Low Cost Enhanced DSP-board for Real-Time Automation Systems

Alceu Heinke Frigeri and Romeu Reginatto
alceu@iee.ufrgs.br and romeu@iee.ufrgs.br
Federal University of Rio Grande do Sul
Electrical Engineering Department
Av. Osvaldo Araujo esq. Sarmento Leite
90.035-190 - Porto Alegre - RS - Brazil
Fone: +55 (51) 228 1633 ext. 3293 - FAX: +55 (51) 226 1171

Abstract

This paper presents some results obtained in the development of an enhanced DSP board for real-time applications in an industrial environment. The developed board is based on the TMS320C25 microprocessor and has support for both fieldbus protocols and a host interface. The present article focuses on the proposed and implemented architecture. The implementation of the proposed board was based on a EPLD technology, which improves hardware qualities as maintainability, extensibility, and reusability. The proposed DSP board suits both for final applications and for development purposes. This DSP board was initially developed to be applied on a field oriented induction motor control system.

1 Introduction

Modern industrial automation systems are characterized by being huge complex systems with a large number of concurrent software and hardware devices. The project and development of real-time automation systems is a complex, time demanding and extremely expensive activity ([2] e [4]). This is the main reason that one should intend to systematize the project of such systems in order to build up more robust, flexible and reusable systems [5].

The constant decreasing costs of microprocessors-like devices is allowing simultaneously the construction of more powerful distributed digital control systems as well as the construction of smart devices (intelligent sensors and actuators) ([6] and [7]). On the other hand, in the developing of a real-time control system, one is easily faced with huge processing demanding tasks ([16] and [3]). For this class of systems, general applications computer are of no help at all, being necessary the development of dedicated processing devices ([11]). To interconnect these dedicated devices, becomes important to use standards industrial protocols (known as fieldbus protocols), like the ones proposed by ISO, in order to be feasible to interconnect devices from different companies in an industrial medium ([8], [9], [10], [11] and [12]).
This paper focuses on the construction of modular and reusable hardware device, for applications that demand huge digital DSP-like processing capabilities and are embedded in real-time automation systems ([4]). This class of applications are characterized by being ([1] and [16]):

- **Algorithm intensive**: the control algorithm demands a huge volume of operations to be executed in order to control the physical associated device.

- **Timed constrained**: associated with the control algorithm there are time constrains imposed by the physical device being controlled. It normally requires that both the embedded software and the hardware to be deterministic, i.e., to have deterministic time behavior even in presence of exceptions.

- **I/O intensive**: in order to control (and monitor) the physical system, usually a large number of "physical quantities" must be handled simultaneously. Here comes a cost bottleneck, since the required I/O facility (mainly analog ones) are presumed to be large and extremely application dependent. Consequently, it is hard to have a cost effective general solution.

- **Communication intensive**: the control device would be part of a larger plant, having to interact with other devices, since it is not supposed that it will be a stand alone system. The common approach is to use a standard protocol (fieldbus), minimizing eventual compatibility problems.

Altogether, those constrains imply that the implemented control program in the DSP-board is constrained by the control algorithm, the nature of the I/O system (which lies on the physical process being controlled) and the intercommunication process. The first two above characteristics are the ground for the use of a DSP processor, since they are the ones that deliver the necessary computing power.

The construction of the hardware module presented in this paper was directed towards the field oriented induction motor control problem. This application summarizes all the above characteristics, mainly when the induction motor is just a small part of a plant or system.

This paper is organized as follows: In the next sections a modular solution for the implementation of a DSP-based control system with embedded communication facilities and extensible I/O facilities is presented. First an overview of the proposed architecture is done, followed by a further description of each modules. In the section three the actual implementation of the proposed architecture is presented, based on an EPLD technology. In the last section the conclusions of this work and some future directions are presented.
2 Architectural Overview

The main goal of the proposed architecture was to implement a DSP-processing board that could be fine tuned for a specific application and, at the same time, not relying on any specific application requirement (following the principles after Parnas [23]). Following the same principle, another goal was to do this independently of the supported fieldbus protocol. So the project of this modular DSP-processing board, was based on these 'ad hoc' requirements:

- System scalability and expandability
- I/O flexibility and scalability: it should not rely on any specific application
- Independence of the supported fieldbus protocol
- Independence of the host computer (if any)
- Support for software and hardware prototype development

As a "modular DSP-board" one should think in a stand-alone processing board capable to be straight connected to other processors boards of the same type and or to a host computer. The former attempts to support the system scalability and expandability requirements while the last attempts to support or aid the development of specific applications. In the same way the board's facilities to support a fieldbus protocol as well as I/O communication should not lie on the board itself, being supported by a daughter board connectable to the main DSP-board.

In Fig. 1 one has an overview of the proposed architecture. This architecture is based on the following premises:

- The interface with the host computer (if any) do not lie directly on the DSP-board.
- The I/O facilities as well as the fieldbus facilities, in the same way, are not part of the DSP-board.
- The connection between DSP-boards (of the same type or not) is made through a memory interface board, based on an arbiter memory access mechanism.

This architecture has the following advantages:

1. The interface board works as a hardware isolation layer, allowing the DSP-board to be connected to a broad number of host computers [1].
2. The same philosophy is applied to the fieldbus connection, allowing the connection/use of a wide number of fieldbus protocols.
3. The memory interface board supports a fast and simple communication channel between different DSP-boards at microprocessor speed ([1]). It is so since there is no interface layer between the memory system and the DSP-microprocessor, but the arbiter mechanism.

4. The I/O sub-system can be straight connected to the DSP-board and, at same time, can be fine tuned for a specific application.

5. The system costs can be hold down, since the central processing unit is generic enough for scale production, as well as the specific interfaces (like the host interface board and the fieldbus board). The only application specific part of the system becomes the I/O board.

![Diagram](image)

**Fig. 1 An overview of the proposed architecture.**

As the DSP-board is supposed to support stand alone operation, the interface board is fundamentally optional. Besides that, it is useful during early development stages: either of a specific I/O facility board or a specific application software. Further, the interface board is also useful in interconnecting the DSP-board to a host computer for the sake of providing huge media storage and, perhaps, an friendly user interface.

In the actual implementation of this architecture, an IBM-PC-AT-486 compatible class computer, issuing an ISA16 bus ([18]), was used as host computer. The DSP-board was projected based on a TMS320C25 DSP-processor from Texas Instruments ([15], [16] and [17]). The Fieldbus Board was based in the 80C592 microcontroller from Philips, which is an embedded 8051 compatible microcontroller with
hardware support up to the 2nd layer of the CANBUS protocol ([10], [13] and [14]). In the following subsections each board of the system will be outlined.

2.1 Standalone DSP board

The actual implementation of the DSP-board was based on the TMS320C25 from Texas Instruments ([15]), which is a fixed point 16bit DSP-processor, delivering 10Mips. This processor has 64K words memory space address (separate code and data spaces), and 16 words I/O space. In Fig. 2 one can see the basic architecture of the DSP-board and in Table 1 and Table 2 the chosen address mapping schema is outlined.

The main policy, in defining this architecture, was that the board should have:

- at least, two connectors for extension/communication purpose;
- only one I/O connector;
- a host system interface connector and
- an shadowed bootstrap EPROM.

Having a minimum of two expansions connectors, it would be possible to interconnect various DSP-board, in a pipe fashion ([1]), when necessary, to deliver the needed processing power for more demanding tasks. On the other hand, one I/O connector is just enough, since the I/O system is, by definition, application dependent and was chosen to be outside the DSP-board. It does not represent any performance/architecture constraint, since the specific I/O sub-system can be fine tuned for the desired application and then connected to the DSP-board through this I/O connector.

As the DSP-board is intended to be able to operate in a standalone mode, it is necessary to have a bootstrap EPROM (or alike) memory bank. But, at same time, as we are concerned about performance, it becomes important to have such a mechanism like memory shadow ([1]). In the proposed architecture, these was implemented as a „memory bank switch“. In the „mode 0“ the DSP-processor can access the bootstrap EPROM memory and part of the RAM memory. In the „mode 1“ the DSP-processor can access all the RAM memory, but the EPROM memory. So it is possible to run the EPROM code at full speed by transferring it into the RAM memory, in mode 0, and then switching to the mode 1.
The I/O map assignment was a problem by itself, because of the extremely limited I/O space of the TMS320C25 processor (only 16 words) ([15]). One option could have been to use a memory mapped I/O policy, with the expense of more costly control logic. The chosen solution was to "expand" the I/O space through a minimal control logic: similarly to the memory sub-system, there are four I/O operating modes. In each of them, the DSP-processor can directly access 16 I/O addresses. The 16 I/O addresses of the mode 0 are already assigned, being the others I/O modes free of assignment for use in daughter boards connected to one of the expansion buses. This design policy ease the implementation and the connection of daughter boards as, for example, the fieldbus interface board, making them seamless to the already implemented system.
Table 2 I/O address Mapping of the DSP-board

<table>
<thead>
<tr>
<th>Address</th>
<th>Device</th>
<th>Modes 1-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-3</td>
<td>Timer</td>
<td>Depends on the board connected to the expansion buses</td>
</tr>
<tr>
<td>4</td>
<td>Host System I/O register</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Expansion 1 I/O register</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Expansion 2 I/O register</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Control Word</td>
<td></td>
</tr>
<tr>
<td>8-F</td>
<td>I/O connector</td>
<td></td>
</tr>
</tbody>
</table>

2.2 Interface Board

In Fig. 3 one can see the actual "Design Architecture" of the Interface Board. It is based on a very simple memory-register interrupt oriented protocol. In such way a minimum functionality is allowed:

- the DSP-board can interrupt the host processor,
- the host processor can interrupt/reset the DSP-board,
- the host processor can down-load an specific code directly in the DSP-board main memory,
- the host processor and the DSP-board can exchange some data through 2 dedicated I/O ports.

![Fig. 3 Architectural overview of the Interface Board.](image)

Such minimal architecture was chose because, for the first intended application (a field oriented control of an induction motor), the DSP-board should operate in a standalone mode, being connected to the rest of the system, only through the fieldbus protocol. So, for the developed process it was implemented only a minimal version of a interface board.
Through the "DSP connection bus" one has a direct access to the memory bus of the DSP-board, issuing a hold/acknowledge protocol: each intended host access to the DSP-board main memory is deferred, through a hold signal, until an acknowledge from the DSP-board is given. This mechanism allow an easy way for the host computer to download code in and monitor the DSP main memory. As an secondary, but faster, communication facility there are two I/O 16 bit registers that can be used for information exchange, where both the host computer and the DSP board can read and write without holding each other. As an additional facility (important when acquiring huge amount of data), there is the possibility for the DSP-board and the host system to interrupt each other. Finally, in this actual implementation there is no provision for the DSP-board to communicate with the host computer (as a master), but through the I/O registers.

2.3 Communication Board

In the Fig. 4 one have an overview of the proposed architecture for the communication board to be attached to one of the expansions busses of the DSP-board. The communication board could be kept extremely simple due to the 80C592 microcontroller from Philips ([14]), which is a 8051 compatible enhanced microcontroller with full support for the CANBUS protocol ([10], [11] and [12]). At the same time, as this board is only intended to be a communication facility with no need of further elements, but two memory banks: one non-volatile (EPROM) and one volatile (SRAM).

![Fig. 4 Overview of the CANBUS-board schema.](image-url)
2.4 Memory board

The main purpose of the Memory board is both to provide a mass memory medium for the DSP-board and, at same time, to provide a seamless communication channel (memory based) between two DSP-boards ([16]). In Fig. 5 one have an overview of the intended board. The access to the main memory is controlled through an arbiter logic, which ensures that only one DSP-board would have access at once to the main memory. The access is controlled in a per access base, and there is no provision for any other hardware based semaphore policy.

![Fig. 5 Overview of the Memory-board architecture.](image)

2.5 I/O board

The I/O board is the only part of the proposed architecture that is application dependent. Here is presented the proposed I/O board architecture for the issued application: field oriented induction motor control. In the Fig. 6 an overview of the implemented I/O board is showed.

![Fig. 6 Overview of the I/O board architecture.](image)
The I/O board interfaces the DSP board with the physical variables, of the process being controlled, through:

- 4 independent 8 bits digital to analog channels with smooth filters;
- 4 multiplexed analog inputs with 12 bits resolution;
- One encoder signal conditioner and digital logic for pulse counting.

3 Implementation

All the boards were implemented using EPLDs devices from ALTERA Inc. (MAX7000 family) ([19], [20] and [21]). It helped keeping the designed board small in size. At same time, as the chosen EPLDs were fast enough, there was also no performance compromise. Another advantage of this PLD-oriented implementation is that the system can be easily upgraded, and as long as the chip account is kept small the system maintenance is also simple.

The control logic of the diverse boards was implemented using the MAXPlusII™ version 3.1 EPLD design software from ALTERA Corporation. Basically, this CAE-tool supports the description, simulation, verification and programming of the control logic (based on a schematic capture tool, VHDL description or even flow chart). The process of choosing the PLD device (including the partition and programming of each device) can be fully automatic. Only two problems were faced due to the EPLD implementation:

- the number of required pins and
- the logic circuit partition.

Both where related with the fact that the MAX7000 devices, besides powerful, are not I/O intensive oriented, but more state machine implementation oriented. For instance, the interface board is basically a bridge between two microprocessor buses, which rapidly increases the required number of pins. The only solution was the use more than one EPLD device. Such solution, however, introduced the necessity of dividing the logical circuit into parts. The automatic partition performed by the MAXPlusII™, as showed in Table 4, resulted in two different EPLD devices with a high number of interconnected pins. In order to obtain a more cost effective solution, a fully manual partition was performed. The general guidelines in this process were: first, try to keep small the number of devices, and second, try to use the same EPLD device for all the control blocks (to minimize implementation costs). The result of the manual partition is shown in Table 3 which presents the number and type of EPLD devices actually used in the implementation of the interface board and the standalone DSP board.
Table 3 Used EPLD devices in the system implementation

<table>
<thead>
<tr>
<th>Board</th>
<th>EPLD</th>
<th>cells</th>
<th>useful pins</th>
<th>used cells</th>
<th>used pins</th>
<th>used pins for interconnection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Board</td>
<td>EPLD7096LC84</td>
<td>96</td>
<td>64</td>
<td>27 28%</td>
<td>64 100%</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>EPLD7096LC84</td>
<td>96</td>
<td>64</td>
<td>72 75%</td>
<td>60 93%</td>
<td>29</td>
</tr>
<tr>
<td>DSP-board</td>
<td>EPLD7096LC84</td>
<td>96</td>
<td>64</td>
<td>44 45%</td>
<td>64 100%</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4 Suggested partition by the automatic tool

<table>
<thead>
<tr>
<th>Board</th>
<th>EPLD</th>
<th>cells</th>
<th>useful pins</th>
<th>used cells</th>
<th>used pins</th>
<th>used pins for interconnection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Board</td>
<td>EPLD7192QC160</td>
<td>128</td>
<td>100</td>
<td>61 47%</td>
<td>92 92%</td>
<td>49</td>
</tr>
<tr>
<td></td>
<td>EPLD7096LC84</td>
<td>96</td>
<td>64</td>
<td>39 40%</td>
<td>55 86%</td>
<td>49</td>
</tr>
</tbody>
</table>

4 Conclusions

The proposed architecture is a cost effective solution for problems demanding huge processing power in an industrial automation system, being scaleable (through expansions buses). One key issue of the proposed architecture is its independence of the supported fieldbus protocol, since it is supported by a daughter board. In that way, the target communication protocol can be easily changed without affecting the DSP-board design.

Another important issue of the proposed architecture is the interface board, which implements a hardware isolation layer between the DSP-board and the host system. In the actual implementation, a minimal version was implemented, since the main bus of the host system (an IBM-PC-486 computer based on an ISA16 bus [18]) does not, easily, support multiprocessing.

One further improvement in the implemented system, can be to implement an interface board to a multi-processor bus like the VME, which would allow two kinds of parallelism in the proposed architecture:

- a general multi-processing parallelism throughout a VME alike bus and
- a dedicated multi-processing parallelism through the memory expansion mechanism.

Finally one main concern, in the implemented system, was to make the DSP-board independent from the intended application. It was acquired letting the I/O sub-system apart from the main DSP-board.
5 References


