Experimental evaluation of new one-chip solution for induction motor drives

Rui Esteves Araújo*, João Moutinho* and Vicente Leite**
* *Faculdade de Engenharia da Universidade do Porto, Porto, Portugal*
** *Instituto Politécnico de Bragança, Bragança, Portugal*
raraujo@ fe.up.pt , jnm@ fe.up.pt and avtl@ipb.pt

***Abstract*- The design of high performance induction motor drives is a complex task, and the every day compelling requirements in energy efficiency and performance assumes the motivation on finding a more integrated solution on implementing induction motor control. The main subjects of this paper are two: to discuss the IFOC one-chip solution and to explore the development of simple graphical applications in order to operate this control in a simple and effective way. Experimental results are presented to illustrate the main points of our paper.**

I. INTRODUCTION

System design of AC motor control is undergoing a series of radical transformations to meet performance, quality, safety, cost, and time-to-market constraints introduced by the pervasive use of induction motors in everyday objects. In addition, the variable speed performance has become increasingly important for appliances in the ordinary home (such as washing machines, refrigerators, heating, ventilation or air conditioning) due to the increased emphasis on energy efficiency and quieter in operation.

In this respect, the design of high performance induction motor drives is a complex task, because they are complex systems composed by elements of different natures: static converters, induction motor and controller. A common method to reduce the prototyping time to develop the controller is to use a rapid prototyping system based on a Digital Signal Processor (DSP) board. This approach consists in two main phases: simulation and implementation. First, simulation is performed using a simulator package to develop the control architecture and tuning the parameters controllers. Then, the control algorithm can be implemented on DSP by using C code generator from the model. However, recent developments in Field Programmable Gate Arrays (FPGA) technology have made possible the integration of complex digital control functions in Application Specific Integrated Circuits (ASIC) [1]. Thus, fully integrated control system in a System-on-Chip (SoC) could move near power elements, improving control structure reduction, low cost implementation and control performance increase.

Therefore, if ASIC are used instead of DSP for controller implementation, it becomes possible to shorten the design and prototyping time by using SoC in the implementing phase. It is then clear that one-chip solution for complete closed loop current control and speed control of induction motor drives will be an essential component of these controllers. To reduce design costs, the absence of code is essential. In particular, since system designers will use more and more frequently software to implement their products, there is a need for design methodologies that allow the use of SoC without requiring any programming effort. This implies that the hardware platform of the implementation is essentially "fixed", i.e., the principal components should remain the same within a certain degree of parameterization. The user can thus configure these registers without changing code. Simply writing an offset address and value does this. Since all control is implemented in one single chip, overall hardware implementation is simpler. This is the reason of the reduced design time that this approach can provide as compared to the conventional approach where debugging is needed in both simulation and implementation phases. We believe that hardware platforms will take the lion's share of the induction motor drives market.

International Rectifier Company has recently introduced a new SoC for induction motor applications namely the IRMCK201, which has three distinctive features: 1) it implements indirect field orientation control (IFOC) on configurable hardware processor (a FPGA); 2) unlike a traditional DSP or microcontroller, the user does not need to design software programs to develop control algorithms; and 3) it provides host communication modules to facilitate writing and reading the registers.

Besides its simplicity, it is claimed that the achieved performance is (in some instances) superior to field oriented strategies implemented on DSP because of low latency and the reduced dependence on parameter variations of the induction motor [2]. The objective in this paper is twofold, first to carry out a qualitative analysis of the IFOC parameter dependence, which helps us assess the achievable performance of the current scheme and provide guidelines for its tuning. Second, to propose a new support tool to facilitate writing and reading the host registers. This tool can also be used as a tool for parameter tuning.

II. FIELD-ORIENTED CONTROL BACKGROUND

Vector or field orientation control, as described by Blaschke [3], is a method of controlling a rotating field machine by manipulating the stator magnetomotive force (mmf) in such a manner to attain independent control over the torque and the flux components of the stator current. Applied to induction motors, it makes it possible to control the machine in a manner similar to the control of a separately excited DC machine, and hence it enables the use of induction motors in applications requiring high-dynamic performance, where traditionally only

This work has been partially funded by the Department of Electrical and Computer Engineering of the Faculty of Engineering of the University of Porto (FEUP), under contract DEEC-ID/2/2005.

DC drives were applied. The concept of Field-Oriented control (FOC) applied to induction motor drives, allows us to perform fast and fully decoupled control of torque and flux. To obtain such a decoupled control, FOC algorithms need to know the rotor flux angular position to correctly align the stator current vector. As a consequence it is possible to control torque and rotor flux in a dc motor control fashion, by acting on two separated components of the stator current. Two main different approaches have been proposed in order to calculate the actual position of the rotor flux:

• the indirect FOC (IFOC) technique that predicts the position of the rotor flux by a simple calculation from the reference values and mechanical speed (position) measurement;

• the direct FOC (DFOC) approach, where the position of the rotor flux is measured by suitable flux-sensing devices or estimated from the model of the induction motor.

A. State of the art of motor parameters variation

In order to obtain the performance required by servo applications, induction motor control is achieved using the vector control strategy. This allows high performance control of torque, speed or position to be achieved. However, both approaches present some characteristic drawbacks.

IFOC is the standard regulation method for high-performance induction motor drives that computes the rotor flux angular frequency by adding the rotor speed and the slip angular frequency, respectively, provided by a suitable transducer and a simple expression obtained from the mathematical model of the induction motor. The IFOC technique is essentially a predictive approach as it estimates the angular position of the rotor flux vector by exploiting the model of the machine. It is, thus, very sensitive to variations of motor parameters as an incorrect value of the rotor time constant causes a misalignment between the stator current and rotor flux and reduces the overall system performance. A sluggish torque response is obtained while at steady state flux amplitudes different from the reference one are reached. This results in an increase saturation effect if the flux amplitude is increased, or in a lower available torque if the flux amplitude is reduced [4]. Therefore, long and accurate setup procedures are usually required when starting an IFOC drive for the first time in order to precisely estimate the rotor time constant. To maintain the dynamic performance of the system, online tuning procedures are needed to compensate any change of the rotor time constant due to motor temperature variations. Different approaches have been presented in literature [5-9]. All of these approaches require additional sensors that were not strictly used in the original IFOC drive, thus, increasing cost and complexity. Most of them introduce complex methodologies that cannot be practically used while others only permit offline estimation of the rotor time constant or can be used if the speed is sufficiently high. The DFOC approach is in nature insensitive to motor parameter variations as it is based on a direct measurement of the actual position of the air-gap flux. However, DFOC schemes can hardly operate at very low or zero frequency due to the lack of back-EMF signals. Consequently, the use of DFOC drives is today practically restricted to those applications not requiring low speed operations.

B. Indirect field oriented control scheme

The indirect field-oriented control (IFOC) scheme for the induction motor implemented in IRMCK201 is a standard version. The PWM current control loop operates in synchronous field-oriented coordinates *d-q* as shown in Fig. 1. The feedback stator currents i_{sd} and i_{sq} are obtained from the motor phases currents i_a, i_b after a three-phase to two-phase conversion, followed by coordinate transformation, $\alpha\beta$ to *dq*. The current controllers generates the voltage control u_d, u_q, which after inverse coordinate transformation, *dq* to $\alpha\beta$, are applied to the space vector modulator (SVM) algorithm. The current controllers are classical PI controllers. Finally, the SVM determines the switching signals d_a, d_b and d_c for the static converter. The mechanical speed is controlled by a PI controller. As in the scheme under study, the flux is controlled in open loop, by using a machine model fed by a current-controlled pulse-width-modulation (PWM) inverter in a *dq* reference frame synchronous with the rotor flux. It is possible to demonstrate that for a fixed stator current vector only one-slip frequency w_{slip} value exists that allows to reach field orientation conditions. In that case, having the rotor flux constant and fixed on the axis, it results that

$$\omega_{slip} = \frac{1}{T_r}\frac{i_{sqref}}{i_{sdref}}. \qquad (1)$$

where T_r is the rotor time constant, and i_{sdref} and i_{sqref} are the current commands in *d* and *q* axes, respectively.

C. Parameter Sensitivity

In the constant flux operation region, when the rotor time-constant T_r, deviates from the nominal value, results in deviation in the slip frequency value calculated from (1). This leads to problems with the failure of decoupling control in the field-oriented control. In this operating condition of the induction motor drive, the power efficiency and the controlled performance are degenerated. The effects of motor parameter detuning have been analyzed in several publications. However, in this paper are presented experimental results that evaluate the performance of the IMCK201.

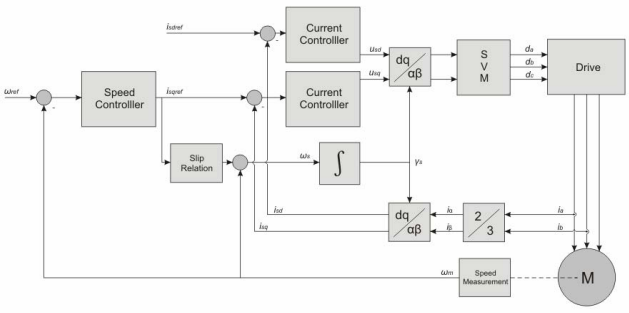

Fig. 1. Standard IFOC drive scheme with speed regulation.

III. INTERFACE DEVELOPMENT

The first issue is to define the objectives of having an interface program to interact with the SoC:

- To establish the status state.
- To control the input variables.
- To monitor the output variables.
- To perform intuitive control.
- Portability.
- Support.

To accomplish all this characteristics, and since one of the greatest interests is to implement a low cost solution, it is necessary to use a software design platform that fulfills our needs. In this work, using JAVA with the Eclipse design platform was our choice. Java is a new, object-oriented programming language developed by Sun Microsystems. In this work we will be motivating the use of Java for building appealing and usable GUI frontends to IRMCK201.

A. Why use Java?

The Java language has become very successful since its formal introduction in 1995. The language itself is very attractive: it is object oriented and standard class libraries provide very natural interfaces to features such as multi-threading, internet communication and protocols, graphical components, Graphical User Interface (GUI) design facilities, customizable security restrictions, etc. Java has gained enormous popularity since it first appeared. Its rapid ascension and wide acceptance can be traced to its design and programming features, particularly in its promise that you can write a program once, and run it anywhere. No language is simple, but Java is considered to be an object-oriented programming language much simple and easy to use when compared to the popular programming language C++. Partially modeled after C++, Java has replaced the complexity of multiple inheritance in C++ with a simple structure called interface, and also has eliminated the use of pointers.

The reason why Java is much simpler than C++ is because Java uses automatic memory allocation and garbage collection where else C++ requires the programmer to allocate memory and to collect garbage. Also, the number of language constructions in Java is small for such a powerful language. The clean syntax makes Java programs easy to write and read. Java is object-oriented and models the real world. Everything in the world can be modeled as an object. For example, a circle is an object, a person is an object, and an IC is an object. Even a serial port can be perceived as an object. Java is object-oriented because programming in Java is centered on creating objects, manipulating objects, and making objects work together.

B. Portability

One of the most compelling reasons to use Java is its platform independence. Java runs on most major hardware and software platforms, including all Windows versions, Macintosh, and several varieties of UNIX and Linux. All Java-compatible browsers support Java applets. By moving existing software to Java, you are able to make it instantly compatible with these software platforms. JAVA programs become more portable. Any hardware and operating system dependencies are removed. In C and C++, each implementation decides the precision and storage requirements for basic data types (short, int, float, double, etc.). This is a major source of porting problems when moving from one kind of system to another, since changes in numeric precision can affect calculations, and assumptions about the size of structs can be violated. Java defines the size of basic types for all implementations; an int on one system is the same size (and can represent the same range of values) as on every other system. It does not permit the use of arbitrary pointer arithmetic, so assumptions about struct packing and sizes can not lead to non-portable coding practices.

Although C and C++ are supported on all platforms that support Java, these languages are not supported in a platform-independent manner. C and C++ applications that are implemented on one operating system platform are usually severely intertwined with the native windowing system and OS-specific networking capabilities. Moving between OS platforms requires recompilation, as a minimum and significant redesign, in most cases. With this, one of the most important barriers has fallen: no more a platform will be an impediment.

C. Limitations of the Java language

Of course, Java has its own problems. Some concerns are the fact that the language is still under development and is growing, and some key elements such as the event model may change from one version to the next. The main disadvantage of Java is speed. Although Java's ability for producing portable, architecturally neutral code is desirable, the method used to create this code is inefficient. Java code is compiled into byte code. An interpreter called a Java Virtual Machine, specifically designed for a computer's architecture, runs the program. However, Java being an interpreted system, is currently an order of magnitude slower than C. Unlike natively compiled code, which is a series of instructions that correlates directly to a microprocessor's instruction set, an interpreter must first translate the Java binary code into the equivalent microprocessor instruction. Obviously, this translation takes some amount of time, no matter how small a length of time this is, and is inherently slower than performing the same operation in machine code. So it is simple to conclude that for exigent performance applications Java may not be the right choice. But for motor monitoring, Java performance is probably more than enough.

D. Technology used and prototype

Eclipse is used to implement the GUI in our prototype. It is a popular free Integrated Development Environment (IDE) for writing Java. The Eclipse Project is an open source project of eclipse.org. It is an enormous project encompassing more than just an IDE and it allows plug-ins for extra functionality. It uses its own compiler that lets it do incremental compiles and hot swapping of code during debugging.

This GUI was completely developed in Eclipse very quickly and it is a good example of how the design engineers can avoid buying those evaluation platforms that usually end up with no application. Due to the object-oriented nature of Java, this application is very customizable, and can be easily converted if a new version of hardware is released or even if the application requisites are changed.

In Fig. 2, expert mode is selected and the user can access all the IRMCK201's registers. It is also possible to see the chart plotting ability of the desired register. Trough sliders it is possible to set the reference input values either in "Speed Control" or in "Torque Control" mode. And at all time, "status", "speed" and "speed error" are monitored to provide useful information about the condition of all the system, either if everything is running or something is faulty.

Another aspect is the system configuration. It is desirable that this system operates with several motor types. Considering this, a Matlab script was built that automatically generates a config file that can be sent trough the application, inserting only a few characteristics of all the parts that constitutes the system (pwm frequency, motor characteristics, encoder details, etc.). This button can be seen in Fig. 2 and it is called "Send file". It works like a preset configuration that is defined in a Matlab script file due to the simplicity of formula building and easiness of use.

The reason for this mixed solution is again related with the need to minimize time in the development. Matlab with all its features allows that everything is done automatically, by just inserting the required information in a few fields and exporting this file that can be imported later. After this, it is just necessary to control the motor, without concerning about configuration, and only intervening on the aspects that are really in the interest of the application, that in this case is Energy Optimal Control.

IV. Experimental Results

An experimental system based on the IRMCK201 has been constructed in order to verify proper operation and evaluate its performance (see Fig. 3). Some experimental tests have been carried out on a test setup. This laboratory setup consists of a test bench with a programmable powder brake, an incremental encoder and a tachometer generator, and a torque sensor, all

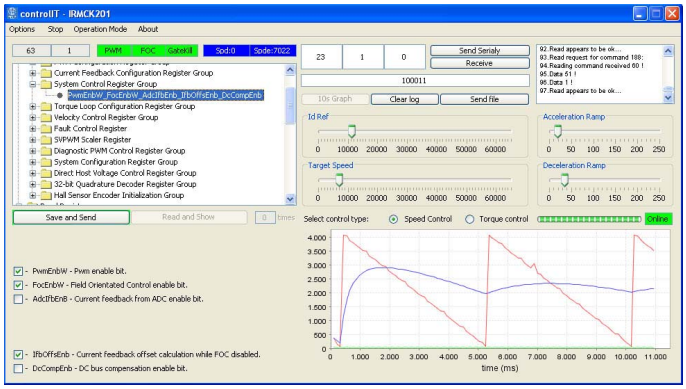

Fig. 2. Screenshot of GUI in a Windows XP platform.

Fig. 3. ControlIT hardware design platform.

from Leroy Somer. The motor was a standard two-pole three-phase cage induction machine (0.18kW 400-V). The motor is powered by an insulated gate bipolar transistor inverter.

Fig. 4 shows a line-to-line voltage, voltage alpha component, and flux vector position γ_s (see Fig. 1) when the controller operates in normal conditions. An oscillogram illustrating the dynamics of voltage vector commands into $\alpha\beta$ coordinates can be seen in Fig. 5. It represents an experimental speed reversal with main zoom active in oscilloscope. Fig. 6 shows signals' waveforms of reference speed and electrical angle during a speed reversal. The oscillograms of Fig. 7 represent a large step change in the speed command that will cause the generated current command from a proportional-integral (PI) speed controller to exceed the prescribed maximum value. Thus a saturator is applied which brings the non-linearity to the system. This is called integrator windup phenomenon that leads to large overshoots, long settling time, and even unstable speed responses.

The oscillograms of fig. 8, on the other hand, represent the case of detuning. They show transient responses of the voltage vector commands in stator coordinates $\alpha\beta$. Note that, when the rotor time-constant T_r , deviates from the nominal value, the actual slip frequency for IFOC becomes:

$$\omega_{slip} = \omega^*_{slip} + \Delta\omega_{slip} \qquad (2)$$

where ω^*_{slip} is the true value of slip frequency. This leads to problems with the failure of decoupling control in the field-oriented control.

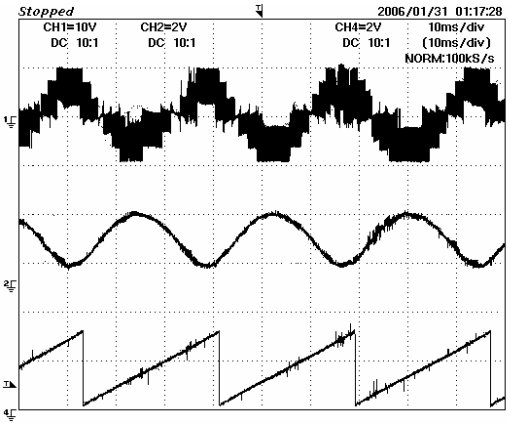

Fig. 4. Experimental oscillograms illustrating normal operation.

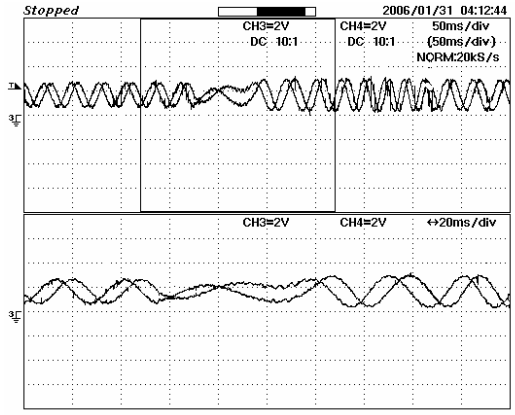

Fig. 5. Experimental oscillograms illustrating a speed reversal.

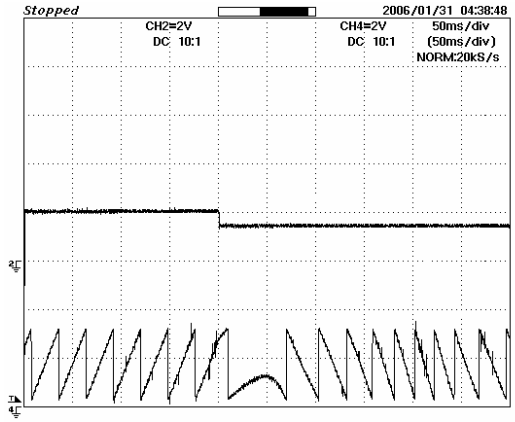

Fig. 6. Electrical angle and reference speed during speed reversal.

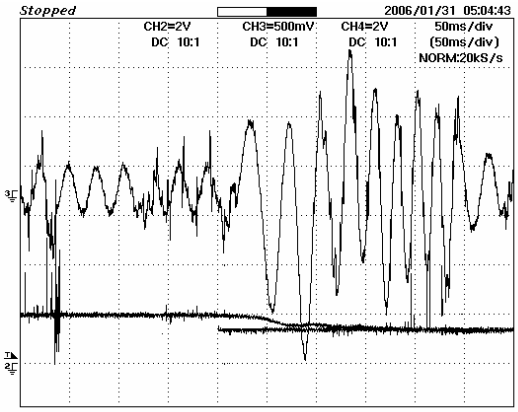

Fig. 7. Phase stator current, actual and reference speed during large step change in speed.

V. Conclusions and Future Work

A. Conclusion

This paper has discussed a speed control scheme for the induction motor based on the new SoC (IRMCK201). In order to evaluate the practical features of the new servo on-chip a new hardware platform has been developed as well the support tool interface. It was shown that the new controller is able to track the rotor speed even in the case of rapid torque change. The presented results were obtained using an experimental test

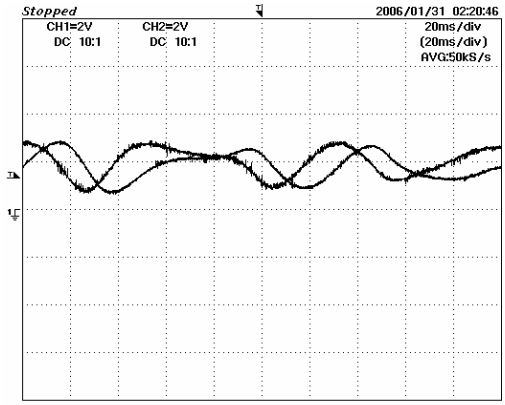

Fig. 8. Detuning of steady-state parameters for rated flux and torque current.

bench with an induction motor loaded by a powder brake. It was also shown that the correct knowledge of rotor constant-time value is essential for that the completely decoupling control for IFOC is exactly established. Finally, the interface software developed for the hardware platform has been tested.

B. Future Work

Future research will be concentrate on methods of rotor resistance estimation. In consequence, we will be able to use more accurate value of rotor time-constant during slip gain computation and improve its performance. We expect to achieve good performance in a wide speed range together with the rotor parameter adaptation ability using a low cost microcontroller connected with IRMCK201.

In addition, a low cost controller will be designed in the near future. This controller will be targeted at appliance specific for light electric vehicles.

References

[1] International Rectifier, “High Performance Configurable Digital AC Servo Control IC”, Data Sheet No. PD60224 Rev.B, January 2004.

[2] International Rectifier, “Complete Encoder Based Servo Drive Design Platform iMOTION Development System”, Reference Design, Mars 2004.

[3] BLASCHKE, F.: “The principle of field orientation as applied to the new transvector closed-loop control system for rotating-fieldmachines,” Siemens Review, 1972, 39, pp. 217-220.

[4] K. B. Nordin, D.W. Novotny, and D. S. Zinger, “The influence of motor parameter deviations in feedforward field orientation drive systems,” IEEE Trans. Ind. Applicat., vol. IA-21, pp. 1009–1015, July/Aug. 1985.

[5] C. Cavallaro, E. Cerruto, A. Consoli, A. Raciti, and A. Testa, “Slip gain tuning in indirect field oriented control drives,” Elect. Mach. Power Electron., vol. 23, no. 1, pp. 63–80, 1995.

[6] L. C. Zai and T. A. Lipo, “An extended Kalman filter approach to rotor time constant measurement in PWM motor drives,” in Conf. Rec. IEEE-IAS Annu. Meeting, 1987, pp. 177–183.

[7] R. D. Lorenz and D. B. Lawson, “A simplified approach to continuous, on line tuning of field oriented induction machine drives,” in Conf. Rec. IEEE-IAS Annu. Meeting, 1988, pp. 444–449.

[8] L. J. Garces, “Parameter adaption for the speed-controlled static AC drive with a squirrel-cage induction motor,” IEEE Trans. Ind. Applicat., vol. IA-16, pp. 173–178, Jan./Feb. 1980.

[9] J. C. Moreira, K. T. Kung, T. A. Lipo, and R. D. Lorenz, “A simple and robust adaptive controller for detuning correction in field oriented induction machines,” in Conf. Rec. IEEE-IAS Annu. Meeting, 1991, pp. 397–403.

[10] Peter van der Linden, *Just Java 2*, Prentice Hall, , ISBN: 0131482114, June 2004.