METHODOLOGY AND TOOLS USED IN DESIGNING AND IMPLEMENTING SOFTWARE APPLICATIONS FOR ELECTRIC DRIVES CONTROL

C. Ilaş; V. Bostan; M. Cuibuş

Department of Electrical Engineering, University "POLITEHNICA" Bucharest
Phone: +40-1-4029753; E-mail: costel@amotion.pub.ro

Abstract: *This paper presents the main development tools that can be used in software applications for digital control of electric drives. It is presented a general three steps work methodology for algorithms development and testing. Each step contains examples in the area of AC drives control, specifying the main programs and tools that were used.*

1. INTRODUCTION

Today the digital control is the preferred solution in most of the electric drives systems. This is because of the easiness of implementing the most recent and modern control methods, adaptive algorithms, parameter identification, optimal control as well as because of its flexibility and stability in operation. But the digital control involves supplementary software and hardware knowledge and the necessary time for designing such applications can be quite long. The optimisation of these criteria can be done respecting a certain work methodology, passing successively through three main steps: simulation, laboratory experimental testing and optimisation for industrial implementation.

2. TESTING THE CONTROL ALGORITHMS USING SIMULATION

First the control method to be used must be identified. This depends on the desired performance and the motor type. The next step is testing the control algorithm by simulation. The main goal of this step is to verify the correctness of the control algorithm. Then, in order to obtain an optimal variant for practical implementation, we can tune it based on the obtained performances and involved resources. The Matlab-Simulink software offers enough facilities in this direction and it is very used in present.
This stage has some main steps, synthesised in figure 1:

- Getting the control system parameters. In a Matlab file, the system data, and initialisation variables are introduced and then the coefficients of the different parts of the algorithm are computed;
- Control diagram design. At this step the control diagram is drawn in Simulink and the program associated to each used Matlab or C function is written. It is recommended to use Matlab/ Simulink functions in order to improve simulation speed;
- Control algorithm verification. This is done by simulating the designed control diagram. In most of the cases it is necessary to adjust the characteristic coefficients of the diagram or/and of certain components.

The entire process end when the obtained results are those requested by project topic.

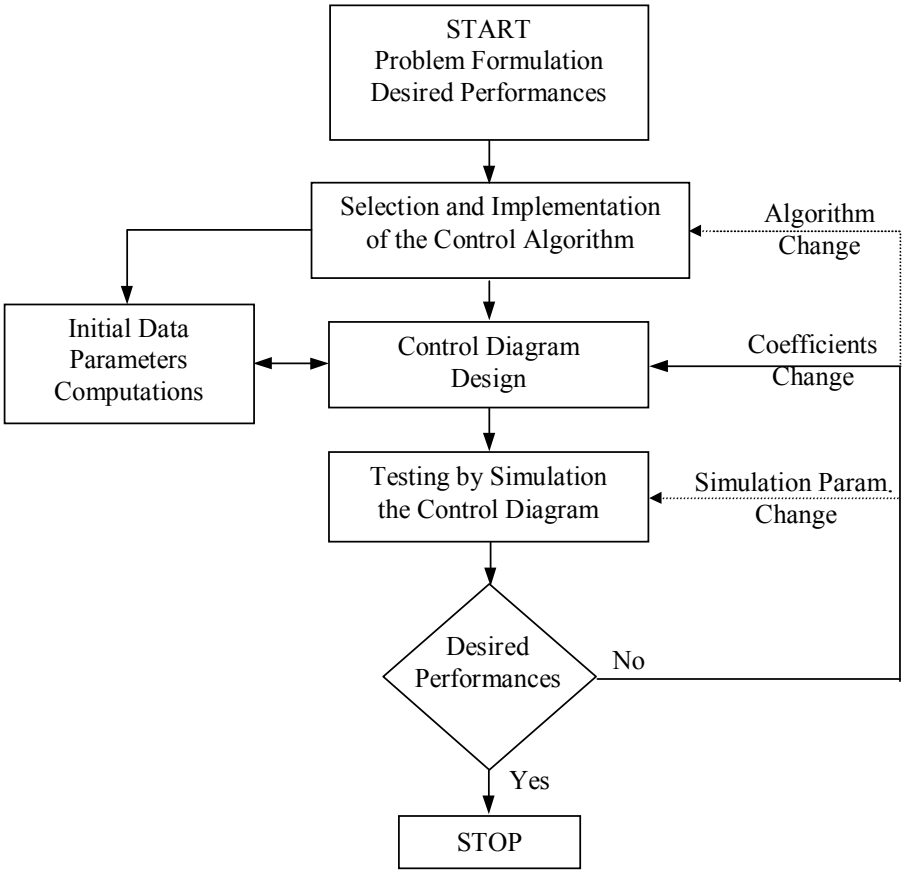


Fig. 1. *Testing by simulation the control algorithms*

In figure 2 is presented, as an example, the simulation diagram for induction motor sensorless MRAS control based on Luenberger observer. In figure 3 a part of the obtained results with the motor functioning at rated speed are presented.

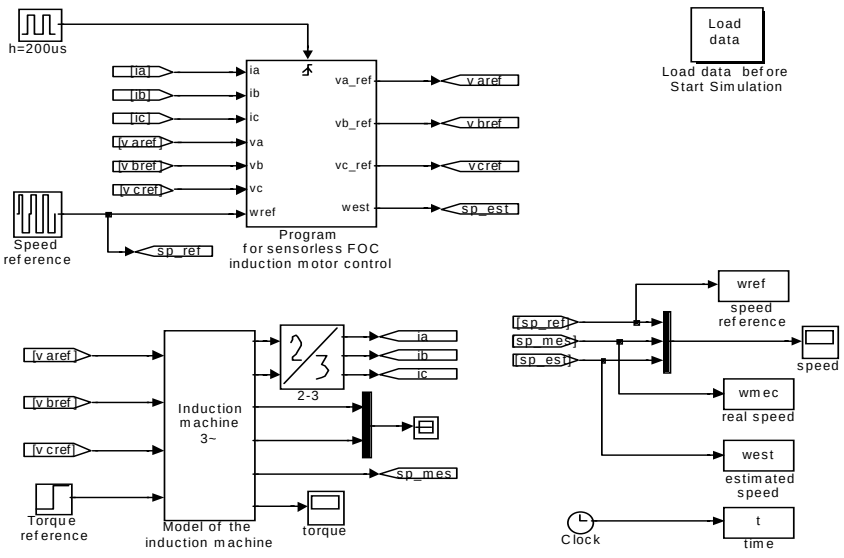
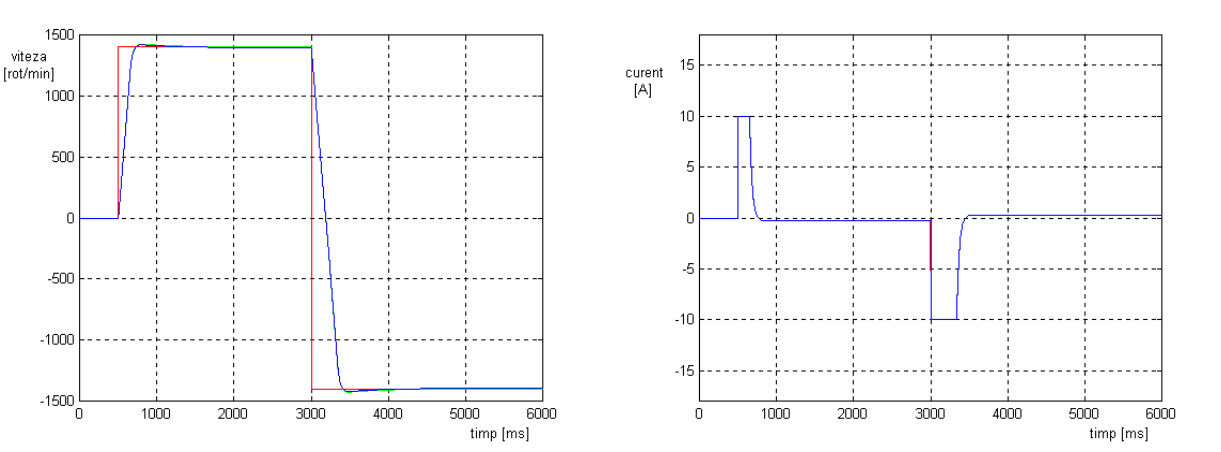


Fig. 2. *Sensorless MRAS Induction Motor Control diagram used in Matlab-Simulink simulations*



a) b)

Fig. 3. *Induction Motor Sensorless MRAS Control results: a) rotor speed (reference-red, estimated-green, measured-blue); b) active component of the stator current reference-red, estimated - blue).*

In this example we used „triggered" functions which allow a precise modelling of the discrete control systems [4]. Another advantage of this method is that the Matlab program used in simulations can easily be transformed in a high level language, usually ***C***, recognised by the digital control systems compilators.

3. EXPERIMENTAL LABORATORY TESTING OF THE CONTROL ALGORITHM

Once we obtained the desired results in simulations, we can pass on testing experimentally the control diagram. At this step it is ideal to have a high performance control system and very easy to use in order to avoid supplementary physically or computationally limits. Below we mention some of the requirements imposed to a digital control system:

- the computation power of the processor must be high;
- the computation precision must be very good;
- the digital system must have enough memory to write the program;
- the development system must have enough peripherals, dedicated for the electric drive applications;
- it is necessary to exist a communication program with a host PC;
- the associated soft must allow an easy program development: editing, debugging, data saving, data visualisations in real time mode an so on.

Numerical systems with floating point digital signal processors (DSP) dedicated to electric drive are the most suiting to accomplish the above mention demands. We present some specific elements for dSpace 1104 system, with particularisation for permanent magnet synchronous motor (PMSM) control. DS1104 system communicates directly with Matlab-Simulink software through the Real Time Interface (RTI) [8]. Also there is a specialised library with different blocs for control of the encoders, CDA, CAD, PWM, etc.

Fig. 4. *ControlDesk capture for PMSM dSpace control system*

A dSpace system has available a ControlDesk application with which it can be visualised and saved in real time characteristic values of the process and it can be set and modified the algorithm parameters while the system is running. In figure 4 is presented a ControlDesk capture while running a vector control algorithm for PMSM. In figure 5 some experimental results are showed.

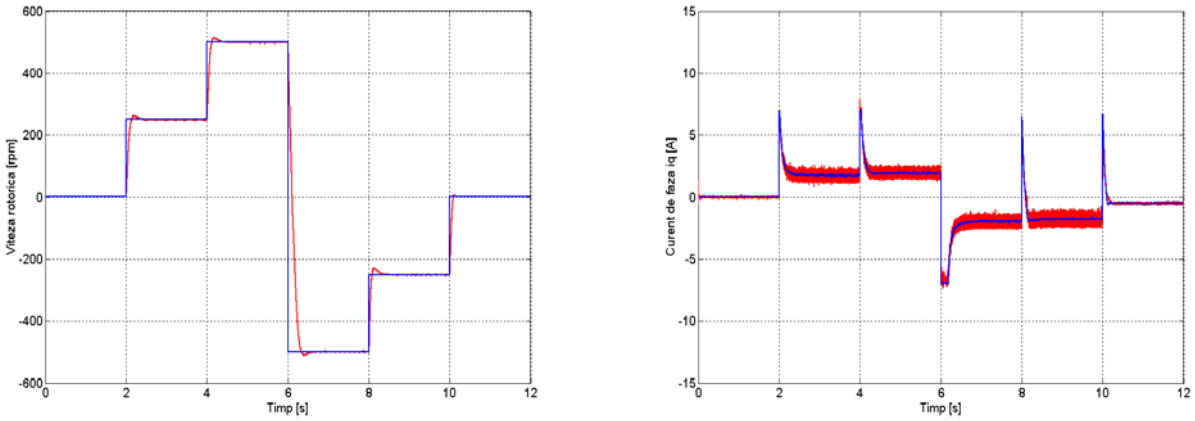


Fig. 5. *PMSM Vector Control results using dSpace system board.*
a) rotor speed and b) stator current (reference-blue, measure-red).

Finally we obtain an optimal solution concerning the performance and practical implementation resources. This will be used in industrial applications.

4. OPTIMIZED THE ALGORITHM FOR INDUSTRIAL APPLICATION

An essential role at this point is the economical aspect. Cost minimisation has as a consequence using fixed point microcontrollers or DSPs.
The necessity of a high power and high computation precision made the fixed point DSP solution most common. The majority of the producing firms had reduced the price of this kind of systems simultaneously with improvement the performances. Also they designed special chips dedicated to electrical drives, which contain all the necessary elements: memory, AD, DA converters, encoder interfaces, PWM blocks.

If the control algorithm optimisation was done well in the previous step, then normally this stage is mainly an optimization. However, the use of fixed point processors brings a series of new aspects which must be followed or resolved in order to minimise the negative effects on the final performances:

- The computation precision is limited – truncated or rounding errors appears. It is necessary to structure the program so that these errors do not cumulate in time.
- The storage capacity of the variables is also limited. In the consequence the quantities which we are working with, must be tuned, brought in to a common value domain (the most familiar is (-1 1));
- The capacity of memory is reduced – program must be optimised, the variables and data numbers must be minimised;
- Low precision of converters and I/O peripherals is also a source of errors.

For this stage we have chosen again as example induction motor sensorless MRAS vector control [1],[4].

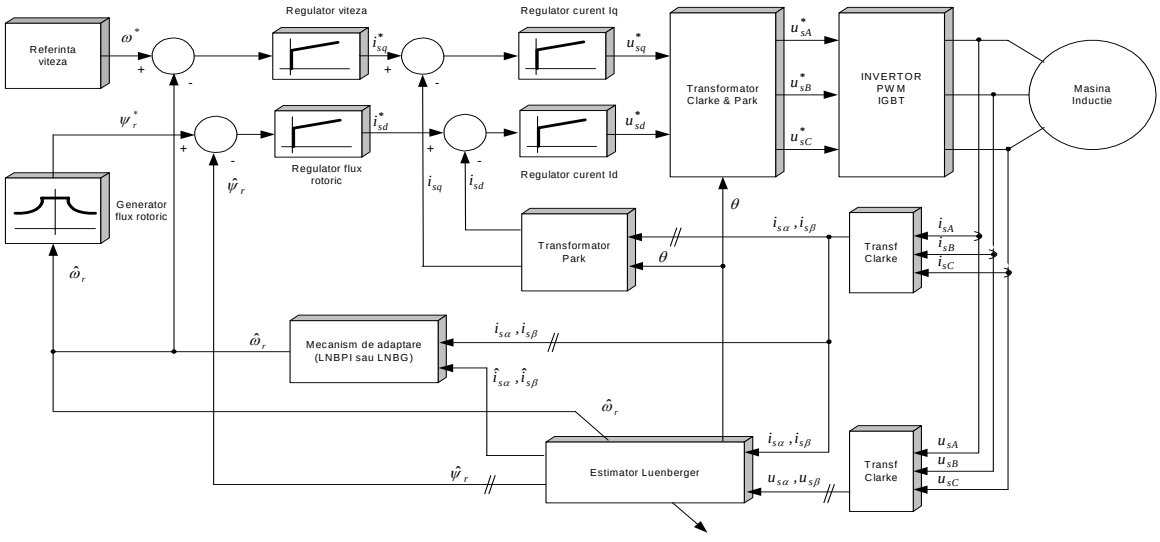
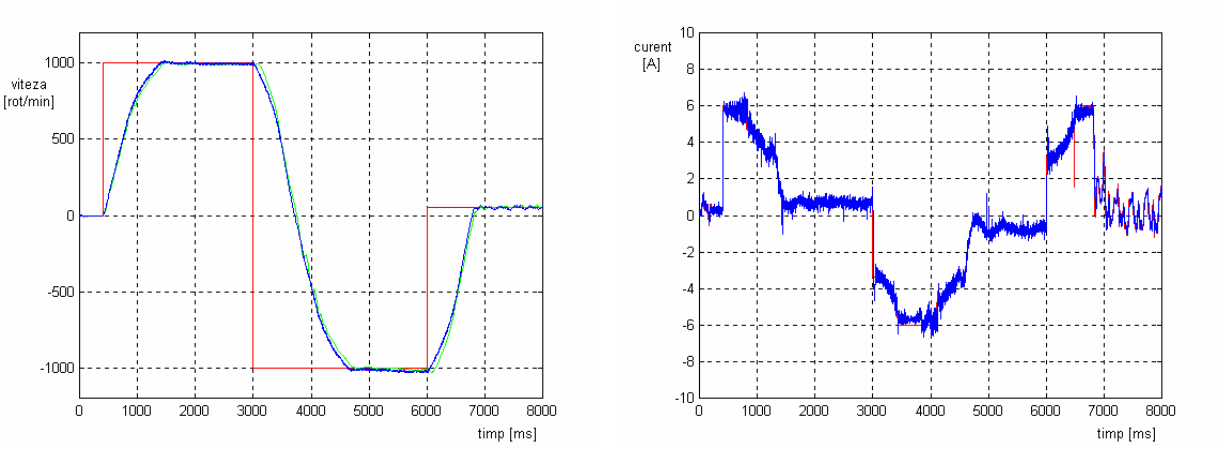


Fig. 6. *Sensorless MRAS based on Luenberger observer induction motor control*

The digital system uses a development board built around a Motorola DSP56F805 fixed point processor, on 16 bits, with 80 MHz working frequency [5]. The evaluation module DSP56F805EVM [6] is used with a software development kit (SDK). This is based on C++ language and has routines and programming examples. The SDK allows working at assembling level, where exists specialised functions for fixed point operations in Q15 or Q31 formats [5]. In the present applications, whose diagram is presented in figure 6, the values where scaled in (-1 1) range, and for the variables the Q15 format were adopted. For the algorithm optimisation were used two methods

[2],[3]: simplified discretization and off-line performed symbolical calculus using Matlab *'Symbolic'* toolbox. In figure 7 are presented some experimental results.



a) b)

Fig. 7. *Induction Motor Sensorless MRAS control – fixed point DSP implementation: a) rotor speed (reference-red, estimated-green, measured-blue); b) active component of stator current (reference-red, estimated - blue).*

5. CONCLUSION

In this paper is presented a general working methodology for designing and developing software applications for electric drives. Three major steps are presented: control algorithms tested by simulations, experimental testing on a laboratory platform and finally application optimisation for industrial implementation. The aim of the first step is control algorithm verifying. In the second step we follow the experimental testing of the algorithm, simultaneously with reduction and effectiveness of the code and minimising the used peripheral number. In the last step we wish to reduce the electrical drive system price, the proposed solution is use the fixed point DSP's. Each step is accompanied by suggestive examples from AC electrical drives area. Applying the presented steps, the necessary time for project finalisation and getting the desired performances is minimised significantly.

6. BIBLIOGRAPHY

1. H. Kubota, K. Matsuse: "Speed Sensorless Field-Oriented Control of Induction Motor with Rotor Resistance Adaption'', *IEEE*, vol. IA-30, nr.5, 1994, pag.1219-1224.
2. M. Cuibus, V. Bostan, R. Magureanu, C. Ilas: "A New Approach for Sensorless Induction Motor Drive", ATEE 2002, Symposium on Advanced Topics in Electrical Engineering, Bucharest, Romania, 29 nov. 2002.
3. G. Griva, F. Profumo, R. Bojoi**,** V. Bostan, M. Cuibus, C. Ilas: "General adaptation law for MRAS high performance sensorless induction motor drives", PESC - 32-nd Power Electronic Specialists Conference, Vancouver, Canada, June 17-22, 2001, pg.1197-1202, vol.2.
4. V. Bostan, M. Cuibus, C. Ilas, R. Magureanu: "High Performances Sensorless Solutions for Induction Motor Control" EPE 2003, 2-4 September, Toulose, France, P1-P9.
5. "DSP56F80x – User's Manual", Motorola INC, 2000.
6. "DSP56F805 – EVM – Hardware Reference Manual", Motorola INC, 2000
7. dSpace - "Implementation Guide" for release 4.0, 2003
8. dSpace - "Matlab - dSpace Interface and Trace Libraries" for release 4.0, 2003.