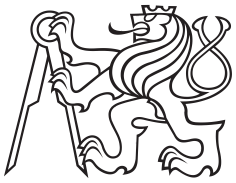
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra řídicí techniky

Řízení tepelně-hydraulického systému pomocí softwaru REXYGEN

Lukáš Navara

Vedoucí: Ing. Petr Havel, Ph.D.
Školitel–specialista: doc. Ing. Zdeněk Hurák, Ph.D.
Obor: Kybernetika a Robotika
Květen 2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Navara** Jméno: **Lukáš** Osobní číslo: **502720**

Fakulta/ústav: **Fakulta elektrotechnická**

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

Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Řízení tepelně-hydraulického systému pomocí softwaru REXYGEN

Název bakalářské práce anglicky:

Control of thermal-hydraulic system by REXYGEN software

Pokyny pro vypracování:

1. Seznamte se s řídicím SW REXYGEN, případně i dalšími SW určenými pro řízení průmyslových procesů, které mohou se senzory a akčními členy komunikovat přes protokol OPC
2. Naprogramujte regulátor, který řídí viskozitu taveniny pomocí regulace teploty v místě jejího dávkování do výrobního stroje
3. Ověřte funkčnost a spolehlivost této realizace v SW REXYGEN na simulačním modelu nebo přímo ve výrobě
4. Shrňte možnosti, vlastnosti a vhodnost SW REXYGEN pro nasazení na takovýto typ úlohy včetně sběru a archivace procesních dat a jejich zpětného zobrazení uživateli

Seznam doporučené literatury:

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Petr Havel, Ph.D. Optimwise s.r.o.

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

doc. Ing. Zdeněk Hurák, Ph.D. katedra řídicí techniky FEL

Datum zadání bakalářské práce: **29.01.2024** Termín odevzdání bakalářské práce: **24.05.2024**

Platnost zadání bakalářské práce:
do konce letního semestru 2024/2025

Ing. Petr Havel, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	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.

. _____ Podpis studenta

Datum převzetí zadání

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce, Ing. Petru Havlovi, Ph.D., za jeho odborné vedení a udávání směru během celé práce, a zároveň za poskytnutou volnost při její realizaci. Dále děkuji Ing. Filipovi Vodňanskému za jeho pomoc s integrací prediktivního řízení založeného na modelu (*MPC*). Také děkuji týmu *REX Controls s.r.o.*, zejména Ing. Pavlovi Baldovi, Ph.D., za poskytnutí licence k jejich softwaru a za rychlé a jasné odpovědi na mé nesčetné dotazy.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškerou použitou literaturu.

V Praze, 24. května 2024

Abstrakt

Tato bakalářská práce se zaměřuje na řízení tepelně-hydraulického systému pomocí softwaru *REXYGEN*. Hlavním cílem je navrhnout regulátor, který řídí viskozitu taveniny pomocí regulace teploty v místě jejího dávkování do výrobního stroje. Práce popisuje implementaci a testování regulátoru na simulačním modelu, přičemž zkoumá možnosti, vlastnosti a vhodnost softwaru *REXYGEN* pro průmyslové použití, včetně sběru a archivace procesních dat a jejich zobrazení uživateli.

Klíčová slova: REXYGEN, regulace teploty, viskozita, OPC, průmyslové procesy

Vedoucí: Ing. Petr Havel, Ph.D.
Optimwise s.r.o.
Salvátorská 931/8, 110 00 Praha 1
www.optimwise.cz

Abstract

This bachelor's thesis focuses on the control of a thermal-hydraulic system using *REXYGEN* software. The main objective is to design a controller that regulates the viscosity of the melt by controlling the temperature at the point of its dosing into the production machine. The thesis describes the implementation and testing of the controller on a simulation model, examining the capabilities, features, and suitability of the *REXYGEN* software for industrial applications, including data collection, archiving, and user display.

Keywords: REXYGEN, temperature control, viscosity, OPC, industrial processes

Title translation: Control of thermal-hydraulic system by REXYGEN software

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Cílem této bakalářské práce je prozkoumat možnosti softwaru *REXYGEN* jako nástroje pro regulaci teploty v rámci modelu *feederu* pro výrobu skelných vláken. Model simuluje teplotní chování v pěti zónách, kterými prochází roztavené sklo z pece do *feederu*. Práce se zaměřuje na vytvoření a testování regulátoru, který řídí viskozitu skla regulací teploty topnými elektrodami a využívá protokol *OPC* pro komunikaci mezi *REXYGENem* a modelem.

Pro firmu *Optimwise s.r.o.* se jedná o potenciálního kandidáta na software, ve kterém by mohla v budoucnu implementovat řídicí algoritmy v rámci realizace projektů pro klienty. Proto je podstatné touto prací prozkoumat možnosti *REXYGENu* pro tyto účely.

Práci začnu tím, že v kapitole 2 přiblížím proces výroby skelných vláken. Následně představím řídicí software *REXYGEN* v kapitole 3. V kapitole 4 vysvětlím, jakým způsobem budu řešit komunikaci mezi řídicím softwarem *REXYGEN* a modelem v *Simulinku*. Poté se budu v kapitole 5 věnovat identifikaci nelineárního systému kolem pracovního bodu. Následně v kapitole 6 navrhnu tři regulátory, které poté v kapitole 7 otestuji na třech experimentech. Na závěr shrnu možnosti zobrazení a archivace procesních dat v kapitole 8.

Kapitola 2

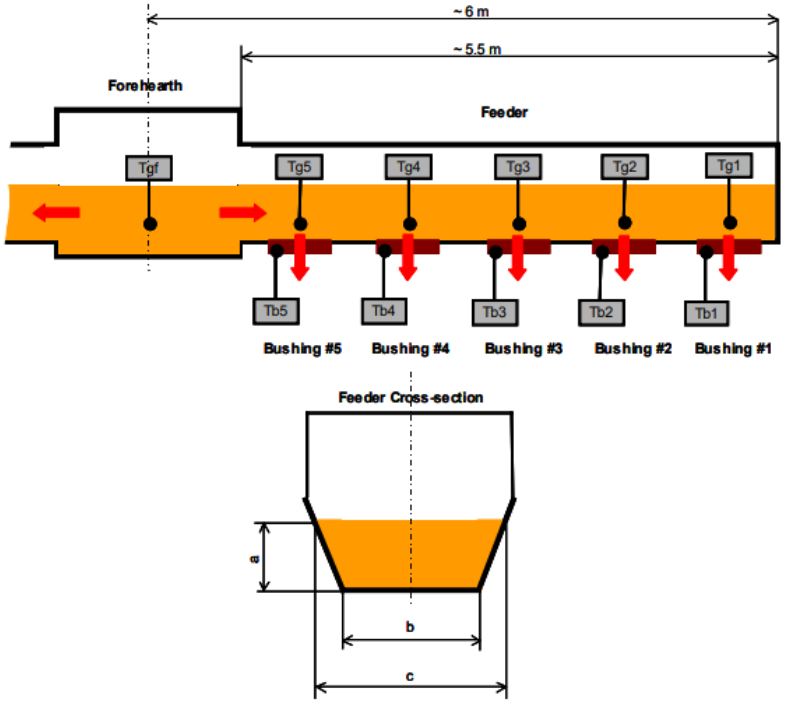
Proces výroby skelných vláken

Výrobní linka skelných vláken se skládá ze tří hlavních částí: pece, rozváděcích kanálů (*forehearths*) a *feederu* (část kanálu obsahující *bushingy*). V peci jsou suroviny roztaveny a promíchány, aby vzniklo roztavené sklo s určitou teplotou, hustotou a viskozitou. Roztavené sklo teče skrze rozváděcí kanál k *feederu*, přičemž na cestě dochází k úbytku tepla do okolního prostředí. Tento úbytek je kompenzován teplem z plynových hořáků, umístěných nad vrstvou skla, aby se udržela správná teplota a viskozita skla. *Feeder* je posledním stadiem výrobního systému, kde se z taveného skla vytahují skrz *bushingy* (platinové mřížky) vlákna. Diagram *feederu* a jedné zóny je zobrazen na obrázcích 2.1 a 2.2. Sklo *feederem* teče ze zóny 5 do zóny 1.

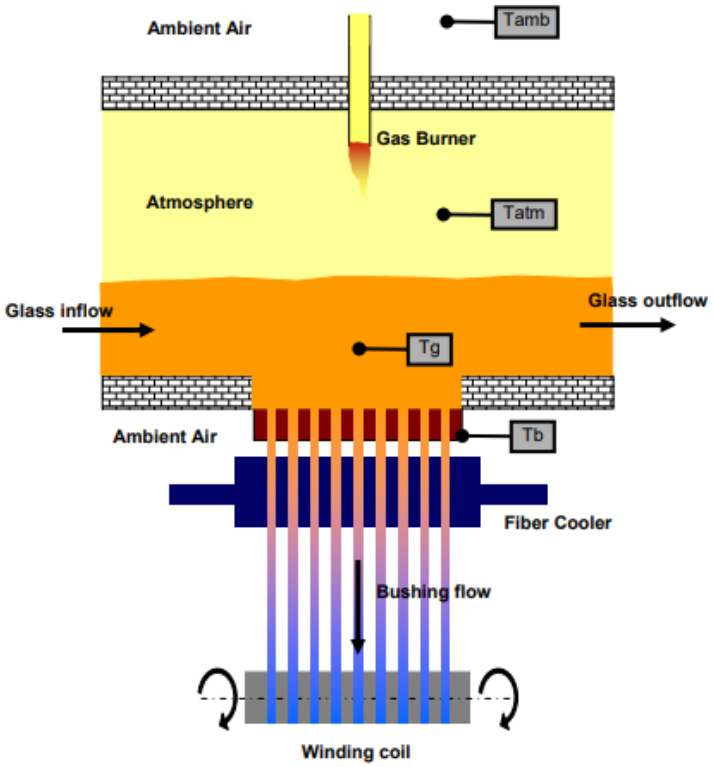
2.1 Model

Model, implementovaný v *Simulinku* v rámci diplomové práce mého vedoucího, se zaměřuje na simulaci teplotních podmínek v jednotlivých zónách. Model účinně simuluje poruchy způsobené přítokem chladnějšího skla z pece, poklesy okolní teploty a změny teploty atmosféry. Dále věrně simuluje proces výměny *bushingu*, kdy dochází k jeho ochlazení na teplotu, která zastaví tok skla, což umožňuje jeho výměnu. Model zachycuje i situace, kdy se ve *feederu* vytváří boule ze studeného skla, což vede k rozdílnému přelévání skla a má významný dopad na teplotní rozložení v systému.

Podrobnější výsledky a ověření modelu jsou diskutovány v kapitole 6



Obrázek 2.1: Diagram feederu převzatý z [1]



Obrázek 2.2: Diagram jedné zóny převzatý z [1]

RESULTS AND MODEL VERIFICATION diplomové práce mého vedoucího [1].

I když zadání bakalářské práce stanovuje návrh regulátoru pro viskozitu taveniny, v této práci se zaměřím na regulaci teploty. Důvodem je závislost mezi teplotou taveniny a její viskozitou. Vysoké teploty vedou k nižší viskozitě, což umožňuje sklu lépe téct, zatímco příliš nízká teplota zvyšuje viskozitu, což může vést k přerušení výrobního procesu kvůli přílišné tuhosti skla. Vzhledem k tomu, že přesné měření viskozity v průmyslovém procesu může být technicky náročné a nespolehlivé, řízení teploty poskytuje praktický způsob, jak nepřímo regulovat viskozitu skla. Detailnější popis závislosti viskozity skla na jeho teplotě můžete nalézt v kapitole 2 *GLASS FIBER PRODUCTION* [1].

V rámci mé bakalářské práce jsem přizpůsobil model *feederu* tak, aby odpovídal reálné konstrukci používané v konkrétní továrně. V ní jsou namísto plynových hořáků pro ohřev skla použity topné elektrody umístěné nad *bushingy* ve spodní zóně *feederu*. Zároveň jsem model zjednodušil tak, že jsem zanedbal tepelnou výměnu mezi atmosférou a sklem. Atmosféra se tedy chová jako dokonalý izolant. Toto zjednodušení bylo možné provést, protože tepelné ztráty do okolí skrze stěny *feederu* jsou řádově větší než tepelné ztráty do atmosféry. Model rozděluje sklo do horní a spodní vrstvy, přičemž teplotu skla lze měřit pouze ve spodní vrstvě. Vstupy a výstupy systému jsou uvedeny v tabulkách 2.1 a 2.2.

Symbol	**Význam**	**Jednotka**
Tamb	Teplota okolí	°C
TGFU	Teplota přitékajícího skla v horní vrstvě	°C
TGFL	Teplota přitékajícího skla ve spodní vrstvě	°C
Tb1	Teplota bushingu 1	°C
Tb2	Teplota bushingu 2	°C
Tb3	Teplota bushingu 3	°C
Tb4	Teplota bushingu 4	°C
Tb5	Teplota bushingu 5	°C
P1	Výkon dodávaný do zóny 1	kW
P2	Výkon dodávaný do zóny 2	kW
P3	Výkon dodávaný do zóny 3	kW
P4	Výkon dodávaný do zóny 4	kW
P5	Výkon dodávaný do zóny 5	kW

Tabulka 2.1: Vstupy systému

Přičemž výkony *P5-P1* jsou říditelné vstupy systému. Teploty *bushingů Tb5-Tb1*, teplota okolí *Tamb* a teplota horní a spodní vrstvy přitékajícího skla z pece *TGFU* a *TGFL* jsou měřitelné poruchy.

Symbol	Význam	Jednotka
TL1	Teplota ve spodní části zóny 1	°C
TL2	Teplota ve spodní části zóny 2	°C
TL3	Teplota ve spodní části zóny 3	°C
TL4	Teplota ve spodní části zóny 4	°C
TL5	Teplota ve spodní části zóny 5	°C

Tabulka 2.2: Výstupy systému

V rámci mé práce jsem si stanovil, že optimální teplota pro výrobu je 1210 °C a výroba probíhá v pořádku, pokud se teplota neodchýlí o víc než ± 10 °C. Tento odhad jsem provedl na základě teplot z historických dat [1]. Výkony topných elektrod jsem omezil na interval 0 až 1 kW. Omezení na výkony jsem odhadl tak, aby do systému mohlo být dodáno podobné množství energie, jako bylo v původním modelu dodáváno plynovým hořákem. V realitě by tyto hodnoty byly sděleny technologem výroby.

Kapitola 3

Software *REXYGEN*

Software *REXYGEN* od společnosti *REX Controls s.r.o.* je real-time řídicí software. K návrhu a implementaci řídicích systémů se používá aplikace *REXYGEN Studio*, což je vývojové prostředí poskytující uživatelům knihovnu funkčních bloků.

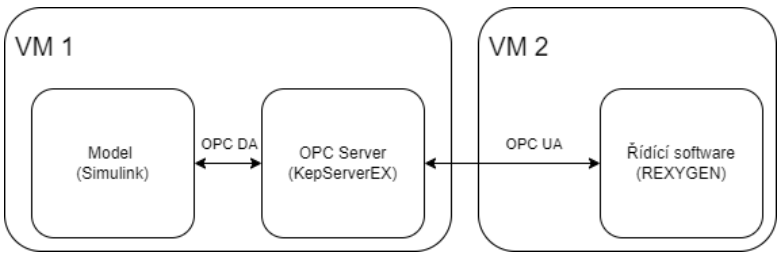
Projekt v *REXYGEN Studio* se dělí na dvě hlavní části: konfigurační soubor `*.mdl`, který obsahuje konfiguraci systémových parametrů, nastavení driverů a definice pořadí úloh, a samotné úlohy (*task*), kde je definováno řízení procesu. *REXYGEN Studio* umožňuje kompilaci projektu a následné nahrání (*download*) přímo na cílové zařízení (*target*), kde pak řídicí algoritmy běží. *REXYGEN* je kompatibilní jak se standardním osobním počítačem, tak například i s *Raspberry Pi*. Řízení je prováděno aplikací *REXYGEN runtime* na úrovni jádra operačního systému cílového zařízení. *REXYGEN* je dostupný v *DEMO* verzi zdarma. Chod demoverze je omezen na dvě hodiny. *REXYGEN* umožňuje také archivaci a zobrazení dat. Podrobnější informace jsou k přečtení na oficiálních stránkách *REXYGENu* [2].

Kapitola 4

Komunikační rozhraní

V reálném průmyslovém prostředí se můžeme setkat s konfigurací, kde je nadřazený řídicí software provozován na *Windows* počítači. V takovém uspořádání bývá běžné, že *OPC server* a další řídicí aplikace běží na různých virtuálních strojích (*VM*). Toto uspořádání umožňuje efektivní oddělení jednotlivých aplikací a zvyšuje bezpečnost systému tím, že izoluje jednotlivé komponenty. Komunikace probíhá tak, že řídicí software si z *OPC serveru* přečte hodnoty měřených veličin, spočítá akční zásahy, které následně zapíše na *OPC server*. Programovatelné logické automaty (*PLC*) ve fabrice použijí spočítané akční zásahy, aktualizují hodnoty měřených veličin, a cyklus se opakuje.

V mém případě jsem se snažil co nejvíce přiblížit této reálné konfiguraci. Můj simulační systém zahrnuje dva virtuální stroje. Na prvním virtuálním stroji běží *OPC server*, na který se připojují klienti běžící v *REXYGENu* a *Simulinku*. Na prvním virtuálním stroji zároveň běží model systému. Druhý virtuální stroj hostí softwarovou platformu *REXYGEN*. Diagram komunikace lze vidět na obrázku 4.1.



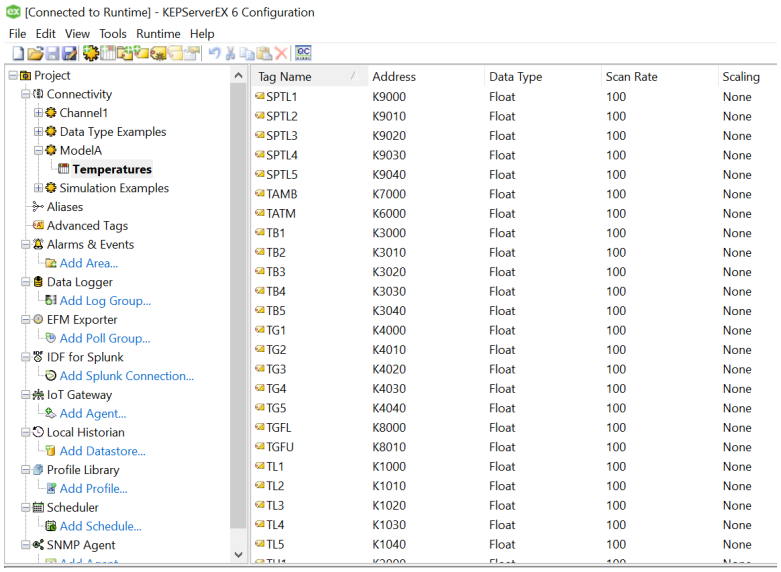
Obrázek 4.1: Diagram komunikace

Komunikace mezi *REXYGENem* a modelem v *Simulinku* je zajištěna

pomocí *OPC serveru KepServerEX*, který je jedním z používaných pro průmyslovou automatizaci. Dvouhodinová *DEMO* verze *KepServerEX* je rovněž zdarma. Tato konfigurace nejen napodobuje reálné průmyslové prostředí, ale také umožňuje snadný přechod k řízení skutečného procesu. Pro realizaci tohoto přechodu je třeba zajistit, aby *PLC* aplikovalo akční zásahy zapsané na *OPC serveru* a aby na *OPC server* byly zapisovány skutečné měřené hodnoty z *PLC* místo simulovaných dat z modelu.

4.1 *OPC* komunikace

Na *OPC serveru* jsem připravil sadu tagů, které reprezentují klíčové proměnné, jako jsou teploty a výkony. Tyto *tagy* jsou využívány pro obousměrnou komunikaci. Připravené tagy jsou zobrazeny na obrázku 4.2.

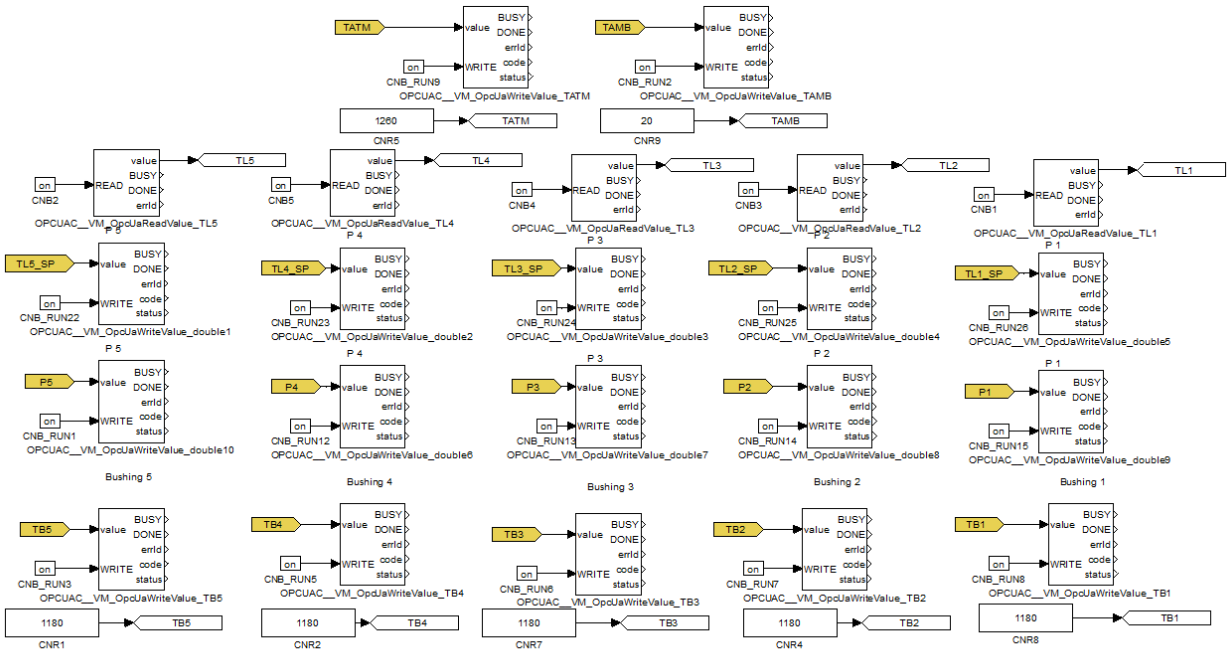


Obrázek 4.2: Připravené *tagy*

V průmyslové automatizaci existují dva hlavní standardy *OPC* komunikace: *OPC DA* (Data Access) a *OPC UA* (Unified Architecture). *OPC DA* umožňuje přístup k datům v reálném čase, ale nezahrnuje komunikaci přes internet a je založen na starších technologiích. Na druhou stranu, *OPC UA* poskytuje rozsáhlejší funkcionalitu, včetně bezpečného přenosu dat přes různé platformy a sítě.

4.1.1 *REXYGEN*

Integrace *OPC* komunikace v *REXYGENu* je realizována prostřednictvím dedikovaného *OPC UA* driveru. Nastavení parametrů pro *OPC UA*, jako jsou *URL serveru*, *timeout*, typ šifrování a další parametry, lze nastavit přímo v grafickém uživatelském rozhraní konfiguračního prostředí (*GUI*) driveru. Komunikace mezi *REXYGENem* a *OPC serverem* je pak realizována pomocí bloků *OPCRead* a *OPCWrite*, do kterých stačí zadat cestu k *tagu*. Podrobnější informace jsou k dispozici v manuálu k *OPC UA* driveru v [3]. Vzorkování čtení a zápisu jsem nastavil na 0.5 sekundy. Realizace *OPC* komunikace na straně *REXYGENu* může být vidět na obrázku 4.3.



Obrázek 4.3: Ukázka bloků *OPCRead* a *OPCWrite* v *REXYGEN Studiu*

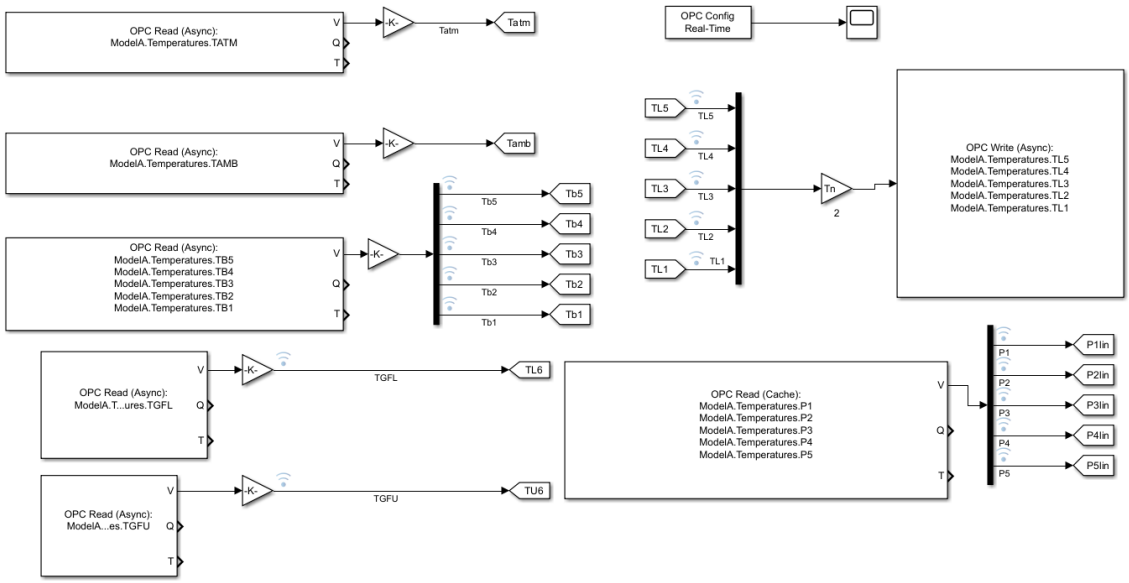
4.1.2 *Simulink*

Model v *Simulinku* jsem upravil tak, aby mohl číst a zapisovat data na *OPC server*. K realizaci této funkcionality jsem využil *Industrial Communication Toolbox*, který poskytuje nástroje pro integraci s různými průmyslovými komunikačními standardy včetně *OPC*. Přestože jsem původně plánoval využít modernější standard *OPC UA*, nakonec jsem se rozhodl pro *OPC DA*. Narazil jsem totiž na technické potíže při konfiguraci *OPC UA*. Vzorkování čtení a zápisu jsem také nastavil na 0.5 sekundy. Zapojení *OPC* komunikace

v *Simulinkuu* může být vidět na obrázku 4.4.

Abych mohl simulovat chování reálného systému, nastavil jsem simulaci do pseudo real-time režimu. Toho jsem dosáhl pomocí bloku *OPC Configuration*, který zajišťuje, že časování simulace odpovídá skutečnému času.

Vzhledem k tomu, že systém obsahuje relativně dlouhé časové konstanty a dopravní zpoždění, model je nastaven tak, že jedna sekunda simulace odpovídá jedné minutě reálného procesu.



Obrázek 4.4: Ukázka bloků *OPC Read* a *OPC Write* v *Simulinku*

Kapitola 5

Identifikace systému

Před návrhem řízení je zapotřebí systém identifikovat. Jednou z možností je vytvořit model systému založený na fyzikálních základech, což obnáší podrobnou analýzu fyzikálních procesů probíhajících v systému. Tento přístup může být však velmi náročný a složitý, jak dokládá model použitý v této práci, který byl vytvořen v rámci diplomové práce.

Rozhodl jsem se použít metodu, která spočívá v aproximaci složitého nelineárního modelu pomocí přenosových funkcí v okolí pracovního bodu. Výhodou této metody je, že by bez velkých změn šla provést i pro identifikaci reálného systému v průmyslové praxi na běžící technologii (výrobní lince).

5.1 Pracovní bod

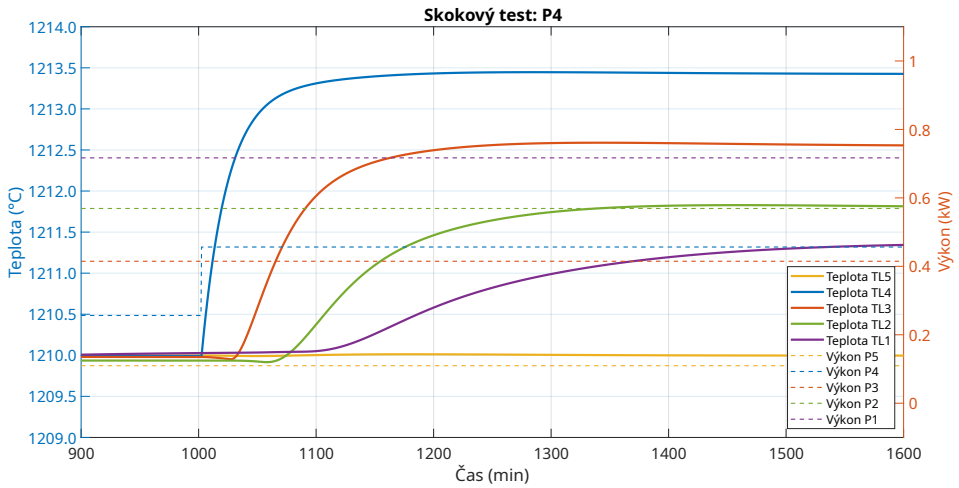
Jak jsem zmínil v 2.1, jako provozní teplotu skla jsem zvolil 1210 °C, a proto budu systém identifikovat kolem této teploty. Bylo tedy zapotřebí najít hodnoty výkonů *P5-P1*, které zajistí žádanou teplotu 1210 °C ve všech pěti zónách. Tento bod jsem našel pomocí pěti *PI* regulátorů. Při nastavování parametrů regulátorů mi nešlo o vlastnosti řízení, ale pouze o to, aby se po nějakém čase teploty dostaly na *setpointy*. Jakmile se systém dostal do ustáleného stavu s teplotami na *setpointech*, poznamenal jsem si hodnoty akčních zásahů.

Zóna	Výkon (kW)
P5	0.1096
P4	0.2564
P3	0.4148
P2	0.5690
P1	0.7169

Tabulka 5.1: Hodnoty výkonů *P5-P1* v pracovním bodě

5.2 Skokový test výkonů

Pro nalazení přenosových funkcí z výkonů na teploty jsem postupoval následovně. Nastavil jsem výkony na hodnoty z tabulky 5.1 a čekal, až se teploty ve všech zónách ustálí. Po dosažení ustáleného stavu jsem skokově zvýšil výkon *P5* o 0.2 kW a opět jsem počkal na ustálený stav. Tento postup jsem opakoval i pro ostatní výkony. Z průběhu teplot jsem následně odhadl jednotlivé přenosové funkce. Ukázka jednoho z testů je zobrazena na obrázku 5.1.



Obrázek 5.1: Skokový test výkonu *P4*

Průběhy se podobaly systému prvního řádu s dopravním zpožděním (*FOPDT*). Podle návodu [4, str. 5] jsem tedy určil parametry systému.

Konkrétně jsem po skokovém zvýšení výkonu nejprve zaznamenal čas tohoto skoku a označil jej jako t_{step}. Poté jsem sledoval, kdy teplota poprvé překročí 2% změnu od ustáleného stavu. Tento čas jsem zaznamenal jako $t_{2\%}$. Následně jsem spočítal dopravní zpoždění jako rozdíl těchto časů:

$$D = t_{2\%} - t_{\text{step}}$$

Rozdíl teplot před a po zvýšení výkonu v ustálených stavech jsem použil k výpočtu zesílení K podle vzorce:

$$K = \frac{\Delta T}{\Delta P}$$

kde ΔT je rozdíl teplot a ΔP je rozdíl výkonů. Dále jsem vypočítal čas, po kterém teplota dosáhla 63% z celkové změny, a označil jej jako $t_{63\%}$. Časovou konstantu T systému jsem následně vypočítal jako:

$$T = t_{63\%} - t_{2\%}$$

Tuto identifikaci jsem provedl v režimu otevřené smyčky (*open loop*) pro všechny zóny a výkony. Výsledkem tohoto experimentu byla matice 5x5 přenosových funkcí z výkonů *P5-P1* na teploty *TL5-TL1*.

	TL5	**TL4**	**TL3**	**TL2**	**TL1**
P5	$\frac{14.85}{26s+1}$	$\frac{11.3}{44s+1}e^{-34s}$	$\frac{8.25}{66s+1}e^{-70s}$	$\frac{5.45}{83s+1}e^{-123s}$	$\frac{3.45}{124s+1}e^{-145s}$
P4	-	$\frac{17.25}{26s+1}$	$\frac{12.95}{49s+1}e^{-34s}$	$\frac{9.15}{79s+1}e^{-77s}$	$\frac{6.7}{150s+1}e^{-106s}$
P3	-	-	$\frac{20.85}{27s+1}$	$\frac{15}{59s+1}e^{-38s}$	$\frac{11}{135s+1}e^{-78s}$
P2	-	-	-	$\frac{27.95}{34s+1}$	$\frac{19.4}{112s+1}e^{-36s}$
P1	-	-	-	-	$\frac{48.4}{60s+1}$

Tabulka 5.2: Přenosové funkce výkonů *P5-P1* na teploty *TL5-TL1*

Vliv výkonů na teploty v předchozích zónách byl minimální, jak je vidět na skokovém testu výkonu *P4* na grafu 5.1, kde teplota *TL5* nebyla změnou ovlivněna. Z tabulky 5.2 je také vidět, že zesílení přenosových funkcí výkonů na další zóny postupně klesá a zároveň roste časová konstanta a dopravní zpoždění. To odpovídá očekávání.

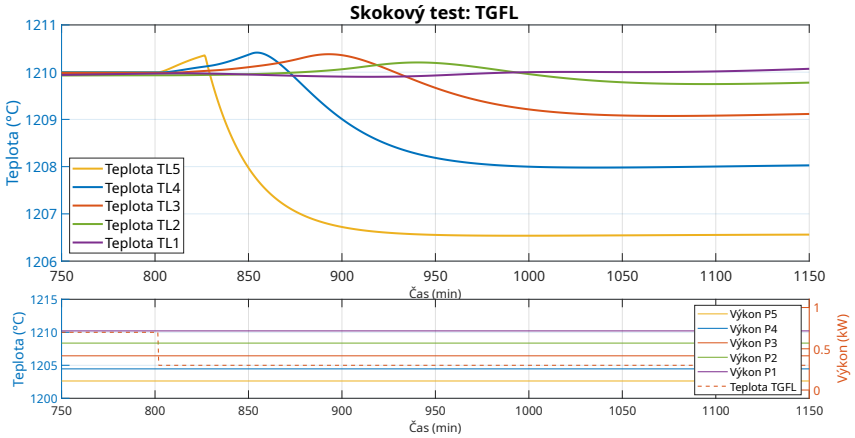
Pro ověření odhadů přenosových funkcí jsem provedl linearizaci modelu v *Simulinku* pomocí nástroje pro lineární analýzu. Tento proces mi také vrátil matici přenosových funkcí rozměru 5x5, přičemž některé funkce dosahovaly až devatenáctého řádu. Výsledky linearizace většinou korespondovaly s mými odhady. Pro dvojice *P5* a *TL2*, *P5* a *TL1*, *P4* a *TL1* však byly výsledky značně odlišné. I přesto jsem je v mém modelu použil. Toto porovnání jsem totiž nebral za směrodatné, jelikož v reálném případě bych tento nelineární model pravděpodobně k dispozici neměl.

5.3 Poruchy

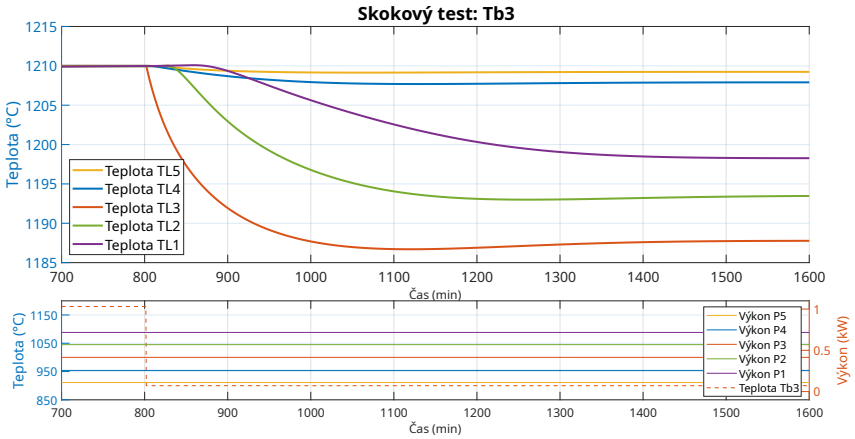
Abych zvýšil přesnost a relevanci linearizovaného modelu vzhledem k nelineárnímu modelu systému, přidal jsem přenosové funkce z poruch na teploty v jednotli-

vých zónách. Konkrétně tedy vliv teploty *bushingu* 3 *Tb3* a teplot přitékajícího skla *TGFU*, *TGFL* na teploty *TL5-TL1*.

- **Skokový test teplot přitékajícího skla *TGFU* a *TGFL***: Provedl jsem identifikaci stejným principem, jaký byl popsán v 5.2. Nejprve jsem provedl skok na teplotě *TGFU* a v následujícím experimentu i na teplotě *TGFL*. Vliv na teploty *TL3*, *TL2* a *TL1* byl zanedbatelný. V mém lineárním modelu jsem tedy uvažoval pouze zbylé přenosové funkce.
- **Skokový test teploty *bushingu* 3 *Tb3***: Jednou z největších poruch, která se může v systému objevit, je výměna *bushingu*. Na konci jeho životnosti je potřeba jej vyměnit za nový. Během této operace je zchlazen z provozních 1180 °C na 900 °C, což má velký vliv na teploty v jednotlivých zónách. Při identifikaci jsem postupoval stejně jako v 5.2, pouze místo výkonu jsem provedl skok na teplotě *bushingu*.



Obrázek 5.2: Skokový test *TGFL*



Obrázek 5.3: Skokový test *Tb3*

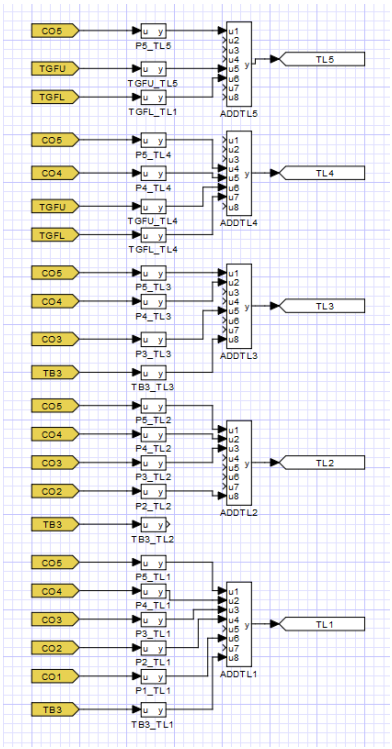
	TL5	TL4	TL3	TL2	TL1
Tb3	-	-	$\frac{0.081}{63s+1}$	$\frac{0.062}{103s+1}e^{-41s}$	$\frac{0.044}{187s+1}e^{-98s}$
TGFU	$\frac{0.228}{50s+1}e^{-40s}$	$\frac{0.406}{79s+1}e^{-77s}$	-	-	-
TGFL	$\frac{0.692}{23s+1}e^{-30s}$	$\frac{0.404}{35s+1}e^{-76s}$	-	-	-

Tabulka 5.3: Přenosové funkce poruch na teploty *TL5-TL1*

Na obrázku 5.2 může být vidět zajímavý jev, kdy při ochlazení přitékajícího skla se teploty v zónách nejprve lehce zvýší a až poté klesnou. Tento jev jsem v mém linearizovaném modelu zanedbal.

Z obrázku 5.3 je vidět, že na teploty zón před *bushingem* 3 (*TL5*, *TL4*) nemá změna v teplotě *bushing* 3 *Tb3* příliš velký vliv. Stejný postup identifikace by šel aplikovat i pro zbylé *bushingy*. Pro moje experimenty jsem se však omezil na *bushing* 3. Vzhledem k tomu, že zóna 3 je uprostřed kanálu, experimenty v této zóně ukáží chování teplot jak v zónách před ní, tak i v zónách za ní.

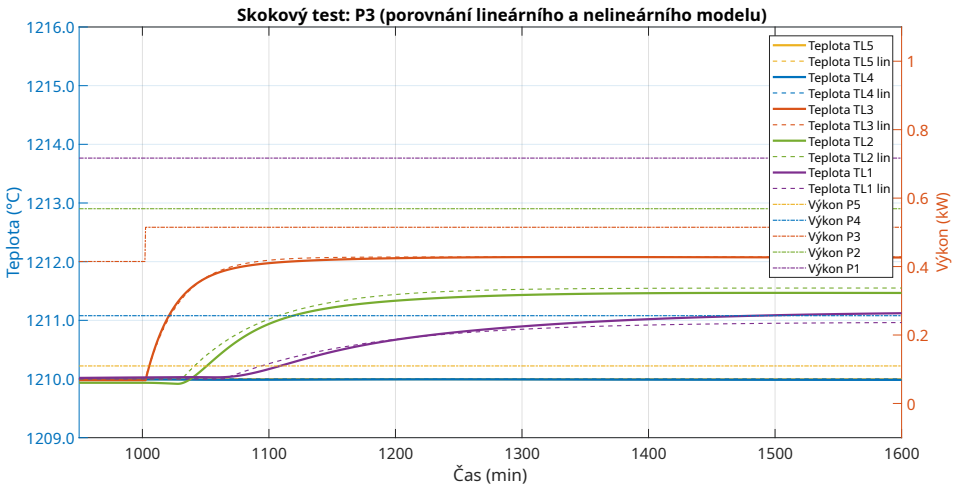
Identifikovaný model jsem zapojil v *REXYGENu* pro následný návrh a testování řídicích algoritmů. Zapojil jsem je pomocí *FOPTD* bloků. Výstupy přenosových funkcí odpovídající stejným teplotám jsem sečetl pomocí sčítačky signálů, jak může být vidět na obrázku 5.4.



Obrázek 5.4: Linearizovaný model zapojený v *REXYGENu*

5.4 Porovnání modelů

Věrohodnost modelu jsem následně otestoval. V okolí pracovního bodu linearizovaný model věrně simuloval chování nelineárního modelu při poruše i při změně výkonu, což může být vidět na obrázku 5.5, kde jsem skokově změnil výkon *P3*.



Obrázek 5.5: Nelineární model v porovnání s linearizovaným modelem

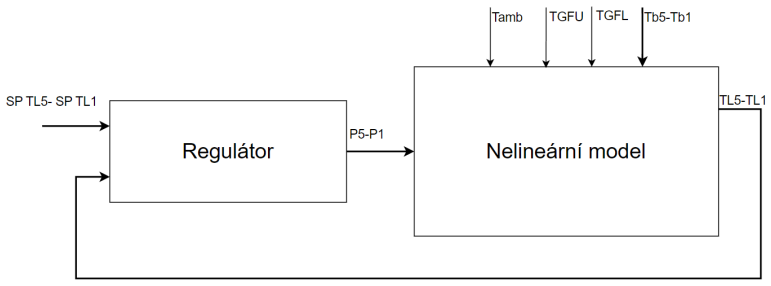
Kapitola 6

Návrh řízení

Zóny se výrazně ovlivňují podél toku skla, jak je vidět v matici přenosových funkcí v kapitole 5. Například při zvýšení *setpointu* ve třetí zóně začne regulátor více topit, což způsobí, že teplejší sklo poteče do zón dvě a jedna a způsobí tam poruchy.

Požadavky na řízení, které jsem si stanovil, jsou: maximální překmit 10 %, nulová ustálená odchylka a co nejkratší doba náběhu a ustálení.

V rámci mé práce jsem navrhl tři různé typy řízení. Prvním typem je řízení pomocí *PI* regulátoru, které jsem naladil pomocí *Ziegler-Nicholsovy* metody. Druhým typem je řízení pomocí *PID* regulátoru, jehož parametry naladil automatický ladicí nástroj (*autotuner*), který *REXYGEN* nabízí. Třetím řízením, které jsem použil, je prediktivní řízení založené na modelu (*MPC*). Jejich porovnání se věnuji v kapitole 7.



Obrázek 6.1: Diagram regulace

6.1 Regulace *PI*

V rámci tohoto přístupu jsem se pokusil naladit pět samostatných *SISO* (*Single Input Single Output*) *PI* regulátorů, každý pro jednu z pěti zón systému. Tato strategie vycházela z předpokladu, že pokud bude řízení dostatečně robustní, jednotlivé regulátory nebudou vzájemně negativně interagovat a nebudou se tak vzájemně ovlivňovat. Každý regulátor byl naladěn tak, aby efektivně zvládal regulaci teploty ve své zóně, ignorujíc interakce mezi jednotlivými zónami.

Pro návrh parametrů regulátoru každé zóny jsem použil pouze diagonální přenosovou funkci. Například pro zónu 5 jsem použil pouze přenosovou funkci z *P5* na *TL5*. Pomocí metody *Ziegler-Nichols* jsem získal hodnoty pro zesílení K a časovou konstantu T_i. Postupoval jsem podle návodu na webových stránkách [5] a konkrétní postup byl následující:

- **Zapojení *P* regulátoru do uzavřené smyčky (*closed-loop*)**: Pro přenosovou funkci k příslušné zóně jsem zapojil P regulátor.
- **Zvýšení zesílení** K: Postupně jsem zvyšoval hodnotu zesílení K, dokud systém nedosáhl hranice stability a nezačal trvale oscilovat.
- **Určení kritických parametrů** K_u **a** T_u: Zaznamenal jsem kritické zesílení K_u, při kterém systém osciloval, a periodu oscilací T_u.
- **Výpočet parametrů *PI* regulátoru**: Pomocí vztahu $K = 0.45 \times K_u$ a $T_i = 0.85 \times T_u$ jsem vypočítal hodnoty parametrů.

Aplikoval jsem tyto kroky na všechny zóny a získal následující parametry.

Zóna	K	T_i
5	7.875	0.510
4	6.796	0.510
3	5.850	0.595
2	5.490	0.510
1	5.589	0.510

Tabulka 6.1: Návrh parametrů *PI* regulátorů pro jednotlivé zóny

Otestoval jsem řízení na lineárním modelu. Regulace fungovala bez problémů. Při přechodu na testování na nelineárním modelu však bylo nutné konstanty upravit, abych odstranil oscilace v systému. Po doladění jsem dospěl k následujícím hodnotám.

Zóna	K	T_i
5	0.388	5.10
4	0.380	7.30
3	0.385	9.95
2	0.249	10.14
1	0.258	10.22

Tabulka 6.2: Hodnoty parametrů *PI* regulátorů pro jednotlivé zóny

6.2 Regulace *PID*

Se stejnými předpoklady jako pro *PI* regulaci jsem chtěl získat pět *SISO PID* regulátorů.

V rámci testování mi tým *REX Controls s.r.o.* poskytl jinak placenou *Autotuning* licenci, která je potřeba při používání bloku *PIDMA*. Použití bylo velmi přímočaré. Bloky jsem zapojil do zpětné vazby s nelineárním modelem. Nastavil jsem je na manuální mód, tedy výstupy regulátorů byly rovny výkonům z tabulky 5.1, a počkal jsem na ustálený stav. Po jednom jsem pak spustil ladění. Nalezené hodnoty jsem pak překopíroval do standardního bloku *PIDU*.

Zóna	K	T_i	T_d
5	0.238	10.83	2.189
4	0.210	12.11	2.445
3	0.165	11.40	2.303
2	0.146	12.73	2.570
1	0.112	14.66	2.962

Tabulka 6.3: Hodnoty parametrů *PID* regulátorů pro jednotlivé zóny

6.3 Prediktivní řízení založené na modelu (*MPC*)

Ačkoliv *REXYGEN* standardně nenabízí přímo blok *MPC*, umožňuje implementaci vlastních algoritmů řízení pomocí bloků *REXLANG* a *PYTHON*. V *REXLANG* bloku může uživatel využít programovací jazyk podobný C k naprogramování algoritmů, které jsou poté integrovány přes vstupy a výstupy bloku do celkového projektu. *PYTHON* blok umožňuje programování v jazyce *Python* s možností importovat standardní i vlastní knihovny. V rámci mého projektu byla využita knihovna *MPC*, kterou implementoval můj kolega Ing. Filip Vodňanský v rámci jeho diplomové práce [6].

MPC je *MIMO* (*Multiple Input Multiple Output*) regulátor, který předpovídá budoucí chování systému na základě modelu. V každém kroku regulace *MPC* řeší optimalizační úlohu, kde hledá optimální řadu akčních zásahů pro následujících N kroků, aby dosáhl co nejmenší hodnoty kriteriální funkce (viz rovnice 3.2 v [6]). Z vypočítané série akčních zásahů je následně použita pouze první hodnota a proces se opakuje, což zajišťuje zpětnou vazbu regulátoru. Váhy přiřazené jednotlivým parametrům a omezení akčních zásahů určují, jaký kompromis regulátor učiní mezi různými zónami a jejich požadavky. Součástí stavů je i integrační složka, aby bylo docíleno nulové ustálené odchylky.

Součástí *MPC* je i *Kalmanův filtr*, který se používá k aktualizaci odhadů stavů systému na základě nových měření a předchozích predikcí. Filtr lze konfigurovat, aby upřednostňoval buď aktuální měření, nebo modelové predikce. To umožňuje regulátoru adaptovat se na měnící se podmínky v reálném čase.

Díky těmto schopnostem může *MPC* teoreticky zlepšit výkon systému tím, že umožňuje lokální adaptace, jako je například zvýšení ohřevu v jedné zóně, které pomáhá stabilizovat podmínky v další zóně.

Jelikož se jedná o poměrně složitý regulátor, seznámil jsem se s jeho fungováním na jednoduchých systémech. Následně jsem regulátoru dal informaci o identifikovaných přenosových funkcích. Pak jsem jej naladil na lineárním modelu, abych získal základní odhad jednotlivých vah. Při testování na nelineárním modelu jsem parametry měnil a došel k následujícím hodnotám (v nastavení *MPC* se objevuje více parametrů, které zde nezmiňuji).

$Q_v = [3, 3, 3, 3, 3]$ (váhy na odchylky od *setpointů*)
$R_v = [150, 150, 150, 150, 150]$ (váhy na změnu akčních zásahu mezi kroky)

Abych si v některých experimentech v jednom výpočetním cyklu mohl dovolit větší horizont (na kolik kroků dopředu *MPC* řeší optimalizační úlohu), přešel jsem na vzorkování po pěti sekundách. To mi umožnilo zajistit, aby *MPC* počítalo 375 sekund dopředu.

Kapitola 7

Experimenty

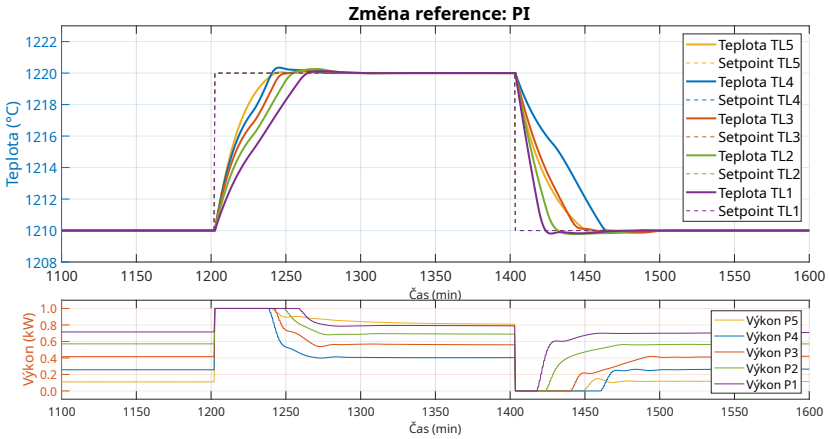
Funkčnost a různé vlastnosti regulátorů ukáži na třech experimentech: změna *setpointů* a působení poruchy v podobě přítoku chladnějšího skla z pece a výměny *bushingu* 3.

7.1 Změna *setpointů*

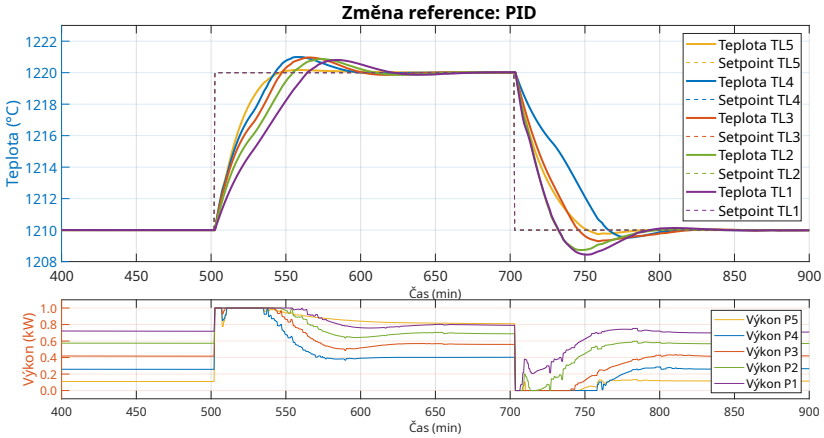
V průmyslové výrobě skelných vláken je často potřeba reagovat na změny kvality vstupních surovin a chemického složení skla. Běžným požadavkem je změna teploty skla vtékajícího do *bushingů*. Tento experiment simuluje požadavek na teplejší sklo zvýšením *setpointů*.

Cílem experimentu je zjistit, jak rychle regulátory reagují na změnu *setpointů* a zda splňují požadavky na maximální překmit 10 %, nulovou ustálenou odchylku a co nejkratší dobu náběhu a ustálení.

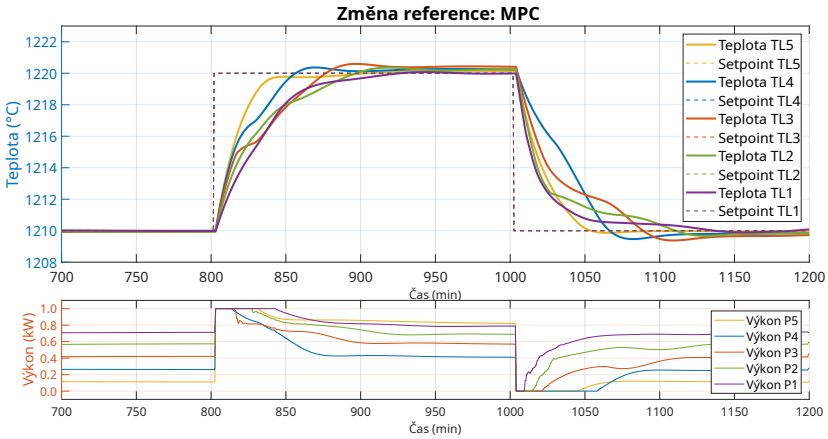
Do tabulky jsem vložil hodnotu největšího překmitu ze všech zón, nejpomalejší dobu náběhu ze všech zón a nejpomalejší dobu ustálení ze všech zón. Pro určení překmitu jsem změřil, o kolik maximálně hodnota překročila *setpoint*. Pak jsem tuto hodnotu vyjádřil jako procentuální podíl ze změny hodnoty *setpointu*. Za dobu náběhu jsem považoval čas, po kterém se hodnota teploty dostala na 90 % ze změny *setpointu* od času jeho změny. Čas ustálení jsem bral jako čas od změny *setpointu* po čas, po kterém teplota neopustila interval



Obrázek 7.1: Změna *setpointů*: *PI*



Obrázek 7.2: Změna *setpointů*: *PID*



Obrázek 7.3: Změna *setpointů*: *MPC*

± 2 % od *setpointu*.

	PI	PID	MPC
Překmit (%)	3.5	10.0	5.9
Doba náběhu (min)	55	56	75
Doba ustálení (min)	76	112	x

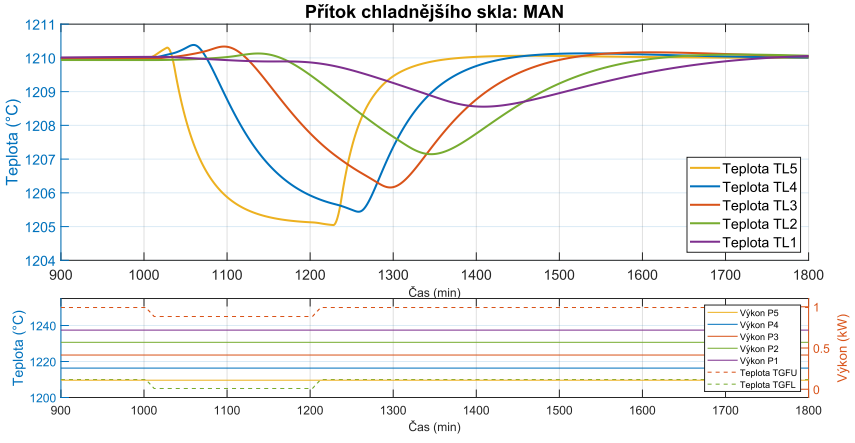
Tabulka 7.1: Výsledky experimentu

Z průběhů 7.1, 7.2 a 7.3 a tabulky 7.1 je vidět, že *PI* i *PID* regulace požadavky splnily. Bohužel *MPC* s tímto nastavením nedosáhlo nulové ustálené odchylky. Při zvýšení váhy na integrační složku bohužel docházelo k příliš velkému překmitu u experimentu, který simuloval výměnu *bushingu* (viz 7.3).

7.2 Porucha: Přítok chladnějšího skla z pece

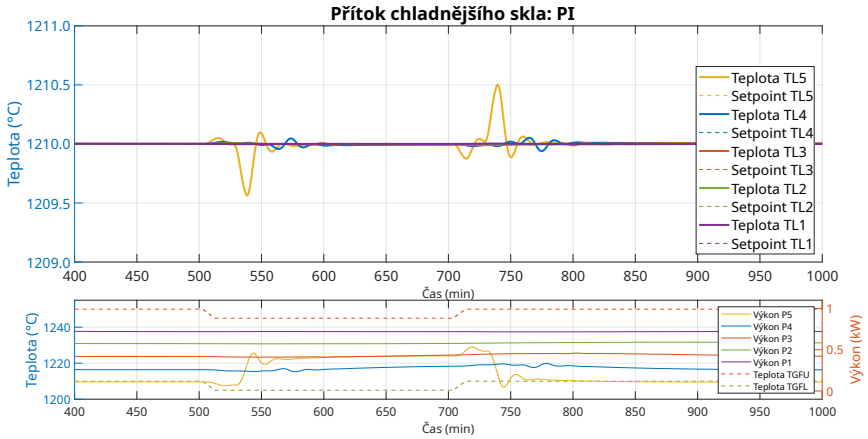
V průmyslové výrobě skelných vláken je zásadní udržet konstantní teplotu skla při jeho výtoku z *bushingu*, aby byla zajištěna kvalita a konzistence výsledného produktu. Tento experiment simuluje situaci, kdy do procesu náhle přitéká chladnější sklo z pece. Tento typ poruchy může nastat z mnoha důvodů, například kvůli změnám ve složení surovin nebo nedostatečnému řízení procesu tavby a ve výrobě je běžným jevem.

Cílem tohoto experimentu je ověřit schopnost regulátorů reagovat na změnu v teplotě přitékajícího skla z pece.

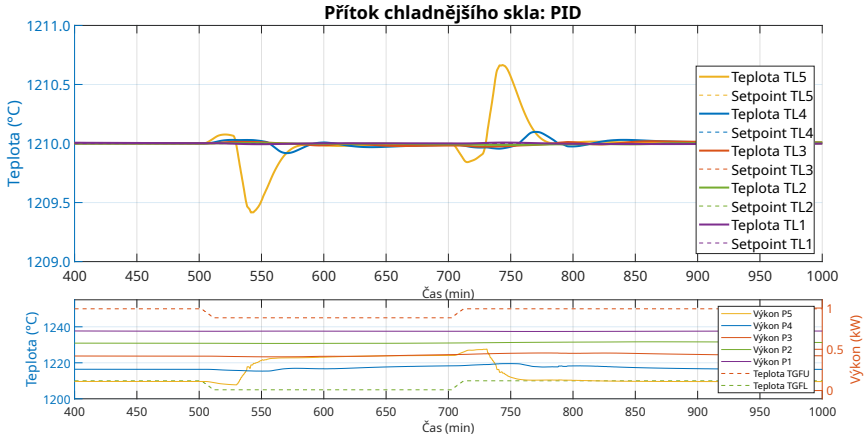


Obrázek 7.4: Porucha *TGFU* a *TGFL*: Manuální mód

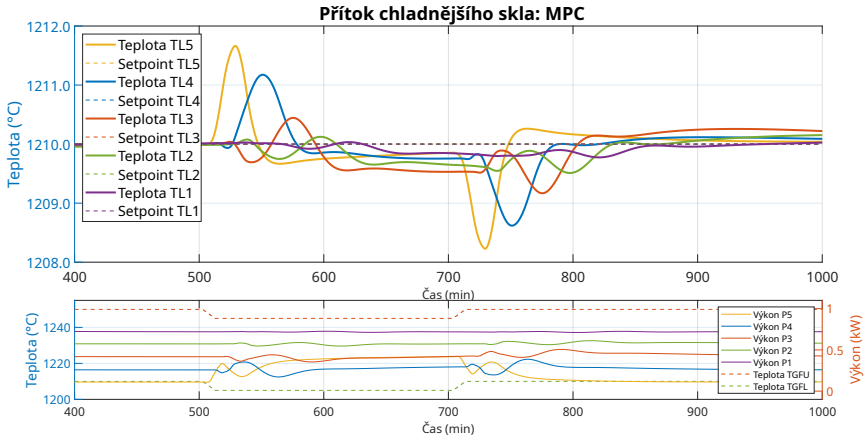
Z průběhů regulace zachycené na grafech 7.5,7.6 a 7.7 je vidět, že *PI* a *PID* regulátory si s poruchou poradily mnohem lépe než *MPC*. Z průběhu je



Obrázek 7.5: Regulace *PID* při poruše *TGFU* a *TGFL*



Obrázek 7.6: Regulace *PID* při poruše *TGFU* a *TGFL*



Obrázek 7.7: Regulace *MPC* při poruše *TGFU* a *TGFL*

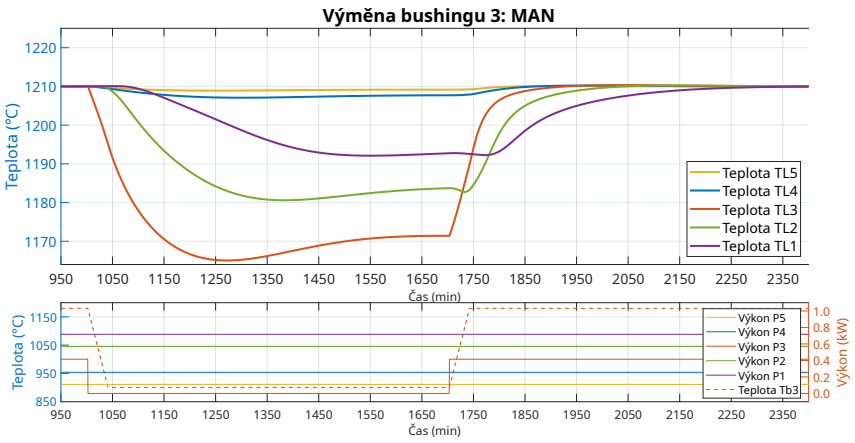
vidět, že *MPC* pravděpodobně počítalo s přítokem chladnějšího skla, ale jeho vliv přecenilo a zbytečně zóny přetopilo. Problém je nejspíše v tom, že můj lineární model se v této situaci příliš liší od nelineárního modelu. Stejně jako v přechozím experimentu je vidět, že *MPC* se nepodařilo zajistit nulovou ustálenou odchylku.

7.3 Porucha: Výměna *bushingu* 3

V předchozí kapitole o identifikaci systému jsem již zmínil situaci spojenou s výměnou *bushingu* 3. V této části otestujeme, jak si s ní poradí regulátory.

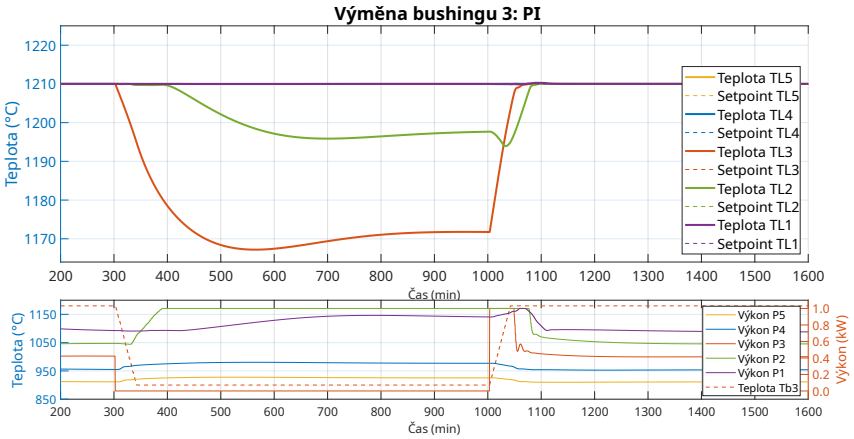
Bushing je klíčovou součástí výrobního procesu. Sklo, které do něj z *feederu* přitéká, z něj vytéká a je dále zpracováváno. Jeho výměna je nezbytná z důvodu opotřebení nebo plánované údržby. Při výměně *bushingu* je nutné *bushing* ochladit, což může mít za následek výrazné snížení teploty v dané zóně. To následně ovlivní teploty v sousedních zónách. Z technologických důvodů je zároveň nutné v této zóně po dobu výměny vypnout ohřev, což má také vliv na teplotu v zóně.

Cílem tohoto experimentu bylo ukázat, zda se podaří regulátorům udržet výrobu v co nejvíce zónách, tedy udržet teploty ve vzdálenosti ± 10 °C od 1210 °C.

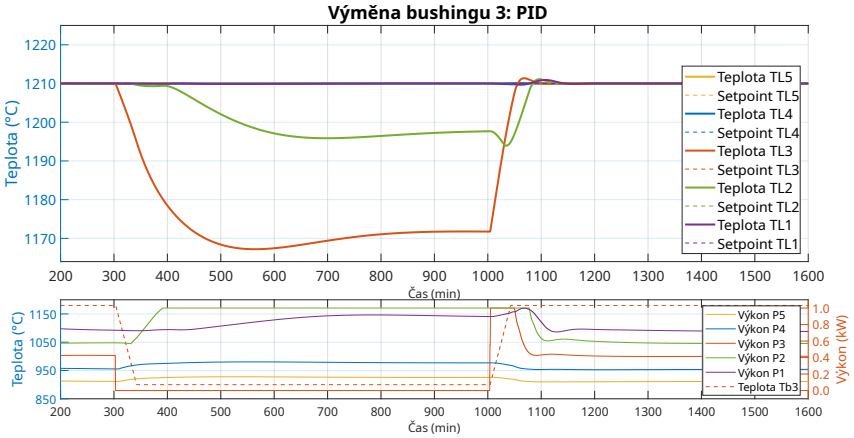


Obrázek 7.8: Výměna *bushingu* 3: Manuální mód

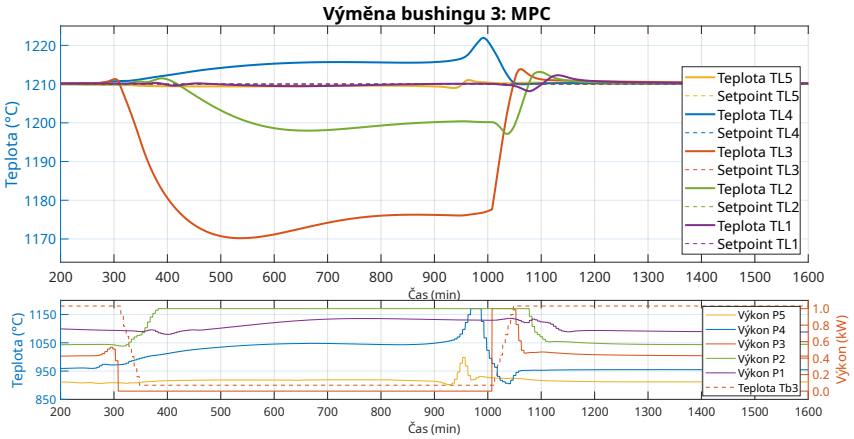
Z průběhů 7.9,7.10 a 7.11 je vidět, že *PI* a *PID* regulátory se chovaly velmi podobně. Ohřev v zóně 4 narazil na saturaci a regulátor tak nebyl schopný udržet teplotu *TL2* nad 1200 °C. U *MPC* byla po dobu výměny



Obrázek 7.9: Výměna *bushingu* 3: *PI*



Obrázek 7.10: Výměna *bushingu* 3: *PID*



Obrázek 7.11: Výměna *bushingu* 3: *MPC*

nastavena váha na teplotu *TL3* na 0, jelikož v této zóně na teplotě v průběhu výměny *bushingu* nezáleží (naopak by neměla být příliš velká, aby šel *bushing* vyměnit). *MPC* se snažilo ohřát zónu 4, aby pomohlo zóně 2, která byla v saturaci. Teplota *TL2* i tak na chvilku klesla pod 1200 °C. Následně při snaze dostat teploty co nejrychleji zpátky po dokončení výměny, *MPC* přetopilo zónu 4 nad 1220 °C. Tyto problémy by mohlo vyřešit nastavení omezení na teploty. *MPC* by v tom případě vědělo, že takto může pomáhat zóně ohřevem zóny předešlé, ale pouze do hranice nastaveného limitu. Tuto funkcionalitu jsem bohužel v čase provádění experimentů neměl k dispozici.

Kapitola 8

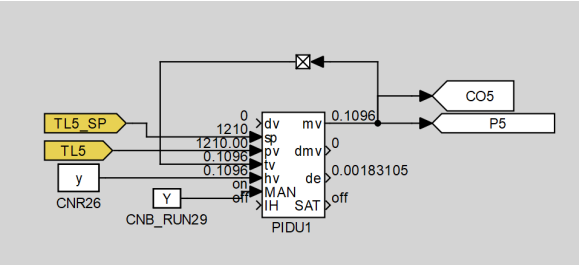
Zobrazení a archivace procesních dat

REXYGEN nabízí nástroje pro zobrazení a archivaci dat z průmyslového procesu, což umožňuje uživatelům efektivně monitorovat průběh regulace v reálném čase i retrospektivně. Tato funkcionalita může být užitečná v situacích, kdy je potřeba analyzovat historická data nebo například identifikovat příčiny vzniklých problémů a dolaďovat na základě toho parametry regulátorů.

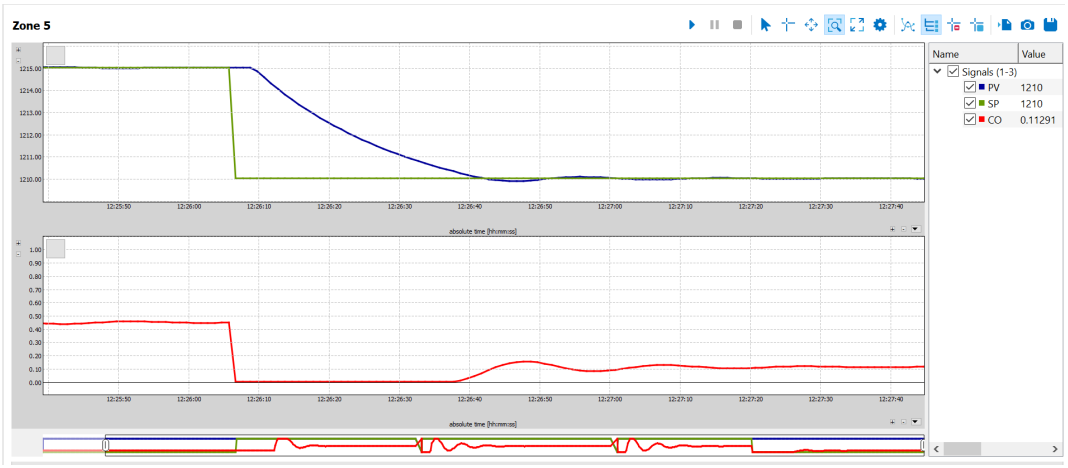
8.1 Zobrazení dat

Živé hodnoty signálů je možno sledovat přímo v *REXYGEN Studio* pomocí *Watch mode*. Pro lepší přehlednost lze využít bloku *Display* (Numeric display of input values), který rovněž ukazuje aktuální hodnotu signálu. Pro zobrazení průběhů regulace jsem využíval blok *TRND* (Real-time trend recording), který nabízí zobrazení až 4 signálů. Při potřebě zobrazit více signálů lze využít blok *TRNDV* (Real-time trend recording with vector input). Tento blok umí zobrazit vektor signálů. Vektor signálů lze vytvořit pomocí bloku *TROV* (Vector multiplexer) z klasických signálů. Data zobrazená bloky *TRND* a *TRNDV* lze exportovat jako *.csv* soubor.

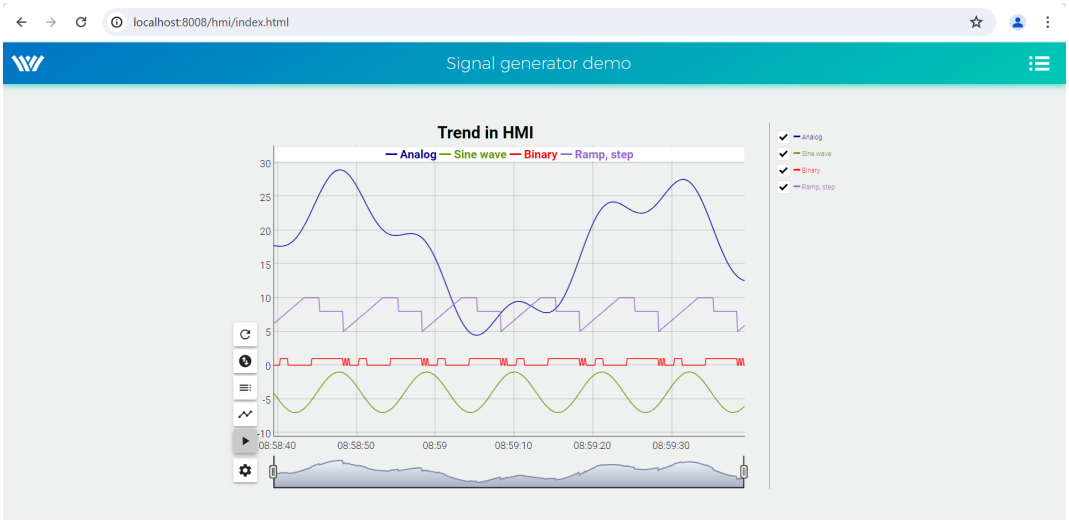
Další možností je využít *Web interface*, kde uživatel po nastavení může docílit zobrazení a interakce s běžícím projektem skrz webový prohlížeč. Funkcionalitu jsem otestoval pouze na ukázkovém *DEMO* projektu *Signal generators, trend in HMI*. To umožňuje, aby si například kolega mohl zobrazit pouze průběhy bez toho, aby měl přístup k celému projektu.



Obrázek 8.1: *Watch mode*



Obrázek 8.2: Blok *TRND*



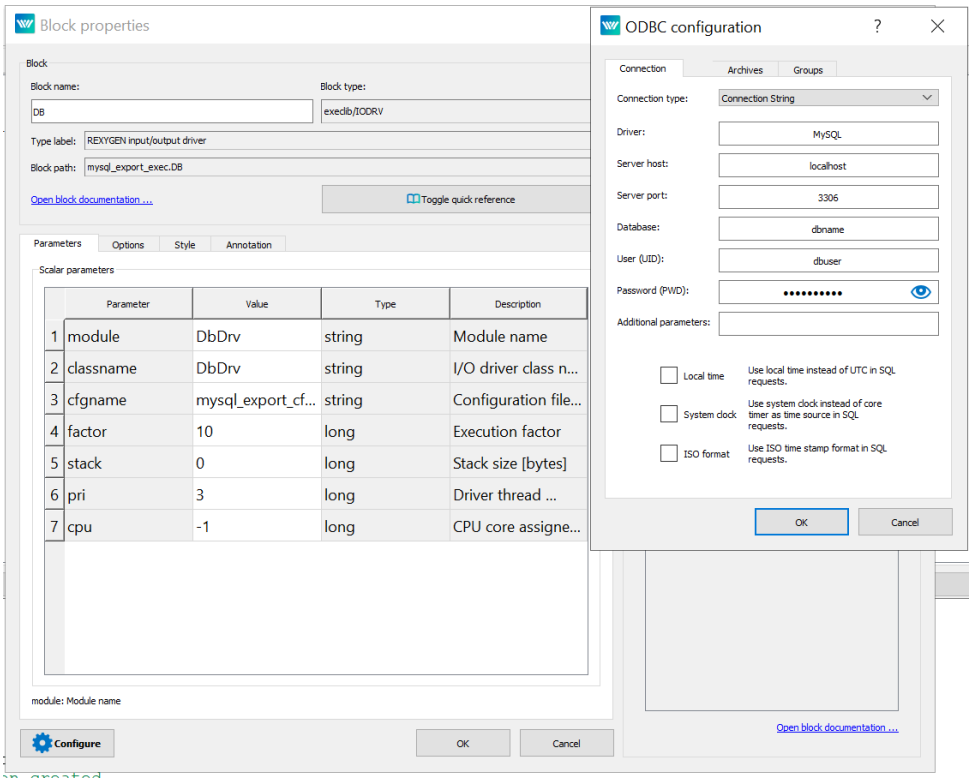
Obrázek 8.3: Ukázka *Web interface*

8.2 Archivace dat

K archivaci dat se používá driver *ARC* (The REXYGEN system archive), který umožňuje nastavit například typ ukládání (*DISK*, *RAM*), periodu ukládání nebo maximální velikost ukládaných dat za den.

V projektu lze následně ukládat signály pomocí napojení na blok *ARS* (Archive store value). V bloku pak stačí zvolit, do kterého archivu chci signál uložit a s jakým ID. Podobným způsobem lze využít bloku *TRND*, ve kterém lze zároveň nastavit zpracování dat. Je možné například ukládat maximální hodnotu, průměr a tak dále. Data v archivech lze zpětně zobrazit přímo v *REXYGEN Studio* v sekci *Diagnostics*.

Další z možností je ukládat data do databáze. Pro moje testování jsem využil databázi *MySQL*. Po zapojení *IODRV* (The REXYGEN system input/output driver) stačí zvolit parametr *module* na *DbDrv*. Následně uživatel spojení nastaví pomocí konfiguračního *GUI*. Signály, které uživatel chce ukládat, lze připojit k bloku Goto nebo OUTSTD (Signal source or output).



Obrázek 8.4: Ukázka *GUI* pro nastavení komunikace s databází

Kapitola 9

Závěr

Práce ukázala, že software *REXYGEN* je vhodným nástrojem pro regulaci v průmyslových procesech. Nabízí širokou knihovnu funkčních bloků a množství ukázkových projektů, které usnadňují pochopení práce s tímto softwarem. Zároveň tým *REX Controls s.r.o.* tento software dále vylepšuje.

Implementované regulátory byly testovány na simulačním modelu s uspokojivými výsledky. Byly zhodnoceny možnosti zobrazení a archivace procesních dat v *REXYGENu*.

Během práce jsem se naučil pracovat s řídicím softwarem *REXYGEN*. Dále jsem se seznámil s komunikací přes průmyslový protokol *OPC*. Práce také zahrnovala použití *Simulinku* pro pseudo-reálnou simulaci, což mi umožnilo testovat řídicí algoritmy v prostředí, které napodobuje skutečný průmyslový proces.

Budoucí plány zahrnují testování navržených regulátorů v reálném *feederu*, což bylo bohužel odloženo na příští rok. Další možností je využití získaných znalostí a zkušeností s *REXYGENem* pro implementaci jiného řídicího systému v dalších průmyslových aplikacích. Jak už bylo zmíněno, *REXYGEN* se ukázal být vhodným nástrojem, který může být použit pro různé typy procesů a aplikací, což otevírá mnoho možností pro budoucí projekty.

Literatura

[1] Ing. Petr Havel, Ph.D, "Modeling and simulation of glass fiber drawing," 2003.

[2] REX Controls s.r.o., "Rexygen - solutions for real-time control," https://www.rexygen.com/, Plzeň, Czech Republic, 2024, [Online; přístup 10.5.2024].

[3] REX Controls s.r.o. , "Opc ua driver manual," https://www.rexygen.com/doc/PDF/ENGLISH/OpcUaDrv_ENG.pdf, 2022, [Online; přístup 10.5.2024].

[4] ControlSoft Inc., *Fundamentals of Process Control*, 1st ed. Avion Park Drive, USA: ControlSoft Inc., 2018.

[5] J. G. Ziegler and N. B. Nichols, "Zieglerova-nicholsova metoda kritických parametrů," http://books.fs.vsb.cz/SyntezaReg/text0302.htm, 2021, [Online; přístup 12.5.2024].

[6] Ing. Filip Vodňanský, "Model-based predictive control for industrial melting furnace," 2023.