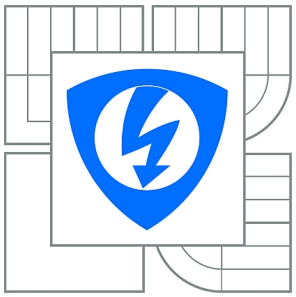
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

IMPLEMENTACE ALGORITMŮ ŘÍZENÍ ELEKTRICKÝCH MOTORŮ V SYSTÉMECH COMPACTRIO

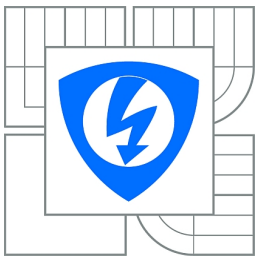
IMPLEMENTATION OF ELECTRIC MOTOR CONTROL ALGORITMS IN COMPACTRIO SYSTEMS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. GERGELY GLÉBA
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. PETR BLAHA, Ph.D.
SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Gergely Gléba	***ID:***	115172
Ročník:	2	***Akademický rok:***	2012/2013

NÁZEV TÉMATU:

Implementace algoritmů řízení elektrických motorů v systémech CompactRIO

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je seznámit se s existujícími algoritmy řízení elektrických střídavých motorů v prostředí Matlab Simulink a jejich následný převod do prostředí LabView Control Design and Simulation.
Výsledkem práce bude:
- model synchronního motoru s permanentními magnety
- model asynchronního motoru (naprogramovaný v jazyce C s použitím EMI funkcí)
- algoritmus vektorového řízení motoru s permanentními magnety
- algoritmus vektorového řízení asynchronního motoru (včetně jednoduchého estimátoru polohy magnetického toku)
- srovnání simulačních nástrojů Matlab a LabView z hlediska rychlosti výpočtu, složitosti implementace v obou prostředích a přehlednosti vytvořených simulačních schémat.

DOPORUČENÁ LITERATURA:

[1] Žídek J.: Grafické programování ve vývojovém prostředí Labview, VŠB-TU Ostrava, říjen 2002.
[2] Neborák, I.: Modelování a simulace elektrických regulovaných pohonů. Vysoká škola báňská - Technická univerzita Ostrava, Fakulta elektrotechniky a informatiky, 2002.
Další dle doporučení vedoucího.

Termín zadání:	11.2.2013	***Termín odevzdání:***	20.5.2013

Vedoucí práce: doc. Ing. Petr Blaha, Ph.D.
Konzultanti diplomové práce:

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

Abstrakt

Diplomová práca sa zaoberá implementáciou existujúcich algoritmov riadenia elektrických striedavých motorov v prostredí Matlab Simulink do prostredia LabVIEW Control Design and Simulation. Prvá časť práce sa zaoberá odvodením matematického modelu synchrónneho motora, asynchrónneho motora s permanentnými magnetmi, a teóriu vektorového riadenia elektrických striedavých motorov. V ďalšej časti prace je stručný návod na vytvorenie simulačných algoritmov v prostredí LabVIEW a na vytvorenie modelu motora pomocou EMI funkcií v jazyku C. V poslednej časti práce sú porovnané simulačné nástroje Matlab Simulink a LabVIEW.

Kľúčové slová

Transformácia podľa Clarkovej, Parkova transformácia, modulácia vektorového priestoru, vektorové riadenie, LabVIEW Control Design &Simulation, EMI funkcie.

Abstract

This master's thesis deals with the implementation of existing algorithms of electric AC motor control in Matlab Simulink to LabVIEW Control Design and Simulation. The first part of the thesis treats with the deduction of a mathematical model of an asynchronous motor, permanent magnet synchronous motor and with the theory of vector control of AC motors. In the next part there is a brief guide to create simulation algorithms in LabVIEW environment and to create the model of a motor with EMI functions in language C. The last part contains the comparison of Matlab Simulink and LabVIEW simulation tools.

Keywords

Clarke's transform, Park's transform, Space Vector Modulation, Vector control, LabVIEW Control Design &Simulation, EMI functions.

Bibliografická citace:

GLÉBA, G. *Implementace algoritmů řízení elektrických motorů v systémech CompactRIO*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2013. 89 s. Vedoucí diplomové práce
doc. Ing. Petr Blaha, Ph.D.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Implementace algoritmů řízení elektrických motorů v systémech CompactRIO jsem vypracoval samostatně pod vedením vedoucího diplomové 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 diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **16. května 2013** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Petr Blaha, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **16. května 2013** ……

podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam symbolov a skratiek

Značka	Veličina	Značka jednotky
$\boldsymbol{i}$	vektor prúdu	A
i	zložka prúdu	A
J	moment zotrvačnosti	kg m^2
L	indukčnosť	H
L_m	magnetická indukčnosť	H
L_l	rozptylová indukčnosť	H
M	moment motora	Nm
M_e	elektromagnetický moment	Nm
M_z	moment záťaže	Nm
p	operátor v Laplaceovej transformácii	-
R	odpor	Ω
s	sklz	-
T_s	perióda vzorkovania	s
$\boldsymbol{u}$	vektor napätia	V
u	zložka napätia	V
z_p	počet pólových dvojíc	-
$\boldsymbol{\Psi}$	vektor magnetického toku	Wb
Ψ	zložka magnetického toku	Wb
σ	činiteľ celkového rozptylu	-
ω	elektrická uhlová rýchlosť	rad/s
ω_m	mechanická uhlová rýchlosť	rad/s
ω_o	uhlová rýchlosť v obecnom súradnicovom systéme	rad/s

Dolné indexy

A,B,C	zložky fázovej premennej
c	celkový
d,q	premenná v súradniciach vektoru magnetického toku
f	budenie
i	indukované
o	v obecnom súradnicovom systéme
r	premenná rotora
ref	referenčná hodnota premennej
s	premenná statora
α,β	premenná v stojacich súradniciach

Skratky

ACIM	AC Induction Motor
CB	Callback
DLL	Dynamic – Link Library
EMI	External Model Interface
FOC	Field Oriented Control
ODE	Ordinary Differential Equation
PMSM	Permanent Magnet Synchronous Motor
VI	Virtual Instrument

1 ÚVOD

Na modelovanie dynamických systémov v našej fakulte sa prevažne používa prostredie Matlab Simulink od firmy MathWorks. Okrem tohto modelu existuje menej zavedené programovacie a vývojové prostredie LabView od firmy National Instruments. Pôvodne LabView bol určený na meranie, analýzu a zber dát. Počas vývoja softvér bol dovedený k vysokej mieri dokonalosti. Boli doplnené nadstavby ktoré umožňujú využitie aj v oblasti modelovania, identifikácie a regulovanie.

1.1 LabVIEW

Počas vývoja LabVIEW (Laboratory Virtual Instruments Engineering Workbench) základným podnetom bola predstava, aby technik, ktorý je schopný svoje predstavy a myšlienky vyjadriť v blokovej schéme, mohol tieto poznatky súčasne zapísať aj do programu. Toto grafické prevedenie umožňuje rýchlejšiu tvorbu programov ako klasické textové programovanie. Tento program môžeme nazývať taktiež ako grafický jazyk, ktorý je vhodný nielen k programovaniu systémov pre riadenie, meranie a analýzu ale aj pre vizualizáciu týchto postupov. Umožňuje nahradiť časovo a finančne náročné a zložité technické prostriedky virtuálnym vyjadrením za pomoci programových prostriedkov. Prepojením LabVIEW s CompactRIO môžeme dostať komplexný riadiaci systém.

Cieľom práce je oboznámiť sa s vývojovým prostredím LabView, vyskúšať si jeho možnosti v oblasti riadenia a modelovania motorov.

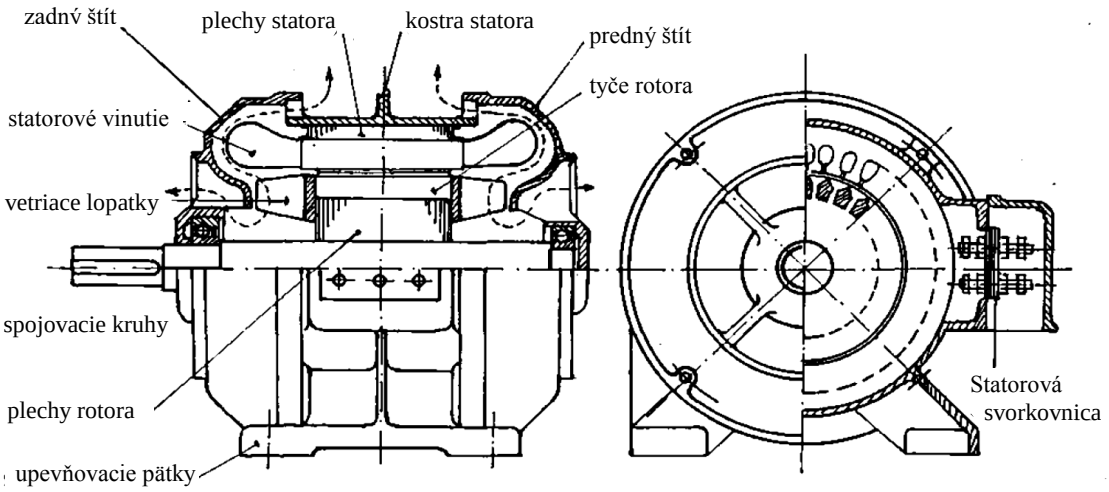
V súčasnej dobe neexistuje slovenská, respektíve česká lokalizácia LabVIEW preto sú tohto textu ponechané niektoré výrazy originálnej podobe alebo sú len voľne preložené.

2 ASYNCHRÓNNE STROJE

Asynchrónny stroj patrí do veľkej skupiny elektrických strojov, umožňujúci elektromechanickú premenu energie. Delíme ich na motory a generátory. Pri motoroch dochádza k premene elektrickej energie na mechanickú energiu a pri generátoroch, mechanická energia na elektrickú energiu.

2.1 Prevedenie asynchrónneho stroja

Ako všetky zariadenia, slúžiace k elektromechanickej premene energií, pozostáva asynchrónny stroj z pevnej časti (statoru) a pohyblivej časti, ktorá sa u strojov s otáčavým pohybom nazýva rotor. Stator sa skladá z liatinovej konštrukcie a dvoj ložiskových štítov. V kostre statora sú zalisované plechy, ktoré sú navzájom izolované a tvoria časť magnetického obvodu stroja. Rotorové plechy sú nalisované na hriadeli, ktorá sa otáča v ložiskách, upevnených v ložiskových štítoch, ktoré vymedzujú polohu rotora vo vnútri statoru. Medzi statorom a rotorom je vzduchová medzera, ktorá umožňuje pohyb rotora. Celkové usporiadanie asynchrónneho stroja je na obr.1



Obr. 1 Usporiadanie hlavných časti asynchrónneho motora [4]

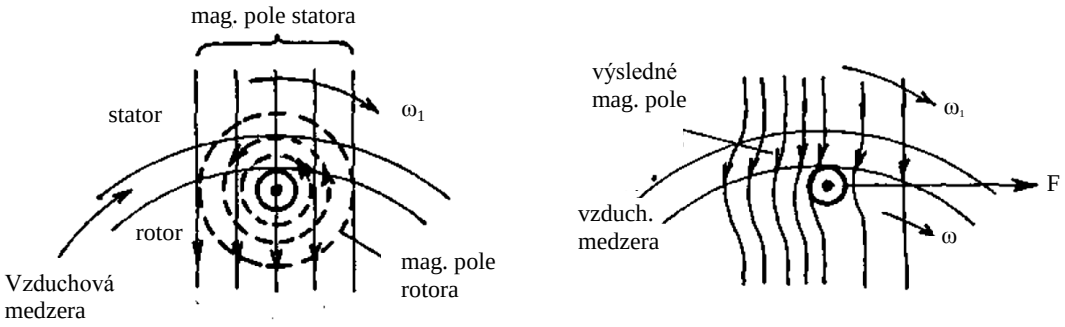
V drážkach statorových a rotorových plechov je uložené vinutie stroja. Na statore býva obvykle trojfázové vinutie (ale také jedno a dvojfázové), jeho začiatky a konce sú vyvedené na svorkovnicu. V rotorových drážkach je uložené vinutie, ktoré sa nazýva kotva. U motorov s kotvou nakrátko sú v drážkach rotora neizolované medené, alebo mosadzné, najčastejšie však hliníkové spojovacie kruhy nakrátko. U motorov s menšími výkonmi sa vinutie odlieva spolu s vetracími lopatkami z hliníka, metódou tlakového liatia. Takému vinutí sa hovorí klietka. U motorov s vinutým rotorom a krúžkom je v drážkach rotora uložené trojfázové vinutie z izolovaných vodičov. Rotorové vinutie,

ktoré je pripojené na tri zberné krúžky umiestnené na hriadeli a cez zberné ústrojenstvo (uhlíkové kefy) je napájané striedavým prúdom.

Text kapitoly bol vytvorený podľa [5]

2.2 Princíp činnosti

Najrozšírenejším typom asynchrónneho stroja je trojfázový asynchrónny motor, jeho statorové vinutie je pripojené na trojfázovú sieť a jeho pomocou vytvorí točivé magnetické pole, ktoré pretína vodiče rotora (obr. 2), indukuje v nich napätie a v prípade, že vinutie rotora je uzavreté, preteká ním prúd, ktorý vytvára magnetické pole rotora. Vzájomným pôsobením magnetických polí vzniká sila, pôsobiaca na vodiče rotora v smere pohybu statorového magnetického poľa.



Obr. 2 Vznik ťažnej sily asynchrónneho motora [4]

Z obr. 2 je viditeľné, že napätie (teda i prúd) sa bude vo vodičoch rotora indukovať len v prípade relatívneho pohybu pola statora voči vodičom rotora. V tomto prípade nemôžu byť otáčky točivého pola statora ω_1 a otáčky rotora ω rovnaké. Rozdiel otáčok ω_1 a ω, vzťahujúci sa na jednu otáčku točivého pola statora, je tzv. sklz s. (2.1 a 2.2)

$$s = \frac{\omega_1 - \omega}{\omega_1} \tag{2.1}$$

$$s_{\%} = \frac{\omega_1 - \omega}{\omega_1} \cdot 100 \tag{2.2}$$

Otáčky magnetického pola statora závisia na počte pólov a frekvencii napájacieho napätia. Sklz (a teda i otáčky rotora) sa mení s mechanickým zaťažením stroja a pri menovitom zaťažením u malých motorov je približne 10%, u veľkých okolo 1%.

Podrobnejšie: Predpokladajme najprv rozpojené rotorové vinutie, stojací rotor ($\omega = 0$) a vinutie statora pripojené na trojfázovú sieť. Trojfázový prúd vo vinutí statora vytvára točivé magnetické pole vo vzduchovom medzere, ktoré indukuje v statorovom vinutí napätie U_{i1} rovné približne napätiu siete a v rozpojenom rotorovom vinutí napätie U_{i2o} (s = l). Prúd rotora je rovný nule a vtedy i tok $\mathbf{\Phi_2} = 0$. Výsledný magnetický tok $\mathbf{\Phi} = \mathbf{\Phi_1}$ a stroj sa javí ako trojfázový transformátor naprázdno. Elektromagnetický moment stroja sa rovná nule. V tomto prípade tečie vinutím statora len magnetizačný

prúd, nutný k vytvoreniu točivého pola statora. Rozpojené rotorové vinutie vtedy nemá žiadny vplyv na pole statora obdobne ako rozpojené sekundárne vinutie transformátora neovplyvňuje činnosť primárneho vinutia. Spojme teraz rotorové vinutie tak, aby ním mohol pretekať prúd a nech sa rotor točí synchrónnymi otáčkami $\omega = \omega_1$ ($s = 0$). Ani v tomto prípade nebude prechádzať rotorovým vinutím prúd a motor nebude vytvárať elektromagnetický moment, pretože točivé pole statora sa otáča rovnakou rýchlosťou ako vodiče rotora, neindukuje sa v nich žiadne napätie a neprechádza žiadny prúd. V skutočnosti motor nemôže v tomto stave pracovať, pretože potrebuje určitý moment na krytie vlastných mechanických strát. Ak zaťažujeme stroj mechanickým momentom na hriadeli, poklesnú otáčky rotora na určitú hodnotu ω a nastane relatívny pohyb medzi otáčavým magnetickým polom statora a vodiči rotora. Vo vinutí rotora sa indukuje napätie U_{i2} a cez vodiče rotora preteká prúd, ktorý vytvára magnetický tok $\boldsymbol{\Phi_2}$ ktorý pôsobí proti toku $\boldsymbol{\Phi_1}$. Pôsobením toku $\boldsymbol{\Phi_2}$ by sa zoslabil výsledný magnetický tok $\boldsymbol{\Phi}$ vo vzduchovej medzere a zmenšilo by sa indukované napätie U_{i1} statorového vinutia. Do vinutia statora ale potečie taký zvýšený prúd, aby sa vo vzduchovej medzere opäť vytvorila pôvodná hodnota magnetického toku $\boldsymbol{\Phi}$, nutná pre indukované napätie U_{i1} približne rovného napätia siete. Výsledný magnetický tok stroja $\boldsymbol{\Phi}$ zostává vtedy za predpokladu konštantného napätia napájacej siete približne konštantný bez ohľadu na veľkosť prúdu vo vinutí stroja.

Text kapitoly bol vytvorený podľa [5]

3 MATEMATICKÝ MODEL ASYNCHRÓNNEHO MOTORA

Modely asynchrónnych motorov môžeme rozdeliť na dynamické a statické. Všeobecnejšími sú dynamické modely, iným triediacim kritériom (podľa použitého súradnicového systému) ich priraďujeme na modely v stojacích súradniciach, alebo na modely v rotujúcich súradniciach. Modely v rotujúcich súradniciach sú spriahnuté s niektorým vektorom magnetického toku. V niektorých prípadoch sú dynamické modely nepoužiteľne všeobecné a je praktickejšie a vhodnejšie ich zjednodušiť na modely statické. Základom pre tieto úpravy je predpoklad harmonických priebehov napätia a prúdu.

Text kapitoly bol vytvorený podľa [1]

3.1 Model v stojacom súradnicovom systéme

Celkové jednoduchšie modelovanie je možné dosiahnuť, keď trojfázový asynchrónny motor je nahradený dvojfázovým rovnocenným motorom, pre ktorý je celkové spracovanie a úpravy dosiahnuté prehľadnejšími matematickými rovnicami a tiež návrhom riadiaceho algoritmu. Pre názornejšie pochopenie a zdôraznenie uvádzame, že popis je pomocou komplexných priestorových vektorov. Pre správne použitie popisu motora priestorovými vektormi je potrebné dodržať nasledovné podmienky.

- lineárna magnetizačná charakteristika
- symetrické rozloženie jednotlivých vinutí statora a rotora
- harmonické rozloženie magnetického toku vo vzduchovej medzere

Komplexné priestorové vektory napätia a prúdu v stojacích súradniciach sú dané rovnicami

$$\boldsymbol{u_s} = \frac{2}{3}(u_A + \boldsymbol{a}u_B + a^2 u_C) = u_{s_\alpha} + j u_{s_\beta}$$
$$\boldsymbol{i_s} = \frac{2}{3}(i_A + \boldsymbol{a}i_B + a^2 i_C) = i_{s_\alpha} + j i_{s_\beta} \tag{3.1}$$

kde

$$\boldsymbol{a} = e^{j\frac{2\pi}{3}} \tag{3.2}$$

u_A, u_B, u_c – okamžité hodnoty fázových napätí
i_A, i_B, i_C – okamžité hodnoty fázových prúdov

Pre motory zapojené do trojuholníka a pre motory zapojené do hviezdy bez vyvedeného stredu, je súčet fázových prúdov je rovný k nule.

$$i_A + i_B + i_C = 0 \tag{3.3}$$

Matematické vyjadrenie asynchrónneho motora v stojacom súradnicovom systéme vieme popísať tromi diferenciálnymi rovnicami s priestorovými vektormi

$$\frac{d\boldsymbol{\Psi}_s}{dt} = \boldsymbol{u}_s - R_s\boldsymbol{i}_s \tag{3.4}$$

$$\frac{d\boldsymbol{\Psi}_r}{dt} = j\omega\boldsymbol{\Psi}_r - R_r\boldsymbol{i}_r \tag{3.5}$$

$$J\frac{d\omega}{dt} = M - M_z \tag{3.6}$$

$$M = \frac{3}{2}z_p\Im(\boldsymbol{\Psi}_s^*\boldsymbol{i}_s) = \frac{3}{2}z_p\Im(\boldsymbol{i}_r^*\boldsymbol{\Psi}_s) = \frac{3}{2}z_p\Im(\boldsymbol{i}_r^*\boldsymbol{\Psi}_r) \tag{3.7}$$

$\boldsymbol{\Psi}_r$ - vektor magnetického toku rotora
$\boldsymbol{\Psi}_s$ - vektor magnetického toku statora
R_r, R_s - odpory rotora a statora
M_z - zaťažovací moment
M - moment motora
z_p - počet pólových dvojíc
$\Im$ - imaginárna časť vektora
J - moment zotrvačnosti rotora
$\boldsymbol{x}^*$ - symbol, vyjadruje vektor komplexne združený k vektoru $\boldsymbol{x}$
$j\omega\boldsymbol{\Psi}_r$ - je sila spôsobená otáčajúcim rotorom ktorá sa účinkuje pri premene elektrickej energie na mechanickú

Nasledujúce dve algebrické rovnice definujú vzťah medzi vektorom magnetických tokov a vektorom prúdov

$$\boldsymbol{\Psi}_s = (L_{sl} + L_m)\boldsymbol{i}_s + L_m\boldsymbol{i}_s = L_s\boldsymbol{i}_s + L_m\boldsymbol{i}_s \tag{3.8}$$

$$\boldsymbol{\Psi}_r = L_m\boldsymbol{i}_s + (L_{sl} + L_m)\boldsymbol{i}_r = L_m\boldsymbol{i}_s + L_r\boldsymbol{i}_r \tag{3.9}$$

L_{sl}, L_{rl} - rozptylové indukčnosti statora a rotora
L_s, L_r - indukčnosti statora a rotora
L_m - magnetizačná indukčnosť

Text kapitoly bol vytvorený podľa [1]

3.2 Model v všeobecnom súradnicovom systéme

Matematický model, v súradnicovom systéme otáčajúcom sa uhľovou rýchlosťou ω_0,je možno získať zo závislosti medzi stojacími a otáčajúcimi sa veličinami a z rovníc v stojacom súradnicovom systéme.

$$\begin{aligned} \boldsymbol{u}_{s_o} &= \boldsymbol{u}_s e^{-j\varphi o} = u_{s_x} + j u_{s_y} \\ \boldsymbol{i}_{s_o} &= \boldsymbol{i}_s e^{-j\varphi o} = i_{s_x} + j i_{s_y} \\ \boldsymbol{\Psi}_{s_o} &= \boldsymbol{\Psi}_s e^{-j\varphi o} = \Psi_{s_x} + j\Psi_{s_y} \end{aligned} \tag{3.10}$$

$$\begin{aligned} \boldsymbol{u}_{r_o} &= \boldsymbol{u}_r e^{-j\varphi o} = u_{r_x} + j u_{r_y} \\ \boldsymbol{i}_{r_o} &= \boldsymbol{i}_r e^{-j\varphi o} = i_{r_x} + j i_{r_y} \\ \boldsymbol{\Psi}_{r_o} &= \boldsymbol{\Psi}_r e^{-j\varphi o} = \Psi_{r_x} + j\Psi_{r_y} \end{aligned} \tag{3.11}$$

Podľa rovníc (3.10) upravíme rovnicu (3.4)

$$\frac{d\boldsymbol{\Psi}_{s_o} e^{j\varphi o}}{dt} = \boldsymbol{u}_s e^{j\varphi o} - R_s \boldsymbol{i}_{s_0} e^{j\varphi o} \tag{3.12}$$

Po derivácii upravíme rovnicu a dostaneme

$$\frac{d\boldsymbol{\Psi}_{s_o}}{dt} = \boldsymbol{u}_{s_0} - R_s \boldsymbol{i}_{s_0} - j\omega_0 \boldsymbol{\Psi}_{s_o} \tag{3.13}$$

Pre vyjadrenie rotorového toku postupujeme podľa postupu statorového toku s tým, že dosadíme (3.11) do (3.5)

$$\frac{d\boldsymbol{\Psi}_{r_o}}{dt} = -R_s \boldsymbol{i}_{s_0} - j(\omega_0 - \omega)\boldsymbol{\Psi}_{r_o} \tag{3.14}$$

Transformácia sa na tvare algebrických rovníc neprejavuje, preto rovnice (3.8) a (3.9) zostávajú po transformácii formálne rovnaké.

$$\boldsymbol{\Psi}_s = (L_{sl} + L_m)\boldsymbol{i}_s + L_m \boldsymbol{i}_s = L_s \boldsymbol{i}_s + L_m \boldsymbol{i}_s \tag{3.15}$$

$$\boldsymbol{\Psi}_r = L_m \boldsymbol{i}_s + (L_{sl} + L_m)\boldsymbol{i}_r = L_m \boldsymbol{i}_s + L_r \boldsymbol{i}_r \tag{3.16}$$

K dispozícii sú dve možnosti, ktoré sú spriahnuté otáčaním so súradnicovým systémom.

Výhodou prvého systému je použitie na analýzu rotorových veličín, kde je súradnicový systém spojený so súradnicovým systémom rotora, otáčajúci sa elektrickou uhlovou rýchlosťou ω.

Druhý variant je zase prednostne používaný v oblasti riadenia asynchrónnych motorov. V danom prípade súradnicový systém je spojený so súradnicovým systémom magnetického pola statora, ktorý sa otáča synchrónnou uhlovou rýchlosťou ω_s. Užitočnosť sa prejavuje v tom, že veličiny sú konštantné v ustálenom stave.

Text kapitoly bol vytvorený podľa [1]

3.3 Stavové rovnice asynchrónneho motora

Základným predpokladom pre používanie stavových rovníc je voľba veličín stavových premenných. Samotná voľba veličín môže byť rôzna, ale musíme vychádzať z tých, ktoré sú pre celkový návrh užitočné, ako prúd statora $\boldsymbol{i}_s$, magnetický tok rotora $\boldsymbol{\Psi}_r$ a elektrická uhľová rýchlosť ω. Nakoľko $\boldsymbol{\Psi}_r$ a $\boldsymbol{i}_s$ sú vektory, celkovo je potrebné počítať s piatimi premennými.

Pri úprave rovníc (3.12), (3.13) platných v súradnicovom systéme otáčajúcich sa s ľubovoľnou rýchlosťou musíme nadobudnúť tvar, aby sme dostali len stavové premenné

$$\frac{d\boldsymbol{i}_s}{dt} = \frac{L_r}{L_s L_r - L_r^2}\left(\boldsymbol{u}_s - \left(R_s + \frac{L_m^2}{L_r^2}R_r + j\omega_0\left(L_s + \frac{L_m^2}{L_r}\right)\right)\boldsymbol{i}_s + \left(\frac{L_m}{L_r^2}R_r - j\omega\frac{L_m}{L_r}\right)\boldsymbol{\Psi}_r\right) \tag{3.17}$$

$$\frac{d\boldsymbol{\Psi}_\mathrm{r}}{dt} = \frac{R_r L_m}{L_r}\boldsymbol{i}_s - \left(\frac{R_r}{L_r} - j(\omega - \omega_0)\right)\Psi_r \tag{3.18}$$

Rovnice sa zjednodušujú zavedením nasledujúce substitúcie. Časová konštanta rotora sa dá vyjadriť ako

$$T_r = \frac{L_r}{R_r} \tag{3.19}$$

a činiteľ celkového rozptylu

$$\sigma = 1 - \frac{L_m^2}{L_s L_r} = \frac{L_s L_r - L_m^2}{L_s L_r} \tag{3.20}$$

Rovnice (3.34) a (3.35) môžeme potom prepísať na

$$\frac{d\boldsymbol{i}_s}{dt} = \frac{1}{\sigma L_s}\left(\boldsymbol{u}_s - \left(R_s + \frac{L_m^2}{L_r^2}R_r + j\omega_0 L_s\right)\boldsymbol{i}_s + \left(\frac{L_m}{L_s T_r} - j\omega\frac{L_m}{L_r}\right)\boldsymbol{\Psi}_r\right) \tag{3.21}$$

$$\frac{d\boldsymbol{\Psi}_\mathrm{r}}{dt} = \frac{L_m}{T_r}\boldsymbol{i}_s - \left(\frac{1}{T_r} - j(\omega - \omega_0)\right)\Psi_r \tag{3.22}$$

Po rozpísaní rovníc (3.21) a (3.22) na jednotlivé zložky v súradniciach d,q a použitím rovníc popisujúcich mechanickú časť motora (3.6) a (3.7) získame maticové rovnice stavového popisu asynchrónneho motora.

$$\begin{pmatrix} \dot{\imath}_{s_x} \\ \dot{\imath}_{s_y} \\ \dot{\Psi}_{s_x} \\ \dot{\Psi}_{s_y} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} -R_s - \frac{L_m^2}{L_r T_r} & \omega_0 \sigma L_s & \frac{L_m}{L_r T_r} & \omega \frac{L_m}{L_r} & 0 \\ -\omega_0 \sigma L_s & -R_s - \frac{L_m^2}{L_r T_r} & -\omega \frac{L_m}{L_r} & \frac{L_m}{L_r T_r} & 0 \\ \frac{L_m}{T_r} & 0 & -\frac{1}{T_r} & -\omega + \omega_0 & 0 \\ 0 & \frac{L_m}{T_r} & -\omega + \omega_0 & -\frac{1}{T_r} & 0 \\ -\frac{3 z_p}{2J} \Psi_{s_q} & \frac{3 z_p}{2J} \Psi_{s_d} & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} i_{s_x} \\ i_{s_y} \\ \Psi_{s_x} \\ \Psi_{s_y} \\ \omega \end{pmatrix} + \begin{pmatrix} \frac{1}{\sigma L_s} & 0 & 0 \\ 0 & \frac{1}{\sigma L_s} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{J} \end{pmatrix} \begin{pmatrix} u_{s_x} \\ u_{s_y} \\ M_z \end{pmatrix} \tag{3.23}$$

Z rovníc uvedených vyššie je patrná nelinearita modelu asynchrónneho motora. Prvé štyri riadky matice $\boldsymbol{A}$ obsahujú na štyroch miestach elektrickú uhlovú rýchlosť ω. Posledný riadok definujúci otáčky, je taktiež výrazne nelineárny, pretože obsahuje zložky vektoru magnetického toku rotora. V prípade vektorového riadenia v súradnicovom systéme spojený s vektorom $\boldsymbol{\Psi}_r$ je možné túto nelinearitu, za podmienky konštantného budenia, eliminovať. Ďalším nedostatkom je vstup zaťahovacieho momentu, ktorý nie je prakticky merateľný.

Text kapitoly bol vytvorený podľa [1]

4 SYNCHRONNÝ MOTOR

4.1 Synchrónny motor budený permanentnými magnetmi na rotore.

Synchrónny motor je elektrický stroj, ktorého rotor sa otáča synchrónne s otáčaním točivého magnetického pola statora. Stator je zhodný so statorom asynchrónneho motora, to znamená že je leštený, s drážkami pre uloženie statorového vinutia. Vinutie je obvykle trojfázové, rozmiestnené do statorových drážok a podľa konštrukčného vyhotovenia môže byť dvojpólové alebo viacpólové.

Rotor môže byť hladký alebo s vyniklými pólmi, ktoré môžu byť opatrené budiacim vinutím, napájaným jednosmerným prúdom. Budiaci prúd sa do rotorového vinutia privádza buď pomocou klzných kontaktov, alebo bezkontaktne rotačným transformátorom a následným usmernením diódami na rotore. Synchrónne motory určené na pripojenie ku striedavej napájacej sieti majú na rotore takzvané tlmiace vinutie, ktoré môže slúžiť pre asynchrónny rozbeh motora.

Konštrukcia synchrónneho motora s permanentnými magnetmi na rotore sa podobá elektricky komutovanému motoru. Magnety môžu byť na povrchu rotora buď ako zapustené, v tomto prípade je menej potlačený vplyv reakcie statorového vinutia na tvar pola vo vzduchovej medzere, alebo magnet je uložený na povrchu rotora, čo predstavuje umiestnenie vo vzduchovej medzere. Výhodou je potlačenie vplyvu reakcie statorového vinutia vplyvom veľkej vzduchovej medzery, pretože permanentné magnety majú prakticky rovnakú permeabilitu ako vzduch. Od elektricky komutovaného motora sa v obidvoch prípadoch líši veľkosťou pólového krytia, zatiaľ kým elektricky komutovaný motor vyžaduje indukciu obdĺžnikového tvaru vo vzduchovej medzere, čo sa dosahuje pólovým krytím blízkym k jednotke, synchrónny motor vyžaduje pole sínusového tvaru vo vzduchovej medzere, čo je približne dosahované dvojtretinovým pólovým krytím. Pólové krytie je pomer obvodu pólových nástavcov, prípadne obvodu povrchu permanentných magnetov vo vzduchovej medzere, ku celkovému obvodu vzduchovej medzery.

Iným konštrukčným princípom je, podobne ako u elektronicky komutovaných motorov, uloženie permanentných magnetov vnútri rotora. Popísaný konštrukčný princíp sa používa najmä pri použití feritových magnetov, pretože umožňuje koncentráciu magnetického toku do vzduchovej medzery. Navyše vhodným tvarom pólových nástavcov (t.j. premennou vzduchovou medzerou) možno ľahko dosiahnuť magnetické pole sínusového tvaru vo vzduchovej medzere. Nevýhodou je väčší účinok reakcie statorového vinutia na tvar tohto poľa pri zaťažení motora.

Materiálom pre výrobu permanentných magnetov sú používané vzácne zeminy, ako napríklad: samárium – kobalt (SmCO5, SmCO17), alebo neodyn – železo – bór

(NdBFe), prípadne i tvrdé ferity, ktoré sú lacnejšie, no s horšími magnetickými vlastnosťami.

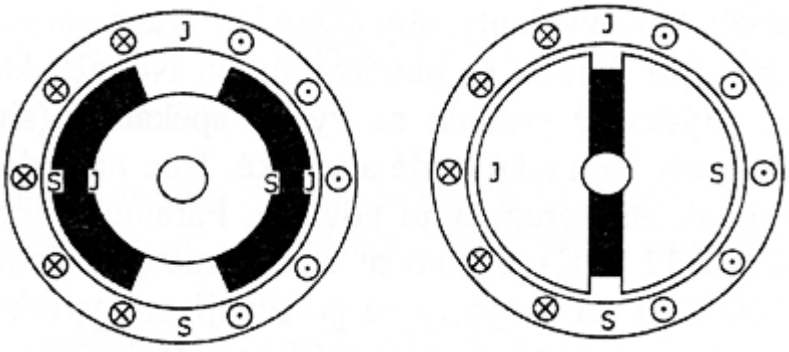
Synchrónne motory budené permanentnými magnetmi na rotore, určené pre servopohony, sú bez tlmiaceho vinutia na rotore, pretože tieto motory pracujú stále v synchrónnom režime, podmienenou spätnou väzbou od polohy rotora.

Hlavným rozlišovacím znakom synchrónnych motorov od motorov s elektronickou komutáciou je použitý princíp snímania polohy rotora pre účely riadenia tranzistorového meniča, napájajúceho statorové vinutie: kým elektronicky komutovaný motor vyžadoval iba diskrétne snímanie vždy po 60 el. stupňoch, synchrónny motor potrebuje trvalú informáciu o polohe rotora.

Synchrónne servomotory s permanentnými magnetmi na rotore sú najbežnejším typom motorov pre polohové servo výrobných strojov, robotov a v automatizačnej technike. Vzhľadom k asynchrónnym motorom pri rovnakom výkone sú menšie a ľahšie, lepšie sa chladia (na rotore nevznikajú straty a nie je teda nutné odvádzať teplo z rotora), v porovnaní s elektronicky komutovaným motorom sa vyznačujú vyšším homogénnym chodom bez momentových pulzácií. Snímač polohy rotora je možné pochopiteľne využiť i pri pozičnej spätnej väzbe pri polohovom riadení servopohonu.

Pohony s týmito servomotormi sú v zahraničnom literatúre označené ako Brussless A.C. Motor Servodrives čiže bezkartáčové, zároveň sa dá stretnúť s označením PMSM (Permament Magnet Synchronous Motor) aj keď tento názov je trochu všeobecnejší, pretože sa niekedy používa aj pre elektricky komutované motory.

Text kapitoly bol vytvorený podľa [7][11]



a, b,

Obr. 3 Rôzne konštrukcie rotora synchronného motora s permanentnými magnetmi[9]

a, magnety umiestnené na povrchu rotora

b, magnety umiestnené vnútri rotora

5 MATEMATICKÝ MODEL SYNCHRÓNNEHO MOTORA S PERMANENTNÝMI MAGNETMI

Pri vytváraní modelu vychádzame z nasledujúcich zjednodušujúcich predpokladov

- Priebeh magnetickej indukcie vo vzduchovej medzere a teda i indukovaného napätia je sínusový, pričom je všeobecne uvažovaný rotor s vyniklími pólmi, t.j. s rôznou magnetickou vodivosťou v pozdĺžnom a priečnom smere
- Parametre (R, L) stroje sú konštantné a rovnaké vo všetkých troch fázach
- Straty v železe sú zanedbané
- Nulový vodič nie je pripojený

Riešenie rovnice modelu je vhodné vykonávať v súradnicovom systéme *d,q*
Napäťové rovnice synchrónneho stroja

$$u_d = R_s i_d + \frac{d\Psi_d}{dt} - \omega\Psi_q \tag{5.1}$$

$$u_q = R_s i_q + \frac{d\Psi_q}{dt} + \omega\Psi_d \tag{5.2}$$

Pre magnetické spriahnutie platí

$$\Psi_d = L_d i_d + \Psi_f \tag{5.3}$$

$$\Psi_q = L_q i_q \tag{5.4}$$

Napäťové rovnice obecného synchrónneho (5.1, 5.2) stroja upravíme s použitím predchádzajúcich rovníc

$$u_d = R_s i_d + \frac{d\Psi_d}{dt} - \omega\Psi_q = R_s i_d + \frac{d\left(L_d i_d + \Psi_f\right)}{dt} - \omega L_q i_q = R_s i_d + L_d \frac{di_d}{dt} - \omega L_q i_q \tag{5.5}$$

$$u_q = R_s i_q + \frac{d\Psi_d}{dt} + \omega\Psi_q = R_s i_q + L_q \frac{di_q}{dt} + \omega\left(L_d i_d + \Psi_f\right) \tag{5.6}$$

Z nich potom určíme deriváciu stavových veličín, čo sú stavové prúdy. Po Laplasovej transformácii dostaneme

$$pI_d = \frac{1}{L_d}\left(U_d - R_s i_d + \omega L_q i_q\right) \tag{5.7}$$

$$pI_q = \frac{1}{L_q}\left(U_q - R_s i_d - \omega L_d i_d - \omega\Psi_f\right) \tag{5.8}$$

kde *p* je Laplasov operátor

Maticový zápis stavových rovníc potom bude nasledovný

$$p\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \begin{bmatrix} -\frac{R_s}{L_d} & \frac{\omega L_q}{L_d} \\ -\frac{\omega L_d}{L_q} & -\frac{R_s}{L_q} \end{bmatrix}\begin{bmatrix} I_d \\ I_q \end{bmatrix} + \begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \end{bmatrix}\begin{bmatrix} U_d \\ U_q - \omega\Psi_f \end{bmatrix} \tag{5.9}$$

Z pohybovej rovnice získame deriváciu tretej stavovej premennej mechanickej rýchlosti

$$p\omega_m = \frac{1}{J_c}(M_e - M_z) \tag{5.10}$$

Elektrická rýchlosť

$$\omega = z_p\, \omega_m \tag{5.11}$$

kde z_p je počet pólových dvojíc
Elektromagnetický moment stroja

$$M_e = \frac{3}{2}\, z_p \left(\Psi_d i_q - \Psi_q i_d\right) = \frac{3}{2}\, z_p \left[\Psi_f + \left(L_d - L_q\right) i_d\right] i_q \tag{5.12}$$

Keď $L_d = L_q$, potom sa rovnica pre moment zjednodušuje na tvar

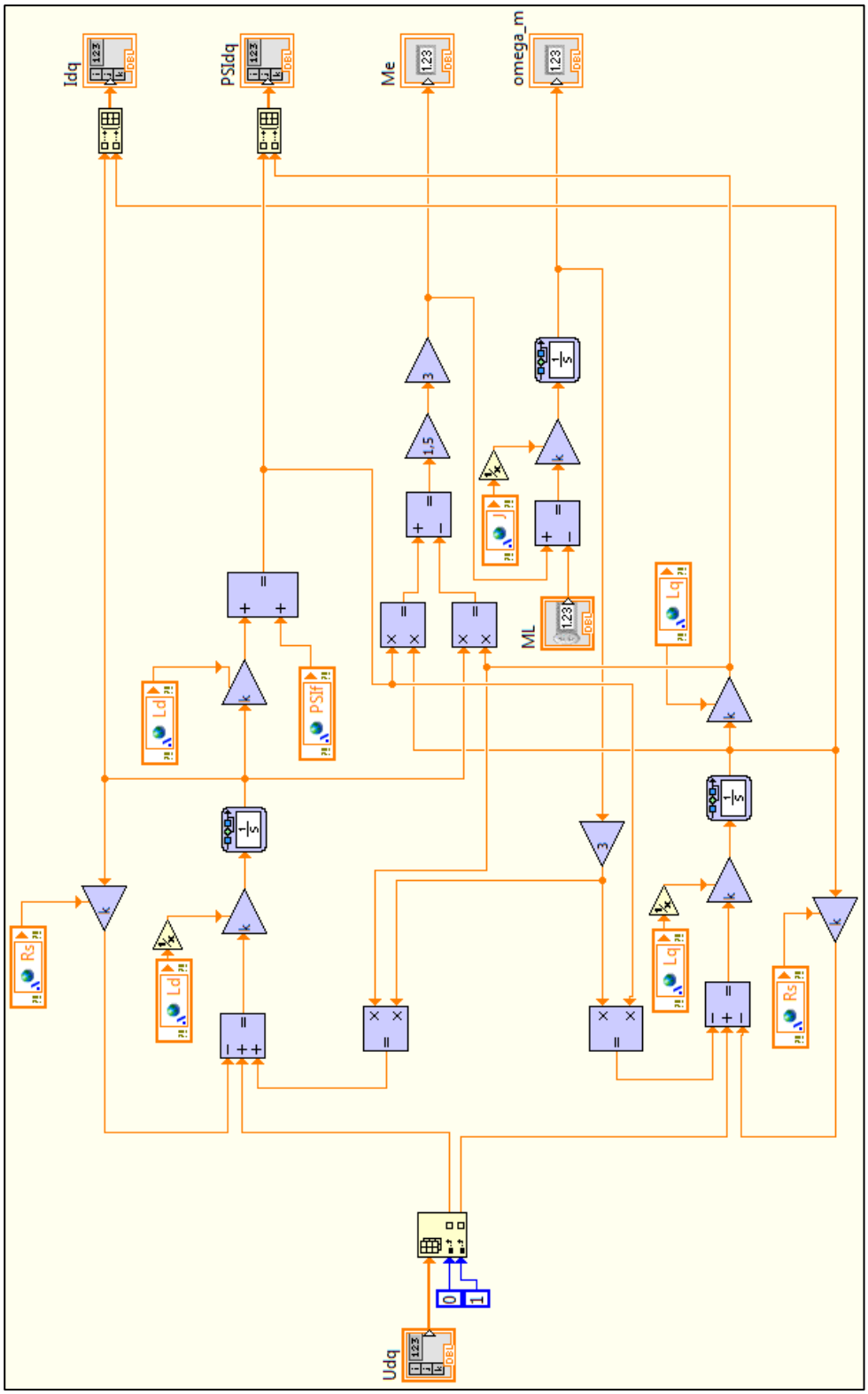
$$M_e = \frac{3}{2}\, z_p\, \Psi_f i_q \tag{5.13}$$

Tento vzťah nám udáva, že moment stroja je daný súčinom kolmej zložky priestorového vektora statorového prúdu a konštantného budiaceho spriahnutého toku, ktorý je daný len permanentnými magnetmi a nie výsledným tokom ovplyvneným statorovým prúdom.

Táto skutočnosť umožňuje navrhnúť pomerne jednoduchú regulačnú štruktúru, v ktorej nemusíme zisťovať veľkosť a polohu celkového magnetického spriahnutého toku, ale stačí sa orientovať len na polohu rotora stroja.

Vyššie uvedený vzťah zodpovedá blokovej schéme na obr. 3 vytvorený v prostredí LabVIEW.

Text kapitoly bol vytvorený podľa [7]

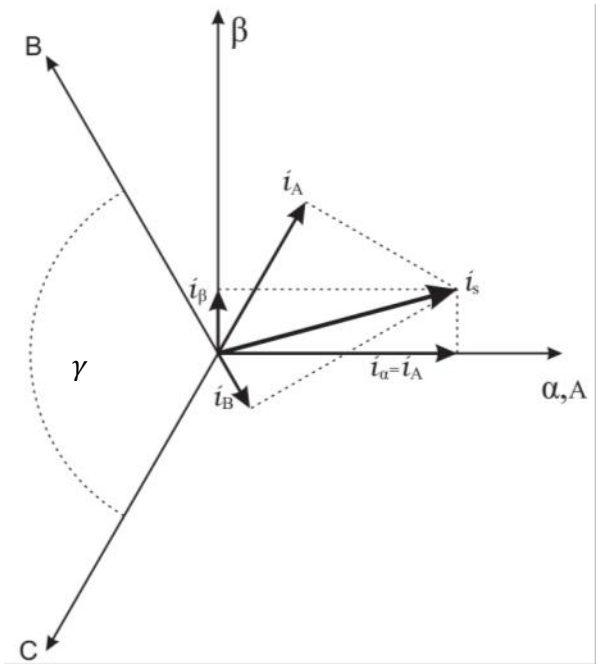


Obr. 4 Matematický model synchrónneho motora s permanentnými magnetmi

6 TRANSFORMÁCIA SÚRADNÍC

6.1 Transformácia podľa E. Clarkovej

Clarkovej transformácia je transformácia, ktorá prevedie tri osi dvojdimenzionálného súradnicového systému statoru (*A, B, C*) do dvoch os (*α, β*) pri zachovaní súradnicového systému statoru. Nazýva sa i transformácia 2→3.



Obr. 5 Grafické znázornenie Clarkovej transformácie [9]

Väčšina trojfázových motorov má zapojené vinutie do hviezdy takže súčet prúdov jednotlivých fáz je rovný nule

$$i_A + i_B + i_C = 0 \tag{6.1}$$

Z tejto rovnice vyplýva, že pre meranie prúdu stačí snímať len dva prúdy a tretí sa dá dopočítať.

Transformácia z trojfázového do dvojfázového systému je zapísaná nasledujúcim vzťahom

$$\begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} = \frac{2}{3} \begin{pmatrix} 1 & \cos\gamma & \cos 2\gamma \\ 0 & \sin\gamma & \sin 2\gamma \end{pmatrix} \begin{pmatrix} i_A \\ i_B \\ i_C \end{pmatrix} \tag{6.2}$$

Kde γ = 2π/3 , a odtiaľ dostaneme

$$\begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} = \begin{pmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \end{pmatrix} \begin{pmatrix} i_A \\ i_B \\ i_C \end{pmatrix} \tag{6.3}$$

Podobným spôsobom sa dá použiť transformáciu pre statorové napätie a spriahnuté toky.

Priestorový vektor prúdu v statorovom súradnicovom systéme môžeme určiť pomocou zložiek prúdu α a β ako

$$\boldsymbol{i_s} = i_\alpha + j i_\beta \tag{6.4}$$

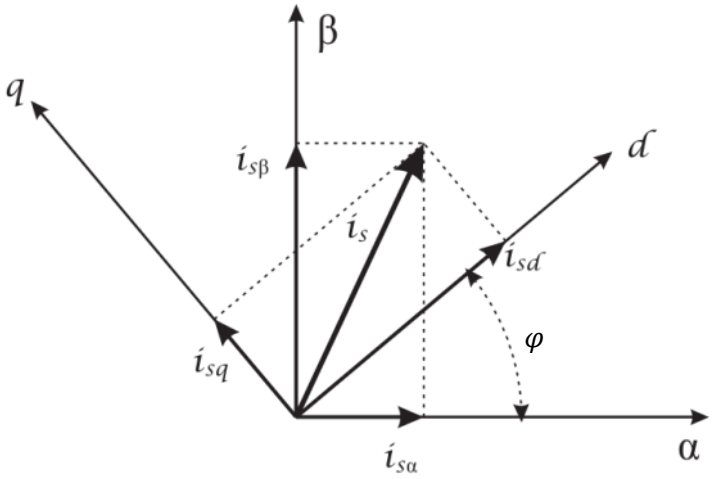
Jedná sa o spätnú transformáciu z dvojfázového do trojfázového systému, ktorá sa prevádza priemetom vektoru do osi jednotlivých cievok pomocou rovnice

$$\begin{pmatrix} i_A \\ i_B \\ i_C \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{pmatrix} \begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} \tag{6.5}$$

Text kapitoly bol vytvorený podľa [1][5][6]

6.2 Parkova transformácia

Parkova transformácia vykonáva transformáciu zo statorového súradnicového systému α,β do rotorového súradnicového systému d,q, kde jedna súradnica je spojená s vektorom magnetického toku rotora $\boldsymbol{\Psi}_r$ a druhá je na ňu kolmá. Motor sa otáča synchrónne, pretože otáčanie rotorového a statorového vinutia má rovnakú rýchlosť a vinutia majú voči sebe rovnakú polohu. Z periodických koeficientov sa stanú konštantné.



Obr. 6 Grafické znázornenie Parkovej transformácie[1]

Matematicky sa táto transformácia dá popísať pomocou rovnice

$$\begin{pmatrix} i_d \\ i_q \end{pmatrix} = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} = \boldsymbol{T} \begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} \tag{6.6}$$

Spätnú transformáciu získame používaním inverznej matice k matici $\boldsymbol{T}$

$$\begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} = \boldsymbol{T}^{-1} \begin{pmatrix} i_d \\ i_q \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} i_d \\ i_q \end{pmatrix} \tag{6.7}$$

Keď chceme riadiť motor v rotorových súradniciach musíme transformovať všetky veličiny, ktoré pri riadení potrebujeme.

Text kapitoly bol vytvorený podľa [1][5][6]

7 VEKTOROVÉ RIADENIE

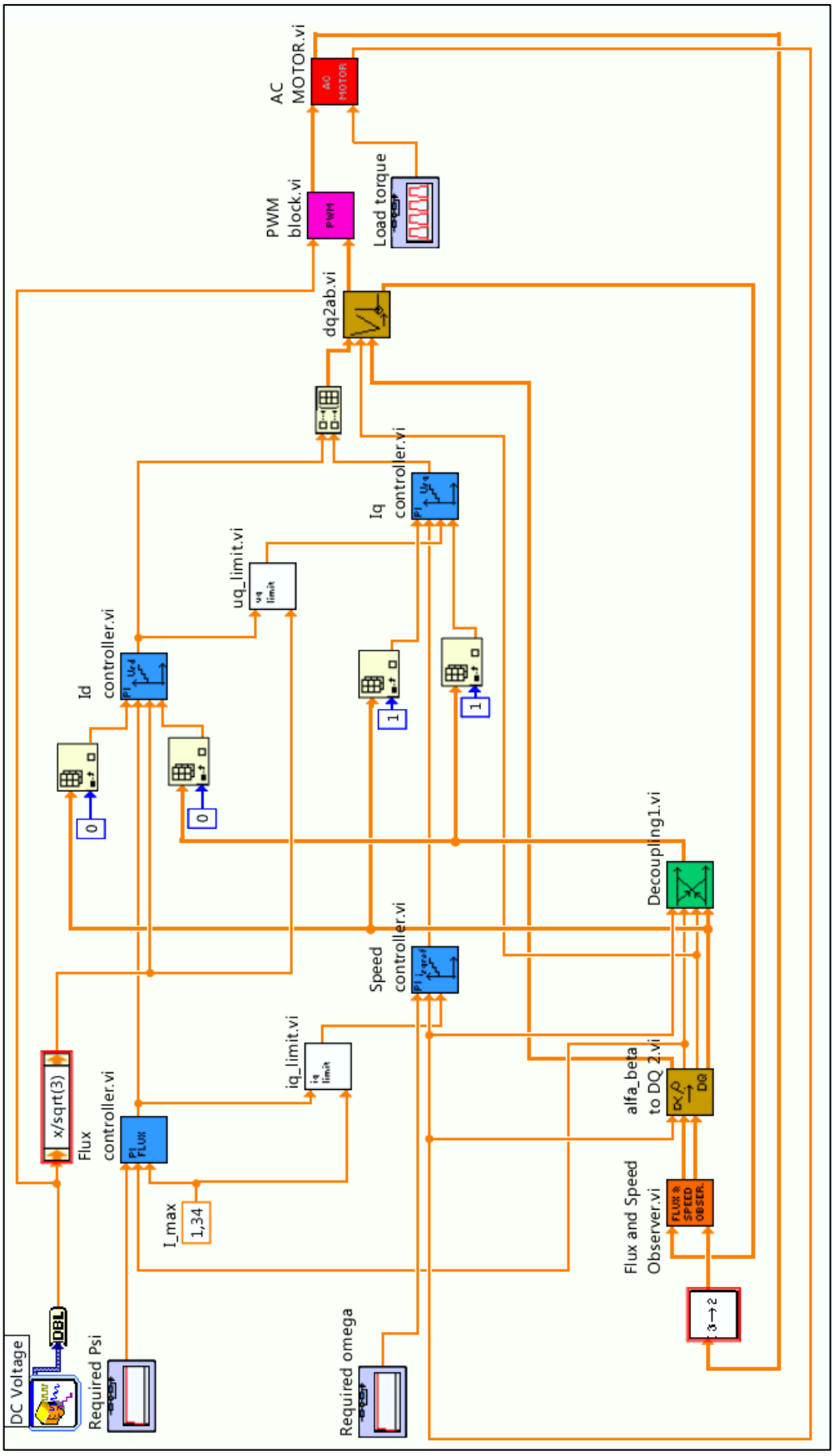
Vektorové riadenie striedavých motorov umožňuje prevádzku motora v optimálnych podmienkach v ustálenom stavu i počas prechodových dejov. Regulácia s vektorovým riadením motora v súradniciach spojené s magnetickým tokom rotora predstavuje najkvalitnejší typ regulácie chodu pohonu. Princípom je založený na samostatnom riadení dvoch zložiek prúdu, komplexoru statorového prúdu. Jedna zložka prúdu je vo fáze s vektorom toku rotora a mení magnetický tok motora a vytvára jalový výkon, druhá zložka (je naň kolmá) vytvára absolútnu hodnotu momentu a činný výkon motora. Zmena jednotlivých zložiek je uskutočniteľná nezávisle (bez vzájomného ovplyvňovania). Týmto postupom môžeme zvlášť riadiť moment a magnetický tok.

Na rozdiel od teórie je prakticky ťažko dosiahnuteľné úplné oddelenie oboch zložiek, pretože pre predikciu vektoru magnetického toku rotora sa využíva model motora, ktorého parametre sa menia s pracovnými podmienkami stroja. Ako nevýhodu tohto spôsobu je potrebné poukázať na riadenie prebiehajúce v otáčajúcom sa riadiacom systéme, čo so sebou prináša nutnosť použiť transformáciu do daného súradnicového systému a následnú transformáciu vypočítaných akčných zásahov naspäť do stojaceho súradnicového systému. Tieto výpočty sú časovo náročné, kvôli používaným goniometrickým funkciám.

Text kapitoly bol vytvorený podľa [1] [10]

7.1 Algoritmus vektorového riadenia

Bloková schéma algoritmu vektorového riadenia v rotorových súradniciach vytvorený v simulačnom prostredí LabView je znázornená na obrázku 7. Merané fázové prúdy motora sa najprv transformujú pomocou Clarkovej transformácie do stojaceho súradnicového systému α,β (3→2). Transformovaný prúd $\boldsymbol{i_s}$ sa vedie spoločne so statorovým napätím u_s do bloku estimátoru polohy vektora magnetického toku. V ďalšom bloku sa prevádza Parkova transformácia jednotlivých veličín potrebných k riadeniu s α,β súradnicami do súradnicového systému d,q. V ďalších výpočtoch sa počíta so skalárnou veličinou Ψ_{rd}, pretože druhá zložka vektora toku je nulová. Do regulátora toku spolu s tou hodnotou vedie aj žiadaná hodnota toku. Na výstupe



Obr. 7 Schéma vektorového riadenia v programe LabVIEW

regulátora je požadovaná hodnota *d* zložky prúdu i_{sdref}. Regulátor je typu PI s obmedzeným akčného zásahu. Ako obmedzenie slúži medzná dovolená hodnota prúdu motora. Žiadaná hodnota prúdu i_{sdref} sa vedie do regulátora *d* zložky prúdu. Na jeho výstupe je požadovaná hodnota napätia u_{sdref}. Regulátor je taktiež typu PI, opäť je obmedzený akčný zásah. Tu napätie je obmedzené na medznú hodnotu dostupného napätia na jednosmernom medziobvode meniča. Na obrázku sa ešte dá všimnúť PI regulátor otáčok. Do neho vstupuje požadovaná hodnota otáčok ω_{ref} a namerané otáčky. Ako ďalší vstup regulátora je hodnota prúdu ktorá sa použije na obmedzenie akčného zásahu, ktorým je žiadaná hodnota *q* zložky prúdu i_{sqref}. Obmedzenie je vypočítané v bloku *iq_limit* tak, aby modul žiadanej hodnoty prúdu statoru $\sqrt{i_{sdref}^2 + i_{sqref}^2}$ neprekročil dovolenú medzu prúdu. Výstup sa vedie do regulátora *q* zložky prúdu, ktorý na výstupe dáva požadovanú hodnotu napätia u_{sqref}. Regulátor je opäť typu PI a je obmedzený pomocou bloku *uq_limit* tak aby modul napätia $\sqrt{u_{sdref}^2 + u_{sqref}^2}$ neprekročil napätie na jednosmernom medziobvode, podobne ako u žiadanej hodnoty prúdu. V bloku spätnoväzobnej linearizácie sa vykonáva linearizácia a rozdelenie motora na dve časti momentotvornú a tokotvornú . Takto upravené napätie sú privedené do príslušných regulátorov prúdu a pričítajú sa k akčnému zásahu. Potom sa napätie prepočíta naspäť z *d,q* súradnicového systému do súradnicového systému *α,β* pomocou spätnej transformácie podľa Parka. V poslednom bloku schémy sa vykonáva výpočet časov zopnutia výkonových tranzistorov napäťového striedača. Napäťový striedač v tejto schéme nie je zakreslený ale je formálne spojený s blokom impulznej šírkovej modulácie.

Text kapitoly bol vytvorený podľa [1]

7.2 Menič striedavých pohonov

Pre riadenie striedavých pohonov je z pohľadu výkonovej časti používaný statický menič frekvencie. Ten je vykonávateľom povelov riadiacej štruktúry. Existuje viac typov meniča, ale najpoužívanejším a najbežnejším je tzv. nepriamy menič frekvencie s medziobvodom. Tento typ meniča sa skladá z troch časti:

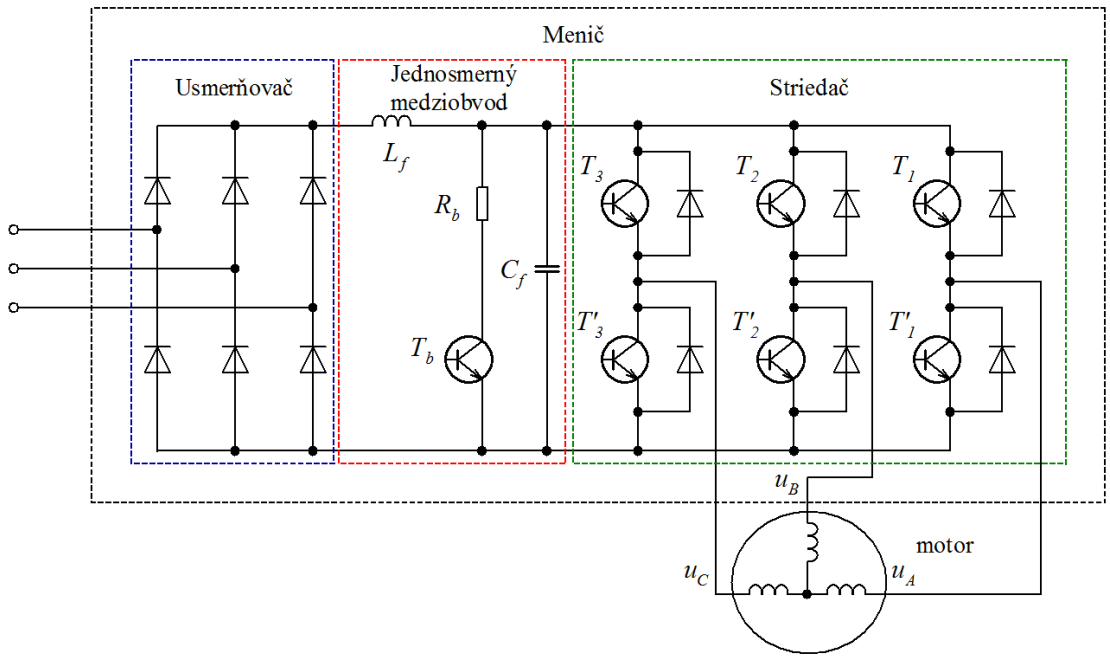
Usmerňovač má za úlohou premeniť striedavé napájacie napätie (jednofázové alebo trojfázové) na jednosmerné, často pulzujúce (podľa použitého typu usmerňovača) napätie. Používa sa diódový most alebo tranzistorový aktívny usmerňovač. Neriadený alebo diódový usmerňovač, ako je aj na obrázku 8 zobrazené nedokáže vracať prebytočnú energiu naspäť do siete. Aktívne usmerňovače sú tvorené pomocou výkonových tranzistorov a vstupných tlmiviek a pracujú v režime impulzne šírkovej

modulácie so sínusovým odberom zo siete a s riaditeľným účinkom. Majú minimálny nežiadajúci vplyv na napájaciu sieť.

Jednosmerný medziobvod má za cieľ stabilizovať usmernené pulzujúce napätie pre napäťový menič. Na obrázku 8 vidíme, že jednosmerný medziobvod obsahuje filtračnú indukčnosť L_f, kondenzátor C_f a brzdiacu časť. Brzdný celok pozostáva z brzdného odporu R_b a tranzistora T_b, ktorý sa spína pri náraste napätia na jednosmernom medziobvode nad dovolenú hodnotu.

Striedač je výkonový prvok spojený s riadeným motorom. Má za úlohu napájať motor okamžitými hodnotami fázových napätí podľa požiadavku riadiaceho algoritmu, alebo vytvoriť okamžitú hodnotu priestorového vektora napätí v motore. Trojfázový striedač je zložený z trojice vzájomne komplementárnych tranzistorov T_1-T'_1, T_2-T'_2 *a* T_3-T'_3. Komplementárnosťou sa dosahuje zopnutie iba jedného tranzistora z dvojice (v jednom čase), čím nedochádza ku skratu medzi napájacími vetvami. Pre spresnenie uvádzame, že medzi vypnutím jedného tranzistora a zapnutím druhého tranzistora je pridaná krátka časová hodnota t_d, keď sú oba tranzistory vypnuté. Táto časová veličina je riešením reálneho stavu, kde zopnutie tranzistoru je rýchlejšie ako rozopnutie. Vyriešením jednej kritickej situácie (oneskorenie spínania o čas t_d) zanášame do systému chybu v napätí statora, ktorá sa musí následne kompenzovať. Všetky tranzistory sú z tohto dôvodu induktívne záťaže antiparalelne spojené s diódami. Výsledné statorové napätie $\boldsymbol{u_{s1}}(1, 0, 0)$ vyjadruje, že sú zopnuté tranzistory T_1, T'_2 a T'_3.

Text kapitoly bol vytvorený podľa [1][8]



Obr. 8 Bloková schéma meniča [1]

7.3 Modulácia vektorového priestoru (SVM – Space Vector Modulation)

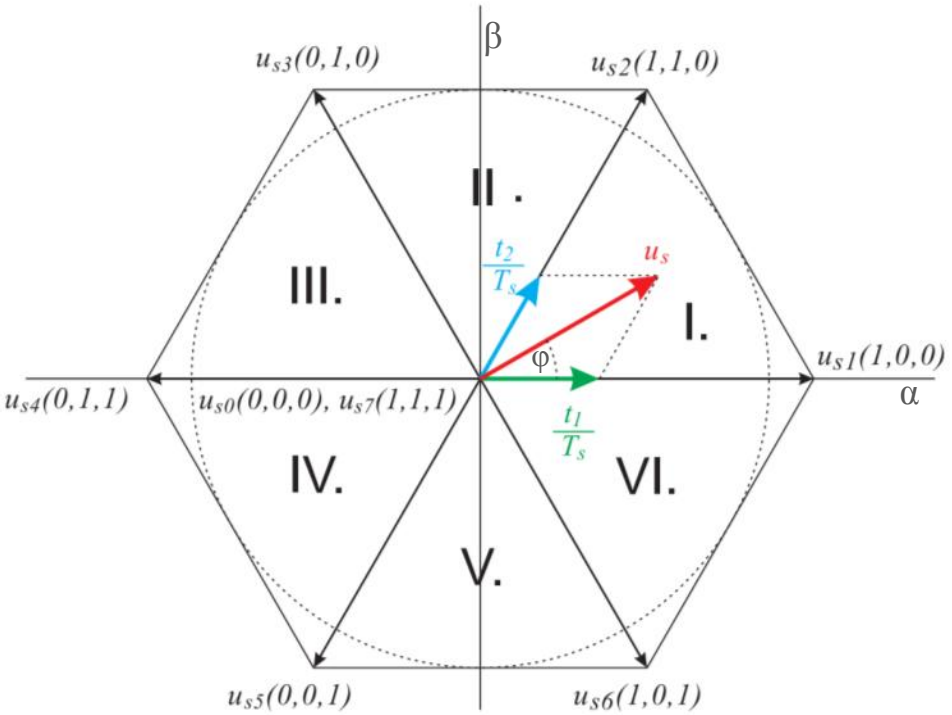
Pre generovanie stried pulzov obvykle sa používa spätná Clarkovej transformácia k získaniu trojfázového napätia. Taká metóda generovania stried avšak nepridáva do výslednej modulácie tretiu harmonickú zložku, a z toho vyplýva že nevyužije najväčšie možné výstupné napätie striedača.

Základom modulácie vektorového priestoru je výpočet polohy a veľkosti priestorového vektora statorového napätia. Pre napäťový striedač so šiestymi spínacími prvkami existuje osem základných polôh vektorov napätia. Šesť vektorov s nenulovými hodnotami $\boldsymbol{u_{s1}}(1, 0, 0) – 0°$, $\boldsymbol{u_{s2}}(1, 1, 0) – 60°$, $\boldsymbol{u_{s3}}(0, 1, 0) – 120°$, $\boldsymbol{u_{s4}}(0, 1, 1) – 180°$, $\boldsymbol{u_{s5}}(0, 0, 1) – 240°$, $\boldsymbol{u_{s6}}(1, 0, 1) – 300°$ a dva vektory s nulovými hodnotami $\boldsymbol{u_{s0}}(0, 0, 0)$, $\boldsymbol{u_{s7}}(1, 1, 1)$. Absolútne hodnoty vektorov $\boldsymbol{u_{s1}}(1, 0, 0)$ až $\boldsymbol{u_{s6}}(1, 0, 1)$ je *2/3* U_{DC}, kde U_{DC} je napätie jednosmerného medziobvodu. Jednotlivé trojice čísiel vektorov napätia sú zástupcom logických hodnôt zopnutých vetiev striedača. Prvé číslo trojice príslušných vetiev označuje fázu *C*, druhé *B* a tretie *A*. Zo žiadanej hodnoty polohy vektoru napätia sa určuje príslušná kombinácia zopnutých a vypnutých tranzistorov, veľkosť napätia udáva doba zopnutia, resp. vypnutia počas periódy.

Behom jednej periódy vzorkovania sa dá výstupný vektor napätia popísať rovnicou

$$\boldsymbol{u_s} = \frac{t_0}{T_s}\boldsymbol{u_{s0}}(0,0,0) + \frac{t_1}{T_s}\boldsymbol{u_{s1}}(1,0,0) + \frac{t_2}{T_s}\boldsymbol{u_{s2}}(1,1,0) + \cdots + \frac{t_7}{T_s}\boldsymbol{u_{s7}}(1,1,1) \qquad (7.1)$$

Keď časy t_0 ... t_7 sú doby zopnutia jednotlivých vektorov napätia. Platí, že ich súčet je rovný perióde vzorkovania T_s.



Obr. 9 Vektory napätia[10]

Z obrázku 9 je zrejmé, že celá oblasť, v ktorej sa môže nachádzať vektor u_s je rozdelená do šiestich sektorov. V každom sektore sa okrem nulových vektorov, využívajú vždy dva priľahlé vektory k požadovanému vektoru $\boldsymbol{u_s}$. Na obrázku 9 ako príklad je ukázaný $\boldsymbol{u_s}$ ležiaci v prvom sektore, a jeho rozklad do smeru vektorov $\boldsymbol{u_{s1}}(1, 0, 0)$ a $\boldsymbol{u_{s2}}(1, 1, 0)$. Vektor napätia sa dá potom vyjadriť rovnicou

$$\boldsymbol{u_s} = \frac{t_1}{T_s}\boldsymbol{u_{s1}}(1,0,0) + \frac{t_2}{T_s}\boldsymbol{u_{s2}}(1,1,0) + \frac{t_0}{T_s}\boldsymbol{u_{s0}}(0,0,0) + \frac{t_7}{T_s}\boldsymbol{u_{s7}}(1,1,1) \tag{7.2}$$

Je znateľné, že rozklad bude pre jednotlivé sektory definovaný inak. Preto sa pri použití metódy modulácie priestorového vektoru musí najprv previesť výber aktuálneho sektoru.

Časy t_1 a t_2 sa dajú určiť pomocou sínusovej vety

$$\frac{\sin 120°}{|\boldsymbol{u_s}|} = \frac{\sin(60° - \varphi)}{|\boldsymbol{u_{s1}}(1,0,0)|\frac{t_1}{T_s}} \tag{7.3}$$

$$t_1 = \sqrt{3}\,\frac{|\boldsymbol{u_s}|}{U_{DC}}\,T_s\,\sin(60° - \varphi) \tag{7.4}$$

$$t_2 = \sqrt{3}\,\frac{|\boldsymbol{u_s}|}{U_{DC}}\,T_s\,\sin\varphi \tag{7.5}$$

Závislosť odvodené pre prvý sektor platí aj v ďalších sektoroch. Stačí len postupne otáčať vektorov po 60°. Môžeme tak obecne vyjadriť pre ľubovoľný sektor v tvaru

$$\boldsymbol{u_s} = \frac{t_{\mathrm{k}}}{T_s}\boldsymbol{u_{sk}}(,,) + \frac{t_{\mathrm{k+1}}}{T_s}\boldsymbol{u_{sk+1}}(,,) + \frac{t_0}{T_s}\boldsymbol{u_{s0}}(1,1,1) + \frac{t_7}{T_s}\boldsymbol{u_{s7}}(0,0,0) \tag{7.6}$$

$$t_k = \sqrt{3}\,\frac{|\boldsymbol{u_s}|}{U_{DC}}\,T_s\,\sin(60° - \varphi) \tag{7.7}$$

$$t_{k+1} = \sqrt{3}\,\frac{|\boldsymbol{u_s}|}{U_{DC}}\,T_s\,\sin\rho \tag{7.8}$$

$$T_s - t_1 - t_2 = t_0 + t_7 \tag{7.9}$$

Pomer $\sqrt{3}\,|\boldsymbol{u_s}|/U_{DC}$ vyjadruje hĺbku modulácie, ktorá maximálne môže byť rovnou jednej. V tomto prípade sa konec vektoru $\boldsymbol{u_s}$ pohybuje po kružnice**.**

Medzná hodnota, do ktorej je schopná striedač pracovať je podmienená vzťahom

$T_s = t_1+t_2$. V tomto prípade oba nulové vektory sú nepripojené a sú nulové. Po aplikácii tejto podmienky pre všetky sektory dostaneme šesťuholník. Keď hodláme mimo tohto šesťuholníka rozložiť napäťový vektor, nastane situácie ktorá je neuskutočniteľná $t_1+t_2 > T_s$.

Text kapitoly bol vytvorený podľa [1][10]

8 PRÁCA V PROGRAME LABVIEW

Programovacie a vývojové prostredie LabVIEW (z angl. *Laboratory Virtual Instruments Engineering Workbench*) - „laboratórne pracovisko virtuálnych nástrojov“ slúži nielen k programovaniu systémov pre meranie a analýzu signálov, ale aj na riadenie a simuláciu technologických procesov.

Užívateľské rozhranie programu v LabVIEW sa často podobné reálnym prístrojom, preto program nazývame ako virtuálny prístroj. Anglický ekvivalent je VI (*Virtual Instrument*).

Každý VI obsahuje tri časti.

- Čelný panel - slúži ako užívateľské rozhranie.
- Blokový diagram - je zdrojovým kódom, ktorý riadi VI.
- Ikon / Konektor - spája jednotlivé VI medzi sebou.

8.1 Úvod do programovania v LabVIEW

Otvorením nového prázdneho VI sa dostaneme do vývojového prostredia (*Development Enviroment*) LabVIEW ktorý ponúka niekoľko okien. Jednotlivé okná sú štrukturálne viazané. Aktivovaním okna čelného panelu sa zobrazí paleta *Controls* a pri aktivovaní okna blokového diagramu sa zobrazí paleta *Functions*. Paleta *Tools* je nezávisle aktívna od zvolených okien.

Užívateľské rozhranie je poskladané zo sady nástrojov a objektov. Každý vložený nástroj na čelnom panely má svoj ekvivalentný objekt v blokovom diagrame. Ovládacie prvky sú použité pre vstupy ako je posuvník na nastavenie hodnoty, prepínač – spínač na zmenu stavu, textové pole na zadanie hodnoty atď. Indikátory sú použité ako výstupy, napríklad ručičkové prístroje, kontrolné indikátory, diagramy zobrazujúce výstupné hodnoty z programu.

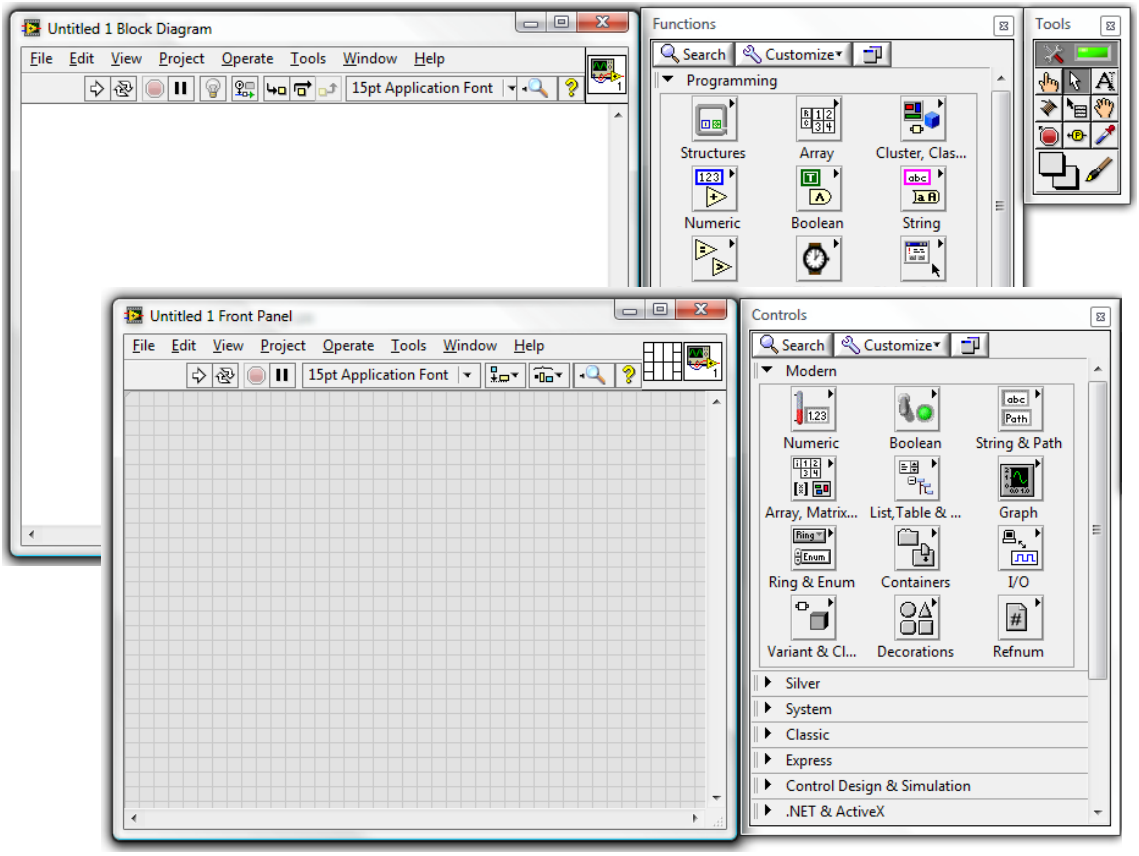
Pospájaním objektov a pridaním ďalších funkcií v blokovom diagrame sa vytvorí program. Keď v spustenom programe užívateľ preladí vstupné hodnoty, zmenené výsledky sa objavia na indikátoroch v reálnom čase.

Paleta *Controls* má stromovú štruktúru. Nachádzajú sa tu všetky objekty ktoré je možné používať v okne čelného panelu.

Paleta *Functions* tvorí stromová štruktúra objektov ktoré sa dajú použiť v okne blokového diagramu.

Keď sa v ďalšom budeme odkazovať na niektorý objekt z palety, uvedieme jeho polohu v palete postupnosti názvu jednotlivých úrovní palety oddelených šípkami. Na konci celej postupnosti uvedieme názov konkrétneho objektu.

Nakoľko LabVIEW je grafický programovací jazyk, všetky akcie sú realizované ovládaním myši. Kurzor funguje v niekoľkých režimoch, ktoré môžeme predvoliť v palete *Tools*.



Obr. 10 Rozhranie LabVIEW

8.2 Rozčlenenie palety Functions

Paleta *Functions* obsahuje veľké množstvo rôznych objektov. Veľmi často však budeme používať len niekoľko z nich, ostatné pravdepodobne len zriedkavo.

Pri rozsiahlejšom simulačnom programe je potrebné si navoliť objekty zo subpalety *Programming*. V *Programming*-u nižšiu úroveň stromovej štruktúry tvoria subpalety: *Numeric*, *Boolean*, *Array*, *Cluster*, *Comparison* atď. ktoré slúžia ako základné skladačky na zhotovenie VI.

Pre vytvorenie simulácie algoritmov riadenie elektrických pohonov je potrebné ešte si vybrať špecifické subpalety z palety *Functions*. Z ponúknutých možností túto požiadavku spĺňa subpaleta *Control Design & Simulation*.

Tab. 1 Subpaleta Control Design and Simulation

Subpaleta Control Design & Simulation		
Subpalta	**Ikona**	**Popis**
Control Design VIs and Functions		Slúžia na vytvorenie, analýzu a na rozvinutie dynamický model systémov.
Fuzzy Logic VIs		Slúži na návrh a riadenie fuzzy systémov. Na

		interaktívny návrh fuzzy systémov sa dá použiť aj *Fuzzy System Designer*.
PID VIs		Obsahuje rôzne druhy PID regulátorov a bloky prídavnou funkcionalitou.
Simulation VIs and Functions		Slúžia na vytvorenie simulačných programov v LabVIEW.
System Identification VIs		Slúžia na vytvorenie a stanovenie matematického modelu dynamického systému. Umožňujú estimáciu matematický model systému podľa pozorovania vstup – výstupných dát.

Tab. 2 Subpaleta Simulation

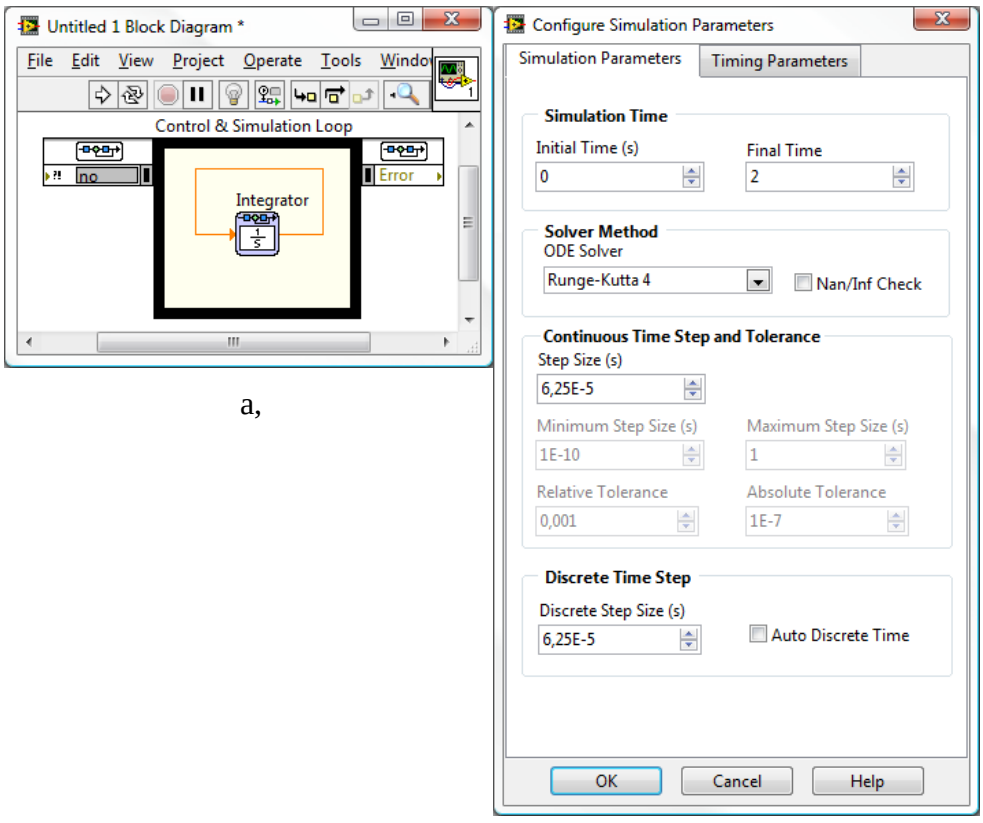
Subpaleta *Simulation*		
Subpalta	**Ikona**	**Popis**
Continous Linear Systems Functions		Tieto bloky slúžia na reprezentáciu spojitých lineárnych systémov s rôznymi rovnicami v simulačnom diagrame.
Discrete Linear System Functions		Tieto bloky slúžia na reprezentáciu diskrétnych lineárnych systémov s rôznymi rovnicami v simulačnom diagrame.
External Models Functions		Tieto bloky slúžia na prístup k modelu vytvorené treťou osobou. Napríklad modely distribuovaných pomocou National Insrtuments Alliacing Partner. Okrem toho sa požívajú aj na simuláciu modelov vytvorených v jazyku C využitím EMI funkcie.
Graph Utilities Functions		Tieto bloky slúžia na zobrazenie výstupov simulácie na graf alebo diagram .
Implicit Systems Functions		Tieto bloky slúžia na výpočet diferenčných rovníc v simulačnom diagrame.
Nonlinear Systems Functions		Tieto bloky slúžia na simuláciu nelineárne javy ktoré sa nachádzajú v reálnom svete.
Optimal Design VIs		Tieto bloky slúži na stanovanie optimálne parametre modelu systému zadané pomocou kritériálnych funkcií (IE, IAE, ITE, ITAE ...) a množinou obmedzenia.
Signal Arithmetic Functions		Tieto bloky slúžia na vykonanie základných aritmetických operácií na signáloch a simulačných systémoch.
Signal Generator Functions		Tieto bloky slúžia na generovanie špecifických priebehových vzorov a rozdelenie ich do vektoru.
Trim & Linearize VIs		Tieto bloky slúžia na zjednodušenie a linearizáciu simulačných subsystémov.

8.3 Vytvorenie riadiaceho a simulačného diagramu v LabVIEW

Na vytvorenie simulačného diagramu sa vloží riadiaci a simulačný rám (*Control & Simulation Loop*). Tento rám obsahuje parametre ktoré ovplyvnia simuláciu, určí ktoré riešenie diferenciálnych rovníc (*ODE solver*) sa má používať a ako dlho má trvať simulácia.

Postup vloženia riadiaceho a simulačného rámu: Otvoríme nový prázdny VI, prepneme do blokového diagramu kde zvolíme *View* → *Functions Palette* → *Control Design & Simulation* → *Simulation*. Klikneme na ikonu *Control & Simulation Loop* a vložíme rám. S dvojitým kliknutím do ľavého horného rohu rámu sa nám objaví dialógové okno *Configure Simulation Parameters*. Tu máme možnosť navoliť si presné vlastnosti simulácie ako je čas, dĺžka kroku, spôsob ODE. Druhým spôsobom je programovo nastaviť tieto parametre, tak že privedieme hodnoty do *Input Node*.

Väčšina funkcií z *Functions Palette* → *Control Design & Simulation* sa dajú používať len vnútri riadiaceho a simulačného rámu podfarbené so žltou farbou.



a,

b,

Obr. 11 a - Simulation Loop, b – Configure Simulation Parameters

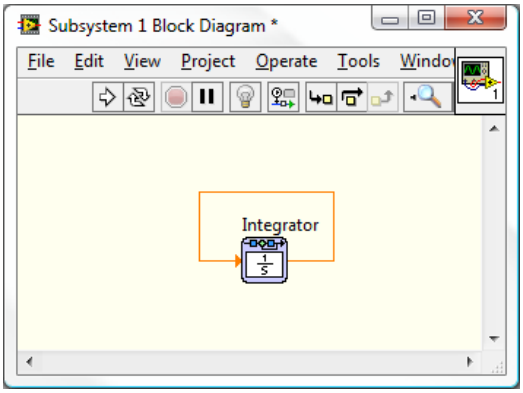
8.4 Vytvorenie subVI

Jednotlivé vytvorené VI v LabVIEW sa ukazujú ako samostatné aplikácie. Vložením jednotlivých VI do seba ktoré sú súčasťou iného VI vytvoríme podprogramový subVI. Aby VI mohlo slúžiť ako subVI je potrebné vybaviť ho vstupnými a výstupnými portami a je účelné spracovať graficky ikonu tak, aby reprezentovala obsiahnuté vlastnosti. V blokovom diagrame vystupuje subVI ako samostatný blok, ktorý je vybavený vstupnými a výstupnými portami a ikonou.

8.5 Simulačný subVI

Je podobný ako subVI, a taktiež môže byť použité ako samostatné VI. Vytvorí sa navolením *File → New → Other Files → Simulation Subsystem*. Založí sa nový subsystém so žltou plochou, ktorá vyjadruje že sa nachádzame v simulačnom prostredí. Parametre simulácie sa nastavujú priamo v hlavnom menu: *Operate → Configure Simulation Parameters*, nakoľko už sa nachádzame vnútri v simulačnom prostredí a nemôžeme kliknúť na simulačný rám.

Postup vytvorenia modelu je rovnaký ako by sme mali vo VI umiestnený simulačný rám cez celú plochu. Vstupy a výstupy sa určujú ako u klasickej subVI. Pomocou terminálov ikony v pravom hornom rohu v čelnom paneli postupne prepojíme jednotlivé termináli s príslušnými objektmi čelného panelu. Keď už máme vytvorený simulačný subVI môžeme spustiť alebo ho vložiť do simulačného rámu iného VI. Vloženie subsystému sa vykoná ako vloženie klasického subVI, s tým rozdielom že simulačný subsystém sa musí vložiť priamo do vnútra simulačného rámu, a s tým automaticky dedí všetky predvolené simulačné vlastnosti nadriadeného simulačného rámu, ako sú napríklad čas trvania simulácie a spôsob výpočtu diferenciálnych rovníc.



Obr. 12 Simulačný subVI

Do subsystémov nie je možné vkladať breakpointy ani sa do nich nedá prekrokovať. Na také breakpointy program ani nezastavuje.

Rozdiel oproti klasickému subVI je, že objekty uložené v subsystémoch prevedú svoju činnosť okamžite keď majú k dispozícií všetky vstupy aj v prípade, že subsystém všetky vstupy nemá.

8.6 Dátové typy a spojenia

Každý objekt v blokovom diagrame má svoje vstupy a výstupy, ktoré môžeme navzájom prepojiť. Dátové typy vyjadrujú aké objekty, vstupy a výstupy sú prepojiteľné. Spojenia sú v rovnakej farby ako sú porty. Keď posuvník má oranžový port môžeme ho prepojiť s ľubovoľným vstupom s oranžovým znakom, avšak nemôžeme ho prepojiť so vstupom so zeleným znakom.

Tab. 3 Bežné type spojenia

Typ spojenia	Skalár	1D Pole	2D Pole	Farba
Numeric				Oranžová – floating point Modrá - integer
Boolean				Zelená
String / Text				Ružová

Dátové typy vyjadrujú aké vstupy, výstupy a objekty môžeme prepojiť. Každé spojenie má jediný zdroj dát, ale môže vtiecť do viacerých VI a funkcií, ktoré tieto dáta spracujú, takže treba dodržať poradie spojenia. Väčšinou všetky vstupy blokov sú na ľavej strane, a výstupy sú osadené do pravej časti, čo znamená že spojenie blokov sa odohráva v smere čítania od ľavej strany k pravej strane. Pretože v simulačnom diagrame mnohokrát používame spätné väzby a bloky ktoré sa nachádzajú v podzložke *Control Design & Simulation* vo *Functions Palette* je možné ich preklopiť okolo zvislej osi, a tak prehodiť umiestnenie vstupov a výstupov, čo uľahčí našu prácu pri pospojovaní blokov.

Spojenia majú rôznu hrúbky, štýly a farby, ktoré závisia od dátového typu. Nevhodné spojenie sa zobrazí čiarkovanou čiarou v strede s červeným X. V takomto prípade porty jednotlivých objektov nie sú spájateľné.

8.7 Externý model dynamického systému v LabView

Externý model vytvorený v jazyku C pomocou funkcií *External Model Interface* (ďalej len EMI), sa dajú používať v LabView *Control and Simulation module*.
Postup vytvorenia EMI môžeme členiť na 3 časti

- Definícia modelu s použitím funkcií EMI v C
- Vytvorenie dynamicky linkovanej knižnice DLL z projektu C
- Odkazovať v LabView s použitím funkcie *External Model* na knižnicu

8.8 Tvorba externého modelu

Ako vývojové prostredie a prekladač C sme použili Microsoft Visual Studio 2011. Vytvorili sme nový prázdny projekt *Win32 Console Application*, kde sme zvolili typ aplikácie DLL.

V každom modely musí byť implementovaná sada funkcií takzvaných *Callback* funkcií, jednoduchšie *Callbacks*, ktoré LabView vyvolá počas simululácie modelu.

Pred samotným používaním Callback funkcií však sa musí naiklúdovať hlavičkový súbor SIM_EMI_API.h, čo sa nachádza v priečinku labview\CCodeGen\Simulation. Táto hlavička obsahuje definície a deklarácie pre všetky funkcie a dátové typy čo sú potrebné v EMI. Potom nasleduje nastavenie modelu pomocou *Callback* funkcií EMI. Kvôli rozoznaniu funkcií *Callback* od ostatných sú označené skratkou CB.

8.8.1 EMI_CB_ModelInterface

Definuje a alokuje časti externého modelu ako sú vstupy, výstupy, parametre a názov modelu. Vstupy, výstupy a parametre môžu byť skalárne, alebo vektorové. Ich maximálny počet nemôže prekročiť 25. Indexovanie vstupov (resp. výstupov, parametrov) prebieha v poradí v akom boli zadefinované.

Počas implementácie modelu sme zistili jednu slabosť a to pri deklarácii vstupov. Najprv sa musia zadefinovať všetky skalárne vstupy a až potom sa smú definovať vektorové vstupy. Striedanie poradia definovania vektorových vstupov (napr. zadanie skalára, vektora a znova skalára) znamená nepredvídateľnú zmenu chovania modelu. Tento nedostatok bol nahlásený u National Instruments, ale do doby písania prace sme sa nedočkali odpovede.

8.8.1.1 Nastavenie názov modelu

```
EMI_SetModelName(model, "My Example");
```

8.8.1.2 Deklaráciu vstupov

`EMI_AddScalarInput(model, "First Input");` - pre skalárny vstup
`EMI_AddVectorInput(model, "Second Input", 3);` - pre vektorový vstup, kde ešte udávame počet elementov vo vektoru

8.8.1.3 Deklarácia výstupov:

`EMI_AddScalarOutput(model, "First Output", EMI_Feedthrough_Direct);` - skalár
`EMI_AddVectorOutput(model, "Second Output", EMI_Feedthrough_Indirect, 3);` - vektor
`EMI_Feedthrough_Direct / EMI_Feedthrough_Indirect` - datový typ ktorý špecifikuje či na výstup modelu je priamo / nepriamo prepojený vstup modelu.

8.8.1.4 Deklarácia parametrov:

Pri deklarácii parametrov je nutné ešte zadať počiatočné hodnoty.

```
double initVals[3] = {1.0, 2.0, 3.0};
EMI_AddScalarParam(model, "First Param", EMI_ParamSource_ConfigPage, 5.0);
EMI_AddVectorParam(model, "Second Param", EMI_ParamSource_ConfigPage, initVals, 3);
```

- skalár

- vektor

8.8.2 EMI_CB_InitializeModel

Špecifikuje počet spojitých a diskrétnych stavov externého modelu

```
EMI_SetNumberOfContinuousStates(model, 2);
```
- počet spojitých stavov

```
EMI_SetNumberOfDiscreteStates(model, 2);
```
- počet diskrétnych stavov

8.8.3 EMI_CB_CalculateDerivatives

Vypočíta derivácie spojitých stavov externého modelu. Táto funkcia musí byť zadefinovaná vtedy, keď model má spojité stavy.

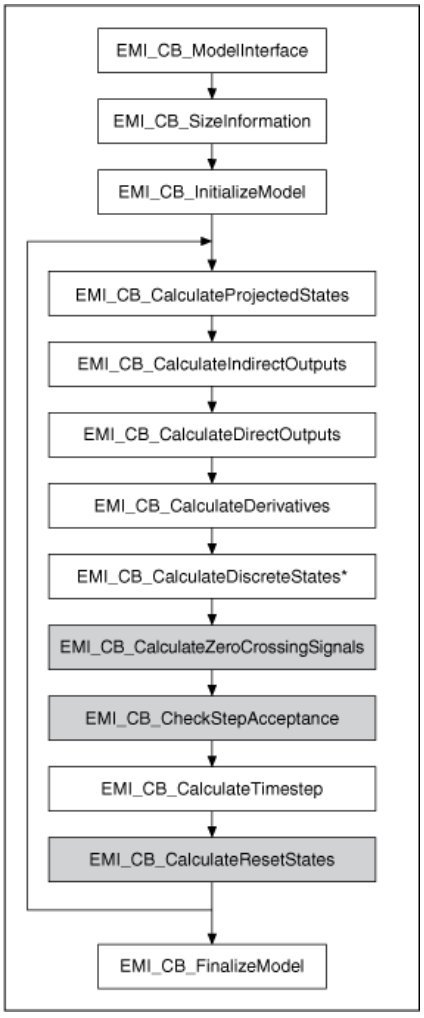
8.8.4 EMI_CB_CalculateIndirectOutputs

Vypočíta výstupy externého modelu, ktoré nie sú priamo prepojené na vstup. Táto funkcia musí byť zadefinovaná, keď model má výstup nepriamo prepojený na vstup.

Predchádzajúci zoznam neobsahuje všetky callback funkcie. Úplný súpis sa nachádza v literatúre [2].

Ako príklad na použitie externého modelu, sme vytvorili model asynchrónneho motora, ktorému zdrojový kód je uvedené v prílohe číslo 5.

Labview Control Design and Simulation Module vykonáva Callback funkcie v určitom poradí čo je znázornená na nasledujúcom obrázku.

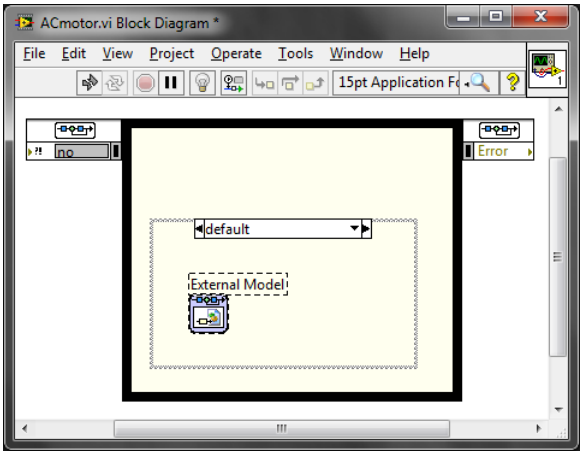


Obr. 13 Poradie volania Callback funkcií [2]

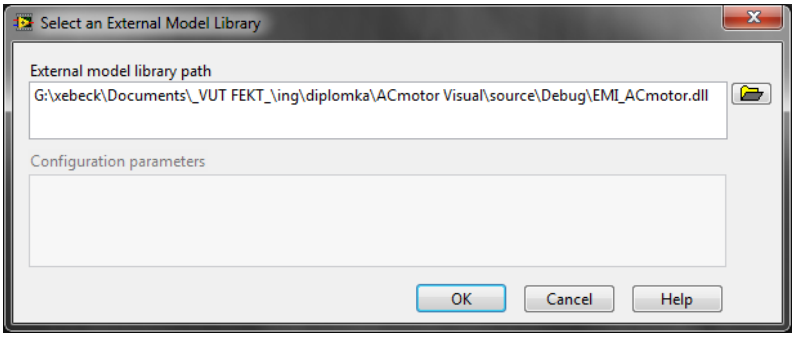
8.9 Použitie externého modelu.

Na simuláciu externého modelu sa používa funkčný blok *External Model*, ktorý sa nachádza v *Functions → Control Design and Simulation → Simulation*.

Po pridaní funkciu *External Model* do simulačného rámu (*Simulation Loop*), je ponúknuté dialógové okno *Select an External Model Library* na výber cesty k žiadanej knižnici DLL.

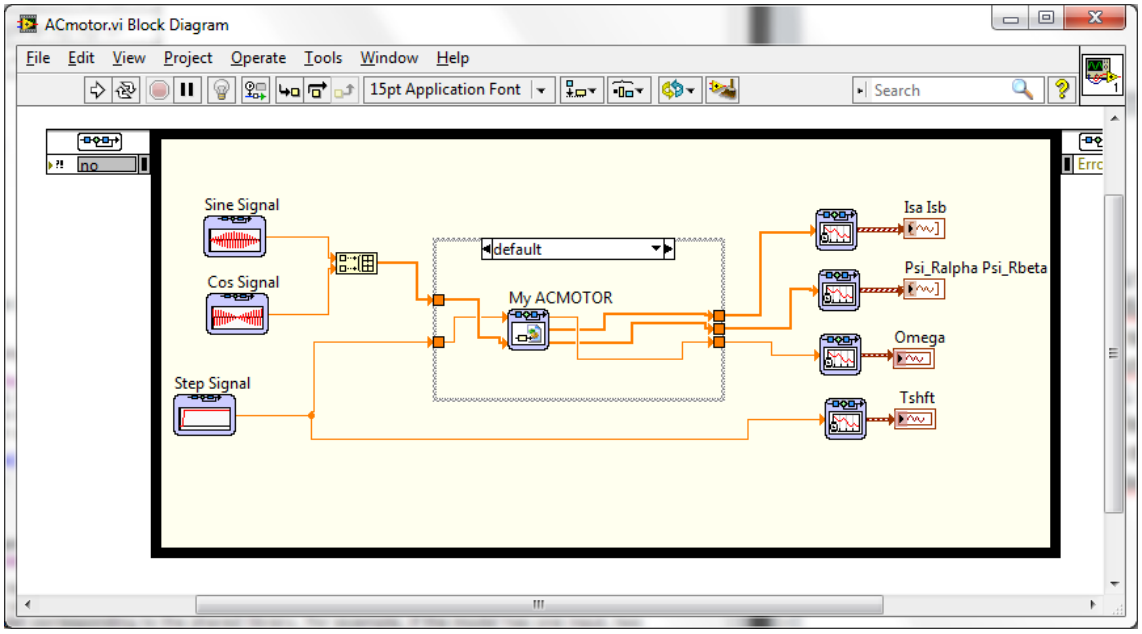


Obr. 14 Funkčný blok External Model

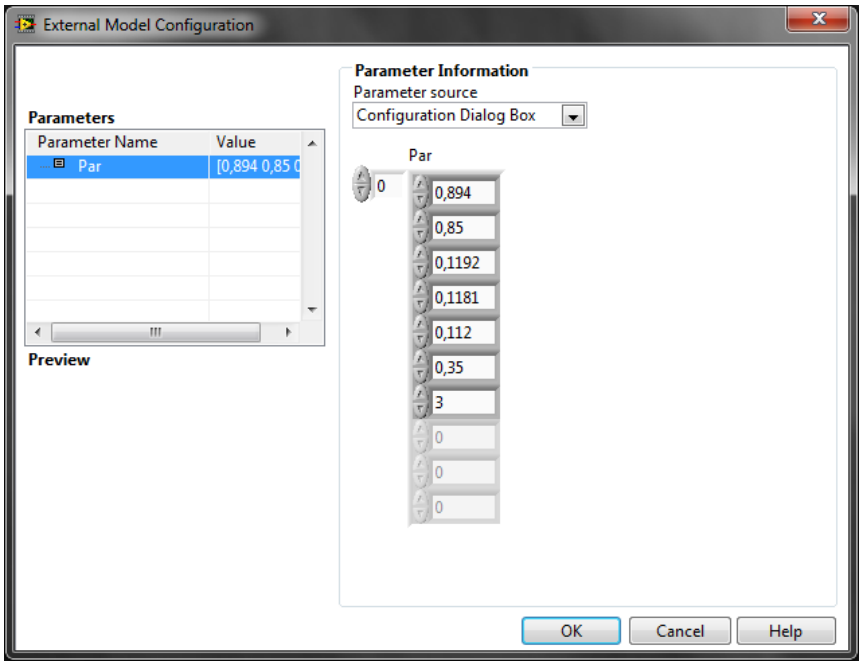


Obr. 15 Dialógové okno: Select an External Model Library

Funkcia *External Model* sa automaticky aktualizuje aby sa vyhovoval štruktúre knižnice DLL. Napríklad keď externý model obsahuje dva vstupy, jeden parameter a tri výstupy, tak funkcia External Model bude mať dva blokové diagramové vstupy, jeden parameter nastaviteľený v *Configuration dialog box*-e, a tri blokové diagramové výstupy.

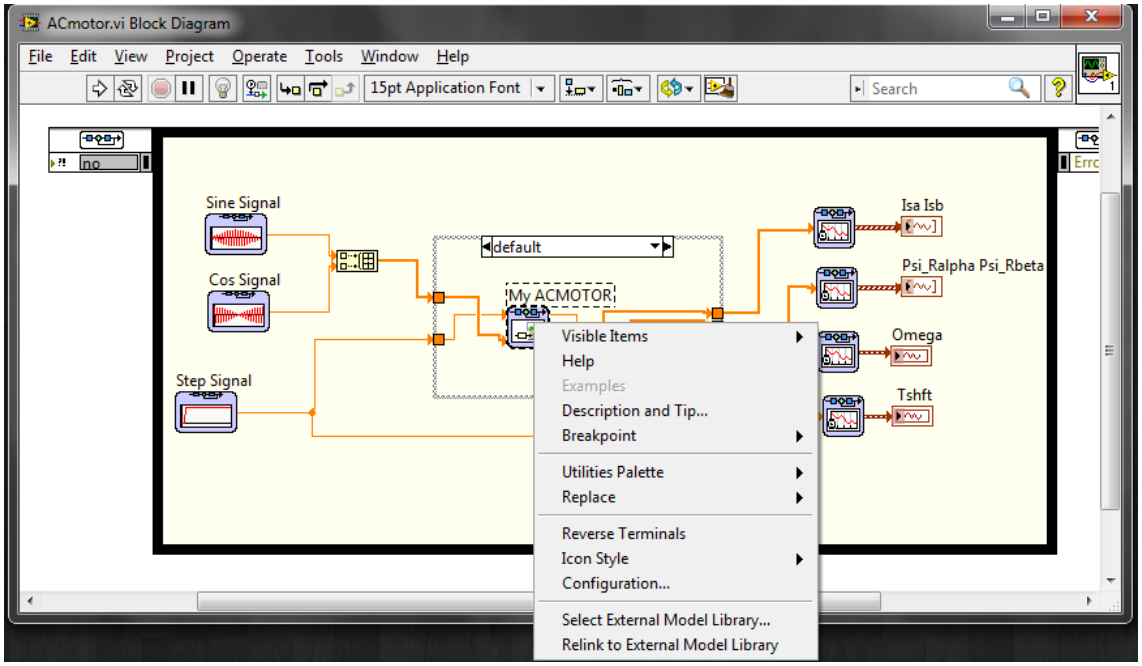


Obr. 16 Pripojené vstupy a výstupy modelu



Obr. 17 Configuration dialog box

Keď sa zmení knižnica DLL, na ktorú sa odkazuje funkcia *Externel Model*, tak kliknutím pravým tlačítkom myši na funkciu *External Model* a zvolením *Relink to External Model Library*, opäť sa načíta knižnica DLL z vopred zvolenej cesty.



Obr. 18 Aktualizácia knižnice DLL

V prípade, keď mienime použiť iný externý model, tak kliknutím pravým tlačítkom myši na funkciu *External Model* a po zvolení *Select an External Model Library* cez otvorené dialógové okno, môžeme si vybrať cestu k potrebnej knižnici DLL.

Pomocou týchto postupov sme vytvorili externý model asynchrónneho motora pomocou EMI funkcií v programovacom jazyku C, uvedené v prílohe č 5. Výsledný časový priebeh simulácie spolu s priebehom z Matlabu sú v prílohe č. 3, 4.

8.10 Implementácia algoritmov riadenia elektrických motorov do prostredia LabView Control Design and Simulation

Pri vytváraní simulačného algoritmu v programe LabVIEW sme vychádzali z daného vzorového simulačného programu v Matlab Simulink. Počas celej práci nám slúžilo ako vodítko k docieleniu správnej funkčnosti našej simulácie.

Podľa stromovej štruktúry blokovej schémy v Matlab Simulink sme vytvorili ekvivalentnú blokovú štruktúru v LabVIEW. Pristúpili sme k samotnej tvorbe hlavného VI. Rozhodli sme si najprv vložiť do čelného panelu potrebné ovládacie prvky a potom tvoriť blokový diagram. Bolo potrebné si rozmyslieť z akého dátového typu budú skladané jednotlivé ovládacie prvky a aký dátový typ budeme používať na spojenie nami vytvorených subsystémov. Počas vytvorenia prvého simulačného subsystému sme sa dostali k poznaniu, že je lepšie vychádzať z blokového diagramu, kde sú priamo dané dátové typy vstupov a výstupov. Najvhodnejšia metóda vytvorenia stromovej štruktúry

subsystémov je vychádzať z najnižšej úrovne. Zvolením a pospojovaním základných blokov z palety *Functions* sme zostrojili algoritmus, z čoho sú podmienené dátové typy vstupov a výstupov. Takto vyhotovený prvý subVI na najnižšej úrovni sme pomenovali a postúpili na vyššiu úroveň. Pripojením k tomuto bloku ďalších základných blokov, respektíve subsystémov sa nám sformuloval ďalší subVI o úroveň vyššie. Postupovaním s touto metódou sme dosiahli úroveň najvyššieho bodu, hlavnej VI (_Main.vi).

Štruktúra algoritmu riadenia asynchrónneho motora je znázornená na obrázku v prílohe č. 6. Výsledné priebehy simulácie sú v prílohe č.25 spolu s priebehmi z Matlabu v prílohe č.26.

Štruktúra algoritmu riadenia synchrónneho motora s permanentnými magnetmi je znázornená na obrázku v prílohe č. 27. Výsledné časové priebehy simulácie sú v prílohe č.38 spolu s priebehmi z Matlabu v prílohe č.39.

Tab. 4 Zoznam nami vytvorených subVI

Názov subVI	**Ikona**	**Popis**
ACIM FOC		AC Induction Motor Field Oriented Control. Blok obsahuje všetky potrebné bloky vektorového riadenia asynchrónneho motora. **Vstupy:** U_DCB: napätie na jednosmernom medziobvode meniča Psi_Rd_ref: žiadaná hodnota magnetického toku Omega_ref: žiadaná hodnota uhlovej rýchlosti motora I_ABC: namerané fázové prúdy motora Omega: nameraná uhlová rýchlosť motora **Výstupy**: U_ABC: trojfázové napájacie napätie motora I_s_dq: prúd statora v d-q súradniciach Duty_ABC: hodnoty duty cycle pre fáze A,B,C
Flux controller		PI regulátor magnetického toku rotora. **Vstupy**: Psi_d_ref: žiadaná hodnota magnetického toku motora Psi_d: skutočná hodnota magnetického toku motora f_limit: hodnota na ktorú sa obmedzuje akčný zásah **Výstupy**: f_u: akčný zásah regulátora **Parametre**: f_kp: hodnota zosilnenie proporcionálnej časti regulátora f_ki: hodnota zosilnenie integračnej časti regulátora

Speed controller		PI regulátor otáčok motora. **Vstupy**: Omega_ref: žiadaná hodnota otáčok motora Omega: skutočná hodnota otáčok motora f_limit: hodnota, na ktorú sa obmedzuje akčný zásah **Výstupy**: f_u: akčný zásah **Parametre**: f_kp: hodnota zosilnenia proporcionálnej časti regulátora f_ki: hodnota zosilnenia integračnej časti regulátora
Id controller		Regulátor d zložky prúdu. **Vstupy**: I_sd_ref: žiadaná hodnota d zložky prúdu statora motora I_sd: skutočná hodnota d zložky prúdu statora motora f_limit: hodnota, na ktorú sa obmedzuje akčný zásah f_u_comp: hodnota d zložky napätia po odstránení väzieb, ktorá sa sčíta s výstupom regulátora **Výstupy**: f_u: akčný zásah regulátora **Parametre**: f_kp: hodnota zosilnenia proporcionálnej časti regulátora f_ki: hodnota zosilnenia integračnej časti regulátora
Iq controller		Regulátor q zložky prúdu. **Vstupy**: I_sq_ref: žiadaná hodnota q zložky prúdu statora motora I_sq: skutočná hodnota q zložky prúdu statora motora f_limit: hodnota, na ktorú sa obmedzuje akčný zásah f_u_comp: hodnota d zložky napätia po odstránení väzieb, ktorá sa sčíta s výstupom regulátora **Výstupy**: f_u: akčný zásah **Parametre**: f_kp: hodnota zosilnenia proporcionálnej časti regulátora f_ki: hodnota zosilnenia integračnej časti regulátora
Subsystem1		Sumarizačná časť regulátorov PI. **Vstupy**: Input1: vstup do subsystému Upper limit: horná hranica obmedzenia výstupu Lower limit: dolná hranica obmedzenia výstupu

		Výstupy: Output1: výstup subsystému
iq_limit		Slúži na výpočet obmedzenia akčného zásahu regulátora otáčok. **Vstupy**: f_i_max: maximálna povolená hodnota prúdu f_i_d_lim: žiadaná hodnota regulátora d zložky prúdu **Výstupy**: i_q_max: maximálna povolená hodnota prúdu q zložky statora
uq_limit		Slúži na výpočet obmedzenia akčného zásahu regulátora q zložky prúdu. **Vstupy**: f_u_max: maximálna povolená hodnota napätia na jednosmernom medziobvode meniča f_u_d_lim: akčný zásah regulátora d zložky prúdu **Výstupy**: u_q_max: maximálna povolená hodnota q zložky napätia
alpha_beta to DQ		Vykoná transformáciu zo súradnicového systému α,β do súradnicového systému d,q. Parkova transformácia. **Vstupy**: alpha,beta: veličina v súradnicovom systéme α,β phi: uhol natočenia medzi súradnicovými systémami **Výstupy**: dq: veličina v súradnicovom systéme d,q
DQ to alpha beta		Vykoná transformáciu zo súradnicového systému α,β do súradnicového systému d,q. inverzná Parkova transformácia. **Vstupy**: dq: veličina v súradnicovom systéme d,q phi: uhol natočenia medzi súradnicovými systémami **Výstupy**: alpha,beta: Veličina v súradnicovom systéme α,β
3 to 2		Prevedie tri osi dvojdimenzionálného súradnicového systému statoru do dvoch os (α, β). Clarkovej transformácia. **Vstupy**: ABC: veličina v súradnicovom systéme ABC **Výstupy**: alpha,beta: veličina v súradnicovom systéme α, β

2 to 3		Vykoná transformáciu z osí α, β do troch osí dvojdimenzionálného súradnicového systému. **Vstupy**: alpha,beta: veličina v súradnicovom systéme α, β **Výstupy**: ABC: veličina v súradnicovom systéme ABC
PWM block		Vypočíta časy zopnutia výkonných tranzistorov meniča. **Vstupy**: U_dc: maximálne napätie na jednosmernom medziobvode meniča U_S: napájacie napätie statoru motora v súradnicovom systéme α, β **Výstupy**: U: trojfázové napájacie napätie statora motoru duty: hodnoty duty cycle pre fáze A,B,C
Space vect pwm		Vypočíta moduláciu vektorového priestoru. **Vstupy**: U_dc: maximálne napätie na jednosmernom medziobvode meniča U_S: napájacie napätie statoru motora v súradnicovom systéme α, β **Výstupy**: pwm: vygenerovaný priebeh PWM
Average bridge		Normalizuje duty cycle a prevedie na napätie. **Vstupy**: Duty a,b,c: vygenerovaný priebeh PWM **Výstupy**: Va,Vb,Vc: trojfázové napätie ABC duty: hodnoty duty cycle pre fáze A,B,C
Flux estimator		Estimátor polohy vektora magnetického toku. **Vstupy**: Us: napätie statoru v α, β súradniciach Is: nameraný prúd statoru v α, β súradniciach Omega: merané otáčky motora **Výstupy**: ^Is: estimovavaný prúd statora v α, β súradniciach ^PSIs: estimovaný magnetický tok rotora v α, β súradniciach ^PSIr: estimovaný magnetický tok statora v α, β súradniciach

Decoupling Asynch		Zruší väzby a rozdeľuje asynchrónny motor na dve časti, na momentotvornú a tokotvornú. **Vstupy**: Omega_est: estimovaná uhlová rýchlosť Psi_rd: q zložka magnetického toku rotora Omega_field: synchrónna uhlová rýchlosť i_sdq: prúd statora v d,q súradniciach **Výstupy**: u_sdq_comp: nezávislé komplementárne napätie
Decoupling PMSM		Zruší väzby a rozdeľuje asynchrónny motor na dve časti, na momentotvornú a tokotvornú. **Vstupy**: omega: elektrická uhlová rýchlosť idq: prúd statora v d,q súradniciach **Výstupy**: udq_e: nezávislé komplementárne napätie
AC MOTOR		Náhradný blok chovania skutočného asynchrónneho motora. **Vstupy**: U_ABC: trojfázové napájacie napätie Tshft: zaťažovací moment **Výstupy**: I_ABC: merané fázové prúdy Psi_rab: magnetický tok rotora v súradnicovom systéme α, β
PMSM		Matematický model synchrónneho motora s permanentnými magnetmi v súradnicovom systéme d,q. **Vstupy** Udq: napájacie napätie motora v d,q súradniciach ML: zaťažovací moment motora **Výstupy** Idq: Merané prúdy v d,q súradniciach PSIdq: magnetický tok motora v d,q súradniciach Omega_m: mechanické otáčky motora Me: elektromagnetický moment motora
Synchronous motor		Náhradná schéma chovania skutočného synchrónneho motora s permanentnými magnetmi. **Vstupy** Uabc: trojfázové napájacie napätie motora ML: zaťažovací moment motora

		Výstupy: Iabc: Merané prúdy v d,q súradniciach PSIab: magnetický tok motora v α, β súradniciach Omega_m: mechanické otáčky motora Me: elektromagnetický moment motora

8.11 Overenie správnosti jednotlivých subsystémov.

Po získaní štruktúry riadenia elektrického pohonu sme sa pustili do preverenia simulácie. Ako už bolo vyššie uvedené, do subsystémov nie je možné vkladať breakpointy ani sa do nich nedá prekrokovať a na také breakpointy program ani nezastavuje. Nakoľko celá štruktúra je veľmi zložitá, skladá sa z rôznych podsystémov ktoré sa delia na ďalšie podsystémy, respektíve bloky, bolo účelné si rozdeliť simuláciu podľa jednotlivých hlavných subsystémov a nezávisle ich vyladiť.

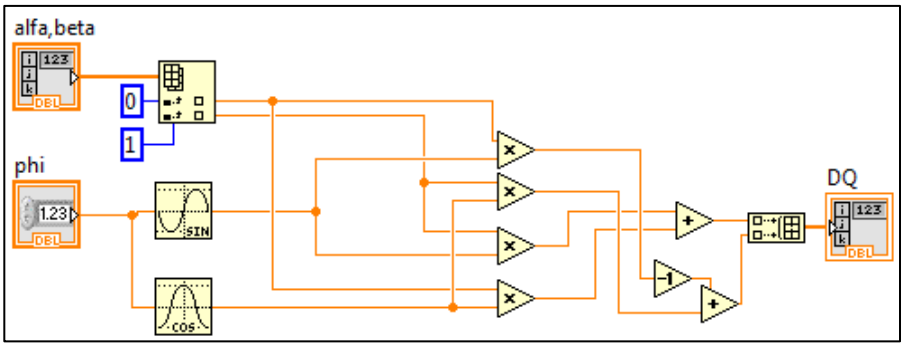
Do testovania sme sa pustili opačným postupom ako pri tvorbe stromovej štruktúry

- Otvorili sme nové VI so simulačným rámom.
- Vybrali sme subsystém na otestovanie.
- Tvar a hodnoty vstupov subVI sme určili podľa výstupov predchádzajúcich blokov, s ktorým je spojený.
- Na vstupy testovaného subsystému sme pripojili signálové generátory, prípade konštanty, ktoré „kopírovali“ požadované tvary a hodnoty.
- Sledovali sme časový priebeh výstupov subsystému.
- Súčasne v programe Matlab sme vybrali korešpondujúci subsystém, vybavený identickými vstupnými signálmi.
- Nadobudnuté časové priebehy z Matlabu nám slúžili ako referenčné.
- Porovnali sme všetky výstupné signály LabVIEW s Matlabom.
- Keď sa časové priebehy zhodovali, tak subsystém bol správne navrhnutý spolu so všetkými subVI, respektíve blokmi ktoré obsahuje.
- V prípade nezhody sme postupovali o úroveň nižšie tak hlboko, kým sme nenarazili na prameň príčiny.

Postupne takto preverený simulačný blokový diagram vytvorený v LabVIEW bol pripravený na simuláciu riadenia elektrického motora.

Počas spúšťania simulácie sme narazili na niekoľko nedostatkov.

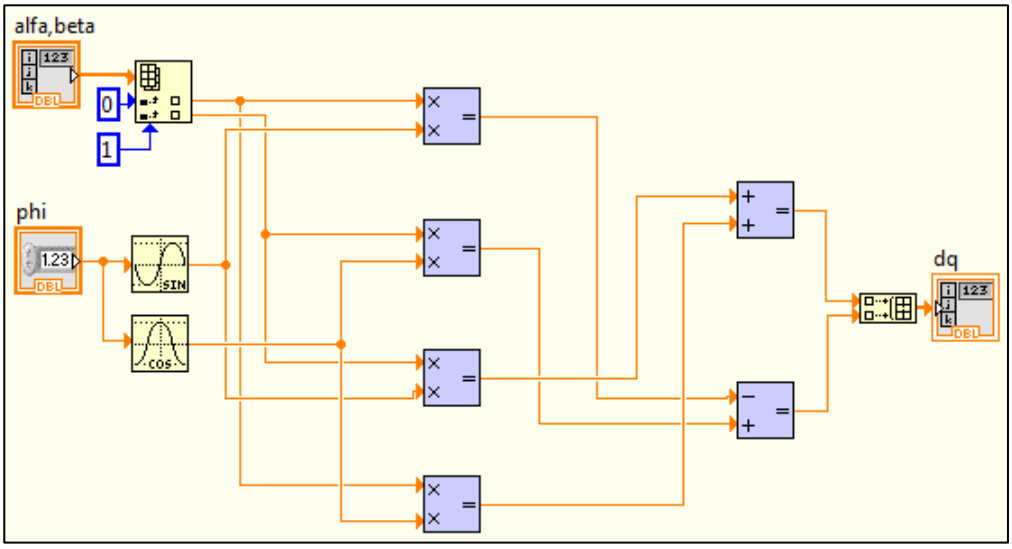
Na nasledujúcom obrázku vidíme implementáciu transformácie zo súradnicového systému alfa beta, do súradnicového systému d,q.



Obr. 19 SubVI alpha_beta to DQ ako klasické subVI

Predpokladali sme že nakoľko tento subVI vykoná jednoduchú transformáciu medzi súradnicovými systémami, a neovplyvní dynamiku celého riadenia, nie je potrebné ho naprogramovať ako simulačný subVI. Počas priebehu simulácie celého riadiaceho algoritmu subVI vykazoval nesprávne chovanie aj napriek tomu že bol matematicky korektne pospojovaný a predtým aj samostatne otestovaný. Podľa vyššie spomenutých tvrdení klasický subVI a simulačný subVI majú odlišné vlastnosti chovania.

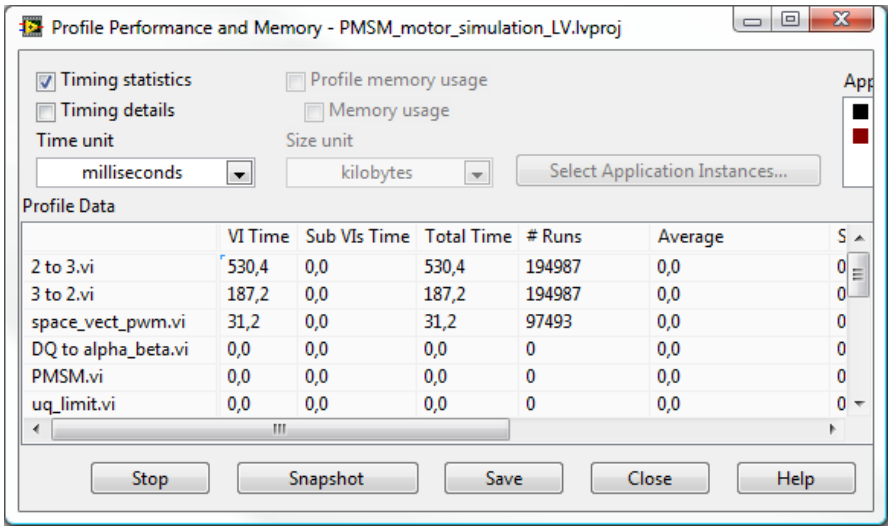
Opätovnou implementáciou tejto transformácie zo súradnicového systému *α,β* do súradnicového systému *d,q* ako simulačný subsystém použitím aritmetických operácií, ktoré sa nachádzajú v palete *Control Design & Simulation* bola chyba korigovaná. Prepracovaný simulačný subVI je znázornený na nasledujúcom obrázku.



Obr. 20 SubVI alpha_beta to DQ ako simulačné subVI

Ďalším nedostatkom bol nezrovnateľne pomalší priebeh simulácie oproti Matlabu. Prvou príčinou, ktorú sme predpokladali bola zložitosť programu. Určité algoritmy sa dajú rôznym spôsobom naprogramovať. Hľadali sme možnosti optimalizácie programu. Na tento účel ponúka LabVIEW testovacie prostredie *Profile Performance and Memory* kde sa zobrazia informácie o čase prevedenia a využitia pamäte jednotlivých VI.

- Otvorili sme cez *Tools* → *Profile* → *Performance and Memory*, a zobrazilo sa nám okno *Profiler*-u.
- Tu sme mohli nastaviť ktoré dáta budú zobrazené.
- *Profiler* sa spustí pomocou tlačidla štart. Následne spustíme aj simuláciu.
- Na vyvolanie aktuálnych hodnôt slúži tlačidlo *Snapshot*.



Obr. 21 Profile Performance and Memory

Po preskúmaní zobrazených hodnôt nám bolo podozrivé, že blok *Simulate Signal* vykazuje vysoké časové hodnoty oproti ostatným blokom. Vo vlastnostiach nastavenia tohto bloku bolo zaškrtnutá voľba *Simulate acquisition timing*, čo spôsobilo spomalenie celého simulačného programu. Po zmene výberu na možnosť *Run as fast as possible* sa zrýchlil celý priebeh programu.

Drobná informácia k *Profiler*-u: Keď sa otvorí z VI, ktorý nie je súčasťou projektu, natiahne všetky VI ktoré sú momentálne otvorené. Keď sa otvorí z VI, ktorý je súčasťou projektu, natiahne iba VI z projektu. Súčasne môžeme profilovať len jeden projekt.

9 POROVNAIE LABVIEW A MATLAB

Myslím, že pôvodné určenie LabVIEW a Matlab je viac odlišné. Cítim, že Matlab je predovšetkým alebo prevažne používaný pre simuláciu a matematiku pre riešenie problémov, zatiaľ čo LabVIEW je určený pre skutočné signály, testovanie a programovanie v prístrojovom svete, aj keď existuje určité prekrývanie v niektorých oblastiach. Práca v LabVIEW požaduje intenzívne a neustále venovanie sa programu, vtedy sa dá s programom slušne pracovať. S pohľadom „občasného“ užívateľa je Simulink nadstavba Matlabu, je rádovo jednoduchší aj keď má aj tento program svoje nedostatky.

Počas implementácie algoritmov do blokovej štruktúry sme narazili na nasledovné rozdiely

- Pri vytváraní jednotlivých blokov, Matlab ponúka pretáčanie blokov o 90 stupňov. LabVIEW umožňuje len preklápanie blokov a len tých ktoré sa nachádzajú v palete Control Design & Simulation. Následné pospájanie týchto blokov môže viesť k neprehľadnosti.
- Veľkosti blokov (ikon) v LabVIEW sú presne dané. Veľkosti blokov v prostredí Matlab sú voľne zmeniteľné a ponúkajú dostatočný priestor na pomenovanie vstupov a výstupov.
- Z hľadiska prehľadnosti je farebné rozlíšenie v LabVIEW výhodnejšie, rozlišuje typy blokov a pri pospojovaní určí aj dátový typ.
- Pri vytváraní subsystémov v Matlab Simulink nepotrebujeme ich uložiť zvlášť ako samostatný súbor. Je súčasťou hlavného bloku. V LabVIEW každý subsystém je uložený do nového súboru a navyše musí byť dopredu stanovený typ (klasický alebo simulačný).
- Nakoľko v LabVIEW každý subsystém je samostatný súbor, ktorý obsahuje čelný panel aj blokový diagram, v zložitejšom stromovom štruktúre to znamená, že v operačnom systéme na hlavnom paneli máme dvojnásobne toľko okien otvorených ako v Simulinku.
- Čas potrebný na nasimulovanie rovnako náročného algoritmu je podstate rovnaký. Pri prvom preklade (kompilácii) LabVIEW ukazuje oneskorenie. Pri každej zmene v programe rozsiahlejších projektov je nutné uvažovať oneskorenie spôsobené prvotnou kompiláciou týchto zmien.

Počas našej práci sme zďaleka neobjasnili všetky výhody a nevýhody jednotlivých prostredí, ktoré by predurčili ich použitie. Podľa dosiahnutých výsledkov však dosvedčujeme predpoklad, že výrobok LabVIEW firmy National Instruments na modelovanie a simulovanie komplexnejších dynamických systémov je vhodným nástrojom.

10 ZÁVER

Cieľom našej práce bolo oboznámiť sa existujúcimi algoritmami riadenia elektrických striedavých motorov v prostredí Matlab Simulink a ich následný prevod do prostredia LabVIEW Control Design & Simulation.

Pri tvorení Externého modelu asynchrónneho motora v LabVIEW sme postupovali nasledovne.

Nadefinovali sme model s použitím funkcie EMI v programovacom jazyku C. Zo zdrojového kódu v C sme vytvorili dynamicky linkovanú knižnicu DLL, na ktorú sme odkazovali z LV pomocou funkčného bloku External Model. Tento vytvorený model sme následne otestovali a z nadobudnutých priebehov sme urobili porovnanie s Matlabovským ekvivalentom (viď príloha 3, 4).

Na algoritme riadenie asynchrónneho motora a synchrónneho motora s permanentnými magnetmi sme vytvorili jednotlivé stromové štruktúry subsystémov vo vývojovom prostredí LabVIEW. Jednotlivé subsystémy sme navzájom prepojili a sfunkčnili. Prejavené nedostatky sme následne odstránili.

Stromová štruktúra jednotlivých algoritmov na riadenie elektrických motorov je znázornená v prílohe (6, 27).

Na našom ústave bola vytvorená knižnica funkcií, ktorá rieši pripojenie a použitie motora na CompactRIO. S použitím tejto knižnice máme možnosť nahradiť model motora v simulačnom algoritme skutočným motorom a tak by sme mohli byť schopný riadiť motor v real time –u s využitím mikroprocesora CompactRIO -a.

Počas riešenia celej problematiky sme získali poznatky ktoré ukazujú prednosti jednotlivých simulačných nástrojov a preferujú ich využitie.

V živote sa často stretávame s priorizáciou, alebo tiež protichodnými názormi testovania reálnych aplikácií tzv. „na živo“ verzus simulácie v SW riešeniach. Výhody a nevýhody oboch riešení sú známe. Úlohou zostáva vhodne posúdiť riešenú problematiku a rozhodnúť o najoptimálnejšom postupe riešenia. Táto časť je veľmi dôležitá a môže v konečnom dôsledku priniesť očakávaný efekt (časová úspora, finančná úspora, odhalenie slabých bodov, kritické uzly, a.i.). V tejto našej práci sme sa viac orientovali na implementáciu modelov a algoritmov v SW nástrojoch, ktoré sú bežne dostupné. Pre jednoduchosť možno povedať, že LabVIEW je viac spojený s reálnym svetom. (Jednoduchšie prepojenie a testovanie s reálnymi HW zariadeniami, ktoré sú už vytvorené. Samotný program je možné priamo použiť v riadiacom systéme.) Naproti tomu Matlab je viac vývojový a simulačný nástroj. Pre správnosť je vhodné uviesť, že pre Matlab tiež existujú HW riešenia, ale od iných dodávateľov - tretích strán, a samotný program je potrebné dodatočne prispôsobiť danému riadiacemu systému. Sú to dva svety, ktoré pri vhodnom návrhu a skĺbení poskytnú viac možností napr. vzájomná komparácia výsledkov a riešení, odhalenie chybných častí, algoritmov, optimalizácia, atď.. Konečné zhrnutie a porovnanie Matlab a LabVIEW pre nami

zvolené zadanie neprináša jednoznačného favorita. Výsledky a možnosti sú rovnocenné, čo do výsledku, odlišné sú len spôsoby samotnej realizácie.

Literatúra

[1] BLAHA, Petr. *Algoritmy pro bezsnímačové řízení asynchronnikch motorů*. Brno, 2006. Habilitační práce, VUT v Brně.

[2] *LabVIEW 2009 Control Design and Simulation Module Help*: *Callbacks* [online]. 2009 [cit. 2012-05-02]. Dostupné z: http://zone.ni.com/reference/en-XX/help/371894D-01/lvsimemi/emi_callbacks/

[3] *LabVIEW 2011 Control Design and Simulation Module Help* [online]. 2011 [cit. 2012-03-27]. Dostupné z: http://zone.ni.com/reference/en-XX/help/371894F-01/

[4] *Elektrické stroje.* Skriptum. VUT v Brně.

[5] CAHA, Zdeněk a Miroslav ČERNÝ. *Elektrické pohony*. 1. vyd. Praha: Státní nakladatelství technické literatury, 1990, 359 s. ISBN 8003004187.

[6] POLIAK, František, Ladislav ZBOŘIL a Viliam FEDÁK. *Elektrické pohony: celoštátna vysokoškolská učebnica pre elektrotechnické fakulty vysokých škôl*. 1. vyd. Bratislava: Alfa, 1987, 614 s.

[7] NEBORÁK, Ivo. *Modelování a simulace elektrických regulovaných pohonů*. 1. vyd. Ostrava: Vysoká škola báňská - Technická univerzita, 2002, 172 s. ISBN 8024800837.

[8] KADANÍK, P.: *Řízení asynchronního motoru bez použití snímače rychlosti*, Praha 2004, Disertační práce, FEL ČVUT v Praze. Dostupné z: http://pohony.kadanik.cz

[9] RYDLO, Pavel. *Řízení elektrických střídavých pohonů*. Vyd. 2. V Liberci: Technická univerzita, 2007, 129 s. ISBN 9788073722234.

[10] BRANDŠTETTER, Pavel. *Střídavé regulační pohony: moderní způsoby řízení*. Ostrava: VŠB-Technická univerzita, 1999, 181 s. ISBN 807078668x.

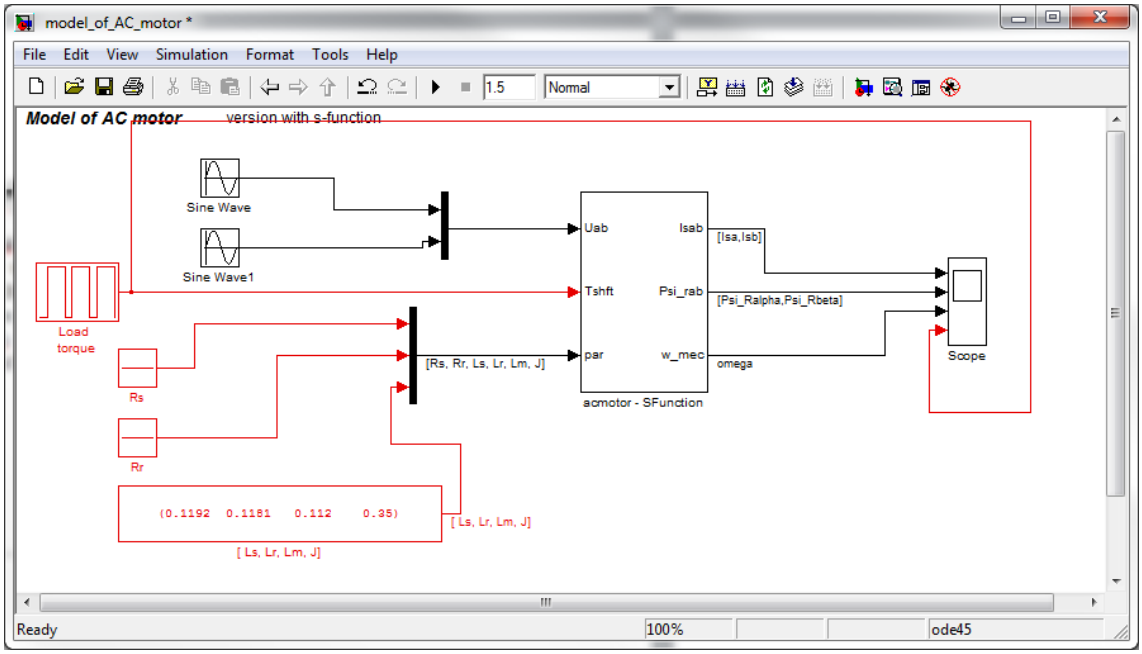
[11] SKALICKÝ, Jiří. *Elektrické regulované pohony.* VUT v Brně, 2007.

Elektronické prílohy (CD)

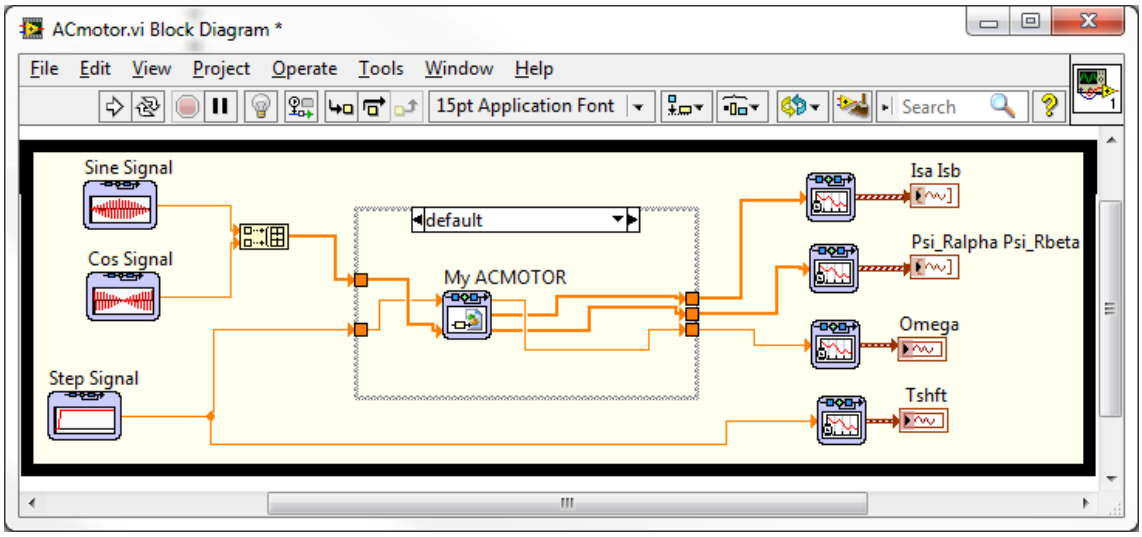
- Diplomová práca
- **ACmotor_model** – priečinok obsahuje matematický model asynchrónneho motora vytvorený pomocou EMI funkcií
- **ACIM_control_simulation_LV** – priečinok obsahuje simuláciu riadenie asynchrónneho motora
    - Otvorí sa: _SIEMENS_motor_simulation.lvproj
    - Simulácia sa spustí zo súboru: *_Main.vi*
- **PMSM_control_simulation_LV** - priečinok obsahuje simuláciu riadenie synchrónneho motora s permanentnými magnetmi
    - Otvorí sa: PMSM_motor_simulation_LV.lvproj
    - Simulácia sa spustí zo súboru: *_Main.vi*

Prílohy

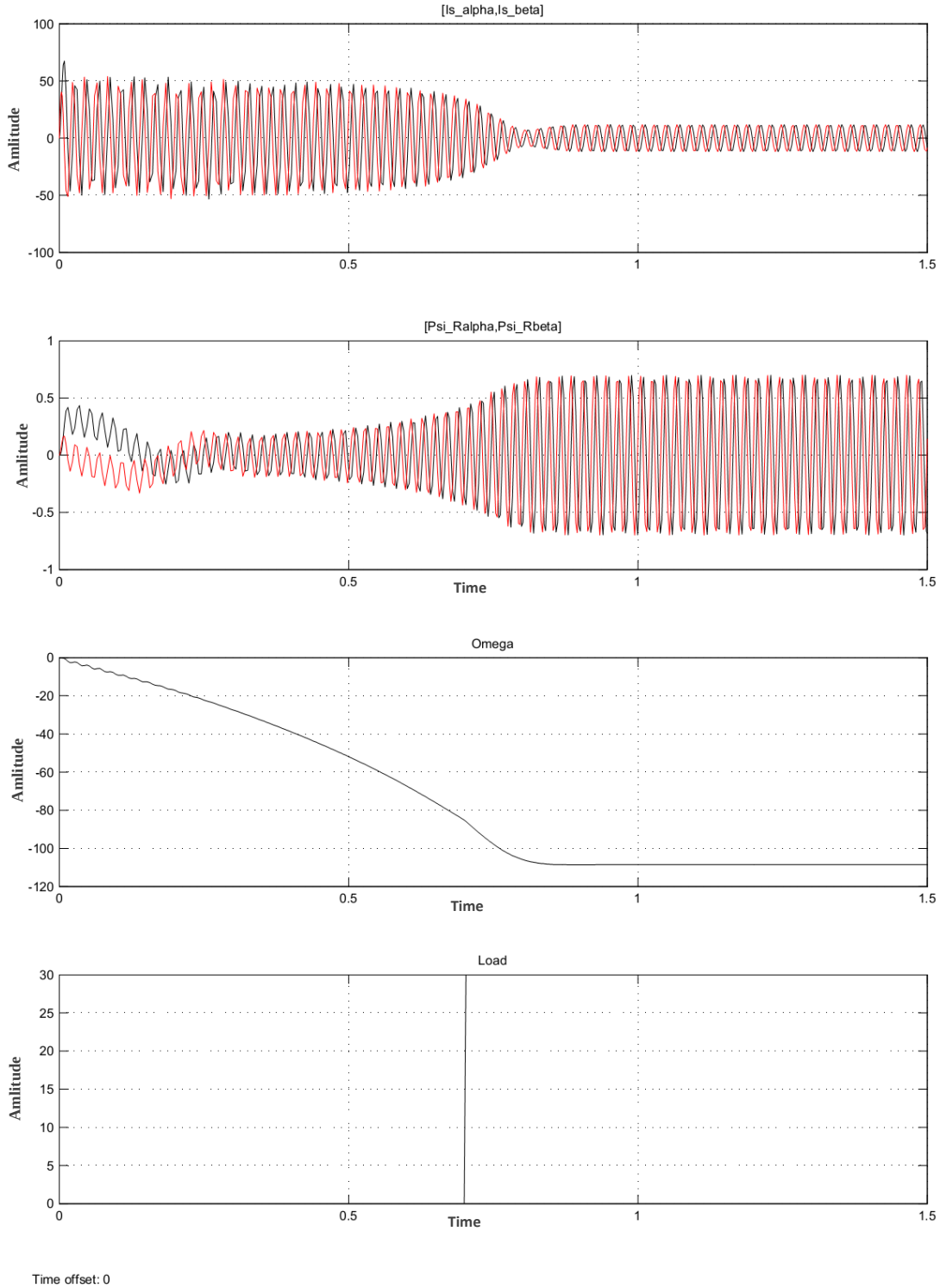
Príloha 1: Testovacie zapojenie modelu ACIM v Matlab Simulink



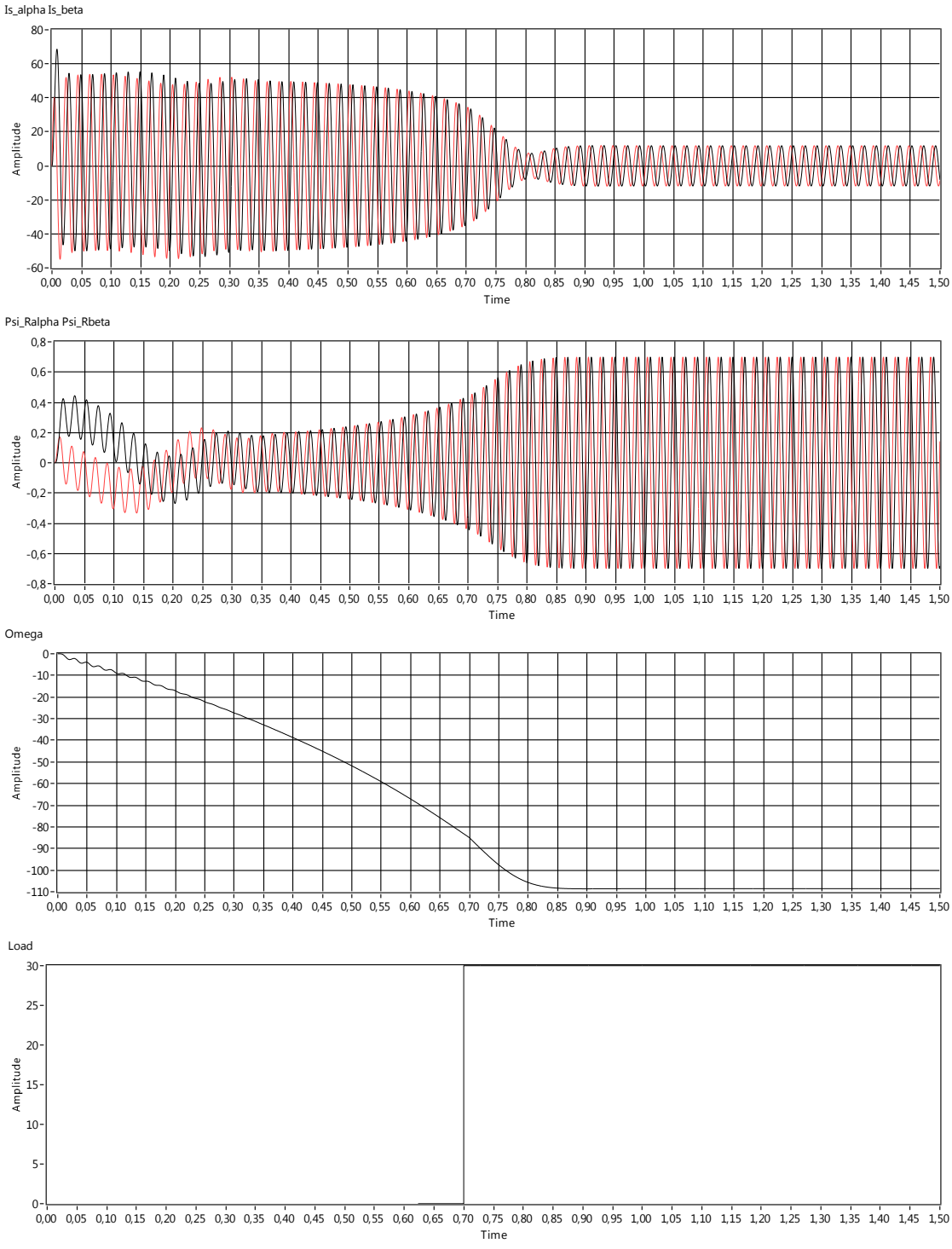
Príloha 2: Testovacie zapojenie ACIM v LabVIEW



Príloha 3: Priebeh simulácie ACIM v Matlab Simulink



Príloha 4: Priebehy simulácie ACIM v LabView

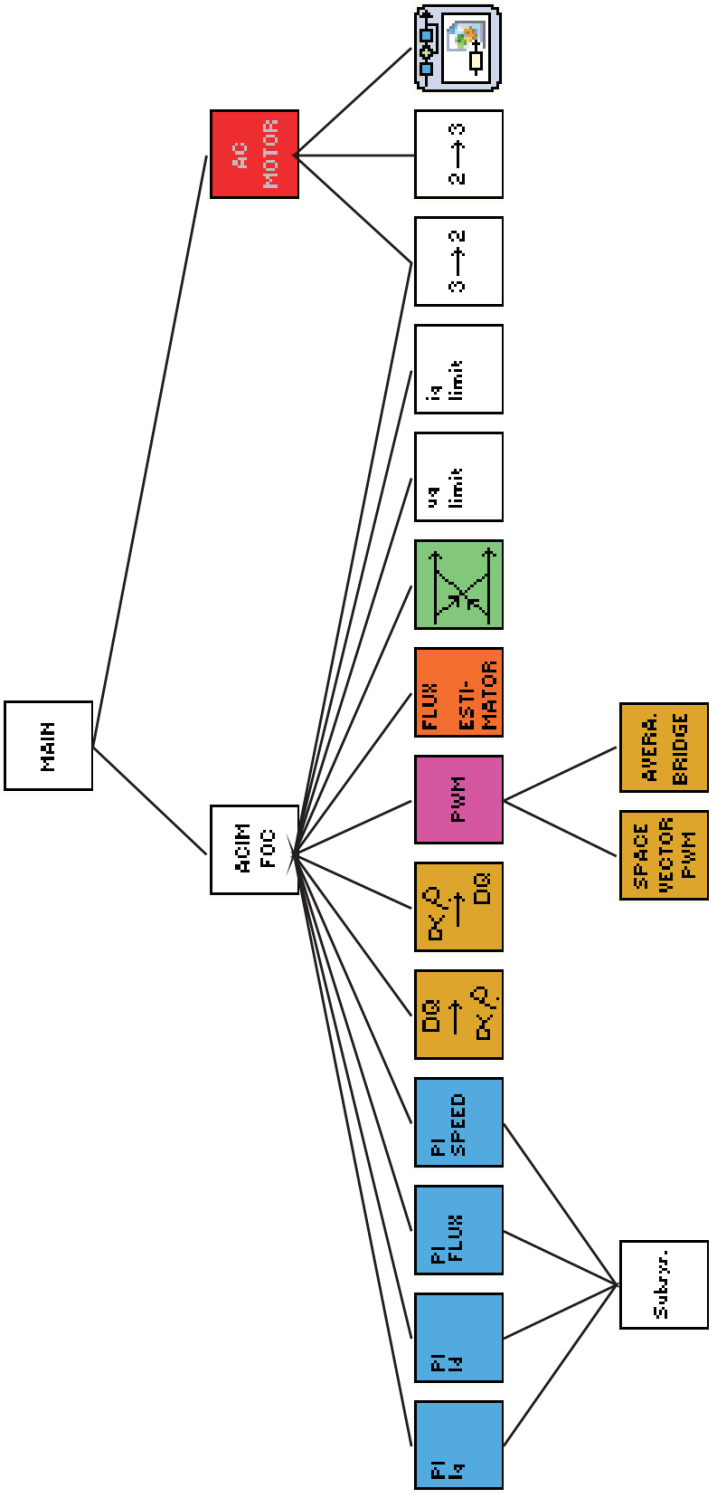


Príloha 5: Implementácia externého modelu asynchrónneho motora použitím funkcie EMI

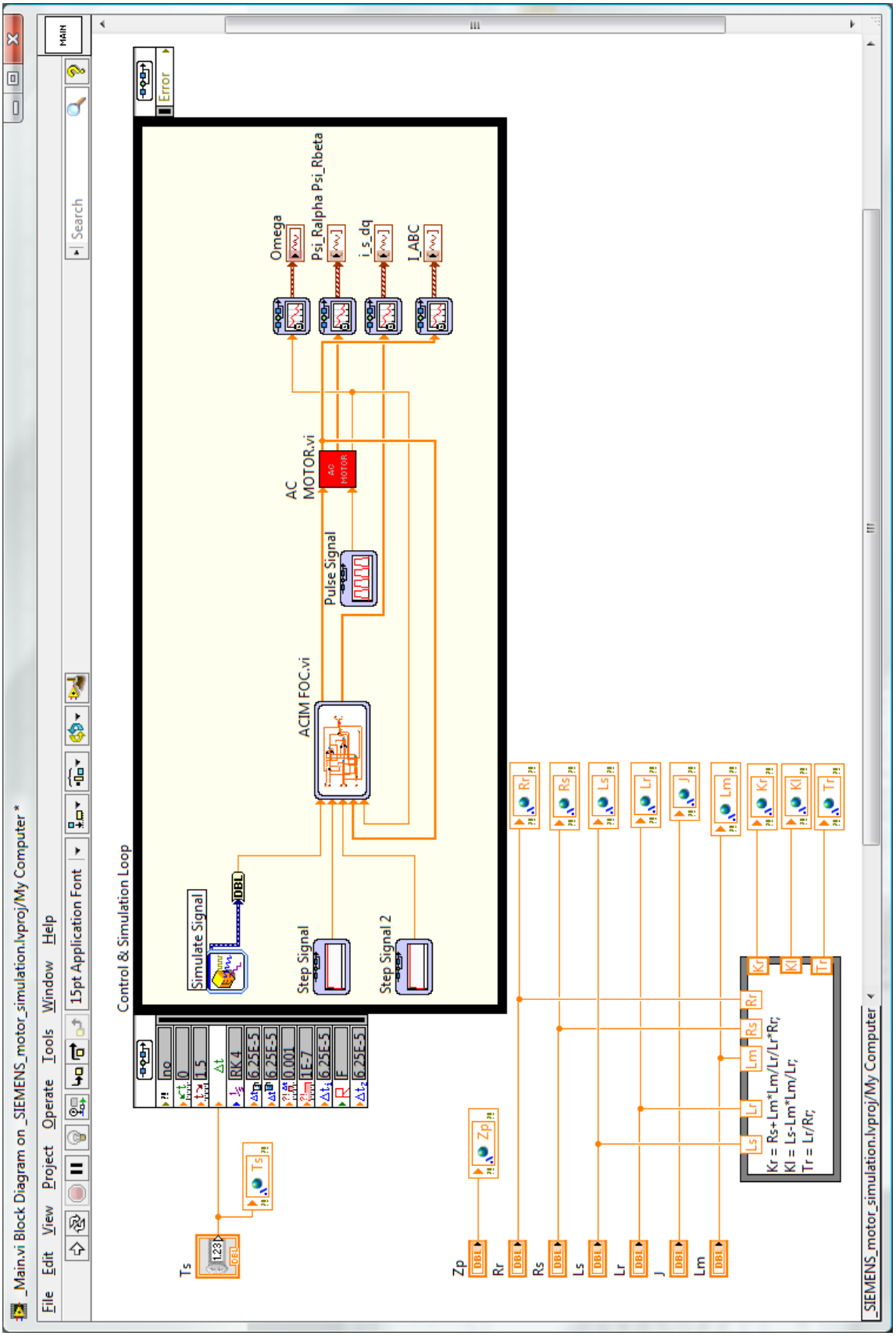
```
/** FIRST INPUT PIN */
/** Ualpha */
#define Ua          (uPtrs[0])
/** Ubeta */
#define Ub          (uPtrs[1])

/** SECOND INPUT PIN */
/** torque on shaft */
#define Tshft (tPtrs[0])

/** THIRD INPUT PIN */
/** Stator resistance */
#define Rs      (paramPtrs[0])
/** Roror resistance */
#define Rr      (paramPtrs[1])
/** Stator inductance */
#define Ls      (paramPtrs[2])
/** Rotor inductance */
#define Lr      (paramPtrs[3])
/** Mutual inductance */
#define Lm      (paramPtrs[4])
/** Motor inertia */
#define J       (paramPtrs[5])
/** Motor pole pairs */
#define pp      (paramPtrs[6])

#include "EMI_ACmotor.h"


EMI_ACMOTOR_C_PRE void EMI_CB_ModelInterface(emiRef model){
       EMI_SetModelName(model, "My ACMOTOR");
       /**INPUTS*/
       EMI_AddVectorInput(model, "Uab", 2);
       EMI_AddScalarInput(model, "Tshft");


       /**PARAMS*/
       double initVals[7] = {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0};
       EMI_AddVectorParam(model, "Par", EMI_ParamSource_ConfigPage, initVals, 7);

       /**OUTPUTS*/
       EMI_AddScalarOutput(model, "Omega", EMI_Feedthrough_Indirect);
       EMI_AddVectorOutput(model, "Isab", EMI_Feedthrough_Indirect, 2);
       EMI_AddVectorOutput(model, "Psi_rab", EMI_Feedthrough_Indirect, 2);

}

EMI_ACMOTOR_C_PRE void EMI_CB_SizeInformation(emiRef model){
       EMI_SetNumberOfContinuousStates(model, 5);
}

EMI_ACMOTOR_C_PRE void EMI_CB_InitializeModel(emiRef model){
       double* xc0 = EMI_GetInitialContinuousStates(model);
       xc0[0] = 0.0;
```

```
	xc0[1] = 0.0;
	xc0[2] = 0.0;
	xc0[3] = 0.0;
	xc0[4] = 0.0;
}

EMI_ACMOTOR_C_PRE void EMI_CB_CalculateDerivatives(emiRef model){
	/*states [Isa Isb Psi_ra Psi_rb omega_el]*/

	double* dx = EMI_GetDerivatives(model);
	const double* x = EMI_GetContinuousStates(model);

	const double* uPtrs = EMI_GetInput(model, 0);
	const double* tPtrs = EMI_GetInput(model, 1);
	const double* paramPtrs = EMI_GetParam(model, 0);
	double k1, k2, k3, k4, k5, tmp1, tmp2;

	k5 = Rr/Lr;
	tmp1 = Lm*Lm/Lr;
	tmp2 = Ls-tmp1;
	k4 = Lm*k5;
	k3 = Lm/Lr/tmp2;
	k2 = Lm*Rr/Lr/Lr/tmp2;
	k1 = (Rs+tmp1*Rr/Lr)/ tmp2;

	dx[0] = -k1*x[0]            + k2*x[2]      + k3*x[3]*x[4] + Ua/tmp2;
	dx[1] =           - k1*x[1] - k3*x[4]*x[2] + k2*x[3]              +Ub/tmp2;
	dx[2] = k4*x[0]             - k5*x[2]      - x[4]*x[3];
	dx[3] =             k4*x[1] + x[4]*x[2]    - k5*x[3];

	dx[4] = pp / J *(1.5*pp*Lm/Lr*(x[1]*x[2] - x[0]*x[3]) - Tshft);
}

EMI_ACMOTOR_C_PRE void EMI_CB_CalculateIndirectOutputs(emiRef model){
	const double* x = EMI_GetContinuousStates(model);
	const double* paramPtrs = EMI_GetParam(model, 0);
	double* omega = EMI_GetOutput(model, 0);
	double* I = EMI_GetOutput(model, 1);
	double* Psi_r = EMI_GetOutput(model, 2);

	I[0] = x[0];
	I[1] = x[1];
	Psi_r[0] = x[2];
	Psi_r[1] = x[3];
	omega[0] = x[4]/pp;
}
```

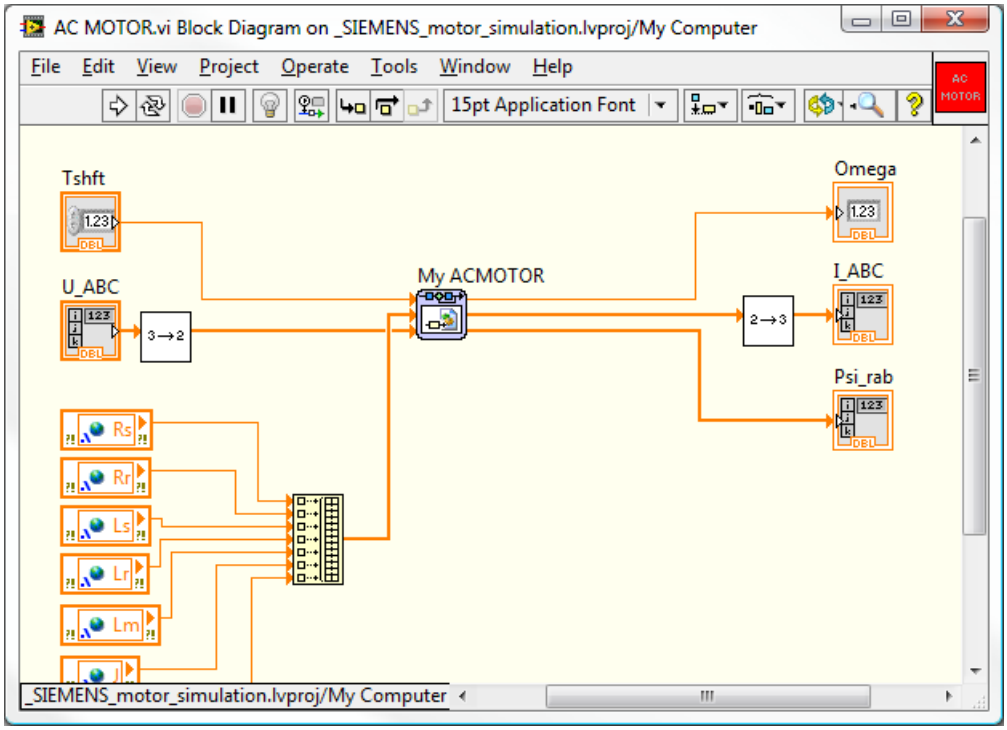
Príloha 6: Stromová štruktúra simulácie vektorového riadenia asynchrónneho motora



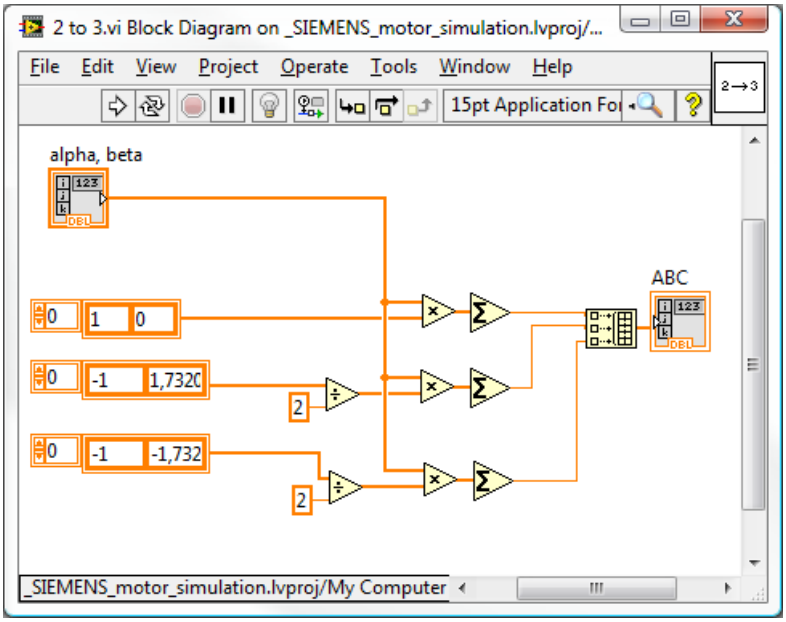
Príloha 7: _MAIN.vi



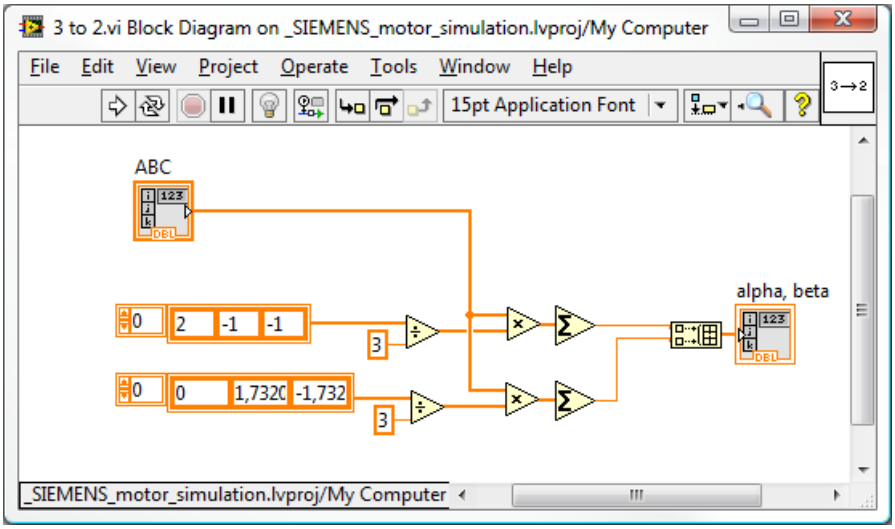
Príloha 8: AC MOTOR.vi



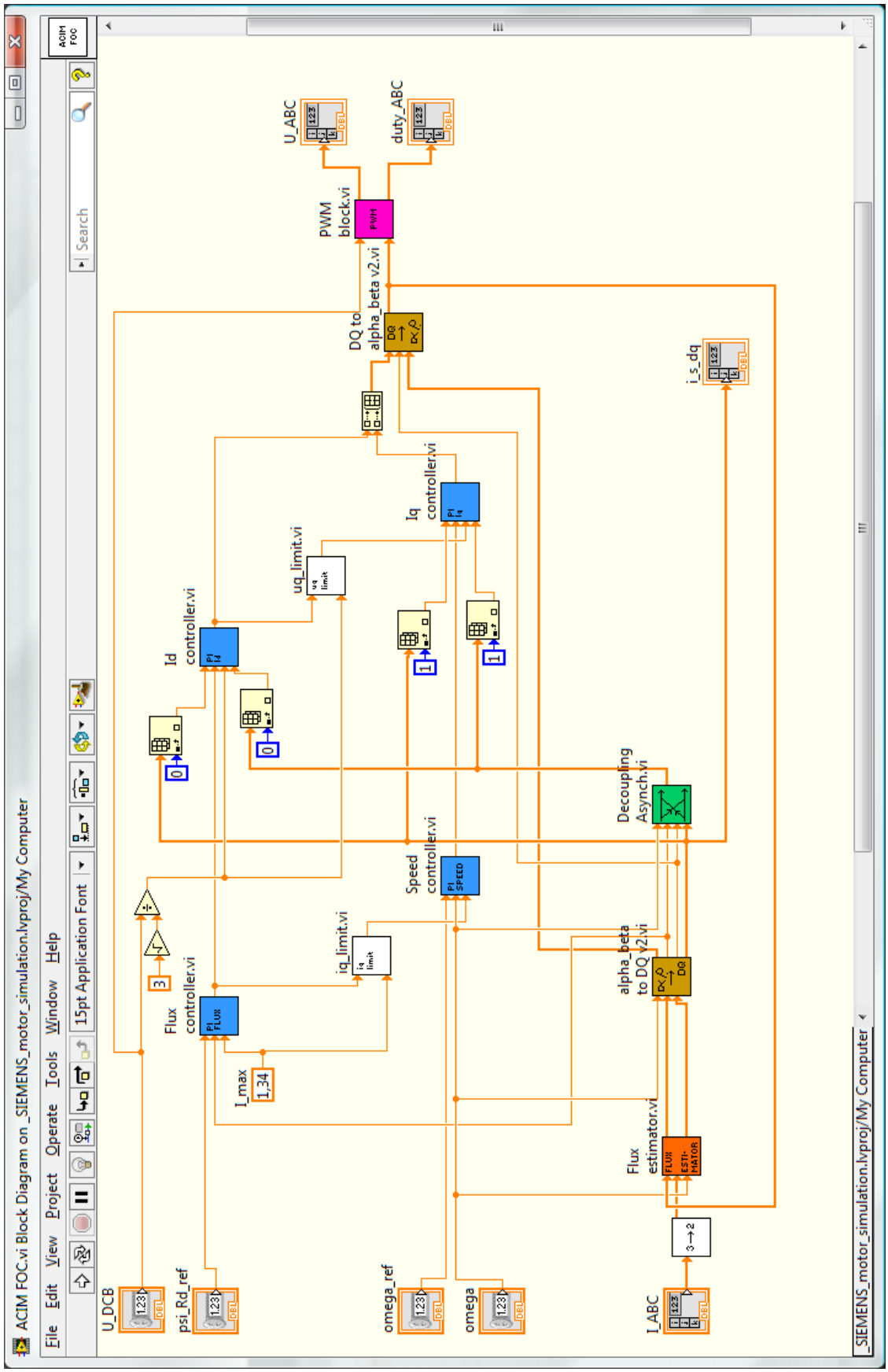
Príloha 9: 2 to 3.vi



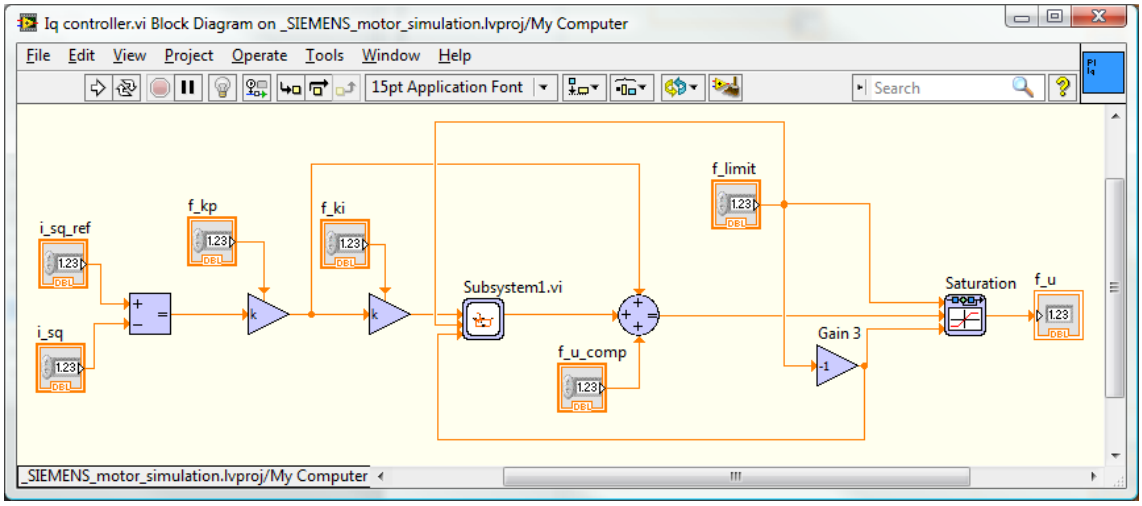
Príloha 10: 3 to 2.vi



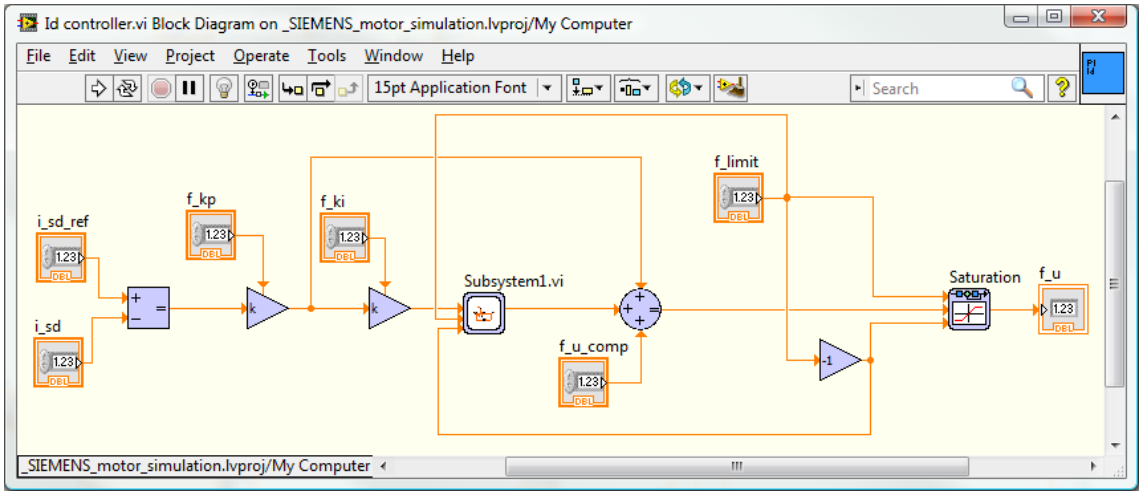
Príloha 11: ACIM FOC.vi



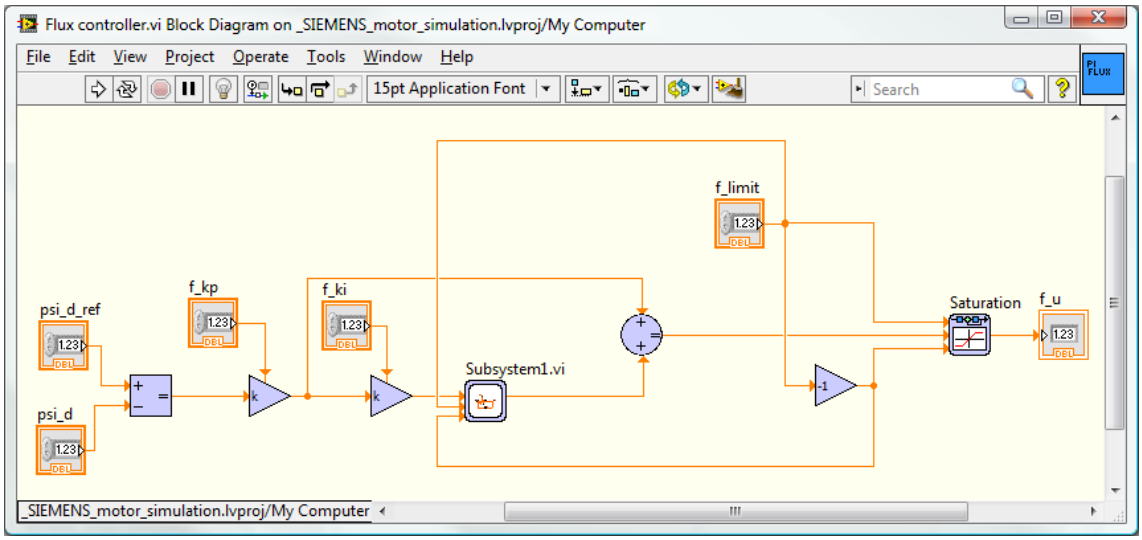
Príloha 12: Iq controller.vi



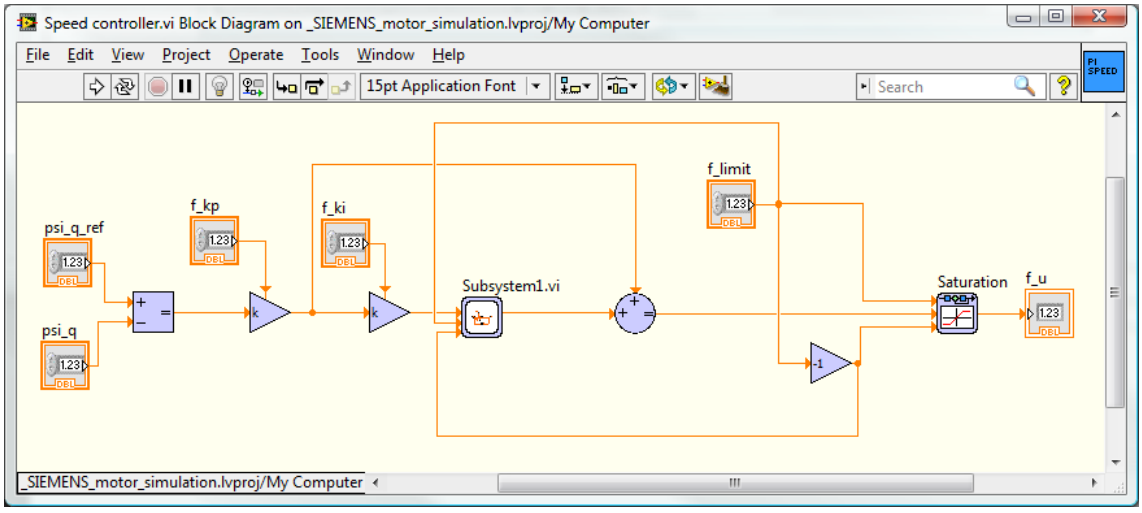
Príloha 13: Id controller.vi



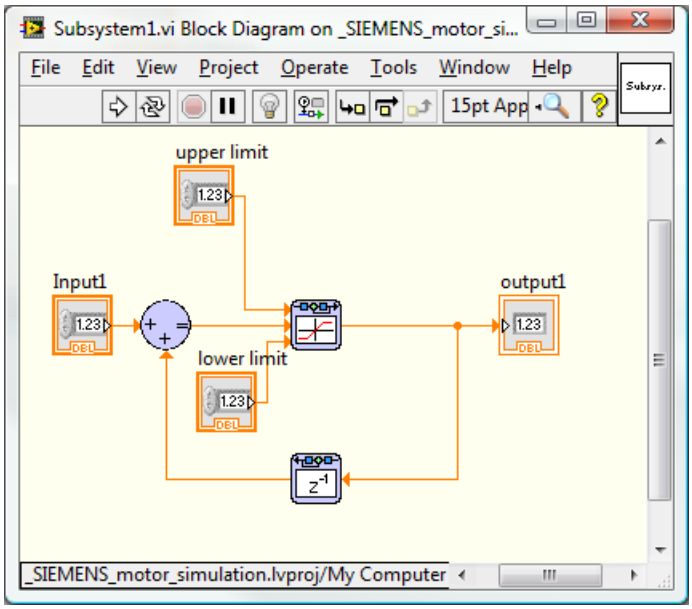
Príloha 14. Flux controller



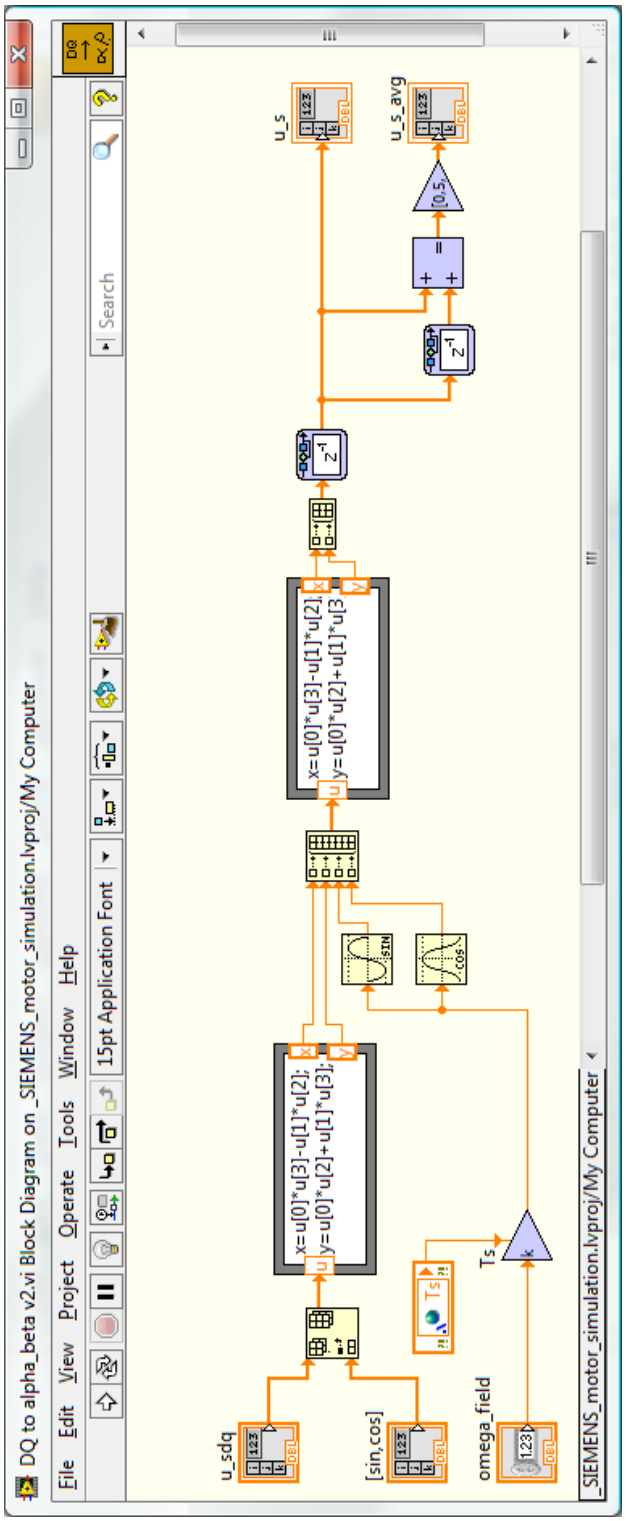
Príloha 15: Speed controller.vi



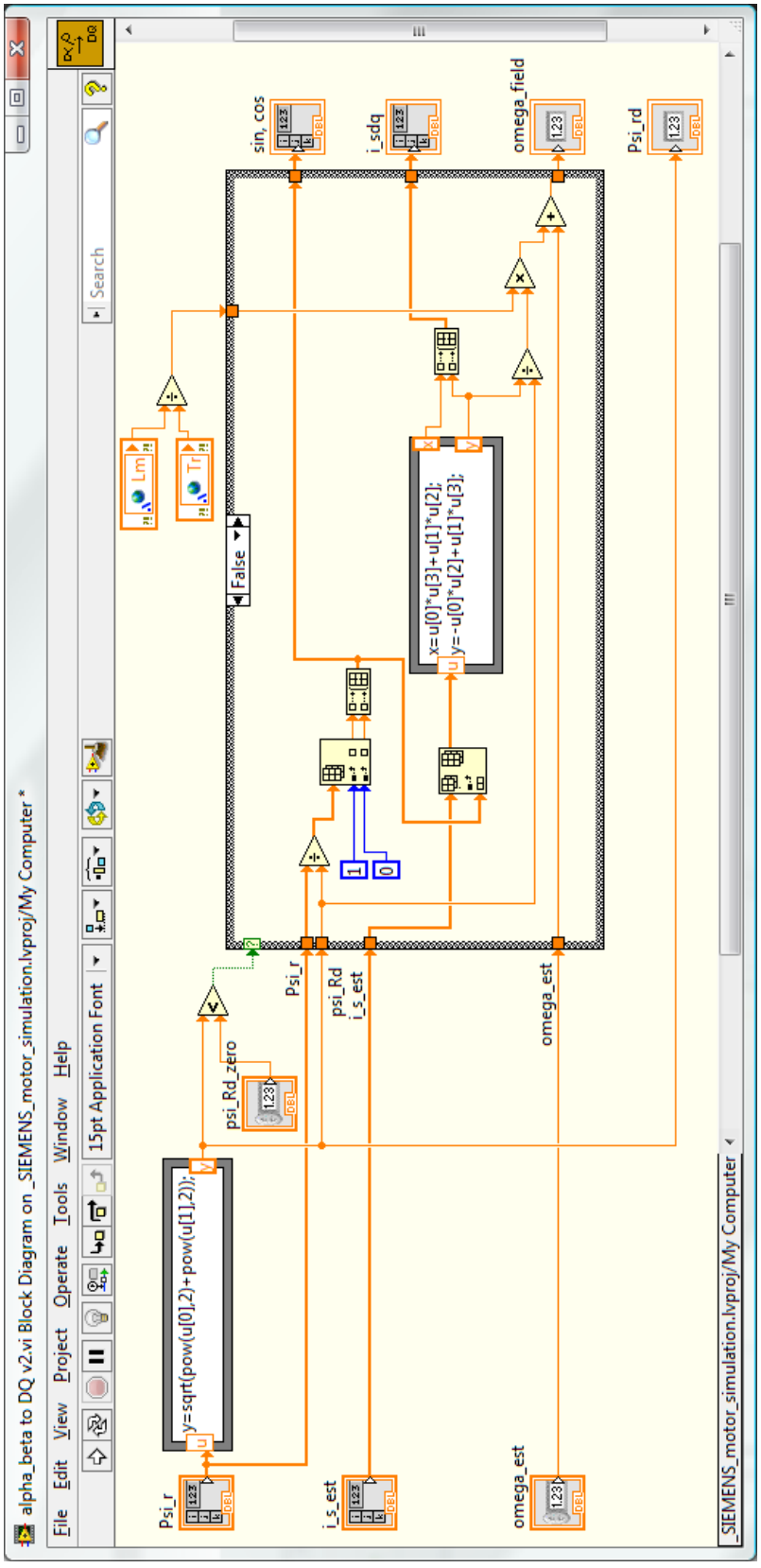
Príloha 16: Subsystem1.vi



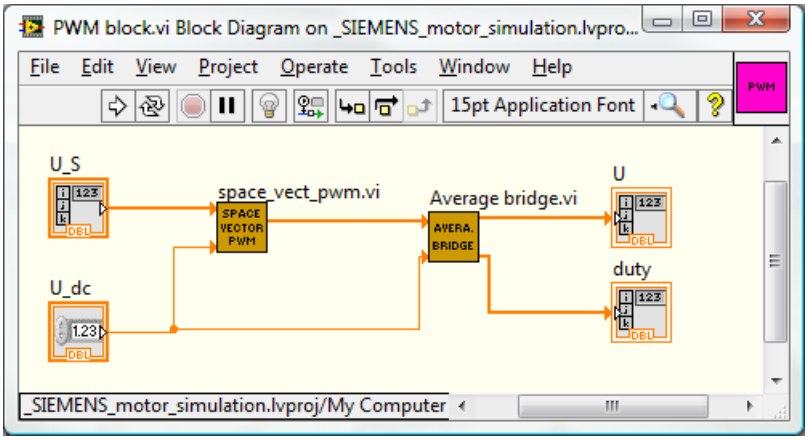
Príloha 17: DQ to alpha_beta v2.vi



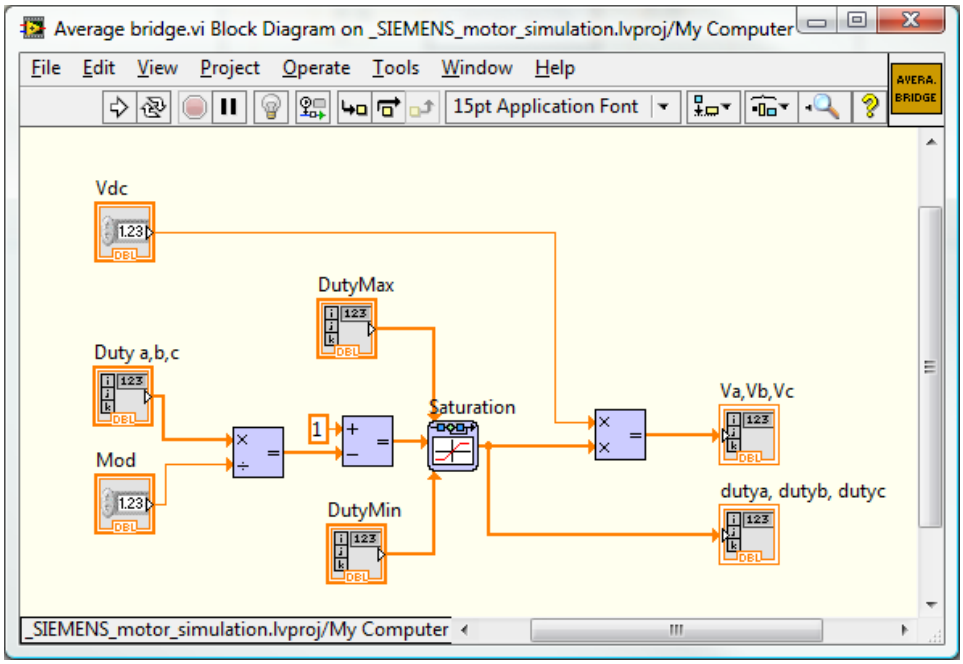
Príloha 18: alpha_beta to DQ v2.vi



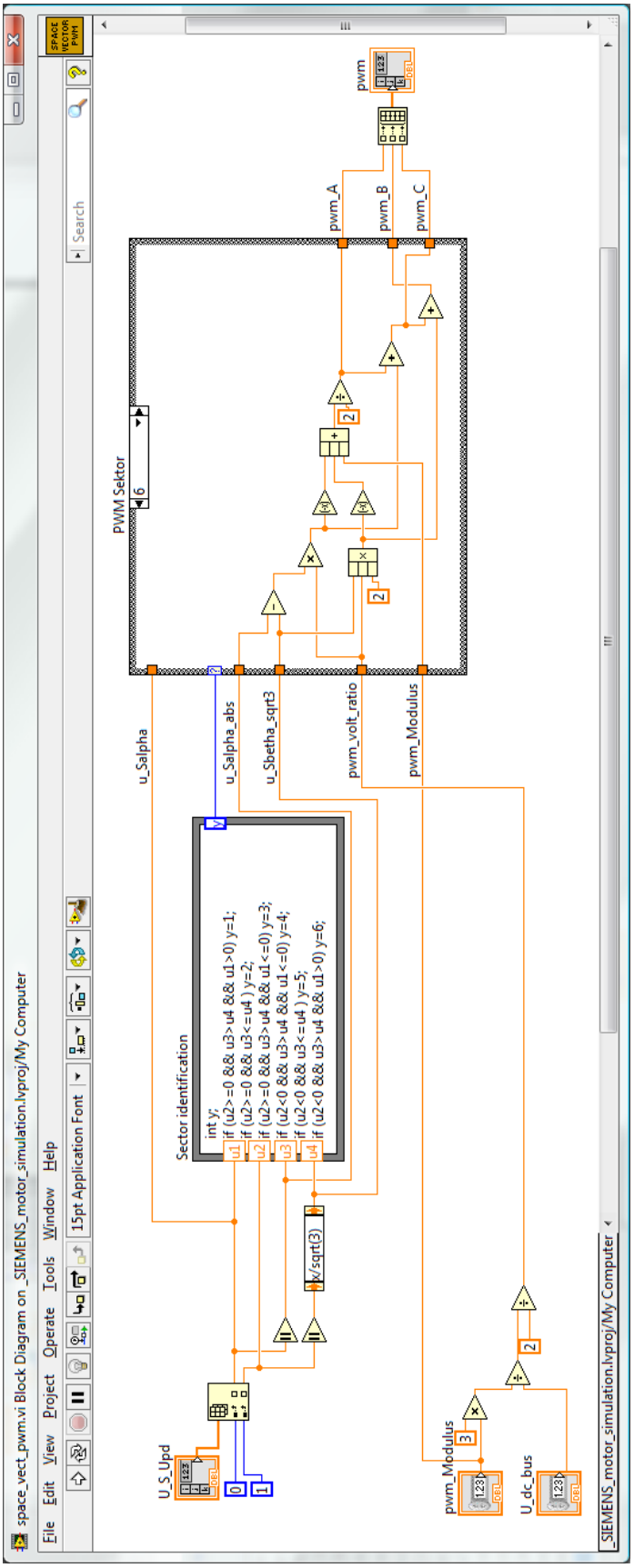
Príloha 19: PWM block.vi



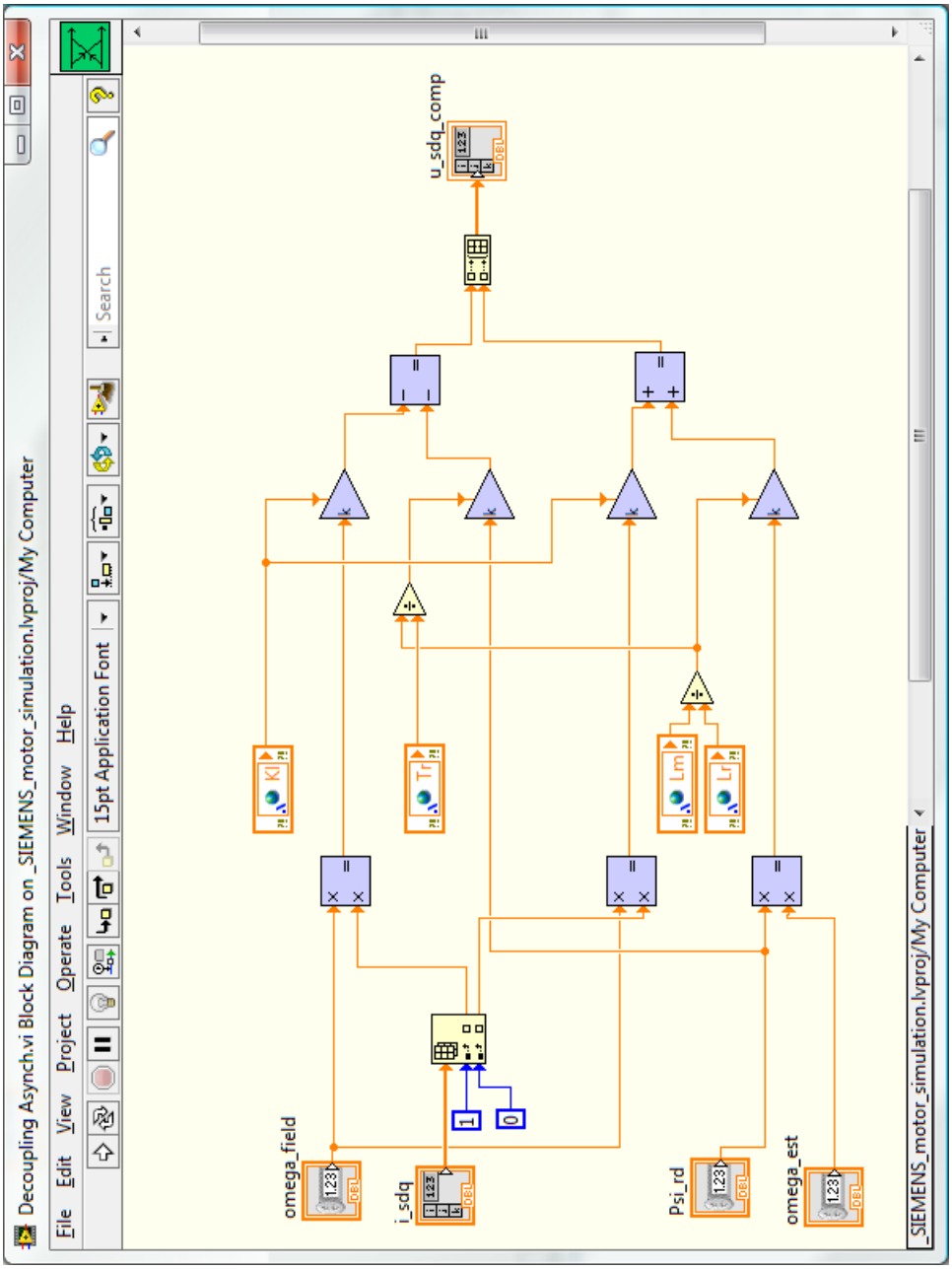
Príloha 20: Average bridge.vi



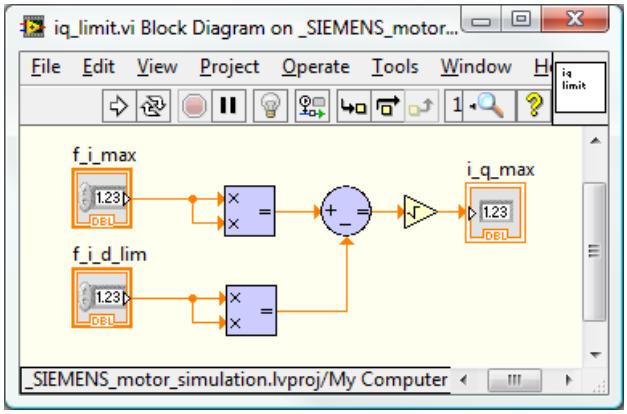
Príloha 21: Space_vect_pwm.vi



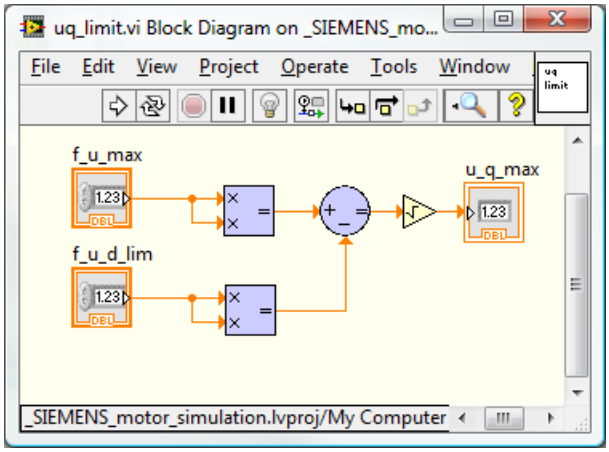
Príloha 22: Decoupling Asynch.vi



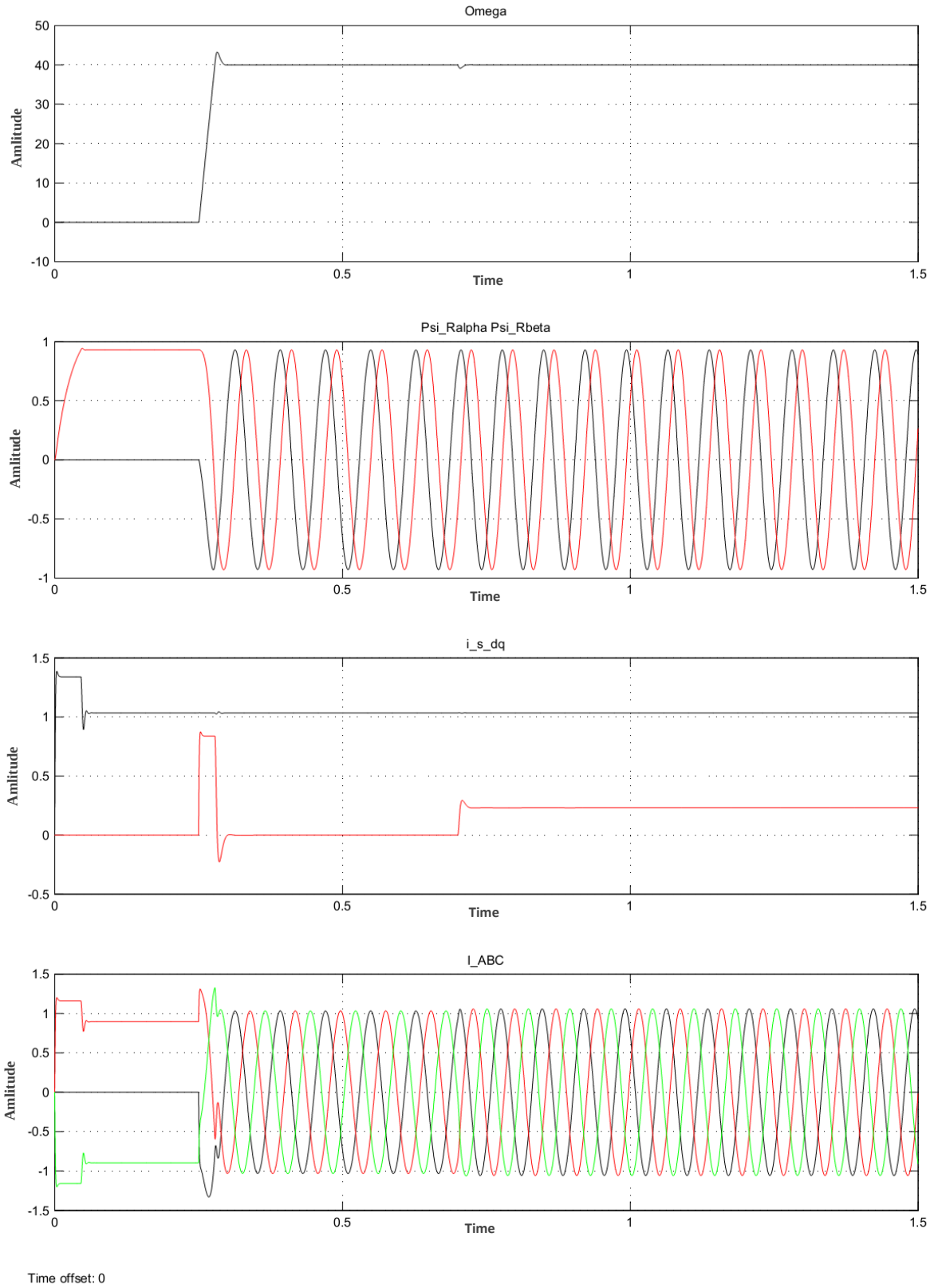
Príloha 23: iq_limit.vi



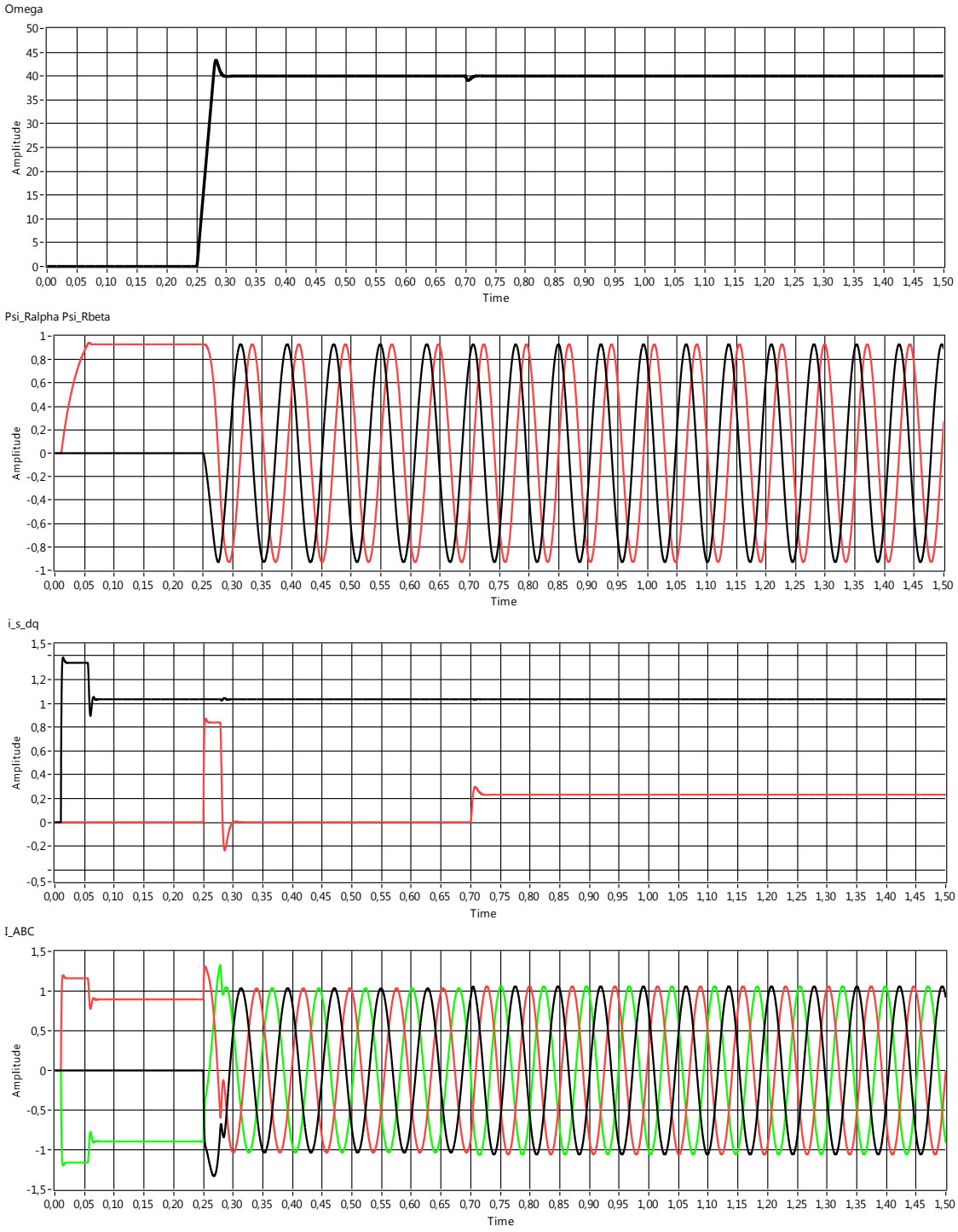
Príloha 24: uq_limit.vi



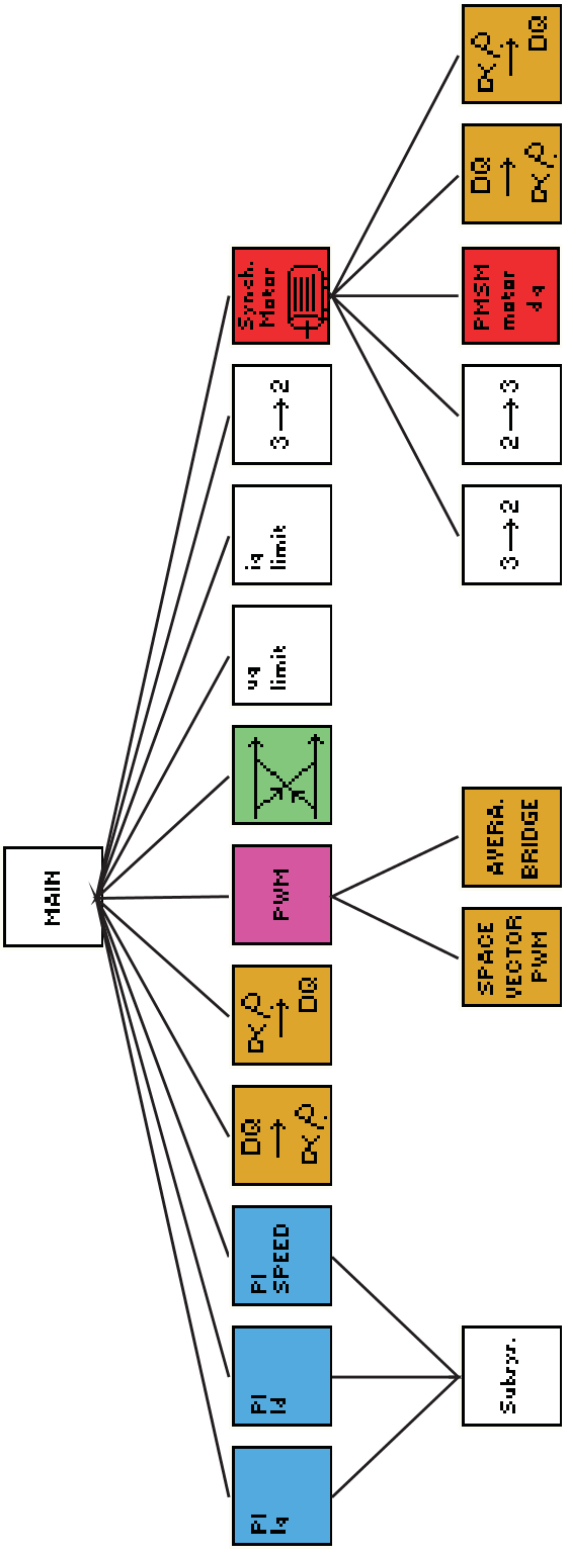
Príloha 25: Výsledné priebehy simulácie riadenia ACIM v prostredí Matlab



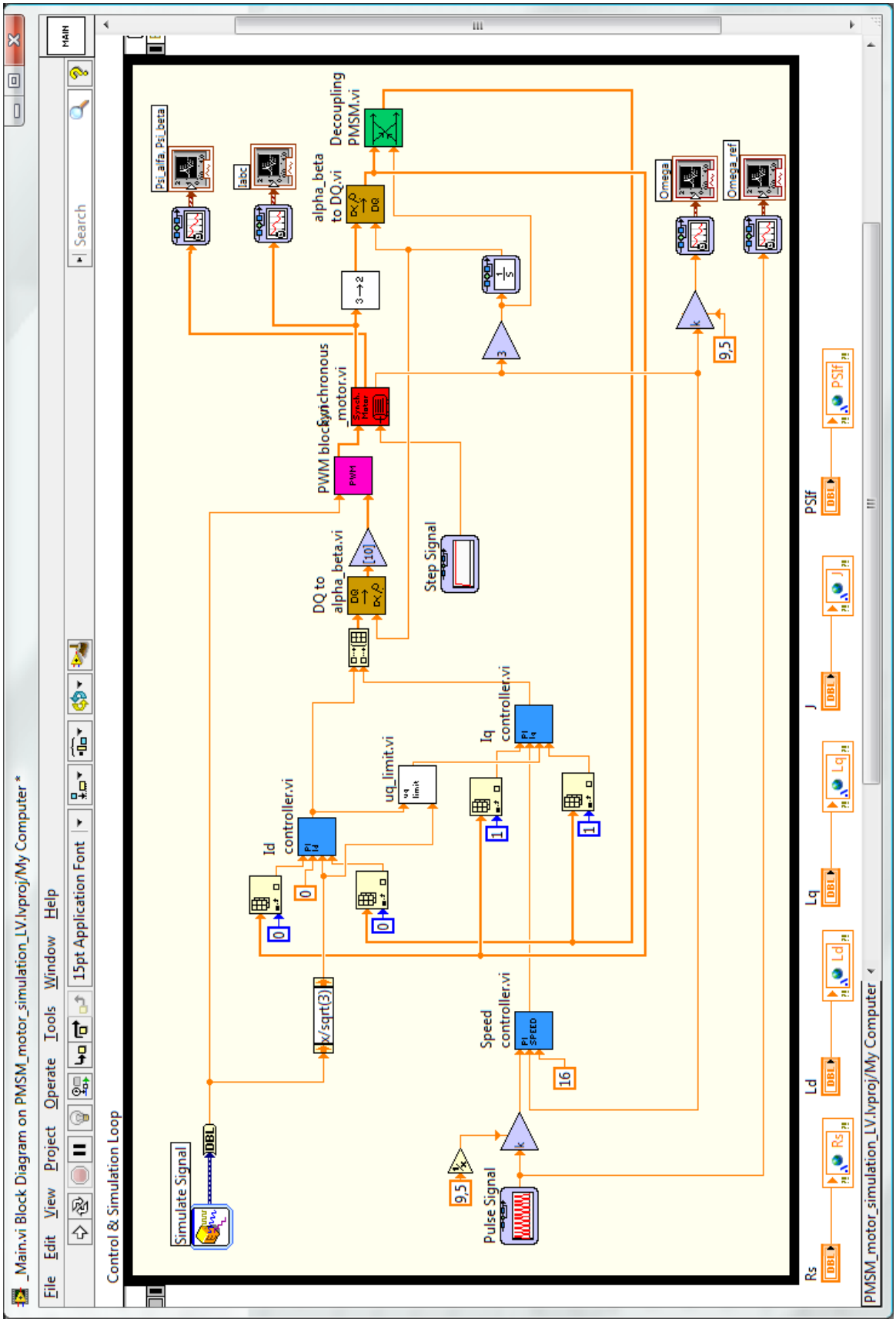
Príloha 26: Výsledné priebehy simulácie riadenia ACIM v prostredí LabVIEW



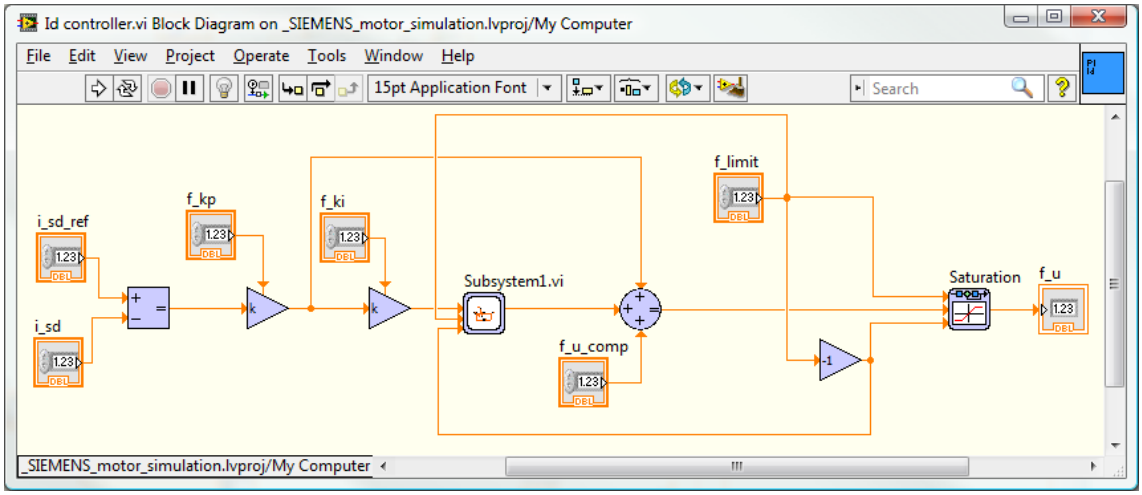
Príloha 27: Stromová štruktúra simulácie vektorového riadenia synchrónneho motora s permanentnými magnetmi



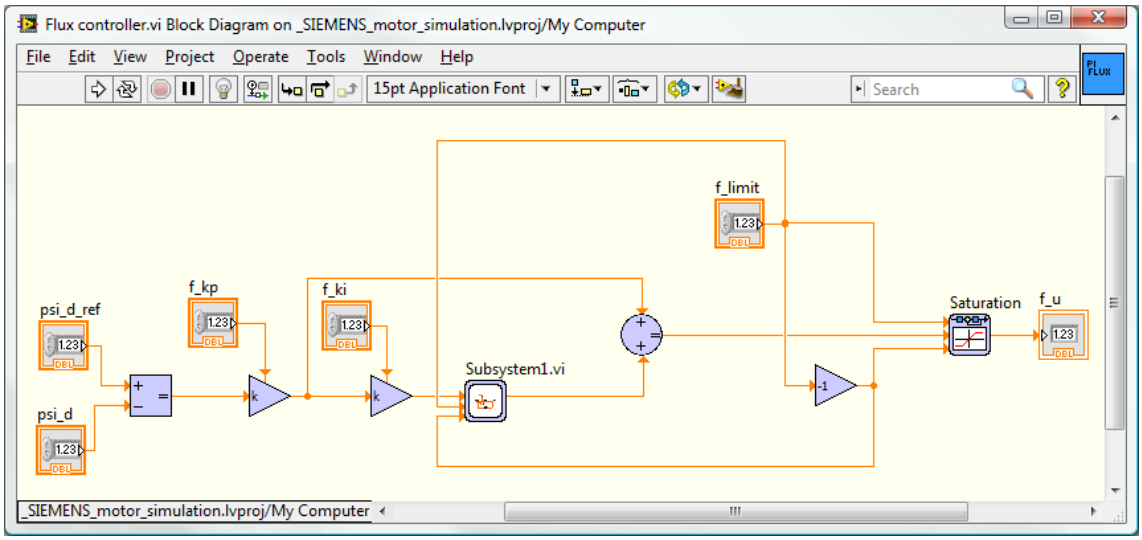
Príloha 28 _MAIN.vi



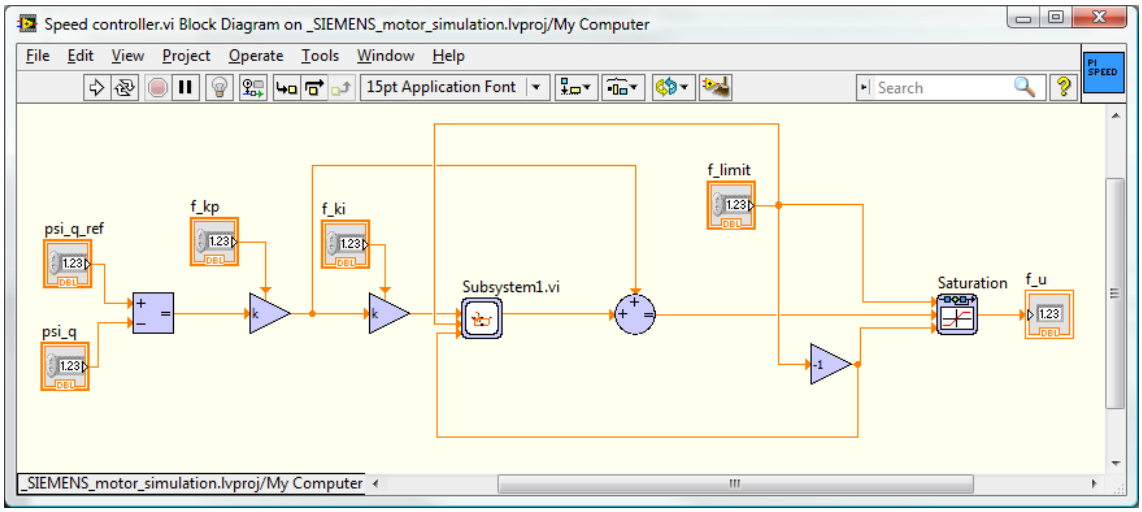
Príloha 29: Id controller.vi



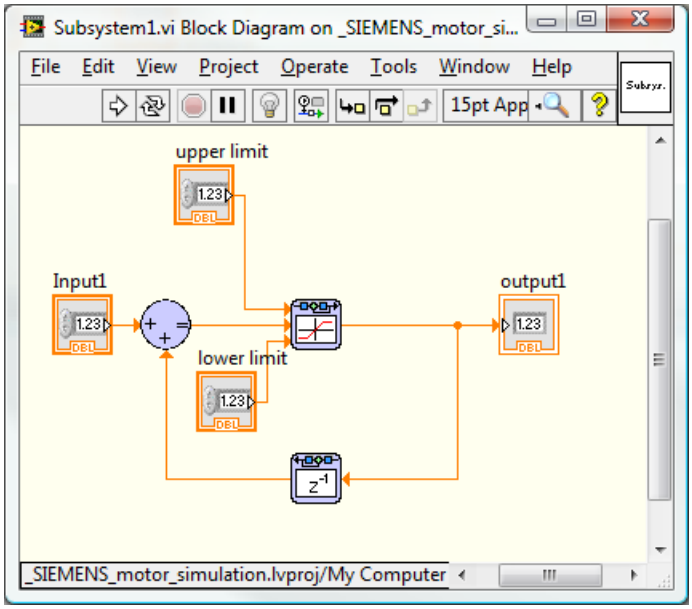
Príloha 30. Flux controller



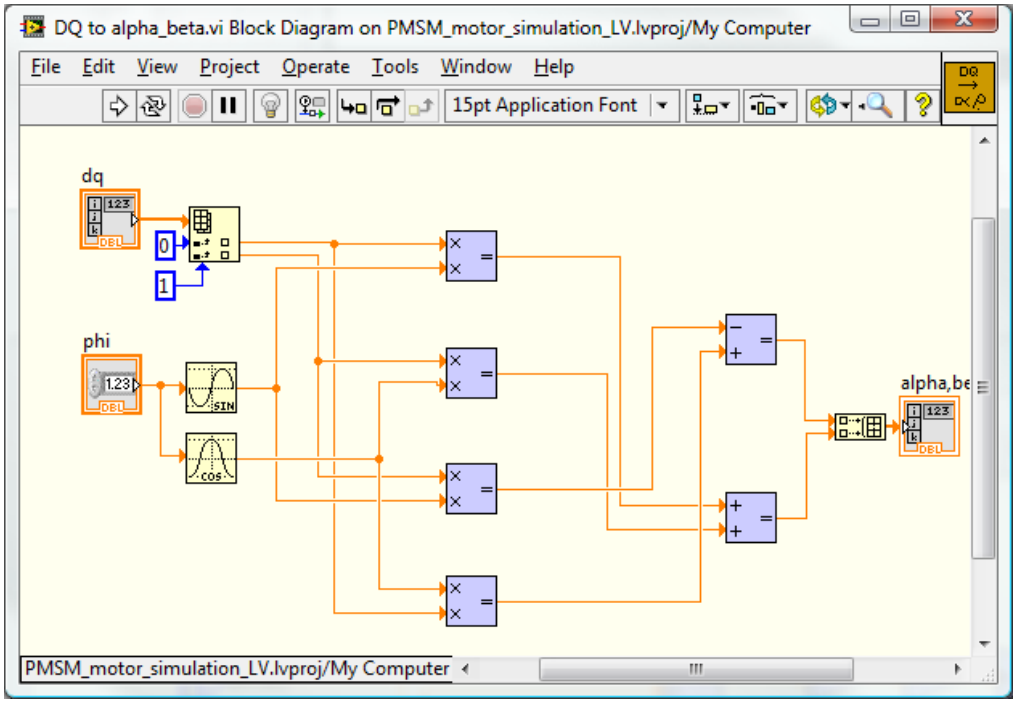
Príloha 31: Speed controller.vi



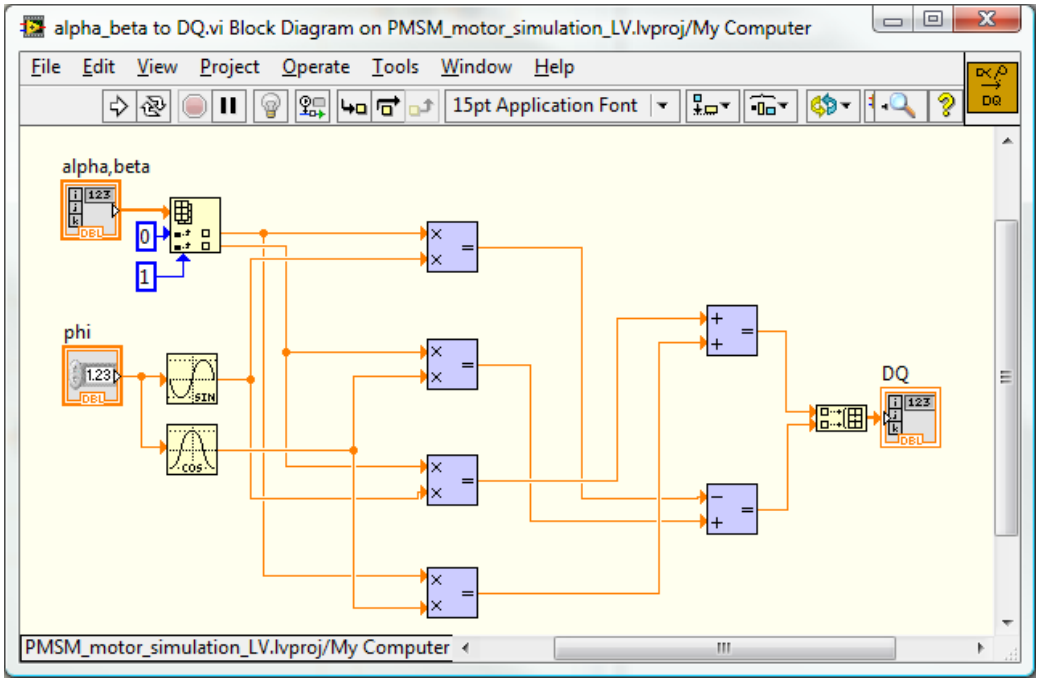
Príloha 32: Subsystem1.vi



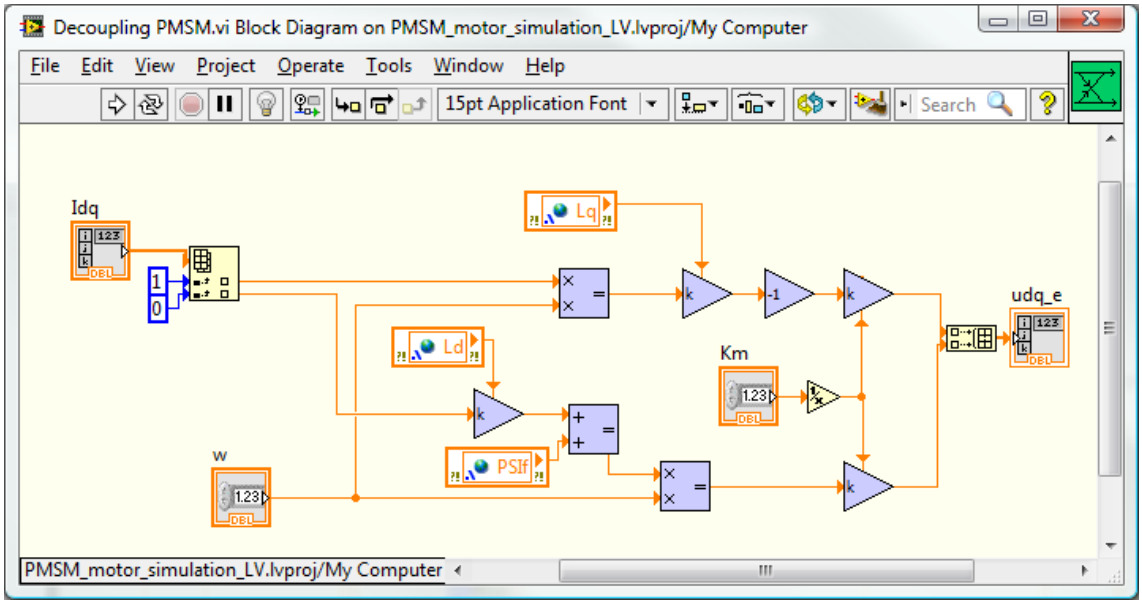
Príloha 33: DQ to alpha_beta.vi



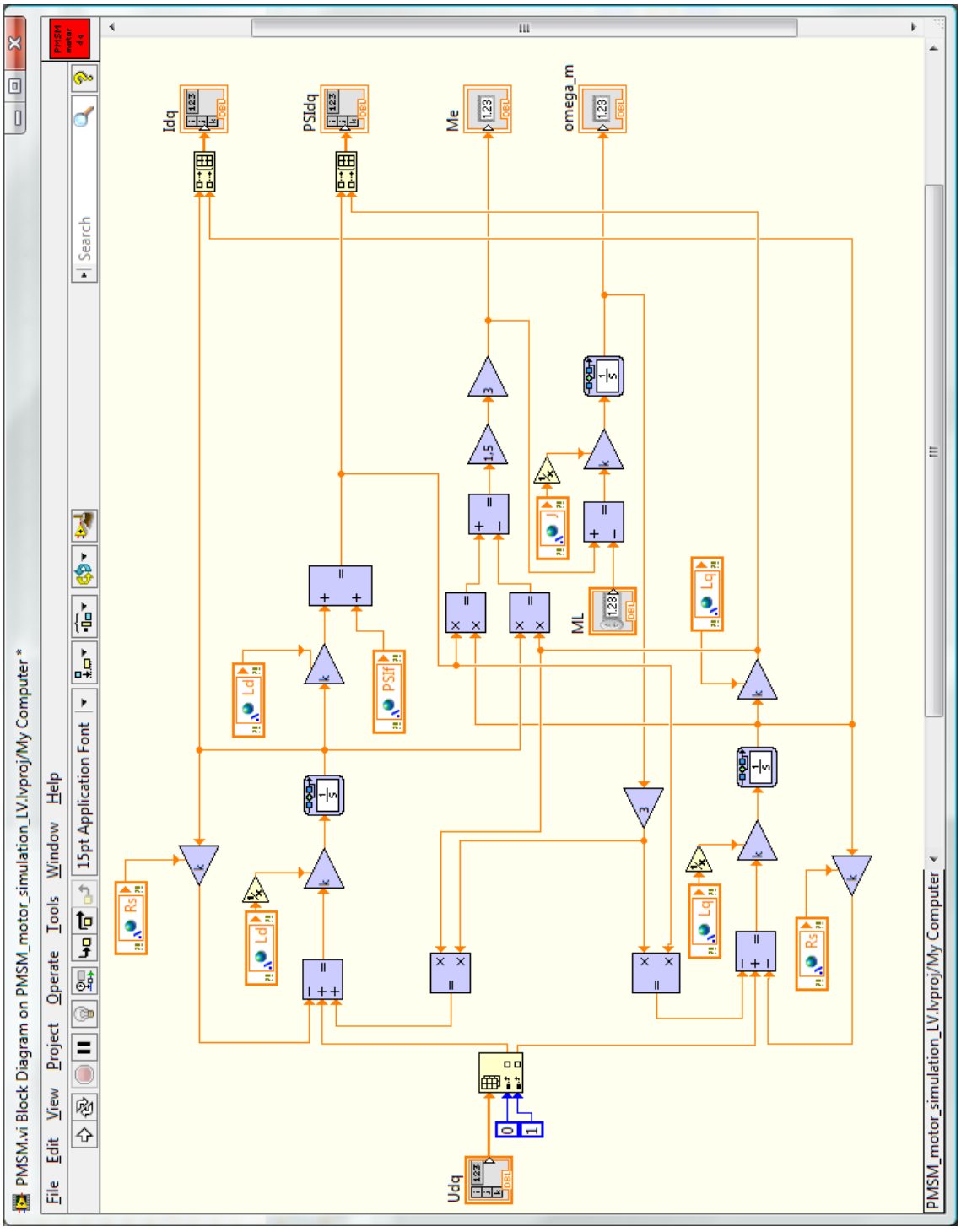
Príloha 34: alpha_beta to DQ.vi



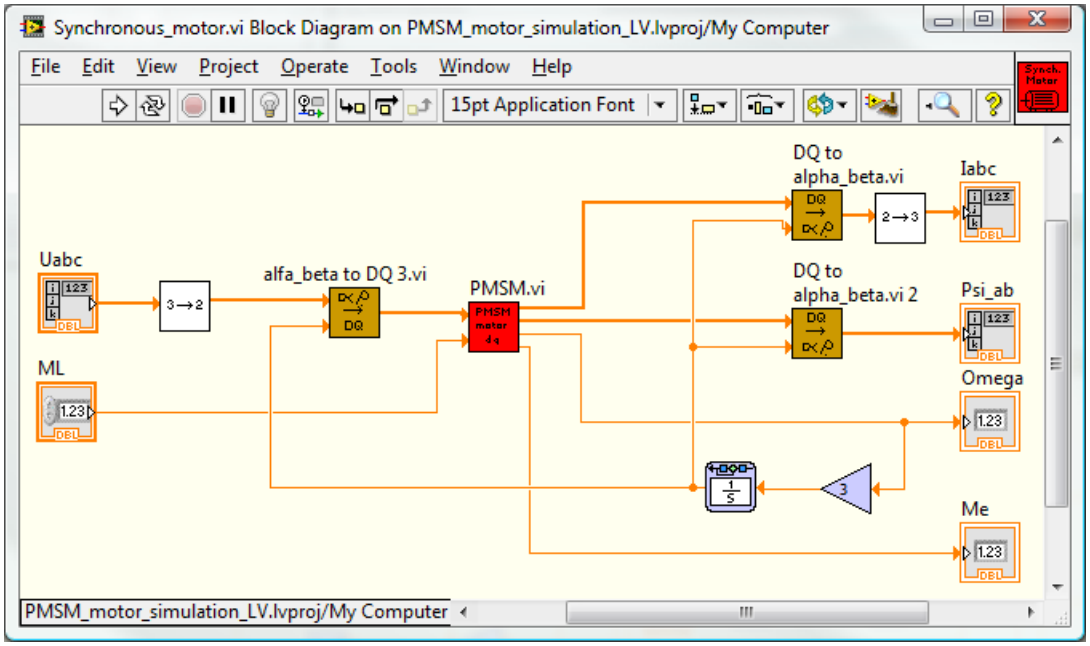
Príloha 35: Decoupling PMSM



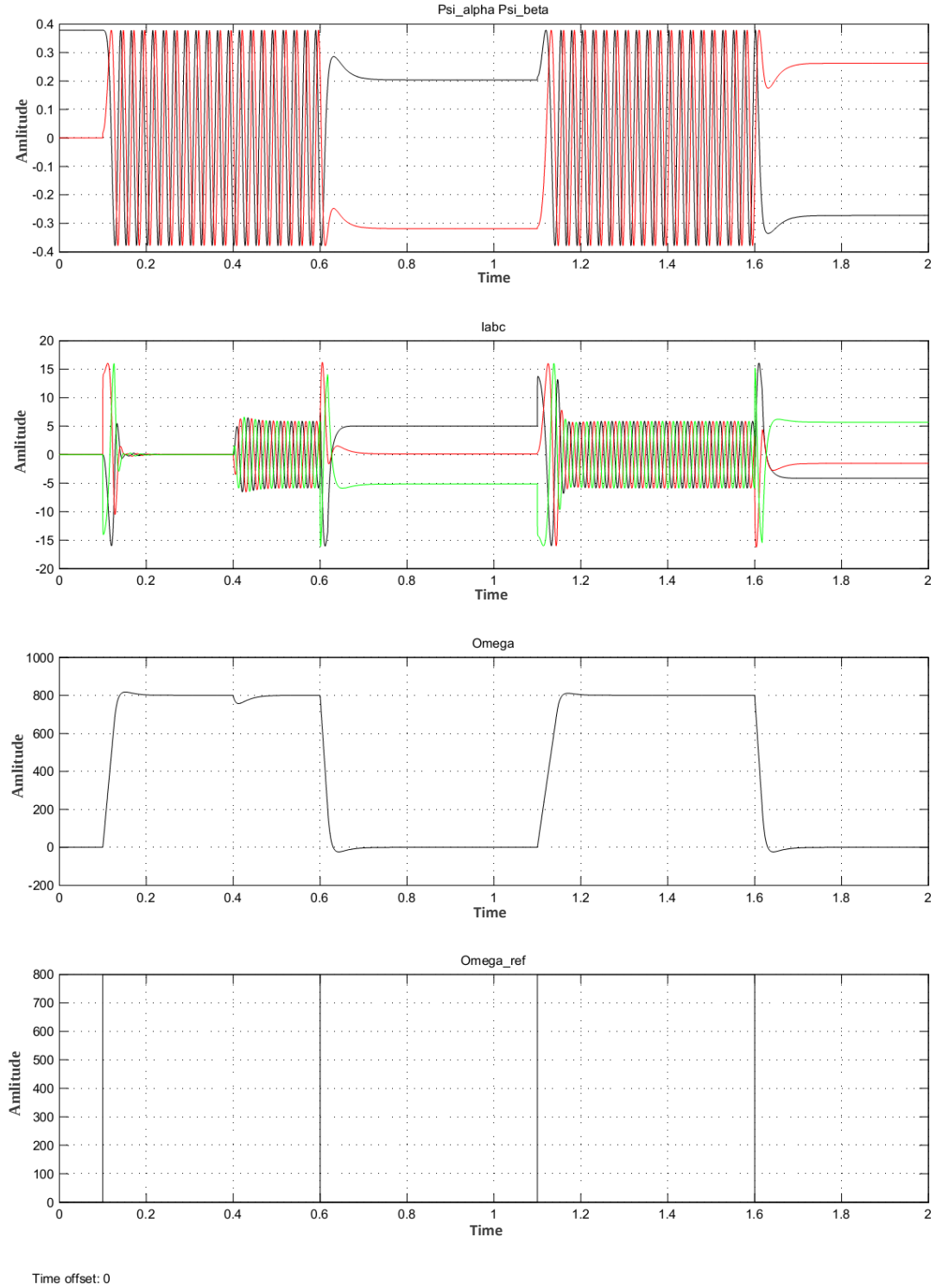
Príloha 36: PMSM.vi



Príloha 37: Synchronous_motor.vi



Príloha 38: Výsledné priebehy simulácie riadenia PMSM v prostredí Matlab



Príloha 39: Výsledné priebehy simulácie riadenia PMSM v prostredí LabVIEW