AN EXPERIMENTAL SETUP FOR THE STUDY OF FIELD-ORIENTED CONTROL OF AC MACHINES

LUIZ G. B. ROLIM, WALTER I. SUEMITSU, RICHARD M. STEPHAN, MARCUS B. DE MEDEIROS

Laboratório de Eletrônica de Potência, Departamento de Eletrotécnica (DEE), Escola de Engenharia (EE), Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 68.515 – CEP 21.945-970 Rio de Janeiro, RJ, Brasil
E-mails: rolim@dee.ufrj.br, walter@dee.ufrj.br, richard@coe.ufrj.br

Resumo—Neste trabalho é apresentado um arranjo experimental para o estudo do controle vetorial (ou orientado pelo campo) de motores de indução em laboratório. O sistema baseia-se na bancada didática modular de eletrônica de potência, desenvolvida no DEE – EE/UFRJ, cujo controle é totalmente executado por um microcomputador pessoal. O trabalho inclui resultados experimentais.

Abstract—In this paper, an experimental setup for the laboratory study of vector (or field-oriented) control of induction motors is presented. The system is based on the modular didactic power electronics workbench, which has been developed at DEE – EE/UFRJ, and which is completely controlled by a personal microcomputer. Experimental results are included.

Keywords—Induction motors; Drives; Laboratory Education.

1 Introduction

Teaching field-oriented control of AC motors in undergraduate courses has been a difficult task at UFRJ and probably elsewhere as well. Undergraduate curricula are generally swollen and do not provide enough time to teach thoroughly the fundamental principles of d-q theory. This makes it even more difficult for students to learn practical aspects of converter-fed AC motor drives in the laboratory. However, AC motor drives are getting increasingly adopted in variable-speed industry applications, and many manufacturers are even discontinuing the production of DC drives for such applications, keeping almost only vector-controlled AC drives in their product palette.

For the reasons presented above, an attempt is being carried out in the Department of Electrical Engineering (DEE) at UFRJ, to bring the study of field-oriented control of AC motor drives to the undergraduate power electronics laboratory, using the modular workbench system described by Rolim et al. (1993, 1995).

It was pointed out in those papers that the modular system presented there is very suitable for use as an integrated teaching laboratory for several subjects of the Electrical Engineering curriculum. Additionally, the use of a personal microcomputer as the unique control equipment is very beneficial, because a high degree of interaction with any part of the control system can be achieved. The student himself can easily modify parts of the control software, using familiar software tools.

Some other examples of electric machine drive applications were also shown in Rolim et al. (1993, 1995), such as speed control of a DC motor with chopper and rectifier circuits and volt/hertz control of an induction motor.

For the purpose of including new experiments on control of AC machines, several new pieces of software have been added to the control section of the modular workbench system mentioned above. Also, some new hardware has been added to the set of interface modules, to enhance the current and speed feedback capabilities.

In this paper, the new software and hardware modules incorporated to the modular power electronics workbench are described, and an application example is shown, consisting of a didactic experiment on field-oriented (or vector) control of a three-phase induction motor.

2 Control Software for the Modular Workbench

In the original laboratory workbench described in Rolim et al. (1993, 1995), the control module consisted of a single DOS application written in Pascal, with a text user interface. The control tasks were performed by an interrupt handler, which was coded directly in the same source file as the main program, which contained basically the user interface routines. The Pascal compiler and the DOS operating system allowed an interrupt service routine to service an interrupt with minimum latency, because Windows virtualizes the programmable interrupt controller (PIC) (Oney, 1996). Furthermore, Windows software can run in one of two different privilege levels, called ring zero and ring three. Critical operating system tasks run in ring zero, while most application programs run in ring three. For an interrupt handler to service an interrupt with minimum latency, it should be executed in ring zero.
Therefore, it has been found out that the control tasks should be put in a ring-zero program, while the user interface was written as a normal ring-three application. A class of ring-zero software that suits very well to our purposes is the so-called virtual device driver (VxD). A VxD is a loadable module, which is intended for interfacing with peripheral devices. It can contain interrupt service routines that execute with minimum latency in ring zero, but it requires the Windows driver development kit (DDK), in order to be created (Oney, 1996).

Figure 1 shows a screen shot of a sample of a graphical user interface (GUI) panel, which has been created with CVI for the induction-motor vector-control application in the modular laboratory. The displays and controls in the GUI panel can be easily modified and new ones can be included with help of the development tools mentioned above. The main components of the sample user interface panel shown in Figure 1 are:

- rotary knobs for adjusting following control system parameters: speed sample time $h$, current sample time $T_i$, speed controller parameters $K_p$, $T_i$ and $T_d$, presumed rotor time constant $T_r$ and reference value for the direct-axis current component $I_{sd}$;
- rotary knob for the rotor speed reference value $W_{ref}$, on/off and start/stop buttons;
- speedometer-like rotor speed indicator and strip-chart showing the dynamic variation of the actual rotor speed along the time (the curve shown is the actual measured speed of a running experiment).

An open and modular framework has been adopted for the software development and the entire source code is written in C, with emphasis on readability. Therefore, it is quite straightforward to include new algorithms or to modify the existing ones. With this approach, it becomes easy for the user to study any part of the control system with more detail, without losing the overview of the totality.

3 Experimental setup

Figure 3 shows the schematic diagram of the experimental setup for the proposed application example. It consists of a didactic experiment on field-oriented control of a three-phase induction motor. The box labeled “interrupt handler in VxD” represents the software module that is responsible for all the control tasks, including angular speed and three-phase current sampling, feedback speed regulation with a PI controller, dq to abc coordinate transformation, current-regulated delta modulation (CRδM) and direct gating of power semiconductor switching devices. A photograph of the experimental workbench is shown in Figure 2.

A simplified indirect field-oriented control algorithm is used, in which the direct-axis component of stator current ($i_{sd}$) is fixed, i.e. there is no feedback control of rotor flux linkage. The reference value for the quadrature-axis component of stator current ($i_{sq}$) is obtained from the output of the outer-loop PI speed controller. Both $i_{sd}$ and $i_{sq}$ are used as inputs to the inner current control loop (CRδM), after being con-
verted to the abc reference frame. The values $i^s_{sd}$ and $i^s_{sq}$ are also used as inputs to the rotor flux-linkage reference position calculation algorithm, which is based on a physical model of the induction motor (Leonhard, 1996; Santisteban et al., 1995).

The speed measurement is done with help from one of the timer/counter circuits available in the interface module. An auxiliary circuit, which has been implemented in a single programmable logic device (a simple PLD of type GAL), isolates one single pulse from the pulse train generated by the incremental encoder; this single pulse is then applied to the “enable” input of the timer/counter. The clock frequency is fixed, but counting actually occurs for the time duration of the isolated pulse only. The difference between counter register contents, after and before the appearance of the “enable” pulse, is thus inversely proportional to rotor speed. The control software then reads the measurement and the auxiliary circuit is reset, to accept a new pulse.

The incremental encoder mounted on the motor shaft produces 800 pulses per revolution and the timer/counter is clocked at 2MHz. This combination yields a speed measurement error less than 2% for speeds up to 3000 revolutions per minute (rpm). The lower speed limit is given by the sample interval used for the speed measurement and digital control loop, and lies around 100 rpm.

A schematic diagram of the interface circuit from the incremental encoder to the timer/counter is shown in Fig. 4.

Fig. 3: Schematic diagram: vector-controlled induction motor drive.

Fig. 4: Schematic diagram of interface circuit from incremental encoder to timer/counter.
in Fig. 4. The signals labeled “A” and “B” are the two 90° phase-shifted pulse-train channels coming from the incremental encoder. The signal labeled “GT2” is the “enable” input to the timer/counter (8253) and the signal labeled “FR” comes from the timer/counter output and is used to reset the interface circuit.

4 Experimental Results

To exemplify the use of the proposed didactic system, some experiments have been carried out with a low-power three-phase motor. The machine used in these experiments is a low-voltage, two-pole induction motor rated 30V, 60W. The motor performance was observed without shaft load at different reference speeds. The actual rotor speeds as well as the line currents were recorded. Figure 5 shows the steady-state motor line currents for no-load operation at 500 rpm. Figure 6 shows the variation of the line currents during startup as the rotor speed changes, for a speed reference value of 1500 rpm.

5 Conclusions and further development

In this paper, an experimental setup for the study of field-oriented control of AC machines has been described. With the proposed system several studies can be carried out with a high degree of interaction and flexibility, which would be otherwise difficult to obtain with other equipment. With the friendly user interface, the influence of any control system parameter can be readily investigated in a very insightful way. Yet another interesting way of using this didactical system is to ask the students to modify parts of the source code, with the purpose of studying, for instance, different control algorithms or different modulation techniques.

In addition to the didactic application presented here, several research works can be carried out with this experimental setup. Some of them are already under development, with help from students with undergraduate research scholarships. These research topics include fault detection, self-tuning, self-commissioning and sensor elimination.

Another project that is being initiated is the development of a www-browser interface for remote access (distance laboratory learning). This involves the development of driver software for open FC UNIX (Linux and FreeBSD).

Acknowledgements

The authors are grateful to FUJB, FINEP (RENGE), FAPERJ, and CNPq for the financial support.

References


