VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A
KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘÍCÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

IMPLEMENTACE IDENTIFIKAČNÍCH ALGORITMŮ V SIGNÁLOVÉM PROCESORU DSP56F8XXX

THE IMPLEMENTATION OF IDENTIFICATION ALGORITHMS IN DIGITAL SIGNAL PROCESSOR DSP56F8XXX

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE Bc. JIŘÍ JANUŠKA
AUTHOR

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

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Implementace identifikačních algoritmů v signálovém procesoru DSP56F8xxx

DIPLOMOVÁ PRÁCE

Studijní obor: Kybernetika, automatizace a měření

Student: Bc. Jiří Januška

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

Abstrakt:

Diplomová práce se zabývá matematickými popisy některých identifikačních metod aplikovaných na asynchronní motor s jejich následným odsimulováním v prostředí MATLAB/Simulink a poté implementací na digitální signálový procesor.

V první části práce je uveden základní matematický popis asynchronního motoru. Následující část obsahuje stručné přiblížení významu identifikačních metod s jejich rozdělením.

Nejdůležitější částí práce je popis principů 4 konkrétních identifikačních metod s jejich následnými simulacemi aplikovanými na model asynchronního motoru v prostředí MATLAB/Simulink. Výpočetní algoritmy těchto metod jsou pro každou metodu zvlášť následně implementovány ze Simulinku do digitálního signálového procesoru řady DSP56Fxxx firmy Freescale.

Klíčová slova: asynchronní motor, identifikace, signálový procesor

BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering and Communication

Department of Control, Measurement and Instrumentation

The implementation of the identification algorithms in digital signal processor DSP56F8xxx

DIPLOMA THESIS

Specialization of study: Cybernetics, Control and Measurement

Student: Bc. Jiří Januška

Supervisor: doc. Ing. Petr Blaha, Ph.D.

Abstract:

This diploma thesis deals with basic mathematic descriptions of some identification methods applicable to the induction motor with program simulation in MATLAB/Simulink and then with implementation of the methods in digital signal processor.

In the first chapter the basic mathematic description of the induction motor is mentioned. The following section includes short approach to meaning of the identification methods with their division.

The most important chapter of the thesis describes the four identification principles with their successive simulations applied to the model of the induction motor with program MATLAB/Simulink. The computational algorithms of these methods are consequently implemented one after another for each method from Simulink to digital signal processor DSP56F8xxx.

Key words: induction motor, identification, digital signal processor

Bibliografická citace

JANUŠKA, J. *Implementace identifikačních algoritmů v signálovém procesoru DSP56F8xxx.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 71 s. Vedoucí diplomové práce doc. Ing. Petr Blaha, Ph.D.

P r o h l á š e n í

„Prohlašuji, že jsem svou diplomovou práci na téma "Implementace identifikačních algoritmů v signálovém procesoru DSP56F8xxx" vypracoval zcela samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny náležitě 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 § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne: Podpis:

P o d ě k o v á n í

Tímto bych chtěl poděkovat vedoucímu této práce panu doc. Ing. Petru Blahovi Ph.D. za veškeré úsilí, snahu a zejména pomoc při řešení této diplomové práce. Také bych chtěl poděkovat svým blízkým za veškerou péči a obětavost v době studia a vypracování diplomové práce.

V Brně dne: Podpis:

OBSAH

1. ÚVOD 10
2. ASYNCHRONNÍ MOTOR 11
2.1 Rozdělení asynchronních motorů 11
2.2 Princip činnosti asynchronního motoru 12
2.3 MATEMATICKÝ POPIS aSYNCHRONNÍHO MOTORU 13
2.3.1 Základní rovnice popisující AM 13
3. IDENTIFIKACE PARAMETRŮ MOTORU 15
3.1 Metody pro identifikaci 15
3.1.1 On-line metody 16
3.1.2 Off-line metody 16
3.2 Modely použité v práci 16
4. DIGITÁLNÍ SIGNÁLOVÉ PROCESORY 17
4.1 Vývojová deska freescale 56F8300 DEMO 18
4.1.1 Klíčové rysy procesorů z rodiny DSP56800E 18
4.1.2 Periférie prvky na desce 20
4.1.3 CPU MC56F8323 21
4.2 Prostředí Codewarrior IDE 22
4.3 SFIO toolbox 24
4.4 Implementace identifikačních algoritmů 25
5. DC TEST 26
5.1 Blokové schéma SIMULACE v simulinku 26
5.2 Princip DC testu 27
5.2.1 Výpočet odporu statoru 28
5.3 Implementace DC testu na DSP 29
5.4 Blokové schéma implementace na DSP 29
5.5 Algoritmus DC testu v jazyce C 30
5.6 Zhodnocení implementace DC testu na DSP 32
5.7 Průběhy okamžitých hodnot veličin v čase 32
6. NO LOAD TEST 34
6.1 Blokové schéma simulace v Simulinku 34
6.2 Princip simulace 35

6.3 Určení parametru L_{sL} + L_m 35
6.3.1 Blok RMS 36
6.3.2 Princip identifikace 36
6.3.3 Výpočet parametrů L_{sl} + L_m 37
6.4 Implementace no-load testu na dsp 39
6.5 Blokové schéma implementace na DSP 40
6.6 Algoritmus No-load testu v jazyce C 40
6.6.1 Generování napájecího napětí 41
6.6.2 Algoritmus výpočtu efektivní hodnoty RMS ze signálu na DSP 42
6.6.3 Algoritmus výpočtu parametrů L_{sl} + L_m na DSP 43
6.7 Zhodnocení implementace No-load testu na DSP 46
6.8 Průběhy okamžitých hodnot veličin v čase 46
7. LOCKED ROTOR TEST 48
7.1 Blokové schéma simulace v Simulinku 48
7.2 Princip simulace 49
7.3 Určení prametrů L_{sl}+L_{rl} 49
7.3.1 Blok RMS 50
7.3.2 Princip identifikace 50
7.3.3 Výpočet parametrů L_{sl} + L_{rl} 50
7.3.4 Výpočet parametrů R_s + R_r 52
7.4 Implementace Locked rotor testu na DSP 52
7.5 Blokové schéma implementace na DSP 53
7.6 Algoritmus Locked rotor testu v jazyce C 54
7.6.1 Generování napájecího napětí 54
7.6.2 Algoritmus výpočtu efektivní hodnoty RMS ze signálu na DSP 54
7.6.3 Algoritmus výpočtu parametrů L_{sl} + L_{rl} na DSP 55
7.6.4 Algoritmus výpočtu parametrů R_s + R_r na DSP 55
7.7 Zhodnocení implementace Locked rotor testu na DSP 56
7.8 Průběhy okamžitých hodnot veličin v čase 56
8. ONE PHASE TEST 58
8.1 Blokové schéma simulace v Simulinku 58
8.2 Princip simulace 58

8.3 Určení prametrů L_{sl}+L_{rl} 59
8.3.1 Blok RMS 59
8.3.2 Princip identifikace 60
8.3.3 Výpočet parametrů L_{sl} + L_{rl} 62
8.3.4 Výpočet parametrů R_s + R_r 64
8.4 Implementace One phase testu na DSP 64
8.5 Blokové schéma implementace na DSP 65
8.6 Algoritmus One phase testu v jazyce C 66
8.6.1 Generování napájecího napětí 66
8.6.2 Algoritmus výpočtu efektivní hodnoty RMS ze signálu na DSP 66
8.6.3 Algoritmus výpočtu parametrů L_{sl} + L_{rl} na DSP 67
8.6.4 Algoritmus výpočtu parametrů R_s + R_r na DSP 67
8.7 Zhodnocení implementace One phase testu na DSP 68
8.8 Průběhy okamžitých hodnot veličin v čase 68
9. ZÁVĚR 70
10. POUŽITÁ LITERATURA 71

SEZNAM OBRÁZKŮ

Obrázek 1: Vývojová deska FREESCALE 56F8300DEMO[9] 20
Obrázek 2: Pohled na uživatelské rozhraní programu CodeWarrior 5.6 22
Obrázek 3: Okno Nápověda pro CodeWarrior 23
Obrázek 4: Blokové schéma DC testu 26
Obrázek 5: Statorové vinutí s připojeným napětím 28
Obrázek 6: Blokové schéma implementace DC testu na DSP 29
Obrázek 7: Okamžitý průběh proudů statorem 32
Obrázek 8: Okamžitý průběh složek toku rotorem 33
Obrázek 9: Okamžitý průběh úhlové rychlosti 33
Obrázek 10: Blokové schéma simulace No-load testu 34
Obrázek 11: Blokové schéma bloku RMS 36
Obrázek 12: Blokové schéma implementace No-load testu na DSP 40
Obrázek 13: Okamžité průběhy proudů statorem 47
Obrázek 14: Okamžitý průběh úhlové rychlosti 47

Obrázek 15: Blokové schéma Locked rotor testu 48
Obrázek 16: Blokové schéma implementace Locked rotor testu na DSP..... 53
Obrázek 17: Okamžité průběhy proudů statorem 56
Obrázek 18: Okamžité průběhy složek magnetického toku rotorem 57
Obrázek 19: Okamžitý průběh úhlové rychlosti 57
Obrázek 20: Blokové schéma simulace One phase testu 58
Obrázek 21: Blokové schéma implementace One phase testu na DSP..... 65
Obrázek 22: Okamžité průběhy proudů statorem 68
Obrázek 23: Okamžité průběhy složek magnetického toku rotorem 69
Obrázek 24: Okamžitý průběh úhlové rychlosti 69

SEZNAM TABULEK

Tabulka 1: Výpočet odporu statoru..... 31

SEZNAM SYMBOLŮ

$\sin(x)$	- goniometrická funkce
x	- obecná proměnná
j	- imaginární jednotka [-]
t	- čas [s]
φ	- fáze napájecího napětí [-]
ω	- mechanická úhlová rychlost [rad/s]
ω_0	- úhlová frekvence napájecí sítě [rad/s]
ω_s	- synchronní rychlost magnetického pole [rad/s]
f_{vz}	- vzorkovací frekvence [Hz]
T_{vz}	- perioda vzorkování [s]
s	- skluz asynchronního motoru [-]
z_p	- počet pólových dvojic motoru [-]
π	- Ludolfovo číslo [-]
U_m	- amplituda napětí fáze [V]
U_a, U_b, U_c	- napájecí napětí jednotlivých fází [V]

I_a, I_b, I_c	- proudy jednotlivými fázemi [A]
R_s	- odpor statoru [Ω]
R_r	- odpor rotoru [Ω]
I_{ef}	- efektivní hodnota proudu [A]
U_{ef}	- efektivní hodnota napětí [V]
Q_a	- jalový výkon jedné fáze [W]
P	- činný výkon motoru [W]
P_{ef}	- efektivní hodnota činného výkonu [W]
L_m	- magnetizační indukčnost [H]
X_m	- magnetizační induktance [Ω]
L_{sl}	- rozptylová indukčnost statoru [H]
X_{sl}	- rozptylová induktance statoru [Ω]
L_{rl}	- rozptylová indukčnost rotoru [H]
a, b, c	- označení svorek na statoru
$\bar{u}_s$	- vektor statorového napětí [V]
$\bar{u}_r$	- vektor rotorového napětí [V]
$\bar{\psi}_s$	- vektor magnetického toku statoru [Wb]
$\bar{\psi}_r$	- vektor magnetického toku rotoru [Wb]
M_z	- zátěžný moment [Nm]

1. ÚVOD

Dnešní doba plná technologií s pohony a akčními členy přináší značné nároky na přesnou a zároveň kvalitní regulaci. Dnešní přístupy s identifikačními algoritmy dokáží přesně a kvalitně identifikovat ty parametry pohonu, které mají největší vliv na dosaženou přesnost a kvalitu regulace. Tyto metody jsou víceméně základem pro přesnou identifikaci elementárních parametrů (např. statorové a rotorové odpory, indukčnosti ad.) strojů, ať už synchronních či asynchronních, jejichž znalost nám umožňuje co nejpřesněji nastavit a následně řídit tyto části určité technologie. Tyto metody se neustále vyvíjejí a zdokonalují zejména pomocí modelovacích prostředků a výpočetní techniky.

V poslední době lze v oblasti elektrických pohonů pozorovat značný nárůst regulovaných pohonů. Vývoj je podmíněn zejména kvalitou polovodičových součástek a prvků, tvořící základní stavební strukturu regulační soustavy (řízení - pohon). Zejména použití prvků výkonové a řídící techniky, založené právě na polovodičových prvcích, umožňuje realizaci pohonů v praxi vyznačující se parametry s vnitřní dynamikou. Právě tyto parametry lze do jisté míry ovlivňovat kvalitním řízení či regulací. Algoritmy schopné pracovat při řízení pohonů se vyznačují vysokou mírou adaptibility a poměrně snadnou aplikací. Bývají téměř výhradně spjaty s využitím mikroprocesorových struktur, na kterých jsou algoritmy implementovány. [2].

Tato diplomová práce se zabývá popisem, následným modelováním s implementací identifikačních metod asynchronního motoru, jejichž výpočetní algoritmus běží v rámci MATLAB/Simulink. Dále jsou právě tyto algoritmy převedeny (modifikovány) tak, aby byly výpočetně zpracovány na straně digitálního signálového procesoru (DSP). Cílem práce bylo seznámit se s vybranými metodami identifikace parametrů asynchronního motoru, ověřit (nasimulovat) správnost těchto metod v prostředí MATLAB/Simulink. Hlavním úkolem však bylo tyto identifikační metody implementovat ze Simulinku do signálového procesoru řady 56F8xxx firmy Freescale. Práce je uceleně rozdělena do několika kapitol. Identifikační metody jsou realizovány v prostředí Simulink a dále pak v prostředí CodeWarrior IDE, kde jsou identifikační algoritmy naprogramovány v jazyku C.

2. ASYNCHRONNÍ MOTOR

Asynchronní motory (AM) jsou spolehlivější, levnější co se týče nákladů na výrobu, lehčí a zejména méně složitější, co se týče konstrukce, ve srovnání se stejnosměrnými motory (SM). AM s rotorem nakrátko patří mezi nejpoužívanější stroje v elektrických pohonech vůbec. Dnešní výrobní průmysl je schopen obsloužit široké spektrum potenciálních zákazníků, kteří pro svoji technologii potřebují pohon především s přijatelnou pořizovací cenou a zejména se specifickými parametry jako např. výkonem od několika W či mnohem výkonnější formu pohonu, např. pro nějakou dynamicky náročnou aplikaci, která si žádá nutný výkon řádově několik MW s velkým rozsahem. Jejich nevýhodou vůči SM je složitější algoritmus pro řízení, zejména otáček. To však v dnešní době plné mikroprocesorové techniky a řídící techniky, založené na polovodičových technologiích, zčásti odpadá. Díky těmto a dalším předpokladům jsou asynchronní motory schopné konkurovat stejnosměrným pohonům (servomotory, krokové motory) pro polohové dynamicky náročné aplikace.[2]

2.1 ROZDĚLENÍ ASYNCHRONNÍCH MOTORŮ

<u>Konstrukční rozdělení:</u>

Podle uspořádání a počtu napájecích vodičů (fází) na statorovém vinutí

a) **jednofázové** – využití pro aplikace s menší výkonovou náročností (pro rozběh je nutné připojit rozběhový kondenzátor, indukčnost či činný odpor), využití zřídka např.u automatických praček pro domácnosti
b) **trojfázové** – nejpoužívanější typy AM motorů, uplatnění v širokém spektru využití (čerpadla, pohony pro elektrickou trakci - lokomotivy, tramvaje)

Dle uspořádání rotorového vinutí

a) **motor s kroužkovou kotvou** – využívá se možnosti snadné změny rotorového odporu souměrným trojfázovým předřadným odporem připojeným na kroužky a tím zmenšit záběrný proud morou nebo jeho otáčky a zvětšit záběrný moment

b) **motor s klecovou kotvou spojenou nakrátko** – nejčastější případ, kdy je rotor zhotoven z kotvy nakrátko, která se skládá z vodivých tyčí (měď, hliník, mosaz), spojených čelními kruhy svařením nebo pájením natvrdo (konstrukce tvoří klec), rotor a tím samotný motor je téměř nezničitelný a značně spolehlivý co se týče konstrukce

2.2 PRINCIP ČINNOSTI ASYNCHRONNÍHO MOTORU

Při analýze vlastností elektrických strojů (statických i dynamických) je nutné vycházet z obecných předpokladů, založených na elektromagnetických a elektromechanických jevech. Indukční stroj lze považovat za obecný příklad několikafázového transformátoru. Rozdíl vzhledem k transformátoru spočívá v jiné konstrukci a uspořádání. Rotorová a statorová vinutí jsou uložena ve dvou oddělených železných jádrech, mezi kterými je vzduchová mezera. Rotor je část motoru, která se vzhledem ke statoru otáčí a je spojena s hřídelí, která je vyvedena mimo samotné tělo motoru. Princip funkce motoru spočívá ve vytvoření točivého magnetického pole ve vzduchové mezeře. Točivé magnetické pole vzniká připojením vinutí statoru na symetrický trojfázový zdroj napětí. [1][2] Fázová napětí je možno vyjádřit následujícími vztahy:

$$\begin{aligned} U_a &= U_m \cdot \sin(\omega_0 t) \\ U_b &= U_m \cdot \sin(\omega_0 t + \frac{2 \cdot \pi}{3}) \\ U_c &= U_m \cdot \sin(\omega_0 t + \frac{4 \cdot \pi}{3}) \end{aligned} \tag{2.1}$$

kde pro úhlovou frekvenci napájecí sítě platí

$$\omega_0 = 2 \cdot \pi \cdot f \tag{2.2}$$

kde π je konstanta a f je frekvence napájecí sítě.

Rotující magnetické pole indukuje v tyčích statoru proudy, které zpětně reagují s magnetickým polem a vzniká elektromotorická síla, která následně roztáčí rotor. Hodnota otáček rotoru se zvyšuje, teče jím proud. Výkon motoru je přímo úměrný úhlové rychlosti otáčivého pole a momentu přeneseného na hřídel motoru. [2] Vztah mezi počtem pólových dvojic motoru, synchronní rychlostí otáčivého pole a skutečnou úhlovou frekvencí napájecí sítě dává následující vztah:

$$\omega_s = \frac{\omega_0}{z_p} \tag{2.3}$$

Skluz je veličina udávající rozdíl mezi skutečnou rychlostí otáčení motoru ω a synchronní rychlostí magnetického pole ω_s.

$$s = \frac{\omega_s - \omega}{\omega_s} \tag{2.4}$$

2.3 MATEMATICKÝ POPIS ASYNCHRONNÍHO MOTORU

Asynchronní motor s klecovou kotvou je nejpoužívanější pohon v elektrických regulovaných soustavách. Pod pojmem regulovaný se má na mysli takový pohon, u kterého řídíme otáčky, polohu, moment atd. Pro ustálené stavy i přechodné děje jsou důležité hodnoty amplitud elektromagnetických veličin. Vzhledem k tomu, že je AM nelineární, vyskytují se v jeho popisu diferenciální rovnice. [2]

2.3.1 Základní rovnice popisující AM

Vztahy uvedené níže platí pro všechny typy asynchronních motorů a vycházejí z teorií o obecném elektrickém stroji.[2]

<u>Vektorové rovnice popisující statorová a rotorová napětí:</u>

$$\bar{u}_s = R_s \bar{i}_s + \frac{d\bar{\psi}_s}{dt} + j\omega_s \bar{\psi}_s \tag{2.5}$$

$$\bar{u}_R = 0 = R_R \bar{i}_R + \frac{d\bar{\psi}_R}{dt} + jz_p(\omega_s - \omega)\bar{\psi}_R \tag{2.6}$$

kde $\bar{u}_s$ je vektor statorového napětí, R_s je odpor statoru, R_R je odpor rotoru, $\bar{i}_s$ je vektor statorového proudu, $\bar{i}_R$ je vektor proudu rotorového, dále $\bar{\psi}_s$ je magnetický

tok statorem a $\overline{\psi}_R$ je magnetický tok rotorem, z_p je počet pólových dvojic a člen $(\omega_s - \omega)$ je rozdíl hodnot úhlové rychlosti magnetického pole a skutečné rychlosti otáčení rotoru (viz. vztah 2.4), t je čas. Ve vztahu (2.6) je $\overline{u}_R = 0$, neboť se jedná o model motoru s kotvou spojenou nakrátko, lépe řečeno jde o motor jehož rotorové vinutí je zapojeno do zkratu.

<u>Momentové rovnice:</u>

$$J\frac{d\omega}{dt} = (M - M_z) \tag{2.7}$$

$$M = 1{,}5 z_p \operatorname{Im}\left[\overline{\psi}_s \cdot \overline{i}_s\right] \tag{2.8}$$

kde J je celkový moment setrvačnosti motoru, člen $(M - M_z)$ udává velikost rozdílu hodnot momentu motoru a zátěžného momentu, člen z_p udává počet pólových dvojic, ω je úhlová rychlost otáčení motoru, t je čas.

Z rovnic (2.6) a (2.7) je vidět nelinearita modelu daného motoru, která je zde reprezentována diferenciálními rovnicemi prvního řádu. [2]

3. IDENTIFIKACE PARAMETRŮ MOTORU

V posledních desetiletích 20. století dosáhnutý stupeň vývoje polovodičové a mikroprocesorové techniky umožnil použití nových algoritmů pro řízení asynchronních motorů. Jedná se zejména o metody, které dosahují vysoké statické a dynamické přesnosti při řízení elektromechanických veličin (proud, magnetický tok, moment, otáčky, poloha). Mezi tyto metody řadíme např. **vektorové řízení momentu**, **skalární řízení**, **přímé řízení momentu** (v literatuře označováno jako DTC) ad. Díky tomu jsou pohony s těmito motory schopny v plné míře konkurovat jednosměrným servopohonům v dynamicky náročných aplikacích. Použití střídavých servopohonů v různých odvětvích průmyslové automatizace zaznamenalo markantní nárůst, tento trend se udržuje nadále a v souvislosti s dalším vývojem a s tím spojeným zdokonalováním technologií střídavých pohonů dozná v budoucna nepochybně výrazných kroků směrem kupředu.

Dnešní pohony musí být schopny vyhovět nejrůznějším požadavkům, které si kladou jejich další aplikace, tj. např. přesná a rychlá regulace úhlové rychlosti motoru v statickém i dynamickém provedení. Aby toto řízení bylo co nejpřesnější a aby byly splněny požadavky, které jsou v krátkosti zmíněny výše, musí existovat metody navržené k tomu, aby nám přesně a rychle identifikovaly parametry samotných pohonů (tj. hodnoty odporů statoru a rotoru, indukčností, časových konstant atd.). Znalost těchto parametrů je nutná k nastavení regulačních prvků v řídícím systému a z toho vyplývá, že bez přesné identifikace není možné tyto pohony kvalitně regulovat či řídit ve svých specifických aplikacích.

Některé parametry jsou téměř nezávislé na provozních podmínkách (indukčnosti ve statoru a rotoru), avšak ty další z většiny jako odpory rotoru a statoru už jsou ovlivnitelné celou řadou faktorů, z nichž mezi hlavními jsou nepochybně teplotní změny ve strukturách použitých materiálů, dále také magnetické a další vlastnosti železa (hystereze, permeabilita).[2]

3.1 METODY PRO IDENTIFIKACI

Tyto metody se od sebe navzájem liší způsobem přístupu hledání a určování parametrů asynchronního motoru. Jedná se o metody **off-line** a **on-line**.

3.1.1 On-line metody

Tyto metody a přístupy na nich založené slouží k určování parametrů asynchronního motoru, který je právě v provozu (on-line). Stroj může být zatížen či jen běžet v režimu no-load, tj. bez mechanického zatížení hřídele. Pohon vykazuje mechanický výkon roven přibližně výkonu nutnému ke krytí ztrát vlivem např. tření či odporu vzduchu. Parametry motoru se mění s časem během činnosti a proto jsou tyto metody implementovány s ohledem na tuto skutečnost. Stěžejními předpoklady u těchto metod jsou kvalitní hodnoty počátečních podmínek. Metody jsou značně závislé na výkonu řídícího mikroprocesoru (důraz na real time). Právě schopnost identifikace v reálném čase staví tyto metody na poměrně kvalitní pozici. [6]

3.1.2 Off-line metody

Jedná se o metody identifikující parametry zařízení, které je v ustáleném stavu, tj. např. před spuštěním (nekoná otáčivý či další jiný pohyb). Získané identifikované parametry jsou nutné k nastavení složek regulátorů a dalších prvků určených pro činnost pohonu. [6]

3.2 MODELY POUŽITÉ V PRÁCI

Simulace, modelování a samotná implementace na DSP, potřebné k identifikaci parametrů asynchronního motoru, byly prováděny na již dříve vytvořeném modelu motoru, který je určen k bezsnímačovému řízení asynchronního motoru. Blokový model samotného motoru, vytvořený za pomocí software MATLAB-Simulink společnosti MATHWORKS, plně kopíruje typ zařízení, který je na ÚAMT (Ústav automatizace a měřící techniky) a především v CAK (Centrum aplikované kybernetiky) využíván pro výzkumné účely.

Tento model či blok je navržen tak, aby byl pokud možno přesnou kopií reálného motoru, co se týče parametrů ovlivňujících jeho charakter. Obsahuje dva vstupní porty (napájecí napětí $u_{a,b,c}$ a zatěžovací moment M_z) a dále výstupní porty, na kterých je možno sledovat veličiny typu proud statorem $i_{a,b,c}$, celkový magnetický tok rotorem ψ_r či případně otáčky motoru ω. Připomínám, že model motoru je nelineární, stejně jako reálný motor, který je jeho předlohou. Jeho matematický popis je postaven na diferenciálních rovnicích.

4. DIGITÁLNÍ SIGNÁLOVÉ PROCESORY

Množství informací, jejich kvalita a zejména rychlost zpracování (práce v reálném čase = real time) patří k rozhodujícím faktorům, které ovlivňují kvalitu průběhu jakéhokoliv procesu. Např. zpracování značného objemu informací - zachycení obrazu, komprese, rozpoznání specifických znaků, záznam na standardní médium či odeslání informací obrazu jinému zařízení připojenému do komunikační sítě. Jiným příkladem může být průběžná (on-line) identifikace parametrů regulované soustavy, kde jsou parametry zjišťovány např. na základě frekvenčního spektra soustavy a dále toto spektrum na základě vstupních a výstupních signálů.[7]

Hlavním a nejčastějším úkolem digitálního signálového řízení či regulace je získání informací ze signálu a jejich zpracování. Informace mohou být obsaženy ve formě amplitudy, frekvence, spektrální hustoty , časového poměru. U dnešních DSP je standardem 16 bitové nebo 32 bitové binární číslo, reprezentující v daném okamžiku jednu složku signálu, což je co se týče rozsahu dostatečně velké číslo pro hodnověrné a přesné vyjádření signálu ve většině průmyslových, řídicích a monitorovacích aplikacích. Použití těchto digitálních signálových procesorů nachází stále větší uplatnění téměř ve všech oblastech elektroniky, a nejen v průmyslu, ale také v televizní, audio a spotřební elektronice. Je totiž výhodnou alternativou analogových systémů složených z mnoha součástek, ve srovnání s digitálním signálovým procesorem. Vlastní součástky u analogových zapojení tvoří zpravidla složitý hardware, který je nahrazen méně komplikovaným zapojením DSP, kde je podstatná část výpočtů a řízení provázena právě v procesoru na základě naprogramovaného softwaru. Tím je možné dosáhnout podstatně nižších výdajů za zařízení srovnatelné ne-li výkonnostně a kvalitativně lepší. [7].

Mezi přední světové výrobce digitálních signálových procesorů a vývojových desek patří koncerny jako Freescale Semiconductor, Analog Devices, Texas Instruments a další.

4.1 VÝVOJOVÁ DESKA FREESCALE 56F8300 DEMO

Digitální signálový procesor má obdobné vlastnosti a způsob programování jako univerzální procesor (mikroprocesor). Rozdíly jsou především v odlišné architektuře a instrukčním souboru. Hlavní výhodou však je mnohem rychlejší a účinnější výpočet pomocí nejpoužívanějších matematických operací při číslicovém zpracování signálu (operace násobení, diskrétní transformace, číslicová filtrace ad.). Převážná většina DSP je založena na tzv. **harvardské architektuře** s odděleným zpracováním instrukcí a dat, což umožňuje současnou (paralelní) manipulaci s daty i adresami bez jakékoliv možnosti samomodifikace či poškození programu (Von Neumann architektura). Harvardská architektura se dnes pro své vlastnosti řadí na první místo před Von Neumann co se týče využití. U rodiny procesorů 56800E se využívá **duální harvardská** architektura. Vysoké rychlosti výpočtu se dosahuje, podobně jako u univerzálních procesorů architekturou, paralelním zpracováním úloh více procesory najednou a zvyšováním taktovací frekvence, která se pohybuje v řádech desítek MIPS (MIPS = 1 000 000 instrukcí provedených za 1s).

Deska je osazena patřící do vývojové řady či rodiny 56800E. Jádro těchto procesorů v sobě kombinuje výpočetní výkon signálového procesoru s programovacími vlastnostmi mikroprocesorů určených pro řídící aplikace. Základem DSP je 16-bitová centrální řídící jednotka (CPU) a univerzální instrukční soubor. Významným prvkem tohoto DSP je integrace emulátoru přímo do čipu procesoru. V praxi to znamená, že odpadá nutnost používat pro ladění aplikací další externí zařízení, ale stačí propojit PC a vývojovou desku příslušnými prvky. Typy procesorů z řady DSP56800E se od sebe navzájem odlišují hlavně počtem a typem periferií a zejména kapacitou pamětí pro program či data (RAM a FLASH).

4.1.1 Klíčové rysy procesorů z rodiny DSP56800E

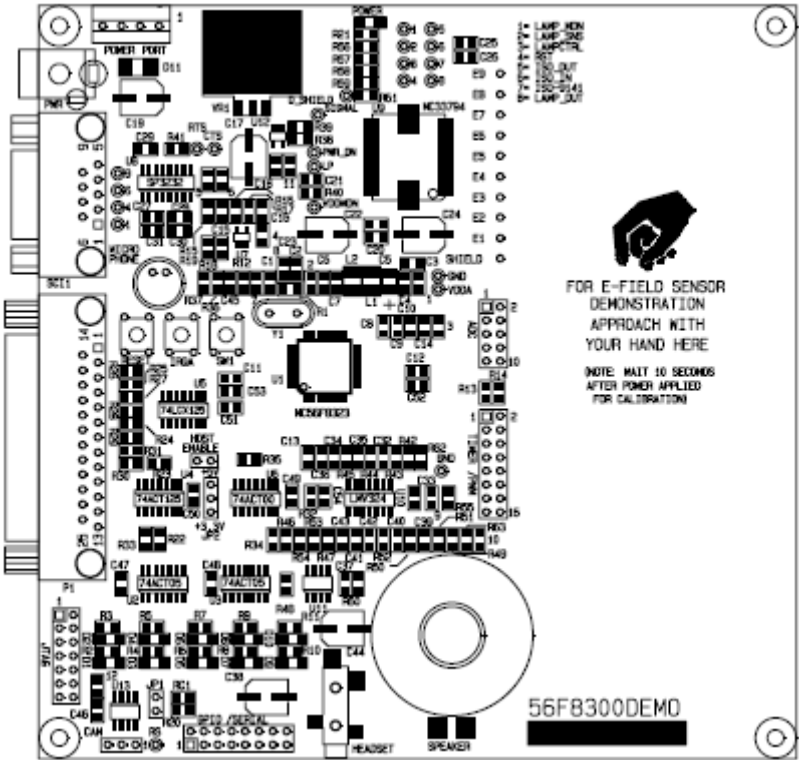
Architektura DSP56800E v sobě sdružuje široké vlastnosti jako např. vysoký výpočetní výkon, nižší cenu aplikací zejména zjednodušují poměrně snadný vývoj. Typické rysy pro DSP z rodiny DSP56800E jsou následující:

- **Vysoký výkon** – architektura DSP56F800E podporuje většinu středně výkonných DSC aplikací (DSC = Digital Signal Controller).

- **Kompatibilita** – podpora kompatibility aplikací vytvořených pro DSP z rodiny DSP56800 na novější variantu DSP56800E.
- **Snadné programování** – je velmi podobné klasickému programovaní tradičních mikroprocesorů. Instrukční sada podporuje jak frakční (zlomkovou) tak celočíselnou aritmetiku poskytující pružnost, která je důležitá pro optimální implementaci algoritmů.
- **Podpora vyšších programovacích jazyků** – především jazyk C, jako jeden z vyšších programovacích jazyků, je vhodný k DSP56800E architektuře
- **Bohatost instrukční sady** – poskytuje řízení, bitové manipulace a celočíselné procesní instrukce. Silné adresní módy a řada datových formátů je rovněž poskytována. Výsledkem je kompaktní a vysoce účinný kód.
- **Vysoká hustota kódu** – základní instrukční slovo pro DSP56800E má rozsah 16-bit spolu s více slovními instrukcemi pro více komplexních operací, ze kterého vyplývá optimální hustota kódu
- **Multi-tasking podpora** – implementace operačního systému reálného času nebo jednoduchého multi-tasking systému je mnohem snadnější na DSP56800E než na DSC. Architektura poskytuje plnou podporu pro softwarové zásobníky, rychlá 32-bit souvislost mezi ukládáním a obnovením do nebo ze systémového zásobníku
- **Přesnost** – architektura umožňuje přesné výpočty.
- **Paralelismus** – souběžnost prováděných instrukcí
- **Ladění v reálném čase** – technologie emulátoru implementovaného na čipu (Enhanced OnCE) poskytuje jednoduché, levné, nerušivé a zejména časové nezávislé přistupování do vnitřní struktury jádra DSP56800E. Při využití OnCE má programátor plnou kontrolu nad procesorem prováděnými operacemi, zjednodušující a zrychlující ladění bez toho, aniž by bylo nutné zdržovat výpočetní jádro

Právě tyto výše uvedené vlastnosti poskytují architektuře DSP56800E vysoký výkon a aplikovatelnost pro řadu aplikací. Jsou vynikající volbou pro bezdrátové DSC aplikace, průmyslové řízení a či pro další embedded DSC aplikace, které potřebují ke své činnosti vysoký výpočetní výkon. [9]

4.1.2 Periférie prvky na desce



Obrázek 1: Vývojová deska FREESCALE 56F8300DEMO[9]

Vývojová deska 56F8300DEMO v sobě sdružuje následující komponenty a periférie:

- CPU MC56F8323VF
- RESET přepínač
- IRQ přepínač
- SW1 přepínač
- GPIO / SERIAL port
- TIMER/ PWM port
- ADC Port (10 port)
- JTAG / OnCE Port (14 port)
- Host JTAG Port (P1- DB25P)
- COM1 Port - SCI - RS-232
- CAN Port - 1Mb transceiver
- PWR – konektor napájení
- Power indikace

- 10 vlastních indikátorů (LED)
- Mikrofon se zesilovačem
- Stereo Audio výstup s filtry AMP 3.5mm Stereo konektor + reproduktor

Na vývojové desce 56F8300DEMO je tedy osazen čip **CPU MC56F8323VF**. Deska dále obsahuje sériový port COM pro rozhraní RS-232, dále paralelní port LPT. K dispozici jsou také další porty a to JTAG či rozhraní High speed CAN. Na desce jsou rovněž implementovány jednoduché spínače, dále signalizační LED diody. Napájení desky je řešeno přes napájecí port pomocí přídavného napěťového zdroje. Vzhledem k tomu, že je na desce implementována napěťová regulace, je možné využít běžné napájení +5V DC a +3.3V DC.

4.1.3 CPU MC56F8323

Jedná se tedy o procesor patřící do rodiny digitálních signálových procesorů 56800E, který v sobě skrývá řadu vlastností uvedených v kapitole 4.1.1. Jedná se tedy o 16-bitový procesor vytvořený na duální harvardské architektuře, někdy označované jako **modifikované**. Její rozdíl oproti klasické harvardské je v tom, že obsahuje minimálně tři paralelně pracující aritmetické jednotky. Jedna slouží ke zpracování dat, druhá k výpočtu adres a třetí umístěná v řadiči k obsluze programového čítače a k obvodové podpoře programových cyklů.

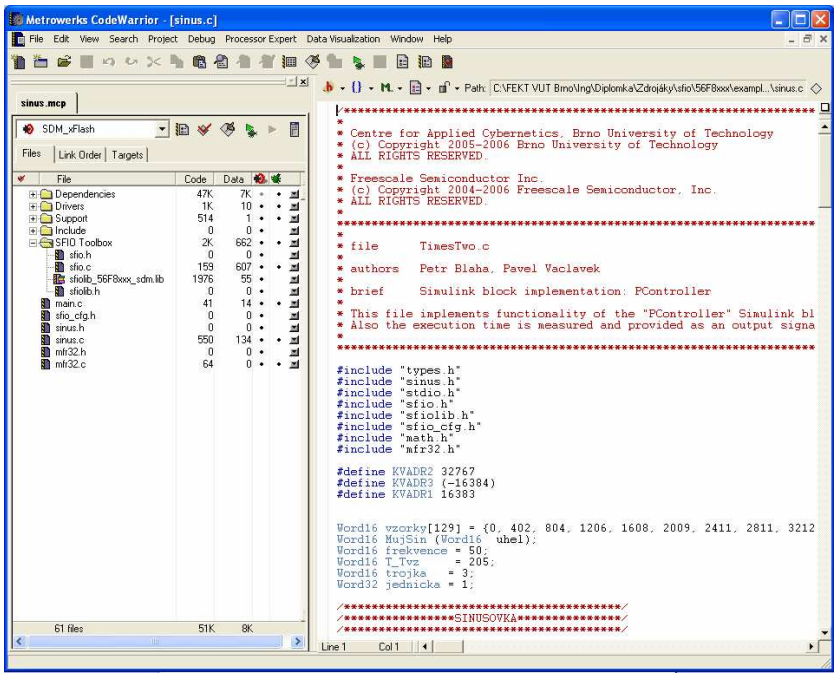
Parametry jádra procesoru:

- Výkonné jádro z rodiny 56800E založené na modifikované harvardské architektuře
- Výpočetní výkon je 60 MIPS při frekvenci jádra 60MHz
- 16 × 16-bit paralelní Multiplier-akumulátory (MAC)
- 4 × 36-bit akumulátory obsahující rozšiřující bity
- Paralelní instrukční sada se specifickými adresovými módy
- Tři interní adresové sběrnice
- Čtyři interní datové sběrnice
- Instrukční sada podporující obojí DSP a řídící funkce
- JTAG/OnCE rozhraní pro odladění programu
- Výkonný kompilátor pro jazyk C s podporou lokálních proměnných

4.2 PROSTŘEDÍ CODEWARRIOR IDE

Tvorbu programu při řešení konkrétní úlohy s daným typem procesoru lze provádět zhruba ve třech úrovních. Nejnižší způsob vyžaduje znalost tzv. strojového kódu (číselná forma instrukce), což značně znesnadňuje tvorbu a ladění programu. Využití jazyku symbolických adres představuje vyšší úroveň programování. Instrukce, adresy a data jsou zde reprezentovány abecedně číslicovými symboly, což zvyšuje přehlednost a srozumitelnost programu. Jazyk symbolických adres je pak do strojového kódu předkládán speciálním programem - *assemblerem*. Třetí nejvyšší a zřejmě nejpoužívanější úroveň programování DSP představují tzv. vyšší programovací jazyky - *Basic*, *Pascal*, *C*, *Fortran*, aj. Jeho přednosti jsou především v tom, že zdrojový kód jazyka C je mnohem čitelnější než assembler, je jednodušší ho zapsat a navíc je snáze přenositelný na jiné architektury. Vytvořený zdrojový text aplikačního programu je přeložen překladačem (kompilátorem) jazyka *C* do jazyka symbolických adres a z něj potom pomocí assembleru do strojového kódu, který už zpracovává samotný procesor.

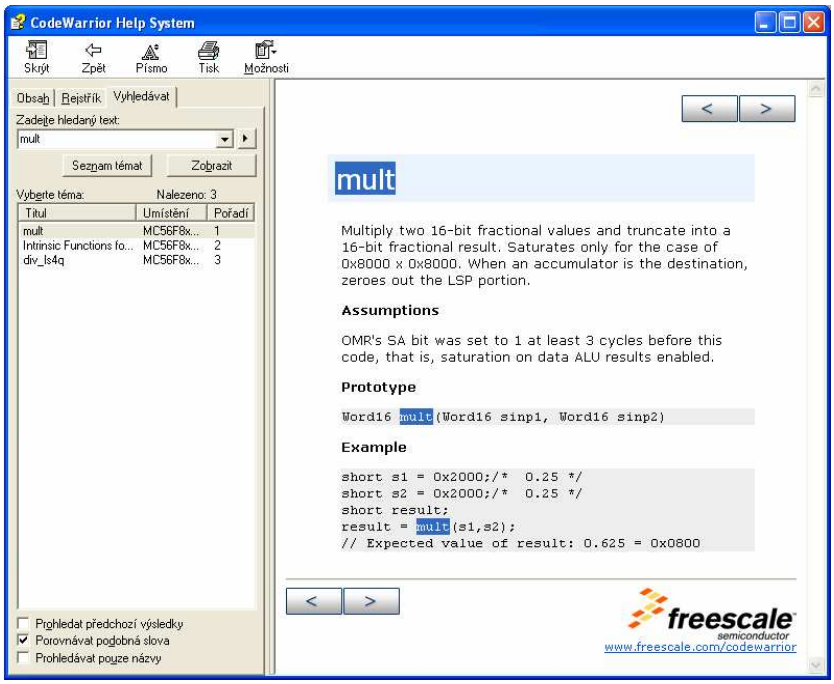
Programovací prostředí CodeWarrior IDE verze 5.6.1. 1658 pochází od společnosti Metrowerks, která se specializuje na vývoj programovacích prostředí pro profesionální tvorbu aplikací.



Obrázek 2: Pohled na uživatelské rozhraní programu CodeWarrior 5.6

Programovací prostředí obsahuje několik oken, z nichž mezi nejpoužívanější patří okno pro psaní samotného zdrojového kódu, dále také okno pro přehled použitých periferií a zařízení, připojených souborů (zdrojové soubory, hlavičkové soubory aj.). Ve své podstatě se jedná svým grafickým a uživatelským provedením o kompilátor a překladač zdrojového kódu podobný např. MS Visual C++ apod. Překlad kódu se děje po stisknutí klávesy F7 (Make) a následné spuštění programu se děje pomocí klávesy F5 (Debug). Samotný program je dále možné krokovat a vcelku pohodlně ladit.

Součástí prostředí je samozřejmě standardně Nápověda, která je při práci v prostředí nemalým pomocníkem. Obsahuje samozřejmě vyhledávač, do něhož lze zadat klíčová slova pro např. hledanou intrinsickou funkci či informace k datovému typu apod. Nápověda se v průběhu řešení stává výkonným pomocníkem.



Obrázek 3: Okno Nápověda pro CodeWarrior

4.3 SFIO TOOLBOX

SFIO znamená zkratku pro S-function input output. Hlavním úkolem tohoto toolboxu je pomoci uživateli při přenosu algoritmu ze schématu vytvořeném v Simulinku do DSP. Přenos je realizován blokovým způsobem krok po kroku což je zřejmě nejjednodušší řešení. Dalším rysem tohoto toolboxu je možnost mít přehled nad kontrolou fyzického systému běžícího v Simulinku pomocí algoritmu, který je naimplementován v reálném zařízení.

Jedná se o jakési přemostění mezi simulací běžící v prostředí MATLAB/Simulink a operacemi na reálném hardware, který je využit v průmyslové aplikaci. Navzdory existenci souboru bloků v Simulinku jako např. FIXED POINT BLOCKSET, REAL TIME WORKSHOP či EMBEDDED TARGETS poskytuje tento software řadu výhod, z nichž mezi nejdůležitější patří:

- levnost
- snadná ovladatelnost
- univerzalita

SFIO toolbox se skládá ze dvou částí. První část běží pod MATLABem. Podstatou je S-funkce, soubor *sfio.dll* a další doprovázející MATLAB funkce, které zjednodušují použití. Tento soubor realizuje komunikaci na straně PC. Druhá část toolboxu závisí na vybraném zařízení, do kterého je algoritmus ze Simulinku přenášen a který realizuje komunikaci na straně zařízení. [10]

4.4 IMPLEMENTACE IDENTIFIKAČNÍCH ALGORITMŮ

Jak bylo zmíněno na začátku práce a zejména v zadání, je třeba identifikační metody a spíše jejich algoritmy modifikovat a hlavně implementovat tak, aby jejich algoritmus mohl být zpracován na straně digitálního signálového procesoru.

Zásadní rozdíl spočívá v přidání některých bloků do struktury simulačních schémat, které mají přímou souvislost s algoritmem výpočtu běžícím na DSP. Jedná se např. o bloky *Tvarovač*, které v sobě skrývají tvarovač nultého řádu a převádí tudíž spojitý signál na diskrétní při pevně zvolené frekvenci vzorkování f_{vz} (přidržuje aktuální hodnotu po dobu T_{vz}). Vzhledem k tomu, že DSP pracuje diskrétně s $f_{vz} = 8kHz$, tj. s periodou vzorkování $T_{vz} = 125\mu s$, je tato frekvence rovna právě frekvenci na které vzorkuje DSP signály.

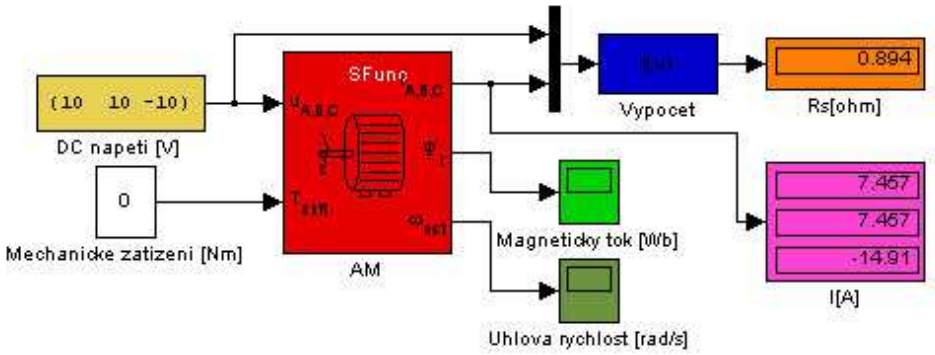
Dalšími novými bloky jsou bloky obsahující proces normalizace hodnot pro číslicové zpracování na DSP (*scale_u* a *scale_i*). Proces normalizace se děje v souvislosti s použitím fractional aritmetiky. Je třeba si nejprve určit maximální možnou hodnotu fyzikální veličiny, kterou může nabývat v průběhu řešení. Např. pro napětí je to hodnota 400V, vyjádřeno ve fractional aritmetice je to hodnota 32768 neboli 1. Podíl $\frac{400}{32768} = 0.0112$ vyjadřuje hodnotu scale hodnoty. Jedná se o hodnotu, kterou když se podělí např. hodnota napětí 10V, tak je výsledkem číslo vyjádřené ve fractional aritmetice. Hodnoty těchto scale parametrů pro napětí a proud jsou uložené v přiloženém m-file souboru pro Matlab. Ten je nutné před spuštěním simulace inicializovat jeho spuštěním. Význam fractional aritmetiky bude zmiňován u každé identifikační metody na konkrétním případě jejího použití.

Hlavním předpokladem pro implementaci je alespoň částečná kvalitní znalost jazyka C, která se ukáže být v průběhu řešení jako velmi užitečná či spíše přímo nutná. Důležitým pomocníkem byla po dobu řešení implementace identifikačních metod kvalitní literatura o základech algoritmizace v jazyce C. Implementace jednotlivých metod vychází svým základem i strukturou ze svých výchozích předpokladů.

5. DC TEST

Jak asi vyplývá z názvu této identifikační metody pro zisk určitého parametru AM, jedná se metodu využívající stejnosměrného proudu. Statorové vinutí vykazuje elektrický odpor R_s či impedanci Z_s v případě průchodu stejnosměrného DC či střídavého proudu AC tímto vinutím. Hlavní důvod, proč identifikovat velikost odporu statoru R_S, je zisk představy o fyzikálních vlastnostech daného materiálu, ze kterého je vinutí zhotoveno a především následná znalost skutečných hodnot nutná pro nastavení složek regulátorů apod. Závislost mezi změnou odporu statoru v závislosti na teplotě daného vinutí je možno považovat za téměř lineární či spíše přesněji za mírně exponenciálně rostoucí v čase *t*, jak je možné shlédnout v článku [3]. V případě tohoto testu se účinky teploty vinutí zanedbávají, neboť se samotný motor při testu netočí (napájí se DC napětím), oteplení vlivem např. mechanického tření v ložiscích motoru je tudíž nulové a je tedy možno s klidem samotné teplotní vlivy úplně zanedbat. Test by se však dal provést na zařízení, které by bylo po nějaké době provozu odstaveno a na jehož parametrech by se tudíž teplotní změny již uplatnily a daly konzultovat. Teplota se však stává výrazným rušivým činitelem při průběžné identifikaci jiných parametrů pomocí dalších metod, při kterých se rotor motoru otáčí (No-load test).

5.1 BLOKOVÉ SCHÉMA SIMULACE V SIMULINKU



Obrázek 4: Blokové schéma DC testu

Obrázek 4 vyjadřuje blokové uspořádání identifikace pomocí DC testu. Hlavním prvkem struktury je nepochybně model asynchronního motoru, který byl

vytvořen samotným vedoucím diplomové práce. Cílem tohoto obrázku je především nastínit blokovou strukturu samotné identifikační metody. Bloková struktura je vytvořena čistě v Simulinku a výpočet tedy běží na straně MATLABu. Vzhledem k tomu, že cílem práce je implementovat tuto metodu tak, aby byl její výpočetní algoritmus zpracován na DSP, proto vychází následná implementace právě z této struktury jak bude dokázáno později.

5.2 PRINCIP DC TESTU

Modelová struktura je tvořen souborem bloků (viz **Obrázek 4**), která spolu s blokem AM tvoří základ pro identifikační metodu DC test. Blok s názvem *DC napětí*, vyjadřující stejnosměrné napájecí napětí, je žlutý blok připojený na svorku motoru u_{abc} který je reprezentován vektorem $\bar{u} = [10;10;-10]$. Hodnoty jsou uvedeny v základních jednotkách vyjadřující elektrické napětí, tj. ve voltech. Vzhledem tedy k tomu, že model AM motoru je 3 fázový, je nutné ho i při tomto testu napájet stejnosměrným napětí na každé fázi motoru (3f stejnosměrným). Blok s názvem *Mechanické zatížení* reprezentuje mechanickou zátěž motoru M_Z, která je při tomto testu nulová, tj. motor je zcela v nezatíženém stavu a platí $M_Z = 0$ [Nm]. V bloku AM je možné kromě dalších parametrů nastavit hodnotu momentu setrvačnosti J, která je v případě simulace nastavena na nekonečnou hodnotu, což má za následek stav zařízení, že rotor nelze přes tak velkou hodnotu J jakkoliv roztočit. Modrý blok s názvem *Výpočet* v sobě zahrnuje vztah pro výpočet parametru odporu statoru R_S z veličin, které jsou v průběhu simulace k dispozici či je lze eventuelně na reálném pohonu měřit (proudy procházející každou fází). Výpočet je uveden v další kapitole. V blocích *Magnetický tok* a *Úhlová rychlost* je možné sledovat okamžité hodnoty těchto veličin v čase *t*.

5.2.1 Výpočet odporu statoru

Z parametrů získaných během simulace lze dalším výpočtem stanovit konečný výsledek v podobě odporu statoru R_S. Ze znalosti stejnosměrného napětí U_a, připojeného na statorové vinutí a z proudu I_a, který prochází tímto vinutím a je jakoby důsledkem připojeného napětí, je možno dle Ohmova zákona vypočítat parametr R_S. [5]

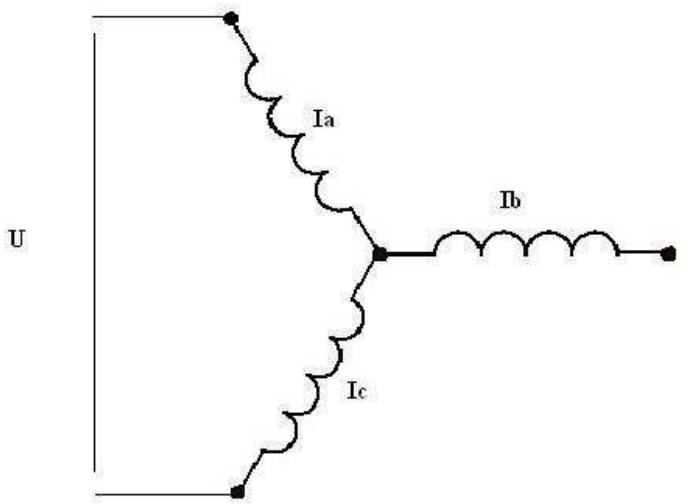
Součet hodnot statorových proudů I_a, I_b, I_c musí být roven nule. Tato podmínka platí všeobecně pro součet všech proudů připojených k uzlu v motoru:

$$I_a + I_b + I_c = 0 \tag{5.1}$$

kde v tomto případě po dosazení hodnot proudů tekoucích statorem přibližně platí

$$7.457 + 7.457 + (-14.91) = 0 \tag{5.2}$$

$$R_S = \frac{2 \cdot U_a}{3 \cdot I_a} \tag{5.3}$$



Obrázek 5: Statorové vinutí s připojeným napětím

Důležitým předpokladem pro výpočet je nutná znalost způsobu tečení proudů jednotlivými fázemi a nakonec samotný způsob zapojení (viz **Obrázek 5**).

V simulaci na příslušném modelu byla velikost napájecího napětí zvolena, jak je uvedeno (viz vektor $\bar{u}$) $U_a = 10V$ a proud tekoucí vinutím statoru byl při toku proudu vinutím fáze statoru o velikosti odporu R_S tím pádem roven hodnotě

$I_a = 7.457A$. Jednoduchým dosazením daných hodnot do vztahu (5.3) byla získána hodnota parametru R_S. Výsledky, které nám metoda poskytuje, nejsou závislé na velikosti napájecího napětí. Důvod je prostý, neboť napětí připojené na svorky statoru způsobí při uzavřeném obvodu vinutím elektrický proud, který je tedy jeho důsledkem.

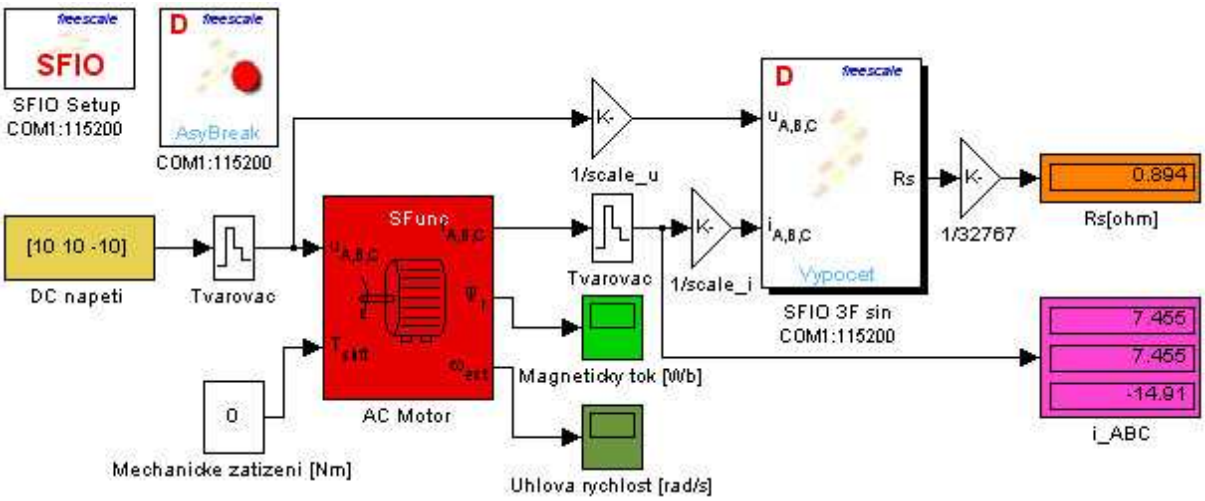
Dosazením hodnot do vztahu (5.3)

$$R_S = \frac{2 \cdot U_a}{3 \cdot I_a} = \frac{2 \cdot 10}{3 \cdot 7.457} = \underline{0.894\Omega} \qquad (5.4)$$

5.3 IMPLEMENTACE DC TESTU NA DSP

Samotná implementace DC testu na DSP vychází nepochybně ze simulace, která je zpracována v prostředí MATLAB/Simulink a popisuje ji **Obrázek 4**. Pro co nejlepší názornost a kvalitní přepracování simulace a zejména výpočtů při DC testu tak, aby výpočet běžel na straně DSP, je implementace realizována podobně jako simulace, jejíž výpočetní algoritmus je prováděn v MATLABu.

5.4 BLOKOVÉ SCHÉMA IMPLEMENTACE NA DSP



Obrázek 6: Blokové schéma implementace DC testu na DSP

Obrázek 6 obsahuje blokové schéma simulace DC testu, jehož algoritmus výpočtu běží na DSP. Struktura simulace je velmi podobná a vycházející ze simulace jejíž výpočet běží na straně MATLABu.

Jak již bylo řečeno, algoritmus implementace DC testu je naprogramován v jazyce C v prostředí CodeWarrior (**Obrázek 2**). Schéma obsahuje přidané funkční bloky, na kterých je vyobrazeno logo společnosti Freescale. Jedná se o bloky z SFIO toolboxu, které mají za úkol např. nastavovat parametry komunikace po sériové lince RS-232 (*SFIO*) a potom také neméně významné bloky, které v sobě nepřímo skrývají naprogramované části zdrojového kódu či spíše funkce výpočetního algoritmu dané identifikační metody (*Výpočet*).

Do schématu jsou také přidány bloky pro normalizaci fyzikálních veličin pro DSP (*1/scale_u* a *1/scale_i*) či proces normalizace výstupního parametru získaného identifikací na skutečnou hodnotu procesem dělení hodnotou 32767.

5.5 ALGORITMUS DC TESTU V JAZYCE C

Při psaní zdrojového kódu algoritmu pro výpočet odporu statoru, tj. výsledného parametru získaného identifikací bylo kvalitním a zdá se že i efektivním postupem vycházet ze struktury dle **Obrázek 4**. Samotný postup výpočtu odporu statoru R_S ze vstupních parametrů je podrobně rozebrán v Kap. 5.3.

Zdrojový kód, psaný jazykem C (soubor *dc_test.c*) v prostředí CodeWarrior, obsahuje na svém začátku deklaraci knihoven jako např. knihovny zajišťující komunikaci a konfiguraci takových parametrů mezi DSP a MATLABem (*sfio.h*, *sfio_cfg.h*), dále standardní knihovnu pro I/O operace (studio.h) či také knihovnu definující příslušné datové typy (*types.h*) ad. Převážnou část souboru *dc_test.c* tvoří deklarace funkce, která se vytvoří automaticky po označení *SFIO* bloku s názvem *Výpočet*, který má předem nadefinované a nastavené vstupní a výstupní porty dle principu výpočtu, a následném spuštění příkazu *codegen* napsaného přímo na příkazový řádek MATLABu. Následuje ihned výpis těla funkce *Výpočet*, který je možné vložit implicitně do příslušného zdrojového souboru. Jak lze vidět z Obr. 5.3, SFIO blok obsahuje dva vstupní a jeden výstupní port. Vstupní porty tvoří jednorozměrné pole se 3 hodnotami napětí, přivedeného na svorky AM a dále jednorozměrné pole se 3 hodnotami proudů, které z bloku AM vystupují. Ve funkci *Výpočet* jsou tato pole nadefinována jako pole obsahující hodnoty typu *short* uložené ve struktuře reprezentující vstupní veličiny daného bloku (*InStruct*). (Pozn.: Struktura je sdružený datový typ). Výstupní veličinou tohoto bloku je výsledný

parametr R_S uložený v datovém typu *short*, který je uložen ve struktuře (*OutStruct*) reprezentující daný výstupní port. Dále tělo funkce obsahuje část s výčtovým typem, který v sobě obsahuje konstanty, které lze použít všude tam, kde lze zadat celé číslo. Právě v tomto výčtovém typu jsou položky obsahující velikost vstupních, výstupních a stavových proměnných v jednotkách byte. (Např. vzhledem k faktu, že struktura pro výstupní veličinu obsahuje jedinou položku a to datový typ *short* mající rozsah typicky 16b, je proto v položce výčtového typu *StructSizes* pro výstupní veličinu uložena hodnota 2 takto LENGTHOF_OutStruct = 2,).

Tělo funkce dále obsahuje řádky kódu, které mají např. za úkol inicializovat tuto funkci při spuštění simulace v Simulinku apod. Mimo to však tělo této funkce obsahuje neméně důležitou část a to syntaxi samotného výpočtu výsledného parametru ze vstupních veličin. Tu tvoří jeden řádek kódu.

```
pOut->Rs = div_ls(L_mult(pIn->u_ABC[0], dvetretiny), pIn->i_ABC[0])
```

Tabulka 1: Výpočet odporu statoru

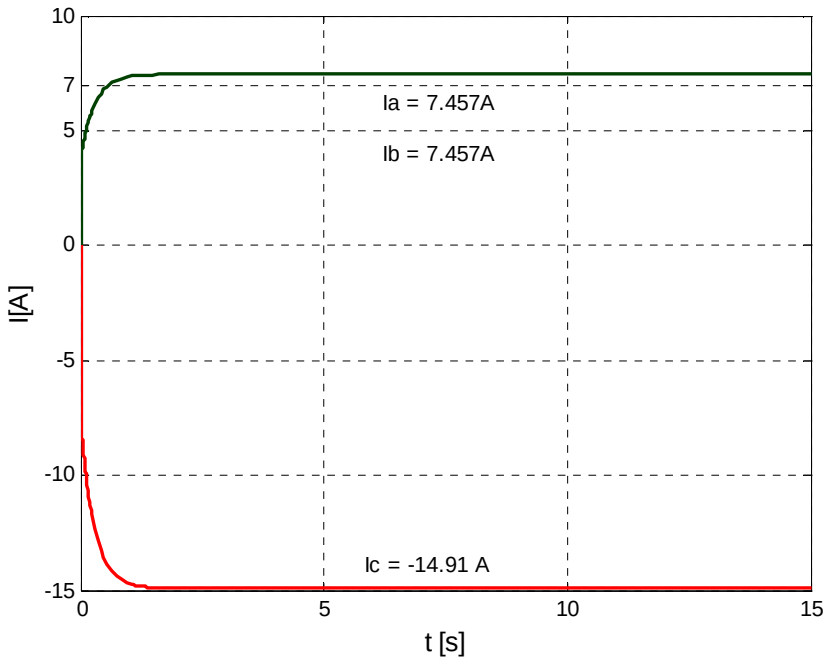
Tabulka 1 obsahuje finální podobu výpočetního algoritmu pro určení odporu statoru R_S. Ten vychází ze vztahu 5.3 tak, že je výpočet rozdělen na několik kroků. Nejprve se provede výpočet součinu dvou 16b čísel (*dvetretiny* a *pIn->u_ABC[0]*) pomocí intrinsické funkce *L_mult*, která vrací výsledek operace v rozsahu 32b čísla. Veličina *dvetretiny* je lokální proměnnou funkce *Výpočet* s hodnotou 21845 (*Frac16* má rozsah -32768 až 32767), která ve FRACTIONAL aritmetice (rozsah hodnot -1 až +1) reprezentuje právě hodnotu podílu $\frac{2}{3}$. Poté zbývá ještě provést matematickou operaci dělení 32b čísla, získaného z předchozího kroku, číslem 16b (*pIn->i_ABC[0]*) pomocí další z intrinsických funkcí *div_ls*. Je na místě připomenout, že hodnoty napětí tak i proudů potřebné pro výpočet jsou normalizovány pro výpočet na DSP (viz. Kap. 4.4). Po těchto krocích je výpočet kompletní, tj. proměnná *pOut->Rs* obsahuje již hodnotu, která se sice v průběhu identifikace mění, ale konverguje však ke skutečné hodnotě identifikovaného parametru AM, tj. k hodnotě $R_S = 0.894\Omega$.

5.6 ZHODNOCENÍ IMPLEMENTACE DC TESTU NA DSP

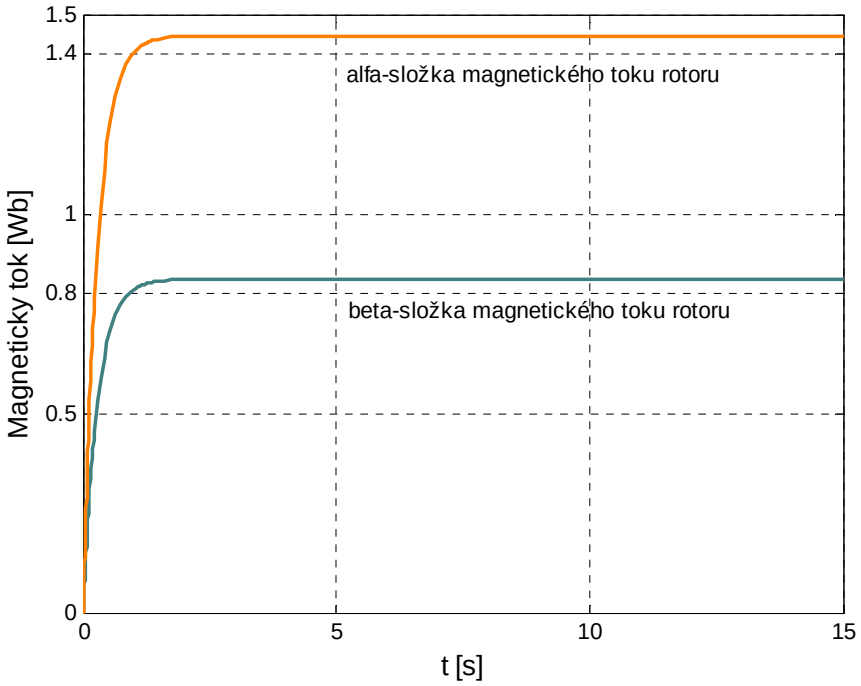
Rozdíl mezi hodnotou získanou jen simulací v MATLABu a hodnotou získanou výpočtem na DSP prakticky není. Za zásadní odlišnost lze považovat především takovou možnost, že by bylo možné po určité úpravě aplikovat implementaci identifikační metody na reálný AM. Oproti běžné simulaci (**Obrázek 4**). kdy se jedná výhradně o výpočet v MATLABu, se identifikační metody díky implementaci na DSP posouvají kupředu vstříc aplikaci už ne v rámci simulace ale už na reálné zařízení a bez použití simulačních prostředků. Realizace by byla zřejmě založena na digitálním měření střídavých proudů a napětí, které jsou vstupními veličinami pro výslednou identifikaci. Problém diskretizace spojité fyzikální veličiny je v simulaci řešen pomocí přidržovače nultého řádu (viz. Kap. 4.4). V následující kapitole jsou vykresleny okamžité hodnoty některých veličin v čase *t*.

5.7 PRŮBĚHY OKAMŽITÝCH HODNOT VELIČIN V ČASE

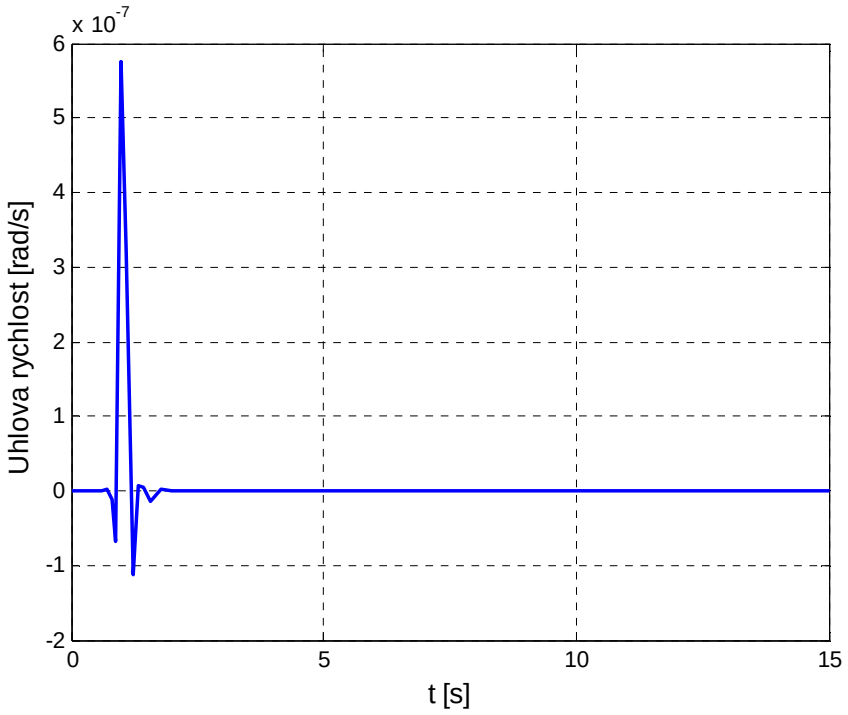
Následující obrázky názorně poukazují na typické průběhy přechodových charakteristik pro okamžité hodnoty proudů statorem (**Obrázek 7**) či pro obě složky magnetického toku rotoru (**Obrázek 8**).



Obrázek 7: Okamžitý průběh proudů statorem



Obrázek 8: Okamžitý průběh složek toku rotorem

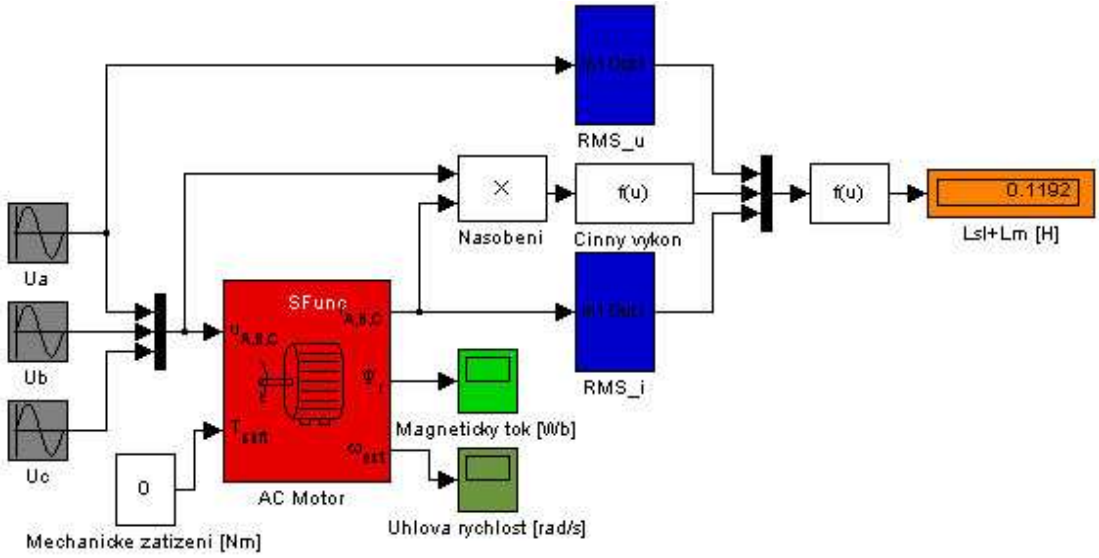


Obrázek 9: Okamžitý průběh úhlové rychlosti

6. NO LOAD TEST

Podstatou tohoto testu prováděném obecně na AM je určení ztrát v točivém magnetickém poli samotného motoru a zejména identifikace některého z jeho parametrů. Jak z názvu vyplývá, test je prováděn na nezatíženém motoru (no load = bez zatížení). Nezatížený motor je však dosti zavádějícím pojmem, neboť tento stav prakticky neexistuje a není možné ho jakýmkoliv způsobem navodit. Je to proto, že v praxi není možné zanedbat vůči testu účinky valivého tření v ložiscích motoru či případně odpor vzduchu ventilátoru potřebného ke chlazení samotného pohonu, pokud jím je tedy vybaven. Při tomto testu se předpokládá stav, kdy jsou mechanické ztráty nulové a tudíž neovlivňují samotnou identifikaci ($M_z = 0$). Výsledkem testu jsou velikosti parametrů rozptylové indukčnosti statorového vinutí L_{sl} a magnetizační indukčnosti L_m. [5]

6.1 BLOKOVÉ SCHÉMA SIMULACE V SIMULINKU



Obrázek 10: Blokové schéma simulace No-load testu

6.2 PRINCIP SIMULACE

Princip simulace vychází z blokového modelu uvedeného na **Obrázku 10**. Základem je, stejně jako u DC testu, blok modelu asynchronního motoru. Schéma obsahuje bloky, které reprezentují napájení, dále bloky pro analýzu a zpracování jednotlivých signálů. V této metodě se využívá 3f napájení statorového vinutí. V tomto modelu je napájecím prvkem soustava třech bloků (označeny šedou barvou), která jsou přivedena na motor. Tento zdroj je tvořen třemi sinusovými průběhy, které reprezentují jednotlivá fázová napětí a mají své amplitudy rovny 220V a své fáze vzájemně posunuty o 120° (viz. vztah 2.1). Moment zatížení motoru symbolizuje blok *Mechanické zatížení* ($M_z = 0$). Měřitelné veličiny na výstupu z motoru jsou veličiny jako magnetický tok rotoru Ψ_r, proud I a úhlová rychlost ω. Tyto veličiny symbolizují bloky *Magnetický tok a Úhlová rychlost*. Jak je vidět z **Obrázku 10**, je hodnota proudu každé fáze motoru dále využita společně s hodnotou napětí každé fáze k výpočtu činného výkonu, jehož konečný výpočet je uveden v bloku *Činný výkon*. Jedná se o celkový výkon motoru na všech jeho fázích. Signál z tohoto bloku je dále veden do multiplexeru, kde se schází tento signál spolu se signálem z bloku *RMS*, který počítá efektivní hodnotu U_{ef} z fázového napětí U_a a se signálem z dalšího bloku *RMS*, který vypočítá efektivní hodnotu I_{ef} z proudu I_a. Pomocí dalšího bloku, který symbolizuje matematické operace, potřebné ke stanovení výsledku identifikace, je následně vypočten parametr $L_{sl} + L_m$ této identifikační metody. Veškeré matematické operace jsou uvedeny v následujících odstavcích.

6.3 URČENÍ PARAMETRU $L_{SL} + L_M$

Určení parametrů $L_{sl} + L_m$ v sobě skrývá několik fyzikálních skutečností a také matematických výpočtů z nich vyplývajících. Samotné výpočty a kroky jsou ukryty v modelu simulace (viz. **Obrázek 10**). V následujícím bude zmíněn výčet a detailní popis těchto operací, které vedou ke stanovení výsledných parametrů AM, které z No-load testu vyplývají.

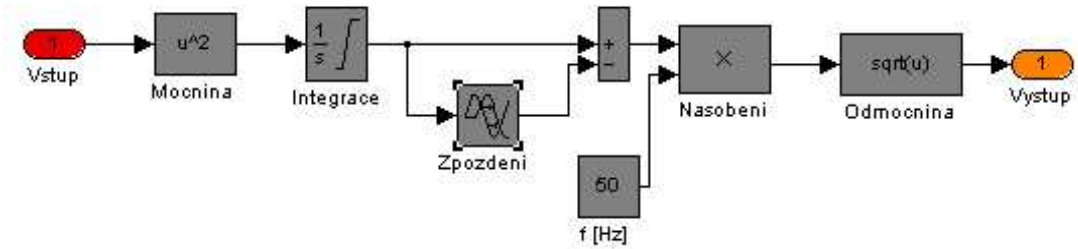
6.3.1 Blok RMS

Blok RMS (anglicky Root Mean Square), obsažený v modelu No-load testu, v sobě skrývá výpočet efektivní hodnoty veličiny (v tomto a dalších testech např. hodnoty proudu I_{ef} a efektivní hodnoty napětí U_{ef}).

Pro obecný matematický vztah výpočtu efektivní hodnoty veličiny platí:

$$X_{ef} = \sqrt{\frac{1}{T} \cdot \int_{t_0}^{t_0+T} x^2(t) \cdot dt} \qquad (6.1)$$

kde X_{ef} je efektivní hodnota dané veličiny, T je perioda signálu, t_0 je počáteční doba, $x^2(t)$ je druhá mocnina vstupní veličiny.



Obrázek 11: Blokové schéma bloku RMS

Princip získání efektivních hodnoty veličin, ať už proudu či napětí, spočívá ve výpočtu druhé odmocniny z integrace mocniny vstupní hodnoty přes jednu periodu signálu *T*. Na **Obrázku 11** je uvedeno blokové uspořádání jednotlivých bloků, které vyplývá přímo ze vztahu (6.1). Tento blok je sice součástí určitého toolboxu Simulinku, který však není k dispozici. Proto bylo nutné si tento blok vytvořit z běžných prostředků, které Simulink poskytuje. Tento blok je nedílnou součástí matematického výpočtu dalších metod k určení parametrů AM, které jsou náplní diplomové práce (Locked rotor test, One phase test).

6.3.2 Princip identifikace

Tato metoda vychází z několika podmínek

$$X_{sl} + X_m \approx \frac{U_{ef}}{I_{ef}} \qquad (6.2)$$

či

$$X_{sl} + X_m = \frac{Q_a}{I_{ef}^2} \tag{6.3}$$

kde symboly $X_{sl} + X_m$ značí součet rozptylové reaktance statoru a magnetizační reaktance, $\frac{U_{ef}}{I_{ef}}$ je podíl efektivní hodnoty napětí a efektivní hodnoty proudu jedné fáze a podíl $\frac{Q_a}{I_{ef}^2}$ je tvořen jalovým výkonem na jedné fázi a mocninou efektivní hodnoty proudu touto fází.

To znamená, že pro nalezení součtu reaktancí (viz. 6.2) a (viz. 6.3), je možné použít dvou přístupů, ale způsob využívající vztah (6.2) je poněkud nepřesný. Nepřesnost vychází z následujícího. V případě No-load testu teče převážná část okamžitého proudu statoru i_s sériovým zapojením odporu statoru R_s a jeho rozptylové indukčnosti L_{sl} spolu s magnetizační indukčností L_m. To znamená, že podle vztahu (6.2) se zanedbává úbytek napětí na odporu R_s. Pro malé hodnoty odporu statoru R_s může být tento přístup možno považovat v jakém si ohledu za kvalitní, ale pro větší hodnoty tohoto odporu už by výsledek daný tímto přístupem mohl být velmi nepřesný a tudíž nepoužitelný. Proto je lepší zvolit cestu, kterou určuje vztah (6.3) a je aplikovatelný na jakýkoliv případ, tj. určit jalový výkon a následně ho podělit mocninou efektivní hodnoty proudu.

6.3.3 Výpočet parametrů $L_{sl} + L_m$

Jak je uvedeno ve vztahu (6.3), je možné určit parametry $L_{sl} + L_m$, resp. $X_{sl} + X_m$ pomocí znalosti jalového výkonu Q_a a efektivní hodnoty proudu I_{ef}. K tomu je však mimo to nutné znát velikost účiníku $\cos\varphi$, který je možno určit z činného výkonu P. Výpočet účiníku udává následující vztah

$$\cos\varphi = \frac{P}{3 \cdot U_{ef} I_{ef}} \tag{6.4}$$

kde P je celkový činný výkon motoru, U_{ef} a I_{ef} jsou efektivní hodnoty fázového napětí a proudu.

Pro výpočet jalového výkonu na jedné fázi motoru platí následující vztah

$$Q_a = U_{ef} I_{ef} \sin\varphi \tag{6.5}$$

kde Q_a je jalový výkon a U_{ef} a I_{ef} jsou efektivní hodnoty fázového napětí a proudu. V rovnici (6.3) se parametry určují jen na jedné fázi, proto je třeba počítat jen s jalovým výkonem jedné fáze.

Z rovnic (6.4) a (6.5) je možno matematickou operací získat rovnici (6.3), přesněji získat podíl $\frac{Q_a}{I_{ef}^2}$, který je přímo roven součtu rozptylové reaktance statoru a magnetizační reaktance.

Následující rovnice obsahují matematickou interpretaci výše uvedených tvrzení vycházející z rovnic (6.4) a (6.5) a odvozují vztah pro výpočet $L_{sl} + L_m$.

$$\sin^2\varphi + \cos^2\varphi = 1 \tag{6.6}$$

$$\sin\varphi = \sqrt{1 - \left(\frac{P}{3 \cdot U_{ef} I_{ef}}\right)^2} \tag{6.7}$$

Rovnice (5.5) obsahuje člen $\sin\varphi$, který je po osamostatnění vložen do levé strany rovnice (5.7)

$$\frac{Q_a}{U_{ef} I_{ef}} = \sqrt{1 - \left(\frac{P}{3 \cdot U_{ef} I_{ef}}\right)^2} \tag{6.8}$$

V tomto vztahu je na levé straně nutné osamostatnit levou část na podíl $\frac{Q_a}{I_{ef}^2}$, tj. je nutné obě strany této rovnice vynásobit podílem $\frac{U_{ef}}{I_{ef}}$. Výsledkem je vztah

$$\frac{Q_a}{I_{ef}^2} = \sqrt{1 - \left(\frac{P}{3 \cdot U_{ef} I_{ef}}\right)^2} \cdot \frac{U_{ef}}{I_{ef}} \tag{6.9}$$

Tento vztah udává součet rozptylové reaktance statoru a magnetizační reaktance. No-load test však identifikuje parametry $L_{sl} + L_m$, proto je tedy nutné přepočítat reaktance na indukčnosti dle následujícího vztahu

$$L_{sl} + L_m = \frac{\frac{Q_a}{I_{ef}^2}}{2 \cdot \pi \cdot f} \tag{6.10}$$

Dosazením konkrétních hodnot veličin, které obsahuje vztah (6.10) by byly vypočítány parametry, které v simulaci vyšly rovny hodnotě $L_{sl} + L_m = 0.1192H$.

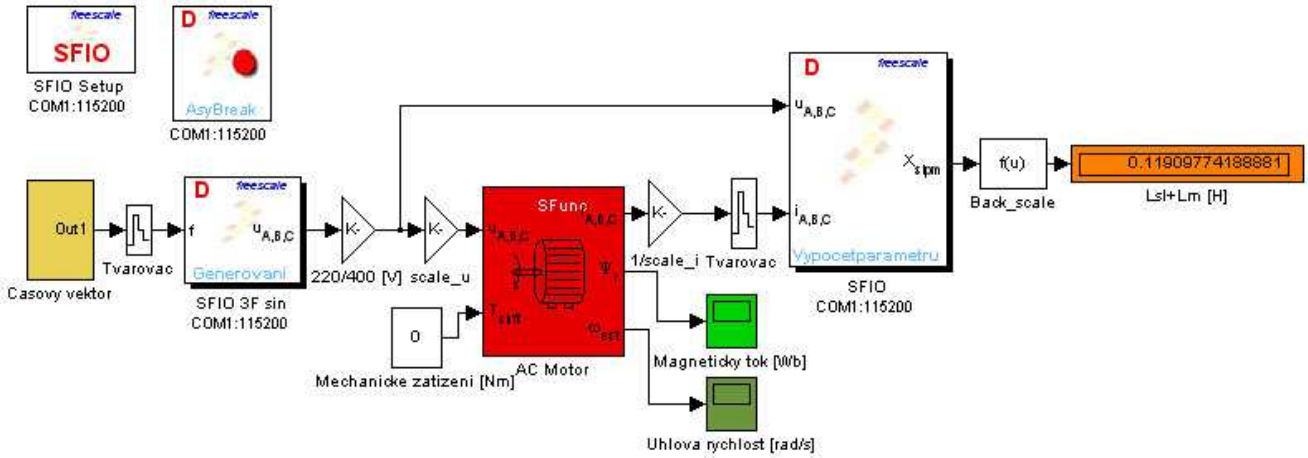
Tyto matematické operace jsou ve schématu simulace reprezentovány blokem *f(u)*, který je vytvořen jako funkce obsahující vztahy (6.9) a (6.10).

6.4 IMPLEMENTACE NO-LOAD TESTU NA DSP

Postup při realizaci implementace výpočetního algoritmu do DSP opět vychází svým pojetím ze simulace běžící výhradně v MATLABu jako u DC testu a stejně tak i u dalších identifikačních metod uvedenými v této práci. Opět bylo dobré pro názornost a přehlednost vycházet z původního blokového uspořádání popisujícího No-load test (**Obrázek 10**) a implementaci provést co nejnázorněji vzhledem k tomuto přístupu. Podstata implementace tedy spočívá v naprogramování zdrojového kódu pro DSP tak, aby byl výpočetní algoritmus No-load testu zpracován právě na DSP.

Zdrojový kód, psaný jazykem C (soubor *no_load.c*) v prostředí CodeWarrior, obsahuje na svém začátku standardně deklarace knihoven jako např. knihovny zajišťující komunikaci a konfiguraci parametrů mezi DSP a MATLABem (*sfio.h*, *sfio_cfg.h*), dále standardní knihovnu pro I/O operace (studio.h) či také knihovnu definující příslušné datové typy (*types.h*) ad. Převážnou část souboru *no_load.c* tvoří deklarace funkcí skrývající podstatu jednotlivých bloků, které se vytvoří automaticky po označení *SFIO* bloků s názvy *Generování* či *Výpočet parametrů*, které má předem nadefinované a nastavené vstupní a výstupní porty dle principu výpočtů (počet portů ad.), a následném spuštění příkazu *codegen* napsaného přímo na příkazový řádek MATLABu. Vygenerované zdrojové kódy jsou poté vloženy do zdrojového souboru *no_load.c*, kde jsou do jejich těla vepisovány potřebné příkazy či vlastní funkce dle konkrétního případu použití. Součástí těchto funkcí jsou struktury, tj. sdružené datové typy jazyka C, obsahující položky pro datové typy vstupních, výstupních a stavových proměnných.

6.5 BLOKOVÉ SCHÉMA IMPLEMENTACE NA DSP



Obrázek 12: Blokové schéma implementace No-load testu na DSP

Obrázek 12 obsahuje blokové schéma simulace No-load testu, jehož výpočetní algoritmus je prováděn na DSP. Struktura simulace je velmi podobná a vycházející ze simulace, jejíž výpočet běží na straně MATLABu.

Jak již bylo řečeno, algoritmus implementace No-load testu je opět naprogramován v jazyce C v prostředí CodeWarrior (**Obrázek 2**). Simulační schéma obsahuje některé přidané funkční bloky nesoucí loga společnosti Freescale. Jedná se o identické bloky jako v případě DC testu z SFIO toolboxu, které mají za úkol např. nastavovat parametry komunikace po sériové lince RS-232 (*SFIO*) a potom také neméně významné bloky a nejdůležitější, které v sobě nepřímo skrývají naprogramované části zdrojového kódu či spíše funkce výpočetního algoritmu dané identifikační metody (*Výpočet parametrů* a *Generování*). Do schématu jsou rovněž také přidány bloky pro normalizaci fyzikálních veličin pro algoritmus na DSP a případný zpětný přepočet *1/scale_i* a *Back_scale*).

6.6 ALGORITMUS NO-LOAD TESTU V JAZYCE C

Podstatu podobnosti výše uvedených blokových schémat (**Obrázek 10** a **Obrázek 12**) je možné poznat vzájemným porovnáním. Hlavní myšlenkou implementace celé simulace je to, že takřka celý výpočetní algoritmus běží na straně DSP. Jedná se zejména o generování napájecího napětí, dále výpočet efektivních hodnot napájecího napětí a výstupního proudu s následným konečným výpočtem identifikovaných parametrů *Lsl+Lm*. Podstat implementace tedy závisí na vytvoření

výpočetního algoritmu, který bude tyto výpočty řešit. Konkrétní postup výpočtů, tedy popis funkcí jednotlivých bloků je popsán jednotlivě v dalších kapitolách. Celá implementace je součástí zdrojového souboru s názvem *no-load.c*.

6.6.1 Generování napájecího napětí

Samotný zdrojový kód psaný jazykem C v prostředí CodeWarrior, který je podstatou funkce bloku *Generování*, v sobě skrývá vytvoření napájecího napětí, který jak je obecně známo tvoří sinusový periodický signál. Hlavním aspektem před návrhem postupu naprogramování sinusového průběhu, jehož jednotlivé vzorky budou pevně uloženy v paměti DSP procesoru, je právě využití paměťového prostoru. Bylo tedy nutné navrhnout takové řešení, které bude co nejméně náročné na obsazení fyzické paměti na vývojové desce. Jako dobré a optimální řešení při programování se jeví nejprve vygenerovat ¼ periody sinusového signálu (tj. 1 kvadrant) na základě procesu interpolace mezi hodnotami pevně uloženými v jednorozměrném poli, s pomocí níž lze jednoduchými matematickými operacemi dotvořit celou periodu pokrývající 4 kvadranty původní ¼ periody. (Pozn.: Hodnoty uložené v jednorozměrném poli s názvem *Vzorky* byly vygenerovány pomocí skriptu v MATLABu, který vypočítá 128 hodnot z ¼ periody sinusového signálu, jehož amplituda je 32767 s tím, že poslední vzorek je 16384-tý).

Zdrojový kód tvoří funkce *Sinus*, jejíž argument tvoří hodnota úhlu, která je generována žlutým blokem z **Obrázek 12** vystupujícím pod názvem *Časový vektor* a vstupující do SFIO bloku *Generování*. Jím generovaný časový vektor pro jednu periodu vzniká na základě generátoru hodinového signálu, který je přímo úměrný frekvenci f. Ta je pro případ simulace nastavena na hodnotu 50Hz.

Pro samotné generování sinusového signálu se využívá algoritmus interpolace hodnot, které jsou uloženy pevně v jednorozměrném poli. Jednorozměrné pole je jak už bylo zmíněno zaplněno 128 vzorky, na které se poté aplikuje interpolační algoritmus, který je schopen „dopočítat“ zbývající vzorky mezi těmi pevně uloženými v poli. Tj. po úspěšně provedené interpolaci tvoří čtvrtperiodu sinusového signálu $128 \times 128 = 16384$ vzorků. Z toho vyplývá, že celkovou periodu sinusového signálu tvoří $4 \times 16384 = 65536$ vzorků. Vzhledem k tomu, že argument funkce je 16-bitovou proměnnou typu Word16, jejíž rozsah je -32768 až 32767, dochází při inkrementaci úhlu až k hodnotě 32767 k přetečení, kterého lze velmi efektně využít

právě při generování úhlu vstupujícího do funkce jejíž prototyp je Word16 Sinus(Word16 uhel). Výstupem této funkce je hodnota vzorku sinusového signálu na základě vstupního úhlu.

Samotné napájecí napětí je generováno SFIO blokem pod názvem Generování a je tvořeno 3-fázovým napětím, tj. 3-mi sinusovými průběhy vzájemně posunutými o úhel 120°. Tato část je obsažena právě ve funkci reprezentující blok *Generování*. Jedná se o 3- násobné volání funkce, které mají počáteční hodnotu úhlu vzájemně od sebe posunutou o 120°. Samotná funkce vznikne takřka automaticky vygenerováním na základě příkazu *codegen* v MATLABu, který se aktivuje spolu s označením příslušného SFIO bloku, v tomto případě blok *Generování*. Funkce poté obsahuje informace o tomto bloku (vstupně/výstupní veličiny, jejich datové typy atd.). Samotnou část, která nese volání vlastní funkce *Sinus* či další příkazy, bylo třeba doprogramovat vlastními silami a vložit do části funkce *Generování* k tomu určené.

Před připojením napětí na svorky motoru je nutno toto napětí normalizovat (viz. Kap. 4.4) z maximálního rozsahu 32767, čemuž odpovídá hodnota 400V ve FRACTIONAL aritmetice na hodnotu 220V (fractional přibližně 18022), která tvoří zvolenou amplitudu napájecího napětí. Poté je nutno tuto hodnotu, vyjádřenou nadále v 16-bitovém formátu Word16 převést za pomocí bloku *scale_u* na hodnotu použitelnou pro samotné připojení na blok motoru, tj. provést převod z FRACTIONAL vyjádření na vyjádření v dekadickém vyjádření skutečné hodnoty napájecího napětí. Stejně tak je poté nutné normalizovat proud vystupující z bloku AM zpět na hodnoty použitelné pro následné matematické operace běžící opět na straně DSP, tj. podělit výstupní proud hodnotou uvedenou v bloku *scale_i*. Hodnoty *scale_u* a *scale_i* jsou pevně uloženy jako konstanty v m-file souboru, který je nutné vždy před simulací spustit, aby byly tyto hodnoty akceptovány pro výpočty, kde se právě ony konstanty volají.

6.6.2 Algoritmus výpočtu efektivní hodnoty RMS ze signálu na DSP

Samotný algoritmus výpočtu efektivní hodnoty je součástí zdrojového kódu psaného v jazyce C. Přesněji řečeno, je součástí funkce, která vznikla na základě vytvoření SFIO bloku s názvem *Výpočet parametrů*, který je součástí blokové struktury na **Obrázek 12**.

Vytvoření algoritmu pro výpočet efektivní hodnoty z napájecího napětí či statorového proudu AM předchází úvaha nad samotným výpočetním vztahem (6.1) či blokovým schématem vytvořeným na základě předchozího vztahu. Obecně vzato, se jako zřejmě největší problém při řešení ukázal proces vytvoření algoritmu integrace signálu přes jednu jeho celou periodu *T*. V číslicových operacích je možno operaci integrace signálu na přesně daném úseku chápat jako sumaci každého vzorku z daného signálu. Výsledkem výpočtu určitého integrálu z libovolné matematické funkce jedné proměnné je poté velikost obsahu plochy pod křivkou, která tvoří ohraničení spolu se stanovenými mezemi pro oblast integrování.

Integrace tedy představuje sumaci jednotlivých vzorků přes jednu periodu signálu. Jedna perioda obsahuje 160 vzorků. Tato hodnota vychází z frekvence vzorkování $f_{vz} = 8kHz$, což při frekvenci napájecího napětí 50Hz znamená právě 160 vzorků na periodu při dané frekvenci napájecího napětí. Proto dochází při běhu kódu k odpočítání konce periody pomocí proměnné *index*, která se inkrementuje o hodnotu 1 každým cyklem a až po neinkrementování na hodnotu 160 dojde k výpočtu odmocniny pomocí funkce jejíž prototyp má tvar Word16 = mfr32Sqrt(Word32). Při integraci vzorků dochází k tomu, že mocninou se každý vzorek změní z typu short na datový typ long a ten se sumuje s ostatními po dobu celé periody. Hodnoty signálu tvoří celočíselná 16-bitová čísla short, proto bylo možné využít intrinsické funkce pro násobení těchto dvou 16-bitových čísel *L_mult*, jejímž výsledkem je 32-bitové číslo long. Rozsah stanovený pro long je -2 147 483 647 až 2 147 483 647. Při sumaci k přetečení nedochází, tudíž je tento typ schopný pojmout hodnotu integrálu celé periody signálu. Dále je ještě hodnota integrálu celé periody násobena hodnotou 205. Poté je hodnota předána jako argument funkci s prototypem *Word16 = mfr32Sqrt(Word32)*, která provede výpočet odmocniny. Podstata výpočtu je stejná i pro výpočet efektivní hodnoty proudu. Operace výpočtu efektivní hodnoty napětí U_{ef} a efektivní hodnoty proudu I_{ef} budou využity v následující kapitole.

6.6.3 Algoritmus výpočtu parametrů $L_{sl} + L_m$ na DSP

Algoritmus výpočtu parametrů $L_{sl} + L_m$ pomocí implementace No-load testu běžícího na straně DSP je podstatou bloku SFIO pod názvem *Výpočet parametrů* a

jeho funkce se stejným názvem, která je součástí souboru *no_load.c*. Jedná se výpočetní blok mající dva vstupní porty a jeden výstup. Prvním vstupem jsou 3 vektory okamžitých hodnot napájecích napětí *u_ABC[3]*, které jsou výstupem z SFIO bloku *Generování*, podobně jako v případě (**Obrázek 10**). Druhým vstupem stejného bloku jsou rovněž 3 vektory, ale okamžitých hodnot výstupních proudů *i_ABC[3]* z AM. Výstupní port je vyhrazen pro hodnotu výsledné proměnné *X*, symbolizující součet rozptylové induktance statoru a magnetizační induktance $X_{sl} + X_m$.

Jak vyplývá z Kap. 6.3.3, kde je uveden podrobný popis principu celého výpočtu pro získání parametrů $L_{sl} + L_m$, z něhož jsou finální záležitostí vztahy 6.9 a 6.10, bylo nutné tedy naprogramovat některé kroky vedoucí k zisku jednotlivých členů daných vztahů spolu s matematickými operacemi, které tyto vztahy skrývají a na kterých je daný vztah založen. Podstata opět vychází ze simulace běžící výhradně na straně MATLABu (**Obrázek 12**). V Kap. 6.3.2 je uveden princip algoritmu pro výpočet efektivní hodnoty jak napětí U_{ef} tak i proudu I_{ef} využívající integrace daného signálu přes jednu jeho periodu *T*.

V dalším je nutné stanovit činný výkon *P*. Jak je uvedeno v blokovém schématu dle **Obrázek 10**, je činný výkon počítán jako součet součinů okamžitých hodnot napětí a proudů pro každou fázi. Pro výpočet tohoto výkonu je využito procesu integrace podobně jako v případě výpočtu efektivní hodnoty pro proud a napětí. V tomto případě se však jedná o sumaci jednotlivých vzorků vypočteného činného výkonu, tj. součinu okamžitého napětí s okamžitým proudem přes jednu periodu T. Tento proces vnáší do řešení určité zpřesnění výsledku, neboť se jedná jakoby o zprůměrňování celkové hodnoty činného výkonu přes 160 jeho vypočtených vzorků. Tj. opět je využita proměnná *pState->index*, která symbolizuje počet aktuálních zpracovávaných vzorků. K tomuto účelu je v algoritmu vytvořena jednoduchá podmínka *if(pState->index > 159)*, která pokud je splněna tak spustí další dílčí výpočty včetně konečného výpočtu činného výkonu uvedeného ve stavové proměnné *pState->P*, dále také spustí výpočet odmocnění pro případ stanovení efektivních hodnot pro napájecí napětí a výstupní proud a poté hlavně konečný výpočet ze zmiňovaných veličin vedoucí k zisku výsledných parametrů

AM. Konečný výpočet s sebou na první pohled nepřináší žádné problémy při jeho přepisu do jazyka C. Využívá se opět intrinsických funkcí jako např. *L_mult, div_ls, mult*_r a dalších. Jedná se o optimalizované funkce, které jsou ideálním řešením při programování základních matematických operací na DSP. Skutečností však jsou některé zvláštnosti, které komplikují řešení a o kterých je nutné se tedy nepochybně zmínit.

Jako hlavní problém se jeví skutečnost, že efektivní hodnota napětí ve stavové proměnné *pState->u_rms* nabývá mnohem větší hodnoty než-li efektivní hodnota proudu *pState->i_rms* (konkrétní hodnoty ze simulace jsou přibližně $U_{ef} = 155.6V$ a $I_{ef} = 4.153\text{A}$). Problém to nese v případě operace podílu těchto dvou hodnot (viz. 6.9), kdy je nutné ošetřit vzniklou saturaci. Podmínku, zda je $U_{ef} > I_{ef}$ řeší *if(pState->i_rms < pState->u_rms).* Daný problém se řeší takovým způsobem, že se provádí operace bitových posunů u proměnných tak, aby byla splněna podmínka $I_{ef} > U_{ef}$. Skutečnou hodnotu obsahující počet míst, o kolik se má dané číslo bitově posunout, řeší následující zápis

```
num_shifts = norm_s(pState->i_rms) - norm_s(pState->u_rms);
```

Intrinsická funkce *norm*_s určí kladnou hodnotu, o kolik míst se může daná hodnota 16-bitového čísla *short* posunout vlevo tak, aby byl výsledek této operace pořád 16-bitového rozsahu, tj. nedošlo k přetečení platného rozsahu. *(Pozn.: Operace bitových posunů vlevo a vpravo symbolizují matematické operace násobení a dělení hodnotou 2).* [8] Výše uvedený zápis obsahuje výpočet, z kterého je možné usoudit, že počet posunutí hodnoty v proměnné *pState->i_rms* je vyšší, než u hodnoty proměnné *pState->u_rms*,což je vhledem ke skutečným hodnotám daných proměnných pravdou. Zpětné bitové posunutí je provedeno už na celkovém výsledku (viz. vztah 6.9), kdy je známa hodnota podílu ze situace se saturací $U_{ef} > I_{ef}$, která je násobena členem pod druhou odmocninou. Tento celkový výsledek je nutné bitově posunout doprava o hodnotu *16-num_shifts*, neboť se jedná o 32-bitové číslo *long*.

Konečný výsledek operací na DSP reprezentuje výsledek, který je poté na výstupním portu bloku *Výpočet parametrů*. Dalším a posledním krokem je provedení

zpětné normalizace daných fyzikálních veličin, které byly součástí výpočtu na DSP dle (6.9) s také přepočet z induktance na indukčnost. To vše je součástí bloku *Back_scale*, kde se nachází přepočetní vztah $\frac{scale_u}{scale_i \cdot 2 \cdot \pi \cdot 50 \cdot 2^{15}}$, který v sobě skrývá vztah (6.10).

6.7 ZHODNOCENÍ IMPLEMENTACE NO-LOAD TESTU NA DSP

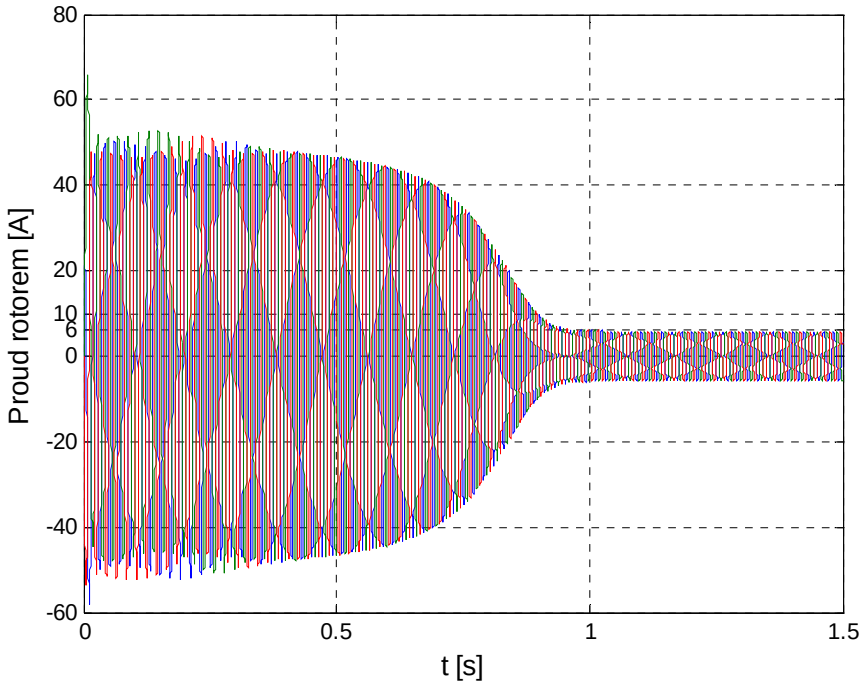
V případě No-load testu, který je v práci odsimulován v Simulinku a jehož výpočty provádí MATLAB, jehož výpočetní jádro pracuje na procesoru osobního počítače, se jedná o čistou simulaci No-load testu jako takového. Jedná se o variantu řešení, která více či méně poukazuje na funkčnost daného přístupu a dané metody, než-li na kvalitu a eventuelní použití této simulace pro reálnou aplikaci.

V druhém případě se simulace, jejíž výpočetní algoritmus je zpracováván na digitálním signálovém procesoru, už krůček po krůčku blíží reálnému využití na skutečném zařízení, jehož parametry jsou předmětem zkoumání. Po určitých specifických úpravách a by bylo možné tuto implementaci posunout svojí funkčností až k reálnému AM. Mezi hlavní a dá se říci i nejdůležitější úpravy by patřilo zejména kvalitní měření skutečných veličin pomocí příslušných měřících zařízení s implementovaným převodem dané veličiny do číslicové podoby takovým způsobem, aby bylo možné následné zpracování na příslušném výpočetním prostředku, např. přímo na DSP.

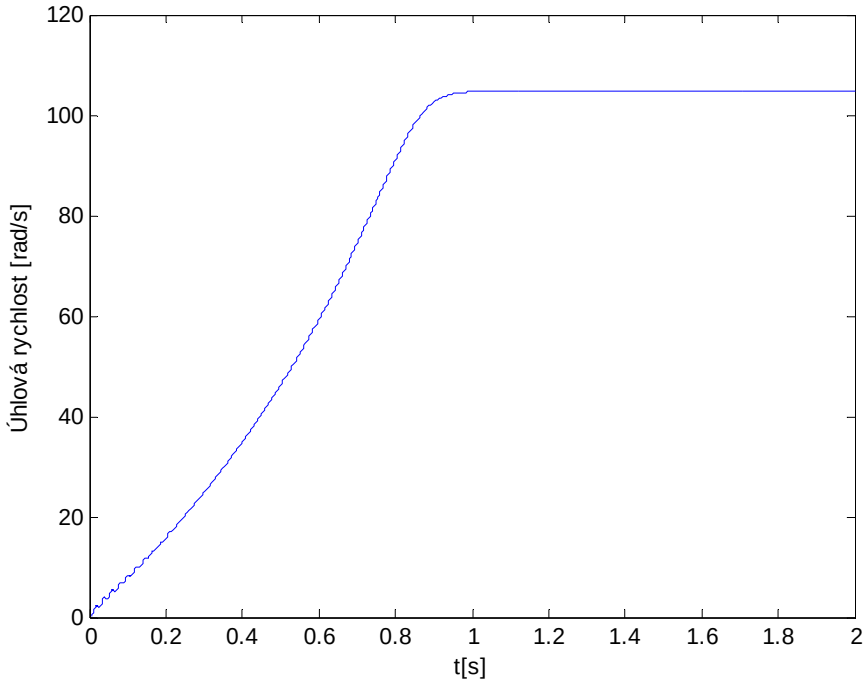
Oba přístupy dávají ve své podstatě kvalitní výsledek v podobě identifikovaných parametrů AM pomocí No-load testu, tj. součet rozptylové indukčnosti statoru a magnetizační indukčnosti $L_{sl} + L_m = 0.1192H$. V poslední kapitole pojednání o No-load testu jsou na obrázcích uvedeny průběhy okamžitých hodnot některých veličin v čase. Tyto obrázky mají ilustrativní charakter pro názornost průběhu změn daných veličin v čase.

6.8 PRŮBĚHY OKAMŽITÝCH HODNOT VELIČIN V ČASE

Na výše uvedených obrázcích je možné shlédnout typické průběhy pro No-load test, tj. charakteristiky výstupních veličin v čase *t*.



Obrázek 13: Okamžité průběhy proudů statorem

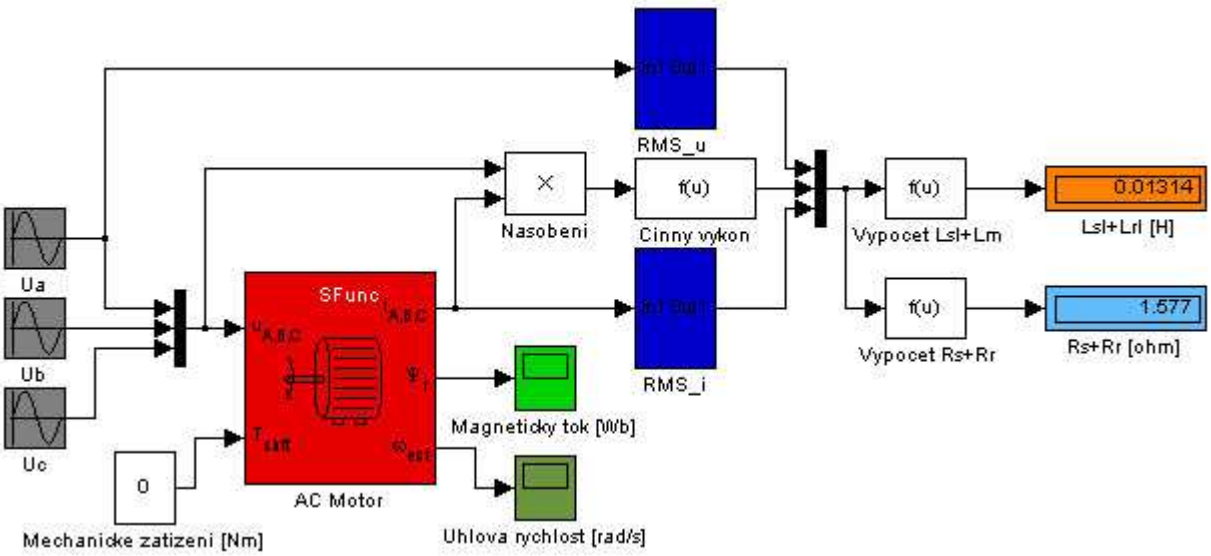


Obrázek 14: Okamžitý průběh úhlové rychlosti

7. LOCKED ROTOR TEST

Hlavní náplní tohoto testu je identifikovat parametry $L_{sl} + L_{rl}$, tj. součet rozptylových indukčností statoru a rotoru. Navíc je tam přidána identifikace $R_s + R_r$, tj. odporu statoru a rotoru. Locked rotor test je svým přístupem velice podobný testu No-load. Využívá takřka identický postup při výpočtech dílčích částí algoritmu, tak také obdobného schématu simulace jako předchozí metoda s menšími rozdíly v jeho blokové struktuře. Tím hlavním rozdílem je však odlišnost v principu testu a to taková, že je moment setrvačnosti motoru nastaven z hodnoty $J = 0.35 kg \cdot m^2$ (případ No-load testu) na nekonečnou hodnotu $J = \inf$ (inf = nekonečně velká hodnota). Význam tohoto spočívá v navození takového stavu, kdy rotor motoru není schopen díky vlivu tohoto parametru se jakkoliv roztočit. Veškerý činný výkon, který je v simulaci určen je výkon „pálící“ se na vinutích statoru a rotoru. Pro simulaci a identifikaci je nutné vinutí statoru napájet napětím menším než je jmenovité napětí a poté je nutno měřit statorové proudy a statorová napětí a nakonec z těchto hodnot určit výkon motoru, tj. činný a také jalový. Využívá se opět stejného modelu motoru jako v předešlých testech. [5]

7.1 BLOKOVÉ SCHÉMA SIMULACE V SIMULINKU



Obrázek 15: Blokové schéma Locked rotor testu

7.2 PRINCIP SIMULACE

Jak již bylo zmíněno, Locked rotor test je obdobou No-load testu s menšími rozdíly. Při tomto testu je rotor jakoby mechanicky ukotven. Je to způsobeno momentem setrvačnosti $J = \inf$, který je implicitně nastaven v bloku motoru jako počáteční parametr a nedovolí tudíž rotoru se vlivem jeho nekonečně velké hodnotě momentu setrvačnosti se roztočit.

Princip simulace vychází z blokového modelu uvedeného na **Obrázek 15**. Základem je, stejně jako u DC a No-load testu blok modelu AM. Schéma obsahuje bloky, které reprezentují napájení, dále bloky pro analýzu a zpracování jednotlivých signálů. V této metodě se rovněž využívá 3f napájení statorového vinutí. I v tomto modelu je napájecím prvkem soustava třech bloků (označeny šedou barvou), která je přivedena pomocí vazeb na motor. Tento zdroj (ve schématu šedé bloky) je tvořen třemi sinusovými průběhy, které reprezentují jednotlivá fázová napětí a mají své amplitudy rovny 120V a své fáze vzájemně symetricky posunuty o 120° (viz. 2.1). Moment zatížení motoru symbolizuje blok Mechanické *zatížení* a je roven nule ($M_z = 0$). Výstupem z motoru jsou veličiny jako magnetický tok Ψ_r, proud I a úhlová rychlost ω. Tyto veličiny symbolizují bloky *Magnetický tok a Úhlová rychlost*. Jak je vidět z **Obrázek 15**, signál z bloku *Činný* výkon, který skrývá výpočet činného výkonu je dále veden do multiplexeru, kde se schází tento signál spolu se signály z bloků *RMS*, který počítá efektivní hodnotu U_{ef} z fázového napětí U_a a s podobným, který vypočítá efektivní hodnotu I_{ef} z proudu I_a. Pomocí dalších bloků, které symbolizují matematické operace, potřebné ke stanovení výsledku identifikace, je v bloku *Výpočet* $L_{sl} + L_{rl}$ vypočten a poté zobrazen výsledek této identifikační metody. Tyto operace budou uvedeny v dalších odstavcích z této identifikační metody.

7.3 URČENÍ PRAMETRŮ L_{SL}+L_{RL}

Identifikace je založena na stejném principu jako No-load test . Zanedbává se však magnetizační indukčnost L_m a proud tedy teče přes odpor statoru R_s, dále přes rozptylovou indukčnost statoru L_{sl}, poté přes rozptylovou indukčnost rotoru a

nakonec samotným odporem rotoru R_r. Princip identifikace vychází, stejně jako u No-load testu, z určení parametrů prostřednictvím činného výkonu P, ze kterého se určí fáze φ (přes účiník $\cos\varphi$) a pomocí něj se určí jalový výkon Q_a.

7.3.1 Blok RMS

Jak je uvedeno ve schématu simulace Locked rotor testu (viz. **Obrázek 15**), je znovu využit blok *RMS* vytvořený v předchozím testu. Schéma bloku *RMS* je uvedeno na **Obrázek 11** spolu s detailním popisem včetně výpočetního principu efektivní hodnoty daných veličin. Efektivní hodnoty proudu I_{ef} a efektivní hodnoty napětí U_{ef} jsou potřebné k výpočtu parametrů AM $L_{sl} + L_{rl}$.

7.3.2 Princip identifikace

Metoda vychází ze stejných rovnic jako No-load test. Pro rozptylové reaktance statoru a rotoru platí

$$X_{sl} + X_{rl} = \frac{Q_a}{I_{ef}^2} \tag{7.1}$$

kde X_{sl} je rozptylová reaktance statoru, X_{rl} je rozptylová reaktance rotoru, podíl $\frac{Q_a}{I_{ef}^2}$ udává jalový výkon na jedné fázi ku mocnině efektivní hodnoty proudu touto fází.

7.3.3 Výpočet parametrů $L_{sl} + L_{rl}$

Jak je uvedeno ve vztahu (6.1), je možné určit parametry $L_{sl} + L_{rl}$, resp. $X_{sl} + X_m$ pomocí znalosti jalového výkonu Q_a a efektivní hodnoty proudu I_{ef}. K tomu je však mimo to nutné znát velikost fáze φ z účiníku $\cos\varphi$, který je možno určit z činného výkonu P. Výpočet účiníku udává následující vztah

$$\cos\varphi = \frac{P}{3 \cdot U_{ef} I_{ef}} \tag{7.2}$$

kde P je celkový činný výkon motoru, U_{ef} a I_{ef} jsou efektivní hodnoty fázového napětí a proudu.

Pro výpočet jalového výkonu na jedné fázi motoru platí následující vztah

$$Q_a = U_{ef} I_{ef} \sin\varphi \qquad (7.3)$$

kde Q_a je jalový výkon a U_{ef} a I_{ef} jsou efektivní hodnoty fázového napětí a proudu. V rovnici (6.1) se parametry určují jen na jedné fázi, proto je třeba počítat jen s jalovým výkonem jedné fáze.

Z rovnic (7.2) a (7.3) je možno matematickou operací získat rovnici (7.1), přesněji tedy získat člen $\frac{Q_a}{I_{ef}^2}$, který je přímo roven součtu rozptylové reaktance statoru a rozptylové reaktance rotoru, který jsou pro další přepočet na indukčnosti stěžejní.

Následující rovnice obsahují matematickou interpretaci výše uvedených tvrzení vycházející z rovnic (7.2) a (7.3) a odvozují vztah pro výpočet $L_{sl} + L_m$.

Platí obecný vztah pro součet mocnin funkcí se stejným argumentem $\sin\varphi$ a $\cos\varphi$

$$\sin^2\varphi + \cos^2\varphi = 1 \qquad (7.4)$$

z něj se matematickou úpravou s následnou aplikací vztahu (6.2) dostane

$$\sin\varphi = \sqrt{1 - \left(\frac{P}{3 \cdot U_{ef} I_{ef}}\right)^2} \qquad (7.5)$$

Rovnice (7.3) obsahuje člen $\sin\varphi$, který je po osamostatnění vložen do levé strany rovnice (7.5)

$$\frac{Q_a}{U_{ef} I_{ef}} = \sqrt{1 - \left(\frac{P}{3 \cdot U_{ef} I_{ef}}\right)^2} \qquad (7.6)$$

V tomto vztahu je na levé straně nutné osamostatnit levou část na podíl $\frac{Q_a}{I_{ef}^2}$, tj. je nutné obě strany této rovnice vynásobit podílem $\frac{U_{ef}}{I_{ef}}$, což je běžná matematická úprava . Výsledkem je vztah

$$\frac{Q_a}{I_{ef}^2} = \sqrt{1 - \left(\frac{P}{3 \cdot U_{ef} I_{ef}}\right)^2} \cdot \frac{U_{ef}}{I_{ef}} \qquad (7.7)$$

Tento vztah udává součet rozptylové reaktance statoru a rozptylové reaktance rotoru. Locked rotor test však identifikuje parametry $L_{sl} + L_{rl}$, proto je tedy nutné

přepočítat reaktance na indukčnosti dle následujícího vztahu, tj. podělit induktance členem $2 \cdot \pi \cdot f$

$$L_{sl} + L_{rl} = \frac{\frac{Q_a}{I_{ef}^2}}{2 \cdot \pi \cdot f} \tag{7.8}$$

kde π je konstanta a f je frekvence napájecí sítě.

Dosazením konkrétních hodnot veličin, které obsahuje vztah (7.7) by byly vypočítány parametry, které v simulaci vyšly rovny hodnotě $L_{sl} + L_{rl} = 0.01304H$.

Matematické operace a vztahy jsou ve schématu simulace reprezentovány blokem *Výpočet* (viz. **Obrázek 15**), který obsahuje vztahy (7.7) a (7.8). Výsledek simulace a celé metody je zobrazen ve výstupním zobrazovacím bloku jako skutečná hodnota získaných parametrů.

7.3.4 Výpočet parametrů $R_s + R_r$

Kroky k získání těchto parametrů jsou přidány k Locked rotor testu jako by navíc. Již v DC testu uvedeném v kapitole 5 je určena hodnota parametru odporu statoru R_s . Určení součtu odporu statoru a rotoru $R_s + R_r$ vychází z následujícího vztahu

$$R_s + R_r = \frac{U_{ef}}{I_{ef}} \cdot \left(\frac{P}{3 \cdot U_{ef} \cdot I_{ef}} \right) \tag{7.9}$$

Tento vztah se dá zjednodušit, neboť člen uvedený v závorkách je vlastně přímo roven účiníku $\cos\varphi$ (viz 7.2)

$$R_s + R_r = \frac{U_{ef}}{I_{ef}} \cdot \cos\varphi \tag{7.10}$$

Tyto matematické operace jsou ukryté ve funkčním bloku *Výpočet* $R_s + R_r$.

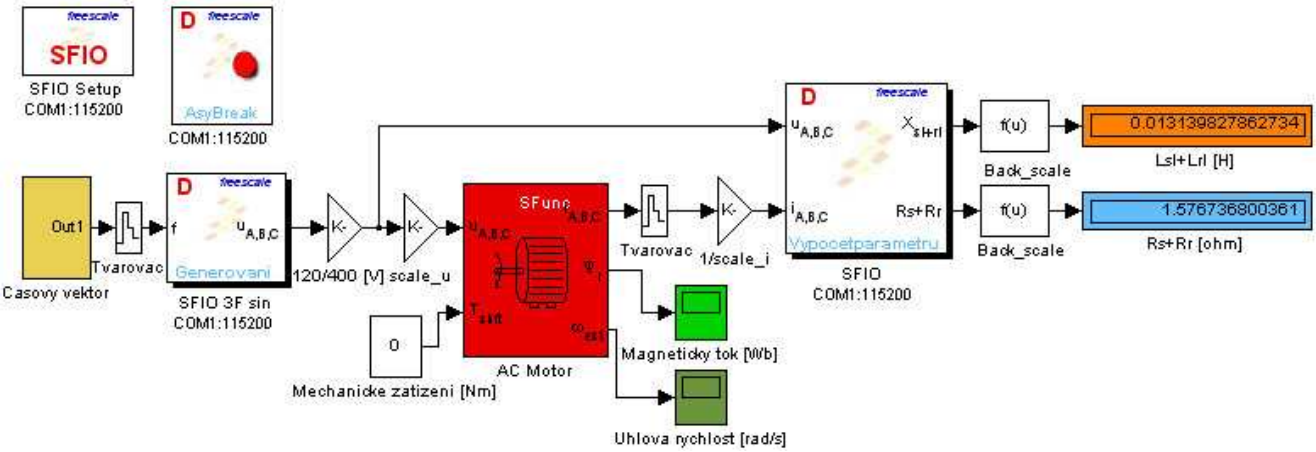
7.4 IMPLEMENTACE LOCKED ROTOR TESTU NA DSP

Postup při realizaci implementace výpočetního algoritmu do DSP opět vychází svým pojetím ze simulace běžící výpočetně v MATLABu jako u předchozích identifikačních metod. Opět bylo dobré pro názornost a přehlednost vycházet z původního blokového uspořádání popisujícího Locked rotor test (**Obrázek 15**) a implementaci provést co nejnázorněji vzhledem k tomuto přístupu.

Podstata implementace tedy spočívá v naprogramování výpočetního algoritmu pro DSP tak, aby byl výpočetně zpracován právě na DSP.

Zdrojový kód, psaný jazykem C (soubor *locked_rotor.c*) v prostředí CodeWarrior, obsahuje na svém začátku standardně deklarace knihoven jako např. knihovny zajišťující komunikaci a konfiguraci parametrů mezi DSP a MATLABem (*sfio.h*, *sfio_cfg.h*), dále standardní knihovnu pro I/O operace (*stdio.h*) či také knihovnu definující příslušné datové typy (*types.h*) ad. Převážnou část souboru *locked_rotor.c* tvoří deklarace funkcí skrývající podstatu jednotlivých bloků, které se vytvoří automaticky po označení *SFIO* bloků s názvy *Generování* či *Výpočet parametrů*, které má předem nadefinované a nastavené vstupní a výstupní porty dle principu výpočtů (počet portů ad.), a následném spuštění příkazu *codegen* napsaného přímo na příkazový řádek MATLABu. Vygenerované zdrojové kódy jsou poté vloženy do zdrojového souboru *locked_rotor.c*, kde jsou do jejich těla vepisovány potřebné příkazy či vlastní vytvořené funkce dle konkrétního řešení. Součástí těchto funkcí, reprezentující SFIO bloky, jsou struktury, tj. sdružené datové typy jazyka C, obsahující položky pro datové typy vstupních, výstupních a stavových proměnných.

7.5 BLOKOVÉ SCHÉMA IMPLEMENTACE NA DSP



Obrázek 16: Blokové schéma implementace Locked rotor testu na DSP

Obrázek 16 obsahuje blokové schéma simulace Locked rotor testu, jehož výpočetní algoritmus je prováděn na DSP. Struktura simulace je velmi podobná a vycházející ze simulace, jejíž výpočet je zpracován na straně MATLABu (**Obrázek 15**).

Algoritmus implementace Locked rotor testu je opět naprogramován v jazyce C v prostředí CodeWarrior. Simulační schéma obsahuje některé přidané funkční bloky nesoucí loga společnosti Freescale. Jedná se o identické bloky jako v případě předchozích testů, tj. bloky SFIO toolboxu, které mají za úkol obstarávat komunikaci po sériové lince RS-232 (*SFIO*) mezi DSP a PC a potom také neméně významné bloky, které v sobě nepřímo skrývají naprogramované části zdrojového kódu či spíše funkce výpočetního algoritmu dané identifikační metody (*Výpočet parametrů* a *Generování*). Do schématu jsou rovněž také přidány důležité bloky pro normalizaci fyzikálních veličin pro algoritmus na DSP a případný zpětný normalizační přepočet (*1/scale_i* a *Back_scale*).

7.6 ALGORITMUS LOCKED ROTOR TESTU V JAZYCE C

Hlavní myšlenkou implementace celé simulace je to, že takřka celý výpočetní algoritmus běží na straně DSP. Jedná se zejména o generování napájecího napětí, dále výpočet efektivních hodnot napájecího napětí a výstupního proudu s následným konečným výpočtem identifikovaných parametrů $L_{sl} + L_{rl}$, tj. součet rozptylových indukčností statoru a rotoru. Podstata implementace tedy závisí na vytvoření algoritmu, který bude tyto výpočty řešit. Konkrétní postup a popis algoritmů výpočtů, tedy popis funkcí jednotlivých bloků je popsán jednotlivě v dalších kapitolách. Celá implementace je součástí zdrojového souboru s názvem *locked_rotor.c*.

7.6.1 Generování napájecího napětí

Řešení generování 3 - fázového napájecího napětí je identické jako v případě implementace No-load testu na DSP. Opět se jedná o zápis funkce v jazyce C, která má jeden vstupní parametr (úhel z bloku *Časový vektor*) a výstupem je vypočtená funkční hodnota dané funkce. Celý vytvořený algoritmus je podrobně popsán v kapitole 6.6.1 a není nutné ho tedy v těchto místech znovu zmiňovat a dále popisovat.

7.6.2 Algoritmus výpočtu efektivní hodnoty RMS ze signálu na DSP

K výpočtu parametrů AM je třeba, stejně jako v předešlém No-load testu, výpočtu efektivních hodnot jak napětí U_{ef} tak i proudu I_{ef}. Celý algoritmus výpočtu

s vysvětlením principu a jeho přepsáním do jazyka C je popsán v kapitole 6.6.2. Princip obecného výpočtu efektivních hodnot je uveden v kapitole 6.3.1.

7.6.3 Algoritmus výpočtu parametrů $L_{sl} + L_{rl}$ na DSP

Algoritmus výpočtu parametrů $L_{sl} + L_{rl}$ pomocí implementace Locked rotor testu běžícího na straně DSP je předmětem SFIO bloku pod názvem *Výpočet parametrů* a jeho funkce se stejným názvem, která je součástí souboru *locked_rotor.c*. Jedná se výpočetní blok mající dva vstupní a dva výstupní porty. Prvním vstupem jsou 3 vektory okamžitých hodnot napájecích napětí *u_ABC[3]*, které jsou výstupem z SFIO bloku *Generování*, podobně jako v případě (**Obrázek 15**). Druhým vstupem stejného bloku jsou rovněž 3 vektory, ale okamžitých hodnot výstupních proudů *i_ABC[3]* z AM. Jeden výstupní port je vyhrazen pro hodnotu výsledné proměnné *X* symbolizující součet rozptylové induktance statoru a rozptylové induktance rotoru $X_{sl} + X_{rl}$ a druhý pro výsledek součtu odporu statoru a rotoru $R_s + R_r$. Identifikace parametrů $R_s + R_r$ pomocí této metody není standardně její součástí a je implementována do metody jakoby navíc. Hlavním předpokladem je však fakt, že Locked rotor test je určitou modifikací No-load testu. Výpočty parametrů ze známých či vypočtených veličin jsou identické jako v případě No-load testu a jejich vysvětlení je předmětem kapitoly 7.3.

7.6.4 Algoritmus výpočtu parametrů $R_s + R_r$ na DSP

Podstata určení parametrů $R_s + R_r$ vychází ze vztahu (7.9). Jedná se o prostý výpočet pomocí účiníku $\cos\varphi$, který tvoří právě člen v závorkách. Tento člen je součástí dílčího výpočtu pro určení parametrů $L_{sl} + L_{rl}$. Ve vztahu (7.9) je možné matematickou operací krácení docílit zjednodušeného vztahu.

$$R_s + R_r = \frac{P}{3 \cdot I_{ef} \cdot I_{ef}} \tag{7.11}$$

Další možností je nekrátit výraz (7.9) a řešit to pomocí dalších dílčích operací, u kterých je nutno řešit problém se saturací vzniklou při podílu dvou hodnot, z nichž hodnota čitatele je větší než jmenovatele.

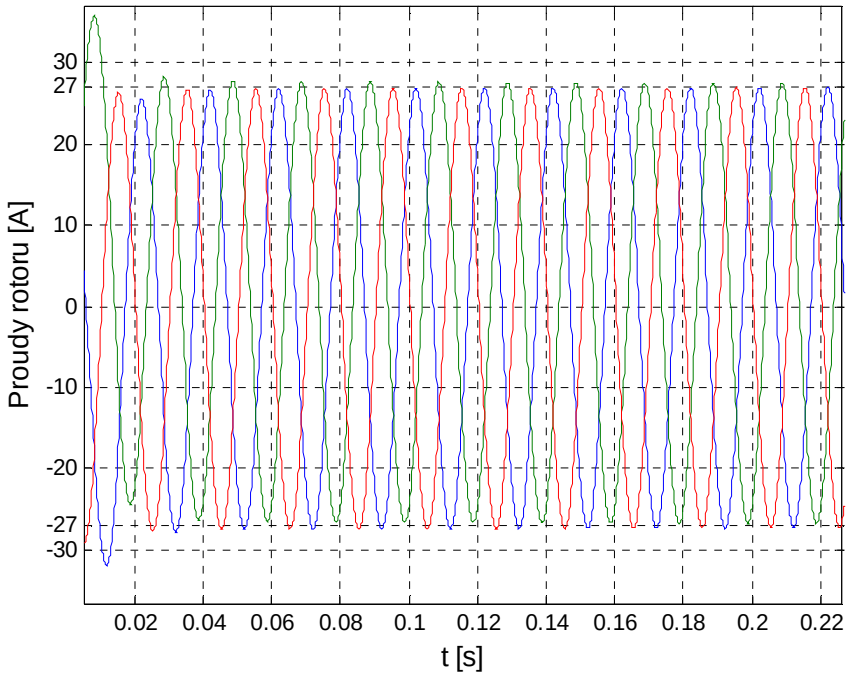
Konečným krokem je zpětná normalizace veličin pomocí bloku *Back_scale*, který obsahuje vztah $\frac{scale_u}{scale_i \cdot 2^{15}}$.

7.7 ZHODNOCENÍ IMPLEMENTACE LOCKED ROTOR TESTU NA DSP

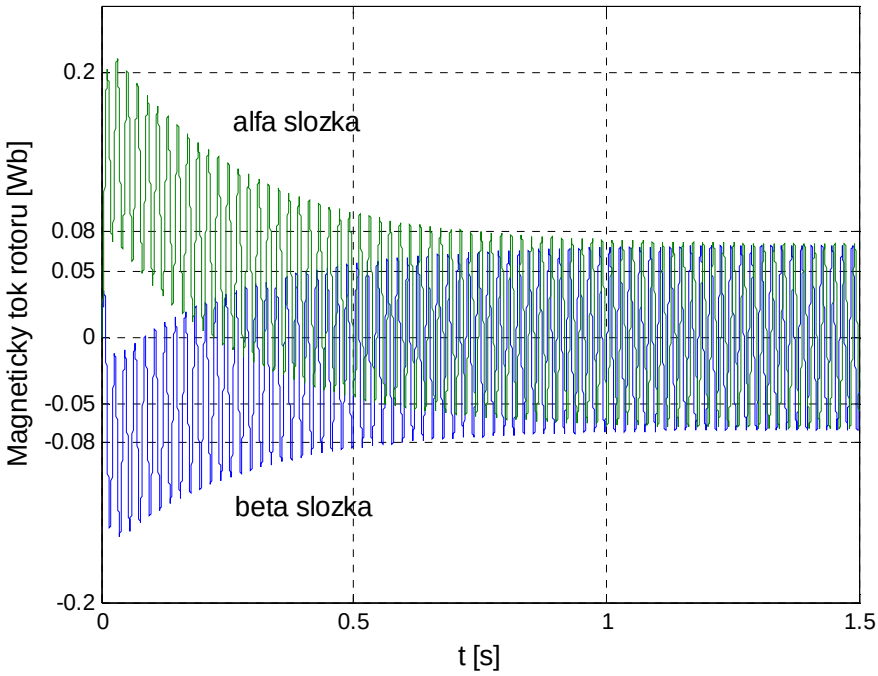
Rozhodnutí o kvalitě zpracování, obecně každého problému, dávají až následné testy takového řešení. Oba přístupy simulace Locked rotor testu zmíněné v práci dávají ve své podstatě kvalitní výsledky v podobě identifikovaných parametrů AM, tj. výsledkem je součet rozptylové indukčnosti statoru a rozptylové indukčnosti rotoru $L_{sl} + L_{rl} = 0.01314H$. Identifikovaný součet odporu statoru a rotoru, který je v práci řešen dodatečně vyšel roven $R_s + R_r = 1.577\Omega$.

7.8 PRŮBĚHY OKAMŽITÝCH HODNOT VELIČIN V ČASE

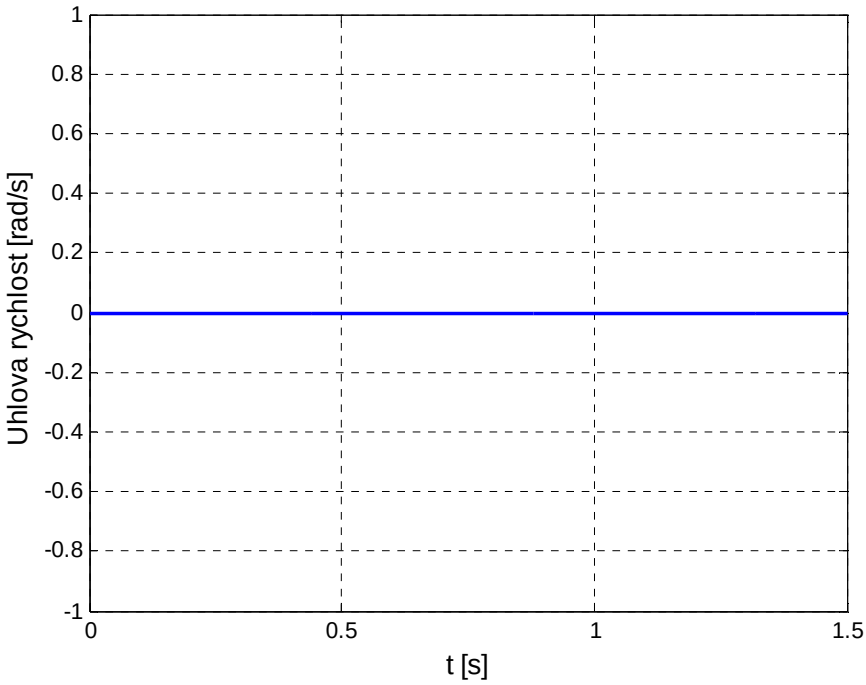
Na níže uvedených obrázcích je možné poukázat na některé jevy, které jsou v Locked rotoru přítomny a stojí nepochybně za zmínku. Např. úhlová rychlost je po celou dobu simulace nulová (rotor ukotven) (viz. **Obrázek 17**) či okamžité průběhy obou složek magnetického toku rotorem, tj. momentotvorné složky (zelená) a tokotvorné složky (modrá) (**Obrázek 18**).



Obrázek 17: Okamžité průběhy proudů statorem



Obrázek 18: Okamžité průběhy složek magnetického toku rotorem

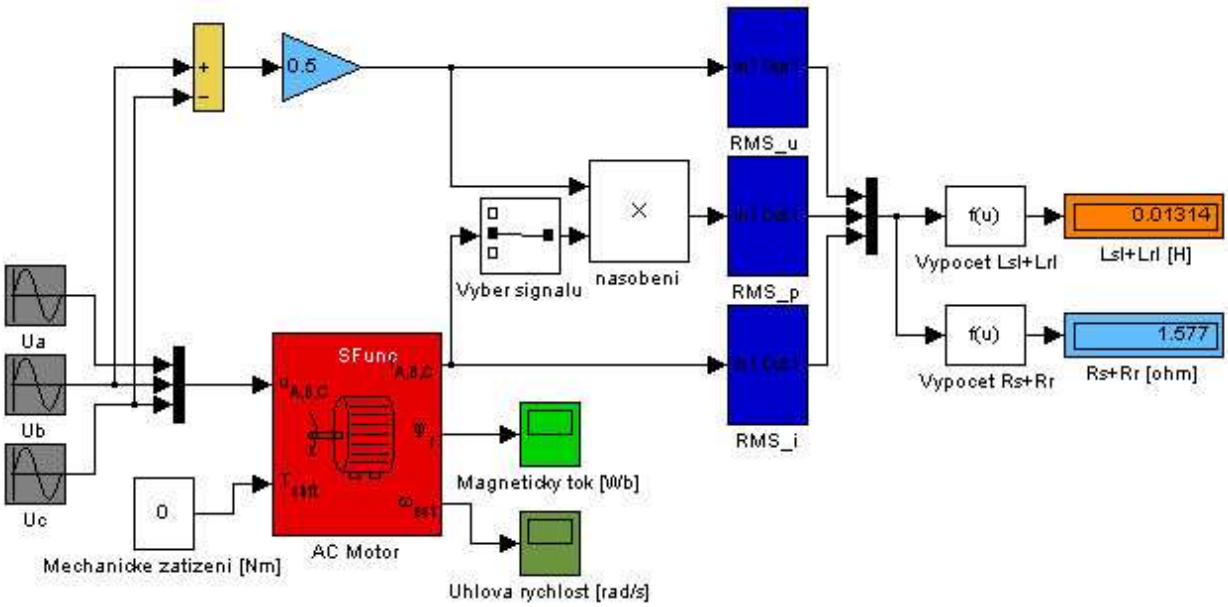


Obrázek 19: Okamžitý průběh úhlové rychlosti

8. ONE PHASE TEST

Tato metoda patří mezi další a zároveň poslední identifikační metody zmiňované v této práci. Svojí podstatou vychází z Locked rotor testu, jehož princip je zmíněn v předchozí části této práce. Cílem One phase testu je, stejně jako u Locked rotor testu, identifikovat $L_{sl} + L_{rl}$, tj. získat součet rozptylových indukčností statoru a rotoru. Opět je do simulace přidána identifikace odporů statorového R_s a rotorového vinutí R_r. Hlavní podstata principu tohoto testu je ve stanovení parametrů AM při situaci, kdy se navodí taková konfigurace napájení AM, aby jednou fází netekl proud (tj. aby platilo např. $I_a = 0A$). Z tohoto předpokladu vychází řešení, které je uvedeno v dalších částech práce. [4]

8.1 BLOKOVÉ SCHÉMA SIMULACE V SIMULINKU



Obrázek 20: Blokové schéma simulace One phase testu

8.2 PRINCIP SIMULACE

Simulace vychází z blokového schématu (viz. **Obrázek 20**) a je velmi podobné schématům předchozích testů (No load test, Locked rotor test) jak podobou tak i samotným principem identifikace. Základem je, stejně jako u předchozích testů, blok modelu asynchronního motoru. Stejně jako u Locked rotor testu je rotor AM

jakoby mechanicky ukotven. Je to způsobeno opět volbou momentu setrvačnosti $J = \inf$ (inf = nekonečná velikost), který nedovolí rotoru se vlivem působení magnetického pole na jeho vinutí roztočit. Schéma obsahuje bloky, které reprezentují napájení, dále bloky pro analýzu a zpracování jednotlivých signálů.

V této metodě se rovněž využívá 3f napájení statorového vinutí. I v tomto modelu představuje zdroj napájení soustava třech bloků (označeny šedou barvou), která je přivedena pomocí vazeb přímo na blok motoru. Tento zdroj (ve schématu šedé bloky) je tvořen třemi sinusovými průběhy, které reprezentují jednotlivá fázová napětí a mají své amplitudy U_m rovny 120V (svorky b a c) a 60V (svorka a). Proč je tomu tak, že jedno fázové napětí je menší než ostatní, bude vysvětleno později na konkrétním případě. Moment zatížení motoru symbolizuje blok *Mechanické zatížení* a je roven nule ($M_z = 0Nm$). Výstupem z motoru jsou veličiny jako magnetický tok rotorem Ψ_r, proudy všech fází I_a, I_b, I_c a úhlová rychlost otáčení rotoru ω. Tyto veličiny symbolizují bloky *Magnetický tok a Úhlová rychlost*.

8.3 URČENÍ PRAMETRŮ L_{SL}+L_{RL}

Podstata identifikace pomocí One phase testu spočívá ve vytvoření určitých podmínek v napájení statorového vinutí a v následném výpočtu identifikovatelných parametrů z potřebných veličin, jejichž velikosti nám tento test přináší a je možné je případně změřením a výpočtem určit. Jak již bylo řečeno, je nutné napájet statorová vinutí fázovými napětími s typickými parametry. Podstata napájení motoru v tomto testu spočívá v připojení napájecího napětí U_b a U_c jen na dvě svorky b a c. Tzn., že první svorka a není jakoby obsazena a neteče jí tedy žádný proud.

8.3.1 Blok RMS

Jak je uvedeno ve schématu simulace One phase testu (viz. **Obrázek 20**), je opět využit blok RMS vytvořený v No-load testu. Schéma bloku RMS je uvedeno na **Obrázek 11** spolu s následným detailním popisem včetně výpočetního principu efektivních hodnot daných signálových veličin. V tomto testu jsou efektivní hodnoty jak proudu I_{ef}, tak i efektivní hodnoty napětí U_{ef} jsou potřebné ke stanovení

parametrů $L_{sl} + L_{rl}$. Oproti Locked rotor testu je nutné ještě určit efektivní hodnotu z činného výkonu P_{ef}.

8.3.2 Princip identifikace

První krok identifikace je založen na určení (výpočtu) napětí U_a, které se musí připojit na napájecí svorku a statoru. Tento úkol musí tedy zabezpečit jev, aby touto svorkou tekl nulový proud, tj. situace, kdy je daný vodič přerušen. Proudy tedy tečou statorem jen přes vinutí, která jsou spojena se svorkami b a c. Z této úvahy je možno jednoduše zjistit napětí na jedné z těchto svorek a pomocí ní dále pokračovat v identifikaci parametrů pomocí One phase testu. Způsob určení napájecího napětí pro svorku a je uveden na následujícím odvození. [4]

Pro fázová napětí připojená ke svorkám statoru b a c platí

$$U_b = U_m \cdot \sin(\omega_0 t)$$
$$U_c = U_m \cdot \sin(\omega_0 t - \frac{2 \cdot \pi}{3}) \tag{8.1}$$

Pro určení napájecího napětí pro svorku a je nutné napájecí napětí obou svorek (8.1) sečíst a podělit dvěmi. Výsledkem tedy bude napětí, které se poté připojí na svorku a. Platí

$$U_a = \frac{U_a + U_b}{2} \tag{8.2}$$

$$U_a = \frac{U_m \cdot \sin(\omega_0 t) + U_m \cdot \sin(\omega_0 t - \frac{2 \cdot \pi}{3})}{2} \tag{8.3}$$

$$U_a = \frac{U_m \left(\sin(\omega_0 t) + \sin\left(\omega_0 t - \frac{2 \cdot \pi}{3} \right) \right)}{2} \tag{8.4}$$

kde matematickou úpravou za použití vztahu pro součet dvou goniometrických funkcí s různými argumenty platí

$$\sin x + \sin y = 2 \cdot \sin \frac{x+y}{2} \cdot \cos \frac{x-y}{2} \tag{8.5}$$

kde po aplikaci vztahu (8.5) na vztah (8.4) platí

$$U_a = \frac{U_m}{2} \cdot 2 \cdot \frac{\sin\left(2\omega_0 t - \frac{2 \cdot \pi}{3}\right)}{2} \cdot \cos\left(\frac{\frac{2 \cdot \pi}{3}}{2}\right) \tag{8.6}$$

$$U_a = U_m \cdot \sin\left(\omega_0 t - \frac{\pi}{3}\right) \cdot \cos\left(\frac{\pi}{3}\right) \tag{8.7}$$

$$U_a = \frac{U_m}{2} \cdot \sin\left(\omega_0 t - \frac{\pi}{3}\right) \tag{8.8}$$

Při připojení napětí na svorku a, jehož velikost udává vztah (8.8), je zaručeno, že touto svorkou poteče nulový proud $I_a = 0$. Tím pádem platí, že připojením napájecího napětí U_b a U_c vznikne v symetrických (rovnocenných) vinutích statoru důsledek v podobě proudů I_b a I_c, které mají stejnou velikost ale opačná znaménka, tj. jejich součet je nulový, což je reálné. Pro výpočet napětí U_{b-c} mezi svorkami b a c tedy platí

$$U_{b-c} = \frac{U_b - U_c}{2} \tag{8.9}$$

$$U_{b-c} = \frac{U_m \cdot \sin(\omega_0 t) - U_m \cdot \sin(\omega_0 t - \frac{2 \cdot \pi}{3})}{2} \tag{8.10}$$

$$U_{b-c} = \frac{U_m \cdot \sin(\omega_0 t) - U_m \cdot \sin(\omega_0 t - \frac{2 \cdot \pi}{3})}{2} \tag{8.11}$$

Zde je nutné použít vztah pro rozdíl goniometrických funkcí s různými argumenty

$$\sin x - \sin y = 2 \cdot \cos\frac{x+y}{2} \cdot \sin\frac{x-y}{2} \tag{8.12}$$

kde potom platí

$$U_{b-c} = U_m \cdot \cos\left(2 \cdot \omega_0 t - \frac{2 \cdot \pi}{3}\right) \cdot \sin\left(\frac{\frac{2 \cdot \pi}{3}}{2}\right) \tag{8.13}$$

kde platí, že funkce $\cos(x)$ je funkcí sudou, je možné vztah (8.13) zjednodušit na

$$U_{b-c} = \frac{\sqrt{3}}{2} U_m \cdot \cos\left(\omega_0 t - \frac{\pi}{3}\right) \tag{8.14}$$

kde převedením na funkci $\sin(x)$ pomocí obecného vztahu

$$\cos(x) = \sin\left(x + \frac{\pi}{2}\right) \tag{8.15}$$

platí

$$U_{b-c} = \frac{\sqrt{3}}{2} U_m \cdot \sin\left(\omega_0 t + \frac{\pi}{6}\right) \tag{8.16}$$

Ve vztahu (8.16) je uvedeno napětí, které je v (viz. **Obrázek 20**) vypočteno za pomocí rozdílového členu (žlutý blok) a následně jeho hodnota násobena hodnotou 0.5 (světle modrý blok).

8.3.3 Výpočet parametrů $L_{sl} + L_{rl}$

Určení parametrů $L_{sl} + L_{rl}$, resp. $X_{sl} + X_m$ pomocí znalosti jalového výkonu Q_a a efektivní hodnoty proudu I_{ef} je stejné jako v Locked rotor testu či No-load testu. K tomu je nutné znát velikost fáze φ z účiníku $\cos\varphi$, který je možno určit z činného výkonu *P*. Výpočet účiníku udává následující vztah

$$\cos\varphi = \frac{P_{ef}}{3 \cdot U_{ef} I_{ef}} \tag{8.17}$$

kde P_{ef} je efektivní hodnota celkového činného výkonu motoru, U_{ef} a I_{ef} jsou efektivní hodnoty napětí (napětí mezi svorkami *a* - *b*) a proudu (fáze *b*).

Pro výpočet jalového výkonu na jedné fázi motoru platí následující vztah

$$Q_a = U_{ef} I_{ef} \sin\varphi \tag{8.18}$$

kde Q_a je jalový výkon a U_{ef} a I_{ef} jsou efektivní hodnoty fázového napětí a proudu.

Z rovnic (8.17) a (8.18) je možno matematickou operací získat člen, přesněji tedy získat podíl $\frac{Q_a}{I_{ef}^2}$, který je přímo roven součtu rozptylové reaktance statoru a rozptylové reaktance rotoru, který je pro další přepočet na indukčnosti stěžejní.

Následující rovnice obsahují matematickou interpretaci výše uvedených tvrzení vycházející z rovnic (8.17) a (8.18) a odvozují vztah pro výpočet $L_{sl} + L_{rl}$.

Platí obecný vztah pro součet mocnin funkcí se stejným argumentem $\sin\varphi$ a $\cos\varphi$

$$\sin^2\varphi + \cos^2\varphi = 1 \tag{8.19}$$

z něj se matematickou úpravou s následnou aplikací vztahu (8.17) dostane

$$\sin\varphi = \sqrt{1 - \left(\frac{P_{ef}}{3 \cdot U_{ef} I_{ef}}\right)^2} \tag{8.20}$$

Rovnice (8.18) obsahuje člen $\sin\varphi$, který je po osamostatnění vložen do levé strany rovnice (8.20)

$$\frac{Q_a}{U_{ef} I_{ef}} = \sqrt{1 - \left(\frac{P_{ef}}{3 \cdot U_{ef} I_{ef}}\right)^2} \tag{8.21}$$

V tomto vztahu je na levé straně nutné osamostatnit levou část na podíl $\frac{Q_a}{I_{ef}^2}$, tj. je nutné obě strany této rovnice vynásobit podílem $\frac{U_{ef}}{I_{ef}}$, což je běžná matematická úprava . Výsledkem je vztah

$$\frac{Q_a}{I_{ef}^2} = \sqrt{1 - \left(\frac{P_{ef}}{3 \cdot U_{ef} I_{ef}}\right)^2} \cdot \frac{U_{ef}}{I_{ef}} \tag{8.22}$$

Tento vztah udává součet rozptylové reaktance statoru a rozptylové reaktance rotoru. Locked rotor test však identifikuje parametry $L_{sl} + L_{rl}$, proto je tedy nutné přepočítat reaktance na indukčnosti dle následujícího vztahu, tj. podělit reaktance členem $2 \cdot \pi \cdot f$

$$L_{sl} + L_{rl} = \frac{\frac{Q_a}{I_{ef}^2}}{2 \cdot \pi \cdot f} \tag{8.23}$$

kde π je konstanta a f je frekvence napájecí sítě.

Dosazením konkrétních hodnot veličin, které obsahuje vztah (8.23) by byly vypočítány parametry, které v simulaci vyšly rovny hodnotě $L_{sl} + L_{rl} = 0.01304H$.

Tyto matematické operace a vztahy jsou ve schématu simulace reprezentovány blokem *Výpočet* $L_{sl} + L_{rl}$ (viz. **Obrázek 20**), které jsou vytvořeny

jako funkce obsahující tyto vztahy (8.22) a (8.23). Výsledek simulace a celé metody je zobrazen ve výstupním bloku.

8.3.4 Výpočet parametrů $R_s + R_r$

Kroky k získání těchto parametrů jsou přidány k One phase testu jako by navíc. Určení součtu odporu statoru a rotoru $R_s + R_r$ vychází z následujícího vztahu

$$R_s + R_r = \frac{U_{ef}}{I_{ef}} \cdot \left(\frac{P_{ef}}{3 \cdot U_{ef} \cdot I_{ef}} \right) \tag{8.24}$$

Tento vztah se dá zjednodušit, neboť člen uvedený v závorkách je vlastně přímo roven účiníku $\cos\varphi$ (viz 8.17)

$$R_s + R_r = \frac{U_{ef}}{I_{ef}} \cdot \cos\varphi \tag{8.25}$$

Tyto matematické operace jsou ukryté ve funkčním bloku *Výpočet* $R_s + R_r$. Výsledek simulace, tj. $R_s + R_r$ navazuje na blok *Výpočet* $R_s + R_r$ a je uveden jako blok, ve kterém je tento údaj přímo zobrazen (viz. **Obrázek 20**). Identifikací vyšla hodnota $R_s + R_r = 1.577\Omega$.

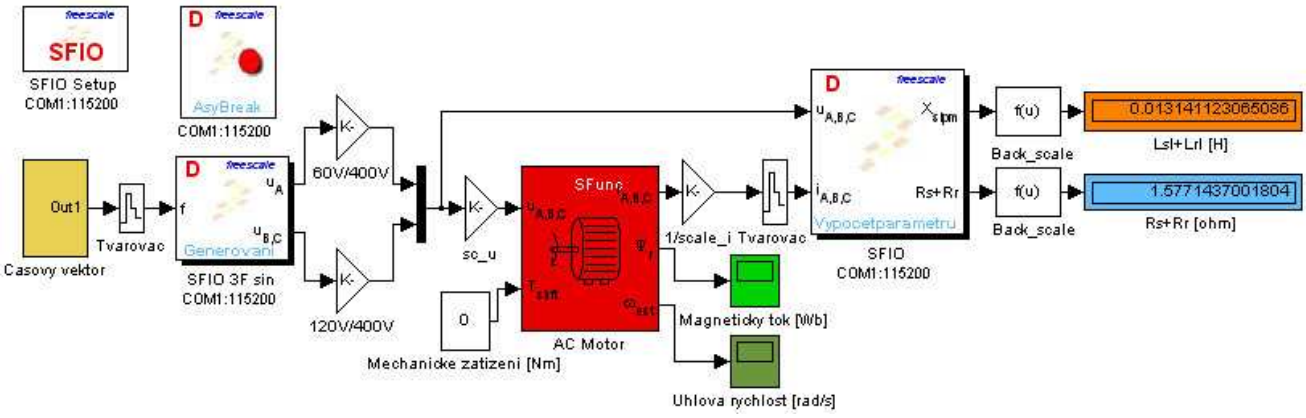
8.4 IMPLEMENTACE ONE PHASE TESTU NA DSP

Postup při realizaci implementace výpočetního algoritmu do DSP opět vychází svým pojetím ze simulace běžící výpočetně v MATLABu jako u předchozích identifikačních metod. Opět bylo dobré pro názornost a přehlednost vycházet z původního blokového uspořádání popisujícího One phase test (Obr. 8.1) a implementaci provést co nejnázorněji vzhledem k tomuto přístupu. Podstata implementace tedy spočívá v naprogramování výpočetního algoritmu pro DSP tak, aby byl výpočetně zpracován právě na DSP.

Zdrojový kód, psaný jazykem C (soubor *one_phase.c*) v prostředí CodeWarrior, obsahuje na svém začátku opět standardně deklarace knihoven jako např. knihovny zajišťující komunikaci a konfiguraci parametrů mezi DSP a MATLABem (*sfio.h, sfio_cfg.h*), dále standardní knihovnu pro I/O operace (*stdio.h*) či také knihovnu definující příslušné datové typy (*types.h*) ad. Převážnou část souboru *one_phase.c* tvoří deklarace funkcí skrývající podstatu jednotlivých bloků,

které se vytvoří automaticky po označení *SFIO* bloků s názvy *Generování* či *Výpočet parametrů*, které má předem nadefinované a nastavené vstupní a výstupní porty dle principu výpočtů (počet portů ad.), a následném spuštění příkazu *codegen* napsaného přímo na příkazový řádek MATLABu. Vygenerované zdrojové kódy jsou poté vloženy přímo do zdrojového souboru *one_phase.c*, kde jsou do jejich těla vepisovány potřebné příkazy či vlastní vytvořené funkce dle konkrétního řešení. Součástí těchto funkcí, reprezentující SFIO bloky, jsou struktury, tj. sdružené datové typy jazyka C, obsahující položky pro datové typy vstupních, výstupních a stavových proměnných.

8.5 BLOKOVÉ SCHÉMA IMPLEMENTACE NA DSP



Obrázek 21: Blokové schéma implementace One phase testu na DSP

Obrázek 21 obsahuje blokové schéma simulace One phase testu, jehož výpočetní algoritmus je prováděn na DSP. Struktura simulace je velmi podobná a vycházející ze simulace, jejíž výpočet je zpracován na straně MATLABu (**Obrázek 20**). Simulační schéma obsahuje některé přidané funkční bloky nesoucí loga společnosti Freescale. Jedná se takřka o identické bloky jako v případě předchozích testů, tj. bloky SFIO toolboxu, které mají za úkol obstarávat komunikaci po sériové lince RS-232 (*SFIO*) mezi DSP a PC a potom také neméně významné bloky, které v sobě nepřímo skrývají naprogramované části zdrojového kódu či spíše funkce výpočetního algoritmu dané identifikační metody (*Výpočet parametrů* a *Generování*). Do schématu jsou rovněž také přidány důležité bloky pro normalizaci fyzikálních veličin pro algoritmus na DSP a případný zpětný normalizační přepočet (*1/scale_i* a *Back_scale*).

8.6 ALGORITMUS ONE PHASE TESTU V JAZYCE C

Hlavní myšlenkou implementace celé simulace je to, že takřka celý výpočetní algoritmus běží na straně DSP. Jedná se zejména o generování napájecího napětí, dále výpočet efektivních hodnot napájecího napětí a výstupního proudu s následným konečným výpočtem identifikovaných parametrů $L_{sl} + L_{rl}$, tj. součet rozptylových indukčností statoru a rotoru. Podstata implementace tedy závisí na vytvoření algoritmu, který bude tyto výpočty řešit. Konkrétní postup a popis algoritmů výpočtů, tedy popis funkcí jednotlivých bloků je popsán jednotlivě v dalších kapitolách. Celá implementace je součástí zdrojového souboru s názvem *one_phase.c*.

8.6.1 Generování napájecího napětí

Generování 3 - fázového napájecího napětí je identické jako v případě implementace No-load či Locked rotor testu na DSP. Opět se jedná o identický zápis a použití funkce v jazyce C, která má jeden vstupní parametr (úhel z bloku *Časový vektor*) a vypočteným výstupem je funkční hodnota dané funkce. Celý vytvořený algoritmus je podrobně popsán v kapitole 6.6.1 a není nutné ho tedy v těchto místech znovu popisovat a rozebírat. Funkce je součástí SFIO bloku pod názvem *Generování* jehož výstupem je vektor 3 napětí o amplitudě 32767 (odpovídá ve skutečnosti 400V), které jsou poté normalizovány na hodnoty použitelné pro napájení bloku AM. Napájecí napětí pro fázi U_a je řešeno výpočtem (viz. vztahy 8.1 až 8.8), za účelem navození situace, kdy právě touto fází teče nulový proud. Výpočet není součástí implementace na DSP, je jen použita výsledná hodnota napětí U_a pro napájení.

8.6.2 Algoritmus výpočtu efektivní hodnoty RMS ze signálu na DSP

K výpočtu parametrů AM je třeba, stejně jako v předešlých No-load či Locked rotor testech, výpočtu efektivních hodnot napětí U_{ef}, proudu I_{ef}. Navíc je tam přidán výpočet efektivní hodnoty ze signálu činného výkonu P_{ef}. Ten je počítán jako určitý integrál ze součinu okamžitého napětí mezi svorkami U_{b-c} a proudu fází I_b. Celý algoritmus obecného výpočtu RMS hodnot s vysvětlením jeho principu a přepsáním do jazyka C je popsán v kapitole 6.6.2. Princip obecného výpočtu efektivních hodnot je uveden v kapitole 6.3.1.

8.6.3 Algoritmus výpočtu parametrů $L_{sl} + L_{rl}$ na DSP

Algoritmus výpočtu parametrů $L_{sl} + L_{rl}$ pomocí implementace One phase testu běžícího na straně DSP je předmětem SFIO bloku pod názvem *Výpočet parametrů* a jeho funkce se stejným názvem, která je součástí souboru *one_phase.c*. Jedná se stejný výpočetní blok mající dva vstupní a dva výstupní porty jako u Locked rotor testu. Prvním vstupem jsou 3 vektory okamžitých hodnot napájecích napětí *u_ABC[3]*, které jsou výstupem z SFIO bloku *Generování*, podobně jako v případě (Obr. 8.1). Druhým vstupem stejného bloku jsou rovněž 3 vektory, ale okamžitých hodnot výstupních proudů *i_ABC[3]* z AM. Jeden výstupní port je vyhrazen pro hodnotu výsledné proměnné *X* symbolizující součet rozptylové induktance statoru a rozptylové induktance rotoru $X_{sl} + X_{rl}$ a druhý pro výsledek součtu odporu statoru a rotoru. Hlavním předpokladem zůstává fakt, že Locked rotor test je určitou modifikací No-load testu použitím nepatrně jiného přístupu. Výpočty parametrů ze známých či vypočtených veličin jsou takřka identické jako v případě No-load či Locked rotor testu a jejich vysvětlení je předmětem kapitoly 7.3. Opět se jedná o stejnou situaci, kdy vytvoření kódu symbolizujícího výpočty ze známých či měřitelných veličin je založeno na přepsání daných výpočetních vztahů do jazyka C. Podstata je identická jako v případě např. No-load testu, kde je vše nastíněno a rozebráno podrobněji.

8.6.4 Algoritmus výpočtu parametrů $R_s + R_r$ na DSP

Podstata určení parametrů $R_s + R_r$ vychází ze vztahu (8.24). Jedná se o prostý výpočet pomocí účiníku $\cos\varphi$, který tvoří právě člen v závorkách. Tento člen je součástí dílčího výpočtu pro určení parametrů $L_{sl} + L_{rl}$. Ve vztahu (8.24) je možné matematickou operací krácení docílit zjednodušeného vztahu.

$$R_s + R_r = \frac{P_{ef}}{3 \cdot I_{ef} \cdot I_{ef}} \tag{8.26}$$

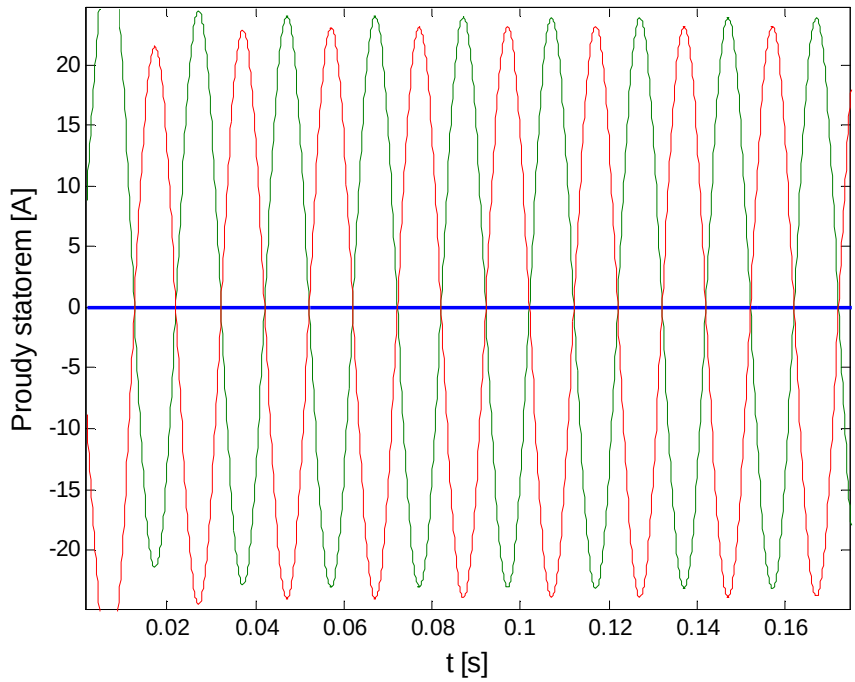
Další možností je nekrátit výraz (8.24) a řešit to pomocí dalších dílčích operací, u kterých je nutno řešit problém se saturací vzniklou při podílu dvou hodnot, z nichž hodnota čitatele je větší než jmenovatele. Problém se saturací je součástí algoritmu výpočtu parametrů $L_{sl} + L_{rl}$.

Konečným krokem je zpětná normalizace veličin pomocí bloku *Back_scale*, který obsahuje vztah $\frac{scale_u}{scale_i \cdot 2^{15}}$.

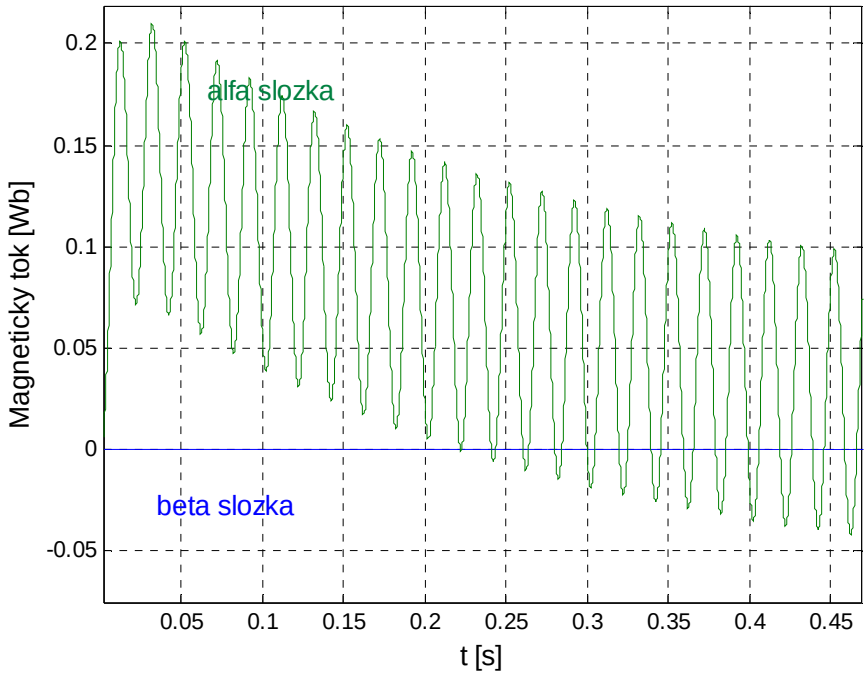
8.7 ZHODNOCENÍ IMPLEMENTACE ONE PHASE TESTU NA DSP

Oba přístupy řešení simulace dané identifikační metody dávají ve své podstatě kvalitní výsledek v podobě identifikovaných parametrů AM pomocí One phase testu, tj. součet rozptylové indukčnosti statoru a rozptylové indukčnosti rotoru $L_{sl} + L_{rl} = 0.01314H$. Součet odporu statoru a rotoru vyšel roven hodnotě $R_s + R_r = 1.577\Omega$. Z hlediska získaných výsledků je dle mého názoru považovat identifikaci oběma způsoby za kvalitní a přesnou vzhledem k řešení.

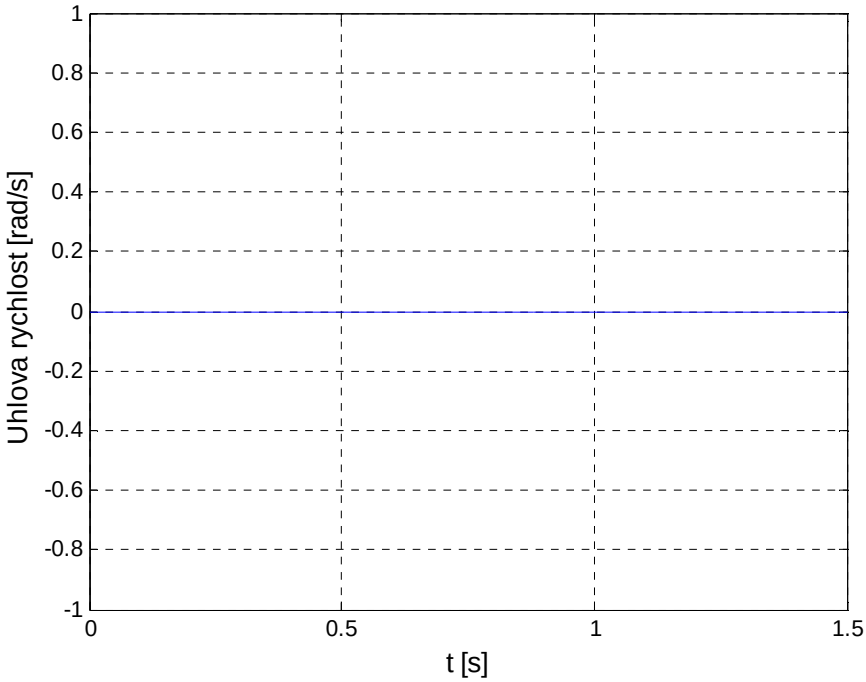
8.8 PRŮBĚHY OKAMŽITÝCH HODNOT VELIČIN V ČASE



Obrázek 22: Okamžité průběhy proudů statorem



Obrázek 23: Okamžité průběhy složek magnetického toku rotorem



Obrázek 24: Okamžitý průběh úhlové rychlosti

Obrázek 22 je důkazem, že je skutečně proud tekoucí jednou z fází statoru nulový.

9. ZÁVĚR

V této diplomové práci jsem se zabýval některými metodami pro identifikaci parametrů jako předpoklady pro dnešní metody řízení asynchronních motorů. V první části jsem se snažil vytvořit ucelený pohled na asynchronní motor jako takový, ale především vytvořit ucelený a srozumitelný přehled o této problematice, i když ne samozřejmě v takové míře a rozsahu. Největší část této práce však tvoří samotné identifikační metody, jejichž výpočetní algoritmus je zpracován na straně výpočetního software MATLAB a dále ve verzi, za kterou se skrývá implementace těchto algoritmů do digitálního signálového procesoru. Vše je zmiňováno přehledně v samostatných kapitolách. Identifikační metody jsou předmětem kapitol 5, 6, 7 a 8.

Výsledky, které každá z identifikačních metod poskytuje, je možné považovat za přesné a kvalitní vzhledem k vlivům, které se při simulacích uplatňují a které se pro zjednodušení čistě zanedbávají. Proto nelze výsledky z metod považovat za směrodatné a rozhodující pro všechny případy, které by byly předmětem identifikace. Je však zřejmé, že implementace identifikačních algoritmů na DSP přiblížila dané identifikační metody vstříc reálnému zařízení. Jako hlavní přínos bych jako autor práce viděl v přehledném a srozumitelném podání identifikačních metod, jejichž funkčnost je ověřena simulačně (výpočetně) v MATLABu či na DSP procesoru.

Na závěr hodnocení bych řekl, že jednotlivé identifikační testy s následnou implementací na DSP, tj. simulace identifikace parametrů, dokázaly svoji plnou funkčnost na modelu zařízení AM. Je obtížné posoudit kvalitu každé z metod, tj. stanovit která z vybraných metod je ta správná a prioritní. Testy či metody zmíněné v této práci přináší výsledky v podobě hodnot parametrů AM, které je možné vzájemně srovnávat. Jak z práce vyplývá, podstatnou její část tvoří kapitola o No-load identifikační metodě tvořící podstatný a důležitý základ pro další zmiňované metody. Proto u dalších metod není podrobně zmiňován princip přepsání algoritmu výpočtu do jazyka C, neboť by práce nabyla značných rozměrů co se týče rozsáhlosti. Implementace ověřila funkčnost metod na skutečném výpočetním prostředku a proces identifikace se tudíž přiblížil skutečné implementaci prováděné na již reálném zařízení.

10. POUŽITÁ LITERATURA

[1] CAHA, Z., ČERNÝ, M. : Elektrické pohony, Praha, SNTL 1990

[2] ŠUBRT, J. : Elektrické střídavé pohony, Vysoké učení technické v Brně FEI, Brno 1999

[3] PROCA, A. : Identification of Variable Frequency Induction Motor Models From Operating Data, IEEE Transactions on Energy Conversion, vol. 17, no. 1, March 2002

[4] GASTLI, A. : Identification of Induction Motor Equivalent Circuit Parameters Using The Single-Phase Test, IEEE Transactions on Energy Conversion, Vol. 14, No. 1, March 1999

[5] SAFFET, A. : Induction Motor Tests Using MATLAB/Simulink and Their Integration Into Undergraduate Electric Machinery Courses, IEEE Transactions on Education, Vol. 48, No. 1, February 2005

[6] HERBERT, S. : Nauč se sám C, SOFTPRESS 2001, ISBN 80-86497-16-X

[7] MACHÁČEK, Z. : Digitální signálové procesory v diagnostice a řízení [online] http://fei1.vsb.cz/wofex/2003/paper/p2645/kybernetika/machacek_zdenek.pdf

[8] TOLIYAT, H. : A review of RFO Induction motor parameter estimation techniques, IEEE Transactions on Energy Conversion, vol. 18, no. 2, June 2003

[9] Katalogový list k vývojové desce Freescale 56F8300DEMO [online] http://www.freescale.com/files/dsp/doc/user_guide/MC56F8300DBUM.pdf?fsrch=1

[10] Katalogový list k SFIO toolboxu