Module Library for Rapid Prototyping and Hardware Implementation of Vector Control Systems

József Vásárhelyi
Department of Automation
University of Miskolc
H-3515 Miskolc Egyetemváros
Hungary
vajo@mazsola.iit.uni-miskolc.hu

Mária Imecs
Department of Electric Drives and Robots
Technical University of Cluj
3400 Cluj-Napoca, PO.1, Box 99
Romania
Maria.Imecs@edr.utcluj.ro

János J. Incze
Department of Electric Drives and Robots
Technical University of Cluj
3400 Cluj-Napoca, PO.1, Box 99
Romania
Incze@edr.utcluj.ro

Csaba Szabó
Department of Electric Drives and Robots
Technical University of Cluj
3400 Cluj-Napoca, PO.1, Box 99
Romania
Csaba.Szabo@edr.utcluj.ro

***Abstract* – The paper focuses on the implementation of a module library for vector control systems of AC drive. The rapid prototyping and fast implementation of vector control systems becomes possible with the created module library. The control system structures are implemented in configurable logic cells using Field Programmable Gate Array (FPGA). The performances of the created control structures were compared with other simulation results.**

I. INTRODUCTION

Most motor control applications concern with vector control for AC drives. Vector control systems for induction motors give the best dynamic behaviour. Analysing these systems some modularity can be observed, which help fast implementation of motor control applications in reconfigurable structures [3], [10].

Reconfigurable hardware was used in vector control in the last years for control system implementations. We speak about dynamic reconfiguration of vector control systems for AC drives when the real-time application (software) changes the computing platform structure (hardware). In vector control systems, the reconfigurability was introduced by Imecs et all. in [1]. In this concept, each configuration is considered as a state of a logic state machine. When reconfiguration condition occurs, the system will start reconfiguration process in which it switches the current configuration to the next corresponding one. This type of configuration is the context switching and was developed by Sanders in [6]. While context switching is a reconfiguration technology for Field Programmable Gate Arrays (FPGA), the logic state machine with different control system structure in each state is a reconfiguration method for vector control systems presented in the following figure:

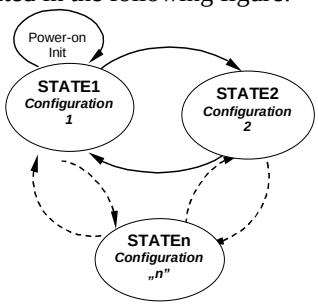

Fig. 1. State machine with different vector control structures in each state.

In order to make the reconfiguration possible, there is a need a close analysis of the known control structures. Kelemen and Imecs in [5] presented most of the known control structures for AC drive. A part of the analysis is presented in the next section.

II. VECTOR CONTROL SCHEME ANALYSIS

The dynamic behaviour of the AC machines is improved by vector control procedures based on the field-orientation principle. The necessity of reconfiguration is based on the observation that the performance of the drive is depending on the vector control structure correlated with the type of the supply power frequency converter [9], [10].

The analysis of the control schemes was performed based on the following criteria:

- Given two vector control structures when common modules exist:
  - Which are the common modules in the same position with the same function?
  - Which are the common modules with different functionality?
  - Which are the particular modules of each reconfigurable structure?
- When reconfiguration condition occurs, is it possible the parameter transfer for the modules on the same position or no parameter transfer allowed?
- Is the parameter transfer of the PI controllers of the different schemes possible?
- Is it possible to give a general mathematical form of all the modules?
- Resulting from the analyses, the module library should be universal for rapid prototyping of any vector control system and from the prototype the implementation should directly result

Let us analyse such a reconfigurable vector control structure for AC drives with two configurable states. This vector control structure presents the generalities of the most common control schemes and in the meantime contains some particular modules. The reconfigurable state machine presented in Fig. 1 for the vector control structure presented in Fig. 2 is working in ***state 1*** as a *tandem converter* [7], [8], [9]. *The tandem converter* is working with two inverters. The two inverters are: Current Source Inverter (CSI), which is controlled in current, and the other the Voltage Source Inverter (VSI) is controlled in voltage.

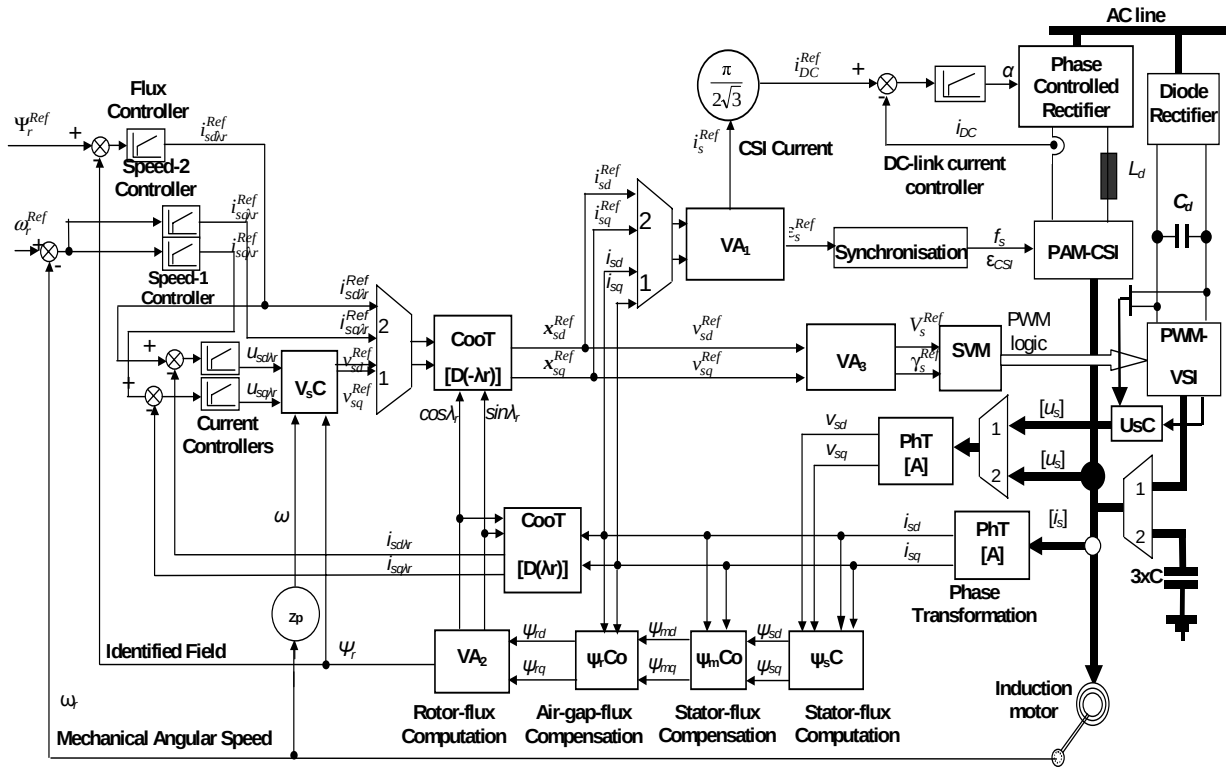

Fig. 2. Reconfigurable vector control structure with two configuration states.

In this way the VSI controlled by Pulse-Width Modulation (PWM) supplies the reactive power required to improve the quality of the motor currents in order to compensate them into sine wave form.

For the case of the logic state machine in ***state 2***, only the current source inverter is working, the inverter currents are synchronized with respect to the control variables of the stator-current d-q components. The Transition from ***state 1*** to ***state 2*** is made when the Voltage Source Inverter fails and the control structure need to be reconfigured in order to be able to control the AC drive with the Current Source Inverter. ***State 3*** (represented in Fig. 1) could be another configuration of the vector control system when the CSI fails and only the Voltage Source Inverter has to control the motor.

A possible representation of the transition from one state to the other, in fact, may be a demultiplexer and a multiplexer, but one should note that, these components might be carried out in reality, while they are intended to be abstract entities, which did not need any implementation at all [3].

One may observe in the module analysis, that the flux computation modules are common in both control schemes and so they will be one of the most used modules in the module library. As the three flux-computation modules can be computed in a single equation, they will represent a single module, with the following equation:

$$\Psi_{rd} = (1+\sigma_r)\int (u_{sd} - R_s i_{sd})dt - [(1+\sigma_r)L_{\sigma s} + L_{\sigma r}]i_{sd} \qquad (1.a)$$

$$\Psi_{rq} = (1+\sigma_r)\int (u_{sq} - R_s i_{sq})dt - [(1+\sigma_r)L_{\sigma s} + L_{\sigma r}]i_{sq} \qquad (1.b)$$

In this way in the module library for the flux computation we will have one module, but when it is needed can easily be created all the modules needed for the flux computation ($\Psi_{sd,q}$, $\Psi_{md.q}$, $\Psi_{rd,q}$) separating the stator field computation ($\Psi_{sd,q}$) and the flux compensation modules ($\Psi_{md,q}$, $\Psi_{rd,q}$). In addition, the library can handle both flux oriented control schemes, such as rotor flux oriented and stator flux oriented vector control system for any converter fed AC drive.

One of the most common modules (often-used modules) is the Vector Analyser (VA), and as is illustrated in Fig. 2. It is used to compute different modulus of different parameters. Its equations given in the general forms as follows:

$$g = \sqrt{g_d^2 + g_q^2}; \quad \sin\lambda = \frac{g_q}{g}; \quad \cos\lambda = \frac{g_d}{g} \qquad (2.)$$

The other two modules, which are common in many control structures, are the Coordinate transformation modules (CooT[D(λ)], CooT[D(-λ)]) with the general equation:

$$g_{sd\lambda r} = g_{sd}\cos\lambda_r \pm g_{sq}\sin\lambda_r; \qquad (3.a)$$

$$g_{sq\lambda r} = g_{sq}\cos\lambda_r \mp g_{sd}\sin\lambda_r; \qquad (3.b)$$

One should mention also the other common modules such as the Phase Transformation module (PhT[A] and PhT[A]$^{-1}$ - for current and voltage). These modules again are general computation modules.

For all the mentioned modules when reconfiguration occurs there is no need for parameter transfer as they do all the computation for the actual sampled values.

The modules where one have to consider the parameter transfer is for example the so called control strategy block, represented in this case by the *PI controllers of speed, current and torque*. These modules are called together "control strategy block", as they can be realised in many ways. The control strategy can be implemented using fuzzy

logic, neural networks, or other intelligent control methods.

The most critical part of the reconfiguration is the parameter transfer of the PI controllers. In the case when (as in Fig. 2) the output variables of the controllers are different in each state (in one case this is the current reference $i_{sd,q\lambda r}^{Ref}$, and on the other case is the voltage reference $v_{sd,q\lambda r}^{Ref}$), the parameter transfer cannot be solved. This explains why the reconfiguration method applied here is context switching.

From the analysis, results, that a module library can be created for fast modelling. The modularity is important when the implementation target is reconfigurable hardware such as Field Programmable Gate Arrays (FPGAs) or Configurable System on a Chip (CSoC) [10].

III. MODULE LIBRARY CHARACTERISTICS

The creation of a module library was motivated by the fact that the simulation of the reconfiguration process it is not possible or it is difficult while no tools exist for this kind of simulation. On the other hand recently it has become possible to implement digital signal processing algorithms on FPGAs directly from Matlab Simulink® environment.

This possibility gave the idea to implement the mentioned module library, which is completely parametrisable and any change on the vector control system's structure can be applied very fast and easy in the implementation hardware.

The elements of the library are the most common modules of vector control systems (as described in the previous paragraph), and each present a standalone unit in the library. As result of this independency, the vector control system can be synthesised module by module or as a whole.

Most of vector control system implementations use 16 bit two's complement fixed-point data format. Here this format was also adopted for the input variables of each module. Inside the module for constant representation it was adopted the same data format, but the binary point has variable position, depending on the motor parameters.

The major advantage of using the module library (when implementation is targeted) is: the computation speed increase. This results from the parallel algorithm computation of both components (d, q) and the parallel computation of each module. This would be a significant advantage compared to the DSP sequential implementations.

IV. SIMULATION AND RAPID PROTOTYPING WITH MODULE LIBRARY

Theoretically with the created module library, any vector control system can be tested, simulated, and implemented. Using the module library in this way a vector control system can be implemented in short time based on reconfigurable hardware.

The motor data used for simulation are: 5.5 *kW*, 50 *Hz*, 220 *V r.m.s.*, 14 *A r.m.s.* and *4 pole-pairs*. The simulation was performed for the presented vector control system structures as follows:

- First: The configuration of State 1 – CSI+VSI driven AC drive was simulated, then the configuration of Sate 2 – CSI driven AC drive was simulated.
- Second: The simulation performed for the reconfiguration process. The motor started in State 1 and after 0.5s was reconfigured to State 2.

The results compared with simulation results produced by the simulation model done with Simulink models. One can conclude that some parameters are working better with the module library (for example the PI implementation), but in some cases, the quantisation errors were not satisfactory against our expectancies.

The following diagrams show the simulation results for the running motor and reconfiguration applied after start at time 0.5s.

Fig. 3 shows the stator current waveform resultant as the sum of the output current of the CSI inverter and the output current of the VSI inverter. Also the figure shows that after reconfiguration the stator current results as the sum of CSI output currents and the capacitor currents. Fig. 4 to Fig. 6 represents several space-phasors of the output currents, CSI and VSI space-phasors. Fig. 7 and Fig. 8 represents the computed rotor-flux space-phasor and the resultant stator-flux space-phasor. While the resultant stator flux and computed rotor flux is represented in Fig. 10. Fig. 9 shows the resultant stator-terminal-voltage space-phasor. The reconfiguration of the control structure (i.e when the VSI fails and the CSI will work alone) it is observable in all the figures. The reconfiguration effects are observable also in the motor parameters (speed and torque) as shown in Fig. 11and Fig. 12.

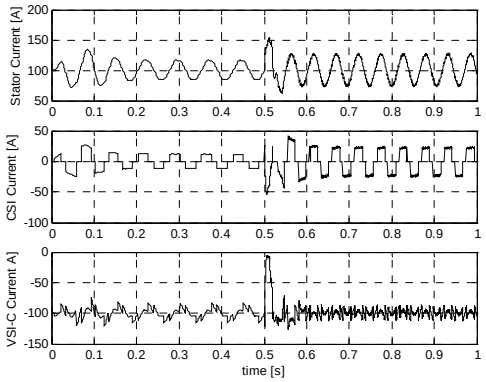

Fig. 3. Current waveforms before and after reconfiguration.

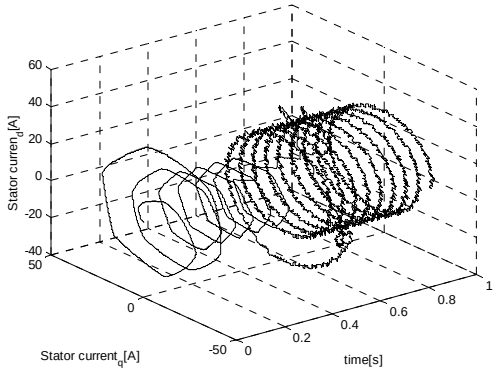

Fig. 4. Motor Stator-current space-phasor.

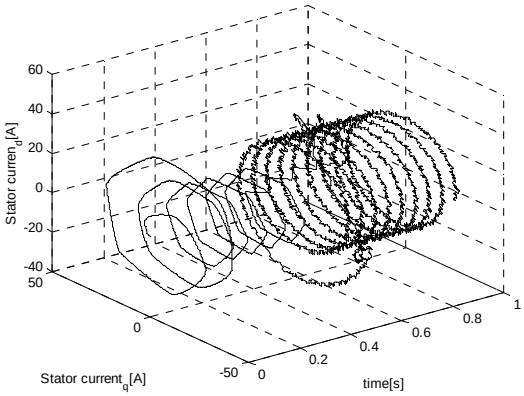

Fig. 5. Current-Source Inverter output current space-phasor.

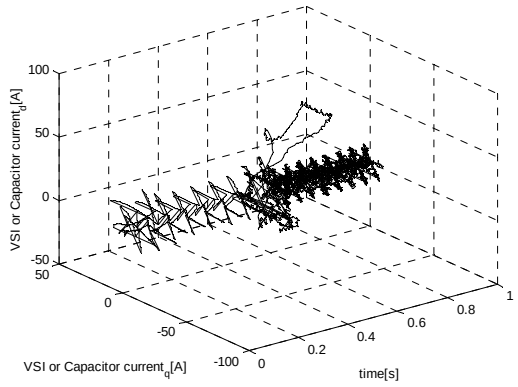

Fig. 6. VSI or Capacitor output-current space phasor.

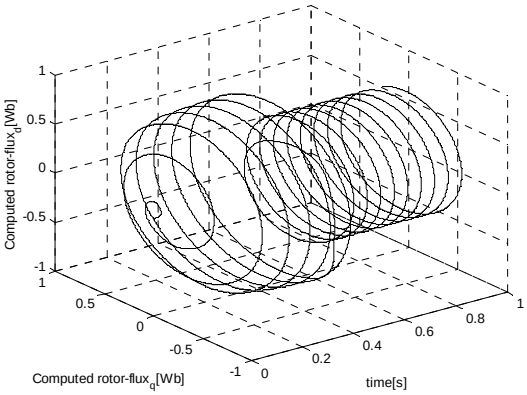

Fig. 7. Computed rotor-flux space-phasor.

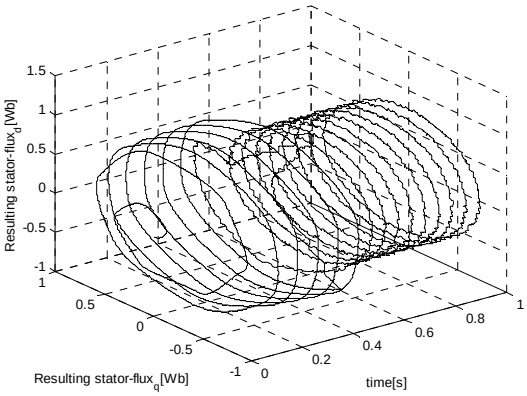

Fig. 8. Resulting stator-flux space-phasor.

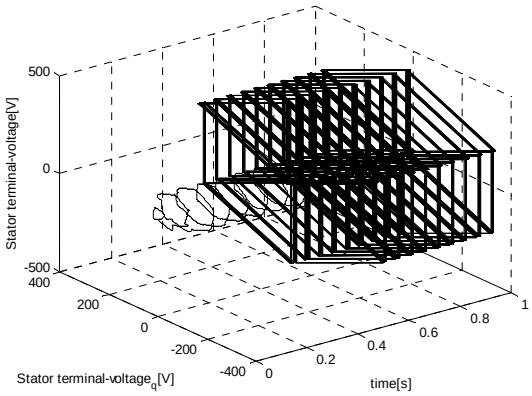

Fig. 9. Resultant Stator-terminal-voltage space-phasor.

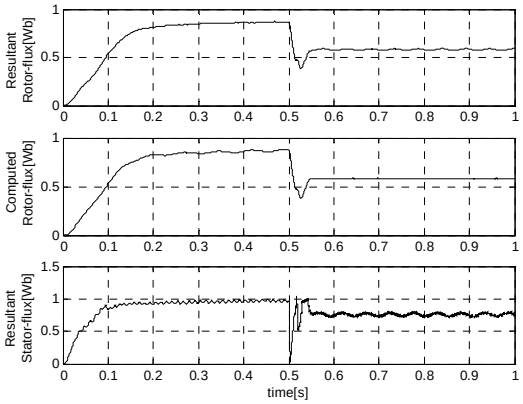

Fig. 10. Resultant, Computed rotor-flux and resultant stator-flux.

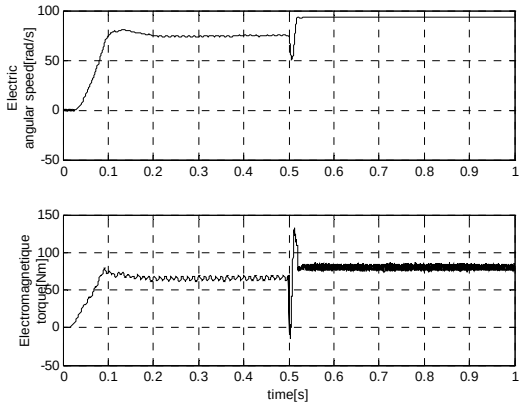

Fig. 11. Electric angular speed and electromagnetic torque.

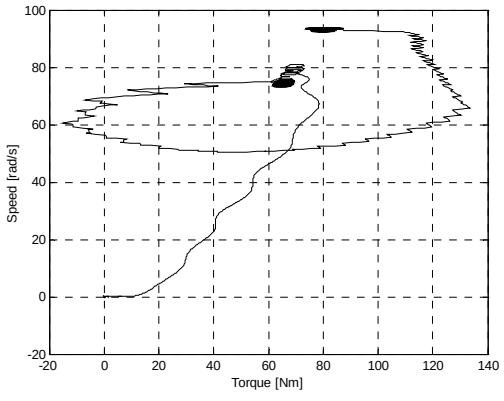

Fig. 12. Resultant dynamic speed-torque mechanical diagram.

The method used for reconfiguration was the context switching method previously named ping-pong [2]. In this case, there is no need for parameter transfer at all, as both vector control systems are working in parallel and all the modules also are working in parallel. This allows us to exploit all the parallelism of the vector control algorithm and the implementation possibilities in FPGA.

V. IMPLEMENTATION POSIBILITIES

Comparing the performances of implemented modules, there one have to consider the following: the evolution of the module computation compared to the model, the quantisation error produced by the module, the time delay introduced by the module, the hardware resources occupied when the module is implemented. On the following we analyse some modules considering the mentioned criteria.

For a simplified analysis of the modules the simulation for the vector control scheme in Fig. 2 was performed without reconfiguration and for the tandem converter structure, which corresponds for selection 1 of the multiplexer inputs. The simulation performed for 1 second and at 0,35s it is observable a speed inversion.

Fig. 13, **Hiba! A hivatkozási forrás nem található.** and Fig. 14, represents the the d-component evolution of the Flux Controller and inverse coordinate transformation CooT[D(-λ)] respectively. The output q component of the module CooT[D(-λ)] is also presented in Fig. 15. As result from the figures (Fig. 13 -Fig. 14), there is no significant difference between the simulation and the output variable resultant from the library module computation. The quantisation error of the computed variables are minimal excepting the module CooT[D(-λ)], where the quantisation error of the reference voltage Usd is between –5 and +5. Even under these circumstances the results are promising.

The time delay and the hardware resource consumed by the analysed modules are presented in TABLE 1

TABLE 1.

HARDWARE RESOURCES CONSUMED AND TIME DELEY INTRODUCED BY THE MODULE FLUX CONTROLLER

Release 4.1.03i - Map E.33
Xilinx Mapping Report File for Design
Design Information

Command Line: map -p xc2v40-cs144-6 -cm area -pr b -k 4 -c 100 -tx off
Target Device: x2v40
Target Package: cs144
Target Speed: -6
Mapped Date: Tue Mar 26 15:16:39 2002
Design Summary

Number of Slices:	24 out	of	256	9%
Number of Slices containing unrelated logic:		0 out of	24	0%
Total Number 4 input LUTs:		24 out of	512	4%
Number used as Shift registers:		24		
IOB Flip Flops:		24		
Number of GCLKs:		1 out of	16	6%
Total equivalent gate count for design:		5,731		

The Average Connection Delay for this design is:	1.283 ns
The Maximum Pin Delay is:	4.126 ns
The Average Connection Delay on the 10 Worst Nets is:	1.614 ns

Listing Pin Delays by value: (ns)

d < 1.00	< d < 2.00	< d < 3.00	< d < 4.00	< d < 5.00	d >= 5.00
178	68	22	9	1	0

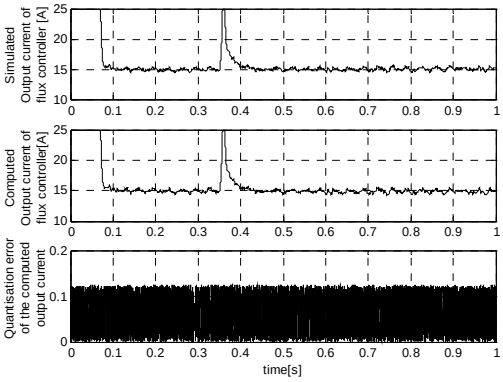

Fig. 13. Outputs of the modelled and implemented flux PI controller and the resultant quantisation error.

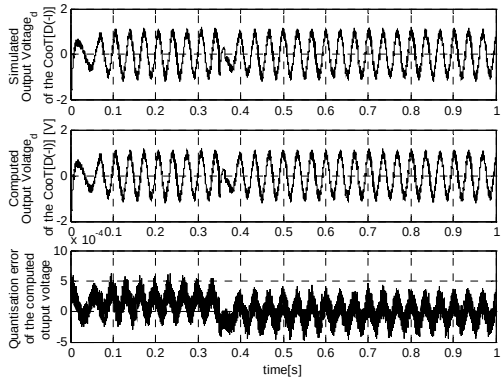

Fig. 14. Output Voltage Reference d component of the module Inverse coordinate transformation CooT[D(-λ)].

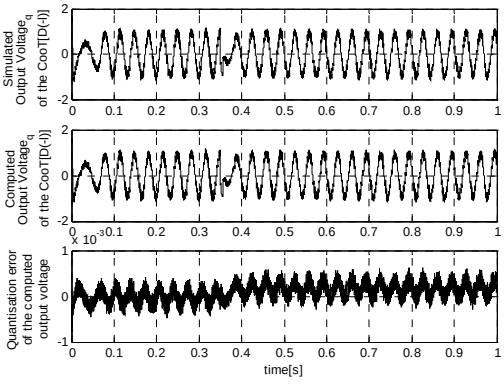

Fig. 15. Output Voltage Reference q component of the module Inverse coordinate transformation CooT[D(-λ)].

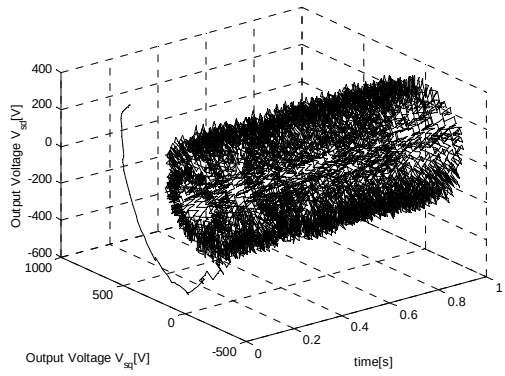

Fig. 16. Computed Stator-terminal-reference voltage space-phasor of module CooT[D(-λ)]

TABLE 2.
HARDWARE RESOURCES CONSUMED AND TIME DELEY INTRODUCED BY THE MODULE CooT[D(-λ)]

Release 4.1.03i - Map E.33
Xilinx Mapping Report File for Design
Design Information

Number of Slices:	25 out of	3,072	20%
Number of Slices containing unrelated logic:	0 out of	625	0%
Total Number 4 input LUTs:	1,222 out of	6,144	19%
Number used as LUTs:	1,208		
Number used as a route-thru:	14		
Total equivalent gate count for design:	15,579		

The Delay Summary Report
The Score for this design is: 5342
The Average Connection Delay for this design is: 1.969 ns
The Maximum Pin Delay is: 10.256 ns
The Average Connection Delay on the 10 Worst Nets is: 7.306 ns
Listing Pin Delays by value: (ns)

d<2.00	2.00< d<4.00	4.00 < d<6.00	6.00 < d< 8.00	8.00 < d < 11.00	d >=11.00
2432	1211	395	92	6	0

As observed from the tables the hardware resources consumed by the modules flux controller and inverse coordinate transformation are significant, and this may be a disadvantage of the developed module library, while the time delay introduced by the module is a positive result, which have to be considered when computation speed is important.

VI. CONCLUSIONS

The created module library like other Matlab® tools, helps the development of rapid model based vector control systems for AC drives. The module parameters are freely modifiable on demand. It allows the simulation of the reconfiguration process and effects on the AC drive.

VII. AKNOWLEDGMENT

A research project in the subject of the "*tandem inverter*" was realized at the Institute of Energy Technology, Aalborg University, Denmark. Special thanks to Prof. A. Trzynadlowski from Nevada University, Reno, USA for the collaboration in this theme, to Prof. F. Blaabjerg from the Aalborg University and to the Danfoss Drive A/S, Denmark for their generous support.

The authors are grateful to Triscend Inc. and Xilinx Inc. for donations, which made possible the research on some aspects of reconfigurable vector control framework.

VIII. REFERENCES

[1] Mária Imecs, P. Bikfalvi, S. Nedevschi, J. Vásárhelyi "Implementation of a Configurable Controller for an AC Drive Control a Case Study", *Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines FCCM 2000*, Napa, California, USA, 2000, pp. 323-324.

[2] Mária Imecs, J. J. Incze, J. Vásárhelyi, Cs. Szabó "Vector Control Of Tandem Converter Fed Induction Motor Drive Using Configurable System On A Chip", *INES 2001 IEEE International Conference on Intelligent Engineering Systems*, September 16-18, 2001, Helsinki-Stockholm-Helsinki, Finland-Sweden, pp. 489-495.

[3] Mária Imecs, J. Vásárhelyi, J. J. Incze, Cs. Szabó, "Tandem Converter Fed Induction Motor Drive Controlled With Re-Configurable Vector Control System", *Power Electronics Intelligent Motion Power Quality Conference PCIM 2001*, Nürnberg, Germany, pp. 341-346

[4] Mária Imecs "Open-Loop Voltage-Controlled PWM Procedures", *ELECTROMOTION'99*, Patras, Greece, Vol. I, pp. 285-290

[5] Á. Kelemen., Mária Imecs "*Vector control of AC Drives"*, OMIKK publisher Budapest, ISBN 963-593-140-9, Budapest, 1991, pp.304.

[6] A. Sanders "The Design and Implementation of Context Switching FPGA", IEEE Symposium on FPGAs for Custom Computing Machines FCCM 1998, Los Alamitos California, USA, April 15-17, 1998, pp. 78-85.

[7] A. M. Trzynadlowski, F. Blaabjerg, J. K. Pedersen, Niculina Patriciu "The Tandem Inverter: Combining the Advantages of Voltage-Source and Current-Source Inverters", *Applied Power Electronics Conference, APEC'98*, Anaheim, USA, pp.315-320.

[8] A. M. Trzynadlowski, Mária Imecs, Niculina Patriciu "Modelling and Simulation of inverter Topologies Used in AC Drives: Comparison and Validation of Models", *ELECTRIMACS'99*, Volume I/3, Lisboa, Portugal, 1999, pp. 47-52.

[9] J. Vásárhelyi, Mária Imecs, J.J. Incze "Run-time Reconfiguration of Tandem Inverter used in Induction Motor Drives", *Proceedings of Symposium on Intelligent Systems in Control and Measurement*, Veszprém, Hungary, 2000, pp. 138-143.

[10] J. Vásárhelyi *"Run-Time Reconfiguration of AC Drive Controllers", Dagstuhl Seminar 0261 on Dynamically Reconfigurable Architectures*, June, 2000, Germany, http://www.ee.qub.ac.uk/dsp/HsD/fpl/.