A Master-Slave DSP Board for Digital Control

ROSA, F. E.[1]; CARRARA, A. R. S.[2]; SOUZA, A.H.[3]
(1) M.Sc. candidate e-mail: fabio_e_rosa@zipmail.com.br
(2) Professor, Ph.D. e-mail: dee2arsc@dcc.fej.udesc.br
(3) Professor, Dr. e-mail: dee2ahs@dcc.fej.udesc.br
State University of Santa Catarina – Brazil

ABSTRACT

This paper presents an integrated environment for development of real-time digital control systems. It constitutes a pilot project in the real-time aspect using Digital Signal Processor (DSP). The hardware platform is composed by a Master-Slave DSP board and the software platform has a friendly user interface.

It is intended to generate a basic real-time tool library that will allow others research projects to be conducted.

KEYWORDS

Digital control systems, Digital Signal Processors, motor control, DSP robot control.

INTRODUCTION

Digital Signal Processors are making a revolution in real-time control boards, mainly in motion control field.

There are several processor boards that allow the implementation of this environment. Resources found in the fixed-point DSP TMS320F240 ('F240) are ideal for motion control applications but not enough to do efficient graphical interface in real-time, what is necessary during the research stage.

On the other hand the 'C6x EVM board, based on the fixed-point DSP TMS320C6201, is extremely fast and it allows an easy interaction with PC applications, but there are a few resources interfaces for the external world.

Thus, the idea is to use the '6Cx EVM, connected to a PC, as the Master of a Slave board optimized for acquisition and control, which has the 'F240.

The basic plant for motor control research is show in figure 1, which is composed by an ac-induction motor and a dc motor as load. Normally, the main purpose is to study the characteristics of asynchronous machines and to test advanced control algorithms. The inputs are voltages, currents, speed and torque. The inverter is driven by pulse width modulated (PWM) signal from the controller board.

Many kind of real-time control platforms have been used in this field of motion control. In [1] the author uses only a PC with a data acquisition board achieving a control cycle of 1ms for vector control.

In [2] the author uses the DS1102 controller board from dSPACE based on the floating-point DSP TMS320C31 with a subsystem based on a TMS320P14 microcontroller DSP.

Recently, dSPACE released the DS1103 board based on a PowerPC604e running at 333MHz with a 'F240 as Slave. What shows the tendency of specialized platform for digital control system.

APPLICATIONS

The characteristics and functions of the proposed system were originated by the goal of to attend the several applications. In our laboratory this platform will be use for the followings applications.

1. Parameters identification of ac-induction motor

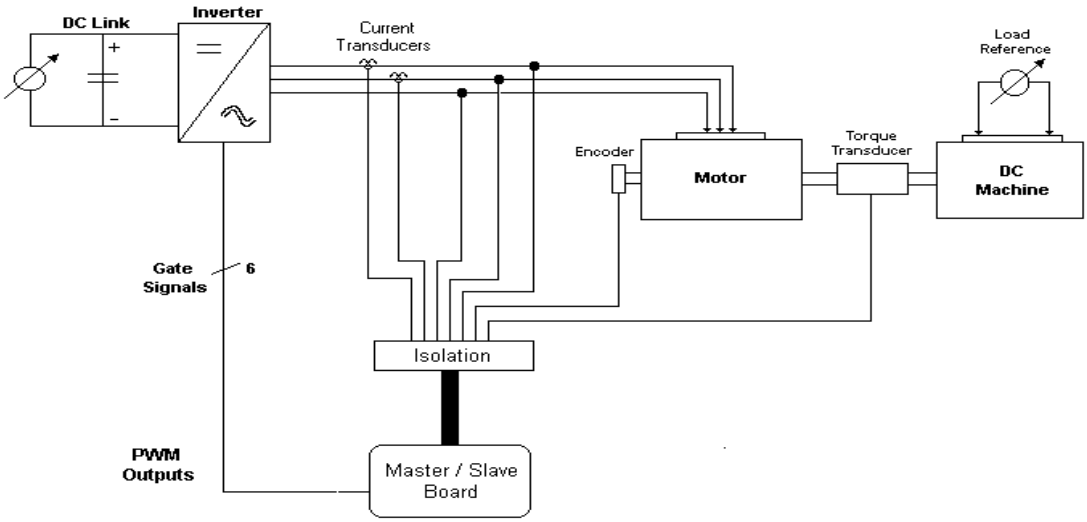


Fig.1 - Block scheme of system for motor control.

The goal is to identify the parameter of an ac-induction motor using a random signal generator to excite the motor (PRBS signal). Identification parameters are stator resistance and inductance, rotor resistance and inductance, and mutual inductance. The identification algorithm used is the least mean square (LMS) method, but others as output error (OE) also can be applied [3].

Figure 2 shows the block diagram for this application. The sample frequency of motor voltage and current is 5KHz. The basic cycle of the PRBS signal is about 5ms. The output voltage is updated with this rate driving the motor using PWM technique. The identification is accomplished off-line aiming at the autocommissioning for induction motor, this process is done using the file containing all data acquisition generated at the end of the run.

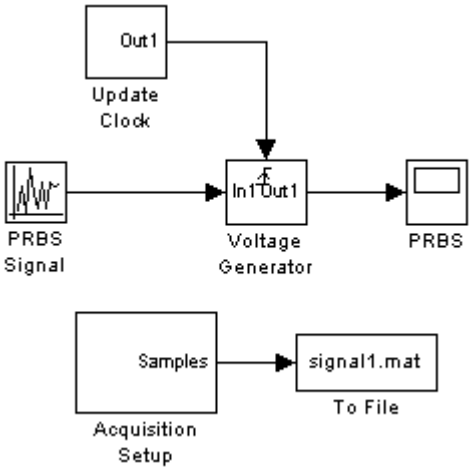


Fig.2 –Block diagram for PRBS generation.

2. Flux estimation

The block scheme of fig.1 can be used to estimate the flux for vector control of an ac-induction motor. This study aims at the robust estimation of the flux, i.e., an estimation that is insensible to the possible parameter variations caused by different operation conditions. These variations are caused by the saturation conditions and the temperature of the motor.

The Master/Slave board does the acquisition of the signal from the torque transducer. This signal can be compared with the torque calculated from the estimated flux.

3. Vector control using sliding mode control

This system can be used for an induction motor drive, for example, using the nonlinear sliding mode torque and flux control combined with the adaptive backstepping [5]. This control type is based on the state-coordinates transformed model, representing the torque and flux magnitude dynamics, the nonlinear sliding mode control is designed to track a linear reference model. The adaptive backstepping control approach is used to obtain robustness for mismatched parameter. The system of fig.1 is also applied in this case.

4. Real-time control of a three degree-of-freedom (DOF) spherical wrist for robots.

Another application is robot controller. Initials tests will be done in a three DOF robot to test simple robot wrist controls. To control this robot is necessary to measure precisely three positions, given by the respective encoders as shown in figure 3. With these positions, the calculations of direct and inverse kinematics are done. The output of the board will generate three independent signals of ±10V range to control the three axes using a dc servo control.

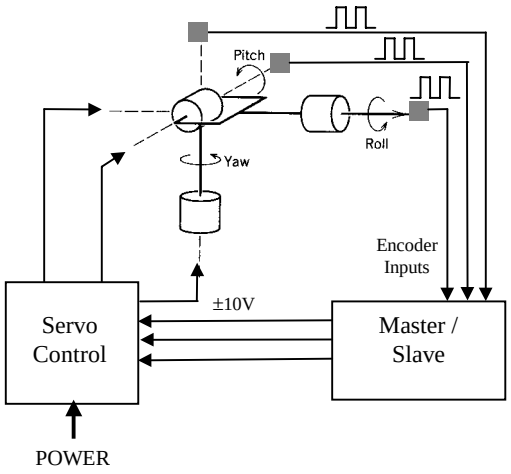


Fig.3 – Robot control for spherical wrist [4].

To optimize the dynamics of robot control, increase precision and reduce ripple (oscillations) more than a good mechanic design is necessary. Fast and precise controls are also required. For this purpose, complex calculations can be done into a control cycle of around 100μs, this is achieved programming the DSP ‘C6x using C language.

ARCHITECTURE OVERVIEW

The system is composed by the high-performance fixed-point DSP TMS320C6201 ('C6x) configured as Master and TMS320F240 ('F240) tailored for motor control configured as Slave.

The main tasks performed by the Master board are tasks management, computer intensive calculation, interface with PC and buffer.

The Slave board is responsible by the interface functions with the real world, such as: PWM generation, analog-to-digital conversion, digital–to-analog conversion, input/output control of digital data and capture/QEP (Quadrature Encoder Pulse).

Communication between the Master and Slave are made by the SPI protocol (Serial Peripheral Interface) at a speed of 2.5 Mbps. Even though this is not the fastest way, it is very simple to implement and eliminates any control device chip between them, because both devices have this protocol integrated.

Figure 4 shows the overview of the system.

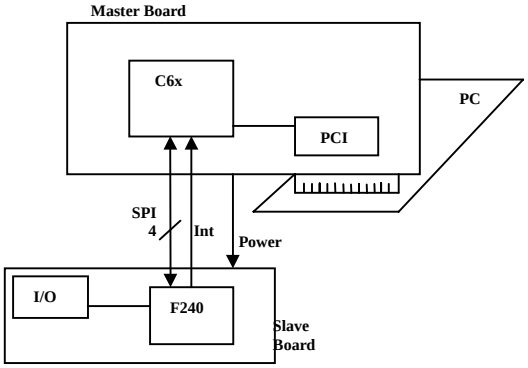


Fig.4 – Master/Slave system.

The interrupt (XENX_INT7) at 'C6x DSP is connected to the Slave board, this allows the Master to be informed when there is new data available, such as external interrupt or end of conversion.

MASTER BOARD CHARACTERISTICS

The Master board is the Evaluation module board, 'C6x EVM, intended for use in a PCI expansion slot inside the PC. Features of the board are listed below:

- TMS320C6201 DSP
  1280 MIPS at 160MHz;
  SPI communication;
- PCI interface;
- 2Mx32-bit SDRAM memory;
- Code Composer 4.0 (with JTAG for emulation)

One of the advantages of this board is that the system can be updated easily. Using the floating-point DSP version no modification is required, since the TMS320C6701 has the same dimension and is pin-to-pin compatible with the 'C6201. And the tool for emulation and compiler is the same.

In the actual context of implementation all it is needed is a slave 'F240 to perform peripheral control functions.

SLAVE BOARD CHARACTERISTICS

The Slave board contains peripherals connected to the 'F240 in order to form composed the system, which has the specifications below:

- 'F240 DSP at 20MIPS;
- Six PWM output + Drive protect interrupt;
- Eight channels 14-bit Analog-to-digital converters (2x4 channels simultaneous sampling) at 76ksps;
- Three channels 16-bit Digital-to-analog converters with selection of ±10V or 4-20mA;
- 3 external interrupts;
- 8 digital inputs;
- 8 digital outputs;
- 32K-16bits of extern RAM for Emulation;
- 2 general purpose leds;

Figure 5 shows the block diagram of the Slave board.

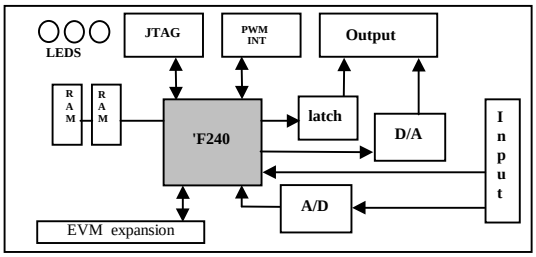


Fig. 5 - Slave board overview.

At the beginning the 'F240 will be underused, since only the Space Vector PWM generation required a portion of it, but new functions will be added to it later in the development.

HOST COMPUTER ENVIRONMENT

The board can be programmed in two ways, one using the Matlab-Simulink software and other using Visual C++. The environment using the Simulink is intended for fast development, simple acquisition and for teaching.

The advantage of using Simulink is the easiness of generation and flexibility to create systems.

Figure 6 shows the hardware library created for Simulink. This library consists of calls that will be executed by Slave board, i.e. they are peripheral routines carried out by the 'F240.

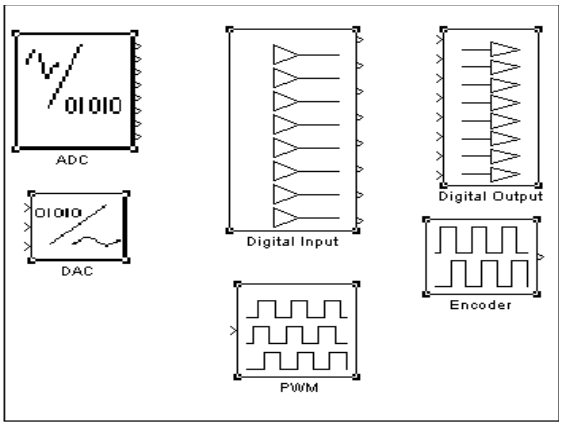
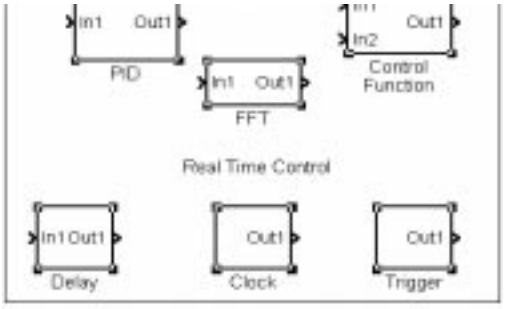


Fig.6– Simulink Hardware library.

Figure 7 contains the software library, i.e., the routines executed by the Master board with the purpose of using the high-speed advantage of the 'C6x DSP.

Fig.7 – Simulink Software library.

To run the simulations using Simulink in real-time task synchronization is necessary, this is done using a triggered subsystem. Triggered subsystems are the



ones that execute its functions each time a trigger event occurs.

The idea is to use the clock from the Master board to generate this trigger. But the problem with the Simulink environment is the slow execution cycle that depends on the host speed (PC), because the Simulink will have to execute all tasks before a new trigger event occurs. In our tests we use a 100MHz Pentium II, that allowed a 1ms cycle for simple systems. Thus, only slow systems can be simulate in real-time using all the Simulink resources in this case.

A simple example of a trigger subsystem is illustrated in figure 8. The subsystem contains Simulink blocks executed by PC and blocks from the Master/Slave library. All synchronized tasks must be inside the subsystem block.

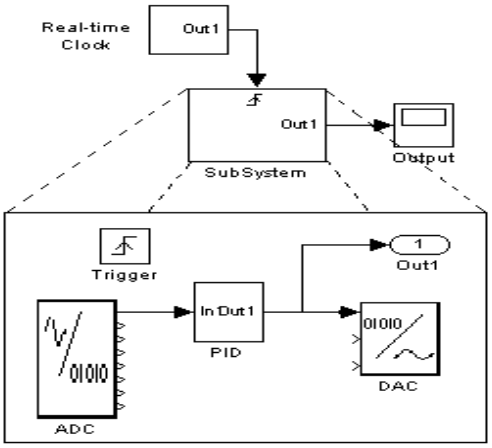


Fig.8 – Triggered subsystem.

One way to speed up the data acquisition (e.g. 20KHz) is to send the parameters to the Master board at the beginning of simulation to allow continuos acquisition. Only at the end of simulation a table or a file with all acquisition points is generated. But in this case there is no real-time interaction with Simulink.

Another way to speed up is to use Visual C++ to implement a visual interface accessing the schedule management explained later.

IMPLEMENTATION USING SIMULINK

Even using the board as a real-time system it does not use the Real Time Workshop (RTW) toolbox from Mathworks. The strategy is to generate a DLL file by MEX function (a Matlab function) from C code source program.

The communication between the Simulink and the board are done by S-function. The Simulink calls these routines during the simulation process.

In a simple manner, the S-functions perform the following operations:

1. Initializing the SimStruct (a Simulink data structure);
2. Initializing the I/O device;
3. Calculating the block outputs:

3.1- Reading values from de I/O device and assigning these values to the block's output vector (if it is an ADC or a digital input);

3.2- Writing values from de block's input vector u, to an I/O device (if it is a DAC or a PWM)

4. Terminating the program (e.g., zeroing the DAC and PWM).

TASK MANAGEMENT

As we are working with a high-speed Master processor with a considerable amount of memory, it would be reasonable to minimize the interaction with the PC and allocate the whole operation in Master memory. For this purpose we are implementing a schedule manager. The main routines are being developed using C language and they are located at the 'C6x board memory.

Parameters such as routine call, data and the basic sample time are passed through the DMA memory using the HPI (Host Port Interface).

Example of parameters is given in figure 9.

Function	←function code
Length	←total register length (**n**)
Cycle	←step execution
...|↓Xint2 |↑Xint1 | ↓Xint1	←trigger association
Priority	←priority definition
Not used	←for expansion
@var1	←address of variable 1
@var2	←address of variable 2
:	
:	
@var **n-1**	
@var**n**	←last variable

Fig.9 Parameters for task management.

Before sending the parameters above, a header list with the function that will be used must be send. With these information the Master organizes its memory. This is done to know where it will put every parameter block.

The schedule of task management is shown in figure 10, where the basic step cycle is giving as 50us. The priority parameter of Routine#1 = 1, while the priority parameter of Routine#2 = 2, in this way, when both routines must be executed at same cycle, Routine#1 is executed first.

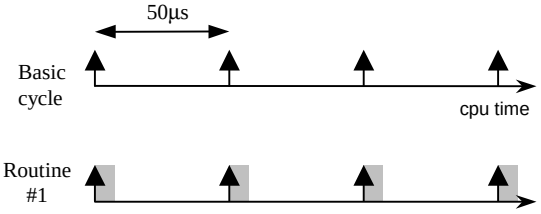


Fig.10 -Task schedule.

One issue here is overlap. This must be avoided, else unpredictable results can occur. The board returns a signal indicating the 'Overlap' in case this happens.

PROTOCOL BETWEEN MASTER/SLAVE

The access to the 'F240 is made by the SPI (8-bit length) through a simple protocol. Most of the functions are made by Master board sending data request for each function. In this way the Master has to wait for the answer, and only after that it should do another request.

The generic protocol structure is very simple, as shows the figure 11:

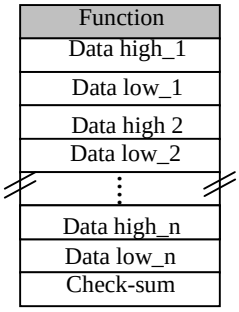


Fig.11 – Master/Slave protocol format.

These functions are implemented in C language for the 'C6x. The external interrupts of the 'F240 can also be enabled and disabled by the Master. The timer function is not implemented into the Slave board.

FINAL REMARKS

This work has shown a platform for digital control system using two DSP. The use of the Slave board has solved the common problem of peripheral limitation found in conventional DSP. The Master board has high performance capabilities and large memory that allows the implementation of complex algorithms.

The next challenges are to increase the library of hardware and software routines at the Master board and to generate code using the RTW software in order to use the whole resources of Simulink being all executed at 'C6x board.

ACKNOWLEDGMENTS

The authors would like to thank Grameyer Company by the partial financial support.

REFERENCES

[1] Hermely, Gründling and Pereida, "An Integrated Environment for Driving Induction Motors" (In Portuguese), Controle&Automação, Brazil, Vol.3, no.3, 1995.

[2] Douad, Lionel, Msc.Thesis, "Plate-Forme D'Essais: Commande de Machines Electriques", Université de Nantes-France, 1995.

[3] L.Ljung, System Identification. Theory for the User. Prentice-Hall International, 1987 (ISBN 0-13-881640-9 025).

[4] Spong, M.W.; Vidyasagar, M. "Robot Dynamics and Control", John Wiley and Sons, New York, 1989 pg. 19.

[5] Hsin-Jang Shieh and Kuo-Kai Shyu, "Nonlinear Sliding Mode Torque Control with Adaptive Backstepping Approach for Induction Motor Drive", IEEE Transactions on Industrial Electronics, Vol.46. no.2, pp 380 - 389, April 1999.