Model Based Design of Electric drives

Jakub Vonkomer [1)], Milan Žalman [2)]

[1)] Institute of Control and Industrial Informatics, FEI STU, Bratislava, Slovakia, e-mail: *jakub.vonkomer@stuba.sk*
[2)] Institute of Control and Industrial Informatics, FEI STU, Bratislava, Slovakia, e-mail: *milan.zalman@stuba.sk*

Abstract— **Electric drives are probably the most widely used electronic devices in the industry. The development process of the drives is very time-intensive, but there are many methods to reduce the development time. In this paper, the development of vector control of induction machine using Simulink is presented. Firstly, the creation of the simulation model is described; then the same model is used for code generation. We used the dSpace platform; however, any platform supported by Matlab/Simulink can be used as well. Finally, the comparison of results between simulation and real motor is presented.**

Keywords— AC drive, vector control, development simulation, Simulink, Code generation, dSpace.

I. INTRODUCTION

The development process of electric drives is itself usually very time-intensive. Simulations can help, but it also takes a lot of time to transform the model to the final code. Manual code writing of the control algorithms introduces a lot of bugs and errors. Some of the bugs even require many hours to be fixed. However, there is no need to do it in this way nowadays, because of the MATLAB code generation possibilities. In our case we used the Real Time Workshop for generating code for dSpace. The generated C code is then compiled and downloaded to the dSpace processor and can be immediately executed. However, there are many platforms supported, manufacturers like *Analog Devices®*, *Atmel®*, *Freescale®*, *Intel®* or *Texas Instruments®* are no exception.

This process of the design and development is called "Model Based Design" and is widely propagated by *Mathworks Inc.*, the creators of Matlab.

II. VECTOR CONTROL

Vector control is a well-known induction motor control method. It allows high-performance control of electromechanical variables like torque, speed, position in both steady and transient states. Main principle of the vector control is the decoupled, independent control of magnetic flux and torque. There are plenty of methods of vector control, but we will focus on the rotor-flux-oriented vector control. The control law can be derived from:

$$M_m = k_m \left| \hat{\Psi}_r \times \hat{i}_s \right| \quad \text{,where } k_m = \frac{3}{2} p' \frac{L_m}{L_r} \qquad (1)$$

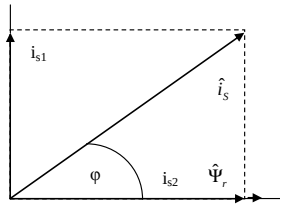

Fig. 1. Vector diagram of stator current and rotor magnetic flux

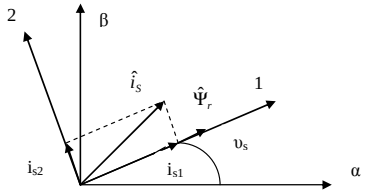

Fig. 2. Vector diagram of the IM state variables

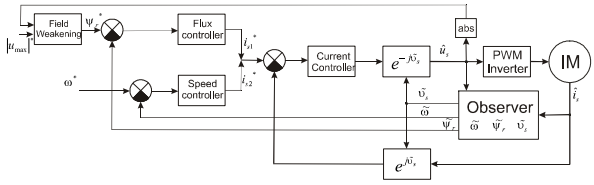

Fig. 3. Simplified flow chart of the vector control

All the controllers and estimators are derived from the mathematical model of the induction machine [1]:

$$\hat{u}_s = R_s \hat{i}_s + \frac{d\hat{\psi}_s}{dt} + j\omega_k \hat{\psi}_s \qquad (2)$$

$$0 = R_r \hat{i}_r + \frac{d\hat{\psi}_r}{dt} + j(\omega_k - \omega)\hat{\psi}_r \qquad (3)$$

$$M_m = \frac{3}{2} p' \frac{L_m}{L_r} \Im\{\hat{i}_s \cdot \psi_r^*\} \qquad (4)$$

$$\omega = p'\omega_m \qquad (5)$$

$$M_m - M_z = J \frac{d\omega_m}{dt} \qquad (6)$$

The structure of all controllers is PI and they are designed by the pole-placement method. All systems are derived from the linearized induction machine model and are described by first-order dynamic equations.

1) Current controllers: Design of current controllers is derived from the following transfer function:

$$G_S(s) = \frac{i_{S1}(s)}{u_{S1}(s)} = \frac{i_{S2}(s)}{u_{S2}(s)} = \frac{1/R_1}{T_1 s + 1} \qquad (7)$$

Final current controller parameters are achieved using the following equations (ω_o is desired dynamics, b desired damping):

$$K_p = \left(2b_1\omega_{01} - \frac{1}{T_1}\right) R_1 \cdot T_1 \qquad T_i = \frac{K_p}{R_1 T_1 \omega_{01}^{\ 2}} \qquad (8)$$

2) Flux controller:

Design of flux controller is derived from the following transfer function:

$$G_S(s)=\frac{\Psi_r(s)}{i_{S1}(s)}=\frac{L_m}{T_r s+1} \tag{9}$$

Final flux controller parameters:

$$K_p=\left(2b_2\omega_{02}-\frac{1}{T_r}\right)\frac{T_r}{L_m} \qquad T_i=\frac{K_p L_m}{T_r\omega_{02}^{\ 2}} \tag{10}$$

3) Speed controller:

Speed controller is designed from the knowledge of the mechanical parameters. Transfer function:

$$G_S(s)=\frac{\omega(s)}{i_{S1}(s)}=\frac{3}{2}\cdot p'.\frac{L_m}{L_r}\psi_r\cdot\frac{1}{Js+B} \tag{11}$$

Final equations:

$$K_p=\frac{(2b_3\omega_{03}J-B)}{\frac{3}{2}\cdot p'.\frac{L_m}{L_r}\psi_r} \quad T_i=\frac{K_p\cdot\frac{3}{2}\cdot p'.\frac{L_m}{L_r}\psi_r}{J\cdot\omega_{03}^{\ 2}} \tag{12}$$

3) Flux weakening controller:

Flux weakening is necessary for achieving speeds higher than rated motor speed . However, the design of the flux weakening controller is out of scope of this paper. In general it is a voltage controller which is activated when the voltage is close to voltage saturation. This controller decreases the flux to reduce the back-emf, allowing high-speed motor operation. Instead of the voltage controller, feed-forward methods can be used as well, but they usually provide lower robustness.

4) Flux estimator:

Flux estimator is the heart of the vector control, because of the control algorithm oriented into rotor magnetic flux reference frame. That is one of the reasons the vector control is often called “Field Oriented Control”. Following equations describe the presented flux estimator (indirect vector control[2][3]).

$$s\hat{\psi}_r=-\frac{1}{T_r}\hat{\psi}_r+j\omega\hat{\psi}_r+\frac{L_m}{T_r}\hat{i}_s \tag{13}$$

$$\tilde{\psi}_r=\frac{L_m}{T_r s+1}i_{s1} \tag{14}$$

$$\tilde{\omega}_{sl}=\frac{L_m}{T_r\tilde{\psi}_r}i_{s2} \tag{15}$$

III. Building the Simulation Model

The aim of the project was to develop the speed vector control of induction machines with the speed feedback from the speed sensor (incremental encoder in our case).

Our primary platform for developing and testing purposes was Simulink with SimPowerSystems toolbox[5]. We used the model of induction machine, three phase power supply and many other power electronic blocks for close-to-reality simulation of the drive [6].

The basic block is the “VECTOR CONTROL” block, which contains all the controllers, observers, state information and error flags. This block will be later copied to the dSpace code generation model only. Figure 4 shows the main window of the simulation SimPowerSystems model.

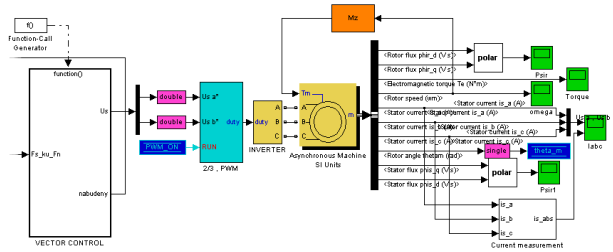

Fig. 4. Screen capture of the simulation model

Figures 5, 6, and 7 show the inner connections of selected blocks. The model is purely discrete with sample time of 200 µs. All the blocks are part of Embedded MATLAB subset[7], so there should be no problem with the code generation. Sample time for simulating the electronics has been chosen as 10 µs, but for a high quality simulation of the hardware as well, lower values are better.

Constant blocks for parameters instead of gain blocks are preferred. Variables inside the constant block may be configured as exported variables as part of global structure or as global variables as well. These variables may be read or written by the user code. In this way, it is possible to change the behavior of the model on the fly.

The entire model is based on IEEE 754 single precision floating point numbers (32 bit), to make this model faster in execution and to allow easy porting to any target with floating point numbers support. Usually, for an application like control of power electronics; the precision of 32 bit floating numbers is pretty enough.

For connecting the signals executed in different subsystems or sampled in different time, *Rate transition* blocks may be used. However, this often involves additional operations for the Simulink to ensure proper sampling of the variables; therefore global variables (*Data Store Memory* blocks) are recommended [7].

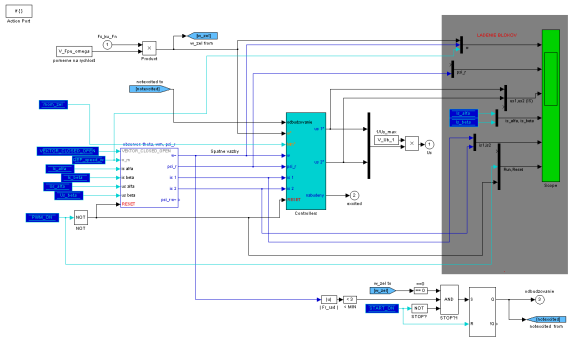

Fig. 5. Screen capture of the VECTOR CONTROL Block

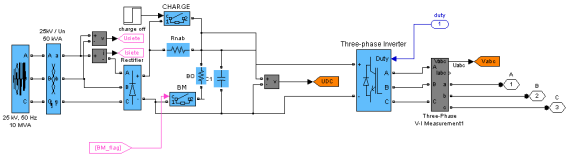

Fig. 6. Screen capture of the inverter block

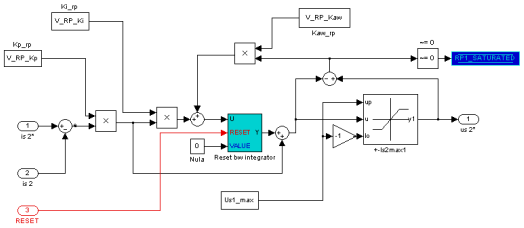

Fig. 7. Screen capture of the current controller block

IV. USING THE MODEL FOR CODE GENERATION

The already tested simulation model can be easily used for code generation purposes. The main block from simulation model is copied to the code generation model. Nevertheless, the main task for the code generation model is communication with the target platform hardware. This task requires special blocks, unique for every platform; therefore it is required to look to the specific documentation. As result, instead of complicated time-intensive rewriting model to the C code, we developed the program in Simulink, added the platform interface, then built and executed the finished model.

We use the laboratory dSpace 1104[4] for testing the model. DSpace board allows real-time control of different systems. Its main advantage is integration into the MATLAB/Simulink environment. Model is executed and controlled via the ControlDesk application. It is also very easy to export any graphs as MATLAB´s MAT files.

As mentioned earlier, constant blocks can be easily tunable, they might be configured as inputs in the ControlDesk applications and its value can be changed on the fly. The same is applied for the code generation for the other platforms. For other platforms, designer of the model can decide whether the variable will be tunable from the user code by selecting the variable from Simulation->Configuration Parameters, Optimization, Configure.

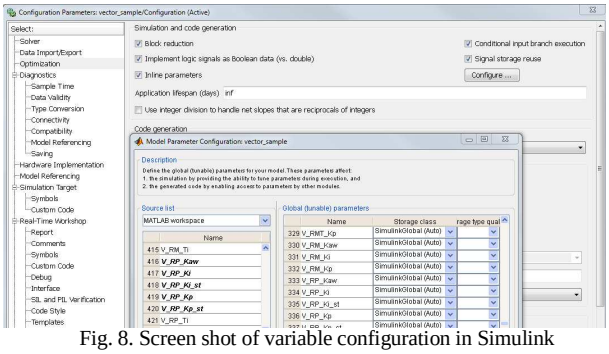

Fig. 8. Screen shot of variable configuration in Simulink

By using global variables defined by *Data Store Memory* blocks, several advantages are gained. These variables can be easily read (and written) in the custom code; they always have the same name as defined in Simulink. However, in contrast to signals transferred over *Rate transition* blocks, no data integrity is guaranteed so data might be for example written three times and read just once, so some information can be easily lost. However, the behavior of data integrity can be detected; it depends on the configuration of the model.

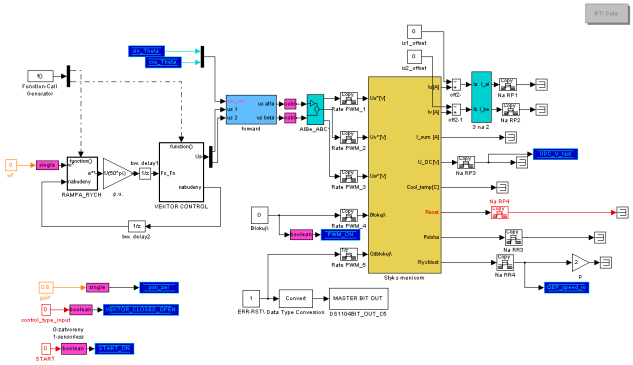

Fig. 9. Screen capture of the proposed dSpace model

Models for both simulation and code generation are pretty similar as can be seen in figure 9. Instead of induction machine and inverter blocks, the block of communication with an inverter is used. Inputs represent reference voltages. The communication block with inverter contains the compensation of DC voltage fluctuations and the computed duty cycles are inputs for the dSpace PWM block.

Figure 10 shows the basic schema of the dSpace controller board to the frequency converter connection.

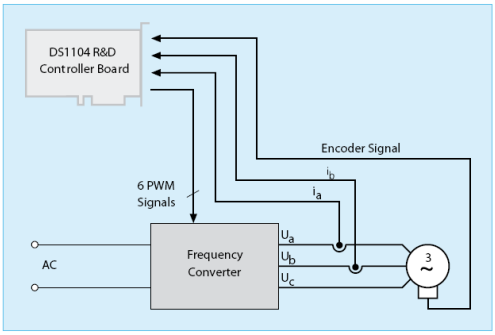

Fig. 10. Schema of connecting dSpace equipped PC with frequency inverter and induction motor

V. COMPARISON

Figures 11-14 show the comparison of simulation and real behavior of the modeled system. Wide range of speed is demonstrated. Small differences and noise in dSpace figures are caused by non-calibrated current sensors, but indeed we can say that we reached comparative results within the toleration.

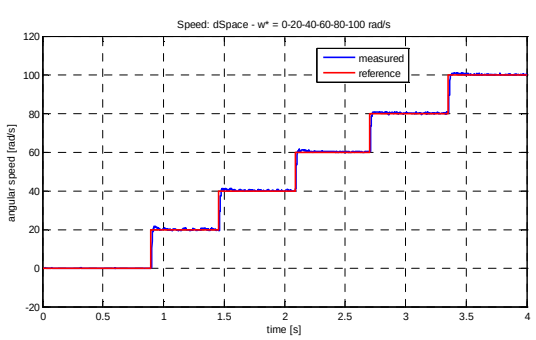

Fig. 11. dSpace experiment: steps of the reference speed up to 100 rad/s

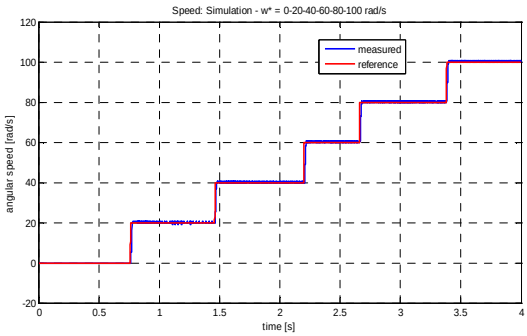

Fig. 12. Simulation: steps of the reference speed up to 100 rad/s

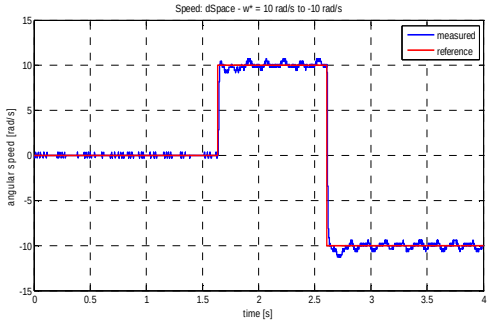

Fig. 13. dSpace experiment: performance of low speed and speed reversal

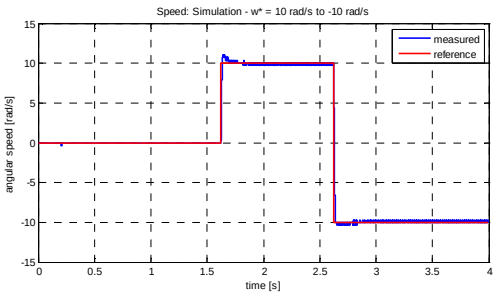

Fig. 14. Simulation: performance of low speed and speed reversal

VI. Conclusion

In this paper we described several aspects regarding the induction machine vector control development. Since the beginning we tried to keep the model simple, robust and fully discrete, because we planned to share the basic model with other platform for code generation – dSpace in our case. The basic testing during the creation and development of the model was done by simulation only on a dSpace platform where real capabilities were demonstrated.

This development process called "Model based design" can be used for any control structure with many advantages. Creating the model and testing in Simulink greatly saves time and other resources. This leads to reduced development time, reduced number of bugs and increased user-friendliness of the whole development process. This paper is an updated version of the paper published in 2009[8].

VII. Appendix 1

Description of symbols and acronyms used in this paper

Symbol	Description
ω_m	motor angular speed
ω_{sl}	slip speed
p'	number of pole pairs
σL_s	stator leakage inductance
L_s	stator winding inductance
L_r	rotor winding inductance
L_m	mutual inductance
R_s	stator resistance
R_r	rotor resistance
R_1	combined resistance ($R_1=R_s+L_m^2/L_r^{2*}R_r$)
T_1	current model time constant ($T_1=\sigma L_s/R_1$)
T_r	rotor time constant ($T_r=L_r/R_r$)
J	moment of inertia
M_m	motor torque
M_z	load torque
$\hat{i}_s$	stator current vector
$\hat{u}_s$	stator voltage vector
$\hat{\psi}_r$	rotor flux vector

Acknowledgment

This work was supported by the Slovak Research and Development Agency under the contract No. VMSP-II-0015-09.

References

[1] Žalman M.: "Akčné členy (Slovak language)", STU Bratislava 2003

[2] Doki, S.; Kinpara, Y.; Okuma, S.; Sangwongwanich, S., "Unified interpretation of indirect and direct vector control [of electric machines]," Power Conversion Conference, 1993. Yokohama 1993., Conference Record of the , vol., no., pp.297-302, 19-21 Apr 1993,URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=264167&isnumber=6639

[3] Marwali, M.N.; Keyhani, A.; Tjanaka, W., "Implementation of indirect vector control on an integrated digital signal processor-based system," Energy conversion, ieee transactions on , vol.14, no.2, pp.139-146, Jun 1999 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=766961&isnumber=16622

[4] DS1104 R&D Controller Board, dSpace Inc. 2009 Catalogue, URL:http://www.dspaceinc.com/ww/en/inc/home/medien/papers/download_page.cfm?FileID=778

[5] The MathWorks, SimPowerSystems Documentation, 2011 URL: http://www.mathworks.com/access/helpdesk/help/toolbox/physmod/powersys/index.html?/access/helpdesk/help/toolbox/physmod/powersys/

[6] The MathWorks, SimPowerSystems Demos

[7] The MathWorks, Code Generation from MATLAB®, User's Guide, R2012b URL: http://www.mathworks.com/help/pdf_doc/eml/eml_ug.pdf

[8] Vonkomer, J.,Radičová T., Žalman M., Suchánek M. "Fast AC Electric Drive Development Process Using Simulink Code Generation Possibilities", Technical Computing Prague 2009 : 17th Annual Conference Proceedings. Prague, Czech Republic, 19.11.2009. - Prague: Humusoft, Ltd., 2009. – ISBN:978-80-7080-733-0. - CD-Rom