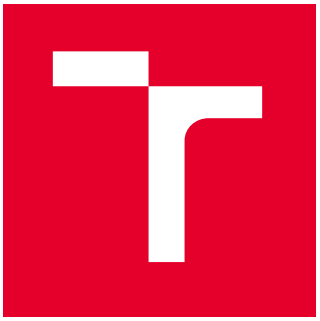


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

APLIKACE PRO SIMULÁTOR DYNAMICKÝCH SYSTÉMŮ

APPLICATION FOR SIMULATOR OF DYNAMIC SYSTEMS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — **Bc. Aleš Korčák**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Miroslav Jirgl, Ph.D.**

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Aleš Korčák ***ID:*** 203257

Ročník: 2 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Aplikace pro simulátor dynamických systémů

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvořit SW realizující výměnu dat mezi fyzickým simulátorem dynamických systémů a prostředím MATLAB. Následně navrhnout a implementovat jednoduchou úlohu pro řízení vybraného systému (Hardware-in-the-loop - HIL simulaci) a zhodnotit použitelnost a omezení celého řešení.

1. Seznamte se s platformou – simulátorem dynamických systémů, osaďte a oživte dodanou DPS.
2. Vydefinujte možnosti ovládání a sběru dat digitální částí simulátoru.
3. Navrhněte a implementujte firmware pro mikrokontrolér na simulátoru zajišťující jednak komunikaci a výměnu dat s připojeným PC a dále ovládání digitálních prvků simulátoru.
4. V prostředí MATLAB vytvořte sadu funkcí či skriptů pro komunikaci se simulátorem, ovládání vydefinovaných částí a měření a vizualizaci signálů.
5. Otestujte funkčnost.
6. Navrhněte a implementujte v prostředí MATLAB/Simulink jednoduchou řídicí úlohu pro demonstraci HIL simulace.
7. Zhodnoťte použitelnost a omezení implementovaného řešení.

DOPORUČENÁ LITERATURA:

[1] PACAL, Stanislav. Řízení reálného systému pomocí PLC s Využitím automaticky generovaného kódu. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, ÚAMT, 2022, 88 s. Diplomová práce.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 17.5.2023

Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

ABSTRAKT

Cílem této práce je vytvořit SW realizující výměnu dat mezi fyzickým simulátorem dynamických systémů a prostředím MATLAB. Součástí práce je také návrh jednoduché řídící úlohy pro vybraný systém.

KLÍČOVÁ SLOVA

Simulátor dynamických systémů, firmware, sběr dat, hardware in the loop, HIL

ABSTRACT

The subject of this thesis is to create a software that implements data exchange between a physical simulator of dynamic systems and the MATLAB environment. The thesis also includes the design of a simple control task for the selected system

KEYWORDS

Simulator of dynamic systems, firmware, data acquisition, hardware in the loop, HIL

KORČÁK, Aleš. *Aplikace pro simulátor dynamických systémů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2023, 65 s. Diplomová práce. Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Aleš Korčák
VUT ID autora:	203257
Typ práce:	Diplomová práce
Akademický rok:	2022/23
Téma závěrečné práce:	Aplikace pro simulátor dynamických systémů

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno 17.5.2023 ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Miroslavu Jirglovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno 17.5.2023

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V dnešní praxi je možné se často setkat s použitím metody hardware in the loop simulace, která umožňuje provádět simulace v prostředí blížící se realitě a to včetně simulace chybových stavů, kterých by na reálném systému nebylo možné dosáhnout bez jeho poškození nebo úplného zničení.

K tomuto účelu jsou simulace vytvořené pomocí softwarových nástrojů jako je například MATLAB/Simulink často nedostatečné, jelikož samotné fungování modelu nedává záruku správné funkčnosti reálného technologického zařízení.

Cílem této práce je vytvořit software realizující výměnu dat mezi fyzickým simulátorem dynamických systémů, který byl navržen a realizován v práci [7], a prostředím MATLAB. Součástí práce bylo osazení a oživení celé DPS a následně ověření funkčnosti implementovaného řešení návrhem a implementací jednoduché úlohy pro řízení vybraného systému na fyzickém simulátoru.

Kapitola 1 popisuje teorii modelování a identifikace systému. Dále je zde popsán System Identification Toolbox a jeho možnosti a také simulace hardware in the loop.

V kapitole 2 jsou popsány části simulátoru dynamických systémů vytvořeného v práci [7], a to jeho analogová a digitální část. Dále jeho napájení a použitý digitální potenciometr. Dále jsou v této kapitole vydefinovány možnosti ovládání a sběru dat na digitální části platformy. Také jsou zde uvedeny úpravy platformy, které bylo nutné provést, aby byla funkční. V závěru kapitoly jsou uvedeny průběhy naměřené na analogové části platformy pomocí PLC.

Kapitola 3 obsahuje návrh a realizaci firmwaru pro simulátor, který bude umožňovat ovládání simulátoru a odesílání dat do PC. Kapitola obsahuje popis vytvořených knihoven pro obsluhu jednotlivých použitých periferií. V závěru kapitoly je uveden popis hlavní smyčky programu.

Kapitola 4 se zabývá návrhem a realizací ovládacích funkcí v prostředí MATLAB. Popisuje vytvořený skript runme sloužící ke spouštění jednotlivých funkcí. Dále popisuje vytvořenou simulaci v programu Simulink pro provádění měření v reálném čase na platformě. V závěru kapitoly jsou uvedeny průběhy naměřené na platformě pomocí vytvořených funkcí a simulace.

Kapitola 5 obsahuje návrh a implementaci demonstrační úlohy na platformě. Kapitola popisuje zvolenou soustavu a její identifikaci a následně návrh regulátoru pro tuto soustavu. A to pomocí integrálního kritéria ITAE a pomocí bloku PID Control.

V závěru kapitoly je uvedeno porovnání výsledků regulace na fyzickém simulátoru dynamických systémů a na simulovaném modelu soustavy v programu simulink.

Kapitola 6 obsahuje zhodnocení použitelnosti a omezení implementovaného řešení, a to z hlediska rychlosti komunikace a dále z hlediska hodnoty periody vzorkování.

1 Modelování a identifikace systémů

Základními pojmy při identifikaci a modelování jsou zkoumaný systém a jeho model. Znalost vlastností zkoumaného systému je v oblasti řízení velmi důležitá. Bez těchto znalostí není možné přesně analyzovat vlastnosti daného systému. [1]

Identifikace a modelování má za cíl získat takový model systému, aby co nejpřesněji popisoval jeho chování. [1]

1.1 Systém

Jedná se o objekt nebo skupinu objektů mezi nimiž existují vzájemné vztahy a jako celek má určité vztahy ke svému okolí.

Systém je charakterizován dvěma základními vlastnostmi [1]:

1. Svým chováním tedy závislost mezi podněty okolí, které působí na jeho vstupy a příslušnými odezvami na jeho výstupu.
2. Struktura systému charakterizuje jeho vnitřní funkční stavy tedy způsob uspořádání vzájemných vazeb mezi jednotlivými prvky systému a jejich chování.

Vnější vlivy na systém, které nemůže pozorovatel ovlivnit se nazývají poruchy. Ty mohou být přímo měřitelné případně, je možné je měřit jen skrze ovlivněný výstupní signál. [1]

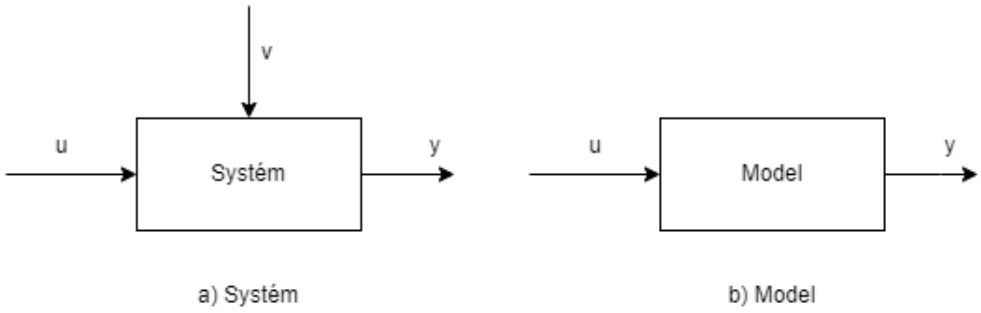
Obrázek 1.1 a) obsahuje schéma systému, vstup u je ovlivnitelný vstupní signál, v je neměřitelná porucha a y je výstupní signál. [2]

1.2 Model

Model je umělý systém zjednodušeně popisující reálný zkoumaný systém používaný pro bližší pochopení složitých systémů. Model zachycuje podstatné části vlastností systému a naopak nepodstatné části zanedbává. [1]

Úkolem modelu je co nejvěrohodnější reprodukce či predikce chování původního systému. Avšak přílišná komplexnost popisu systému je při modelování nežádoucí, jelikož zapříčiní vysokou složitost modelu a těžkopádnost analýzy.

Obrázek 1.1 b) zobrazuje schéma modelu, kde u je vstupní signál a y je výstupní signál. [2]



Obr. 1.1: Ukázka systému a modelu

1.3 Identifikace

Jedná se o proces určení struktury a parametrů modelu na základě vyšetřování dynamických vlastností reálného systému.

Model je sestaven na základě pozorovaných dat. V automatizační technice jsou používány v zásadě tři cesty k získání matematického modelu. Tyto cesty jsou [2]:

- Analytický způsob identifikace (white box)
- Experimentální způsob identifikace (black box)
- Kombinace předchozích přístupů (grey box)

Analytický způsob identifikace

Tento způsob spočívá v rozdělení identifikovaného systému na dílčí podsystémy, které je již možné matematicky popsat na základě empirických znalostí. Takto popsané systémy je možné matematicky spojit a tím získat matematický model celého systému. Získávání modelu tímto způsobem se nazývá modelování. Při tvorbě modelu tímto způsobem není nutné provádět experimentální měření na zkoumaném reálném systému.[2] [1]

Experimentální způsob identifikace

Při použití tohoto způsobu identifikace je matematický model systému získán vyhodnocením odezvy systému na definovaný vstupní signál. [2]

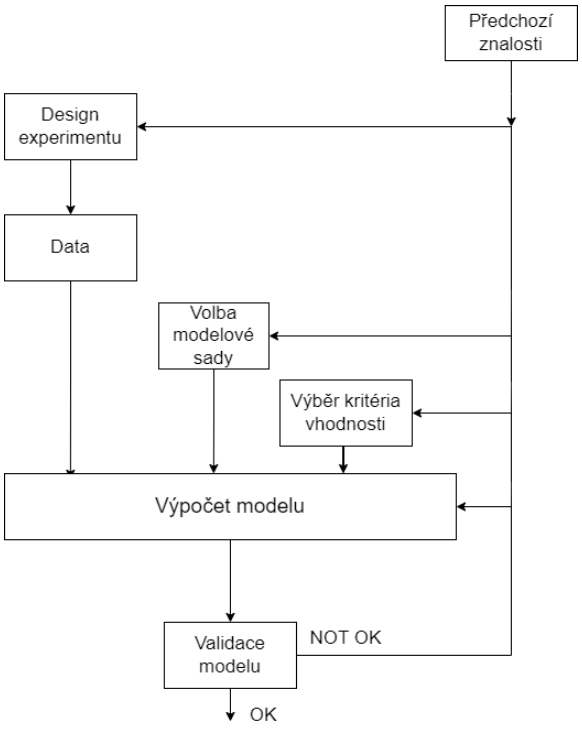
Během používání tohoto způsobu identifikace je se zkoumaným systémem zacházeno jako s černou skřínkou a jedná se tedy o vnější popis systému. Tento způsob získání modelu se nazývá systémová identifikace. [2]

1.4 Postup při identifikaci systému

Při procesu identifikace je nejprve nutné rozhodnout jaká data je nutné naměřit, jaké vstupní signály bude nutné použít a v jakých časových okamžicích bude měření prováděno, aby bylo možné získat o měřeném systému co nejvíce informací. Dalším důležitým krokem identifikace systému je volba vhodného modelu pro popis zkoumaného systému. Tento model je volen na základě vlastností systému a na základě znalostí a zkušeností experimentátora.[2]

Po získání modelu je nutné na základě vhodného kritéria posoudit kvalitu modelu. Tento proces je postaven na tom jak přesně je schopen model reprodukovat naměřená data na zkoumaném systému. Podle zvoleného kritéria je následně vyhodnoceno, zda model dosahuje požadované přesnosti. Jelikož je model pouze aproximací zkoumaného systému nebude se nikdy chovat jako skutečný zkoumaný systém. [2]

Tvorba modelu je velmi často iterativní proces, kdy jsou jednotlivé části postupu opakovány, aby byl získán model s požadovanou přesností. Blokové schéma postupu identifikace modelu je na obrázku 1.2



Obr. 1.2: Blokové schéma postupu při identifikaci systému.[2]

1.5 Matlab System Identification Toolbox

Tato podkapitola se zabývá popisem System Identification Toolbox pro program MATLAB.

Tento toolbox umožňuje návrh matematických modelů dynamických systémů pomocí naměřených vstupních a výstupních dat systému. Tímto postupem je možné modelovat systémy obtížně popsatelné pomocí základních fyzikálních zákonů.

Pro jednodušší organizaci dat a modelů nabízí toolbox grafické rozhraní skrze které je možné aplikaci ovládat. Následně je z grafického prostředí možné vytvořené modely importovat do prostředí MATLAB či Simulink a dále s nimi pracovat. [6]

Naměřená vstupní a výstupní data systému pro použití při identifikaci je možné vkládat v časové oblasti, frekvenční oblasti nebo jako objekty vytvořené pomocí funkcí *IDDATA* nebo *IDFRD*. Tento toolbox umožňuje vytvoření modelů ve formě spojitých nebo diskrétních přenosových funkcí, ale také ve formě nelineárních modelů či popisu ve stavovém prostoru. [6]

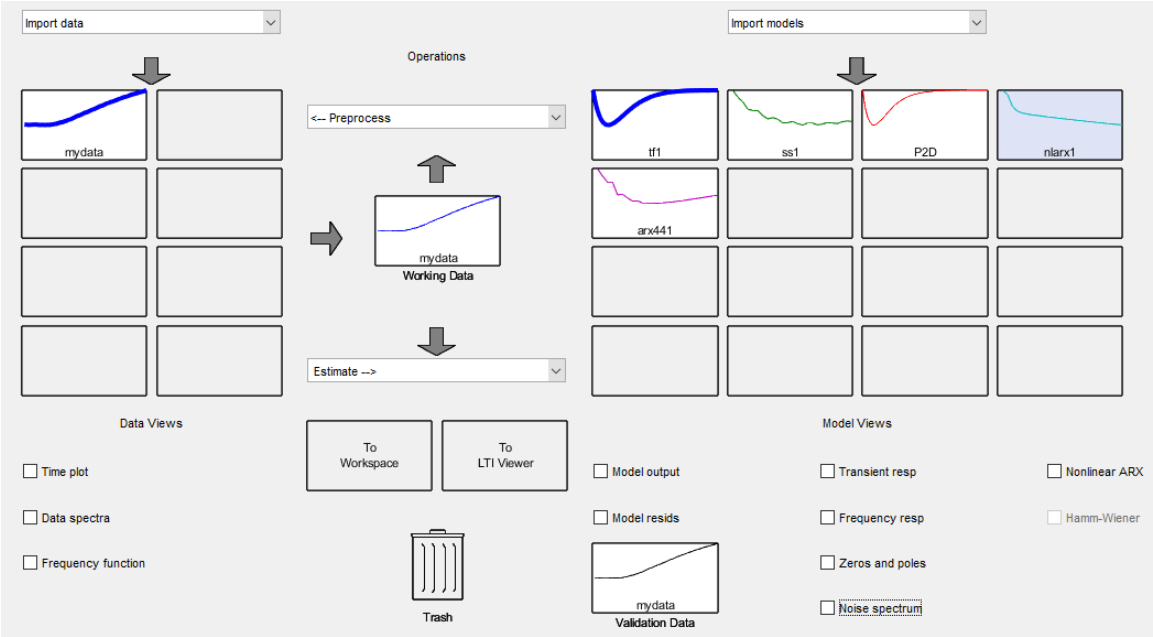
Grafické prostředí toolboxu je rozděleno na tři části. Ukázka grafického prostředí tooboxu je na obrázku 1.3.

V levé části prostředí je umožněno vkládání naměřených dat. Ta je možné vkládat naměřená v diskrétním čase, ve frekvenční oblasti, případně jako datový objekt vytvořený pomocí funkcí *IDDATA* nebo *IDFRD*, a to pomocí rozevíracího menu *Import data*. Po nahrání dat jsou data zobrazena v jednom z polí v levé části prostředí. V rámci této práce byla do toolboxu vkládána data naměřená v diskrétním čase, tedy ve formátu *time domain data*. [6]

Pro vložená data je možné vykreslit časové průběhy, datové spektrum a závislost na frekvenci pomocí funkcí umístěnými v levé spodní části prostředí. Pro výběr dat na kterých je potřeba pracovat je nutné vybraná data přetáhnout do pole v prostření části obrazovky.

Na načtených datech je možné provést jejich předzpracování, k tomuto účelu slouží rozevírací menu *Preprocess*. To nabízí možnosti k odstranění nedostatků vstupních dat před identifikací jako jsou například chybná data nebo případně zašuměná data. [6]

Pro identifikaci modelu slouží rozevírací menu *Estimate*, ve kterém je možné vybrat vhodný model a jeho strukturu. Struktura modelu je důležitá pro kvalitu výsledného modelu a je volena v závislosti na předchozích znalostech o zkoumaném systému.



Obr. 1.3: Hlavní obrazovka System Identification Toolboxu

Toolbox nabízí metody k odhadu lineárních a nelineárních modelů:[6]

- Přenosové funkce
- Procesní modely
- Modely ve stavovém prostoru
- Polynomické modely
- Nelineární ARX modely
- Nelineární Hammerstein-Wiener modely

Výsledky identifikace modelů jsou zobrazeny v pravé části aplikace. Zde je možné modely porovnávat s ostatními a hodnotit jejich kvalitu. Dále je možné vykreslovat impulsní a přechodovou charakteristiku, frekvenční odezvu systému či rozložení pólů a nul modelu. Pro vykreslení těchto průběhů slouží funkce v pravé spodní části aplikace. [6]

Pro porovnání kvality modelu s ostatními slouží funkce *Model ouput*. Funkce také obsahuje ohodnocení přesnosti modelu pomocí parametru *Best fit*. Rovnice pro výpočet parametru Best fit je

$$Best\ Fit = (1 - \frac{|y - \hat{y}|}{|y - \bar{y}|}) \times 100, \tag{1.1}$$

kde y je naměřená hodnota, $\hat{y}$ je simulovaná nebo předpokládaná hodnota modelu a $\bar{y}$ je průměrná hodnota y. [13]

Po vytvoření dostatečně přesného modelu je možné daný model exportovat do prostředí MATLAB či do aplikace *LTI Viewer*, kde je možné následně s modelem dále pracovat.

Nepotřebná či chybná vstupní data, případně vytvořené modely, je možné odstranit přetažením příslušného modelu na ikonu odpadkového koše.[6]

1.6 Simulace hardware in the loop

Velkou výhodou testovací metody hardware in the loop (HIL) je především v snížení nákladů a přesnosti získaných výsledků, které se téměř shodují s reálnými. Je tedy kladen vysoký důraz na výkonnost použitého hardwaru a samotnou realizaci této metody, kdy je nutné nalézt prostředky pro tuto metodu. Tyto počáteční náklady se mnohonásobně vrátí v pozdější fázi vývoje, kdy není nutná výroba drahých prototypů a jejich testování.[3]

Mezi další výhody testovací metody HIL je umožnění vývojářům provádět testování během různých fází vývoje, díky čemuž je možné zvýšit efektivitu a produktivitu. Testováním v reálném prostředí je možné získat informace o chování součástí produktu v různých podmínkách, což může vést k vylepšení výkonu a spolehlivosti produktu.[3]

Metoda testování HIL má velmi široké možnosti uplatnění například v letectví, armádě, ale také v automobilovém průmyslu, kde se používá pro simulaci elektromechanických systémů. [3]

1.6.1 In the loop metody

Správnou funkčnost reálné soustavy s reálnou řídící jednotkou nemusí být na matematickém modelu vytvořeném pomocí softwarového nástroje zaručena. Tyto nástroje jako je například MATLAB/Simulink idealizují systémy a jejich chování, což může v praxi při propojení takto vytvořených zařízení způsobit nefunkčnost řešení.[3]
Mezi in the loop metody se řadí:

- Model in the loop (MIL)
- Software in the loop (SIL)
- Processor in the loop (PIL)
- Hardware in the loop (HIL)

1.6.2 Model in the loop (MIL)

Jedná se idealizovaný prvotní návrh, kde se neuvažují například vzájemné propojení zařízení, nelinearity zařízení, výpočtový výkon a rychlost komunikace.

Řízená soustava je pomocí identifikace pouze aproximovaná, případně se jedná pouze o definované odezvy na jednotlivé vstupy. Návrh a tedy vytvořený model nedosahuje požadovaných přesností [3]

1.6.3 Software in the loop (SIL)

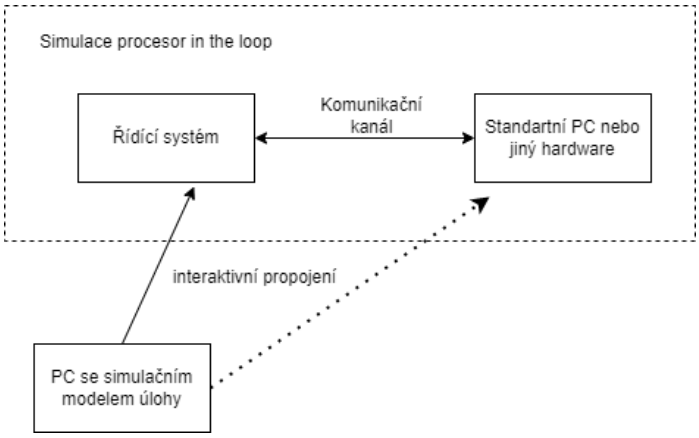
Testování je prováděno na spustitelném kódu, který je vygenerován z celého nebo jen z části modelu realizovaného například v nástroji Simulink. Tento vygenerovaný kód je následně spuštěn na počítači, na kterém probíhá vývoj. Simulace tak netestuje kód kompilovaný pro cílový hardware, ale kód kompilovaný pro počítač na kterém probíhá vývoj. Například je porovnávána funkčnost navrženého modelu a funkčnost vygenerované S-funkce spuštěné v Simulinku. [5]

1.6.4 Processor in the loop (PIL)

Pokud dosahuje simulace pomocí metody MIL rozumných výsledků je možné přistoupit simulaci metodou PIL. Tato simulace již vyžaduje reálný hardware, a to v podobě například univerzální vývojové desky se stejným procesorem, který bude následně použit pro řízení výsledné technologie. Použití reálné soustavy v tomto bodě není nutné.

Model soustavy poběží na libovolném PC a model řídící jednotky je nahrán do vývojové desky. Tato dvě zařízení jsou propojena komunikačním rozhraním, obvykle RS-232, sběrnice CAN či Ethernet. Celá simulace se řídí z PC, které obsahuje model řešené úlohy. Simulaci je možné spouštět v reálném čase. Při použití simulace PIL je možné ověřit funkčnost za přítomnosti některých vlivů přítomných i při řízení reálného zařízení, které nejsou dostupné v softwarovém simulačním nástroji.[3]

Blokové schéma simulace processor in the loop je na obrázku 1.4



Obr. 1.4: Blokové schéma PIL simulace[3]

1.6.5 Princip Hardware in the loop (HIL)

Jedná se o metodu, kde je v řídící smyčce zapojená část reálného hardwaru, a to například řídící jednotka společně s matematickým modelem řízeného systému, který je na HIL testovací platformě simulován.

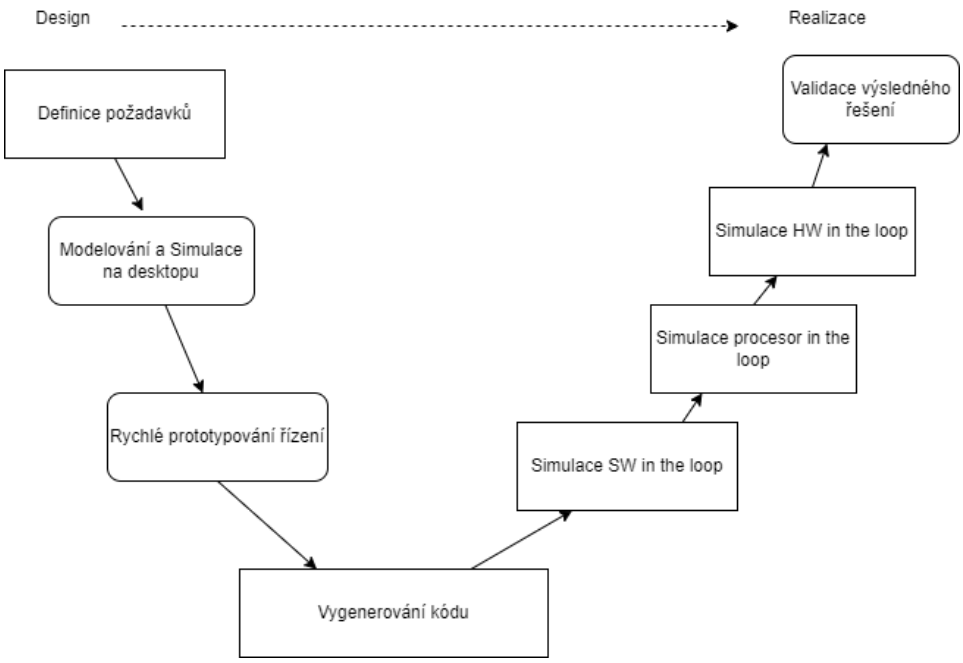
Model soustavy běží na speciálním hardware. Ten simuluje chování soustavy pomocí reálného komunikačního rozhraní, tedy pomocí A/D a D/A převodníků. Tento postup zapříčiní, že řídící jednotka nepozná zda komunikuje s reálnou soustavou a nebo s jejím simulovaným modelem. Akční zásahy a připojené snímače jsou realizovány stejně jako tomu bude i ve výsledné aplikaci. Výhodou je také možnost simulace poruch na simulované soustavě, a to i takové, které by nebylo možné simulovat na reálné soustavě, nastavením hodnot některých snímačů na konstantní nebo chybové hodnoty.

Další nespornou výhodou je opakovatelnost podmínek simulace a její případné zrychlení, což při měření na reálné soustavě je mnohdy velmi obtížné. Tato metoda také šetří časové a finanční prostředky, jelikož znemožňuje poškození testované soustavy. [3]

Blokové schéma zapojení této metody simulace je velmi podobné blokovému schématu na obrázku 1.4, avšak místo použití vývojové desky je využita již výsledná technologie.[3]

1.6.6 Model based design (MBD)

Tento vývojový cyklus je možné popsat na modelu ve tvaru písmene V. Tento vývojový cyklus je na obrázku 1.5.



Obr. 1.5: Simulace metodou HIL během vývojového cyklu MBD[4]

Ve vývojovém cyklu jsou během vývoje využívány metody popsané výše a to hlavně pro testování, měření na zařízení a na kalibraci. Metoda HIL je často používána pro realizaci soustavy, na které je měření prováděno.

Vstupem modelu na obrázku 1.5 jsou vstupní požadavky na výsledné zařízení. Po definování požadavků se přechází do oblasti modelovaní a modelové analýzy. Poté následuje rychlá tvorba prototypových zařízení.

Na levé straně modelu následuje systémová integrace a následné testování produktu. Na konci tohoto cyklu je výsledné řešení validováno a následně může být zařízení nasazeno v reálném provozu. [4]

2 Popis platformy a možností jejího ovládání

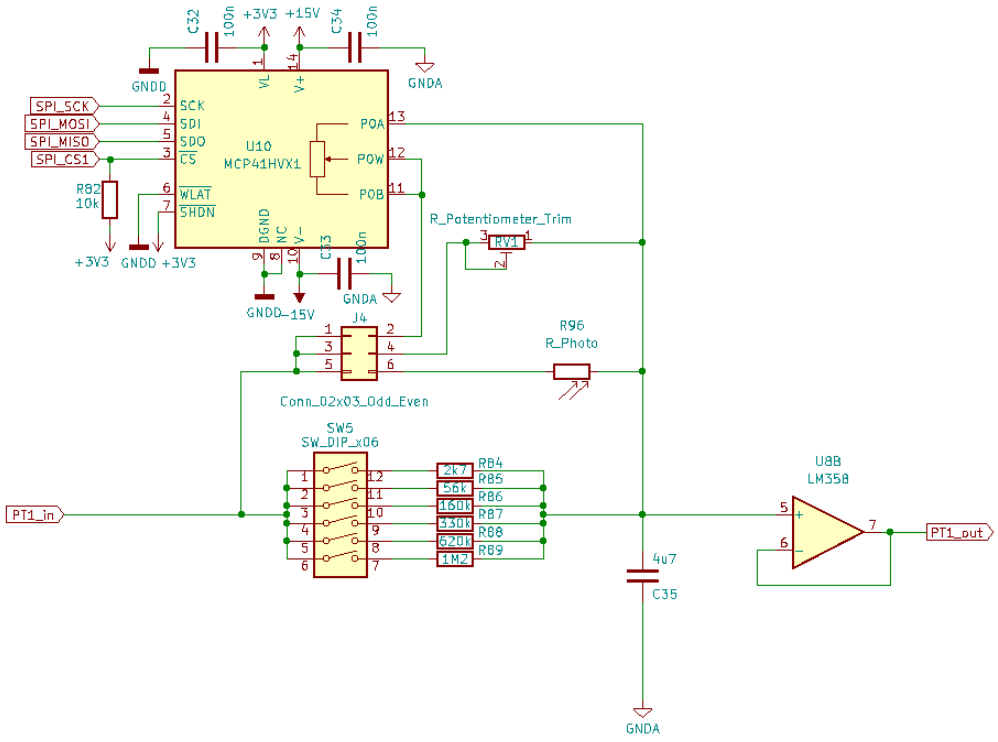
Tato kapitola se zabývá popisem platformy vytvořené v práci [7] a popisem možností jejího ovládání pomocí digitální části.

2.1 Analogová část

Analogová část obsahuje pět samostatných bloků, které mohou být nebo nemusí být připojeny do obvodu. Tyto bloky je možné do obvodu připojit v libovolném pořadí. Analogová část obsahuje tyto bloky:

Setrvačný člen prvního řádu

První blok je setrvačný člen prvního řádu realizovaný jako RC článek, jehož časovou konstantu je možné měnit pomocí změny velikosti odporu. Změna odporu je umožněna šestimístným DIP přepínačem *SW5*. Ke zvolenému odporu je možné paralelně připojit proměnné odpory simulující působení poruchy/nelinearity. Tyto proměnné odpory jsou fotorezistor, otáčkový potenciometr a číslicový potenciometr.[7]



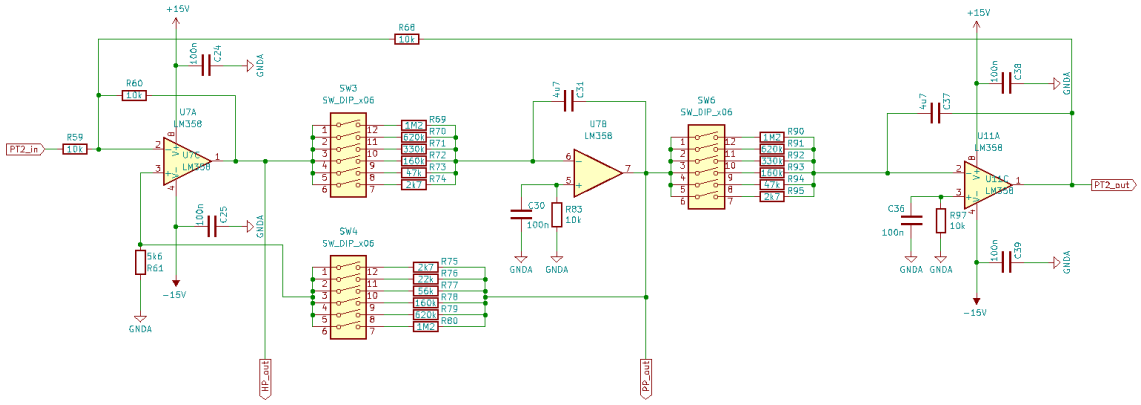
Obr. 2.1: Setrvačný člen prvního řádu

Přenos tohoto členu je

$$F(p) = \frac{1}{1 + \Delta RCp}, \tag{2.1}$$

Setrvačný člen druhého řádu

Dalším blokem je setrvačný článek druhého řádu realizovaný jako zapojení se třemi operačními zesilovači. Změnou odporu, která je umožněna pomocí DIP přepínače, je možné měnit parametry zesílení a tlumení.[7]



Obr. 2.2: Setrvačný člen druhého řádu

Přenosová funkce tohoto členu je

$$F(p) = -\frac{\omega^2}{p^2 + p\omega(2+m)d + \omega^2}, \tag{2.2}$$

kde m = 1 a

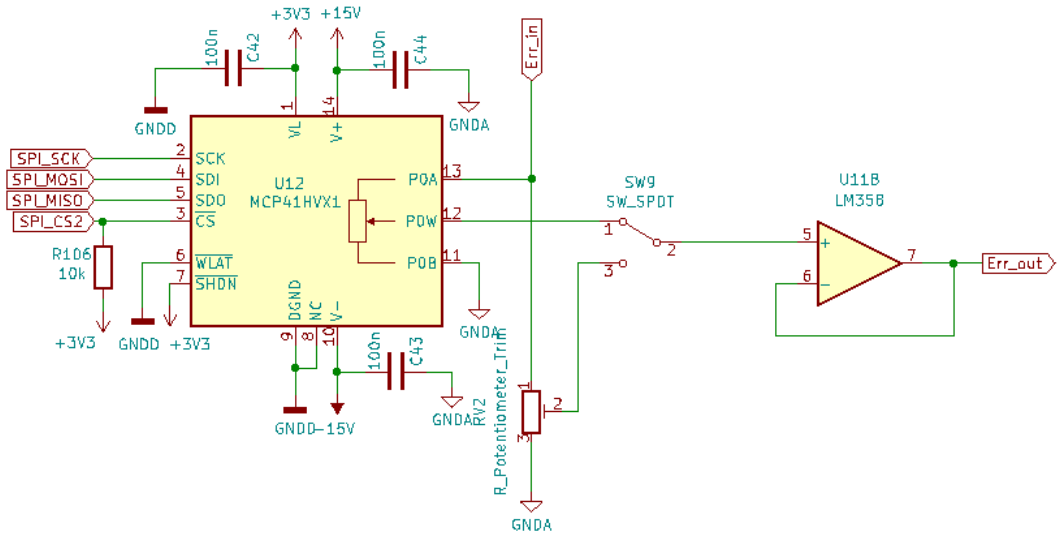
$$d = \frac{R_2}{R_2 + \Delta R_5}, \tag{2.3}$$

$$\omega^2 = \frac{1}{C\Delta R_4}, \tag{2.4}$$

kde Δ R je nastavován pomocí SW4, ΔR_4 je nastavován pomocí *SW3* a *SW6*. Aby výsledná přenosová rovnice odpovídala tvaru 2.2 musí být přepínače *SW3* a *SW6* nastaveny na stejnou hodnotu.[7]

Rozdílový člen

Pro simulaci chyby působící na systém slouží rozdílový člen. Ten je realizován jako dělič použitím otáčecího potenciometru, případně pomocí digitálního potenciometru, jehož dělící poměr je možné nastavit pomocí mikroprocesoru. Změnou odporu potenciometru je měněn dělící poměr děliče a tím je snižováno, případně zvyšováno výstupní napětí. Pro přepínání mezi otáčkovým potenciometrem a digitálním potenciometrem slouží DIP přepínač *SW9*.[7]



Obr. 2.3: Rozdílový člen

Proporcionální člen

Pro simulaci proporcionálních systémů slouží proporcionální člen, realizovaný jako invertující zapojení operačního zesilovače s nastavitelnými parametry, kterými je možné měnit zesílení. Při nastavení zesílení na hodnotu K $\doteq$ 1 je možné tento člen použít jako invertor signálu.[7]

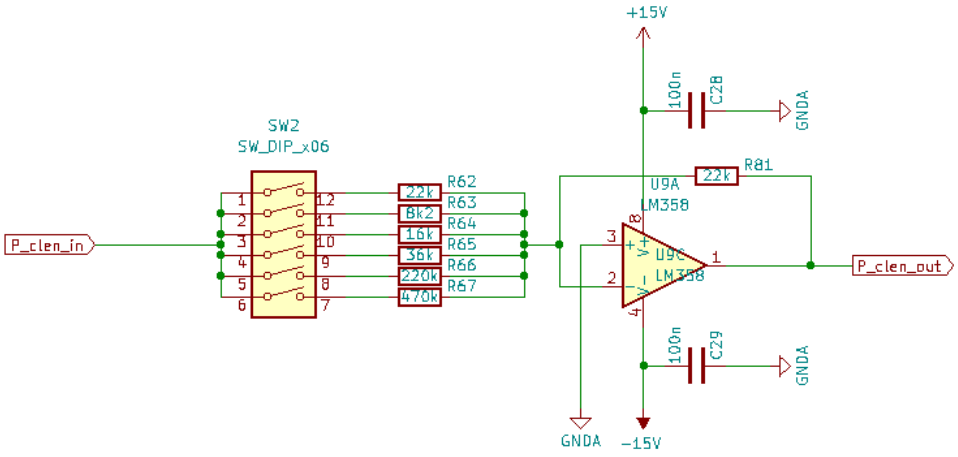
Přenosová rovnice tohoto členu je

$$F = -\frac{R_2}{\Delta R_1}, \tag{2.5}$$

kde ΔR_1 je nastavitelný pomocí *SW2*.

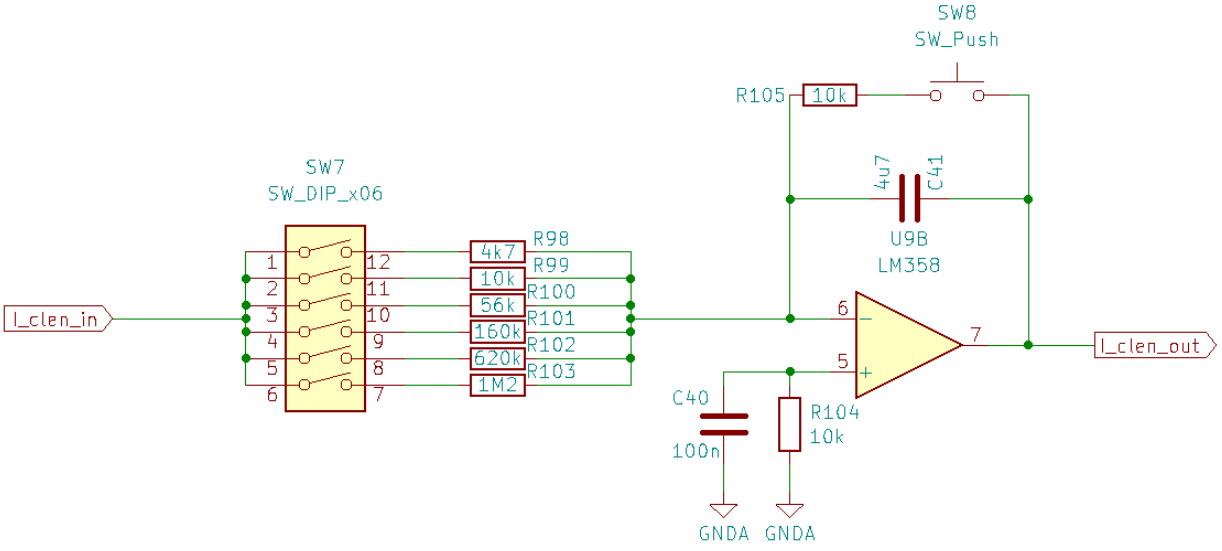
Integrační člen

Posledním blokem je integrátor s nastavitelnou rychlostí integrace pomocí kterého je možné simulovat soustavy s astatismem. Pro vybití kondenzátoru, který po přivedení vstupního signálu drží na výstupu hodnotu naintegrovaného napětí, slouží vybíjecí



Obr. 2.4: Proporcionální člen

obvod, tvořený odporem a tlačítkem, paralelně připojený ke kondenzátoru.[7]



Obr. 2.5: Integrační člen

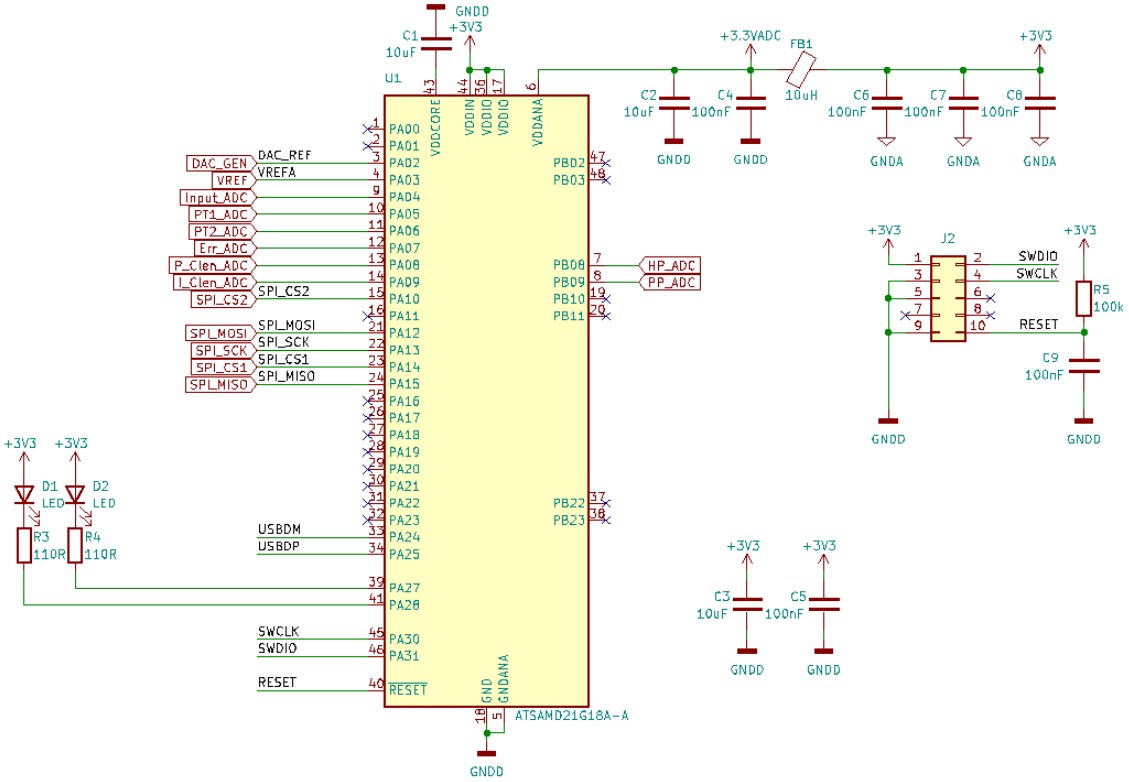
Přenosová rovnice integračního členu je

$$F = \frac{1}{\Delta RCp}, \tag{2.6}$$

V případě, že se integrační člen a systém druhého řádu dostanou do saturace, jsou rozsvíceny příslušné LED v pravém dolním rohu na horní straně desky.

2.2 Digitální část

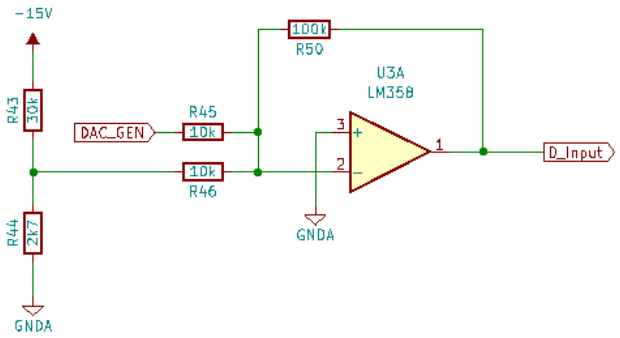
Pro ovládání digitální části slouží mikroprocesor *ATSAMD21G18A*, z periferií procesoru jsou využity ADC převodník pro měření výstupů z jednotlivých bloků, DAC pro možnost generování vstupního signálu do analogové části a pro komunikaci s číslicovými potenciometry slouží periferie SPI. Pro programování mikroprocesoru slouží kolíkový programovací konektor a pro komunikaci slouží USB-C. Schéma zapojení mikrokontroléru a kolíkového programovacího konektoru je na obrázku 2.6[7]



Obr. 2.6: Schéma zapojení mikrokontroléru

2.2.1 Digitálně analogový převodník

Pomocí integrovaného desetibitového digitálně analogového převodníku je možné generovat vstupní signál pro analogovou část. Převodník umožňuje generovat napětí v rozsahu 0 až 2 V. Výstupní signál převodníku je následně zesílen pomocí součtového zesilovače, aby bylo možné generovat napětí v rozsahu ± 10 V. Schéma zapojení pro obvod pro převod napětí je uvedeno na obrázku 2.7 [7]



Obr. 2.7: Schéma zapojení obvodu pro převod napětí [7]

2.2.2 Analogově digitální převodník

Mikroprocesor obsahuje dvanáctibitový AD převodník až s třiceti dvěma vstupy. Převodník umožňuje také rozlišení osm a deset bitů. Tato nižší rozlišení je možné volit pro snížení doby převodu. Převodník je také možné nastavit na rozlišení šestnáct bitů. Toto rozlišení je nejpřesnější z výše zmíněných, avšak je také nejpomalejší. Pro nastavení tohoto rozlišení je nutné zapnou také průměrování hodnot. Výstupní napětí na jednotlivých blocích desky plošných spojů může dosahovat ±13,5 V, jelikož maximální vstupní napětí převodníku je 2,7 V, je toto napětí před vstupem do převodníku převedeno na rozsah 0 až 2 V.

2.2.3 Napájení platformy

Pro napájení operačních zesilovačů slouží stepdown stejnosměrný měnič *DCW08B-15*, který převádí vstupní napětí na rozsah ±15 V pro symetrické napájení operačních zesilovačů.

Pro napájení digitální části slouží měnič *LM317*, který převádí +15 V z měniče *DBW08B-15* nebo +5 V přivedených přes USB-C. Při použití tohoto zapojení je možné komunikovat s procesorem, i když je odpojeno napájení analogové části.[7]

2.2.4 Digitální potenciometr MCP41HV31

Jedná se o digitální potenciometr s odporem 10 kΩ se sedmibitovým rozlišením. Odporová síť se skládá z modulu rezistorů, jezdce a ovládání vypnutí. Modul rezistorů

obsahuje do série zapojené odpory stejné hodnoty, které svou velikostí odpovídají velikosti jednoho kroku. Pro zařízení se sedmibitovým rozlišením je počet rezistorů 127, pomocí jezdce je možné se připojit k libovolnému rezistoru a je tedy možné nastavit 128 možných pozic. Odpor jednoho kroku je stanoven pomocí zjednodušené rovnice[8]

$$R_S = \frac{R_{AB}}{n} \tag{2.7}$$

kde R_S je odpor jednoho kroku
R_{AB} je maximální odpor mezi piny A a B
a n je počet rezistorů v modulu rezistorů pro sedmibitový potenciometr s odporem 10 kΩ. Hodnota odporu pro jeden krok je tedy

$$R_S = \frac{10000}{127} = 78,74\ \Omega \tag{2.8}$$

Pro komunikaci slouží protokol SPI. Tento je možné využívat na frekvenci 1 MHz nebo 10 MHz. Digitální potenciometr je přes SPI možné ovládat pomocí čtyřech příkazů, které jsou [8]

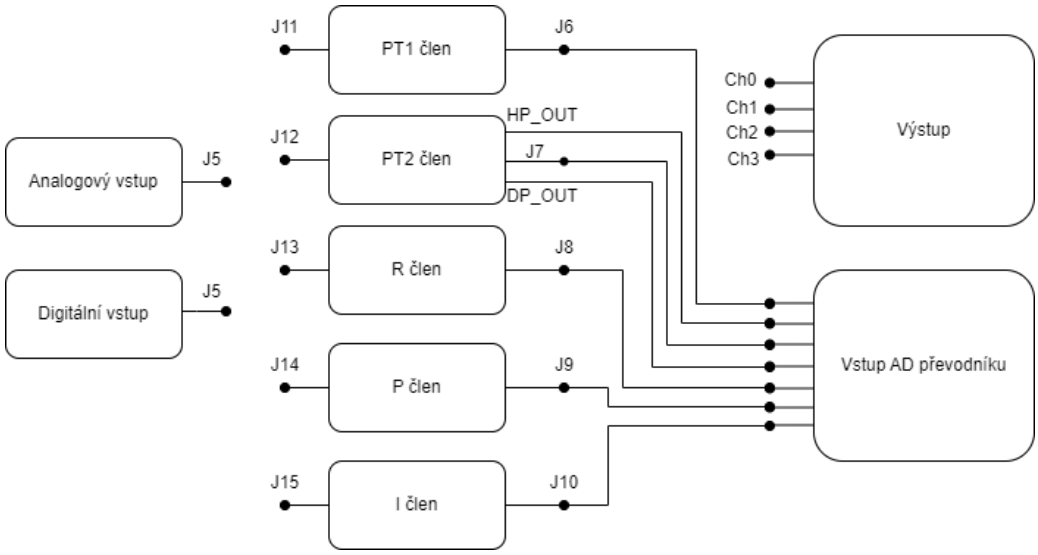
- *Read data* - jedná se o šestnáctibitový příkaz, kterým je možné vyčítat aktuální nastavenou polohu jezdce.
- *Write data* - šestnáctibitový příkaz. Datový byte obsahuje pozici, na kterou má být nastaven jezdec.
- *Increment Wiper* - osmibitový příkaz zvyšuje hodnotu pozice jezdce o jednu pozici.
- *Decrement Wiper* - osmibitový příkaz pro snížení hodnoty jezdce o jednu pozici.

Pokud je jezdec nastaven na maximální hodnotě odporu a přijde příkaz pro zvýšení hodnoty pozice jezdce, je tento příkaz ignorován. Obdobně pokud je nastavena nulová hodnota odporu a přijde příkaz na snížení pozice jezdce, je tento příkaz také ignorován.[8]

2.3 Možnosti ovládání platformy pomocí digitální části

Jednotlivé bloky je možné řadit za sebe v libovolném pořadí pomocí propojení přes kolíkové piny pomocí propojovacích vodičů. Všechny členy mají na levé straně vstupní kolík a na pravé straně výstupní kolík. Všechny bloky jsou také z výstupu připojeny na vstup mikroprocesoru, kterým je možné měřit průběhy na všech blocích

na platformě. Pro blok setrvačného členu druhého řádu jsou na vstupy mikrokontroléru přivedeny i výstupy za dílčími operačními zesilovači, tedy výstupy za horní propustí a za dolní propustí.
Pomocí přepínače *SW1* je možné připojit na vstup do analogové části výstup digitálně analogového převodníku na mikroprocesoru, tím je možné generovat libovolný vstupní signál pro analogovou část platformy. Setrvačný člen prvního řádu a rozdílový člen obsahují digitální potenciometr. Ten je možné nastavovat pomocí mikroprocesoru použitím funkcí popsaných v kapitole 2.2.4. Pro připojení digitálního potenciometru slouží *SW9* pro rozdílový člen a pin číslo dva na jumperu *J4*.



Obr. 2.8: Blokové schéma platformy

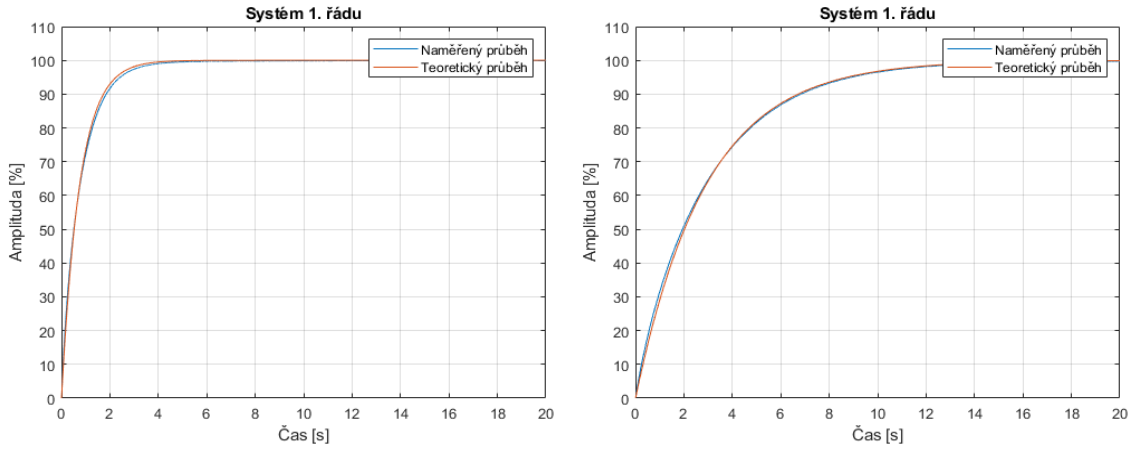
2.4 Problémy při oživování platformy

Při testování platformy po osazení se vyskytlo několik problémů, které byly způsobeny během výroby desky plošných spojů. Nejprve bylo nutné vyměnit vstupy na operačních zesilovačích LM358 sloužících k detekci saturace integračního členu a setrvačného členu druhého řádu. Vývody 5 a 6 na pouzdře operačního zesilovače U2 a U4 bylo nutné vzájemně zaměnit.
Dále byla u operačních zesilovačů špatná orientace diod D11, D12, D14 a D15, tato chyba způsobila nefunkčnost obvodů pro indikaci saturace těchto členů. Dalším problémem byla špatná orientace D17 a D18 z obvodu pro převod napětí +5 V

přivedeného přes USB-C nebo +15 V přivedeného z *DCW08B-15* na hodnotu 3.3 V pro napájení mikrokontroléru. To zapříčinilo, že mikrokontrolér nebyl napájen a nebylo možné jej programovat.
Při testování funkčnosti platformy se objevilo napětí 10 V na analogovém vstupu platformy, pokud nebylo na vstup platformy přiváděno žádné napětí.To bylo způsobeno nedokonalostí operačního zesilovače. Tento problém byl odstraněn přidáním odporu velikosti $1,2\ M\Omega$ paralelně k diodě D22.

2.5 Ověření funkčnosti platformy

Funkčnost analogové části platformy byla otestována pomocí PLC SIMATIC S7-1500. Výstup karty s analogovými vstupy a výstupy byl přiveden na analogový vstup platformy. Výstup z příslušného bloku byl přiveden na analogový výstup platformy na kanál 3, který je připojen na vstup analogové karty. Průběh signálu byl naměřen pomocí nástroje Traces v programu TIA portal V15.1, tento průběh byl následně uložen jako soubor typu csv, aby bylo možné jej dále zpracovávat v prostředí MATLAB. Zde jsou tyto průběhy vykresleny a porovnány s teoretickými průběhy vypočtenými dle uvedených přenosových rovnic bloků uvedených v kapitole 2.1.



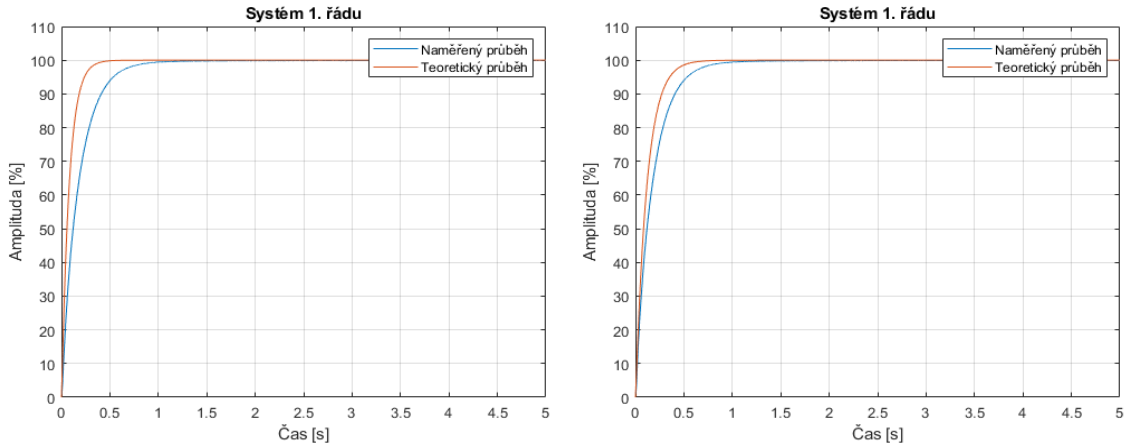
(a) PT1 člen SW5-3

(b) PT1 člen SW5-5

Obr. 2.9: Porovnání naměřených průběhů s teoretickými průběhy pro setrvačný člen prvního řádu.

V grafu 2.9 lze vidět naměřené průběhy pro setrvačný článek prvního řádu pro různé nastavení odporů pomocí DIP přepínačů. Byly naměřeny průběhy pro odpory $\Delta R = 160\ k\Omega$ a $\Delta R = 620\ k\Omega$.

Pro setrvačný článek prvního řádu byly naměřeny také průběhy s paralelně připojeným otáčkovým potenciometrem. Pro odpor $\Delta R = 160\ k\Omega$ byly naměřeny průběhy pro nastavení otáčkového potenciometru na $RV1 = 18,6\ k\Omega$ a $RV1 = 29,5\ k\Omega$. Naměřené průběhy jsou v grafu 4.3.



(a) PT1 člen s $\Delta R = 18,6\ k\Omega$ (b) PT1 člen s $\Delta R = 29,5\ k\Omega$

Obr. 2.10: Porovnání naměřených průběhů s teoretickými průběhy pro setrvačný člen prvního řádu s paralelně připojeným otáčkovým potenciometrem.

3 Návrh a realizace firmware pro simulátor

Tato kapitola se zabývá návrhem a implementací firmware pro mikrokotrolér na simulátoru.

3.1 Návrh programového řešení

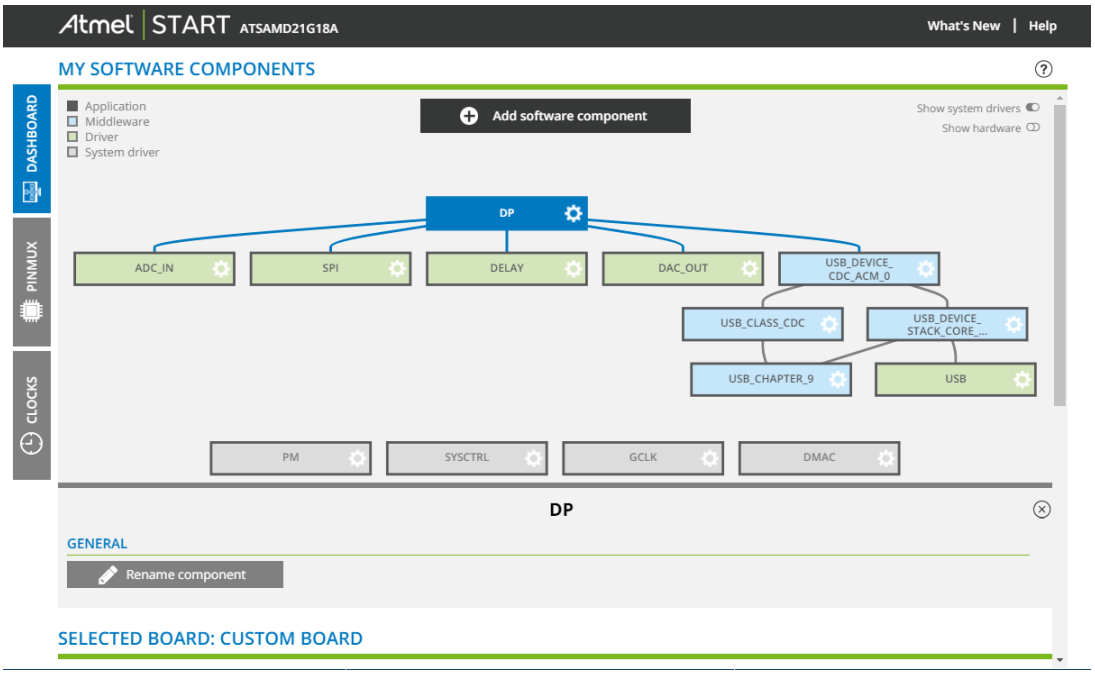
Firmware musí umožňovat příjímání dat z obslužné aplikace, a to jak dat pro nastavení digitálních potenciometrů, tak pro generování signálu pro analogovou část platformy. Pro generování vstupního signálu je nutné data převádět na vstupní rozsah DA převodníku.
Jak bylo definováno v kapitole 2.3 je možné měřit průběhy na všech dílčích blocích. Je tedy nutné, aby firmware umožňoval spolehlivé měření na libovolné konfiguraci analogové části platformy.
Po změření je nutné data převést zpět na hodnoty napětí. Následně je nutné naměřená data odesílat zpět do obslužné aplikace, kde tato data budou zobrazena. Pro komunikaci byla vybrána komunikace pomocí USB, která v režimu full speed umožňuje rychlost komunikace až 12 Mbit/s.
Pro komunikaci s digitálními potenciometry bude sloužit komunikace SPI. Komunikace bude ve tvaru jeden master a více slave zařízení. Mezi slave zařízeními bude možné přepínat pomocí signálu chip select. K realizaci programu byly využity následující prostředky.

3.1.1 Atmel START

Jedná se o webový nástroj pro konfiguraci pro různé softwarové nástavbové knihovny pro vestavné systémy. Nástroj umožňuje vybírat z předepsaných softwarových komponent jako jsou ovladače a middleware pro vytvoření použitelného a optimalizovaného základu pro vyvíjenou aplikaci. Nástroj také obsahuje ukázkové projekty pro různé mikrokontroléry a jejich dostupné periferie.[12]

Nástroj umožňuje:

- Výběr mikrokontroléru dle jak softwarových tak hardwarových požadavků.
- Konfiguraci ovladačů, middleware a projektů s příklady.
- Umožňuje správné nastavení jednotlivých pinů mikrokontroléru pomocí nástroje PINMUX
- Nastavení hodin mikrokontroléru a jednotlivých periferií



Obr. 3.1: Ukázka nastaveného projektu

ASF4

Používané ovladače a middleware je definován v ASF (Advanced Software Framework). Jedná se o open sourceovou knihovnu obsahující ovladače a middleware pro usnadnění a urychlení vývoje uživatelských aplikací. ASF4 je čtvrtá generace knihovny podporující také mikroprocesory z produktové řady SAM. ASF4 je možné využívat pouze s nástrojem Atmel START.[12]

3.2 Realizace programového řešení

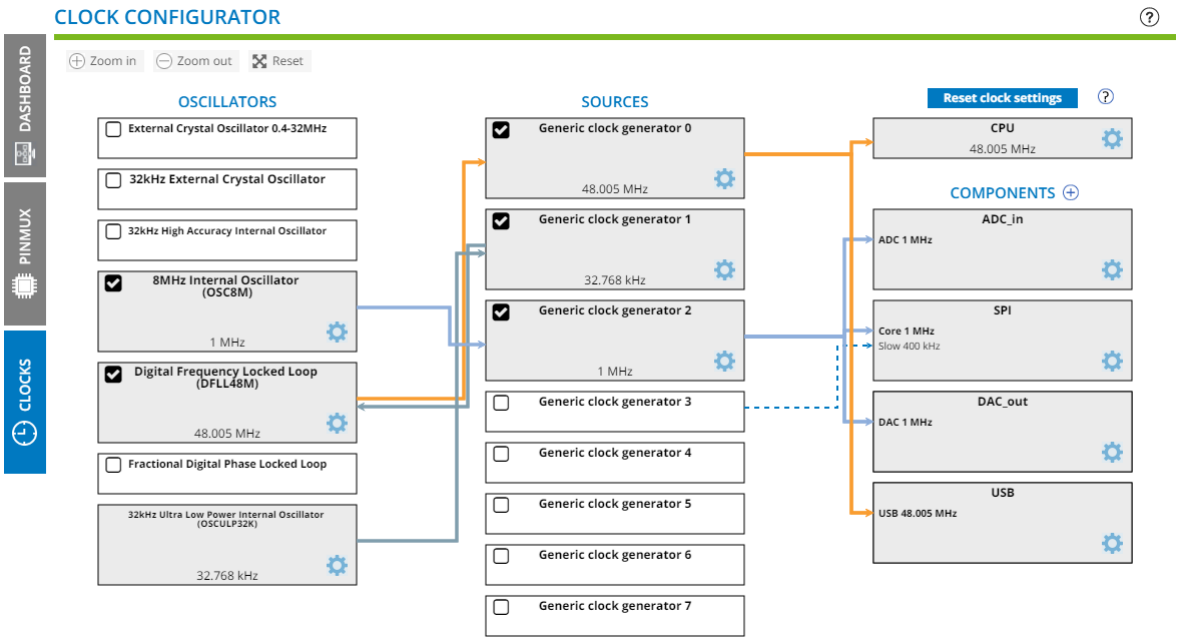
3.2.1 USB

Pro přenos dat mezi platformou a PC byla využita periferie USB, kterou použitý mikrokontrolér obsahuje. Pro vytvoření knihovny pro obsluhu USB byl využit nástroj Atmel START. Z dostupných možných nastavení byla zvolena třída pro komunikační zařízení s abstraktním ovládacím modelem. Komunikace probíhá asynchronně a pro zpracování dat využívá zpětné volání. Jako piny pro komunikaci byly využity piny PA25 jako Data+ a PA24 jako Data-. Pro komunikaci byla nastavena hodnota rychlosti přenosu na plnou rychlost, která podporuje maximální rychlost přenosu až 12 Mbit/s. Velikost bufferů pro příjímání a odesílání dat byla nastavena na 64 bytů.[9]

Hodiny

Pro správnou funkčnost USB komunikace je nutné nastavit hodiny na frekvenci 48 MHz pro nízkou a plnou rychlost komunikace. Pro vysokou rychlost komunikace je nutné nastavit frekvenci 480 MHz.

V této práci byla nastavena maximální hodnota frekvence procesoru 48 MHz. Pro



Obr. 3.2: Nastavení hodin mikrokontroléru

generování této frekvence musí být použita uzavřená smyčka digitální frekvence. Aby bylo možné tuto frekvenci generovat, musí být nastavení hodin v režimu uzavřené smyčky a musí být nastaven násobící faktor na hodnotu 1465. Jako hodinová reference byl zvolen interní oscilátor s frekvencí 32 kHz.[10]

Vytvořené funkce

Pro usnadnění použití funkcí vytvořených v knihovně pro obsluhu USB byly vytvořeny následující funkce.

USB_serial_bytes_available

Funkce má jako návratovou hodnotu počet bytů, které je možné vyčítat. Tato hodnota je průběžně upravována pomocí funkcí knihovny pro čtení vstupů. Pokud není aktivní žádná operace pro čtení, je zahájeno další čtení vstupu.

USB_serial_read

Funkce slouží pro čtení dat přijatých při komunikaci. Jako vstupní parametry slouží buffer pro data dále používaný v programu a jeho délka.
Pokud je buffer pro příjímání dat prázdný a není aktivní žádná operace čtení, je čtení aktivováno pomocí funkce z vygenerovaného API *cdc_read_start*. Pokud jsou data dostupná a délka přijatých dat je menší než velikost bufferu pro data je jako hodnota pro počet dat uložena hodnota délky přijatých dat. Pokud je délka přijatých dat větší než velikost buffer pro data je uložena také hodnota rozdílu hodnot délky a index prvního bytu mimo rozsah.
Aby bylo možné data vyčítat správně je nutné zajistit, aby nedošlo během čtení dat k jejich změně. K tomu slouží funkce z vygenerovaného API
CRITICAL_SECTION_ENTER a *CRITICAL_SECTION_LEAVE*. Ty zajistí, aby nedocházelo ke skoku do přerušení a neočekávané změně dat. V oblasti mezi těmito funkcemi jsou data z bufferu pro příjímání dat přesunuty do bufferu pro data.

Pokud již buffer pro příjímání dat neobsahuje žádná další data, je vynulován. Obsahuje-li tento buffer data, jsou tato data posunuta na začátek tohoto bufferu a již nepotřebná data jsou vynulována.
Funkce vrací jako návratovou hodnotu počet bytů pro čtení.

USB_serial_write

Pro odesílání dat slouží funkce *USB_serial_write*. Jako vstupní parametry má funkce buffer pro odeslání hodnot a délku hodnot pro odeslání. Funkce využívá funkci pro odeslání dat *cdc_write*. Funkce má jako návratovou hodnotu počet odeslaných bytů.

cdc_write

Funkce má jako vstupní parametry buffer pro odeslání hodnot a jeho délku. Funkce prochází tento buffer po znaku, pokud narazí na znak nové řádky nebo je dosaženo maximální hodnoty nastavené velikosti bufferu. Je zavolána funkce z vygenerovaného API *cdcdf_acm_write*, která data odešle. Funkce čeká na dokončení přenosu. Funkce má jako návratovou hodnotu počet odeslaných bytů. Funkce je funční pouze pokud je zařízení připojeno.

3.2.2 ADC

Jako vstupy AD převodníky byly nastaveny piny s postfixem *_ADC* připojené na vstupy mikrokontroléru. Piny použité jako vstupy pro AD převodník jsou na obrázku 2.6. AD převodník byl nastaven na rozlišení dvanáct bitů, to představuje kompromis mezi rychlostí převodu a rozlišením výsledné hodnoty.[9]

Jako reference pro převod byla zvolena externí reference A, která odpovídá 2 V generovaným z napájení analogové části. Hodnota děličky pro hodiny AD převodníku byla nastavena na hodnotu čtyři. Hodiny AD převodníku byly nastaveny na hodnotu 1MHz. Nízkoúrovňový ovladač pro AD převodník byl vytvořen pomocí nástroje Atmel START.

adc_sync_enable_channel

V programu je nutné pro použití AD převodníku nastavit vstupní kanál. K tomu slouží funkce *adc_sync_enable_channel*. Funkce má jako vstupní parametry ukazatel na hardwarovou abstrakci a číslo kanálu.

adc_read_values

K vyčítání dat z AD převodníku slouží funkce *adc_read_values*. V této funkci je volána funkce *adc_sync_read_channel*, která má jako vstupní parametry ukazatel na hardwarovou abstrakci AD převodníku, číslo kanálu, buffer pro uložení vyčtené hodnoty a délka bufferu.

Tato funkce slouží k vyčtení hodnot AD převodníku. Tyto hodnoty jsou v rozsahu 0 až 4095. Výsledné napětí v rozsahu 0 až 2 V je možné vypočítat dle následujícího vzorce:

$$V_{OUT} = \frac{DATA}{4095} \cdot V_{REF} \tag{3.1}$$

kde DATA je hodnota naměřená AD převodníkem,
hodnota *4095* odpovídá maximální hodnotě, kterou je možné změřit pomocí převodníku při rozlišení dvanáct bitů. V_{REF} je referenční napětí o hodnotě 2 V. V této funkci je provedeno průměrování několika naměřených hodnot pro zvýšení přesnosti výsledku. Hodnoty napětí jsou naměřeny v mV a jsou tedy převedeny na V. Aby naměřené napětí odpovídalo celému rozsahu napětí platformy je následně převedeno na rozsah ±13,5 V.

V případě měření více kanálů funkce přepíná měřený kanál a postupně měří hodnoty na požadovaných kanálech po naměření dostatečného počtu hodnot jsou naměřené hodnoty převedeny na rozsah ±13,5 V.

adc_calibrate

Pro zvýšení přesnosti měřených hodnot byla vytvořena adc_calibrate, ta provede kompenzaci nuly AD převodníku tak, že jsou naměřeny hodnoty pro vstupní napětí simulátoru 0 V na požadovaných kanálech. Tato funkce je volána po změnách nastavení vybraných měřených kanálů a hodnot digitálních potenciometrů.

3.2.3 DAC

Nízkoúrovňový ovladač pro DA převodník byl vytvořen pomocí nástroje Atmel START. Jako výstup pro převedené napětí byl využit výstup *PA02*. Jako referenční napětí pro převod slouží externí reference 2 V generovaná z napájení analogové části platformy +15 V . Toto napětí je přivedeno na vstup *PA03*. Aby DA převodník využíval toto referenční napětí je potřeba nastavit ve výběru reference položku externí reference. Hodiny DA převodníku byly nastaveny na 1 MHz. Jak je patrné z obrázku 2.7 je převod napětí pro vstup do analogové části platformy realizován pomocí invertujícího zapojení. Pro získání požadovaných hodnot je tedy nutné generovat hodnoty pro DA převodník pro požadované napětí s opačnou polaritou.

dac_sync_enable_channel
Aby bylo možné využívat DA převodník je nutné nastavit kanál převodníku. K tomu slouží funkce *dac_sync_enable_channel*. Funkce má jako vstupní parametry ukazatel na hardwarovou abstrakci a číslo kanálu.

dac_generate_values
Funkce slouží pro nastavení výstupní hodnoty napětí DA převodníku. Jako vstupní parametr funkce slouží požadovaná hodnota napětí. Tato hodnota je nejprve převedena na rozsah 0 až 2 V. Z této hodnoty je následně určena požadovaná hodnota pro nastavení DA převodníku pomocí následujícího vzorce:

$$DATA = \frac{V_{OUT}}{V_{REF}} * 0x3FF \tag{3.2}$$

kde DATA je hodnota pro nastavení DA převodníku,
V_{OUT} je hodnota požadovaného napětí na výstupu DA převodníku, hodnota *0x3FF* odpovídá maximální nastavitelné hodnotě DA převodníku v šestnáctkové soustavě v dekadické soustavě tato hodnota odpovídá 1023.
V_{REF} je referenční napětí 2 V.

V této funkci je volána funkce *dac_sync_write*, která slouží k provedení převodu a přivedené převedené hodnoty na výstup.

3.2.4 Knihovna pro digitální potenciometry

Pro komunikaci s digitálními potenciometry využívá asynchronní komunikaci SPI. Komunikace využívá piny *PA15* jako *MISO* (Master in Slave out), *PA12* jako *MOSI* (Master out Slave in) a pin *PA13* jako hodinový signál komunikace. Hodiny komunikace jsou nastaveny na 1 MHz, rychlost komunikace je nastavena na 50000 Baudů.

Pro výběr digitálního potenciometru pro nastavení slouží piny *PA10* a *PA14*, které jsou nastaveny jako chip select pro první a druhý potenciometr. Pro tvorbu nízkoúrovňových funkcí pro asynchronní komunikaci SPI byl využit nástroj Atmel START. Pro obsluhu digitálních potenciometrů slouží následující funkce obsluhující potenciometry pomocí kódů funkcí uvedených v podkapitole 2.2.4.

MPC41HV31_init
Funkce slouží pro nastavení a povolení SPI komunikace pro obsluhu digitálních potenciometrů.

MPC41HV31_transfer
Pro přenos dat do digitálního potenciometru slouží funkce *MPC41HV31_transfer*. Funkce má jako vstupní parametry pin určující požadovaný digitální potenciometr, buffer pro zápis, buffer pro čtení a délku odesílaných dat. Funkce nejprve nastaví chip select pin příslušného potenciometru do 0. Následně je zahájen přenos dat pomocí funkce SPI pro přenos dat. Poté se v cyklu provádí kontrola, zda byla komunikace dokončena, pokud ne, tak funkce počká a následně opakuje kontrolu. Pokud byl přenos dokončen je nastaven chip select pin na hodnotu 1 a tím je komunikace ukončena. Další funkce využívají tuto funkci pro odesílání dat do digitálního potenciometru.

MPC41HV31_wiper_set_position
Funkce slouží pro přímé nastavení pozice jezdce digitálního potenciometru. Vstupní parametry funkce jsou pin pro výběr digitálního potenciometru a požadovaná pozice potenciometru. Hodnoty příkazu pro nastavení pozice jezdce a požadované polohy jsou uloženy do proměnné sloužící jako buffer pro zápis a jsou následně odeslány pomocí funkce *MPC41HV31_transfer* do potenciometru. Jako výstupní parametr funkce vrací aktuální polohu jezdce.

MPC41HV31_wiper_get_position
Pro vyčítání aktuální polohy jezdce slouží funkce *MPC41HV31_wiper_get_position*. Funkce má jako vstupní parametr pin pro výběr digitálního potenciometru. Do registru pro zápis je uložen byte s hodnotou příkazu pro čtení aktuální polohy jezdce a jeden prázdný byte. Následně je příkaz odeslán do digitálního potenciometru, po dokončení přenosu jsou v bufferu pro čtení dva byty, první byte obsahuje příkaz pro čtení a chybové bity, druhý byte obsahuje hodnotu polohy jezdce. Ta je následně vrácena jako návratová hodnota této funkce.

MPC41HV31_wiper_increment

Funkce slouží pro zvýšení pozice jezdce o jednu pozici. Jako vstupní parametr funkce slouží pin pro výběr digitálního potenciometru. Do bufferu pro zápis je uložen příkaz pro zvýšení pozice jezdce a následně jsou data odeslána do potenciometru. Pokud je jezdec potenciometru již na nejvyšší pozici, je tento příkaz ignorován, jinak je pozice jezdce zvýšena o jedna. Jako návratovou hodnotu funkce vrací aktuální hodnotu polohy jezdce potenciometru.

MPC41HV31_wiper_decrement

Pro snížení pozice jezdce potenciometru o jedna slouží funkce *MPC41HV31_wiper_decrement*. Jako vstupní parametr funkce slouží pin pro výběr digitálního potenciometru. Do bufferu pro zápis je uložen příkaz pro snížení pozice jezdce a následně jsou data odeslána do potenciometru. Pokud je jezdec potenciometru již na pozici nula, je tento příkaz ignorován, jinak je pozice jezdce snížena o jedna. Jako návratovou hodnotu funkce vrací aktuální hodnotu polohy jezdce potenciometru.

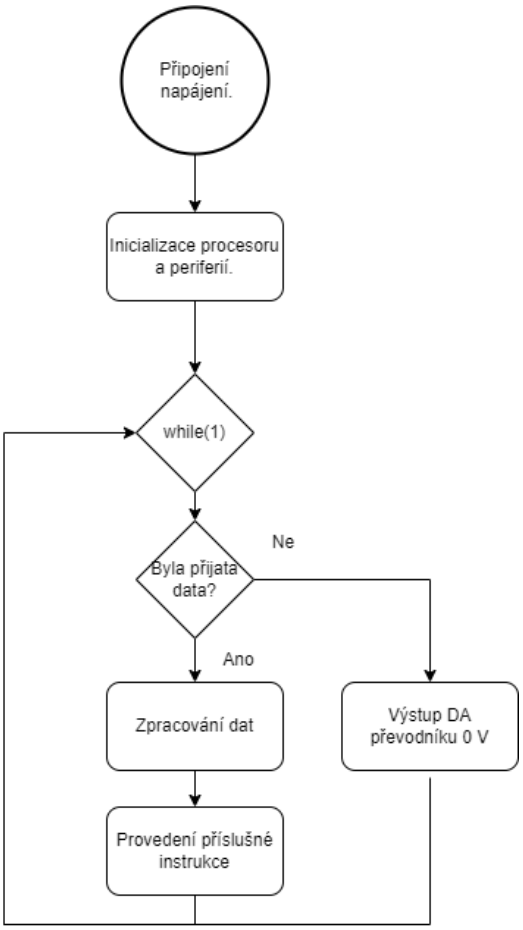
3.2.5 Hlavní smyčka programu

Po připojení napájení platformy je procesor nakonfigurován a jsou inicializovány všechny použité periferie. Následně přechází procesor do hlavní smyčky. Ta je realizována jako stavový diagram se dvěma stavy.

Ve stavu jedna je na výstupu DAC generováno napětí 1 V, aby na vstupu do systému byla hodnota napětí po převodu 0 V. Pokud po sériové lince přijdou data, přechází se do stavu pro zpracování dat. Přijatá data jsou následně rozparsována na jednotlivé hodnoty. Na začátku zprávy je číslo instrukce, které určuje následující počet hodnot. Pomocí čísla instrukce se poté skočí do příslušného stavu a provede se příslušná operace.

Čísla instrukcí byla zvolena následujícím způsobem:

- "1"- Měření.
- "11"- Volba měřených kanálů ADC.
- "21"- Změna hodnoty digitálních potenciometrů zvyšováním a snižováním jeho hodnoty.
- "22"- Změna hodnoty digitálních potenciometrů pomocí hodnoty odporu.
- "31"- Reset měření.



Obr. 3.3: Vývojový diagram hlavní smyčky

Měření

Při přijmutí čísla instrukce této funkce přijme hodnotu generovaného napětí na výstup DA převodníku. Poté je změřena hodnota na požadovaných vstupech AD převodníku, následně jsou tyto naměřené hodnoty odeslány přes sériovou komunikaci zpět do simulace.

Volba měřených kanálů ADC

Dalším vstupním parametrem tohoto stavu je hodnota pro nastavení měřených kanálů, dle které jsou nastaveny kanály AD převodníku, na kterých bude prováděno měření. Tyto nastavené kanály jsou následně odeslány do simulace pro kontrolu správnosti nastavení. Poté je provedena kalibrace pro zvýšení přesnosti měření na zvolených kanálech.

Změna hodnoty digitálních potenciometrů zvyšováním a snižováním jeho hodnoty

Další hodnoty pro tuto část je číslo označující digitální potenciometr a ukazatel zda se má jeho odpor zvýšit či snížit. Pomocí funkcí popsaných v kapitole 3.2.4 je vyčtena aktuální hodnota nastaveného odporu. Následně je provedena změna hodnoty dle požadavku a je provedena kontrola správnosti nastavení. Pokud je hodnota odporu již na maximální nebo minimální hodnotě, tak zvýšení nebo snížení nastaveného odporu neproběhne a je odeslána informace do simulace, že ke změně nedošlo. Po správném provedení nastavení odporu je provedena kalibrace na zvolených kanálech AD převodníku.

Změna hodnoty digitálních potenciometrů pomocí hodnoty odporu.

Pro nastavení hodnoty funkce přijímá číslo digitálního potenciometru a požadovanou hodnotu odporu, která je zkontrolována, zda-li se nachází v platném rozsahu hodnot a to od 0 do 127. Pokud ne, je do simulace odeslána chybová hodnota. Pokud ano, je nastavena pomocí funkce MPC41HV31_wiper_set_position, poté je zkontrolováno úspěšné nastavení této hodnoty. Dle výsledku nastavení je odeslána zpráva do simulace. Poté je provedena kalibrace AD převodníku na zvolených kanálech.

Reset měření

Aby během měření nedocházelo ke zkreslování měřených dat generováním 0 V na výstupu DA převodníku, byl vytvořen stav pro reset měření, který deaktivuje příznak značící měření aktivovaný ve stavu pro měření. Tím nedochází ke zkreslování dat během měření. A také zaručí generování 0 V na výstupu DA převodníku pokud měření neprobíhá.

Vývojový diagram hlavního programu je na obrázku 3.3.

4 Návrh a realizace MATLAB funkcí pro ovládání simulátoru

Tato kapitola popisuje návrh a realizaci ovládacích funkcí pro simulátor. Pomocí těchto funkcí bude možné se připojit k simulátoru, nastavovat hodnoty digitálních potenciometrů, měřené kanály na simulátoru a zobrazení naměřených hodnot.

4.1 MATLAB serialport

Pro obsluhu zařízení využívající sériovou a USB komunikaci poskytuje MATLAB skupinu funkcí využívající objekt *serialport*, který slouží jako náhrada za starší objekt *serial* a jeho přidružené funkce.

Pro navázání komunikace se zařízením je nutné pomocí funkce *serialport* vytvořit objekt pro sériovou komunikaci. Tento objekt obsahuje vlastnosti dané komunikace. Při volání této funkce je nutné zadat název portu, na kterém je zařízení připojeno a rychlost komunikace.

Pro zobrazení všech připojených zařízení k PC slouží příkaz *serialportlist* pro zobrazení pouze dostupných zařízení, tento příkaz je možné zavolat s parametrem *'available'*. Pro komunikaci je možné také nastavit znaky značící konce zpráv pomocí funkce *configureTerminator*. Funkce umožňuje nastavit rozdílné znaky, jak pro zápis, tak pro čtení [11].

Pro odesílání dat do zařízení slouží funkce:

- *write* - umožňuje odeslání dat specifického datového typu do zařízení.
- *writeline* - umožňuje odeslání ASCII řetězce zakončeného nastaveným znakem pro ukončení zprávy do zařízení.

Data ze zařízení je možné vyčítat pomocí funkcí:

- *read* - umožňuje vyčítání definovaného počtu dat specifického datového typu do zařízení.
- *writeline* - umožňuje čtení ASCII řetězce zakončeného nastaveným znakem pro ukončení zprávy ze zařízení.

Po dokončení komunikace je možné zrušit spojení se zařízením pomocí funkce *clear*. [11]

4.2 Realizace ovládacích funkcí

Funkce byly realizovány, aby umožnily ovládání simulátoru podle kapitoly 3.1. Každá funkce komunikující se simulátorem obsahuje kontrolu, zda je aktivní komunikace se simulátorem. Pokud není, tak funkce zobrazí chybové hlášení a ukončí vykonávání.

Pokud dojde k chybě během komunikace funkce také zobrazí příslušnou chybovou zprávu. Pro všechny funkce je možné pomocí příkazu help zobrazit jejich popis a možné vstupní parametry. Pro obsluhu simulátoru byly vytvořeny následující funkce:

ADC_channels
Funkce slouží pro výběr kanálů, na kterých bude prováděno měření. Funkce má jako vstupní parametr pole řetězců s názvy požadovaných kanálů. Vstupní parametr může nabývat těchto hodnot:

- "input"- vstupní napětí generované digitálně analogovým převodníkem
- "pt1"- setrvačný člen prvního řádu
- "pt2"- setrvačný člen druhého řádu
- "err"- rozdílový člen
- "p"- proporcionální člen
- "i"- integrační člen
- "hp"- horní propust setrvačného členu druhého řádu
- "pp"- pásmová propust setrvačného členu druhého řádu

Tyto hodnoty mohou být ve vstupním poli v libovolném pořadí. Vstupní parametr je ve funkci převeden na osmi-bitové číslo, kde každý bit reprezentuje jeden měřící bod. Po převedení celého vstupního řetězce na číslo, je tato hodnota společně s číslem instrukce odeslána do simulátoru. Pokud bude některá z hodnot vstupního parametru neplatná nebo se bude opakovat, funkce zobrazí chybovou zprávu a ukončí vykonávání. Funkce jako návratovou hodnotu vrací počet kanálů, na kterých je prováděno měření. Po nastavení kanálů jsou tyto kanály vytištěny do konzole.

dp_increment_decrement
Pro nastavování hodnoty digitálních potenciometrů slouží funkce dp_increment_decrement. Funkce má jako vstupní parametry číslo digitálního potenciometru a směr změny odporu.

Hodnota čísla digitálního potenciometru může nabývat hodnot 0 pro výběr digitálního potenciometru setrvačného členu prvního řádu a 1 pro rozdílový člen. Směr změny odporu lze volit mezi zvýšením pomocí "increment"a snížením pomocí "decrement". Po provedení této funkce je aktuální odpor potenciometru zobrazen do konzole. Pokud je již hodnota potenciometru na maximální, případně minimální nastavitelné hodnotě, je zobrazeno varování a nedochází ke změně odporu. Funkce má jako návratovou hodnotu číslo chyby.

dp_set_val
Nastavení hodnoty digitálních potenciometrů je umožněno pomocí funkce dp_set_val. Funkce má jako vstupní parametry číslo digitálního potenciometru, a to 0 pro po-

tenciometr pro setrvačný článek prvního řádu a 1 potenciometr rozdílového členu. Dalším vstupním parametrem je požadovaná hodnota odporu pro nastavení, tato hodnota může být v rozsahu 0 až 10 kΩ.

Tato hodnota je ve funkci převedena na hodnotu od 0 do 127, kterou je možné nastavovat potenciometr. Po dokončení nastavení potenciometru je jeho aktuální hodnota zobrazena do konzole. Pokud je hodnota pro nastavení mimo definovaný rozsah, funkce zobrazí chybovou zprávu.

data_send_receive

Pro odesílání a příjímání dat do a ze simulátoru slouží funkce data_send_receive. Funkce má jako vstupní parametry hodnotu napětí signálu v rozsahu ±10 V. Při hodnotě mimo rozsah funkce zobrazí chybovou zprávu. Druhým parametrem funkce je počet kanálů, na kterých je prováděno měření. Tento parametr je omezen na počet dostupných kanálů, tedy hodnoty 0 až 8. Pro odesílání a příjímání zpráv do a ze simulátoru využívá funkce read a write v kapitole 4.1.

data_display

Vykreslení vstupních dat ze simulátoru je možné pomocí funkce data_display. Funkce má jako vstupní parametr naměřená data uložená ve výstupní proměnné simulace a zvolené měřené kanály. Druhý parametr slouží pro sestavení legendy grafu pro vykreslení naměřených průběhů na platformě. Hodnota tohoto parametru by měla být stejná jako vstupní hodnota funkce ADC_channels, aby legenda grafu odpovídala měřeným kanálům.

Dalším grafem, který funkce zobrazuje, je porovnání naměřených dat na platformě a na simulované soustavě. Posledním grafem je porovnání vstupních dat z nástroje MATLAB do simulátoru a zvoleným výstupem platformy. Funkce jako návratové hodnoty vrací naměřená data vrací ve formě příslušných matic hodnot. Další výstupní hodnota obsahuje počty neúspěšných pokusů o komunikaci.

init

Funkce slouží pro navázání komunikace se simulátorem pomocí sériové linky. Funkce má jako vstupní parametry port pro připojení komunikace a rychlost komunikace v Baudech. Funkce jako návratovou hodnotu vrací objekt sériové komunikace. Pro správnou funkčnost ostatních funkcí komunikujících se simulátorem je nutné, aby proměnná do které bude tento objekt uložen, měla název s. Pro zrušení komunikace je možné využít funkci clear.

Dostupné porty k připojení je možné zobrazit pomocí funkce serialportlist, případně pomocí Správce zařízení v záložce Porty(COM a LPT).

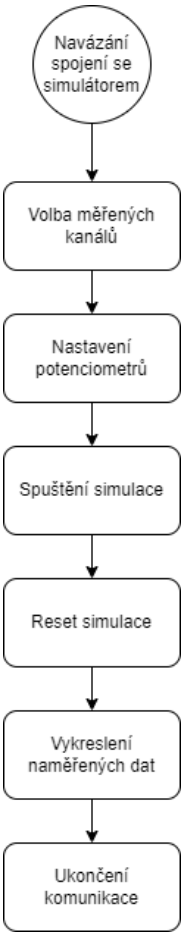
measure_reset

Pro zachování přesnosti měření je nutné měření po dokončení resetovat, a to z důvodu zapojení pro převod napětí 0 až 2 generovaného pomocí DA převodníku na ±10 V, kdy dochází ke generování hodnoty -10 V, pokud je pin sloužící jako výstup DA převodníku neaktivní.

Tato funkce zajistí opětovné generovaní 0 napětí na výstupu a následně počká na vybití setrvačných členů, aby nedocházelo ke zkreslení průběhu následujícího měření.

runme

Pro ukázku správného volání funkcí byl vytvořen skript, ve kterém jsou jednotlivé funkce volány tak, aby bylo provedeno nastavení. Nejprve je zavolána funkce init pomocí které je navázána komunikace. Pro komunikaci byl zvolen port "COM3", ke kterému je dle Správce zařízení simulátor připojen. Rychlost komunikace byla zvolena 250 000 Baudů. Dále byla také zvolena vzorkovací frekvence měření, a to na hodnotu 100 ms.



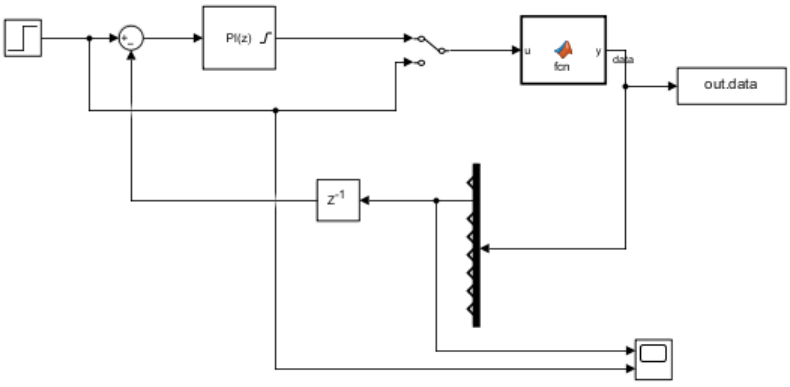
Obr. 4.1: Vývojový diagram skriptu runme

Následně jsou zvoleny požadované kanály, na kterých bude prováděno měření pomocí funkce ADC_channels. Při provádění měření na článku, který obsahuje digitální potenciometr, je možné v tomto místě volat funkce dp_set_val nebo dp_increment_decrement sloužící pro nastavení odporu digitálních potenciometrů.

Po nastavení simulátoru je spuštěna simulace v programu Simulink, která provede měření na simulátoru. V této simulaci je volána funkce data_send_receive pomocí bloku MATLAB Function umožnující volání funkcí vytvořených v programu MATLAB. Po dokončení měření je zavolána funkce measure_reset, která simulátor připraví na další měření. Poté jsou naměřená data vykreslena pomocí funkce data_display. Poté je komunikace se simulátorem přerušena smazáním objektu sériové komunikace pomocí funkce clear. Skript je rozdělen na dílčí sekce, které je možné spouštět samostatně, aby bylo možné provádět více měření a nebylo nutné neustále navazovat komunikaci.

4.3 Simulace v programu Simulink

Simulace se skládá ze dvou zapojení regulačních smyček se zápornou zpětnou vazbou. První smyčka slouží pro simulaci vypočtené soustavy, druhá smyčka slouží k simulaci na platformě simulátoru. K volání funkcí nástroje MATLAB slouží blok MATLAB Function, ve kterém je volána funkce data_send_receive. Aby bylo možné tuto funkci zavolat, je nutné využít funkce nástroje coder extrinsic a evalin. Pro regulátory byly použity bloky *PID Controller*. Zapojení jedné smyčky je na obrázku 4.2



Obr. 4.2: Regulační smyčka pro simulaci na platformě

Pro zajištění real-timeového průběhu simulace slouží blok *Real-Time Sync* z real-timeového toolboxu pro nástroj Simulink. V bloku je nutné zadat vzorkovací periodu, tu je možné nastavovat v skriptu runme jako T_{VZ}, další hodnotou je dovolený počet nepřijatých zpráv. Blok má jako výstup počet nepřijatých zpráv pro každý čas, kdy je simulace prováděna.

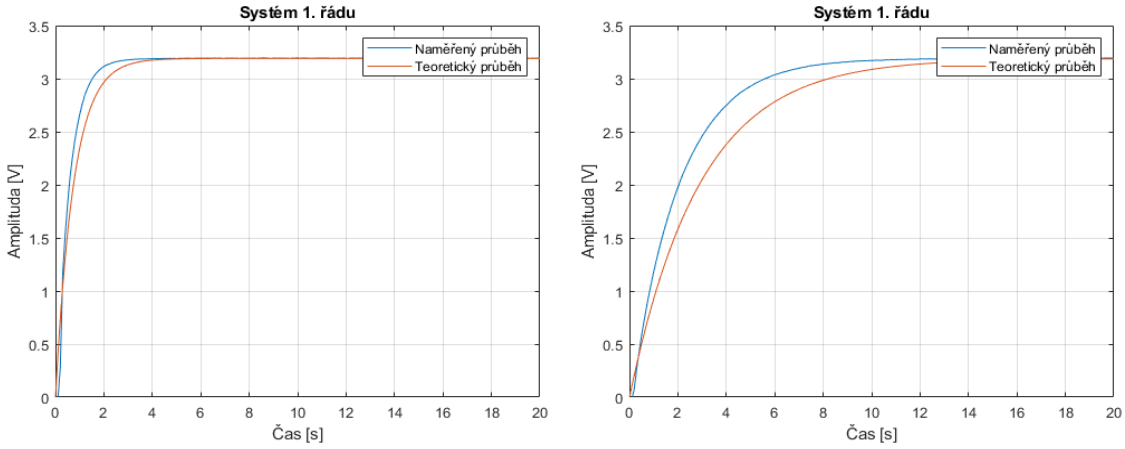
Po získání dat ze simulátoru jsou tato data uložena do matice, která je dále využívána v simulaci. Pro získání příslušného signálu slouží blok *Demux*, naměřená data jsou v matici seřazena do sloupců dle pořadí měřených kanálů a to jak je uvedeno v kapitole 4.2. Pořadí kanálů se mění dle vybraných měřených kanálů. Například při měření vstupního kanálu a setrvačného členu druhého řádu je pro získání dat druhého kanálu potřeba připojit druhý výstup bloku *Demux*.

Ve druhém případě pokud bude prováděno měření na vstupním kanálu, na setrvačném členu prvního řádu a na setrvačném členu druhého řádu tak pro získání naměřených hodnot na setrvačném členu druhého řádu je nutné připojit výstup na třetí výstup bloku *Demux*. Simulace umožňuje přepínání mezi měřením regulační smyčky a měření přechodové charakteristiky soustavy pomocí bloků *Manual Switch*.

Výstupními hodnotami simulace je matice naměřených dat na simulátoru, porovnání průběhů na simulované a na reálné soustavě a průběhy vstupních dat a odezvy zvoleného kanálu. Tato data jsou uložena v proměnné *out*, která je po ukončení měření dostupná ve Workspace v MATLABu.

4.4 Ověření funkčnosti firmware a funkcí v prostředí MATLAB

Funkčnost digitální části platformy byla otestována pomocí skriptu runme.m popsaného v kapitole 4.2. Naměřené průběhy byly následně porovnány s teoretickými průběhy vypočtenými pomocí nástroje MATLAB Simulink.

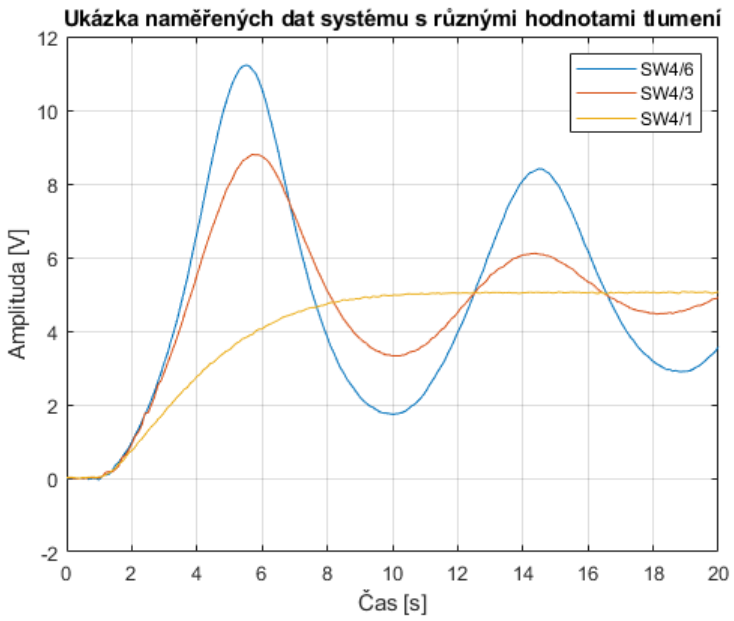


(a) PT1 člen SW5-3 (b) PT1 člen SW5-5

Obr. 4.3: Průběhy naměřené pomocí digitální části platformy.

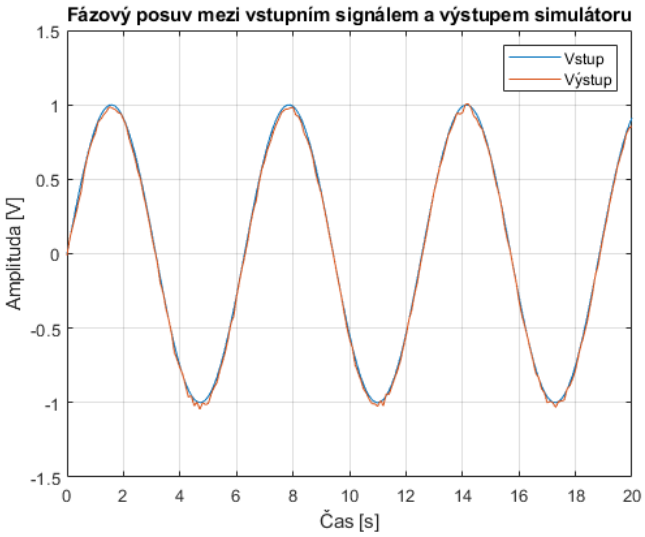
V grafu 4.3 lze vidět naměřené průběhy pro setrvačný článek prvního řádu pro různé nastavení odporů pomocí DIP přepínačů. Byly naměřeny průběhy pro odpory $\Delta R = 160\ k\Omega$ a $\Delta R = 620\ k\Omega$.

Dalším členem, na kterém byla funkčnost testována, byl setrvačný člen druhého řádu. Zde byly naměřeny průběhy pro různé hodnoty tlumení, které je možné měnit pomocí přepínače DIP SW4. Tyto průběhy byly naměřeny pro hodnoty odporů $\Delta R_5 = 1,2\ M\Omega$, $\Delta R_5 = 56\ k\Omega$ a $\Delta R_5 = 2,7\ k\Omega$. Naměřené průběhy jsou v grafu 4.4.



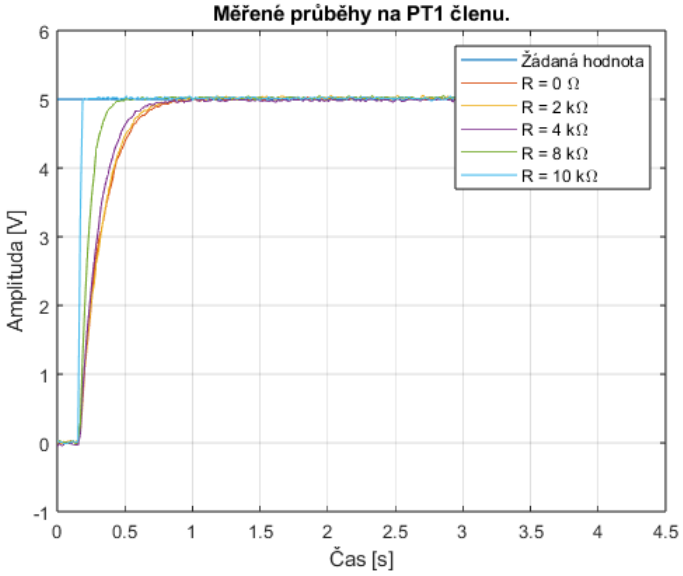
Obr. 4.4: Průběhy naměřené pro různé hodnoty tlumení

Následující průběh v grafu 4.5, byl naměřen pro zhodnocení fázového posuvu mezi vstupem a výstupem simulátoru. Tento průběh je možné naměřit na proporcionálním členu prvního řádu s nastavených zesílením na hodnotu jedna. Tomu odpovídá nastavení DIP přepínače SW2 na pozici 1, tedy $\Delta R = 22\ k\Omega$. Jako vstupní signál byl použit sinusový průběh s amplitudou 5 V a frekvencí 1 $rad.s^{-1}$.



Obr. 4.5: Fázový posuv mezi vstupem a výstupem simulátoru

V grafu 4.6 je možné vidět naměřené průběhy pro setrvačný článek prvního řádu pro různé hodnoty odporů nastavené na digitálním potenciometru. Odpor připojený paralelně k odporu digitálního potenciometru byl nastaven pomocí DIP přepínače SW5 na hodnotu $R = 56\ k\Omega$. Pomocí vytvořené funkce byl měněn odpor digitálního potenciometru. Byly naměřeny průběhy pro hodnoty nastaveného odporu 0 až 10 kΩ s krokem 2 kΩ.



Obr. 4.6: Měřené průběhy na setrvačném členu prvního řádu.

5 Návrh a implementace demonstrační úlohy

Kapitola se zabývá návrhem a implementací jednoduché řídící úlohy pro demonstraci HIL simulace. Kapitola popisuje zvolenou regulovanou soustavu a její identifikaci, návrh regulátoru, porovnání regulace na reálném hardware a simulované soustavě. V závěru kapitoly je uvedeno zhodnocení použitelnosti a omezení implementovaného řešení.

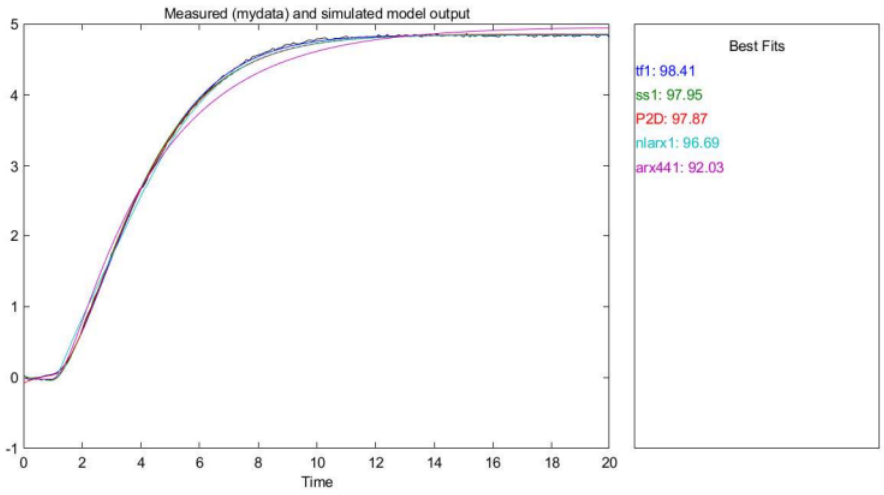
5.1 Zvolená soustava

Pro demonstrační úlohu byl zvolen setrvačný článek druhého řádu. Pomocí DIP přepínačů byly nastaveny hodnoty odporů $\Delta R_5 = 2.7\ k\Omega$ nastaveného pomocí SW4 na pozici jedna. $\Delta R_4 = 56\ k\Omega$ pomocí hodnot DIP přepínačů SW3 a SW6 na hodnotu na pozici tři. Nastavení těchto přepínačů musí být stejné, aby přenosová rovnice odpovídala tvaru 2.2.

Na desce platformy je zapojení provedeno přivedením vstupního signálu z pinu J5 na vstupní pin setrvačného členu druhého řádu J12. Jako výstupní pin tohoto členu slouží J17, na tomto pinu bude výstupní signál invertovaný oproti vstupnímu. Pro získání správné polarity signálu je nutné tento výstupní signál připojit na vstup proporcionálního členu s nastaveným zesílením jedna, přepínač SW2 na pozici jedna.

Po nastavení a zapojení požadovaného členu je možné změřit přechodovou charakteristiku pomocí programu uvedeného v kapitole 4.2. Ta bude použita pro následnou identifikaci přenosu soustavy. Identifikaci je možné provést pomocí *MATLAB Identification toolboxu*. Naměřený průběh slouží jako vstupní data pro identifikaci. Následně je možné zvolit metodu pro odhad modelu. Pro demonstraci možností byly zvoleny metody *State space*, *Transfer function*, *Polynomical model* a *ARX model*.

Dle kritéria Best Fit bylo dosaženo nejlepších výsledků pomocí metody *Transfer function* s přesností výsledku 98,41 %.



Obr. 5.1: Výsledek identifikace zvoleného systému

Výsledný identifikovaný přenos zvoleného systému je

$$F_s(p) = \frac{0,3207}{p^2 + 1,0872p + 0,3341} \tag{5.1}$$

5.2 Návrh regulátoru

Pro soustavu identifikovanou v kapitole 5.1 byl vytvořen regulátor metodou určení parametrů regulátoru pomocí integrálního kritéria ITAE, jako další metoda byl využit nástroj PID tune, který poskytuje blok PID Control dostupný v knihovně bloků nástroje Simulink.

Pro návrh regulátoru byla vytvořena simulace navrh_regulator v nástroji Simulink, ve které je možné regulátor ladit na identifikovaném modelu soustavy. Simulace také obsahuje blok s regulátorem navrženým pomocí ITAE, pro testovaní regulátoru navrženého touto metodou.

Pro následné testování regulátoru na platformě je nutné po doladění přesunout konstanty regulátoru do simulace hw_sim. Pro zhodnocení vlastností navržených regulátorů slouží přímo v simulaci výpočet integrálního kritéria ITAE, jehož hodnota je zobrazena na displeji v simulaci.

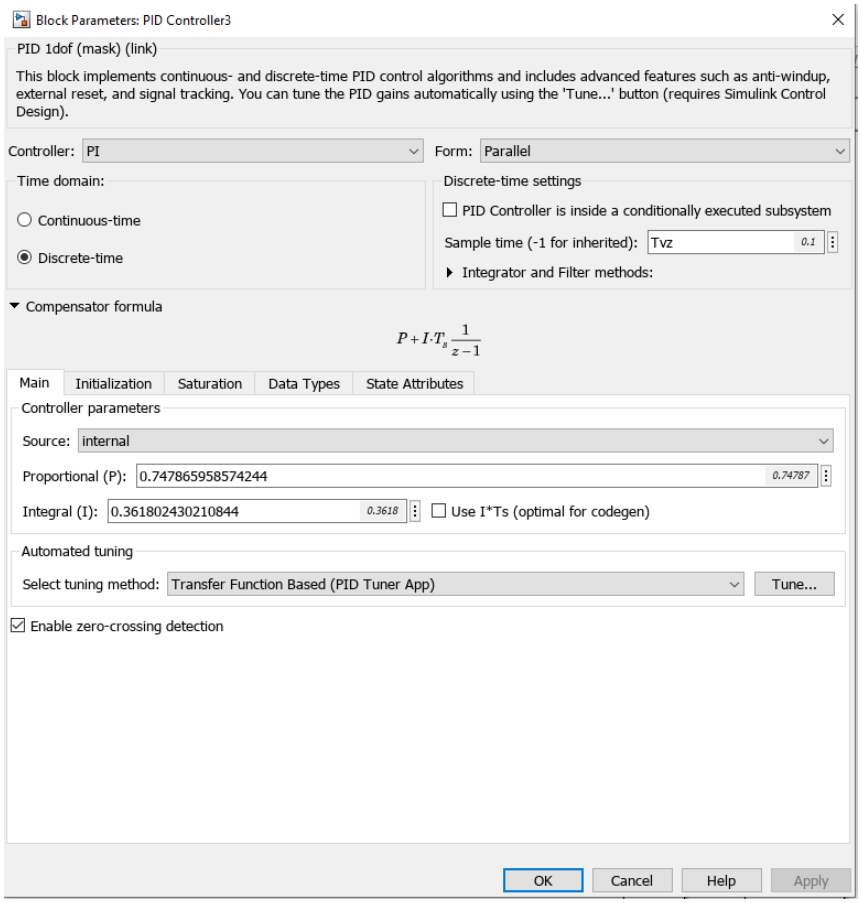
5.2.1 PID Controller

Pro návrh regulátoru soustavy je možné využít nástroje PID tune v bloku PID Controller. V tomto bloku je možné volit typ regulátoru, zda-li se bude jednat o diskrétní nebo spojitý regulátor a jeho formu. Pro diskrétní regulátor je možné volit jeho vzorkovací periodu.

Pro tuto demonstrační úlohu byl zvolen PI regulátor v paralelním tvaru. Tento typ regulátoru má následující přenos:

$$F_R(z) = P + I \cdot T_{vz} \frac{1}{z-1}. \tag{5.2}$$

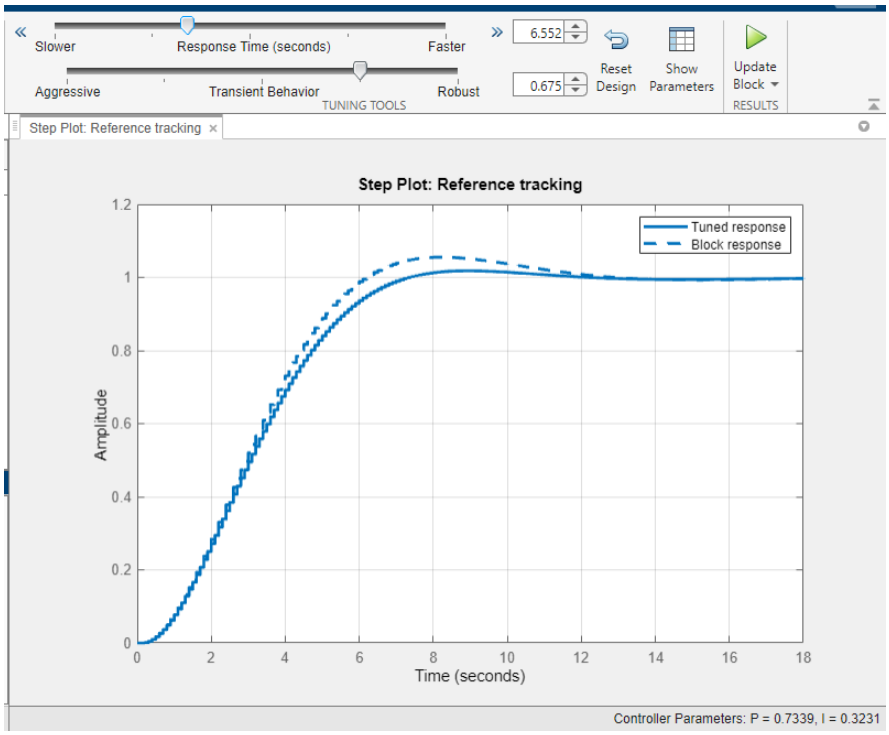
Regulátor byl nastaven jako diskrétní s periodou vzorkování určenou ve skriptu runme tedy $T_{vz} = 100\ ms$. Dále je možné volit hodnoty saturace regulátoru, pro tuto úlohu byla zvolena maximální možná hodnota napětí, kterou je možné generovat na vstupu do platformy tedy $\pm 10\ V$.



Obr. 5.2: Ukázka nastavovacího okna regulátoru

Pro regulátory s I složkou je výhodné využít anti-windup metodu pro snížení regulační odchylky v případě saturace akčního členu. Metodu je možné zvolit jako *back-calculation* nebo *clamping*. Pro tento regulátor byla zvolena metoda *clamping*.

Po dokončení nastavení požadovaných vlastností je možné tyto změny aplikovat a následně přejít k ladění regulátoru pomocí tlačítka Tune. Ladění regulátoru je možné pomocí dvou posuvníků, kdy první posuvník mění rychlost odezvy regulátoru. Druhý upravuje chování přechodového jevu.



Obr. 5.3: Ukázka ladicího okna regulátoru

Změny chování regulátoru je možné sledovat v grafu přechodového děje. Po nastavení parametrů na vyhovující hodnotu je možné hodnoty pro jednotlivé členy regulátoru nahrát do regulátoru pomocí tlačítka *Update Block*. Následně je možné testovat regulaci spuštěním simulace. Ladění regulátoru je nutné uskutečnit v simulaci popsané v úvodů této kapitoly, jelikož není možné reálnou soustavu linearizovat z důvodu vysoké vzorkovací frekvence, kterou blok využívá při linearizaci.

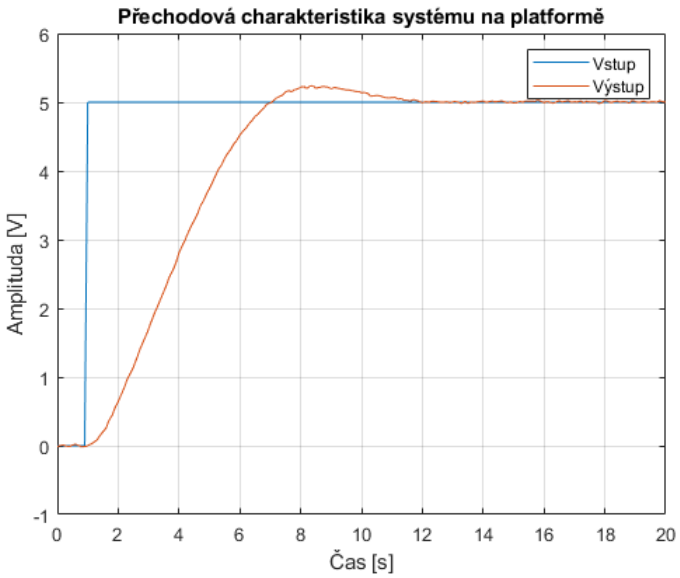
Výsledná odezva soustavy s PI regulátorem vytvořeným touto metodou je v grafu 5.7.

Výsledný přenos regulátoru je

$$F_R(z) = 0,7479 + 0,3618 \cdot T_{vz} \frac{1}{z-1}. \tag{5.3}$$

5.2.2 ITAE kritérium

Další metodou je výpočet parametrů regulátoru pomocí integrálního kritéria ITAE, pro který byl vytvořen skript ITAE_navrh_regulatoru. Pro výpočet je možné použít identifikovanou soustavu, případně je možné pomocí hodnot nastavených odporů vypočítat přenos soustavy pomocí vzorce 2.2. Funkce následně pro definovaný přenos vypočítá parametry pro regulátor pomocí funkce *fminsearch*.



Obr. 5.4: Odezvy soustavy s připojeným PI regulátorem navrženým pomocí bloku PI Control

Pro tu bylo nutné realizovat funkci pro výpočet ITAE kritéria. Tato funkce pro zadanou soustavu a navrhovaný regulátor vypočte přenos řízení, na kterém je následně vypočtena odezva na jednotkový skok. Data z výpočtu jednotkového skoku jsou poté použita pro výpočet kritéria ITAE. Výsledek výpočtu závisí na počátečních hodnotách konstant regulátoru definovaných ve skriptu.

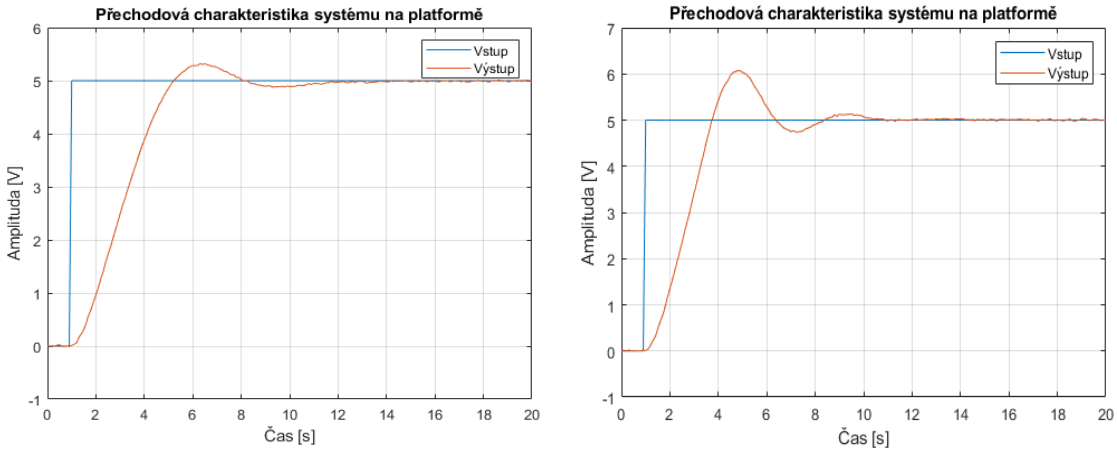
Po dokončení výpočtu a nalezení parametrů regulátoru, s nejmenší hodnotou kritéria ITAE, jsou tyto parametry zobrazeny. Ty je následně možné použít v simulaci navrh_regulátoru pro testování regulátoru na simulované soustavě, případně v simulaci na reálném hardwaru. Pro použití parametrů regulátoru je možné využít blok PID Control, případně je možné sestavit diskrétní PI regulátor ze základních bloků z knihovny Simulinku.

Pro tuto metodu byly provedeny výpočty pro více počátečních hodnot zesílení regulátoru, odezvy regulovaných soustav s navrženými regulátory jsou zobrazeny v grafu 5.6. Tyto regulátory byly navrženy pro počáteční podmínky $K_p = 1$, $K_i = 0.5$ pro první regulátor a $K_p = 0.1$, $K_i = 0.1$ pro druhý regulátor. Výsledný přenos regulátoru pro počáteční hodnoty zesílení $K_p = 1$ a $K_i = 0.5$ je

$$F_R(z) = 1,2631 + 0,4711 \cdot T_{vz} \frac{1}{z-1}. \tag{5.4}$$

Výsledný přenos regulátoru pro počáteční hodnoty zesílení $K_p = 0.1$ a $K_i = 0.1$ je

$$F_R(z) = 2,8698 + 0,758 \cdot T_{vz} \frac{1}{z-1}. \tag{5.5}$$



(a) Odezva soustavy s prvním regulátorem (b) Odezva soustavy s druhým regulátorem

Obr. 5.5: Odezvy soustavy s připojenými PI regulátory navrženými pomocí ITAE kritéria

Pro různé počáteční podmínky jsou rozdílné výsledné hodnoty zesílení regulátoru. Tento rozdíl je pravděpodobně způsoben nalezením lokálního minima během výpočtu parametrů regulátoru.

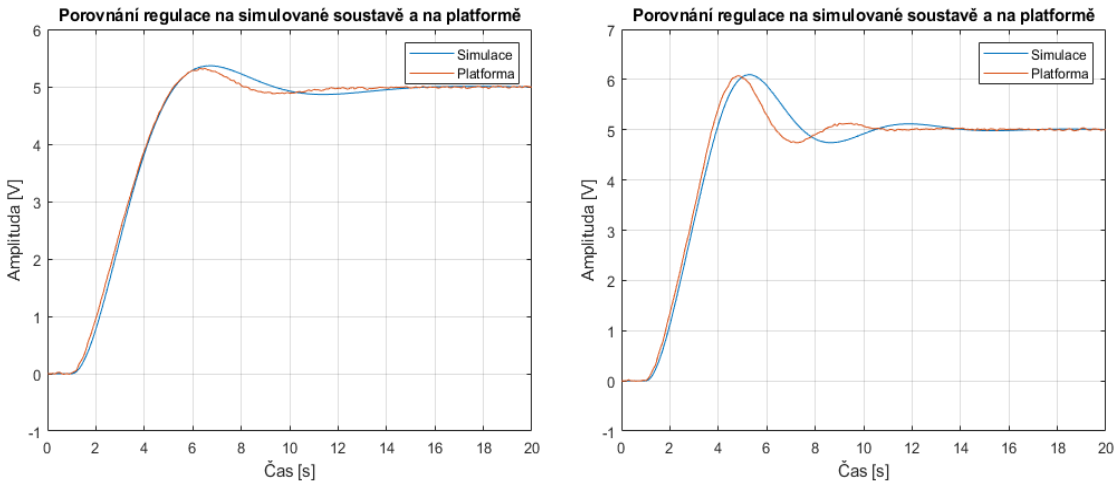
5.3 Porovnání regulace na simulátoru a na simulované soustavě

Po dokončení návrhu regulátorů jsou navržené regulátory vloženy do simulace hw_sim, zde je možné provádět měření jak na simulovaném modelu soustavy, tak na reálné platformě jak bylo popsáno v kapitole 4.2.

V simulaci jsou umístěny bloky regulátorů, a to jak pro regulátory navržené metodou ITAE, tak pro regulátor navržený pomocí nástroje PID Controller. Pro regulátory navržené metodou ITAE je nutné mít konstanty uloženy v MATLAB workspace, aby bylo možné simulaci spustit. Pro měření s požadovaným regulátorem je nutné jej umístit do obou větví simulace a následně je možné simulaci spustit. Kvalita regulace je zhodnocena pomocí integrálního kritéria ITAE, to je počítáno do času 20s po spuštění regulace. Doba ustálení, velikost překmitu, hodnota kritéria ITAE a doba dosažení 95 % žádané hodnoty jsou zobrazeny v tabulce 5.1

Tab. 5.1: Hodnocení kvality regulace navržených regulátorů

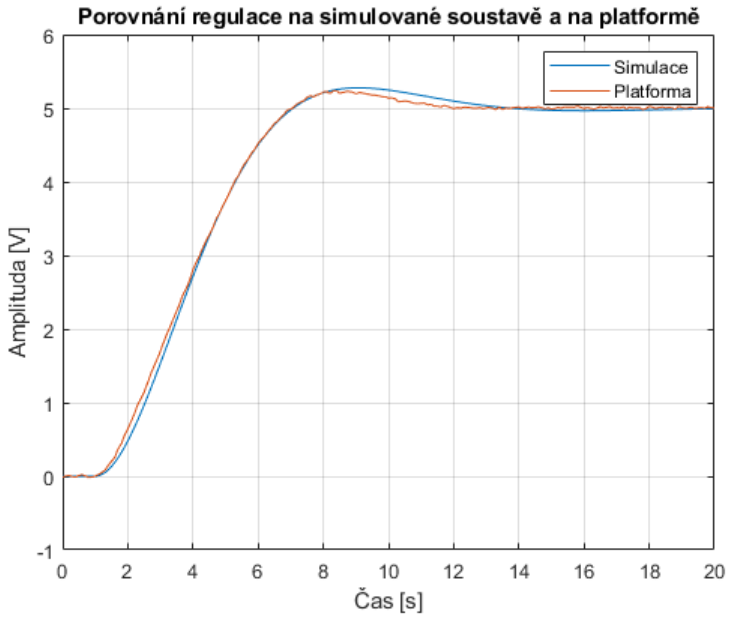
	ITAE	Překmit [%]	t_r 95% [s]	Doba ustálení [s]
PI ITAE 1	49,96	5,27	4	18.4
PI ITAE 2	35,78	15.41	5,5	16.1
PI Control	56,8	4,29	5,6	16,1



(a) Porovnání regulace pro první regulátor (b) Porovnání regulace pro druhý regulátor

Obr. 5.6: Porovnání regulace na simulované soustavě a na platformě regulátory navrženými metodou ITAE

Z grafů 5.6 a 5.7 a z tabulky 5.1 je patrné, že regulátory navržené pomocí nástroje PID Tune a metody ITAE pro počáteční hodnoty zesílení $K_p = 1$ a $K_i = 0,5$ mají dle předpokladu velmi podobnou odezvu na reálné platformě jako na modelované soustavě v Simulinku. Regulátor navržený metodou ITAE pro počáteční hodnoty zesílení $K_p = 0,1$ a $K_i = 0,1$ má výrazně vyšší překmit oproti ostatním regulátorům a také jeho odezva na platformě je rozdílná oproti odezvě v simulaci.



Obr. 5.7: Porovnání regulace na simulované soustavě a na platformě s navrženým regulátorem blokem PI Control

6 Zhodnocení použitelnosti a omezení implementovaného řešení

Tato kapitola se zabývá zhodnocením použitelnosti a omezení implementovaného řešení na platformě.

Testování implementovaného řešení bylo testováno na dvou rozdílných PC. Prvním testovacím PC byl notebook Dell Vostro 5568. Druhým PC na kterém bylo testování prováděno, byl stolní počítač. Parametry testovacích počítačů jsou uvedeny v tabulce 6.1. Na obou testovacích počítačích je nainstalován operační systém Windows 10 a MATLAB/Simulink verze 2022b a všechny potřebné toolboxy v této verzi.

Tab. 6.1: Parametry testovaných PC

	Stolní PC	Dell Vostro 5568
Procesor	AMD Ryzen 3600	Intel Core i7-7500U
Počet fyzických jader	6	2
Základní rychlost [GHz]	3,6	2,7
Paměť RAM [GB]	16	8

6.1 Omezení z hlediska periody vzorkování

Použitelnost měření na platformě byla testována pro různé hodnoty periody vzorkování. Pro zajištění Real-Timového měření je použit blok Real-Time Synchronization z toolboxu Simulink Desktop Real-Time. V tomto bloku byl nastaven maximální povolený počet chyb v komunikaci na hodnotu 10. Při překročení této hodnoty je simulace ukončena a je zobrazena chybová zpráva.

Tab. 6.2: Hodnoty chyb komunikace při změnách periody vzorkování

Tvz [ms]	Počet chyb stolní	Počet chyb Dell	Počet naměřených dat
100	0	0	201
25	1	3	801
20	4,2	8,2	1001
18	6	13	1112
17	11,2	26	1177
16	46	113	1251

Pro každou z vybraných period vzorkování bylo provedeno 10 měření. Následně byla vypočtena průměrná hodnota chyb v komunikaci pro tato měření. Výsledky měření na platformě jsou uvedeny v tabulce 6.2. Všechna měření byla prováděna pro délku simulace 20 s a byla spouštěna ze skriptu runme.

Jak je patrné z tabulky 6.2, tak na stolním PC s vyšším výkonem nedochází k takovému počtu chyb během komunikace. Pro nižší periody vzorkování než je 16 ms nebylo možné spolehlivě provést testovací měření. Měření na těchto periodách bylo většinou předčasně ukončeno z důvodu limitu chyb v komunikaci.

Nejmenší hodnota periody vzorkování, kdy je možné provádět měření na platformě tak, aby bylo možné jej opakovat je tedy 16 ms. Nejnižší hodnota periody vzorkování kdy nedochází téměř k žádným ztrátám v komunikaci je pro stolní PC 26 ms a pro notebook Dell Vostro je 37 ms.

6.2 Omezení z hlediska rychlosti komunikace

Při změnách hodnot rychlosti komunikace pomocí nastavovací funkce init nedochází ke změnám v rychlosti odesílání dat do a zpět z platformy. Toto je způsobeno použitou komunikací přes rozhraní USB. Nutnost zadávat rychlost komunikace je způsobena požadavkem funkce serialport na tuto vstupní hodnotu.

Na rychlost komunikace nemá také vliv použití funkcí evalin, která je používána pro volání funkce *data_send_receive*.

6.3 Omezení z hlediska vykonávání kódu na platformě

Během provádění měření na platformě bylo zjištěno, že cyklus zpracování jedné instrukce pro měření odeslané v prováděné simulaci trvá přibližně 14 ms. Z [9] bylo zjištěno, že potřebný čas na převod jedné hodnoty naměřené na vstupu AD převodníku je přibližně 20 μs. Pro zvýšení přesnosti naměřených hodnot platforma naměří 64 hodnot. Z toho vyplývá, že pro naměření jedné hodnoty je potřeba přibližně 1,3 ms. Pro měření na více kanálech najednou se bude tento čas navyšovat podle počtu kanálů.

Zbylý čas je tedy věnován obsluze periferií, zpracování příchozích dat a odesílání naměřených dat, a to jak v části vytvořené v MATLABu, tak v části na platformě.

6.4 Zhodnocení použitelnosti

Platforma je schopna provádět měření na všech systémech popsaných v kapitole 2.1. A to včetně připojených digitálních potenciometrů, avšak nastavená perioda vzorkování musí být vyšší něž 16 ms. Pro hodnoty nižší než 16 ms již systém není schopen spolehlivě měřit přechodové charakteristiky. Systém je schopen generovat vstupní napětí na definovaném rozsahu ±10 V. Dále je schopen měřit napětí v rozsahu ± 13,5 V.

Z důvodu realizovaného zapojení obvodu pro převod generovaného napětí je výsledná hodnota generovaného napětí invertována. Tento problém je možné odstranit zapojením členu P s nastaveným zesílením na hodnotu jedna a měřením na jeho výstupu jak je uvedeno v kapitole 5.1.

Nevýhodou je nemožnost porovnání soustavy vypočtené pomocí vzorců uvedených v kapitole 2.1 pro setrvačný člen druhého řádu, kde během výpočtu vznikají rozdíly, které by bylo vhodné dále analyzovat.

Z průběhů naměřených na platformě je možné identifikovat model této soustavy, pro který je možné následně navrhnout regulátor, například metodami popsanými v kapitole 5.2. S těmito regulátory je poté možné měřit odezvy systémů.

Pomocí vytvořených funkcí v MATLABu je také možné zobrazovat průběhy naměřených hodnot, případně naměřená data dále zpracovávat.

Závěr

Cílem této práce bylo vytvořit software realizující výměnu dat mezi fyzickým simulátorem dynamických systémů a prostředím MATLAB. Dále bylo cílem navrhnout a implementovat jednoduchou úlohu pro řízení vybraného systému (HIL simulaci) a zhodnotit použitelnost a omezení celého řešení.

V rámci této práce byla nejprve dodaná DPS osazena a oživena. Problémy s dodanou DPS způsobené během výroby jsou popsány v kapitole 2.4. Dále byly vydefinovány možnosti ovládání a sběru dat na digitální části desky.

V této práci byl také implementován firmware pro mikrokontroler umožňující komunikaci a výměnu dat s připojeným PC a také ovládání vydefinovaných digitálních prvků simulátoru.

Simulátor je možné ovládat pomocí funkcí vytvořených v kapitole 4.2. Dále byla realizována jednoduchá řídící úloha pro demonstraci HIL simulace. Pro tuto úlohu byl vytvořen simulinkový model umožňující měření jak na reálném simulátoru, tak na simulované soustavě. Výsledky regulace s využitím realizované platformy a porovnání s regulací na simulované soustavě jsou v grafech 5.6 a 5.7 a dále v tabulce 5.1, ze kterých je patrné, že regulátory mají dle předpokladu velmi podobnou odezvu na simulované soustavě jako na realizované platformě.

Vytvořený software je možné použít pro měření na všech systémech simulátoru, a to včetně zapojených digitálních potenciometrů. Omezením systému je nejnižší měřitelná hodnota periody vzorkování, která byla stanovena na 16 ms. Pro hodnoty pod touto hranicí již není možné spolehlivě provádět měření. Systém je také schopen generovat napětí na vstupu soustavy v celém rozsahu ± 10 V a měřit hodnoty výstupu soustavy v celém rozsahu ± 13.5 V.

Literatura

[1] BALÁTĚ, J. *Automatizace řízení* Praha: BEN-technická literatura,2003, ISBN 80-730-0020-2

[2] LJUNG, Lennart. *System Identification: Theory for the user. 2nd. Edition* Upper Saddle River, New Jersey 07458: Prentice Hall PTR, 1999. ISBN 0-13-656695-2

[3] Jelínek, P. *Simulace Processor In the Loop a Hardware In the Loop.* AUTOMA.2007, 5, [cit. 12. 5. 2023].
Dostupné z URL:<`https://automa.cz/cz/casopis-clanky/simulace-processor-in-the-loop-a-hardware-in-the-loop-2007_05_34311_2055/`>.

[4] *Basics of Hardware-in-the-Loop simulation [Online]*The MathWorks, Inc. , [cit. 13. 5. 2023] Dostupné z URL:
<`https://www.mathworks.com/help/simscape/ug/what-is-hardware-in-the-loop-simulation.html`>.

[5] *SIL and PIL Simulations [Online]*The MathWorks, Inc. , [cit. 13. 5. 2023] Dostupné z URL:
<`https://www.mathworks.com/help/ecoder/ug/about-sil-and-pil-simulations.html`>.

[6] LJUNG, Lennart. *System Identification Toolbox: User's Guide* [online]. R2022. © 1994-2022 The MathWorks [cit. 12. 5. 2023]. Dostupné z URL:
<`https://www.mathworks.com/help/ident/`>.

[7] PACAL, Stanislav. *Řízení reálného systému pomocí PLC s Využitím automaticky generovaného kódu.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 88 s. Diplomová práce. Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.

[8] *MCP41HVX1 Datasheet [Online].* Microchip Technology Inc. 2013 , poslední aktualizace červen 2022 [cit. 22. 12. 2022] Dostupné z URL:
<`https://ww1.microchip.com/downloads/aemDocuments/documents/MSLD/ProductDocuments/DataSheets/MCP41HVX1-Data-Sheet-20005207.pdf`>.

[9] *SAM D21/DA1 Family Datasheet [Online].* Microchip Technology Inc. 2021 , [cit. 1. 1. 2023] Dostupné z URL:
<`https://ww1.microchip.com/downloads/aemDocuments/`

documents/MCU32/ProductDocuments/DataSheets/
SAM-D21-DA1-Family-Data-Sheet-DS40001882H.pdf>.

[10] *ASF4 API Reference Manual [Online].* Microchip Technology Inc. 2018 , [cit. 1. 1. 2023] Dostupné z URL:
<https://ww1.microchip.com/downloads/en/DeviceDoc/50002633B.
pdf>.

[11] *Serial and USB Communication [Online].* The MathWorks, Inc. , [cit. 1. 1. 2023] Dostupné z URL:
<https://www.mathworks.com/help/matlab/serial-port-devices.
html>.

[12] *Atmel START User's Guide [Online].* Microchip Technology Inc. 2018, [cit. 3. 1. 2023] Dostupné z URL:<https://onlinedocs.microchip.com/
pr/GUID-4E095027-601A-4343-844F-2034603B4C9C-en-US-4/index.html?
GUID-31CAFDCB-DD38-462B-893D-B5A7DC63B24A>.

[13] *Simulation and Prediction in the App [Online]*The MathWorks, Inc. , [cit. 16. 5. 2023] Dostupné z URL:
<https://www.mathworks.com/help/ident/ug/
simulation-and-prediction-in-the-app.html>.

A Obsah elektronické přílohy

- `/` ..... kořenový adresář přiloženého archivu
  - `DP` ..... Projekt Atmel studio s kodem pro platformu
  - `Matlab funkce` ..... Použitá verze softwaru je MATLAB 2022b
  - `xkorca03_thesis.pdf` ..... Text práce