek 11 - Lokalizace periferií v IO_variables

Po úspěšném zjištění přesné syntaxe námi vybraných portů se vrátíme do programu main a můžeme začít programovat – pro tyto účely nám poslouží hlavně menu „Elements“ na pravé straně obrazovky. Z tohoto menu budou pro naše účely důležité 3 položky:

- Block – tímto tlačítkem vkládáme funkční bloky a hradla do našeho programu
- Variable Ref – touto možností vkládáme výstupní periferie našeho programu
- Expression – touto volbou definujeme vstupní periferie programu

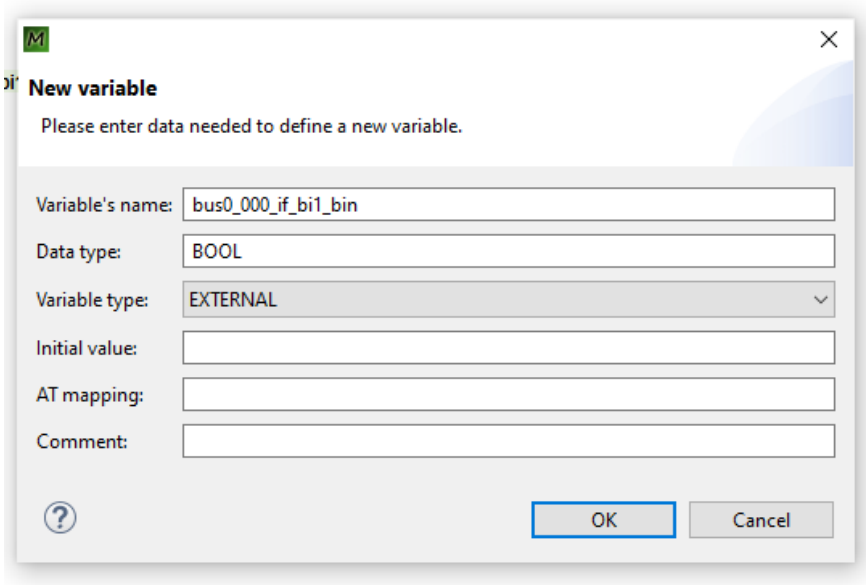
Jako první vložíme do programu právě vstup a výstup pomocí dříve zmíněných tlačítek. Zvolíme možnost v menu „Elements“ a klikneme na bílou kreslící plochu do místa, kam chceme daný element umístit. Pro umístění zmáčkneme levé tlačítko myši a program se nás zeptá na jméno našeho elementu. Pro vstupy a výstupy se jméno musí shodovat se syntaxí, kterou jsme našli v IO_variables. Program main s vloženým vstupem a výstupem můžeme vidět na obrázku 12.



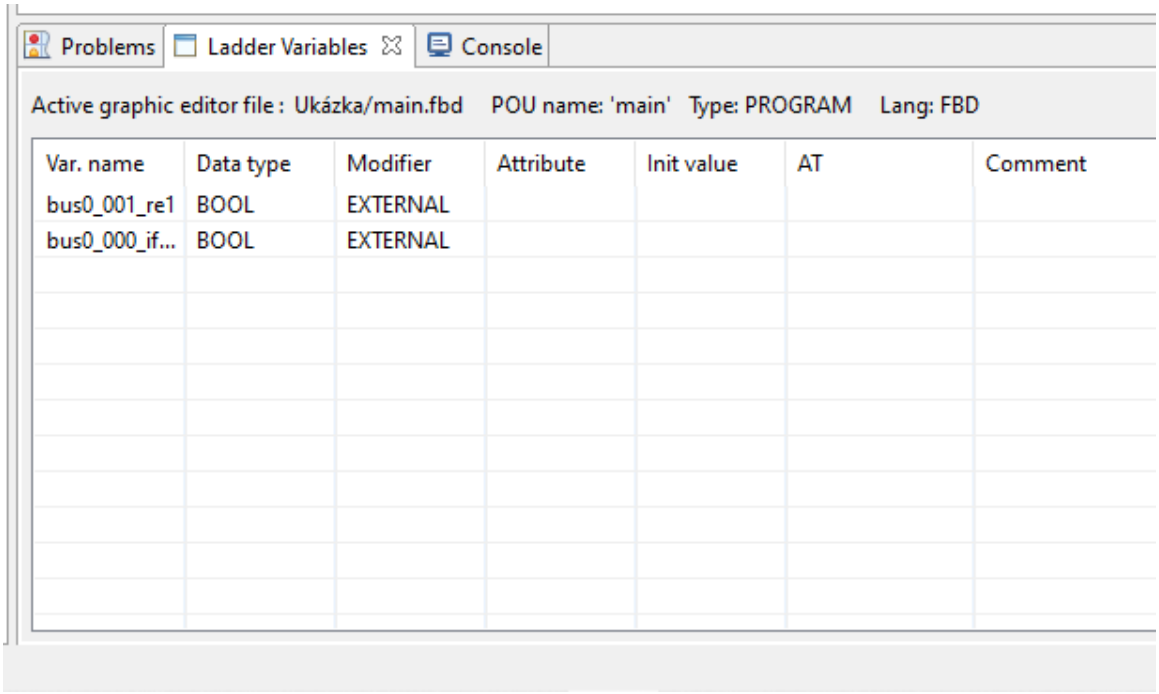
Obrázek 12 - Program main s vloženým vstupem a výstupem

Dále musíme tyto elementy definovat jako externí vstupy a výstupy – tedy jako konečné periferie řídicí jednotky. Pod kreslící plochou si překlikneme na tabulku s názvem „Ladder Variables“ a klikneme na „Add variable“. Zobrazí se nám tabulka (Obr. 13). Do první kolonky „Name“ musíme opět překopírovat jméno periferie tak, aby se shodovalo jak s programem, tak IO_variables. U kolonky „Data type“ musíme vybrat datový typ rovněž se shodující s IO_variables – v našem případě tedy „BOOL“. Jako „Variable type“ vybereme možnost „External“, která nám říká, že se jedná o fyzický vstup/výstup jednotky. Kliknutím „OK“

potvrdíme náš výběr a tím přidáme variable do tabulky. Stejným způsobem postupujeme i u druhého elementu. Tabulku „Variables“ po přidání obou elementů vidíme na obrázku 14.

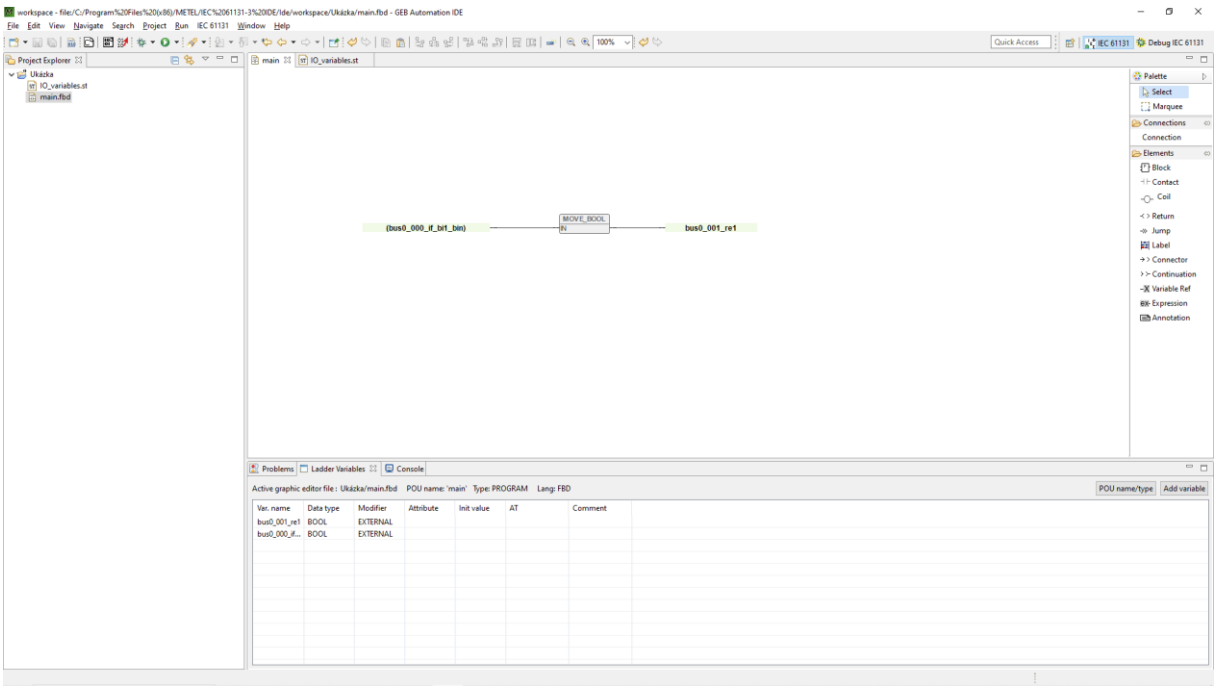


Obrázek 13 - Přidání nové variable



Obrázek 14 - Vyplněná tabulka variables

Tímto jsme definovali oba elementy jako vstupy a výstupy programu. Teď už nám jen stačí poslat ze vstupy signál na výstup. V menu „Elements“ zvolíme „Block“ a vyhledáme blok „MOVE_BOOL“ a vložíme jej mezi vstup a výstup. Tento blok jen na výstupu replikuje signál ze vstupu, a proto je pro naše potřeby ideální. V základní knihovně je však předdefinovaných bloků nespočet. Všechny dostupné bloky a jejich popis je k dispozici z [6]. Poté nám jen stačí propojit vstup tohoto bloku s naším vstupem a jeho výstup s naším výstupem. Tím je program hotov.

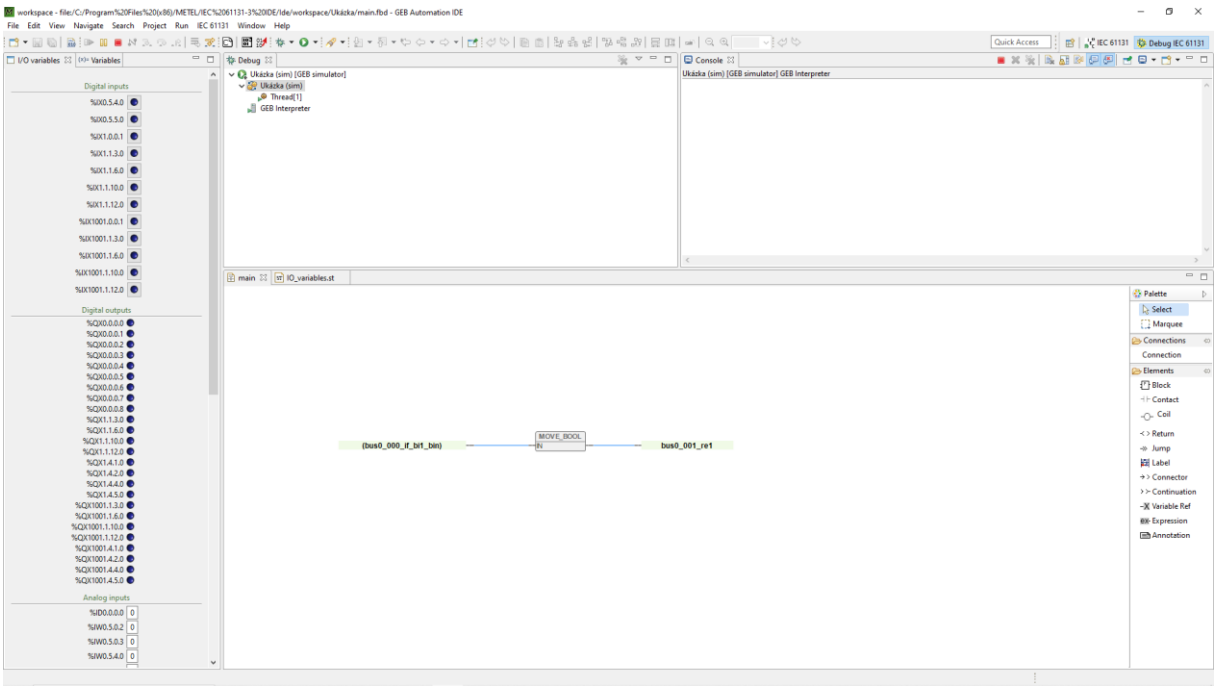


Obrázek 15 - Výsledný program main

2.4.2. Simulace a spuštění programu

Výsledný program nám již jen stačí spustit. I v této oblasti nám studio poskytuje několik možností. Pravým kliknutím na náš projekt se nám otevře menu. Hned pod tlačítkem, které jsme požívali na tvorbu POU pak máme dvě možnost „Run as“ a „Debug as“, které se obě dále větví na „GEB Application“ a „GEB Simulator“. Než si jednu z těchto možností zvolíme, rozeberu účely těchto možností:

- Debug as Simulator – program se neodešle na jednotku, ale otevře se nám jeho simulace přímo ve studiu (Obr.16). V této simulaci můžeme přímo na počítači měnit stav vstupů a výstupů a ověřit tak funkčnost programu.
- Debug as Application – program se nahraje na řídicí jednotku, ale opět se nám ukáže prostředí podobné jako v případě simulace. Se vstupy a výstupy již sice nemůžeme interagovat, ale při běhu programu na jednotce vidíme jejich stavy a také logické stavy mezi jednotlivými bloky a hradly. Program však není na jednotce uložen a při odpojení ethernetového kabelu se program zastaví. Tato možnost je tak ideální pro fyzické testování programu nebo pro hledání místa chyby při nefunkčnosti programu.
- Run as Application – program se nahraje do paměti řídicí jednotky a spustí se. Nemáme však již k dispozici žádné údaje o jeho běhu. Vhodné pro nahrání finální verze programu do řídicí jednotky.



Obrázek 16 - Ukázka prostředí simulátoru

Jakou možnost si zvolíme záleží již na našich aktuálních záměrech. Tímto jsme dokončili programování demonstračního programu a můžeme pokračovat v programování podle našich konkrétních potřeb.

3. Vlastní konstrukce

Po přiblížení metodického postupu práce s jednotkou se budu ve zbývajících kapitolách mé práce věnovat vlastnímu řešení modelu řízeného jednotkou IPLOG.

3.1. Návrh modelu a jeho konstrukce

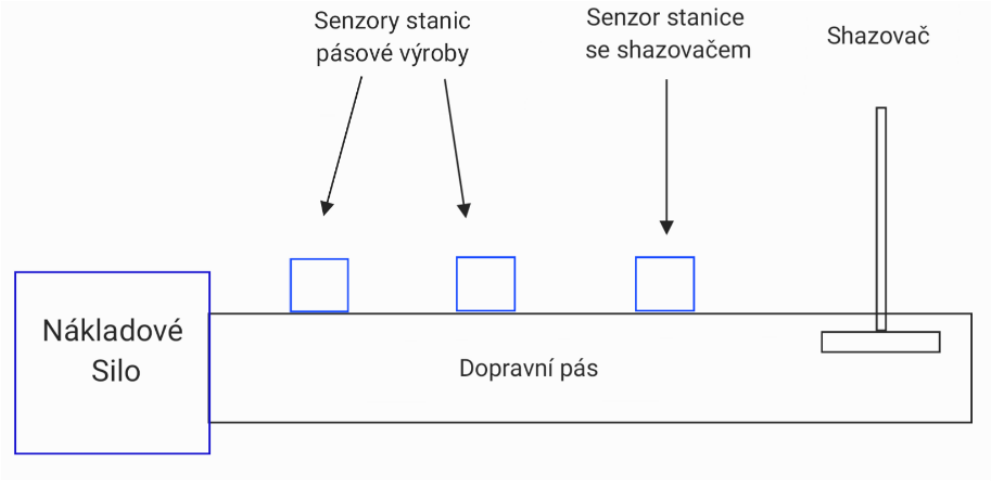
Jako první se zaměřím na problematiku výběru vhodného modelu pro demonstraci a vytvoření jeho nosné konstrukce.

3.1.1. Výběr vhodného zařízení pro realizaci

Prvním krokem projektu byl výběr vhodného průmyslového zařízení na provedení demonstrace. Při tomto výběru jsem zohledňoval několik faktorů – osobní preference a zájem, náročnost na realizaci a průmyslové využití. Po zohlednění prvních dvou faktorů jsem výběr zúžil na „Průmyslový výtah“ a „Dopravní pás – simulace pásové výroby“ z nichž mi druhá varianta přišla více splňující zmíněné třetí kritérium.

3.1.2. Fyzické vlastnosti modelu

V předchozím kroku jsme vybrali dopravní pás jako vhodný model pro naší demonstraci. Dalším krokem v realizaci byl návrh samotného modelu a jeho vlastností. Jelikož chceme demonstrovat pásovou výrobu, bude náš model představovat pásovou linku s několika zastávkami. Na začátku dopravníku bude náklad automaticky naložen na pás. Na konci pásu pak bude umístěn „shazovač“, který náklad z pásu shodí. Náčrt schématu celého modelu je vidět na obrázku 17.



Obrázek 17 - Schéma modelu

3.1.3. Výběr vhodného způsobu konstrukce

Následným krokem byl výběr vhodného způsobu nosné konstrukce modelu. Jelikož je model vlastně prototyp, byla hlavním faktorem v této kategorii nízká náročnost vytvoření konstrukce, ale také možnost její lehké změny, ke které bude v rámci postupného vývoje modelu jistě často docházet. Z tohoto důvodu je i v doporučení pro realizaci bakalářské práce navrhnuto model realizovat pomocí stavebnic Lego nebo Merkur. Zvážené výhody a nevýhody těchto možností shrnuje Tabulka 2.

Tabulka 2 - Výhody a nevýhody uvažovaných způsobů konstrukce

Lego		**Merkur**	
Výhody	**Nevýhody**	**Výhody**	**Nevýhody**
Rychlé spojování a změny konstrukce	Nižší odolnost	Vysoká odolnost	Pomalejší konstrukce a její změny – šroubování
Větší tolerance vzdáleností	Vyšší cena	Obsahuje ideální součástky pro pohyblivé části	Dírky v merkuru v jasné vzdálenosti – limit tolerance

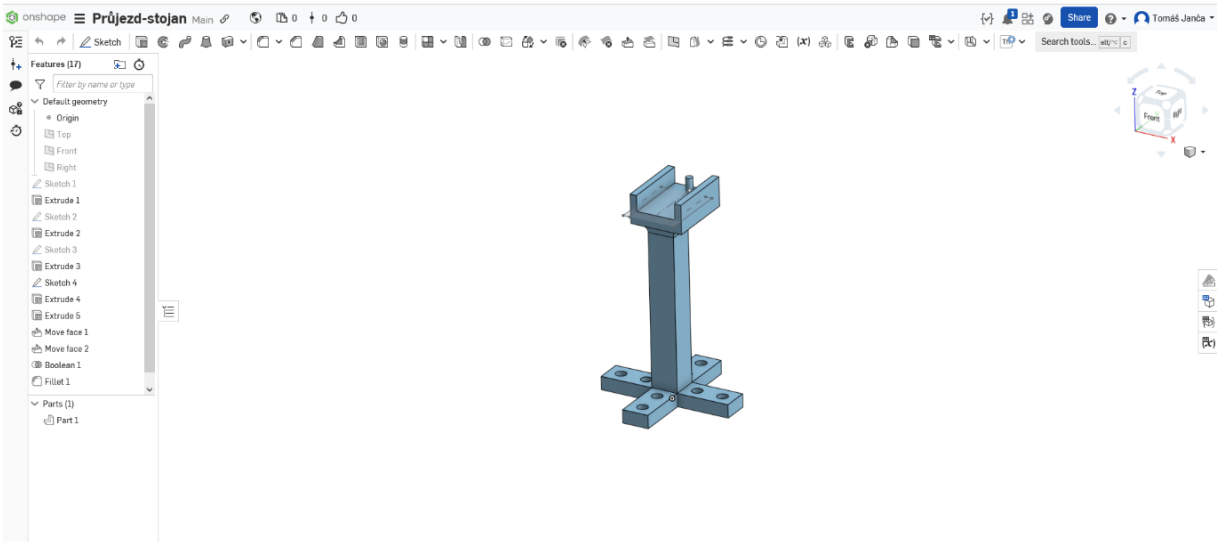
Po zvážení těchto parametrů jsem nakonec zvolil stavebnici Merkur. Ačkoliv Lego (zejména Lego Technic) také obsahuje různé hřídele a ozubená kola, robustnost těchto komponent je díky jejich plastovému provedení výrazně nižší než právě u Merkuru. Samotná konstrukce Merkuru je také výrazně blíž skutečným strojům.

3.1.4. Vytvoření podpůrné konstrukce pro elektrické prvky

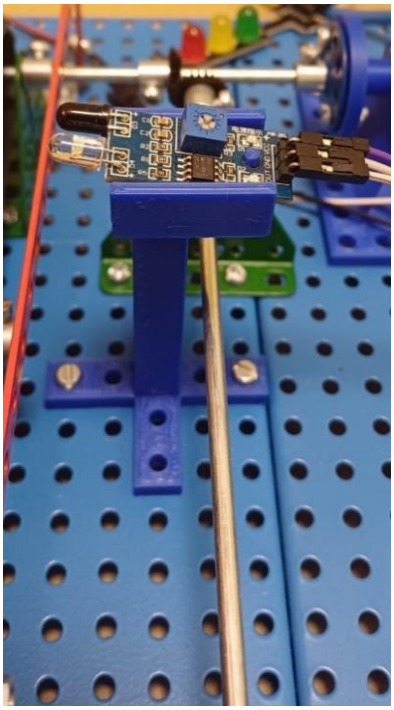
Nedílnou součástí konstrukce modelu jsou i podpůrné prvky pro elektroniku. Elektronice jako takové se budu věnovat v dalších kapitolách této práce, zde bych proto zhodnotil opravdu jen fyzickou konstrukci, na které jsou potom tyto prvky připevněny.

Při konstruování těchto částí se naplno projevil dříve zmíněný nedostatek stavebnice Merkur, a to neflexibilita co se týče vzdáleností. Merkur má všechny svoje díly dělané na celé centimetry, což v tuto chvíli přestalo dostačovat. Některé senzory musely být připevněny s tolerancí na jednotlivé milimetry, což nebylo v silách stavebnice. Bylo proto potřeba zvolit metodu, která by suplementovala Merkur v oblastech, které jsou pro něj problematické, ale musela být zároveň se stavebnicí plně kompatibilní. Jediná mně dostupná technologie splňující oba tyto požadavky byl 3D tisk. V programu CAD jsem tedy navrhl jednotlivé konstrukční prvky tak, aby byly zároveň kompatibilní se stavebnicí. Na obrázku 18 vidíme návrh jednoho

tohoto dílu – konkrétně stojanu na IR senzor, na obrázku 19 poté stejný stojan v praxi, perfektně kompatibilní se stavebnici Merkur. Symbiózu 3D tisku a Merkuru jsem poté používal napříč celým projektem a velice se osvědčila.



Obrázek 18 - Návrh Stojanu na IR senzor



Obrázek 19 - Stojan na IR senzor v praxi

3.2. Elektronicko-mechanická část modelu

Samotná fyzická konstrukce samozřejmě pro funkční model nestačí. V této kapitole tak ve stručnosti popíši elektronické a mechanické specifikace jednotlivých funkčních bloků modelu.

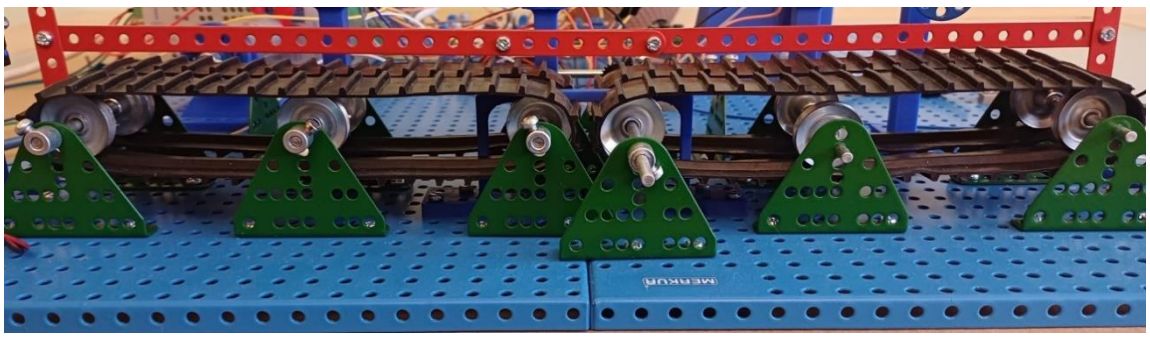
3.2.1. Konstrukce a pohon pásu, signalizace jeho spuštění

Prvním krokem samotné konstrukce byla stavba dominantního prvku modelu – pásu samotného. S tímto se pojil také výběr konkrétního gumového pásu, ve kterém se opět osvědčil výběr stavebnice merkur, jelikož bylo možno použít přímo pásy Merkur určené na modely traktorů a stavební techniky. Jediným problémem tohoto řešení je příliš krátká délka těchto pásu, a tudíž je nutné použít 2 pásy za sebou. Toto řešení se však zatím neprojevilo jako problematické.

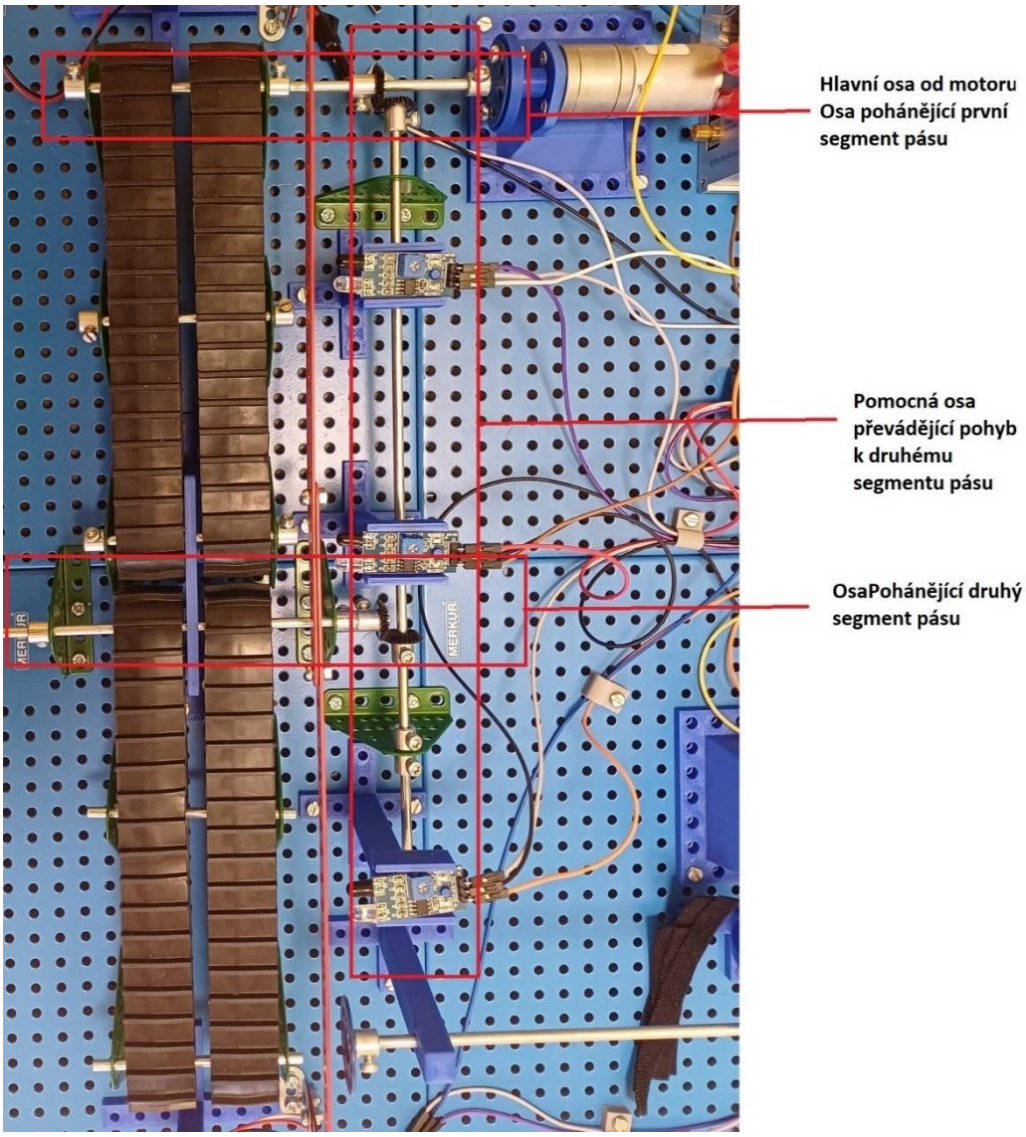
Celý pás byl tak bezproblémově zkonstruován za použití pouze Merkur dílů. Jeho konstrukci můžeme vidět na obrázku 20.

Samotný pohon pásu je zajišťován 12 V DC motorem se 60 otáčkami za minutu [7]. Pro upevnění motoru ve správné výšce byl opět využit 3D tisk, který byl také použit pro převedení točivého momentu motoru na Merkurovou soustavu os, které pohání samotné pásy. Uspořádání těchto os je možné vidět na obrázku 21.

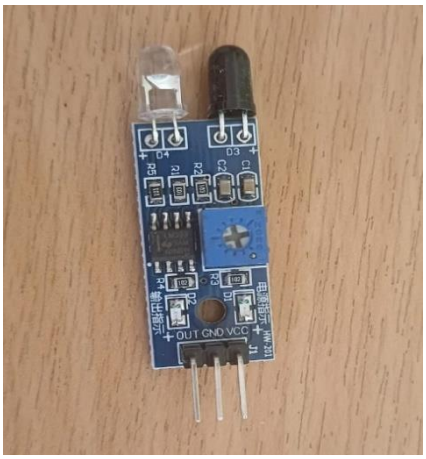
Signalizace spouštění pásu prošla dvěma prototypy. První realizace využívala IR senzor překážky [8], který měl snímat náklad na pásu – viz obrázek 22. I přes inzerovaný dosah až 40 cm nepřekročila reálná efektivní vzdálenost detekce 15 cm, což bylo zcela nedostatečné. Zvolil jsem proto způsob IR brány [9] – viz obrázek 23, kdy na jednom konci pásu je přijímač a na druhé vysílač IR světla. Pokud je proud světla nepřerušen, víme, že na pásu náklad není, dojde-li k jeho přerušení, víme že je na pásu přítomen náklad.



Obrázek 20 - Detail na konstrukci pásu



Obrázek 21 - Ukázka systému pohánějící pás



Obrázek 22 - IR senzor překážky

Obrázek 23 - Senzory tvořící IR bránu

3.2.2. Realizace stanic pásové výroby

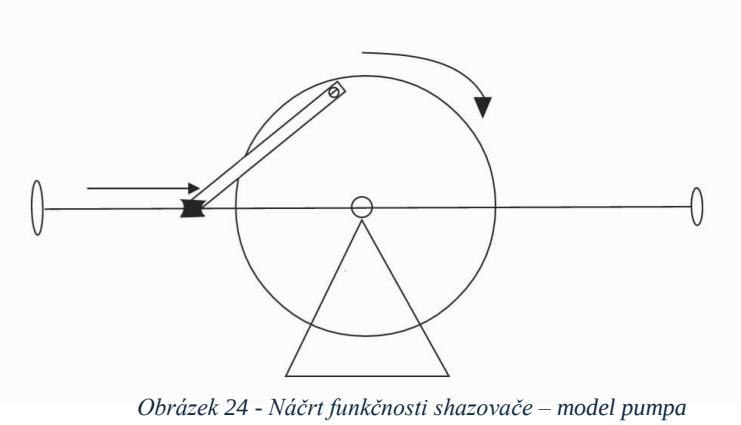
Pro jednotlivé stanice, které simulují pásovou výrobu, jsou použity IR senzory překážek [8], které jsem zmínil v předchozí kapitole. Pro detekci průjezdu nákladu je jejich omezený dosah zcela dostačující. Samotnou funkcí stanice se budu detailněji věnovat v kapitole o použitém softwaru.

3.2.3. Realizace shazovače

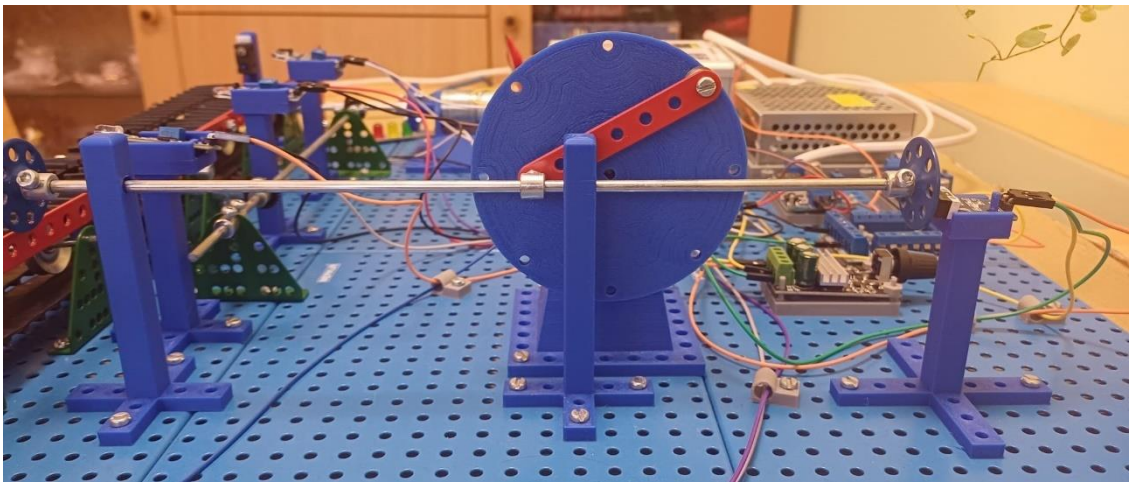
Shazovač na konci pásu je díky stylu svého pohybu nejkomplexnější mechanickou komponentou modelu. Při jeho realizaci bylo potřeba vyřešit několik stěžejních problémů:

- Převod točivého momentu na pohyb posuvný
- Správně detekovat dokončení otáčky – tj. shození nákladu
- Nastavit správnou rychlost shozu pro nepoškození nákladu.

Pro vyřešení prvního bodu jsem zvolil řešení typu „pumpa“, které můžeme vidět načrtnuto na obrázku 24. Nejobtížnější bylo zvolit správný rozměr hnacího kola, aby byl pohyb shazovače přesně tak dlouhý, jak potřebujeme. Realizaci tohoto systému pomocí 3D tisku vidíme poté na obrázku 25.



Obrázek 24 - Náčrt funkčnosti shazovače – model pumpa

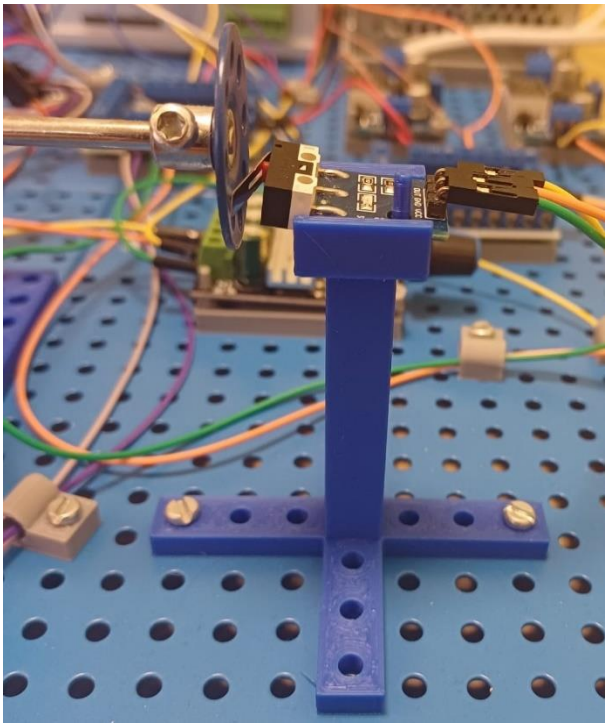


Obrázek 25 - Mechanická konstrukce shazovače v praxi

Jako řešení druhého bodu byly testovány dvě alternativy:

- Doraz konce osy shazovače na dorazový senzor
- Detekce magnetu umístěného na konci osy pomocí Halovy sondy

Po otestování těchto dvou alternativ jsem došel k závěru, že detekce magnetu není dostatečně přesná pro správné změření dokončení otáčky a zvolil jsem tedy způsob detekce dorazu. Detail na dorazový senzor [10] vidíme na obrázku 26.

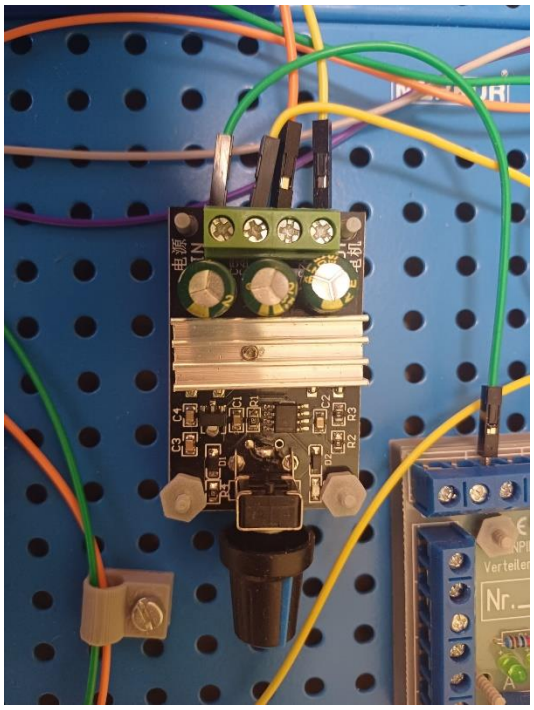


Obrázek 26 - Detekce dokončení otáčky pomocí dorazového senzoru

Pro vyřešení třetího bodu jsem opět porovnával 2 alternativy:

- Fyzická redukce rychlosti otáčení pomocí soustavy ozubených kol
- Elektronická redukce otáček za pomocí pulzní modulace

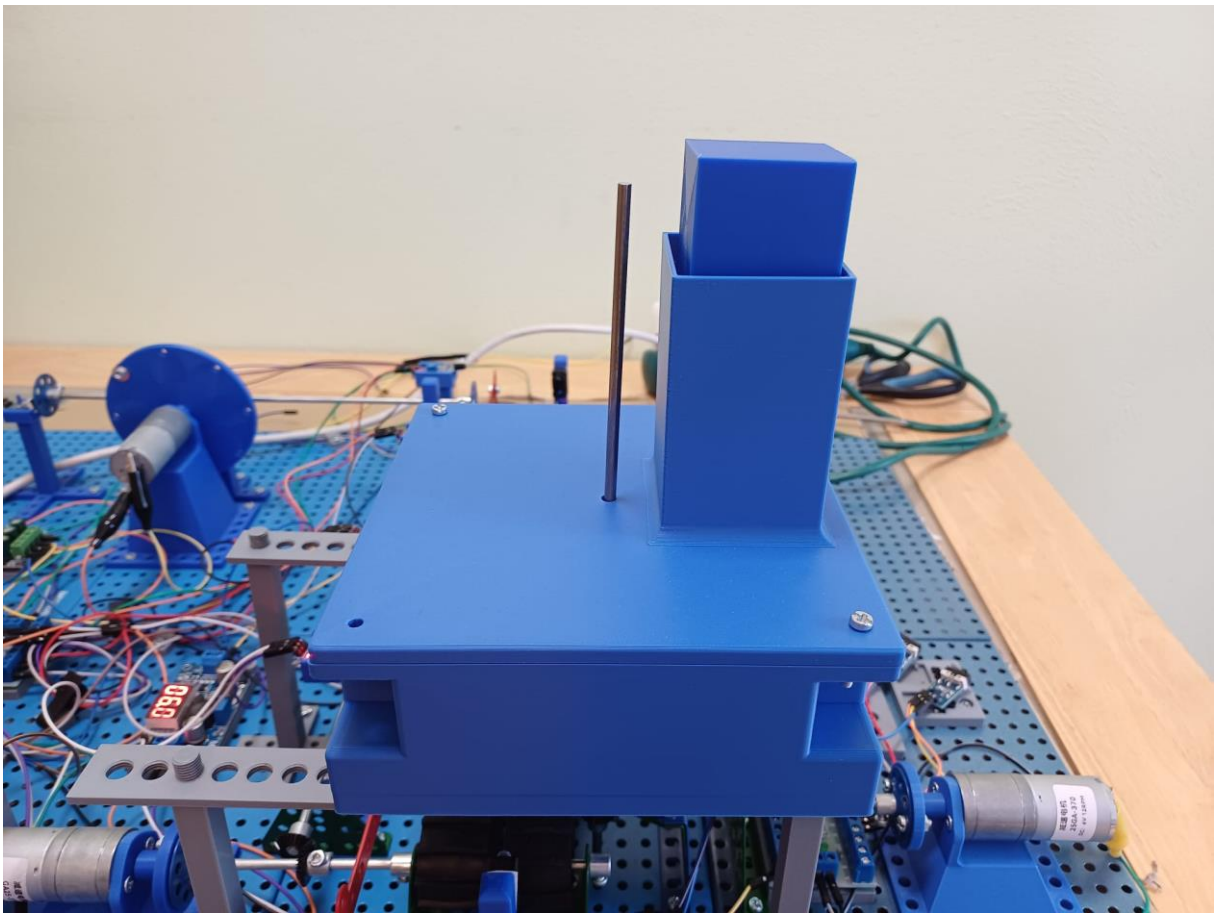
Z těchto dvou alternativ mi přijde smysluplnější varianta elektronického zpomalení motoru, jelikož je realizačně jednodušší, a také umožňuje dynamicky měnit rychlost motoru, kdežto pro změnu rychlosti u ozubených kol by se muselo opět celé ústrojí předělat. Použitý pulzní modulátor [11] vidíme na obrázku 27.



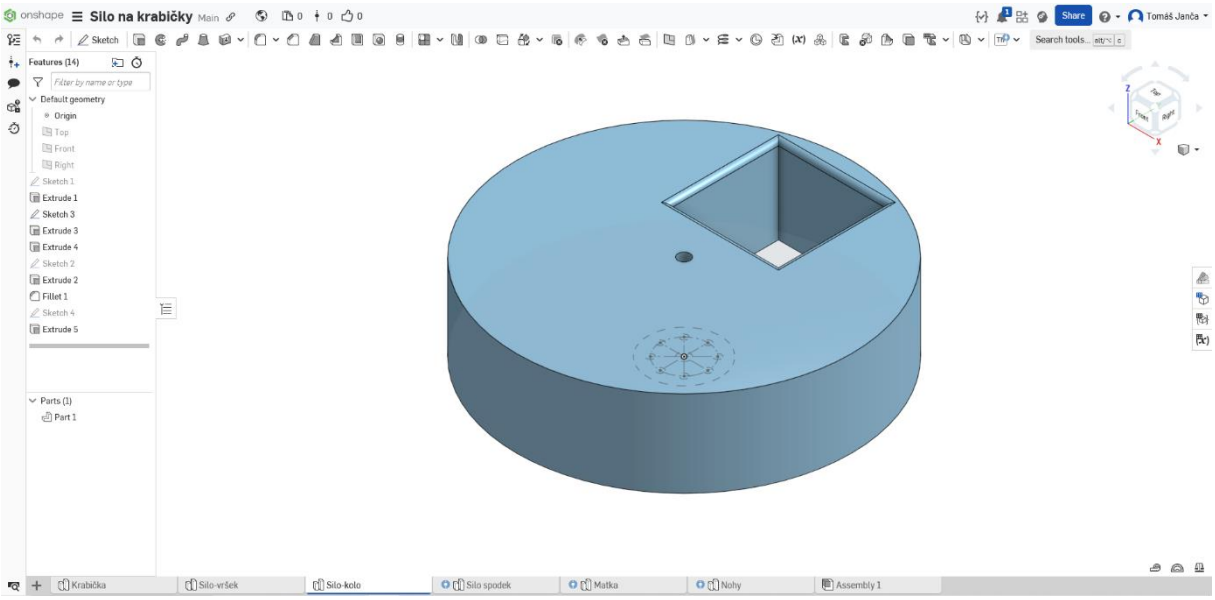
Obrázek 27 - Použitý pulzní modulátor pro zpomalení otáček shazovače

3.2.4. Realizace nakládacího sila

Při řešení automatického nákladového sytému byla největší překážkou nemožnost jednoduchého ovládání krokového motoru řídicí jednotkou. Pohybová soustava nakladače tak musela být navržena tak, aby vyhovovala spíše dlouhému plynulému pohybu DC motorů než kratším a přesným pohybům dosažitelných pomocí krokových motorů. Hlavní součástí konstrukce nakladače se tak stalo nákladové silo (Obr. 28), které využívá plynulý pohyb velkého nákladového kola, kterým přesouvá krabičky ze zásobníku na pás. Návrh nákladového kola v návrhovém prostředí je ukázán na obrázku 29. Toto kolo je poháněno 6 V DC motorem s 12 otáčkami za minutu [12], což se během testování ukázalo jako ideální kombinace rychlosti a spolehlivosti. Kromě DC motoru nepřináší silo žádnou další elektroniku a využívá informace ze senzorů, které jsou již na modelu nainstalovány pro provoz předešlých stanic. Tento proces bude podrobně popsán v kapitole věnující se softwaru modelu.



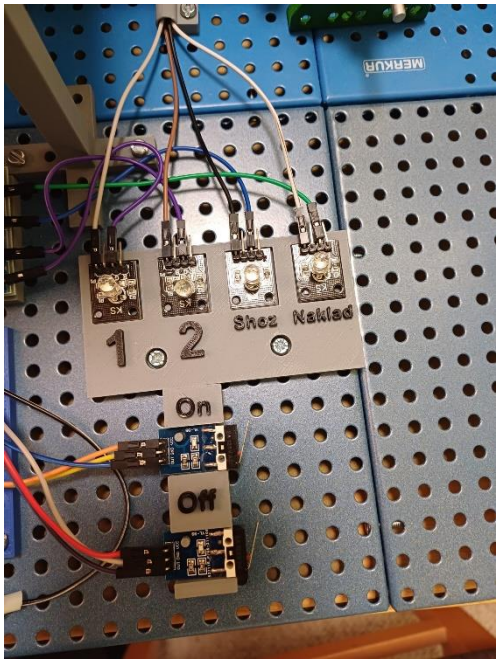
Obrázek 28 - Nákladové silo



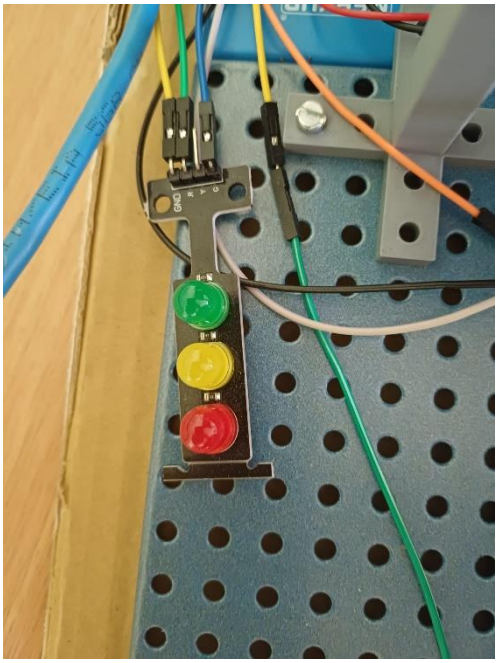
Obrázek 29 - Nákladové kolo

3.2.5. Prvky pro interakci s uživatelem

V průmyslovém provozu je velice důležité, aby obsluha měla jednak aktuální a také přesné informace o stavu stroje a mohla jej také jednoduše ovládat, proto je nezbytné vybavit model prvky pro komunikaci s obsluhou. Pro ovládání stroje jsou k dispozici dvě tlačítka [10] (Obr. 30). Pro sledování běhu stroje 4 LED diody [13] (Obr. 30) a barevný semafor [14] (Obr. 31). Tlačítka slouží k zapínaní a vypínání dopravníku. LED diody uživatele informují, jakou činnost dopravník právě vykonává a semafor nám poskytuje informace o případných chybách modelu. Za tímto účelem byl na konec pásu naistalován čtvrtý IR senzor překážky [8]. Pokud dojde k sepnutí tohoto senzoru, víme, že se náklad nachází v místě, kde by být správně neměl, a tím dostáváme chybový stav. Dopodrobna bude činnost těchto prvků popsána v příslušné části kapitoly věnující se softwaru těchto součástek.



Obrázek 30 - Ovládací tlačítka a LED diody



Obrázek 31 - Semafor pro indikaci chybových stavů

3.2.6. Další použité elektronické prvky

Krom elektronických komponent zastávajících konkrétní koncové funkce modelu byly v modelu použity i další podpůrné komponenty, bez kterých by realizace nebyla možná a kterým bych se rád věnoval v této kapitole.

3.2.6.1. Napájecí zdroj

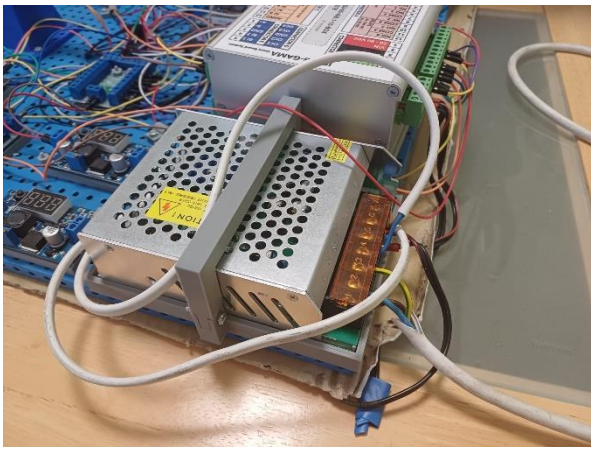
K napájení celého modelu byl využit spínaný zdroj 24 V/4,16 A [15] (Obr.32). Díky voltáži tohoto zdroje jej bylo možno využít na přímé napájení řídící jednotky i všech ostatních komponent.

3.2.6.2. Step-Down měniče

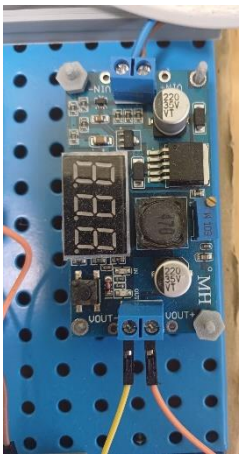
Všechny použité komponenty kromě řídicí jednotky samotné jsou však stavěny na nižší napětí než 24 V dodávaných zdrojem. Pro jejich bezproblémové zapojení byly využity step-down měniče [16] (Obr.33). Pro jejich lehčí použití a také za účelem přehlednosti jsem vybral měniče s integrovaným voltmetrem výstupního napětí. V modelu jsou využity celkem 3 měniče. Dva z nich jsou napojeny přímo na zdroj a provádí převod 24 V → 12 V, respektive 24 V → 5 V. Třetí poté provádí dodatečný převod 12 V → 6 V, který je potřebný pouze pro DC motor pohánějící nákladové silo.

3.2.6.3. Svorkovnice

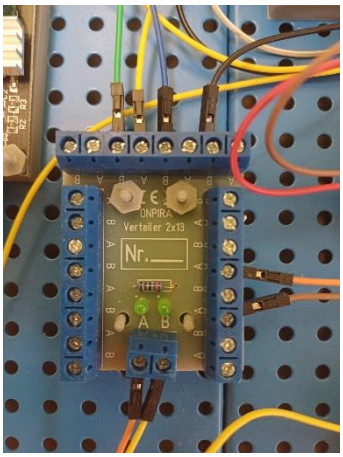
Pro jednodušší sestavování modelu jsem zvolil možnost šroubovaných svorkovnic namísto pájení jednotlivých kabelů. Použité svorkovnice [17] můžeme vidět na obrázku 34. Výstupy ze zmíněných měničů jsou vyvedeny každý do jedné svorkovnice, ze kterých jsou napájeny jednotlivé komponenty modelu. Okruh 5 V slouží k napájení všech senzorů a signalizačních LED. Okruh 12 V poté napájí dva DC motory – motor pohánějící pás a motor pohánějící shazovač.



Obrázek 32 - Napájecí zdroj



Obrázek 33 - Step-Down měnič



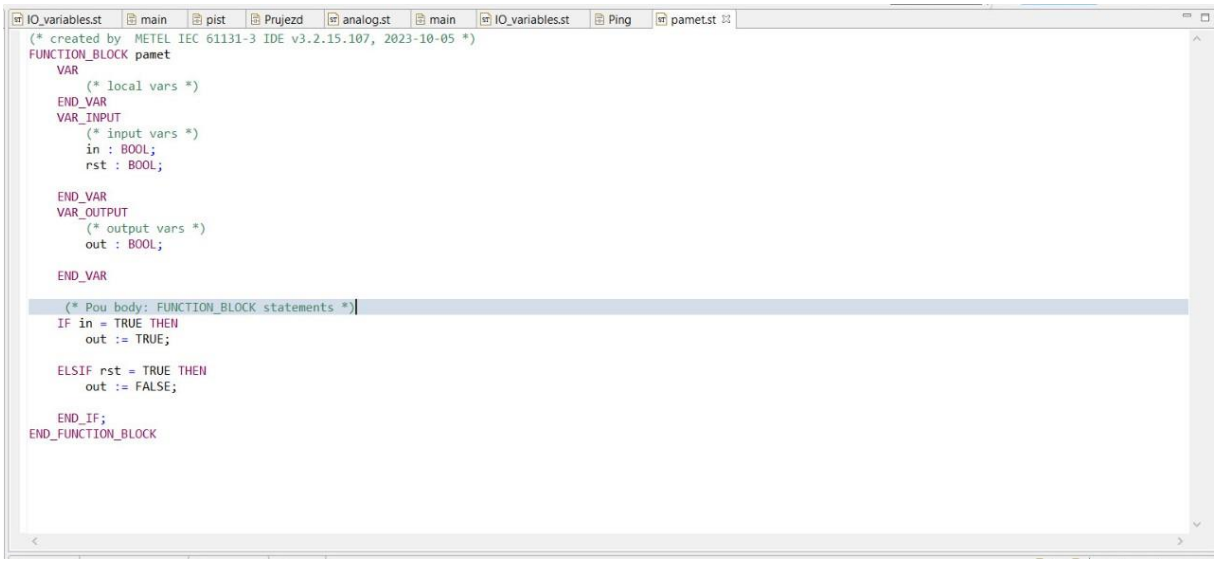
Obrázek 34 - Svorkovnice

3.3. Řídící program modelu

Ani nejlepší elektronika by nemohla fungovat bez řídicí jednotky a programu, který ji ovládá. V této kapitole nastíním řídící software, který jsem pro dopravník vytvořil. Nejprve projdu programy jednotlivých vlastních funkčních bloků, které spolu ve výsledku tvoří hlavní řídicí program modelu. Ten celkově shrnu ke konci této kapitoly.

3.3.1. Funkční blok Paměť

Blok „Paměť“ je jeden ze dvou bloků v mém projektu, který je realizován metodou ST (Structured text), tedy přímým psaním kódu. Jeho syntaxi můžeme vidět na obrázku 35. Jedná se o pomocný blok, který přímo neovládá žádnou komponentu modelu, ale jehož funkce je důležitá při realizaci dalších funkčních bloků.



Obrázek 35 - Funkční blok Paměť

Blok „Paměť“ disponuje dvěma vstupy (in, rst) a jedním výstupem (out). Cílem tohoto bloku je pamatovat si stav logické jedničky, dokud není vyresetován. Pokud na vstup „in“ přivedeme logickou 1, blok nastaví výstup „out“ rovněž do tohoto stavu. Výstup ve stavu logické jedničky zůstává, i když se již jednička na vstupu „in“ nenachází. K vynulování výstupu dochází až ve chvíli, kdy je přiveden signál (log.1) na vstup „rst“. [3]

3.3.2. Funkční blok Analog

Blok „Analog“ je druhým příkladem programování pomocí ST. IPLOG má bohužel pouze 2 digitální vstupy, což je pro náš model zcela nedostatečné. Funkcí bloku Analog je tak převádět signály z analogových vstupů jednotky do digitální podoby. Jeho syntaxi vidíme na obrázku 36.

IO_variables.st | main | pist | Prujezd | analog.st | main | IO_variables.st | Ping | pamet.st

```
(* created by  METEL IEC 61131-3 IDE v3.2.15.107, 2023-11-05 *)
FUNCTION_BLOCK analog
    VAR
        (* local vars *)
    END_VAR
    VAR_INPUT
        (* input vars *)
        in : INT;

    END_VAR
    VAR_OUTPUT
        (* output vars *)
        out : BOOL;
    END_VAR

     (* Pou body: FUNCTION_BLOCK statements *)
     IF in > 500 THEN
        out := TRUE;

    ELSE
        out := FALSE;

    END_IF;

END_FUNCTION_BLOCK
```

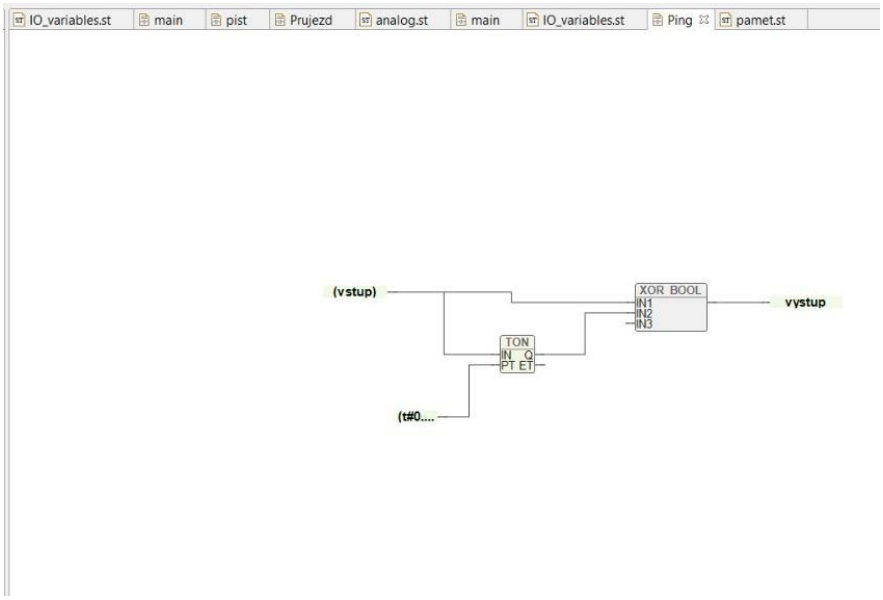
Obrázek 36 - Funkční blok Analog

Blok disponuje jedním vstupem „in“ a jedním výstupem „out“. Na vstup „in“ přichází analogový signál „INT“ – tedy číslo vyjadřující registrované napětí na analogovém vstupu jednotky v milivoltech. Blok pak má definovanou hranici (v našem případě 500 mV), podle které přepíná svůj digitální výstup do stavu log. 1 nebo log.0.

V programu se ještě nachází blok „analog_IR“. Jedná se o funkčně stejný blok jako právě Analog, jen s invertovanou logikou. Některé senzory od různých výrobců totiž pracují s opačnou logikou než ostatní. Cílem tohoto bloku je tak zjednodušení výsledného programu.

3.3.3. Funkční blok Ping

Tento blok je posledním z ryze pomocných funkčních bloků. Jeho cílem je transformovat kontinuální signál na pulz. Na rozdíl od předešlých bloků je již realizován přímo nákresem (Obr.37).



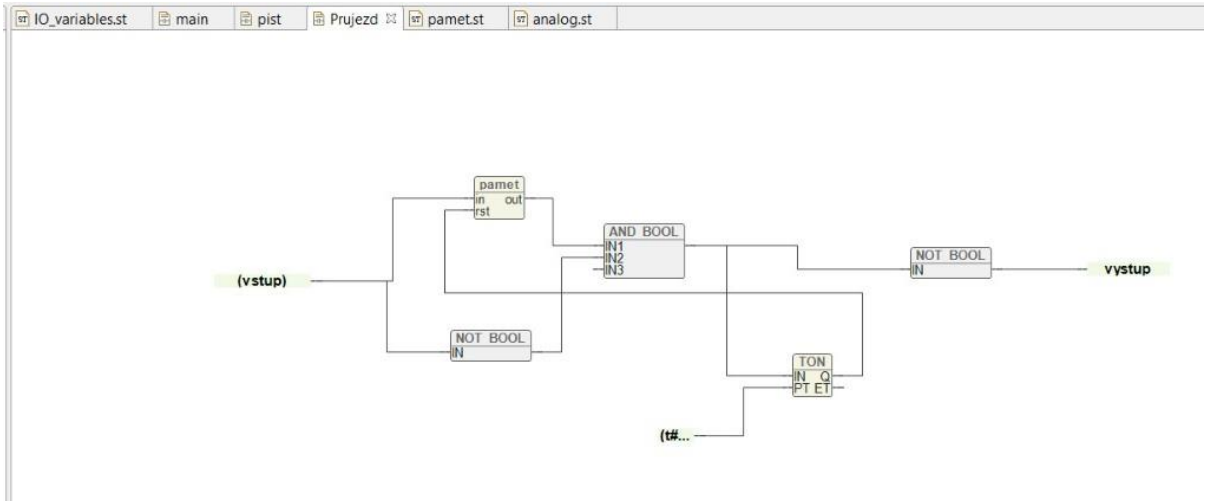
Obrázek 37 - Funkční blok Ping

Blok disponuje jedním vstupem a jedním výstupem. Dále se skládá z hradla XOR a bloku TON, který zpožďuje přechod log. 0 → log.1 o nastavený časový interval – v našem případě 0.5 s. Pokud je na vstup přiveden signál, tak právě toto zpoždění způsobí, že na půl sekundy sepne hradlo XOR a vytvoří tak žádaný pulz. Jelikož blok TON způsobuje zpoždění jen pro změnu log. 0 → log.1 a nikoliv při změně log. 1 → log.0, nevytváří tento blok nežádoucí pulz při zpětné změně logických hodnot.

3.3.4. Fukční blok Průjezd

Blok Průjezd je první funkčním blokem specificky vytvořeným pro ovládání fyzické komponenty modelu – konkrétně stanice pásové výroby. Jeho schéma můžeme vidět na obrázku 38. Tento blok zpracovává signál s IR senzoru příslušné stanice a až po projetí celého nákladu vydává pokyn k zastavení pásu. Signalizace až po celém projetí nákladu má 2 velké výhody:

- Aplikační – pás se po skončení zastávky může bezproblémově rozjet a není potřeba řešit trvající signál ze senzoru, pokud by došlo k zastavení hned při první identifikaci nákladu.
- Praktickou – díky tomuto řešení bude stanice fungovat s libovolně dlouhým nákladem, což přispívá její univerzálnosti v průmyslovém použití.



Obrázek 38 - Funkční blok Průjezd

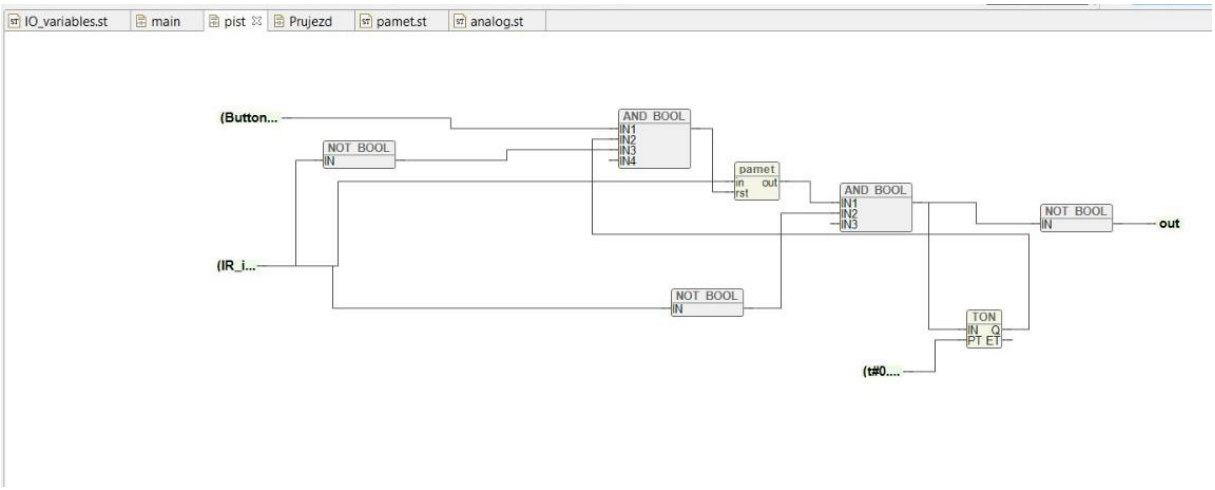
Blok disponuje jedním vstupem a jedním výstupem. Na vstup je přiveden signál z IR senzoru konkrétní stanice, svým výstupem pak blok vydává pokyn k zastavení pásu – výstup na hodnotě log. 0 znamená požadavek k zastavení pásu.

Funkce bloku probíhá v následujících fázích:

- Klidový stav – vstup má hodnotu log.0 → horní větev je také log.0, spodní (negovaná) větev má hodnotu log.1 → hradlo AND má na výstupu hodnotu log.0 → výstup má hodnotu log.1 (Pás může jet).
- Prvotní zachycení nákladu senzorem – senzor zachytil náklad, vstup se změnil na log.1 → horní větev rovněž log.1 → spodní větev však log.0 → hradlo AND zůstává neaktivní, pás stále může jet
- Náklad projel stanicí, senzor již nevidí překážku – vstup padá zpět do log.0 → horní větev díky bloku Paměť zůstává na hodnotě log. 1, spodní větev však neguje vstup – tudíž rovněž log. 1 → sepnutí hradla AND – nastavení výstupu do log.0 a tím zastavení pásu → signál z hradla AND přiveden rovněž na blok TON s nastaveným zpožděním 3s → po uplynutí nastaveného zpoždění vysílá blok TON signál na resetovací vstup bloku Paměť → reset bloku paměť, horní větev se vrací do stavu log. 0 čímž se rozepíná hradlo AND → výstup nastaven zpět na log.1 – návrat do klidového stavu.

3.3.5. Funkční blok Píst

Blok Píst je variací bloku Průjezd speciálně vytvořeným pro poslední stanici se shazovačem. U této stanice bylo třeba kromě samotné aktivace stanice a zastavení pásu rovněž kontrolovat stav a funkčnost shazovače. Strukturu bloku vidíme na obrázku 39.

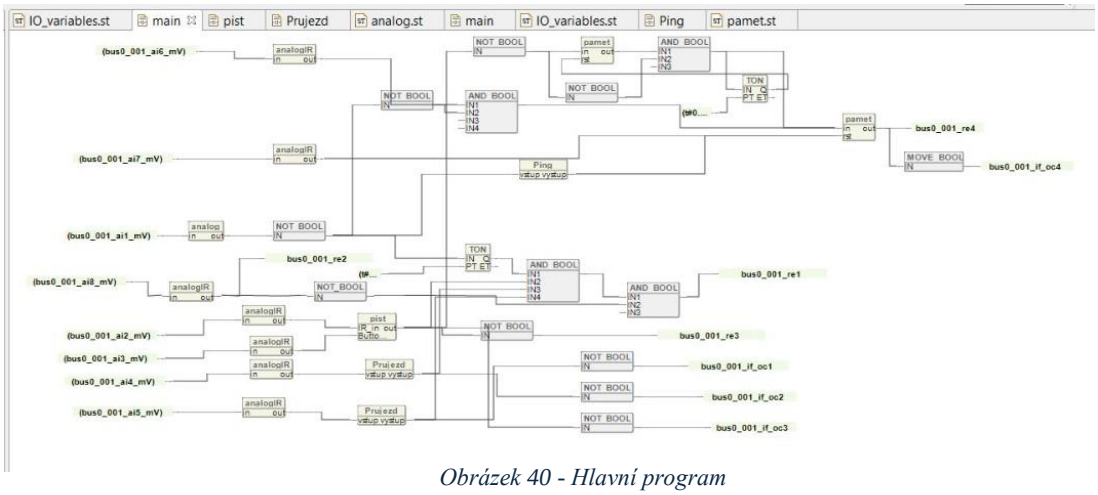


Obrázek 39 - Funkční blok Píst

Struktura i funkce bloku jsou velice podobné předchozímu bloku Průjezd. Blok má o jeden vstup navíc (Button), na který je přiveden signál z dorazového senzoru shazovače. Rozdíl funkčnosti bloku oproti Průjezdu se projevuje až v poslední fázi, kdy hlavním signálem pro reset paměťového bloku není zpožděný signál bloku z TON, ale signál z dorazového senzoru indikující návrat shazovače do výchozí polohy a tím pádem dokončení shozu. Jelikož je dorazový senzor shazovače v klidovém stavu sepnutý, je zde i přesto blok TON přítomen, tentokrát však jen se zpožděním 0.5 s, které dává shazovači čas, aby svým pohybem odepnul dorazový spínač a my tak poté mohli detekovat jeho opětovné sepnutí.

3.3.6. Hlavní program

Všechny předešle popsané funkční bloky tvoří společně s dalšími hradly výsledný program, řídící celý model. Jeho schéma vidíme na obrázku 40. Díky komplexnosti tohoto programu budu jeho funkčnost popisovat po jednotlivých blocích. Zároveň pro přehlednost propojení programu s modelem jsem vypracoval jsem tabulku (Tab.3), která ukazuje, k jakým konkrétním periferiím řídicí jednotky jsou jednotlivé probrané komponenty a funkční bloky připojeny.



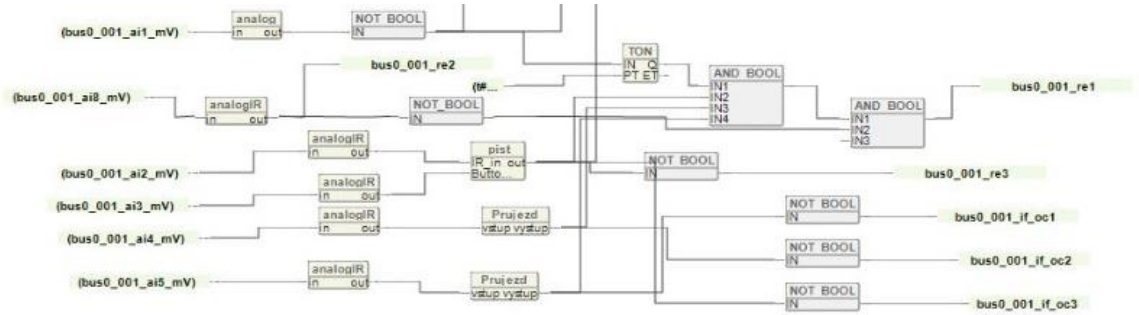
Obrázek 40 - Hlavní program

Tabulka 3 - Seznam připojení komponent na periferie řídicí jednotky

Connector	Porty	Typ periferie	Funkce v modelu
A	3-5	Relé 1	Spínač pohybu pásu
A	6-8	Relé 2	Ovládání semaforu – signalizace chyby
A	9-10	Relé 3	Spínač pohybu shazovače
A	11-12	Relé 4	Spínač pohybu nakládacího kola
C	3	Analogový vstup 1	Příjem signálu z IR brány – signalizace přítomnosti nákladu
C	4	Analogový vstup 2	Příjem signálu z IR senzoru – stanice se shazovačem
C	5	Analogový vstup 3	Příjem signálu ze senzoru dorazu u shazovače – detekce shozu
C	6	Analogový vstup 4	Příjem signálu z IR senzoru – stanice č. 2
C	8	Analogový vstup 5	Příjem signálu z IR senzoru – stanice č. 1
C	9	Analogový vstup 6	Příjem signálu z uživatelského tlačítka „ON“
C	10	Analogový vstup 7	Příjem signálu z uživatelského tlačítka „Off“
C	11	Analogový vstup 8	Příjem signálu z IR senzoru – detekce překážky na konci pásu
D	1	Výstup otevřený kolektor 4	Signalizační LED – Náklad
D	2	Výstup otevřený kolektor 3	Signalizační LED – Shazovač
D	4	Výstup otevřený kolektor 2	Signalizační LED – stanice č. 2
D	5	Výstup otevřený kolektor 1	Signalizační LED – stanice č.1

3.3.6.1. Ovládání pohybu pásu

Výřez sekce, která se stará o ovládání pohybu pásu vidíme na obrázku 41.



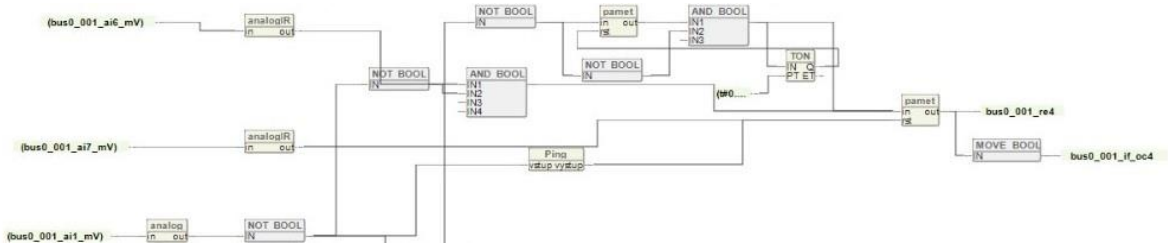
Obrázek 41 - Sekce ovládající pás

Celý pás (výstup re1) je ovládán pomocí velkého hradla AND. Na první vstup tohoto hradla je přiveden signál z IR brány, která sleduje přítomnost nákladu na pásu. Na další vstupy tohoto hradla jsou přivedeny signály z bloků Průjezd (resp. Píst) jednotlivých stanic a také signál z kontrolního IR senzoru na konci pásu. Pokud je tedy na pásu přítomen náklad a žádná ze stanic nevydala pokyn k zastavení pásu, je výstup hradla AND nastaven na log.1 a pás se pohybuje.

V této sekci také můžeme vidět vyvedení signálu ze stanice Píst k motoru shazovače (výstup re3), vyvedení signálů z jednotlivých stanic na informační LED (výstup oc1-3) a vývod signálu z kontrolního senzoru na semafor (výstup re2).

3.3.6.2. Ovládání nákladového sila a uživatelských vstupů

Druhá půlka hlavního programu má na starosti ovládání nákladového sila a zpracovávat uživatelské vstupy – tlačítka on, off. Výřez této sekce hlavního programu vidíme na obrázku 42.



Obrázek 42 - Sekce ovládající silo a uživatelské vstupy

Nákladové silo (výstup re4) je ovládáno přes blok Paměť. Pro zapnutí sila je na vstup „in“ tohoto bloku přiveden signál. To může nastat dvěma způsoby:

- Detekce dokončení shozu – v horní části programu se nachází sekce detekující shoz, tato sekce využívá stejnou logiku, která je uplatněna v blocích Průjezd a Píst. Na vstup je přiveden signál z bloku Píst (stanice se shazovačem). Sekce sleduje aktivaci a následnou deaktivaci tohoto signálu (indikující dokončený shoz předchozího nákladu) a poté vyšle signál pro aktivaci nákladového sila.
- Uživatelem – uživatel má možnost zapnout silo stiskem tlačítka On (vstup ai6). Toto tlačítko má rovněž přidanou ochranu proti uživatelské chybě. Z nákresu vidíme, že signál z tohoto tlačítka je nejdřív přiveden do hradla AND. Na druhý vstup tohoto hradla je přiveden negovaný signál ze senzorové brány. Tlačítko On je tak neaktivní, pokud se na pásu již náklad nachází.

Pro vypnutí sila je potřeba přivést signál na vstup „rst“ paměťového bloku. I zde máme dvě možnosti přivedení signálu.

- Vyložení nákladu – na vstup „rst“ je přiveden signál ze senzorové brány tudíž se silo zastavuje ve chvíli, kdy splnilo svojí funkci a senzorová brána zachytila přítomnost nákladu na dopravníku (neboli úspěšný náklad).
- Uživatelem – uživatel může kdykoliv silo zastavit stiskem tlačítka „off“.

U výstupu bloku Paměť můžeme rovněž vidět vyvedení signálu sila do příslušné informační LED (výstup oc4).

4. Závěr

V práci jsem čtenáře důkladně seznámil s řídící jednotkou IPLOG od české společnosti Metel. Nejdříve jsem popsal její důležité fyzické vlastnosti a také prošel jednotlivé kroky jejího prvotního spuštění. Čtenář byl seznámen s programovacím studiem pro danou jednotku a při naprogramování jednoduchého demonstračního programu mu byly ukázány jednotlivé možnosti a funkce, které studio nabízí. Možné použití jednotky v praxi jsem demonstroval na modelu pásového dopravníku. Pro tento model jsem po zvážení všech kladů a záporů vybral jakožto základní kámen jeho konstrukce stavebnici Merkur. Ten však v některých aspektech nedostačoval a bylo potřeba navrhnout a pomocí 3D tisku vyrobit další specifické konstrukční prvky. Krom samotné řídicí jednotky jsem do modelu instaloval velké množství další elektroniky, jako senzory různých typů, měniče, pulzní modulátor či motory. Hlavní část návrhu probíhala v rovině programování samotné jednotky. Pro můj model jsem vytvořil program, odpovídající jeho specifikám. Funkčnost modelu jsem poté ověřil důkladným testováním na modelech zpracovávaných objektů, a postupným laděním programu a úpravou časových konstant jsem zajistil spolehlivost tohoto modelu.

V práci jsem se zaměřil na důkladné vysvětlení jednotlivých myšlenkových pochodů vedoucích k úspěšnému návrhu, aby mohla tato práce posloužit dalším studentům, kteří se budou problematikou zabývat.

Reference

[1] Metel. Řada PLC BOX. *metel.eu.* [Online] [Citace: 18. Duben 2024.] https://metel.eu/products/automation_systems/plc_and_io_modules/iplog_series_box.

[2] PuTTY. *putty.org.* [Online] [Citace: 18. Duben 2024.] https://www.putty.org/.

[3] Tomašica, Tomáš. *Demonstrační model průmyslového řízení.* Praha : ČVUT v Praze, 2020.

[4] Metel. METEL IEC 61131-3 IDE. *metel.eu.* [Online] [Citace: 18. Duben 2024.] https://www.metel.eu/products/software/configuration_software/metel_iec_61131_3_ide.

[5] Urban, Ing. Luboš. *Programování PLC podle normy IEC EN 61131-3 – víc než jednotné jazyky.* [Odborný časopis] Děčín : Automa.

[6] GEB Automation IDE. *Included functions and functions blocks.* [Online] [Citace: 2. Květen 2024.] https://www.gebautomation.com/help/index.jsp?topic=%2Fcom.gebautomation.help%2Fhtml%2Foverview.html.

[7] Motor s převodovkou 25GA-370 12V 60RPM. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/1466-motor-12v-dc-60rpm-s-prevodovkou-silny-tocivy-moment-25mm.html.

[8] Infračervený senzor překážek. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/3086-infracerveny-senzor-prekazek.html.

[9] Senzor přerušení infračerveného paprsku - LED 5 mm - 0-50 cm. *Botland.cz.* [Online] [Citace: 4. Květen 2024.] https://botland.cz/pohybove-senzory/18690-senzor-preruseni-infracerveneho-paprsku-led-5-mm-0-50-cm-5904422366483.html.

[10] Koncový spínač a doraz. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/7711-koncovy-spinac-doraz.html.

[11] PWM regulátor otáček motoru DC 6V - 28V 3A. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/1141-pwm-regulator-otacek-motoru-dc-6v-28v-3a.html.

[12] Motor s převodovkou 25GA-370 6V DC 12RPM. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/120983-motor-12v-dc-12rpm-s-prevodovkou-silny-tocivy-moment-25mm.html.

[13] KY-016 RGB LED modul pro Arduino AVR, PIC, Raspberry - 3 barvy. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/1403-ky-016-rgb-led-modul-3-barvy-pro-arduino-avr-pic-raspberry.html.

[14] Modul LED semafor. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/7719-modul-led-semafor.html.

[15] Zdroj 24V 4,16A 100W spínaný SANPU EPS100-W1V24. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/74707-zdroj-24v-4-16a-100w-spinany-sanpu-eps100-w1v24.html.

[16] Step/down měnič a napájecí modul s voltmetrem - LM2596S. *Drátek.cz.* [Online] [Citace: 4. Květen 2024.] https://dratek.cz/arduino/148727-step-down-menic-napajeci-modul-a-voltmetr-s-lm2596s.html?gad_source=1&gclid=Cj0KCQjwudexBhDKARIsAI-GWYW5n9mtl-J490lrT1DCRptrftJ1X9nbApkPfrD-rYNYgbV31jB4wbwaAu1XEALw_wcB.

[17] Šroubová svorkovnice 8A. *Onpira.cz.* [Online] [Citace: 4. Květen 2024.] https://www.onpira.cz/zbozi/sroubova-svorkovnice-8a/?variantId=15560&gad_source=1&gclid=Cj0KCQjwudexBhDKARIsAI-GWYXqatDI4X09PU7mWXfC1y4I_xfCmv8NmEhobkvpY9beMvtYPoSv3eMaAmXAEALw_wcB.

Přílohy

Přílohy jsou dostupné pouze v elektronické podobě ve formátu zip.

Příloha 1

Příloha 1 obsahuje exportovaný řídicí program z programovacího studia.

Příloha 2

Příloha 2 obsahuje soubory všech 3D tištěných modelů použitých v konstrukci modelu.