ilmedia

TECHNISCHE UNIVERSITÄT ILMENAU

Däne, Bernd; Berger, Falk:

A multiprocessor DSP System for a high throughput control application

Publikation entstand im Rahmen der Veranstaltung:
EDERS-2004: The European DSP Education and Research Symposium, 16th November 2004, Birmingham, UK

A Multiprocessor DSP System for a High Throughput Control Application

Bernd Däne *(bernd.daene@tu-ilmenau.de)*
Falk Berger *(falk.berger@tu-ilmenau.de)*

Ilmenau Technical University

P.O. Box 100565, 98684 Ilmenau, Germany
http://tin.tu-ilmenau.de/ra/

Abstract

This paper describes the hardware concept and realization of an experimental multiprocessor system with Digital Signal Processors TMS320C6701. For convenience and efficiency piggyback modules have been used. The structure is organized in master-slave manner. The slaves are accessed from the master through their host port interfaces. Special address decoding provides broadcasted write from the master to all slaves besides individual read and write access. Some experimental results are provided.

1. INTRODUCTION

This contribution describes concept, realization and experimental results for a multiprocessor system with up to six DSPs (Digital Signal Processors) from the TMS320C6000 family. It is a low-volume experimental project that shall provide a platform for researching multiprocessor hardware and software.

Additionally it is a first step for developing a control system that can be used in a high precision measurement machine [1]. Here the system will calculate complex control algorithms for control loops at high rate. High throughput and low latency are required.

For communication purposes a High Speed USB interface, controlled by a separate DSP, is provided. Sensors and actuators are accessible via input/output ports that are located at a parallel bus.

Since both the research interests and the measurement system deal with high performance operation high priority is given to fast and efficient data transport between the processor nodes.

2. HARDWARE CONCEPT

Processor nodes

The processor nodes are based on off-the-shelf piggyback modules [2]. Each of these modules is equipped with a floating point DSP TMS320C6701, static and dynamic memory, flash memory, bus interface, serial interface, core voltage and clock supply, and some other functions. In this configuration one such module autonomously can act as a microcomputer, requiring only one 3,3 V power supply.

For interfacing with other components the modules provide both the asynchronous EMIF (external memory interface) bus interface where the DSP acts as a master and the DSP's HPI (host port interface), where it can be accessed as a memory slave.

For all signals that leave the module the latter has bus switches rather then drivers or direct connection to the processor pins. The bus switches provide isolation during cycles that access on-module components and tolerance against 5V-TTL-levels from outside. In contrast to drivers they do not increase the load capabilities for output signals but do not (or not so much) contribute to signal delays.

The use of these modules reduces the development efforts and has economical advantages for this low-volume project. Similar modules with (nearly) matching pinouts are available with other members of the C6000 processor family.

Communication structure

It was decided to use a bus-like communication structure with one master and four slaves. The master is responsible for communicating with the sensors and actuators through input/output ports, for distributing data to the slaves (most likely at fixed high rate) and for collecting data from them (control algorithm outputs, for instance). The slaves run control algorithms

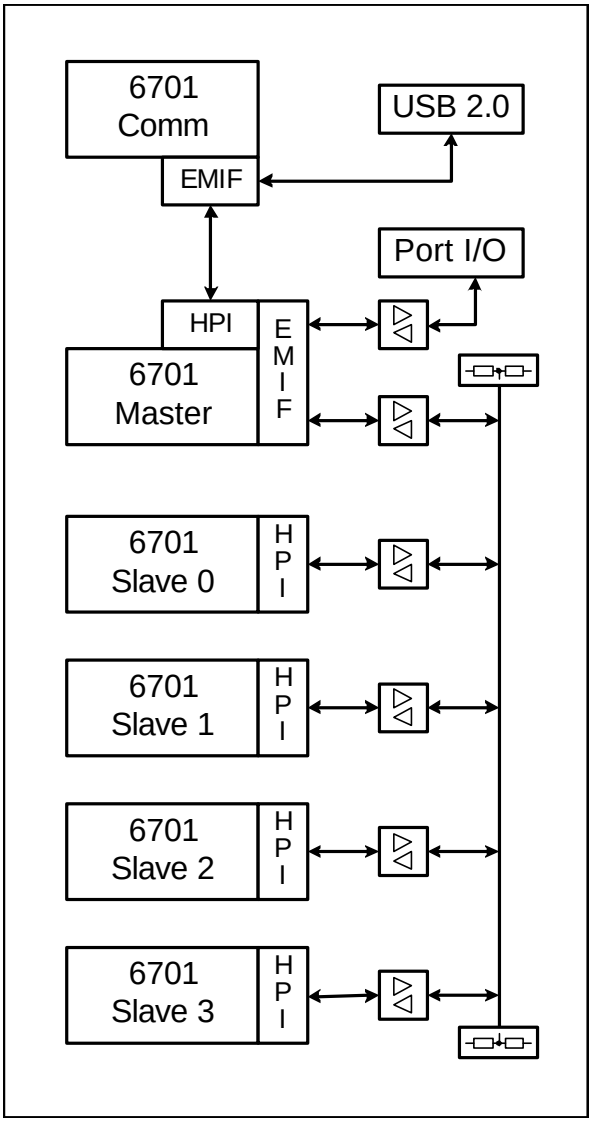

Fig. 1: Overall hardware structure.

at full speed. So a closed-loop control system can be realized that supports multiple control loops with some interaction between them.

Since such a system uses rate-monotonic sampling most communication and processing occurs in rate-monotonic manner with fixed deadlines. In the project it is to be studied what rates are achievable by such a structure.

The slaves are accessed via their host port interfaces only. This removes all communication overhead from them and enables them to use all of their processing time for calculating algorithms. Input data will be written into and read out of predefined locations in the slaves' internal memories by the master. The master itself uses its EMIF interface for accessing the slaves.

Another processor module ('Comm') manages external communication using a high speed USB connection. Besides data transfer and formatting it has to process compression algorithms in order to speed up outgoing transport of large amounts of data. This processor is connected to the master by a separate path. It uses it's EMIF interface to access the other processor's HPI interface as described in [3].

Figure 1 illustrates the hardware structure as described above. It shows main blocks and busses, but not minor components and control logic. The system is to be implemented on a 400 mm by 175 mm printed circuit board (four layers). Input/output ports and USB hardware are actually not part of the design. The former is located at another board that is attached with a bus connector, while the latter is located at another piggyback module that is stacked atop the according processor module.

Master address ranges

The master processor actively accesses numerous blocks and devices. At first there are ports and memory blocks at the module itself. These are defined by the module manufacturer [2]. Next there are addresses for the memory-mapped input/output ports.

But the interesting part is the addressing of the slave modules. Due to the principal function of the HPI they appear to the master similar to memory devices. Each slave is assigned to a memory range that the master can use to perform individual read and write cycles from or to this slave.

From the surrounding project it turned out that most data that is to be transferred from the master to the slaves will have to be duplicated to some or all slaves. This derives from the coupled control loops, where most of input data has to be processed by some or all controllers. This property applies to write cycles of the master only.

For the reason mentioned above a special 'broadcast' feature has been implemented for write operations from the master to the slaves. This function allows the master to write simultaneously into memory locations of all slaves. Timing behavior during such cycles is virtually equal to normal write cycles, saving much time in contrast to accessing all slaves in sequence.

Additional functionality to access an arbitrary subset of the slaves is not realized because it does not contribute to time savings since the cycle would not shorten. This is true when assuming that DSP internal memory is large enough to contain all data and that

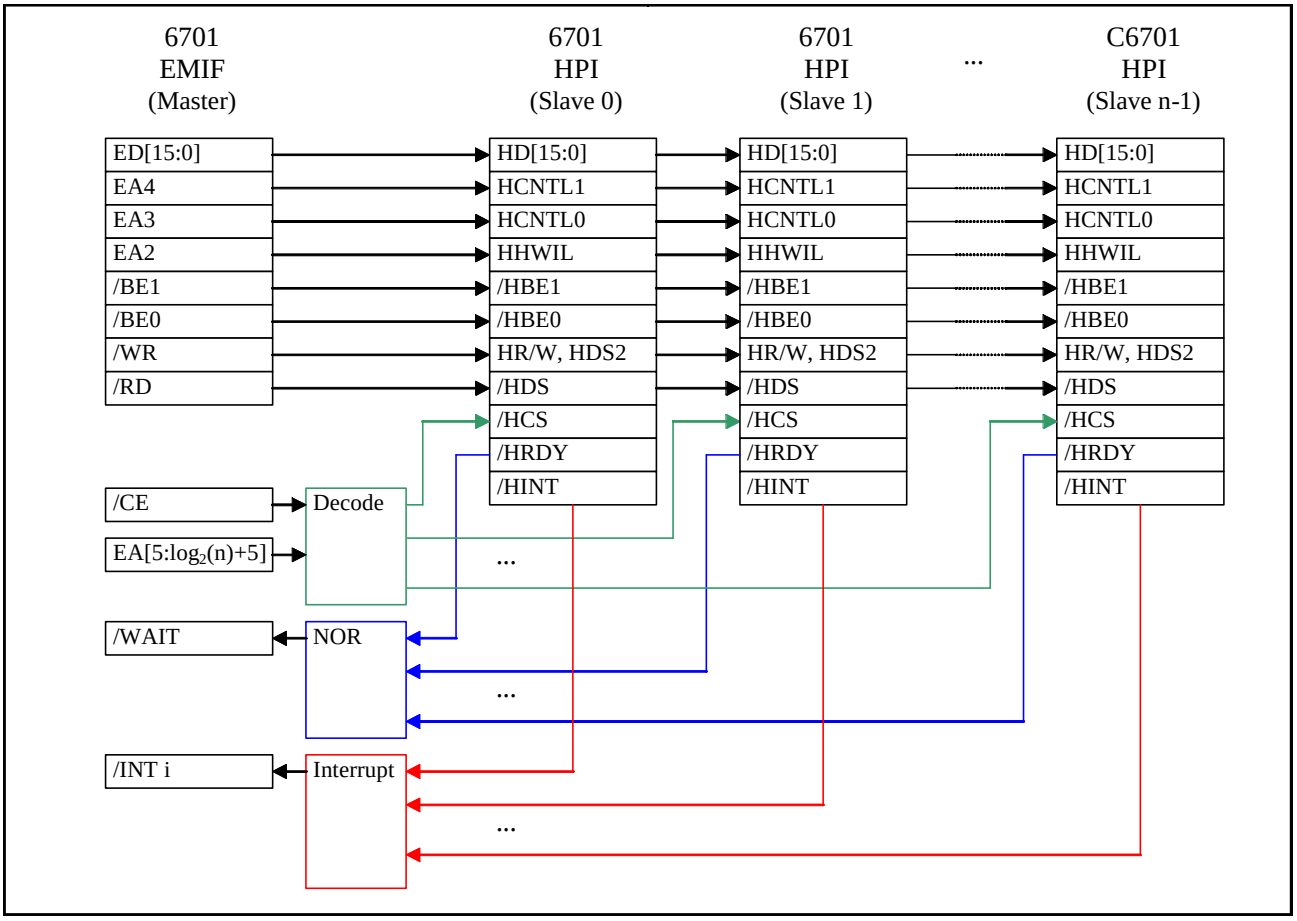

Fig. 2: Control signals (simplified).

on-chip memory conflicts will not significantly increase.

The broadcast functionality is assigned to a dedicated address range so the master can choose one or all slaves by address variation. A special handling of the control signals during broadcast cycles had to be implemented.

Control logic

Address decoding logic handles the different situations, generates the appropriate control signals and activates the data paths needed. In particular the numerous bus drivers must be controlled in activity and direction. To reduce electrical problems all parts should be disabled that are not used by the current cycle.

Figure 2 shows additional control logic (simplified) that handles the cycles of the master-slave-communication [4]. The address decoder generates the appropriate chip enable signals for the slaves as mentioned above. The WAIT signals from the slaves' host ports must be merged in order to handle situations when in broadcast mode one of the slaves requires more time then the others (due to internal access conflict).

Electrical Considerations

Since the electrical paths of the master-slave-bus extend over nearly 300 mm due to the dimensions of the modules and other requirements it was necessary to carefully design their configuration. Otherwise the bus cycles would have to be slowed down in order to achieve reliable operation.

It was decided to use impedance-controlled stripes with parallel resistor termination at both ends. An impedance of 68 Ohm has been chosen, controlled by the path width on the circuit board. The lines are terminated by 100 Ohm to Vcc and 220 Ohm to GND at each end [4].

Since the modules' outputs are not able to drive the currents needed, drivers had to be included near each module. These drivers are visible in figure 1. The devices were chosen from the BiCMOS logic family

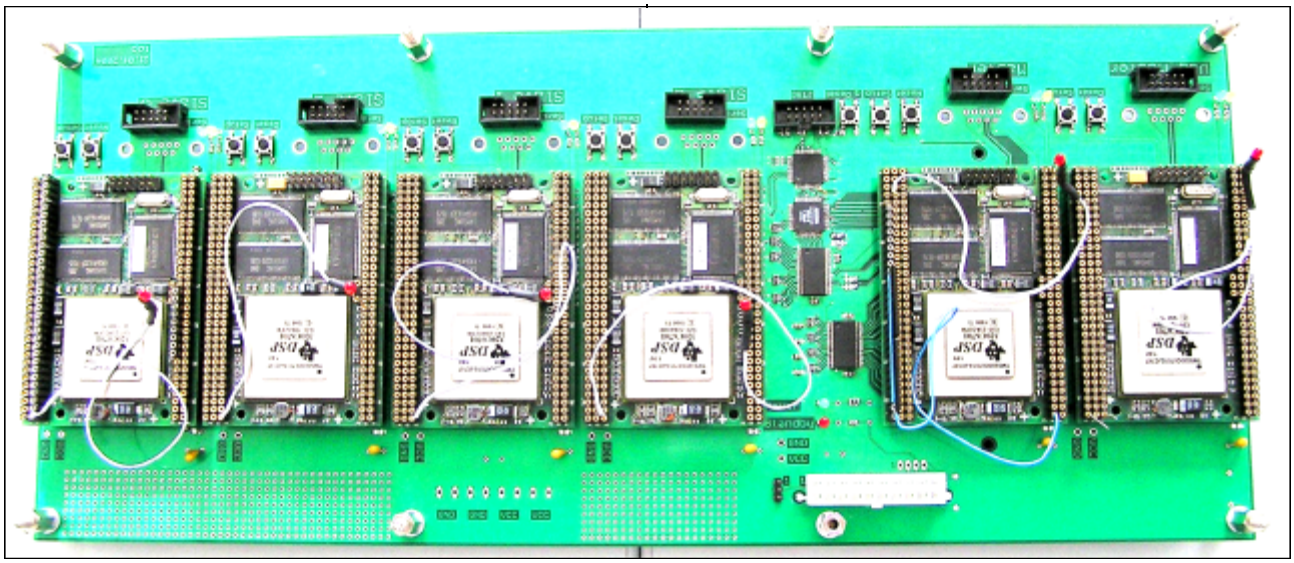

Fig. 3: Board (top view). Modules from right to left: Comm, Master, Slaves 0 through 3.

74LVT, delivering the output current needed and providing short delays.

Additional functions

Additionally the hardware supports a number of minor functions. Among them is a global reset signal for all modules. Interrupt signals from the slave modules and the input/output ports can be merged in order to fit the limited interrupt channels of the master module. All control logic is implemented into two FPGA devices located at the board.

3. SOFTWARE CONCEPT

Control and supervision of the whole system is concentrated in the master node. Since it has to do numerous data transfers from and to the slaves and the input/output ports it was desirable to rid its processor core of stalls due to waiting bus cycles [5]. This is done by using DMA channels for most of these transfers.

A set of basic communication procedures that support DMA access has already been implemented, supporting data transfers with and without acknowledge between the master and the slaves. Control of the broadcast feature is included. It is planned to extend the software into an application interface that includes higher protocol levels and is ready for usage in larger software projects.

Since the first task was to evaluate the communication system's performance and reliability special test software has been developed. It uses the basic communication procedures as mentioned above and provides sustained data transfers for all directions and all modes. Data integrity is always checked. This software allows measurement of communication rates and long-duration testing of data reliability.

4. EXPERIMENTAL RESULTS

Figures 3 and 4 show the printed circuit board during evaluation. On the top side (figure 3) the DSP modules and some control logic can be seen. On the bottom side (figure 4) the long stripes of the master-slave bus are visible (left part). The connector at the right side attaches to the board with input/output ports.

The timing of the master's bus cycles when accessing the master-slave-bus has been tuned experimentally. It is controlled by the C6701 processor's timing registers [5]. At 167 MHz clock frequency, a 9-9-3 figure (setup-strobe-hold clock cycles) has been reached. It turned out that the strobe duration was most critical because WAIT signals must reliably be recognized.

Measured data rates are listed in table 1. It can be seen that broadcast mode does not differ from individual access to the slaves, just as expected. The HPI's burst mode provides substantial advantage for large data blocks. It may be possible to further improve the timing behavior by applying minor changes to the design.

With the interface from the 'Comm' processor to the master the timing is much better, because no drivers are involved and the lines are short. This is the standard situation for EMIF-HPI interfacing [3] and does not need further explanation.

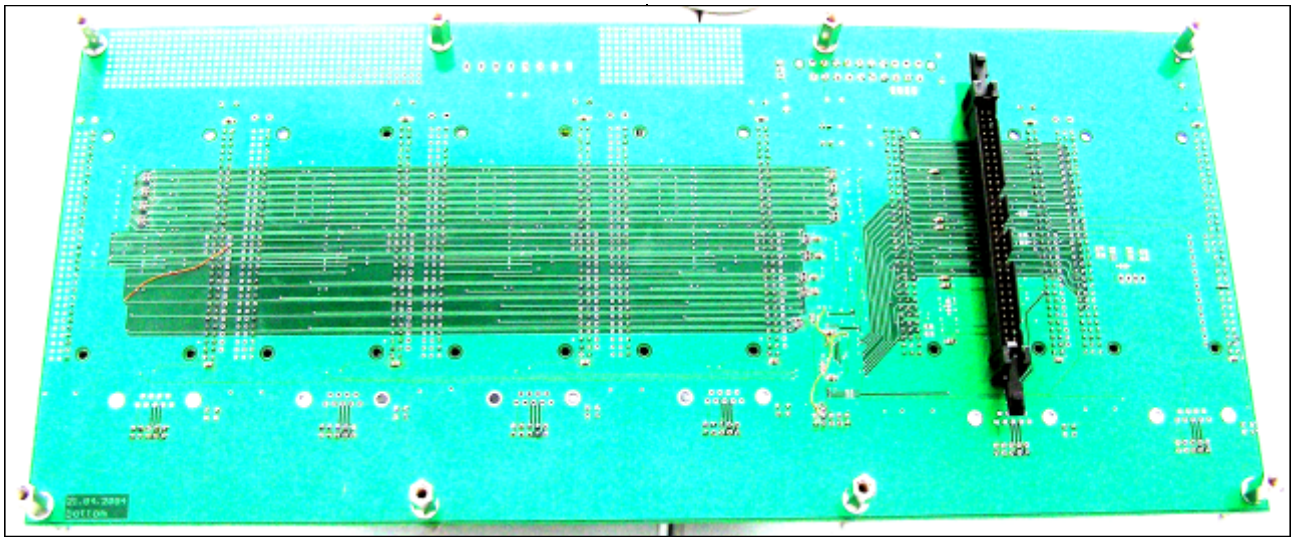

Fig. 4: Board (bottom view).

5. CONCLUSION AND FURTHER WORK

The developed system is being evaluated and will further be used in the project. Meanwhile software development and some fine tuning in hardware occur. Future work will include the extension to other DSP family members and the additional utilization of the processors' multichannel buffered serial ports for improving interprocessor communication performance.

6. ACKNOWLEDGEMENT

This work is supported by Deutsche Forschungsgemeinschaft (German Research Council) under SFB 622.

Direction and Mode	**Data Rate** (MByte/s)
Read from slave	6,1
Read from slave (burst)	13,3
Write to slave (burst)	14,5
Broadcast write to slaves (burst)	14,5

Tab. 1: Measured data rates.

References

[1] Wolfgang Fengler, Bernd Däne, Vesselka Duridanova: Design Methodology for an Embedded System for High-Performance Computing. In: Colnaric, Adamski, Wegrzyn (Eds.): Real-Time Programming 2003, Proceedings of WRTP 2003, 27th IFAC/IFIP/IEEE Workshop on Real-Time Programming, pp. 99-104, Lagow, Poland, May 14-17, 2003.

[2] D.Module.C6x01: Technical Data Sheet. Document Revision 2.1. D.SignT Digital Signalprocessing Technology GbR, Kerken, Germany, June 2002.

[3] Zoran Nikolic: TMS320C6000 EMIF to TMS320C6000 Host Port Interface. Application Report SPRA536B. Texas Instruments Incorporated, September 2003.

[4] Alexander Pacholik: Concept and Realization of a Multi-DSP-System and its Interfaces. Diploma Thesis, Ilmenau Technical University, Department of Computer Architectures, Ilmenau, Germany, July 2004.

[5] Naim Dahnoun: Digital Signal Processing Implementation Using the TM320C6000 DSP Platform. Prentice Hall PTR Indianapolis, USA, 2000.