An Example of Practice Based Engineering Education : the Design of a Microprogrammed MIPS Processor with the Alliance CAD System

Marie-Martine PAGET E-mail: Marie-Martine.Paget@lip6.fr

Pirouz BAZARGAN SABET E-mail: Pirouz.Bazargan-Sabet@lip6.fr

Alain GREINER E-mail: Alain.Greiner@lip6.fr

University of Paris 6, LIP6-ASIM Laboratory, http://www-asim.lip6.fr
BP 167, 4 Place Jussieu, F75252 Paris Cedex 05, France

ABSTRACT
This paper describes a VLSI project that aimed at the design of a microprogrammed Mips R3000 microprocessor. The project starts from the functional specifications of the circuit and goes through several design steps to end with the obtaining of the factory masks to be send to the foundry. The project has been defined in the framework of a postgraduate course of VLSI design at the University of Paris 6 (Pierre et Marie Curie) and uses the public domain Alliance CAD System developed at the same university. Each team of five students, advised by a researcher, was in charge of designing a processor within a period of three weeks. This project comes after a set of four courses during which a design methodology and the usage of the Alliance CAD System has been detailed.

KEYWORDS: Computer Tools for Engineering Education, Computer Aided Teaching, MIPS R3000 Processor

INTRODUCTION
Teaching Computer Science and Computer Architecture covers a great number of fields ranging from artificial intelligence, data base management, networks to domains more closely related to electronic design such as VLSI or integrated analog design. In University of Paris 6 (Pierre et Marie Curie), the courses of Computer Science and Computer Architecture are proposed to students within a formation of two years for Master graduation and an additional specialized year for post graduation. A great effort has been carried out in the Department of Computer Science to preserve a global coherence between the difference courses all along the 3-year formation.

The Mips R3000 processor has been chosen as the practical support of several courses within the Computer Science and Computer Architecture formation. The architecture of the Mips R3000 is simple enough for teaching the basic concepts of computer architecture and still enough complete and realistic to introduce more advanced features such as pipelining, caching, operating system development, performance versus complexity and etc. In university of Paris 6, this processor is used for teaching compiling techniques, microprogrammed processors' architecture, computer architecture and VLSI design. As mentioned by Waldron in [6] "Mips is the preferred choice for teaching computer architecture in 2000's, just as the Motorola 68000 was during the 1980's".

In this paper, we describe a project proposed to post graduate students in Computer Design. The projects consists in the implementation of a microprogrammed version of the Mips R3000 using a CMOS technology. Since the main objective of the project was to provide a practical example of VLSI design to the students, a microprogrammed version of the processor has been preferred rather the a pipeline version with higher performance but much more complex to understand and to lay down on silicon.

The project is an example of practice based engineering education. It represents the result of the experiences accumulated by the staff members during several years of computer aided education in the field of microelectronic design. Its goal is to complete the traditional teaching by giving the students the possibility of playing an active role in the learning process. The overall objective is to "educate students in design competence and give the knowledge, skills and attitudes leading to relevant professional competence" as explained by Aas [1] or, to "help students to understand the most up-to-date technologies together with the methods and tools they will use in their future industrial positions" as reported by Tchoumatchenko [5].

We will first present a short summary of the Mips internal architecture and then focus on the methodology followed by the students for the design of the processor. At each step, we have tried to involve the learner in a creative task to increase his motivation and to stimulate his thinking capacity by an active participation to a real team work associating analysis and synthesis.

MIPS INTERNAL ARCHITECTURE

The internal architecture of the chip is depicted in Figure 1. The core is composed of a data path and a control part. The data path includes registers and operating units and performs elementary data transfers between source and destination registers in one clock cycle. The data path is controlled by the control part.

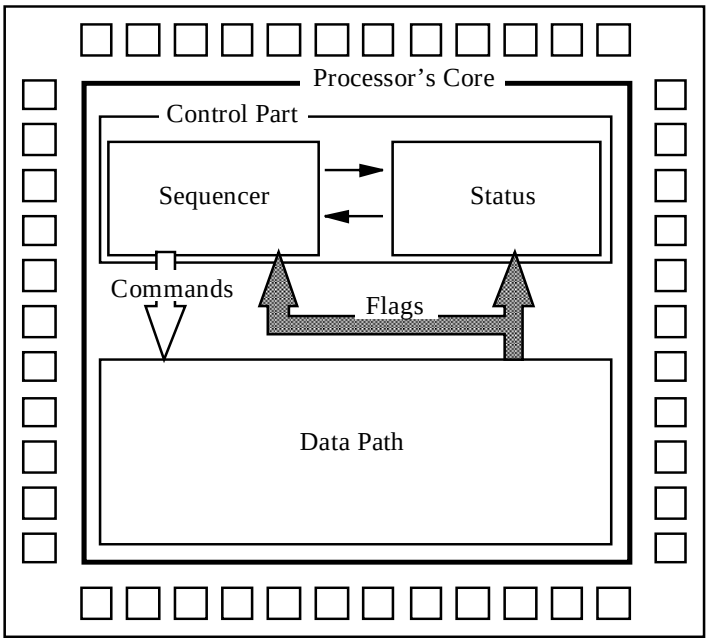

Figure 1: Mips Processor's overview

The control part is split into two separate blocks: the sequencer and the status block. At each cycle, the sequencer (a finite state machine) goes to a new state. The state calculated by the sequencer can be seen as a micro instruction that defines all the operations that must be performed at the next cycle. A part of the same micro instruction is used by the sequencer to determine the next state. An other part is send to the data path. The status block is mainly in charge of managing the interrupt and exception mechanisms during the execution of instructions. It receives external events such as reset, interrupt requests, etc. and informs the sequencer through a set of flags.

The interface between the data path and the control part is composed of two sets of signals. A decoded micro instruction word calculated by the sequencer provides all the commands to the data path. In turn, a set of flags such as overflow are calculated by the data path and notified to the sequencer. These flags are used by the sequencer to define the next state in regard of the particular situation that has happened in the data path.

A set of pads are placed around the chip's core to drive the external signals.

THE DESIGN METHOD

The design methodology of the processor is a top-down oriented approach. The design starts from the specifications of the circuit and goes through several steps to end with the description of the masks send to the foundry.

The project follows the methodology and the design tools provided by the public domain Alliance CAD System [4]. Alliance is a complete set of CAD tools and portable libraries for VLSI design developed at the Computer Science Department of the University of Paris 6. A detailed description of Alliance CAD Sytem can be found in Alliance [2].

The VHDL language is used to write the specifications of the processor at the RT level (Register Transfer Level). Again the VHDL language is called, at the different steps all along the design, to build the description of the circuit at the structural level (interconnection of blocs or cells).

The design of the circuit is based on a standard cell approach. Three separate libraries are provided by the Alliance CAD System. SXLIB is used for random logic design (control part). DP_SXLIB contains a set of cells and operating unit's generators that can be placed and routed in regular blocks (data path). PADLIB provides a set of input-output pads to connect the core to the external signals.

DESIGNING THE CONTROL PART

The design of the control part and the data path does not follow the same scheme. When designing a circuit, the first step is to split the circuit into several parts. Each part, is then designed separately following a given scheme. The partitioning is done manually. The design hierarchy of the Mips processor is presented in Figure 2.

In our project, given the shortness of the time, a first partition of the circuit into control part (composed of the sequencer and the status block) and data path has been given to the students.

The sequencer is written as a FSM (Finite State Machine). A specific design tool (FSM Synthesizer) translates the description of the finite state machine into an RTL description by giving a binary code to each state. The resulted RTL code is the reduced by a Boolean optimizer and synthesized into a netlist of gates using the SXLIB standard cells library. The design of the status part follows the same scheme.

The place and route represent the last step. First the two netlists are merged. Then, a scan path is inserted to increase the testability of the block. Finally, the gates that compose the resulted netlist are placed and routed using a standard cells place and route tool.

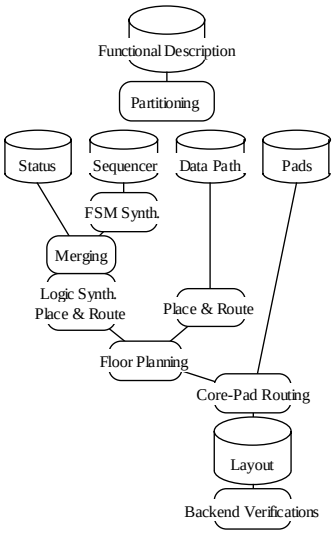

Figure 2: Mips Processor's Design Flow

DESIGNING THE DATA PATH

Once again, because of the shortness of the project time, the RTL description of the data path is provided. This RTL description is used as the specifications of the data path. But unlike the control part, the implementation of the data-path is manually designed by the students. Therefore, the first step consists in writing a structural description (a netlist) using the generic components available in DP_SXLIB library. Each component is build by calling a given operating unit generator provided by the library. Then, in the next step, the resulted netlist is placed and routed.

CHIP LAYOUT

The last step to obtain the physical design of the chip consists in assembling the different blocks. First, the control part and the data path are placed and routed using a router to obtain the processor's core. Then, the different input-output pads are placed and connected to the core.

LOGIC SIMULATION ENVIRONMENT

At each step of the design, a logic simulator is called to achieve the functional verification of a VLSI. Usually, the input interface of the circuit is excited through specific test patterns and the output interface is observed to check the relevance of the calculation realized by the circuit. In the case of the Mips processor, such functional verification method is unrealistic since a close interaction with the processor is needed to control its interface and to feed it with the correct instruction at the correct cycle.

In our project, the functional verification is achieved by placing the processor on a simple CPU board surrounded by its natural environment: a set of memory components (Figure 3). This logic simulation environment is described in VHDL language. This verification approach has been detailed in [3]. Briefly, a functional test consists in writing an assembly program. This program is assembled and the resulted executable code is loaded into a ROM (Read Only Memory) and put on the CPU board. Then, a simulation session starts simply by resetting the processor which begins to execute the program loaded into the memory. If the description does not contain any functional error, the program is executed till the end.

BACKEND VERIFICATIONS

The backend verifications represents an important step before sending the layout to the foundry. It consists in checking that the resulted layout does not contain any error and is conform to the initial specifications. Here we focus on the timing analysis. The Alliance CAD System includes a static timing analysis tool. The input of the tool is a netlist of transistors and capacitors extracted from the layout. This netlist is analyzed to identify the critical paths inside the chip. If the propagation through the longest path of the chip cannot be obtained within the clock period, then the design must be revisited, some part of the circuit must be modified to reduce the propagation delay.

RESULTS

The project of implementing a simplified microprogrammed version of the Mips R3000 processor has been proposed to postgraduate students since three years. Each year five teams have designed their own implementation of the processor starting from the same specifications. Thanks to a coherent set of CAD tools provided by the Alliance System, most of the time, the projects have been achieved successfully within three weeks. The resulted circuit comprises approximately 50 000 transistors and uses a 0.35 micron CMOS technology.

In addition, this project represents a unique opportunity to improve the reliability of the Alliance tools and to check, in a life size experience, the availability of new features and tools. To increase the interaction between CAD tool developers and the students, a Web based bug report system has been developed.

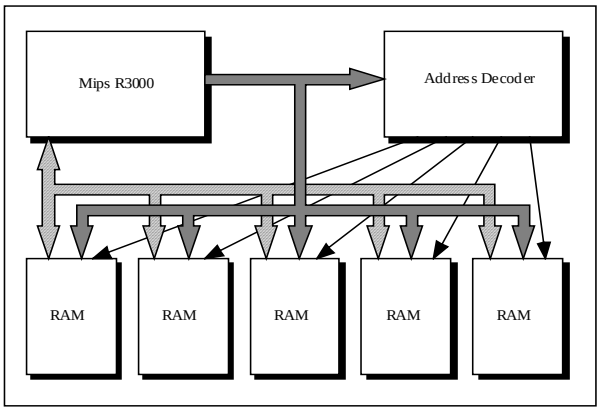

Figure 3: Logic Simulation Environment

CONCLUSION

The project of implementing the 32-bit Mips R3000 processor covers nearly all the aspects of a real VLSI design. Even if a real chip cannot be fully implemented within three weeks, the students' work by a learning-in-doing approach help them in understanding what working in a small design team means. Moreover, it learns them the design methodology of VLSI circuits and the constraints of an industrial project, in other terms, the respect of a given specifications within a limited time.

ACKNOWLEDGMENTS

A great number of the members of the Department of Computer Science of the University of Paris 6 has been involved in this project. This acknowledgments is to thank them all for their participation.

REFERENCES

1. Aas, E. J. On the design of Microelectronic Design Projects to fulfill given Learning Objectives, CAEE'99, 5th International Conference on Computer Aided Engineering Education (Sofia, September 22-24, 1999), pp. 11-12

2. Alliance http://www-asim.lip6.fr/alliance - Email alliance-support@asim.lip6.fr

3. Bazargan Sabet, P., Dunoyer J., Greiner A., Rosset-Louerat M. M. A system level teaching environment for designing the 32 bit DLX Microprocessor, in World Scientific, editor, Proceedings of the 1st European Workshop Microelectronics Education (Grenoble, February 1996), pp. 197-200

4. Greiner, A. et al. Alliance : A complete set of CAD tools for teaching VLSI Design, Proceedings of the Third EuroChip Workshop on VLSI Design Training (Grenoble, September 1992), pp. 230-237

5. Tchoumatchenko, V. and Vassileva, T. Rapid Prototyping Methodology in ASIC Design Education, CAEE'99, 5th International Conference on Computer Aided Engineering Education (Sofia, September 22-24, 1999), pp. 320-325

6. Waldron, J. Introduction to RISC Assembly Language Programming, Addison-Wesley, 1998, ISBN 0-201-39828-1, pp. 1-180