UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

IMPLEMENTACE DISKRÉTNÍHO LQG REGULÁTORU S VYUŽITÍM SCADA SYSTÉMU PROMOTIC

Bc. Daniel Opršal

Diplomová práce
2021

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Daniel Opršal**
Osobní číslo:	**I19302**
Studijní program:	**N0714A150005 Automatické řízení**
Studijní obor:	**Automatické řízení**
Téma práce:	**Implementace diskrétního LQQ regulátoru s využitím SCADA systému PROMOTIC**
Zadávající katedra:	**Katedra řízení procesů**

Zásady pro vypracování

Cíl: Vytvořte program pro PC v prostředí PROMOTIC, který realizuje LQ řízení lineárního SISO systému se známým modelem připojeným pomocí Arduino Due

Teoretická část:

a) návrh LQ regulátoru s asymptotickým sledováním žádané

b) Kalmánův estimátor stavu

c) možnosti SCADA systému PROMOTIC, komunikace přes sériovou linku, návrh komunikačního protokolu

Praktická část:

a) vytvoření programu pro Arduino Due pro měření a generování napěťových signálů se zasíláním hodnot přes USB sběrnici (virtuální sériový port na PC)

b) ověření modelu a návrhu LQG regulátoru v prostředí MATLAB

d) implementace LQG regulátoru v systému PROMOTIC

Rozsah pracovní zprávy: **50 – 60 stran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

BALÁTĚ, Jaroslav, 2004. *Automatické řízení*. 2., přeprac. vyd. Praha: BEN – technická literatura. ISBN 80-730-0148-9.
OGATA, Katsuhiko. *Discrete-time control systems*. 2nd ed. Englewood Cliffs, N.J.: Prentice Hall, c1995, xi, 745 p. ISBN 01-303-4281-5.
Obsah dokumentace PROMOTIC. MICROSYS, spol. s r.o. *SCADA/HMI systém PROMOTIC* [online]. 2020 [cit. 2020-09-21]. Dostupné z: http://www.promotic.eu/cz/pmdoc/PmDocDefault.htm
Arduino Home [online], c2020. [cit. 2020-09-21]. Dostupné z: https://www.arduino.cc/

Vedoucí diplomové práce: **doc. Ing. František Dušek, CSc.**
Katedra řízení procesů

Datum zadání diplomové práce: **6. listopadu 2020**
Termín odevzdání diplomové práce: **21. května 2021**

L.S.

Ing. Zdeněk Němec, Ph.D.
děkan

Ing. Daniel Honc, Ph.D.
vedoucí katedry

V Pardubicích dne 16. listopadu 2020

Prohlášení

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne

Bc. Daniel Opršal

Poděkování

Rád bych poděkoval panu doc. Ing. Františkovi Duškovi, CSc. za odborné vedení, jeho cenné rady, trpělivost a ochotu, kterou mi v průběhu zpracování diplomové práce věnoval. Dále bych rád poděkoval své rodině a přátelům, kteří mě podporovali během studia.

V Pardubicích dne

Bc. Daniel Opršal

ANOTACE

Tato diplomová práce se zabývá implementací diskrétního LQ regulátoru s asymptotickým sledováním žádané hodnoty implementovaného do SCADA systému PROMOTIC. V teoretické části je popsán návrh diskrétního Luenbergerova estimátoru a LQ regulátoru s asymptotickým sledováním žádané hodnoty. V praktické části je popsán program pro Arduino Due, který komunikuje s programem PROMOTIC prostřednictvím uživatelsky konfigurovatelného ASCII protokolu.

KLÍČOVÁ SLOVA

Stavová regulace, LQ, Arduino Due, PROMOTIC.

TITLE

IMPLEMENTATION OF A DISCRETE LQG REGULATOR USING SCADA SYSTEM PROMOTIC

ANNOTATION

This diploma thesis deals with the implementation of a discrete LQ controller with an asymptotic monitoring of the required value implemented in the SCADA system PROMOTIC. The theoretical part focuses mainly on the design of a discrete Luenberger estimator and LQ controller with asymptotic setpoint monitoring. The practical part describes the program for Arduino Due, which communicates with the PROMOTIC program via a user configurable ASCII protocol.

KEYWORDS

State regulation, LQ, Arduino Due, PROMOTIC

OBSAH

SEZNAM ZKRATEK A ZNAČEK

ASCII	Americký standardní kód pro výměnu informací
A/D	analogově digitální převodník
COM	sériové rozhraní
DAC	digitálně analogový převodník
FLASH	elektricky programovatelná paměť typu ROM-RAM
LQ	lineárně kvadratický
LTI	lineární časově invariantní (neměnné)
MIMO	více vstupů, více výstupů
PC	osobní počítač
SCADA	systém pro dohled, řízení a sběr dat
SISO	jeden vstup, jeden výstup
SPI	sériové periferní rozhraní
SRAM	statická paměť pro čtení nebo zápis
USB	univerzální sériová sběrnice
UART	univerzální sériový asynchronní přijímač / vysílač
TWI	dvoudrátové rozhraní

SEZNAM SYMBOLŮ PROMĚNNÝCH VELIČIN A FUNKCÍ

λ	vlastní čísla
$\boldsymbol{A}$	matice systému
$\boldsymbol{A_E}$	matice estimátoru
$\boldsymbol{B}$	matice buzení systému
$\boldsymbol{B_E}$	matice buzení estimátoru
$\boldsymbol{C}$	výstupní matice systému
C	kapacita, F
$\boldsymbol{D}$	matice převodu
$\boldsymbol{E}$	matice odporů
$\boldsymbol{F}$	matice kapacit
f, g	obecná nelineární funkce
$\boldsymbol{H_E}$	matice zpětné vazby estimátoru
$\boldsymbol{I}$	jednotková matice
$\boldsymbol{i}$	vektor proudů
$i_{A\text{-}Z}$	smyčkové proudy, A
$i_{1\text{-}4}$	proudy v jednotlivých větvích, A
J	kritérium
k	diskrétní vzorky času
$\boldsymbol{L}$	matice zpětné vazby regulátoru
L_W	zesílení žádané hodnoty
$\boldsymbol{m}$	pomocný vektor
N	délka intervalu řízení
$\boldsymbol{P}$	váhová matice kritéria
$\boldsymbol{Q}$	váhová matice kritéria
$\boldsymbol{R}$	váhová matice kritéria
R	odpor, Ω
R_Z	odpor zátěže, Ω
t	spojitý čas, s
U	napětí, V
$\boldsymbol{U}$	pomocná matice
$\boldsymbol{u}$	vektor vstupních veličin
u	akční zásah

u_0	napájecí napětí, V
u_3	napětí na zátěži, V
w	žádaná hodnota
x	jednotky převodníku
$\boldsymbol{x}$	vektor stavových veličin
$\boldsymbol{x_0}$	vektor počátečních podmínek
$\boldsymbol{x'}$	derivace vektoru stavových veličin
$\hat{\boldsymbol{x}}$	estimovaný vektor stavových veličin
$\boldsymbol{y}$	vektor výstupních veličin
$\boldsymbol{y_m}$	vektor výstupních veličin změřený na reálné soustavě
y	výstup soustavy

SEZNAM ILUSTRACÍ

SEZNAM TABULEK

ÚVOD

Cílem této diplomové práce bylo vytvořit program pro PC v prostředí PROMOTIC, kterým se realizuje LQ řízení lineárního SISO systému připojeného pomocí Arduino Due.

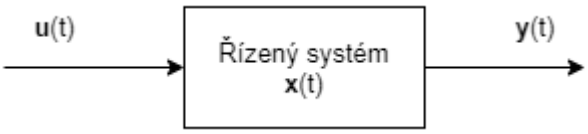
V této práci bude popsána potřebná teorie pro popis regulovaného systému, způsoby získání matematického modelu a popis regulované soustavy, kterou tvoří RC obvod čtvrtého řádu. Dále bude popsána teorie potřebná pro návrh LQ regulátoru s asymptotickým sledováním žádané hodnoty a Luenbergerova deterministického estimátoru.

Programová část práce se bude zbývat návrhem programu pro řídicí systém Arduino Due, který bude zajišťovat měření a generování napěťových signálů včetně zasílání hodnot přes USB do PC. V programovém prostředí MATLAB bude provedena identifikace parametrů modelu na základě změřených dat, které reprezentují skutečné chování soustavy. Pro tento model bude navržen diskrétní estimátor jako duální úloha k návrhu diskrétního LQ regulátoru s asymptotickým sledováním žádané hodnoty. Navržený LQ regulátor s diskrétním estimátorem bude následně implementován do SCADA systému PROMOTIC.

1 TEORETICKÁ ČÁST

1.1 POPIS SYSTÉMU

Řízený lineární dynamický systém zobrazený na obr. 1.1 má obecně vstupní veličiny **u**(t), výstupní veličiny **y**(t) a stavové (vnitřní) veličiny **x**(t). Pokud má systém jednu vstupní a jednu výstupní veličinu, pak se jedná o jednorozměrový systém (SISO). Pokud je vstupních a výstupních veličin více, jedná se o systém mnohorozměrový (MIMO). Počet vstupních signálů tak tvoří vektor vstupních veličin **u**(t), počet výstupních signálů pak vektor výstupních veličin **y**(t). Jednorozměrový systém je pak zvláštním případem, kdy vstupní i výstupní vektory mají jen jeden prvek. Stavový vektor **x**(t) má počet prvků takový, jaký je řád systému (Balátě, 2004).



Obr. 1.1 – Řízený dynamický systém

Pro popis vlastností libovolného systému je zapotřebí matematického modelu, který charakterizuje (popisuje) chování systému. Matematické modely lze rozdělit na dvě skupiny, a to na vnější a vnitřní popis systému (Balátě, 2004).

Vnější popis systému vyjadřuje dynamické vlastnosti vztahů mezi vstupem a výstupem systému. Na systém se díváme jako na černou skříňku (black box) se vstupem a výstupem. Nevíme, co se uvnitř této skříňky děje a nemusíme znát ani strukturu analyzovaného systému. Analyzuje pouze reakci systému na vstupní signály. Jedná se o popis dynamiky systému přenosovou funkcí, případně diferenciální či diferenční rovnicí daného řádu (Balátě, 2004).

Vnitřní popis systému pak vyjadřuje dynamické vlastnosti reakcí mezi vstupem **u**(t), vnitřním stavem **x**(t) a výstupem systému **y**(t) a vede na stavový model systému (Balátě, 2004).

1.1.1 Způsoby získání modelu

Model je uměle vytvořený systém, jehož chování je podobné skutečnému systému. Proces tvorby modelu se nazývá modelování, kdy při tvorbě modelu dochází z hlediska přesnosti k redukci (zjednodušení) na matematický model. Tento proces se pak nazývá identifikace soustavy. Pokud máme k dispozici matematický model systému, pak můžeme

provádět experimenty a zkoumat chování soustavy na tomto modelu. Tyto činnosti prováděné na modelu se nazývají simulace (Balátě, 2004).

Zkoumaný systém pak lze identifikovat analyticky (matematicko-fyzikální analýza), nebo experimentálně. Analytický způsob identifikace je založen na matematickém popisu elementárních jevů, které v soustavě probíhají. Tím dostáváme soustavu algebraických a diferenciálních rovnic, které vycházejí z bilance hmoty, energie, nebo rovnice kontinuity. V těchto rovnicích se vyskytují veličiny jak vstupní a výstupní, tak stavové. Je-li matematicko-fyzikální popis soustavy úplný, pak můžeme jednotlivé rovnice tohoto popisu uspořádat do stavových rovnic. Složky stavového vektoru zde mají fyzikální význam. Analyticky získaný model lze charakterizovat jako vnitřní popis systému (Strejc, 1978).

Experimentální analýzou se určuje matematický model pomocí údajů změřených na vstupech a výstupech soustavy. Hodnoty vstupních a výstupních signálů musí být měřitelné. Metody experimentální analýzy lze rozdělit na metody deterministické a stochastické. U deterministických metod je nutná znalost počátečního stavu systému a relativně jednoduchý vstupní signál. U stochastických metod se připouští jakýkoliv vstupní signál a předem neurčený počáteční stav. Na soustavu pak může působit i parazitní šum, jehož vlastnosti nemusí být předem známé. Takto získaný model lze klasifikovat jako vnější popis chování daného systému. Výsledný model má většinou poměrně jednoduchý tvar a jeho parametry se relativně snadno určují (Strejc, 1978).

1.1.2 Spojitý stavový model

Je uvažován vícerozměrný systém se vstupním vektorem **u**(t), výstupním vektorem **y**(t) a stavovým vektorem **x**(t). Chování systému lze popsat diferenciální rovnicí n-tého řádu, nebo pomocí n diferenciálních rovnic prvního řádu, přičemž tyto způsoby zápisu mezi sebou lze převádět. Diferenciální rovnice vyššího řádu tak odpovídá vnějšímu popisu systému. Soustava diferenciálních rovnic prvního řádu pak odpovídá popisu vnitřnímu, kde nižší derivace mají význam stavových proměnných. Stavový popis je pak určen stavovou rovnicí

$$\boldsymbol{x}'(t) = f[\boldsymbol{x}(t), \boldsymbol{u}(t), t], \tag{1.1}$$

kde $\boldsymbol{x}$ –vektor stavových veličin,

$\boldsymbol{u}$ – vektor vstupních veličin,

t – čas, s (Balátě, 2004).

Výstupy systému se pak určí výstupní rovnicí

$$\boldsymbol{y}(t) = g[\boldsymbol{x}(t), \boldsymbol{u}(t), t], \tag{1.2}$$

kde $\boldsymbol{y}$ –vektor výstupních veličin.

V případě, že je systém lineární a časově invariantní, pak jsou i funkce *f* a *g* lineární s konstantními parametry a můžeme rovnice (1.1) a (1.2) přepsat do vektorově maticového zápisu

$$\begin{aligned} \boldsymbol{x}'(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) \end{aligned}, \tag{1.3}$$

kde $\boldsymbol{x}$ –vektor stavových veličin,

$\boldsymbol{u}$ – vektor vstupních veličin,

$\boldsymbol{y}$ – vektor výstupních veličin,

$\boldsymbol{A}$ – matice systému,

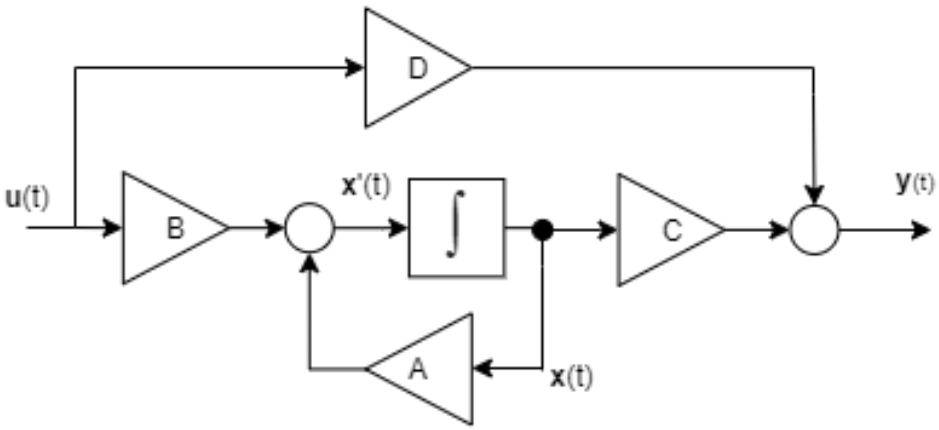
$\boldsymbol{B}$ – matice buzení systému,

$\boldsymbol{C}$ – výstupní matice systému,

$\boldsymbol{D}$ – matice převodu,

t – čas, s.

Matice $\boldsymbol{D}$ je pak u systémů, které nemají přímou vazbu mezi vstupem a výstupem rovnu nule. Vztah (1.3) pak odpovídá spojitému stavovému popisu, kterému odpovídá blokové schéma zapojení zobrazené na obr. 1.2 (Balátě, 2004).



Obr. 1.2 – Blokové schéma spojitého stavového popisu

1.1.3 Diskrétní stavový model

Podobně jako u spojitého systému je chování diskrétní soustavy popsáno diferenciální rovnicí *n*-tého řádu, nebo pomocí n diferenciálních rovnic prvního řádu. V případě diskrétních soustav přechází vektory vstupních, výstupních a stavových veličin do diskrétního tvaru, kde jsou známé jejich hodnoty pouze v určité okamžiky. Tyto okamžiky jsou definované pomocí vzorkovací periody *T*, díky které lze definovat okamžiky času jako $t = t_0 + kT$. Rovnice určující stavy systému (1.1) pak přejde do tvaru

$$\boldsymbol{x}(k+1) = f[\boldsymbol{x}(k), \boldsymbol{u}(k), k], \tag{1.4}$$

kde $\boldsymbol{x}(k+1)$ – vektor stavových veličin v následujícím kroku,
$\boldsymbol{u}(k)$ – vektor vstupních veličin,
$\boldsymbol{x}(k)$ – vektor stavových veličin,
k – diskrétní čas.

Rovnice určující výstup (1.2) přejde do tvaru

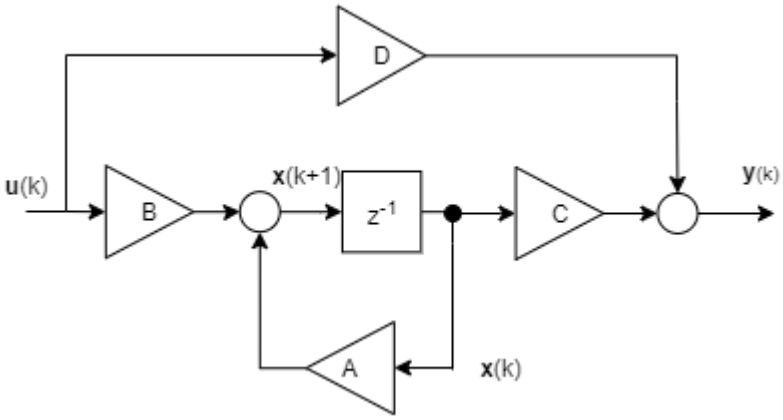
$$\boldsymbol{y}(k) = g[\boldsymbol{x}(k), \boldsymbol{u}(k), k], \tag{1.5}$$

kde $\boldsymbol{y}(k)$ – vektor výstupních veličin.

Rovnice (1.4) a (1.5) pak lze přepsat do vektorově maticového zápisu

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k), \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k) \end{aligned} \tag{1.6}$$

Matice $\boldsymbol{D}$ je pak u systémů, které nemají přímou vazbu mezi vstupem a výstupem rovnu nule. Vztah (1.6) pak odpovídá diskrétnímu stavovému popisu, kterému odpovídá blokové schéma zapojení zobrazené na obr. 1.3 (Balátě, 2004).



Obr. 1.3 – Blokové schéma diskrétního stavového popisu

1.2 LQ REGULÁTOR

LQ regulátor spadá do kategorie stavových regulátorů, kde se hledá posloupnost akčních zásahů, které přivedou dynamický systém z počátečního (nenulového) stavu do stavu nulového (koncového) tak, aby hodnota účelové funkce byla minimální. Jedná se o řízení na nulovou žádanou hodnotu. LQ regulátory se vyznačují tím, že při jejich návrhu je použito kvadratické kritérium. L v názvu znamená, že se jedná o lineární systémy. Písmeno Q pak značí použití kvadratického kritéria. Kritérium pro LQ regulátor je obecně definováno jako (Kupka, 2020)

$$J = \boldsymbol{x}^{\boldsymbol{T}}(N)\boldsymbol{P}(N)\boldsymbol{x}(N) + \sum_{k=0}^{N-1}[\boldsymbol{x}^{\boldsymbol{T}}(k)\boldsymbol{Q}(k)\boldsymbol{x}(k) + \boldsymbol{u}^{T}\boldsymbol{R}(k)\boldsymbol{u}(k)]\,, \tag{1.7}$$

kde $\boldsymbol{x}$ – vektor stavových veličin,

$\boldsymbol{u}$ – vektor vstupních veličin,

J – kritérium,

N – délka intervalu řízení,

$\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{P}$ – váhové matice kritéria.

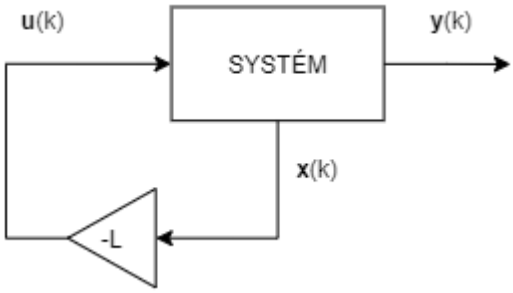
Vztah (1.7) je kritérium pro konečný horizont řízení. Váhové matice $\boldsymbol{Q}$, $\boldsymbol{R}$ a $\boldsymbol{P}$ musí být symetrické a pozitivně semidefinitní. Na intervalu v čase $0 < k < (N - 1)$ se hodnotí dle druhého členu výrazu (1.7) kvadráty stavových i akčních veličin. Od okamžiku, kdy $k = N$, se v kritériu (1.7) uplatňují pouze kvadráty stavových veličin, protože vektor $\boldsymbol{u}(N)$ již nemá vliv na systém posuzovaný v daném intervalu. Řízení je optimální ve smyslu kvadratického kritéria, jestliže

pro libovolný počáteční stav $\boldsymbol{x}(0)$ platí, že funkcionál (1.7) nabývá minimální hodnoty a regulační obvod je stabilní (Strejc, 1978).

Při řízení na nekonečném horizontu řízení bude první člen vztahu (1.7) roven nule a pak lze vztah (1.7) přepsat do tvaru

$$J = \sum_{k=0}^{\infty} [\boldsymbol{x}^{\boldsymbol{T}}(k)\boldsymbol{Q}(k)\boldsymbol{x}(k) + \boldsymbol{u}^{T}\boldsymbol{R}(k)\boldsymbol{u}(k)]\,, \tag{1.8}$$

Člen, který obsahuje matici $\boldsymbol{R}$, minimalizuje energii dodávanou do systému, která je potřebná pro převedení systému z počátečního stavu $\boldsymbol{x}(0)$ do konečného stavu. Člen obsahující matici $\boldsymbol{Q}$ váží odchylky stavů. Tyto matice jsou volitelnými parametry, které volí uživatel během návrhu a určují výsledné chování řízeného systému. Čím větší bude hodnota matice $\boldsymbol{Q}$, tím vyšší bude rychlost ustálení regulované veličiny. Velikost matice $\boldsymbol{R}$ pak ovlivňuje velikost akčních zásahů, kde malé hodnoty matice $\boldsymbol{R}$ odpovídají velkým akčním zásahům. Blokové schéma zapojení LQ regulátoru pak zobrazuje obr. 1.4 (Machek, 2019).



Obr. 1.4 – Blokové schéma LQ regulátoru

Řešení minimalizace kvadratického kritéria LQ regulátoru dle vztahu (1.8) se převádí na řešení Riccatiho rovnice a výpočet akčního zásahu pro soustavu popsanou dle vztahu (1.6) vyjde jako zesílení od stavu

$$\boldsymbol{u}(k) = -\boldsymbol{L}\boldsymbol{x}(k)\,, \tag{1.9}$$

kde $\boldsymbol{L}$ – matice zpětné vazby regulátoru.

Kde se matice $\boldsymbol{L}$ vypočítá dle následujícího vztahu

$$\boldsymbol{L} = [\boldsymbol{R} + \boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{P}\boldsymbol{B}]^{-1}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{P}\boldsymbol{A}, \tag{1.10}$$

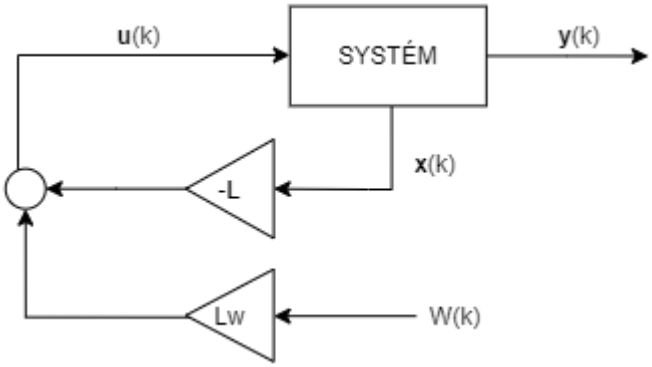
Matice $\boldsymbol{P}$ je řešením ustáleného tvaru Riccatiho rovnice

$$\boldsymbol{P} = \boldsymbol{Q} + \boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{P}\boldsymbol{A} - \boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{P}\boldsymbol{B}[\boldsymbol{R} + \boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{P}\boldsymbol{B}]^{-1}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{P}\boldsymbol{A}, \tag{1.11}$$

Na konečném horizontu řízení je výsledná matice zpětné vazby $\boldsymbol{L}$ proměnná v čase. Zavedením řízení na nekonečném horizontu se řešení Riccatiho rovnice ustálí a matice $\boldsymbol{L}$ tak bude konstantní. Je nutné, aby systém splňoval podmínky úplné řiditelnosti a byl úplně rekonstruovatelný (Machek, 2019).

1.3 LQ REGULÁTOR S ASYMPTOTICKÝM SLEDOVÁNÍM ŽÁDANÉ

Modifikací pro sledování žádané hodnoty bylo vyvinuto několik. V této kapitole je popsána modifikace pro LQ regulátor s asymptotickým sledováním žádané, která zajistí požadovanou hodnotu na výstupu řízeného systému. Blokové schéma regulátoru s asymptotickým sledováním žádané je zobrazeno na obr. 1.4 (Dušek, 2021).



Obr. 1.5 – Blokové schéma LQ regulátoru
s asymptotickým sledováním žádané hodnoty

Mějme systém, jehož chování je popsáno stavovým popisem dle vztahu (1.6). Z obr. 1.5 pak lze odvodit rovnici pro výpočet akčního zásahu jako

$$u(k) = -\boldsymbol{L}\boldsymbol{x}(k) + L_W(w), \tag{1.12}$$

kde $\boldsymbol{x}$ – vektor stavových veličin,

u – akční zásah,

$L_W(w)$ – hledaná funkce žádané hodnoty,

$\boldsymbol{L}$ – matice zpětné vazby regulátoru.

V ustáleném stavu platí

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) = \boldsymbol{x}, \tag{1.13}$$

Dosazením vztahu (1.12) do rovnice určující stav systému (1.6) dostáváme

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}(-\boldsymbol{L}\boldsymbol{x} + L_W(w)), \tag{1.14}$$

kde $\boldsymbol{A}$ – matice systému,

$\boldsymbol{B}$ – matice buzení systému.

Po roznásobení závorky a vytknutí $\boldsymbol{x}$ na pravé straně z rovnice (1.14) dostáváme

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}(-\boldsymbol{L}\boldsymbol{x} + L_W(w)), \tag{1.15}$$

Z rovnice (1.15) pak lze vyjádřit $\boldsymbol{x}$

$$\boldsymbol{x} = [\boldsymbol{I} - (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{L})]^{-1}\boldsymbol{B}L_W(w), \tag{1.16}$$

kde $\boldsymbol{I}$ – jednotková matice.

V ustáleném stavu se výstupní hodnota bude rovnat žádané. Výstupní rovnice pro ustálený stav pak bude ve tvaru

$$w = y = \boldsymbol{C}\boldsymbol{x}, \tag{1.17}$$

kde w – žádaná hodnota,

$\boldsymbol{C}$ – výstupní matice systému,

y – výstup soustavy.

Dosazením rovnice (1.16) do rovnice (1.17) vznikne

$$w = \boldsymbol{C}[\boldsymbol{I} - (\boldsymbol{A} - \boldsymbol{BL})]^{-1}\boldsymbol{B}L_W(w), \tag{1.18}$$

Z rovnice (1.18) pak lze získat vztah pro výpočet L_W

$$L_W(w) = \{\boldsymbol{C}[\boldsymbol{I} - (\boldsymbol{A} - \boldsymbol{BL})]^{-1}\boldsymbol{B}\}^{-1}w, \tag{1.19}$$

1.4 ESTIMÁTORY

Při regulaci stavovým regulátorem, který jako zpětnou vazbu využívá stavové veličiny, je třeba znát všechny vnitřní stavy soustavy. Velmi často se stává, že buď všechny, nebo jen některé složky stavového vektoru jsou neměřitelné, nebo je jejich měření příliš drahé nebo nebezpečné. V takovém případě je třeba neměřitelné stavy soustavy odhadovat výpočtem neboli estimovat. Estimátor na základě znalosti vstupů a výstupů soustavy a jejího modelu průběžně odhaduje hodnoty stavového vektoru soustavy. Často se lze setkat i s pojmy, jako je rekonstruktor nebo stavový pozorovatel. Pro realizaci estimátoru je nutná znalost stavového popisu soustavy. Estimátory se pak dělí na deterministické a stochastické (náhodné, ovlivněné náhodným šumem) a na spojité a diskrétní (Strejc, 1978).

Estimátor se často využívá v regulační smyčce při regulaci stavovým regulátorem. Získané stavové veličiny jsou pak použity jako vstupy do stavového regulátoru. Mezi nejčastěji používané estimátory patří například estimátor úplného řádu (Luenbergerův), estimátor redukovaného řádu, Kalmánův estimátor (filtr) (Svatoš, 2015).

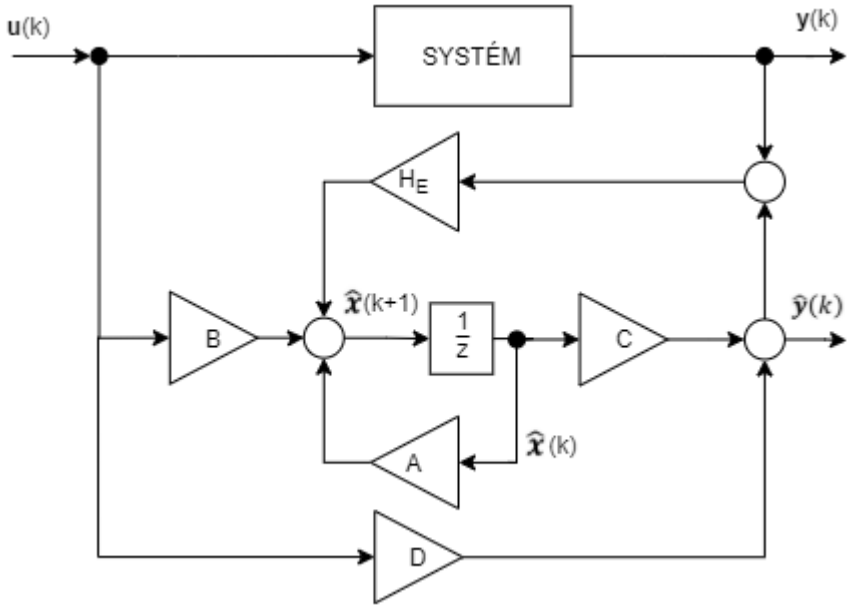
Estimátor úplného řádu odhaduje všechny stavy systému. Pokud měřená výstupní veličina nebo stavové veličiny neobsahují žádný aditivní šumový signál, tak se jedná o deterministickou estimaci. Parametry estimátoru jsou optimalizovány vzhledem k dynamice estimovaného procesu. Soustava musí splňovat podmínku pozorovatelnosti a předpokládá se znalost stavových matic (Strejc, 1978).

Estimátor redukovaného řádu je zjednodušenou formou estimátoru úplného řádu. Uvažuje se, že část stavového vektoru systému je měřitelná, tj. jde o výstupní veličiny. Pokud má vektor výstupu $\boldsymbol{y}$ m lineárně nezávislých složek, tak lze řád estimátoru redukovat na hodnotu $n - m$, kde n je řád estimátoru (Strejc, 1978).

V případě Kalmánova filtru uvažujeme, že na systém působí náhodné vstupní signály.

1.4.1 Luenbergův deterministický estimátor

Mějme systém, jehož chování je popsáno stavovým popisem dle vztahu (1.6). Blokové schéma zapojení estimátoru pak znázorňuje obr. 1.6. Pro přehlednost jsou vektory a matice estimátoru označeny symbolem „^".



Obr. 1.6 – Blokové schéma zapojení estimátoru

Ze znalosti schématu zapojení pak lze psát rovnici

$$\begin{aligned} \hat{\boldsymbol{x}}(k+1) &= \boldsymbol{A}\hat{\boldsymbol{x}}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{H_E}[\boldsymbol{y}(k) - \hat{\boldsymbol{y}}(k)], \\ \hat{\boldsymbol{y}}(k) &= \boldsymbol{C}\hat{\boldsymbol{x}}(k) + \boldsymbol{D}\boldsymbol{u}(k) \end{aligned} \tag{1.20}$$

kde $\boldsymbol{x}$ – vektor stavových veličin,

$\boldsymbol{A}$ – matice systému,

$\boldsymbol{B}$ – matice buzení systému,

$\boldsymbol{C}$ – výstupní matice systému,

$\boldsymbol{D}$ – matice převodu,

$\boldsymbol{H}_\mathrm{E}$ – matice zpětné vazby estimátoru,

$\boldsymbol{y}$ – vektor výstupních veličin.

Dosazením druhé rovnice do první ve vztahu (1.20) a roznásobením závorek dostaneme

$$\hat{\boldsymbol{x}}(k+1) = \boldsymbol{A}\hat{\boldsymbol{x}}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{H_E}\boldsymbol{y}(k) - \boldsymbol{H_E}\boldsymbol{C}\hat{\boldsymbol{x}}(k) + \boldsymbol{H_E}\boldsymbol{D}\boldsymbol{u}(k), \tag{1.21}$$

Po vytknutí vektorů $\boldsymbol{x}$ a $\boldsymbol{u}$ dostaneme rovnici ve tvaru

$$\hat{\boldsymbol{x}}(k+1) = \underbrace{(\boldsymbol{A} - \boldsymbol{H_E}\boldsymbol{C})}_{\boldsymbol{A_E}}\hat{\boldsymbol{x}}(k) + \underbrace{(\boldsymbol{B} + \boldsymbol{H_E}\boldsymbol{D})}_{\boldsymbol{B_E}}\boldsymbol{u}(k) + \boldsymbol{H_E}\boldsymbol{y}(k), \tag{1.22}$$

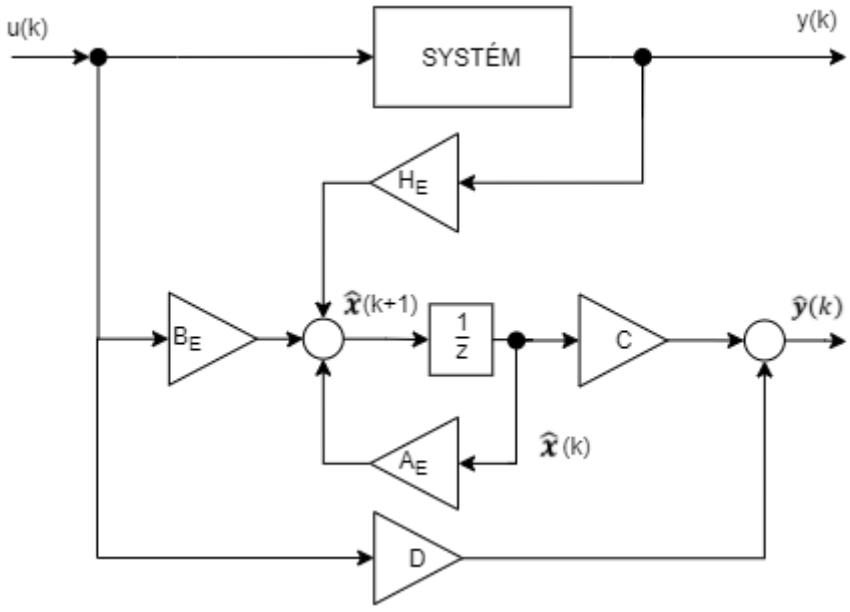
kde $\boldsymbol{A}_\mathbf{E}$ – matice estimátoru,

$\boldsymbol{B}_\mathbf{E}$ – matice buzení estimátoru.

Vztah (1.22) lze přepsat do tvaru

$$\hat{\boldsymbol{x}}(k+1) = \boldsymbol{A_E}\hat{\boldsymbol{x}}(k) + \boldsymbol{B_E}\boldsymbol{u}(k) + \boldsymbol{H_E}\boldsymbol{y}(k), \tag{1.23}$$

Vztahu (1.23) odpovídá blokové schéma zapojení zobrazené na obr. 1.7, které je druhou možností zapojení estimátoru zobrazeného na obr. 1.6.



Obr. 1.7 – Modifikované schéma zapojení estimátoru

Matice estimátoru se určí vztahem

$$\begin{aligned} \boldsymbol{A_E} &= \boldsymbol{A} - \boldsymbol{H_E C}, \\ \boldsymbol{B_E} &= \boldsymbol{B} + \boldsymbol{H_E D}, \end{aligned} \tag{1.24}$$

Vektor $\boldsymbol{H}_\mathbf{E}$ volíme tak, aby vlastní čísla λ_i matice $\boldsymbol{A}_\mathbf{E}$ ležela uvnitř jednotkové kružnice

$$det(\lambda \boldsymbol{I} - \boldsymbol{A_E}) = \det(\lambda \boldsymbol{E} - \boldsymbol{A} + \boldsymbol{H_E C}) = (\lambda - \lambda_1)(\lambda - \lambda_2) \cdots, \tag{1.25}$$

kde $\boldsymbol{I}$ – jednotková matice.

Podmínkou stability estimace je, aby matice $\boldsymbol{A}_\mathbf{E}$ byla stabilní. Pokud se $\lambda_i = 0$, pak bude estimátor minimální a konečný. Pro SISO systémy je počet prvků matice $\boldsymbol{H}_\mathbf{E}$ stejný, jako počet vlastních čísel λ_i a je dán řádem n dynamického systému (Kupka, 2020).

Návrh estimátoru lze řešit i jako duální úlohu k návrhu LQ regulátoru, která byla popsána v kapitole 1.2 . Kritérium popsané vztahem (1.8) přejde do tvaru

$$J = \sum_{k=0}^{\infty} [\hat{\boldsymbol{x}}^T(k) \boldsymbol{Q}(k) \hat{\boldsymbol{x}}(k) + \boldsymbol{y}^T(k) \boldsymbol{R}(k) \boldsymbol{y}(k)], \tag{1.26}$$

kde $\hat{\boldsymbol{x}}$ – estimovaný vektor stavových veličin,

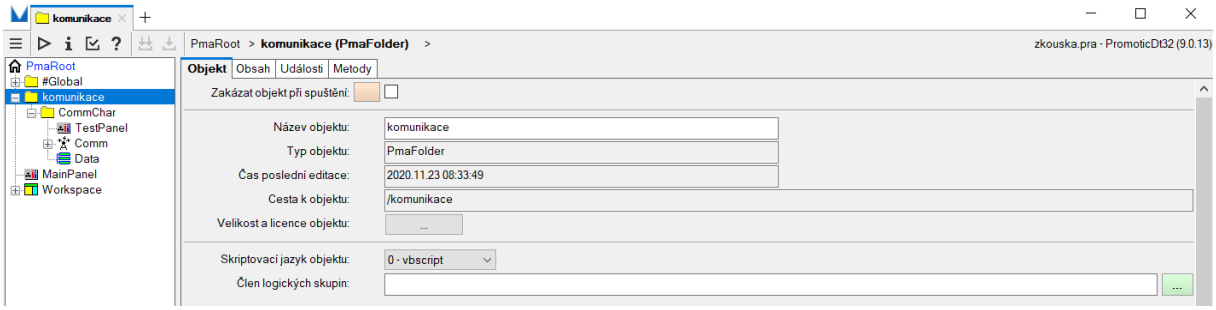
$\boldsymbol{Q}$, $\boldsymbol{R}$ – váhové matice kritéria,

$\boldsymbol{y}$ – vektor výstupních veličin.

1.5 MOŽNOSTI SCADA SYSTÉMU PROMOTIC

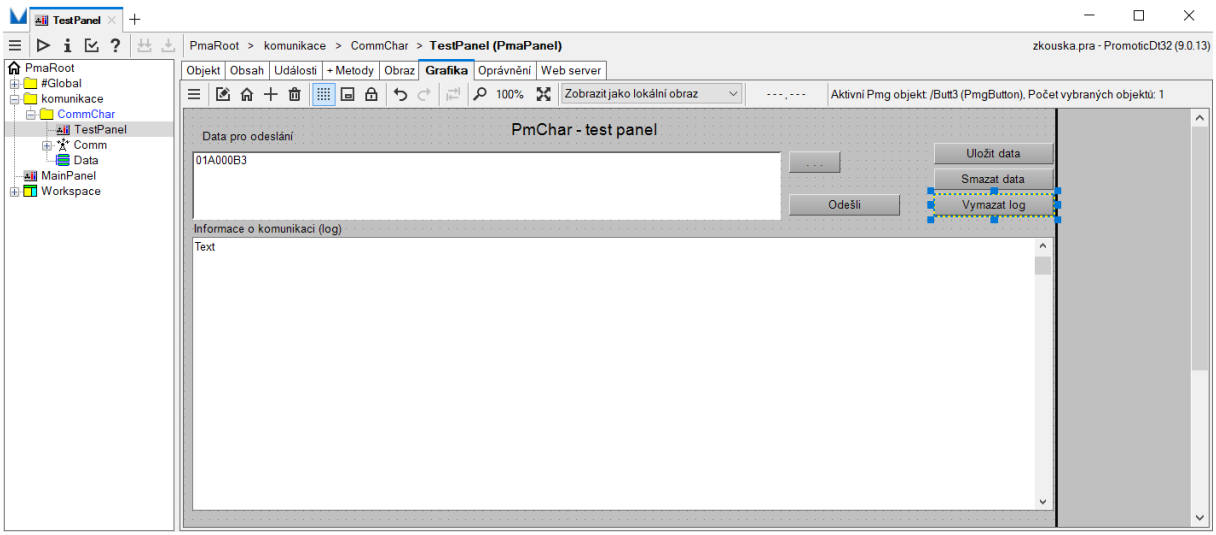
Zkratka SCADA znamená v překladu systém pro dohled, řízení a sběr dat. Systém PROMOTIC je komplexní SCADA objektový softwarový nástroj určený pro tvorbu aplikací, které monitorují, řídí a zobrazují technologické procesy v různých oblastech průmyslu. Systém PROMOTIC obsahuje všechny nezbytné komponenty a předdefinované objekty pro tvorbu jednoduchých i rozsáhlých řídicích a vizualizačních systémů. Systém PROMOTIC také nabízí od verze 8.0 bezplatné vývojové prostředí bez nutnosti zakoupení licence. Díky tomu lze vyvíjet a provozovat malé aplikace zdarma. Freeware režim umožňuje využívat všechny komunikační ovladače a rozhraní, která jsou k dispozici v komerčním režimu. Jediným omezením je maximální velikost aplikace do sta proměnných a maximálně deset obrazů (Obsah dokumentace PROMOTIC).

Vývojové prostředí, které tvoří základní nástroj pro tvorbu aplikací, se dělí na dvě základní části. První částí je editor Pma objektů, který má charakter stromové struktury a obsahuje základní Pma objekty aplikace. Pro nastavování jednotlivých Pma objektů slouží okno pro definování jejich vlastností. Editor Pma objektů je znázorněn na obr. 1.8 (Obsah dokumentace PROMOTIC).



Obr. 1.8 – Editor Pma objektů

Druhou částí je editor grafiky, kde se vytváří a nastavují grafické Pmg objekty. Speciální částí je pak editor scriptů, který umožňuje v každém Pma nebo Pmg objektu v kartě události nebo metody vytvořit program obsluhy události či programovou část metody. Na obr. 1.9 je znázorněn editor Pmg objektů.



Obr. 1.9 – Editor Pmg objektů

Systém PROMOTIC používá pro psaní algoritmů skriptovací jazyky VBScript nebo JavaScript, který lze od verze 9.0 používat všude.

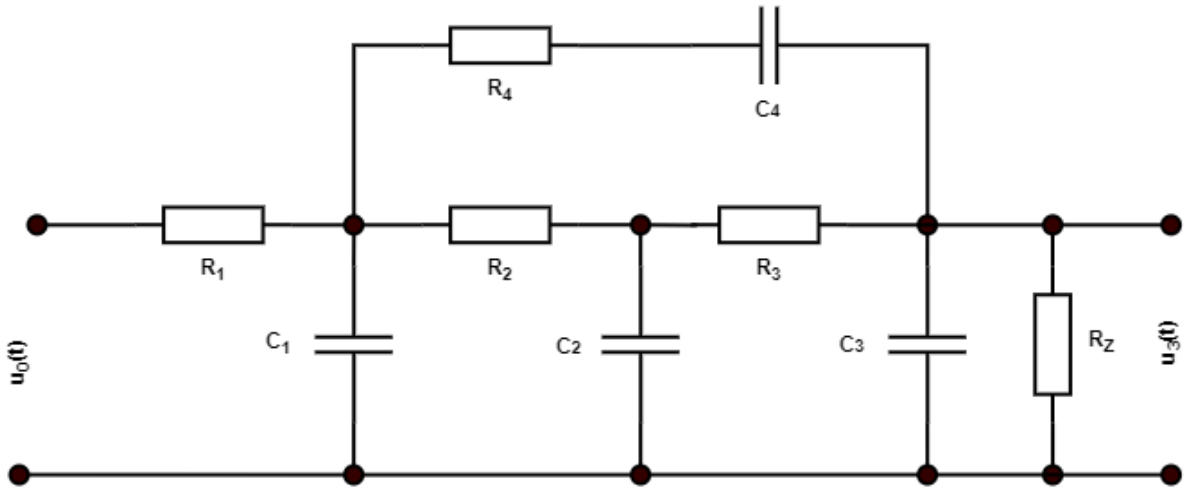
Každá aplikace, která je zaměřená na monitorování a řízení technologických procesů, bude velmi často komunikovat s externím zařízením. PROMOTIC proto disponuje širokou škálou komunikačních možností (zabudované komunikační ovladače, standardizovaná rozhraní, přístup k databázím, web server) (Obsah dokumentace PROMOTIC).

Pro ukládání hodnot a jejich zobrazování je PROMOTIC vybaven systémem trendů, které umožnují ukládání zvolených proměnných s časovou známkou do paměti, nebo na disk počítače. Tyto hodnoty je pak možné zobrazovat graficky nebo tabulkově. Pro upozornění na určitý stav nebo překročení hodnot některých proměnných jsou určeny alarmy (Obsah dokumentace PROMOTIC).

2 PRAKTICKÁ ČÁST

2.1 POPIS REGULOVANÉ SOUSTAVY

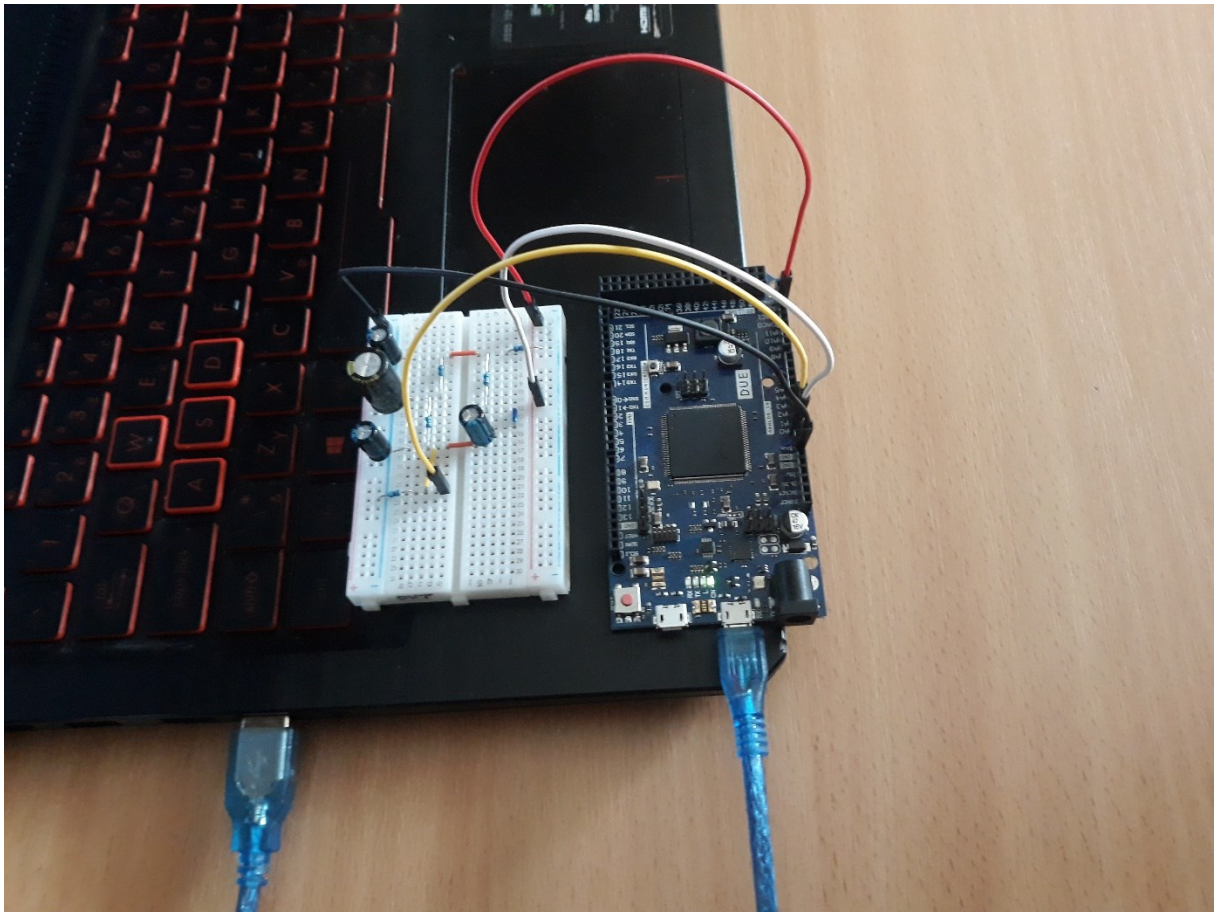
Regulovanou soustavu s jedním vstupem a jedním výstupem (SISO) tvoří zapojení pasivních součástek na nepájivém poli. Schéma zapojení je znázorněno na obr. 2.1. Zapojení tvoří čtyři RC články se zatěžovacím odporem R_Z. Vstupní napětí u_0 je přiváděno na odpor R_1 a výstupní napětí u_3 je měřeno na kondenzátoru C_3. Jelikož byly použity pouze pasivní součástky, tak není potřeba samostatného napájení.



Obr. 2.1 – Schéma zapojení RC článku

Pro generování vstupního napětí a měření výstupního, případně dalších napětí, byla použita vývojová deska Arduino Due. Generování a měření napětí je řízeno programem v Arduino Due, který prostřednictvím sériové linky přenáší měřená napětí a požadavky od

řídicího programu z připojeného počítače. Celý řízený systém na nepájivém poli a připojený k vývojové desce Arduino Due je zobrazen na obr. 2.2.



Obr. 2.2 – Řízený systém a Arduino Due

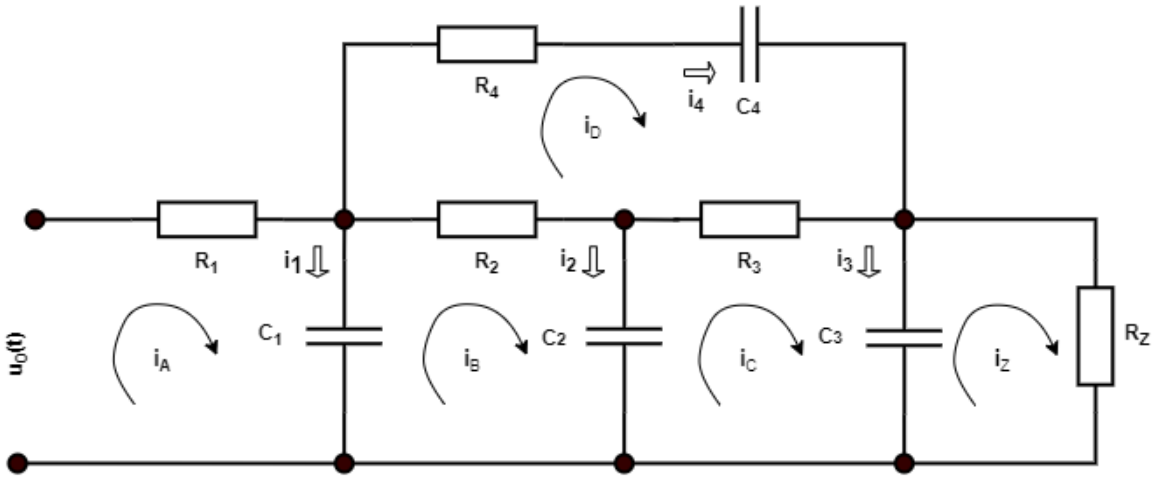
Hodnoty jednotlivých součástek jsou uvedeny v tab. 2.1.

Tab. 2.1 – Hodnoty součástek

R_1, kΩ	R_2, kΩ	R_3, kΩ	R_4, Ω	R_Z, kΩ	C_1, µF	C_2, µF	C_3, µF	C_4, µF
1,5	1,5	1,5	56	47	47	820	47	47

2.1.1 Odvození diferenciálních rovnic

Pro získání spojitého dynamického matematického modelu ve formě diferenciálních rovnic bylo využito analytického přístupu. Použitím platných fyzikálních zákonů, které se běžně užívají v elektrotechnice a aplikací metody smyčkových proudů, na obvod zobrazený na obr. 2.1 bude schéma zapojení doplněné o smyčkové proudy, jak je uvedeno na obr. 2.3.



Obr. 2.3 – Schéma zapojení popsané metodou smyčkových proudů

Aplikací smyčkových proudů (využívajících Kirchhoffovy zákony) tak dostaneme soustavu rovnic

$$
\begin{aligned}
u_0 &= R_1 i_A + \frac{1}{C_1}\int_0^t (i_A - i_B)dt \\
0 &= R_2(i_B - i_D) + \frac{1}{C_2}\int_0^t (i_B - i_C)dt + \frac{1}{C_1}\int_0^t (i_B - i_A)dt \\
0 &= R_3(i_C - i_D) + \frac{1}{C_3}\int_0^t (i_C - i_Z)dt + \frac{1}{C_2}\int_0^t (i_C - i_B)dt, \\
0 &= R_4 i_D + \frac{1}{C_4}\int_0^t i_D dt + R_2(i_D - i_B) + R_3(i_D - i_C) \\
0 &= R_Z i_Z + \frac{1}{C_3}\int_0^t (i_Z - i_C)
\end{aligned}
\tag{2.1}
$$

kde t – spojitý čas, s,

R – odpor, Ω,

$i_{A\text{-}Z}$ – smyčkové proudy, A,

C – kapacita, F,

u_0 – napájecí napětí, V.

Také platí obecné vztahy

$$u = \frac{1}{C}\int i\,dt \quad a \quad i = C\frac{du}{dt}, \tag{2.2}$$

kde u – napětí na kondenzátoru, V,

i – proud protékající kondenzátorem, A,

C – kapacita kondenzátoru, F.

Ze znalosti schématu zapojení dle obr. 2.3 pak lze psát

$$\begin{aligned} i_1 &= i_A - i_B \quad \rightarrow \quad i_A = i_1 + i_2 + i_3 + i_Z \\ i_2 &= i_B - i_C \quad \rightarrow \quad i_B = i_2 + i_3 + i_Z \\ i_3 &= i_C - i_Z \quad \rightarrow \quad i_C = i_3 + i_Z \\ i_4 &= i_D \quad \rightarrow \quad i_D = i_4 \\ i_Z &= \frac{1}{R_Z}\,u_3 \end{aligned} \quad , \tag{2.3}$$

kde $i_{1\text{-}4}$ – proudy v jednotlivých větvích, A,

$i_{A\text{-}Z}$ – smyčkové proudy, A,

u_3 – napětí na zátěži, V,

R_Z – odpor zátěže, Ω,

Následně lze v rovnici (2.1) nahradit proudy i_A, i_B, i_C, i_D dle vztahu (2.3) a za integrály dosadit příslušná napětí dle vztahu (2.2) a dostaneme

$$\begin{aligned} u_0 &= R_1(i_1 + i_2 + i_3 + i_Z) + u_1 \\ 0 &= R_2(i_1 + i_2 + i_3 + i_Z) + u_2 - u_1 \\ 0 &= R_3(i_3 + i_Z - i_4) + u_3 - u_2 \\ 0 &= R_4 i_4 + u_4 + R_2(i_4 - i_2 - i_3 - i_Z) + R_3(i_4 - i_3 - i_Z)' \\ 0 &= R_Z i_Z - u_3 \;\rightarrow\; i_Z = \frac{1}{R_Z}u_3 \end{aligned} \tag{2.4}$$

Nyní dosazením za proud i_Z dle vztahu (2.4) a převedením všech napětí na jednu stranu rovnice a proudů na druhou stranu dostáváme

$$
\begin{aligned}
u_0 - u_1 - \frac{R_1}{R_Z}u_3 &= R_1 i_1 + R_1 i_2 + R_1 i_3 \\
u_1 - u_2 - \frac{R_2}{R_Z}u_3 &= R_2 i_2 + R_2 i_3 - R_2 i_4 \\
u_2 - u_3 - \frac{R_3}{R_Z}u_3 &= R_3 i_3 - R_3 i_4 \\
\frac{R_2}{R_Z}u_3 + \frac{R_3}{R_Z}u_3 - u_4 &= R_4 i_4 + R_2 i_4 - R_2 i_2 - R_2 i_3 + R_3 i_4 - R_3 i_3
\end{aligned}
\quad , \tag{2.5}
$$

Následně můžeme vztah (2.5) přepsat do maticové formy

$$
\begin{bmatrix}
-1 & 0 & -\frac{R_1}{R_Z} & 0 \\
1 & -1 & -\frac{R_2}{R_Z} & 0 \\
0 & 1 & \left(-1-\frac{R_3}{R_Z}\right) & 0 \\
0 & 0 & \left(\frac{R_2+R_3}{R_Z}\right) & -1
\end{bmatrix}
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}
+ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u_0 =
\tag{2.6}
$$

$$
\begin{bmatrix}
R_1 & R_1 & R_1 & 0 \\
0 & R_2 & R_2 & -R_2 \\
0 & 0 & R_3 & -R_3 \\
0 & -R_2 & (-R_2-R_3) & (R_2+R_3+R_4)
\end{bmatrix}
\begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix},
$$

Rovnici (2.6) lze přepsat obecně jako

$$
\boldsymbol{U}\boldsymbol{u} + \boldsymbol{m}u_0 = \boldsymbol{E}\boldsymbol{i}, \tag{2.7}
$$

kde $\boldsymbol{U}$ – pomocná matice,

$\boldsymbol{u}$ – vektor vstupních veličin,

$\boldsymbol{m}$ – pomocný vektor,

u_0 – napájecí napětí, V,

$\boldsymbol{E}$ – matice odporů, Ω,

$\boldsymbol{i}$ – vektor proudů, A.

Po vyjádření vektoru proudů $\boldsymbol{i}$ ze vztahu (2.7) dostaneme

$$\boldsymbol{i} = \boldsymbol{E}^{-1}(\boldsymbol{U}\boldsymbol{u} + \boldsymbol{m}u_0), \tag{2.8}$$

Nyní dosadíme vztah (2.8) do druhé rovnice ze vztahu (2.2) a dostaneme

$$\boldsymbol{F}\frac{d\boldsymbol{u}}{dt} = \boldsymbol{E}^{-1}(\boldsymbol{U}\boldsymbol{u} + \boldsymbol{m}u_0), \tag{2.9}$$

kde $\boldsymbol{F}$ – matice kapacit.

Ve vztahu (2.9) je matice $\boldsymbol{F}$ čtvercová a v tomto případě má rozměr 4×4. Na diagonále jsou hodnoty kondenzátorů C_1, C_2, C_3, C_4. Nyní je potřeba osamostatnit derivaci vektoru napětí, a to za použití inverze, čímž dostaneme

$$\frac{d\boldsymbol{u}}{dt} = \boldsymbol{F}^{-1}\boldsymbol{E}^{-1}(\boldsymbol{U}\boldsymbol{u} + \boldsymbol{m}u_0), \tag{2.10}$$

Po roznásobení závorek ve vztahu (2.10) dostáváme finální vztah, ze kterého vyplývá výpočet matic $\boldsymbol{A}$ a $\boldsymbol{B}$. Rovnici (2.10) tedy převedeme do standardního tvaru stavového modelu (2.12).

$$\frac{d\boldsymbol{u}}{dt} = \underbrace{\boldsymbol{F}^{-1}\boldsymbol{E}^{-1}\boldsymbol{U}}_{\boldsymbol{A}}\boldsymbol{u} + \underbrace{\boldsymbol{C}^{-1}\boldsymbol{I}^{-1}\boldsymbol{m}}_{\boldsymbol{B}}\ u_0, \tag{2.11}$$

kde $\boldsymbol{A}$ – matice systému,

$\boldsymbol{B}$ – matice buzení systému.

Následně můžeme vztah (2.11) přepsat do obecného tvaru

$$\begin{aligned} \frac{d\boldsymbol{x}}{dt} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ y &= \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u} \end{aligned}, \tag{2.12}$$

kde t – spojitý čas,

$\boldsymbol{x}$ – vektor stavových veličin,

$\boldsymbol{A}$ – matice systému,

$\boldsymbol{B}$ – matice buzení systému,

u – vektor vstupních veličin,

C – výstupní matice systému,

D – matice převodu.

Matice soustavy pak budou

$$A = \begin{bmatrix} -\frac{1}{C_1}\left(\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_4}\right) & \frac{1}{C_1R_2} & \frac{1}{C_1R_4} & \frac{1}{C_1R_4} \\ \frac{1}{C_2R_2} & -\frac{1}{C_2}\left(\frac{1}{R_2}+\frac{1}{R_3}\right) & \frac{1}{C_1R_3} & 0 \\ \frac{1}{C_3R_4} & \frac{1}{C_3R_3} & -\frac{1}{C_3}\left(\frac{1}{R_3}+\frac{1}{R_4}+\frac{1}{R_Z}\right) & -\frac{1}{C_4R_4} \\ \frac{1}{C_4R_4} & 0 & -\frac{1}{C_4R_4} & -\frac{1}{C_4R_4} \end{bmatrix}, \quad (2.13)$$

$$B = \begin{bmatrix} \frac{1}{C_1R_1} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad C = [0 \quad 0 \quad 1 \quad 0] \quad D = 0, \quad (2.14)$$

Rovnice (2.13) a (2.14) pak zobrazují výslednou podobu matic ***A***, ***B***, ***C*** a ***D***.

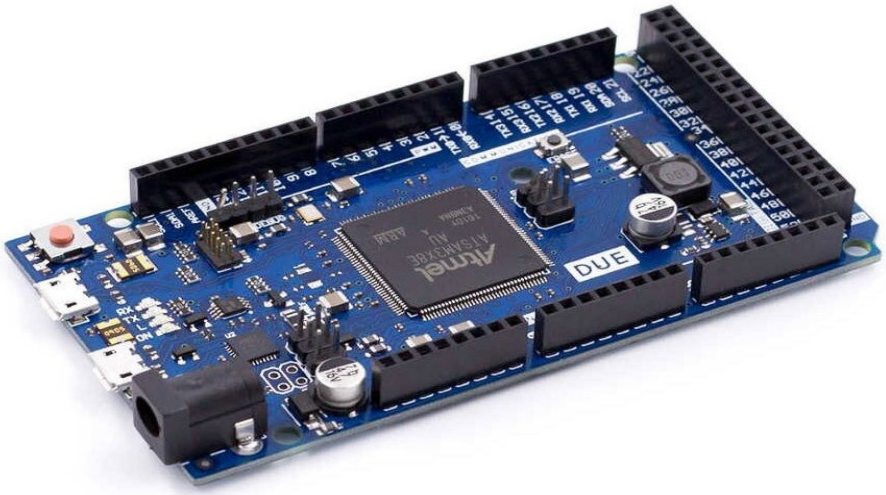
2.2 ARDUINO DUE

Tato deska je osazená procesorem Atmel ATSAM3X8EA ARM Cortex-M3. Jedná se o první desku projektu Arduino, která je založená na 32bitovém jádru ARM s frekvencí 84 MHz, což je proti standardním 8bitovým mikroprocesorům Atmel AVR s taktovací frekvencí maximálně 16 MHz velký skok. Deska má 54 digitálních I/O pinů (12 z nich je možné použít jako PWM výstup), 12 analogových vstupů, 4 UART (hardwarový sériový port), 2 DAC (analogový výstup), 2 microUSB konektory (kde jeden slouží pro programování procesoru a druhý pro připojení dalších zařízení) s převodníkem ATMEGA16U2-MU, 2 TWI, SPI, tlačítkem pro reset a napájecím konektorem (Arduino DUE, 2021).

Procesor disponuje pamětí FLASH o velikosti 512 kB a pamětí SRAM o velikosti 96 kB, která je rozdělena na 2 části (64 kB a 32 kB). Doporučená úroveň napájecího napětí

je ÷ 12 V. Limitní úroveň napájecího napětí pak je v rozsahu 6 ÷ 16 V. Maximální napětí, které všechny I/O piny tolerují, je 3,3 V s maximálním proudovým zatížením 130 mA. Maximální proud pro pin s 3,3 V a 5 V (určené pro napájení připojených zařízení) je 800 mA (Arduino DUE, 2021).

Deska Arduino DUE je zobrazena na obr. 2.4.



Obr. 2.4 – Arduino DUE (Arduino DUE, 2021)

3 PROGRAMOVÉ VYBAVENÍ – ARDUINO DUE

V této kapitole je popsán návrh a implementace programu pro Arduino Due, které bude realizovat měření a generování napětí dle požadavků přenášených sériovou linkou s využitím znakově (ASCII) orientovaného protokolu z připojeného systému.

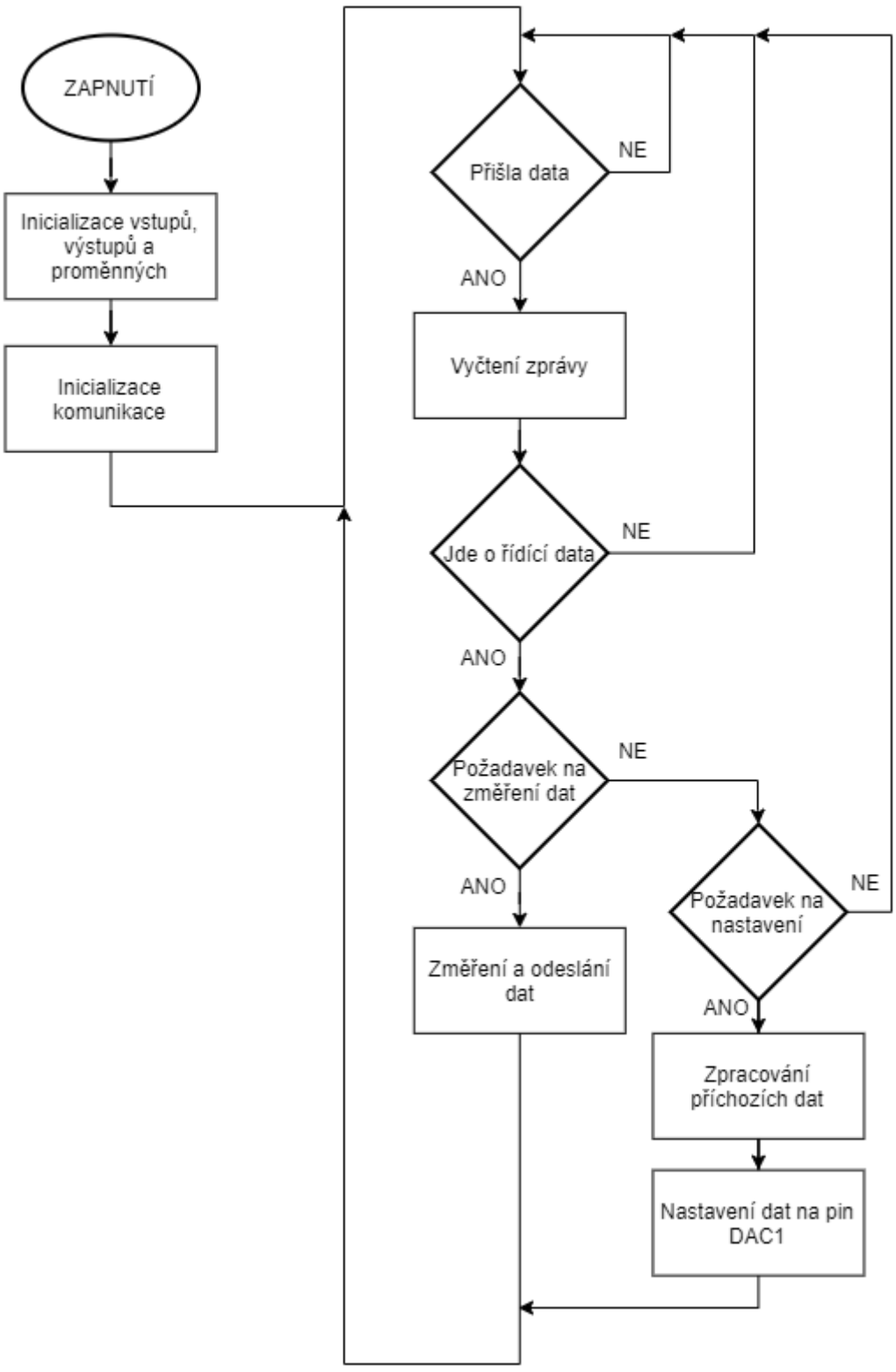
3.1 PROGRAM PRO MĚŘENÍ A GENEROVÁNÍ NAPĚŤOVÝCH SIGNÁLŮ

Tato část kódu byla napsána ve volně dostupném softwaru Arduino IDE ve verzi 1.8.13. Tento program byl navržen tak, aby reagoval na příchozí požadavky řídicího systému, které jsou zasílané přes sériovou linku, a vykonával tak předem stanovené činnosti. První činností je generování napěťových signálů na vstup do RC obvodu. Druhou funkcí je pak měření napětí a zasílání naměřených hodnot přes USB sběrnici do PC.

Komunikace je typu master – slave. Arduino zde funguje jako slave, který odpovídá na dotazy mastera, kterým je řídicí aplikace v počítači (řídicí aplikace je v Promoticu, nebo v MATLABu, který byl použit pro návrh a ověření).

Řídicí aplikace zasílá požadavky přes sériové rozhraní. Komunikace probíhá podle stanoveného protokolu, kde je zpráva tvořena posloupností ASCII znaků = data (několik bytů dle velikosti zprávy) následovaných dvěma ukončovacími znaky „CR“+„LF“. První dva znaky datové části zprávy určují daný požadavek. Požadavky byly použity dva. První požadavek je indikován znaky „A:“. Při přijetí tohoto požadavku Arduino provede měření a odešle naměřená data zpět. Druhým požadavkem „N:xxxx“ je požadavek na nastavení hodnoty xxxx na vstup do RC článku.

Vývojový diagram je znázorněn na obr. 3.1.



Obr. 3.1 – Vývojový diagram pro Arduino

Po spuštění programu (připojení napájení) dojde k prvotnímu nastavení a definování proměnných. Zde dojde k definování vstupů a výstupů, nastavení přenosové rychlosti pro sériovou komunikaci a nastavení rozlišení A/D převodníku z výchozích 10 bit na 12 bit, díky čemuž pak Arduino vrací hodnoty v rozsahu 0 ÷ 4095. Program pak běhá v nekonečné smyčce, kde se čeká na splnění podmínky příchodu dat. Pokud je podmínka splněna, dochází k zjištění počtu přijatých bajtů pomocí funkce *n = Serial.available()*. Následně pomocí příkazu *Serial.readbytes()* dojde k vyčtení všech bajtů do proměnné, čímž zároveň dojde k smazání dat v přijímacím bufferu.

Nyní dojde k ověření, zda jsou data validní a má na ně program reagovat. Tato podmínka je splněna za předpokladu, že je na druhé pozici znak „:“. V případě splnění této podmínky je ověřeno, že se jedná o řídicí data a dochází k ověření toho, který příkaz se má vykonat. Toho je docíleno pomocí struktury switch, kde se kontroluje první znak ve zprávě.

Pokud je první znak zprávy „A“, jedná se o požadavek na změření dat. Dochází ke změření napětí na příslušných pinech a jejich odeslání přes sériovou linku do PC.

Pokud je první znak „N“, jedná se o požadavek na nastavení určité hodnoty na vstup RC článku. Z pomocné proměnné, ve které je uložena příchozí zpráva, se tak vezmou čísla následující za znakem „:“ a ta se nastaví na vstup RC článku.

K ukončení programu dojde až vypnutím, odpojením napájení.

3.2 KALIBRACE A/D A DAC PŘEVODNÍKU

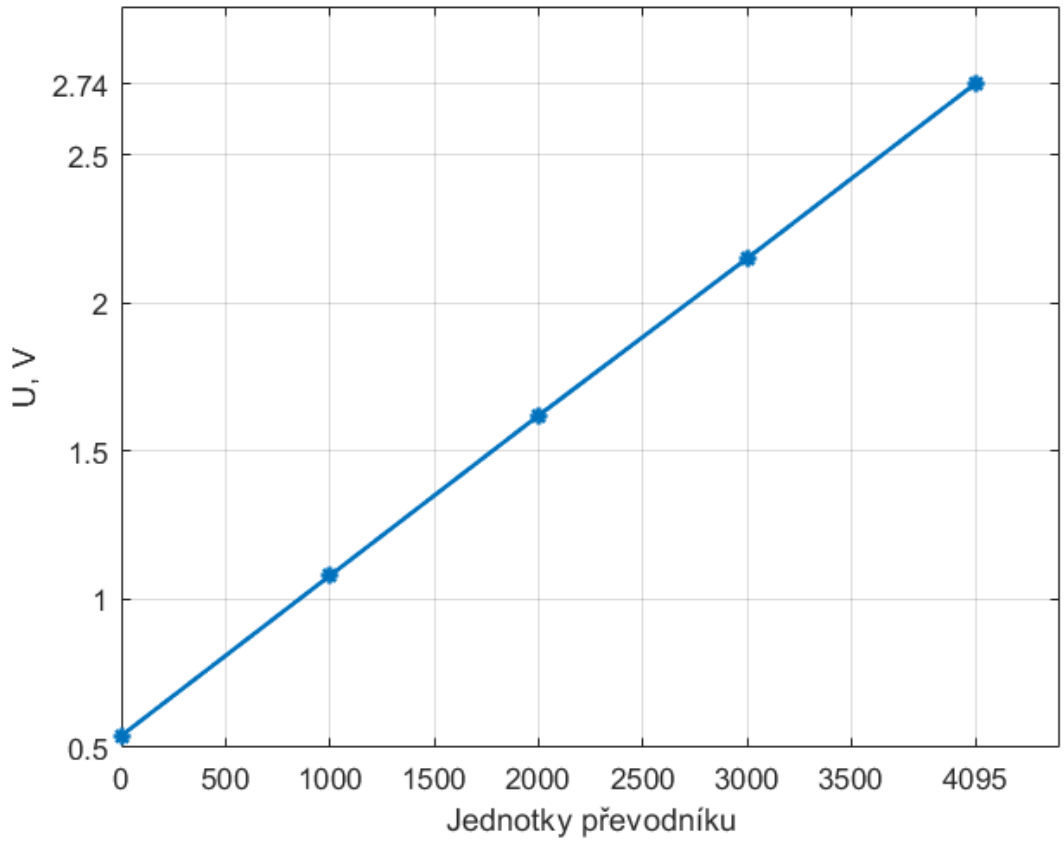
Bylo zapotřebí provést kalibraci obou převodníků na Arduinu, aby měřené a nastavované hodnoty napětí odpovídaly skutečnosti. Měření probíhalo v několika bodech, čímž byla ověřena linearita obou převodníků.

Kalibrace DAC převodníku probíhala experimentem, kdy se pomocí multimetru ověřovalo napětí, které bylo na pin nastavováno pomocí DAC převodníku z Arduina. Hodnoty nastavované na převodník a hodnoty změřené multimetrem jsou uvedené v tab. 3.1.

Tab. 3.1 – Kalibrace DAC převodníku

Napětí, V	0,54	1,08	1,62	2,15	2,74
Jednotky převodníku	0	1000	2000	3000	4095

Na obr. 3.2 je pak znázorněna kalibrační přímka DAC převodníku.



Obr. 3.2 – Kalibrační přímka DAC převodníku

Z tabulky je patrné, že napěťový rozsah převodníku je 0,54 − 2,74 = 2,2 V. Díky tomu můžeme psát rovnici (3.1) která se použije pro přepočet na napětí jako

$$U = \frac{2{,}2\,x}{4095} + 0{,}54, \qquad x \in \{0{,}1{,}2, \cdots, 4094{,}4095\}, \tag{3.1}$$

kde U – napětí, V,

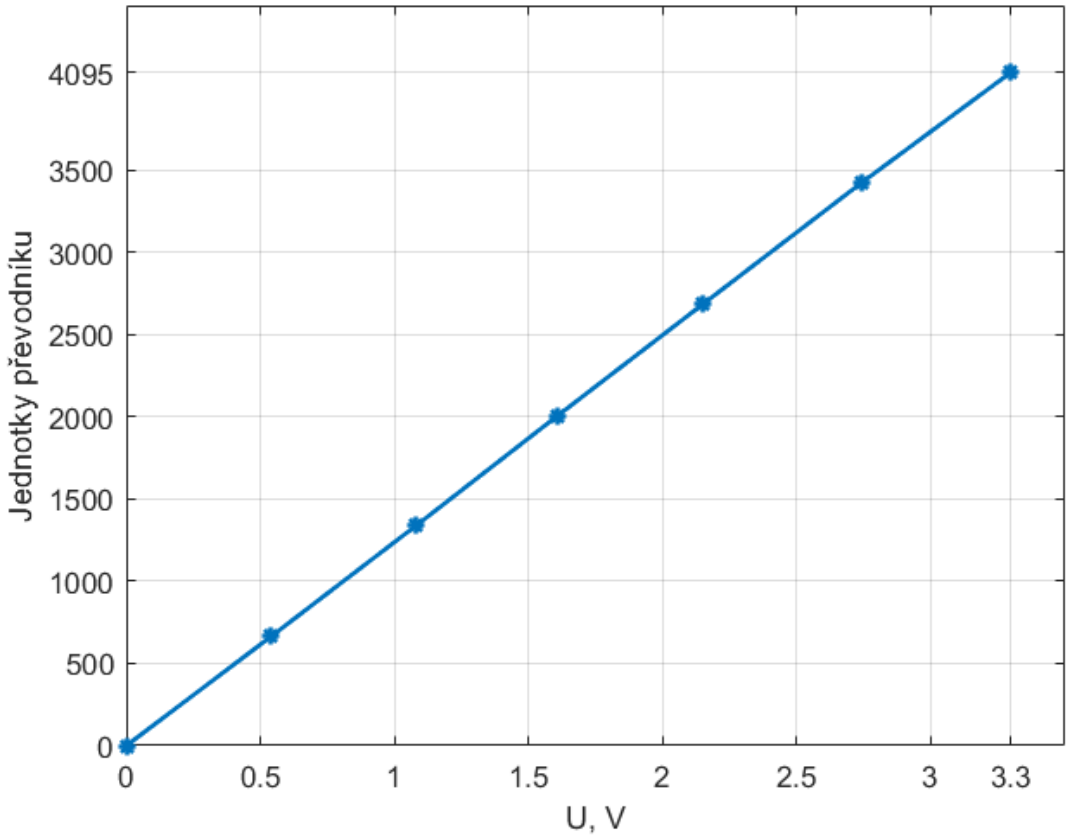
x – j.p (jednotky převodníku).

Kalibrace A/D převodníku probíhala rovněž experimentem, kdy se na pin Arduina přivedlo napětí, které bylo zároveň měřeno pomocí multimetru a na sériovém monitoru se zobrazoval počet jednotek převodníku, který charakterizoval napětí připojené na daný pin. Napětí, které bylo přivedeno na pin Arduina a tomu odpovídající počet jednotek převodníku, jsou znázorněny v tab. 3.2.

Tab. 3.2 – Kalibrace A/D převodníku

Napětí, V	0	0,54	1,08	1,61	2,15	2,74	3,3
Jednotky převodníku	0	664	1337	2008	2684	3420	4095

Na obr. 3.3 je zobrazena kalibrační přímka A/D převodníku.



Obr. 3.3 – Kalibrační přímka A/D převodníku

Nyní můžeme psát vzorec

$$U = \frac{3{,}3\,x}{4095}, \tag{3.2}$$

kde U – napětí, V,

x – j.p (jednotky převodníku).

Vzorec (3.2) pak lze použít pro přepočet na napětí ze znalosti hodnoty, kterou nám vrací A/D převodník.

4 PROGRAMOVÉ VYBAVENÍ – MATLAB

V následující kapitole bude popsán návrh a implementace programu pro identifikaci parametrů modelu, návrh včetně ověření estimátoru a regulátoru v prostředí MATLAB.

4.1 ZÍSKÁNÍ DAT PRO IDENTIFIKACI

V této kapitole bude popsán kód, který byl vyvinut pro získání experimentálních dat potřebných pro identifikaci parametrů modelu (2.13), tj. hodnot součástek. Kód je rozdělen do dvou hlavních částí. První část zajišťuje přípravu experimentu a nastavení komunikace. Druhá část pak zajišťuje samotný experiment pro získání dat. Naměřená data by měla obsahovat informaci zejména o chování dynamických (přechodových) pochodů soustavy, aby byla pro optimalizaci modelu využitelná. Pokud by změřená data obsahovala pouze průběhy po odeznění přechodových dějů, pak by byla pro identifikaci nepoužitelná.

K sestavení následujícího kódu byl kvůli své praktičnosti a uživatelské přívětivosti využit program MATLAB.

V první části kódu dochází k inicializaci proměnných, které budou potřeba pro získání dat. Jde především o vzorkovací periodu, vytvoření vstupního vektoru akční veličiny, jehož hodnoty budou odesílány do Arduina, a přípravu komunikace. Komunikace je inicializována voláním funkce *ArOpen()*.

```
function s=ArOpen(port)

s = serialport(port,115200);
pause(0.3);
configureTerminator(s,"CR/LF");
s.Timeout=1;
flush(s);
end
```

V této funkci dochází k nastavení přenosové rychlosti na 115200 bit/s, definování používaného portu COM, ukončovacího znaku, který byl nastaven na „CRLF“ a timeoutu, který byl stanoven na 1 s.

Poté, co se provede prvotní nastavení, je zahájena komunikace, kdy je do Arduina odeslán požadavek na nastavení nulové hodnoty na vstup a čeká se na ustálení, aby experiment a získaná data byla z ustáleného stavu. Způsob odeslání požadavku na změření dat, příjmu a dekódování zprávy pak ukazuje následující kód.

```
function AI=ArGetAI(s)
```

```
AI = [-1;-1];            % příznak chybné odpovědi
writeline(s,'A:');       % vyšli požadavek
r = char(readline(s));   % vyčti odpověď
if isempty(r)||~strcmp(r(1:2),"A:") % kontrola odpovědi
    return               % chyba, konec
end
c = strfind(r,',');      % poloha čárky
AI(1) = str2double(r(3:c-1));
AI(2) = str2double(r(c+1:end));
```

Následně je zahájen experiment, kdy po dobu sedmdesáti sekund s periodou vzorkování 0,1 s dochází k odeslání požadavku na změření dat, a následnému odeslání požadavku na nastavení hodnoty uvedené ve vstupním vektoru. Synchronizaci provádění kódu s periodou vzorkování pak zajišťuje následující část kódu.

```
tic
while (t(k)>toc), pause(0.001), end
```

Po ukončení experimentu dostaneme data, která znázorňují, jak reagovala soustava na vstupní signál a tato data použijeme pro optimalizaci.

4.2 IDENTIFIKACE PARAMETRŮ MODELU

V této kapitole se bude pracovat s výstupem soustavy s nominálními hodnotami součástek, výstupem změřeným na reálné soustavě a výstupem s kalibrovanými hodnotami součástek. Vzhledem k tomu, že nominální hodnoty součástek nesouhlasí s hodnotami skutečnými, které se od nich nepatrně liší, bylo třeba tyto hodnoty zkorigovat, čímž dojde k zpřesnění matematického modelu. Nejprve je potřeba načíst data, která byla získána z přechozího měření a která popisují chování reálné soustavy včetně přechodových jevů. Následně je z nominálních matic (nominálních proto, že jsou sestaveny z nominálních hodnot součástek) vytvořen v programu MATLAB LTI objekt a jsou dopočítány počáteční podmínky dle vztahu

$$\boldsymbol{x_0} = \boldsymbol{A}^{-1}\boldsymbol{B}u_{(0)}, \tag{4.1}$$

kde $\boldsymbol{x_0}$ – vektor počátečních podmínek,

$\boldsymbol{A}$ – matice systému,

$\boldsymbol{B}$ – matice buzení systému,

$u_{(0)}$ – akční veličina.

Aby mohl být použit vztah (4.1), musí soustava při měření vycházet z ustáleného stavu. Dále se pomocí příkazu *lsim* dopočítá výstup soustavy s nominálními maticemi. Příkaz *lsim* má jako vstupní parametry vektor času, počátečních podmínek, vstupu a řeší soustavu diferenciálních rovnic prvního řádu.

V samotné optimalizaci je potřeba optimalizovat devět parametrů z 24 celkem, kde těchto devět parametrů má fyzikální význam (hodnoty součástek) a mají vliv na výsledné matice popisující chování soustavy. Z matematického hlediska je vhodné, aby parametry byly, pokud možno, stejné alespoň řádově. V tomto případě tomu tak není, neboť hodnoty odporů se pohybují v tisících a hodnoty kondenzátorů v hodnotách kolem 10^{-6}. Z tohoto důvodu byl zaveden korekční vektor o devíti prvcích, kterým se jednotlivé hodnoty součástek násobí. Tím se docílilo sjednocení velikosti korigovaných parametrů. Optimalizuje se diskrétní model, který je popsán stavovými rovnicemi dle vztahu (1.6).

Pro optimalizaci byla zvolena funkce *fminsearch*. Jako kritérium byla zvolena metoda nejmenších čtverců, kde se porovnává chyba mezi změřeným výstupem soustavy (reálným) a výstupem spočítaným simulací. Kritérium je formulováno jako

$$J = (\boldsymbol{y_m} - \boldsymbol{y})^T(\boldsymbol{y_m} - \boldsymbol{y}), \tag{4.2}$$

kde $\boldsymbol{y_m}$ – vektor výstupních veličin změřený na reálné soustavě,

$\boldsymbol{y}$ – vektor výstupních veličin spočítaný simulací.

Z důvodu devíti parametrů korekčního vektoru dochází ke vzniku mnoha extrémů, z nichž funkce *fminsearch* nalezne pouze jeden. Funkce začne hledat z nominálních hodnot součástek (tedy korekční vektor je plný jedniček).

Kód pro optimalizaci pak vypadá následovně.

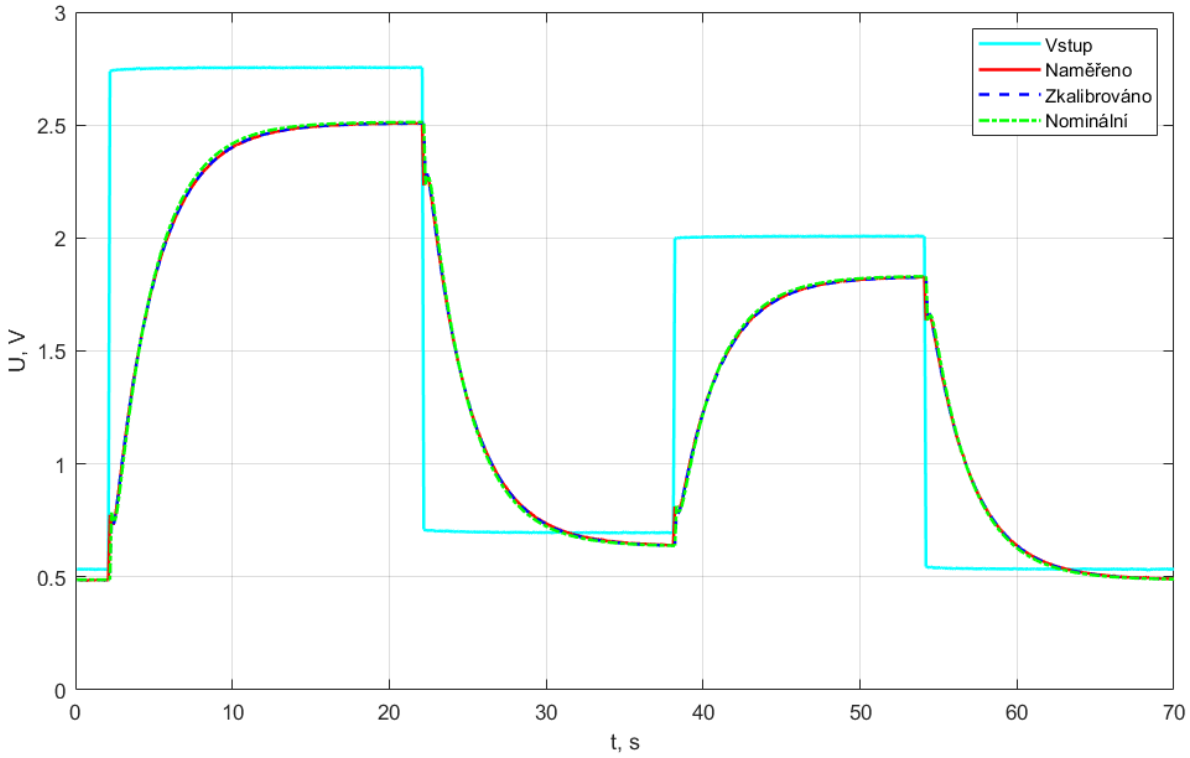
```
kor0 = ones(9,1);           % první koreční parametry
kor = fminsearch(@(kor) chyba_R5C4(kor,t,U,YM),kor0);
```

Kód pro vyhodnocení kritéria.

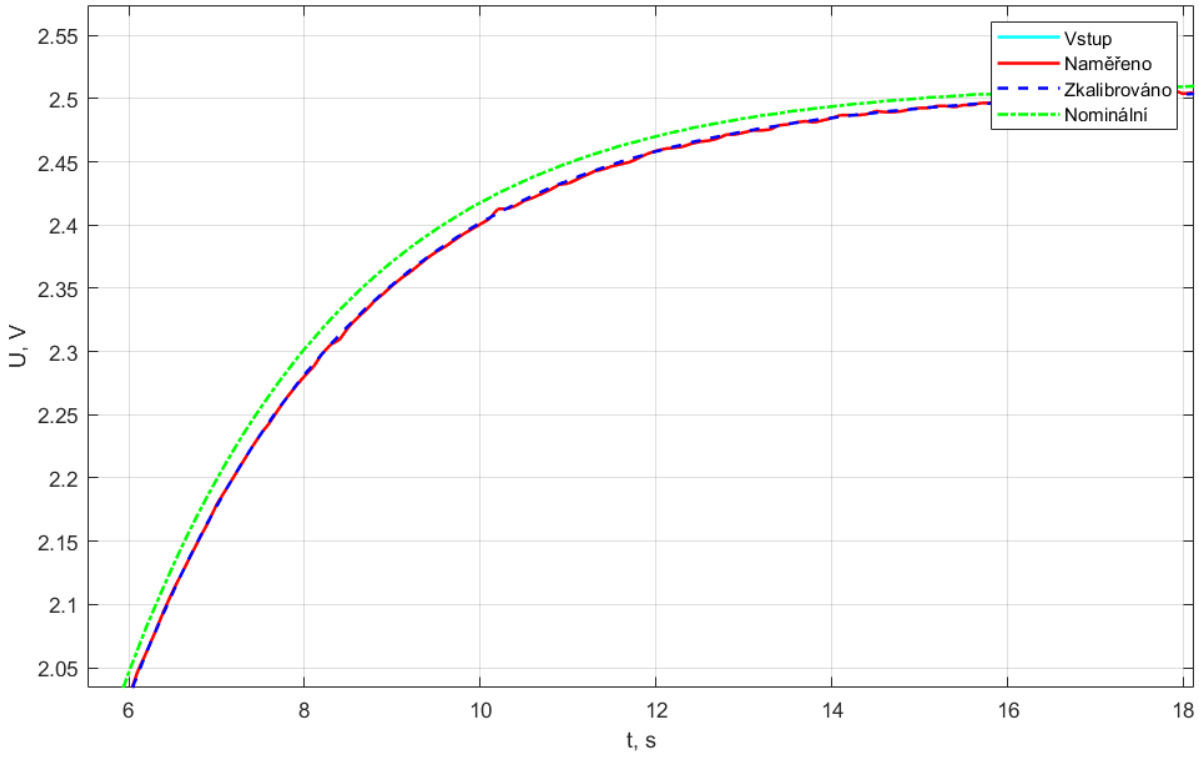
```
function [err] = chyba_R5C4(kor,t,UM,YM)


[A,B,C] = R5C4(kor);     % získání korigovaných matic
SS = ss(A,B,C,0);        % vytvoření LTI objektu
x0 = -A\B*UM(1);         % dopočet počátečních podmínek
Y = lsim(SS,UM,t,x0);
err = (YM-Y)'*(YM-Y);    % chyba
end
```

Po skončení optimalizace se opět vytvoří LTI objekt, tentokrát již z optimalizovaných matic. Znovu se spočítá vektor počátečních podmínek dle vztahu (4.1) a pomocí příkazu *lsim* se dopočítá průběh *Y* pro soustavu s optimalizovanými maticemi. Výsledek optimalizace a porovnání průběhů je zobrazen na obr. 4.1, detail, jak se průběhy lišily pak na obr. 4.2.



Obr. 4.1 – Porovnání průběhů před a po optimalizaci



Obr. 4.2 – Detail rozdílu před a po optimalizaci

4.3 NÁVRH A OVĚŘENÍ ESTIMÁTORU

Návrh a ověření estimátoru probíhalo v programu MATLAB. Vzhledem k tomu, že není možné měřit všechny stavy soustavy, byl pro odhad všech stavů použit diskrétní deterministický estimátor, jehož schéma zapojení je zobrazeno na obr. 1.6 a je popsán stavovým modelem (1.6). Návrh estimátoru vychází ze stavového modelu (1.20) s maticemi, které byly získány pomocí identifikace popsané v kapitole 4.2 . Následující část kódu popisuje způsob získání diskrétního stavového popisu.

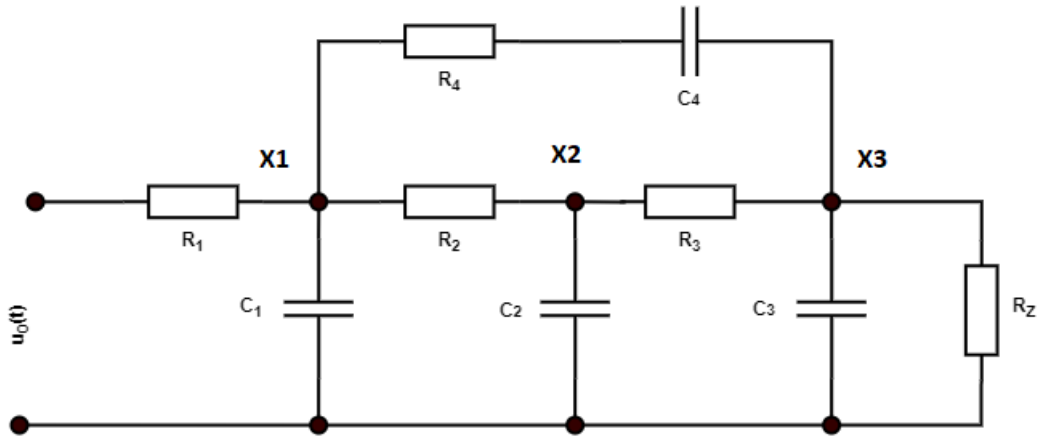
```
[A,B,C] = R5C4(kor);          % vrátí korigované matice A, B, C
FSS = c2d(ss(A,B,C,0),Ts);    % převod na diskrétní popis (LTI objekt)
[Ad,Bd,Cd,Dd] = ssdata(FSS); % získání diskrétních matic
```

Vektor zesílení $\boldsymbol{H_E}$ byl určen jako duální úloha k návrhu diskrétního LQ regulátoru, tj. se stejnými hodnotami matic $\boldsymbol{Q}$ a $\boldsymbol{R}$. Pro konkrétní řešení této úlohy byla použita funkce MATLABu s modifikovanými parametry pro duální návrh.

```
[He,~,~] = dlqr(Ad',Cd',Q,R); He = He';
```

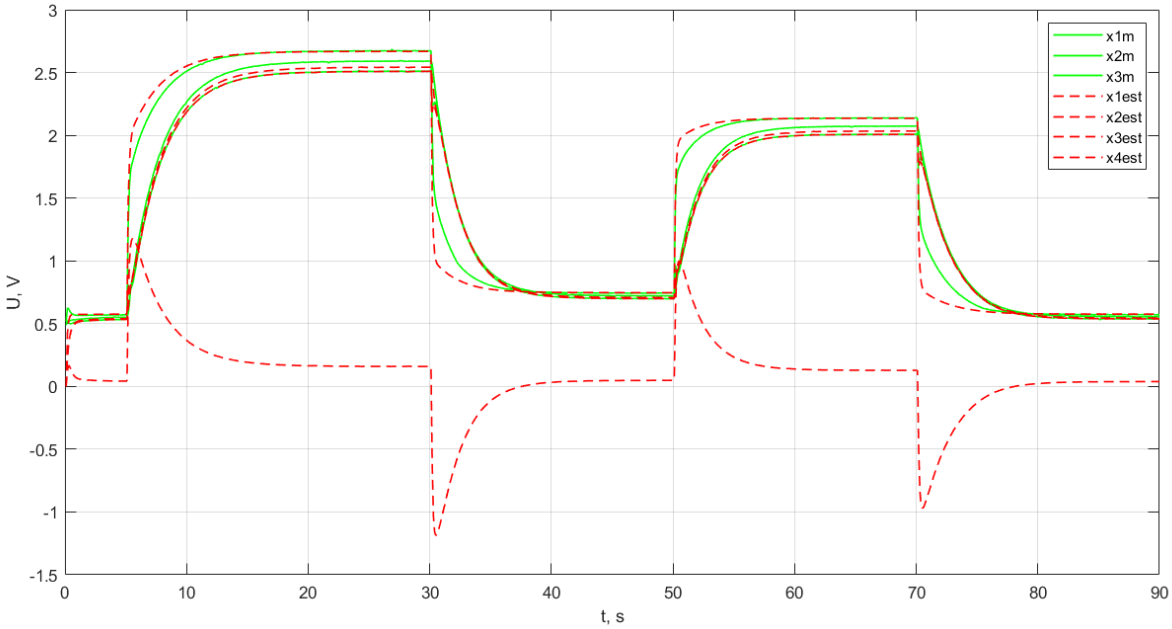
Hodnoty parametrů regulátoru $\boldsymbol{Q}$ a $\boldsymbol{R}$, které byly pro návrh regulátoru použity, byly nastaveny na hodnotu $\boldsymbol{R} = 2$ a $\boldsymbol{Q}$ pak byla jednotková matice o rozměrech 4×4. Abychom mohli

porovnat a ověřit funkčnost estimace, je třeba měřit vnitřní stavy na reálném modelu a ty následně porovnat s těmi, které počítá estimátor. Bylo tedy potřeba drobně modifikovat kód v Arduinu tak, aby měřil tři ze čtyř vnitřních stavů (čtvrtý stav měřit nelze).



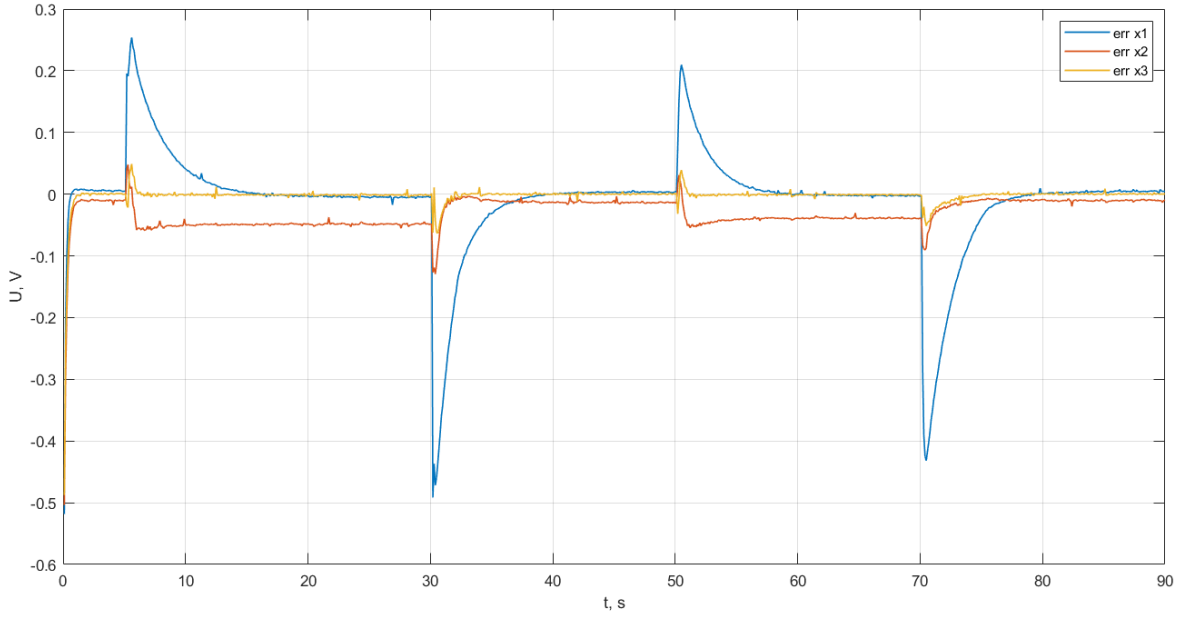
Obr. 4.3 – Měření vnitřních stavů

Ověření funkčnosti takto navrženého estimátoru bylo provedeno experimentálně porovnáním průběhů tří měřených napětí (třech stavových veličin ze čtyř celkem) s odpovídajícími průběhy estimovaných stavů pro stejný průběh vstupního napětí. Měření průběhu napětí začínalo z počátečního nenulového ustáleného stavu a počáteční stav estimátoru byl nulový. Perioda vzorkování byla 0,1 s. Konkrétní měřená napětí jsou označena ve schématu na obr. 4.3 a porovnání průběhů je znázorněno na obr. 4.4.



Obr. 4.4 – Porovnání stavů estimovaných a měřených

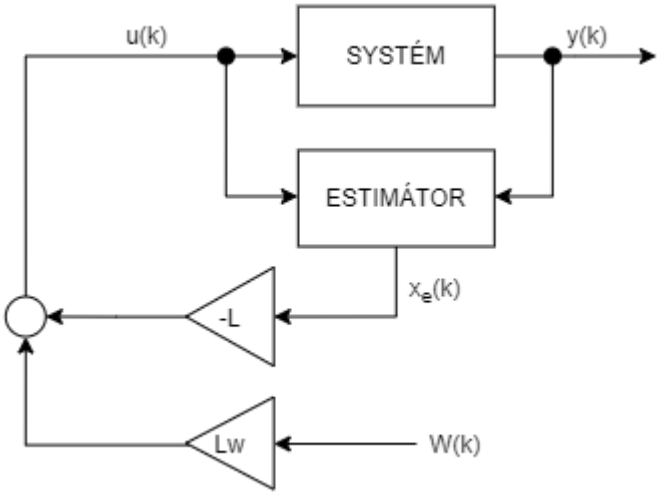
Z obr. 4.4 je patrné, že stavy estimované relativně přesně odpovídají stavům měřeným a funkce estimátoru tak byla ověřena. Na obr. 4.5 je pak zobrazena chyba estimace.



Obr. 4.5 – Chyba estimace

4.4 NÁVRH A OVĚŘENÍ LQ REGULÁTORU

Návrh a ověření regulátoru bude provedeno simulačně v programu MATLAB. Jako regulátor byl použit LQ regulátor s asymptotickým sledováním žádané hodnoty. Blokové schéma regulátoru s asymptotickým sledováním žádané hodnoty a estimátoru je zobrazeno na obr. 4.6.



Obr. 4.6 – Blokové schéma regulátoru s estimátorem

Systém je popsán soustavou rovnic dle vztahu (2.12) a akční zásah se vypočte dle vztahu (4.3).

$$u = -\boldsymbol{L}\hat{\boldsymbol{x}} + L_W w \,, \tag{4.3}$$

kde u – akční zásah,

$\hat{\boldsymbol{x}}$ – estimovaný vektor stavových veličin,

$\boldsymbol{L}$, – matice zpětné vazby regulátoru,

w – žádaná hodnota,

L_W – zesílení žádané hodnoty.

V MATLABu je pak zapotřebí určit hodnotu vektoru zesílení $\boldsymbol{L}$, k čemuž bude použit příkaz *dlqr*.

```
[L,~,~] = dlqr(Ad,Bd,Q,R);
```

Pro výpočet akčního zásahu je třeba znát ještě parametr L_W, ten se vypočte pomocí vztahu (4.4) jež byl odvozen v teoretické části.

$$L_W = [-\boldsymbol{C}(\boldsymbol{A} - \boldsymbol{BL})^{-1}\boldsymbol{B}]^{-1}, \tag{4.4}$$

kde L_W – zesílení žádané hodnoty,

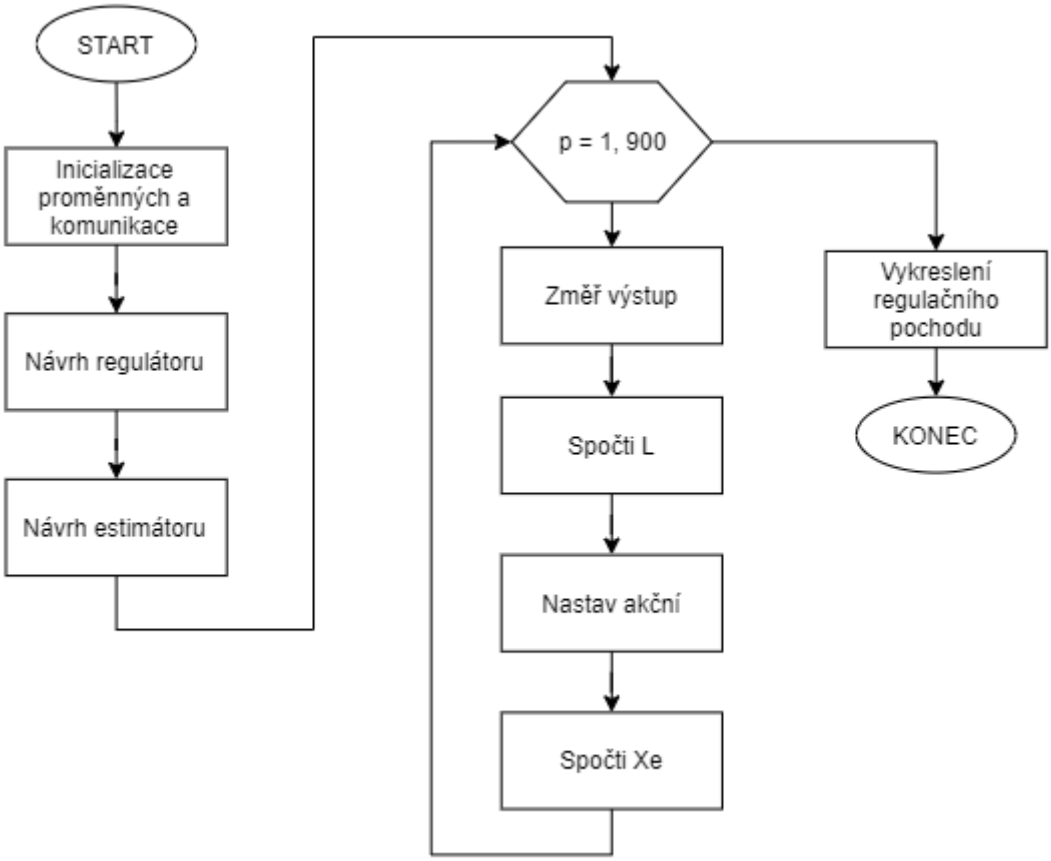
$\boldsymbol{C}$ – výstupní matice systému,

$\boldsymbol{A}$ – matice systému,

$\boldsymbol{B}$ – matice buzení systému,

$\boldsymbol{L}$ – matice zpětné vazby regulátoru.

Princip návrhu a ověření funkce regulátoru zobrazuje vývojový diagram na obr. 4.7.



Obr. 4.7 – Vývojový diagram návrhu a ověření regulace

Po spuštění programu dojde k inicializaci proměnných a komunikace, dojde k ustálení soustavy a následně se pomocí funkce *dlqr* navrhne zesílení regulátoru a stejným způsobem, jen s jinými vstupními parametry, navrhneme zesílení estimátoru (viz kapitola 4.3).

```
[L,~,~] = dlqr(Ad,Bd,Q,R);   % návrh regulátoru
[He,~,~] = dlqr(Ad',Cd',Q,R);  % návrh estimátoru (duální úloha)
He = He';
```

Parametry $\boldsymbol{R}$ a $\boldsymbol{Q}$ byly nastaveny jako $\boldsymbol{R} = 2$ a $\boldsymbol{Q}$ odpovídá jednotkové matici o rozměru 4×4. Následně je potřeba vypočítat ustálený stav x.

```
% výpočet ustáleného stavu x pro ustálenou hodnotu u
pom = (eye(nx)-Ad)\Bd;
u0 = W(1)/(C*pom);    % ustálený vstup k žádané hodnotě
x = pom*u0;           % ustálený stav k žádané hodnotě
```

Poté již probíhá experiment, kdy se s vzorkovací periodou 0,1 s odesílá požadavek na změření dat.

```
AI = ArGetAI(s);
```

Dochází k výpočtu akční veličiny včetně kontroly, aby se nepřekročila maximální nebo minimální hodnota akční veličiny.

```
u = -L*xe+Lw*W(k);
if(u>2.74) u=2.74; end
if(u<0) u=0; end
```

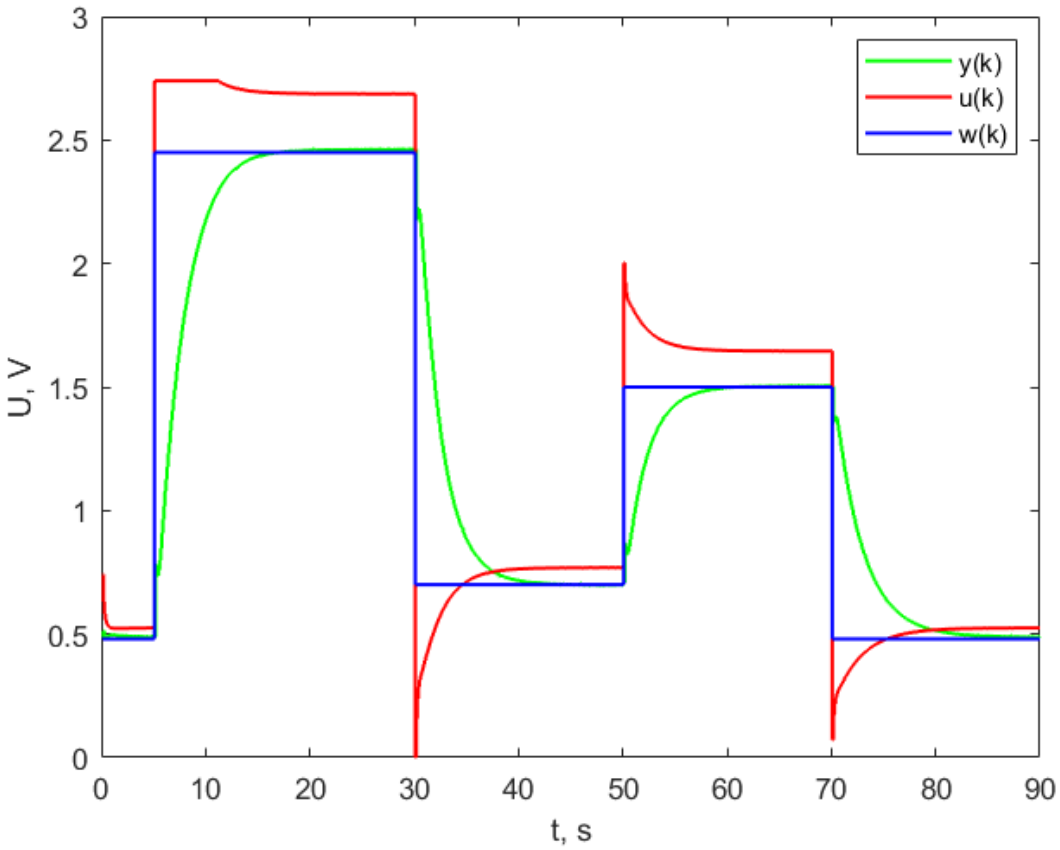
Nyní se vypočítaný akční zásah převede na jednotky převodníku dle vztahu (3.1) a odešle se do Arduina

```
nastav = ((u-0.54)*4095)/2.2;
nastav = round(nastav);
AO=ArSetAO(s,nastav);
```

V poslední řadě dochází k výpočtu následujícího (estimovaného) stavu

```
xe = Ae*xe+Bd*u+He*y;
```

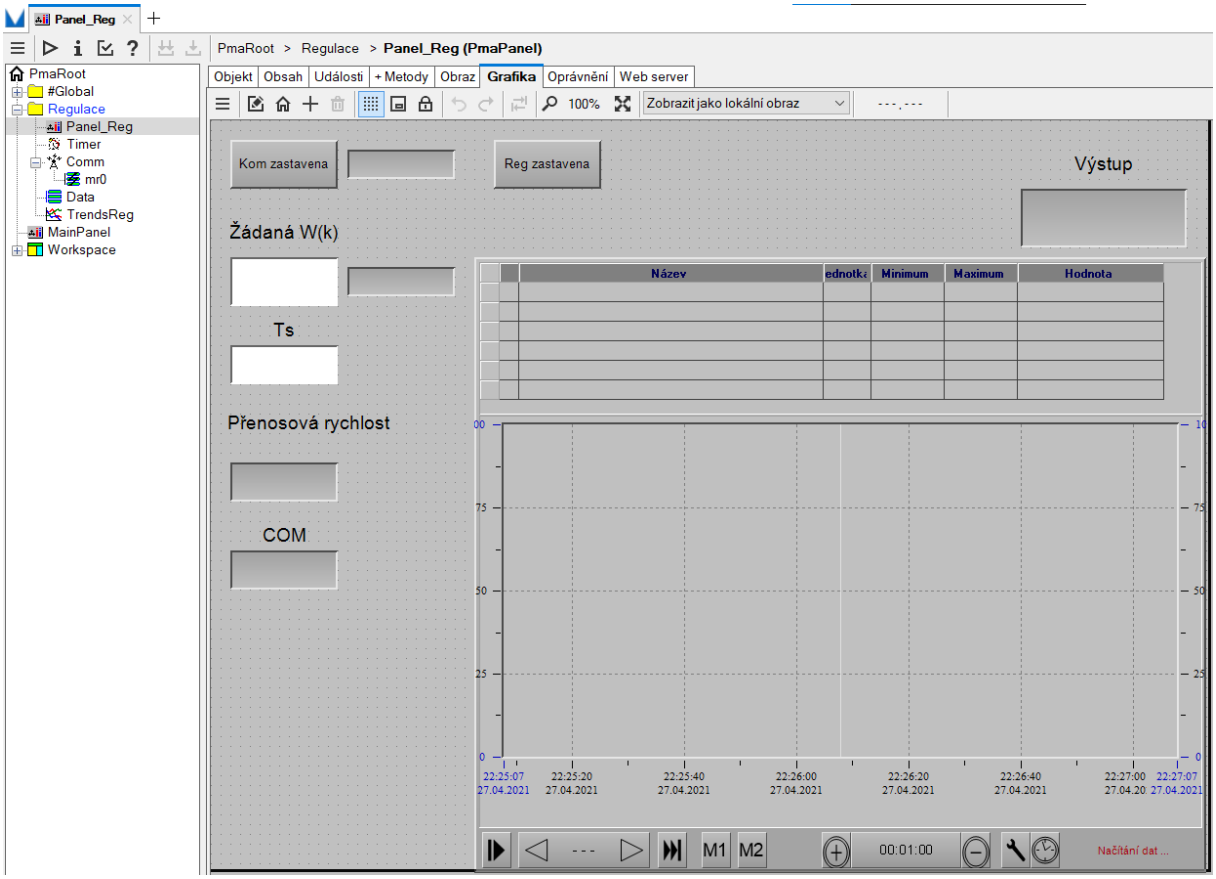
Na obr. 4.8 je pak znázorněn regulační pochod pro ověření funkce regulátoru.



Obr. 4.8 – Regulační pochod pro ověření funkce LQ regulátoru

4.5 IMPLEMENTACE LQ REGULÁTORU V PROSTŘEDÍ PROMOTIC

Výsledná aplikace pro regulaci zadané soustavy byla napsána v programu PROMOTIC. Vzhled grafické části práce je zobrazen na obr. 4.9.



Obr. 4.9 – Aplikace v programu PROMOTIC

V levé části lze pak vidět strom Pma objektů, které tvoří aplikaci. Všechny Pma objekty byly pro lepší přehlednost umístěny do jedné složky.

Panel pro regulaci obsahuje několik ovládacích prvků. Těmi jsou dvě tlačítka pro zapínání a vypínání komunikace (měření dat) a regulace. Pro kontrolu, zda tyto procesy běží, byla do programu přidána dvě pole s textem, kam se vypisuje počet odeslaných požadavků na změření dat (pole vedle tlačítka povolujícího komunikaci) a pole pro výpis počtu odeslaných požadavku s nastavením konkrétní hodnoty na vstup soustavy. Dále program zobrazuje aktuální přenosovou rychlost a aktuálně používaný komunikační port. Vpravo nahoře je pak umístěno okno, kde je zobrazován aktuální výstup soustavy, pod kterým se nachází okno pro zobrazování historie průběhu (trend). Jako editovatelné položky byla nastavena okna pro zadávání žádané hodnoty a periody, s jakou je komunikace a regulace prováděna.

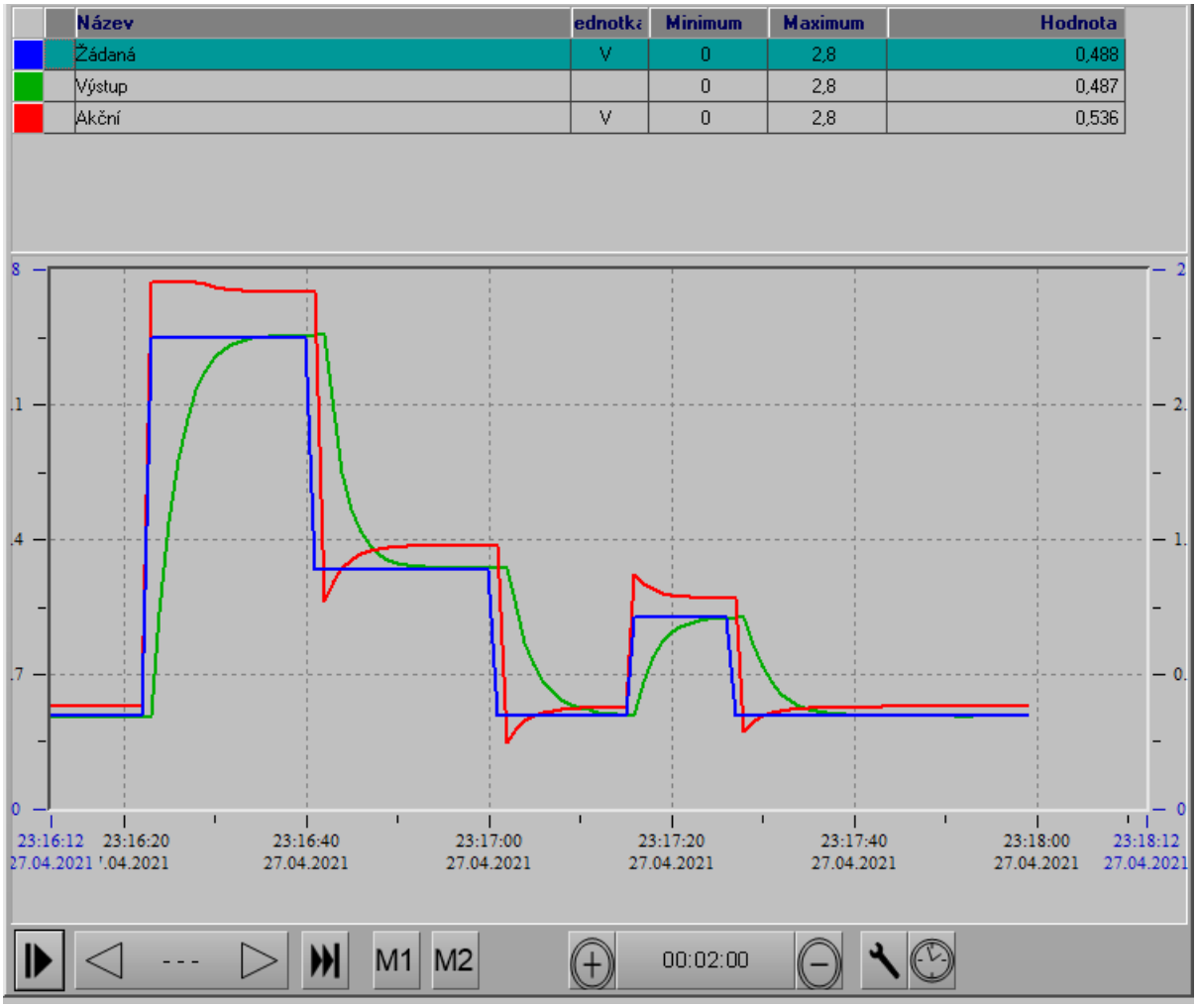
Timer je časovač, u kterého je v definované periodě vyvolána událost „onTic“, která zajišťuje pravidelné provádění regulace a komunikace. Dalším prvkem je komunikační ovladač PmChar, což je uživatelsky konfigurovatelný ASCII/BIN protokol. V tomto objektu byly nastaveny veškeré parametry komunikace. Typ komunikace byl nastaven na „sériová

komunikace“ s přenosovou rychlostí 115000 bps a využívaným sériovým portem „COM5“. Počet datových bitů byl nastaven na osm a ovladač očekává 1 stop bit. Znak označující konec zprávy byl nastaven na „LF“. Aplikace figuruje v roli mastera a je tak iniciátorem komunikace, na který slave (arduino) reaguje a odpovídá. Předposlední položkou jsou data, kde jsou proměnné používané v aplikaci a potřebné pro regulaci. Zde jsou uložené hodnoty matic, které byly získány v předchozích kapitolách této práce. Posledním objektem je trends, který zajišťuje ukládání historie průběhů.

V objektu timer byla vytvořena metoda, která je volána pokaždé, když je vyvolána událost „OnTic“ časovačem. Tato metoda se zeptá, zda je komunikace připravena a pokud ano, uloží do karty „Data-zasílání“ požadavek na změření dat, který odešle. V komunikaci v objektu PmaCommMsg jsou pak definované dvě události („onDataRecive“, „OnEndOfTransfer“) První událost je vyvolána po úspěšném přijetí dat a jejich uložení do karty „Data příjem“. Zde pak dochází ke kontrole přijatých dat, a pokud obsahují odpověď na požadavek pro změření dat, dojde k převedení dat z jednotek převodníku na napětí dle vztahu (3.2) a následnému uložení do proměnné v datech. Druhá zmíněná událost je vyvolána po ukončení komunikace. V této události jsou postupně volány metody zajišťující regulaci.

Nejprve je volána metoda, která počítá akční zásah dle vzorce (1.12), který byl odvozen v teoretické části. Následuje metoda, která má za úkol odeslat požadavek na nastavení akčního zásahu do Arduina. V této metodě dojde k přepočítání hodnoty akčního zásahu na jednotky převodníku dle vzorce (3.1) a po ověření, že komunikace není obsazena, dochází k odeslání této hodnoty. Poslední funkce má za úkol spočítat následující (estimovaný) stav systému. Zde byl použit vzorec (1.23), který byl rovněž odvozen v teoretické části.

Ukázka regulačního pochodu je znázorněna na obr. 4.10.



Obr. 4.10 – Regulační pochod

Perioda vzorkování byla zvolena jako 1 s. Vzhledem k tomu, že doba do ustálení je pro tuto soustavu přibližně 15 s, je taková vzorkovací perioda dostačující. Jako minimální vzorkovací perioda byl stanoven čas 0,3 s. Čas potřebný pro odeslání jedné zprávy a odeslání požadavku na nastavení byl změřen jako 0,25 s. Z fyzických vlastností použité desky arduino Due plyne omezení ve formě minimální a maximální hodnoty, kterých lze na výstupu docílit. Maximální hodnota výstupu je 2,51 V a minimální hodnota, která se dá nastavit na výstup, je 0,488 V.

5 ZÁVĚR

Cílem práce bylo vytvořit program pro PC v prostředí PROMOTIC, který realizuje řízení lineární SISO systému pomocí LQ regulátoru s asymptotickým sledováním žádané hodnoty, který byl připojen pomocí Arduino Due. Zhotovený program dokáže spolehlivě regulovat připojenou soustavu bez výskytu trvalé regulační odchylky a nežádoucích překmitů.

V teoretické části byla popsána problematika potřebná k realizaci uvedeného regulátoru. Byla popsána potřebná teorie pro stavový popis systému a jeho následnou identifikaci včetně postupu odvození pro návrh diskrétního LQ regulátoru s asymtotickým sledováním žádané hodnoty a diskrétního Luenbergerova estimátoru.

V praktické části pak byly popsány jednotlivé části kódu realizující optimalizaci stavového popisu a návrhy regulátoru s estimátorem, u kterých byla následně ověřena jejich funkčnost. V poslední části práce je pak popsána aplikace v prostředí PROMOTIC, v které byl navržený regulátor s estimátorem implementován.

V průběhu vývoje a testování navrženého řízení se ukázalo, že u diskrétních regulátorů nemá délka periody vzorkování výrazný vliv na kvalitu řízení. Pokud byla doba do ustálení soustavy přibližně 15 s a perioda vzorkování byla změněna z 1 s na 0,5 s, nebo naopak na 2 s, tak se kvalita regulačního pochodu výrazně nesnížila. Z měření na soustavě také vyplynulo, že na soustavu nepůsobí parazitní šum, a byl tak místo Kalmánova estimátoru použit deterministický estimátor.

Jednou z možností vylepšení by mohla být možnost přidat nastavení parametrů komunikace (přenosové rychlosti a portu), případně možnosti měnit parametry regulátoru, což by vyžadovalo do programu přidat výpočty ovlivňující návrh regulátoru a estimátou, které jsou již v programu MATLAB předdefinovány.

POUŽITÁ LITERATURA

MACHEK, Ondřej, 2019. *LQ regulace systému motor generátor*. Pardubice. Diplomová práce. Univerzita Pardubice. Vedoucí práce F. Dušek.

SVATOŠ, Radovan, 2015. *Syntéza regulačního obvodu se stavovým regulátorem*. Pardubice. Dostupné také z: https://dk.upce.cz/bitstream/handle/10195/60072/Svato%C5%A1R_Stavov%C3%BDRegul%C3%A1tor_LK_2015.pdf?sequence=3&isAllowed=y. Diplomová práce. Univerzita Pardubice. Vedoucí práce L. Kupka.

DUŠEK, František, 2021. *Osobní konzultace*. Pardubice.

KUPKA, Libor, 2020. *Diskrétní řízení*. Osobní sdělení. Pardubice.

Obsah dokumentace PROMOTIC. *SCADA/HMI systém PROMOTIC* [online]. Ostrava: MICROSYS, spol. s.r.o [cit. 2021-5-2]. Dostupné z: https://www.promotic.eu/cz/pmdoc/PmDocDefault.htm

Arduino - Home [online], 2021. [cit. 2021-5-2]. Dostupné z: https://www.arduino.cc/

BALÁTĚ, Jaroslav, 2004. *Automatické řízení*. 2., přeprac. vyd. Praha: BEN - technická literatura. ISBN 80-730-0148-9.

STREJC, Vladimír, 1978. *Stavová teorie lineárního diskrétního řízení*. Praha: Academia.

Arduino DUE, 2021. In: Arduino [online]. Arduino.cc, 2021 [cit. 2021-3-27]. Dostupné z: https://store.arduino.cc/arduino-due

PŘÍLOHY

A – CD

Příloha A

Příloha k diplomové práci

Implementace diskrétního LQG regulátoru s využitím SCADA systému PROMOTIC

Bc. Daniel Opršal

CD

Obsah

1 Text diplomové práce ve formátu PDF.

2 Úplný zdrojový kód aplikace v programu PROMOTIC, zdrojový kód pro měření a generování napěťových signálů v Arduinu a zdrojový kód pro ověření návrhu modelu LQ regulátoru v programu MATLAB.