DESIGN EDUCATION IN MICROELECTRONICS AT THE UNIVERSITY OF STUTTGART: VHDL - AND SYNTHESIS BASED DESIGN OF A 32-BIT RISC PROCESSOR IN A FOUR MONTHS COURSE

M. GUMM, M. BÜHLER, U. BAITINGER

ISE - IPVR, University of Stuttgart, Breitwiesenstr. 20-22, D-70565 Stuttgart,
Phone +49 711 7816 216, Email gumm@informatik.uni-stuttgart.de

This paper describes a new course which has been setup recently at the University of Stuttgart in cooperation between the department of computer science and the Institute for Microelectronics Stuttgart. This class, held within the curriculum of computer science, introduces VHDL and synthesis based VLSI design to the students. It is composed of a short lecture part and a major laboratory part where state-of the-art synthesis software is used. A 32-bit RISC processor, defined in the literature [1], has been taken as a design example to be implemented by the students. The scope of this course reaches from behavioural modelling to the simulation of the synthesized gate-level description.

1 Introduction

We describe a new course which has been setup recently at the University of Stuttgart. It is a cooperation between the department of computer science and the Institute for Microelectronics Stuttgart (IMS). As this course is part of the curriculum in computer science, main emphasis is laid on the top-down design methodology using hardware description languages (especially VHDL) and logic synthesis.

2 The DLXS processor

The design task in this course is the implementation of the DLX processor, defined by Hennessy and Patterson in [1] as a hypothetical RISC machine with a simple 32-bit load/store architecture. It is very well suited for teaching purposes because of its easy to understand architecture, which yet demonstrates all the major issues of the RISC principle. The DLX has already been used for tutorial purposes in [2] where the authors focused more on the tool use rather than on modelling, synthesis, and computer architecture aspects. Nevertheless, they developed an interesting validation strategy which we adopted for this course.

The instruction set of the DLX contains, among others, instructions for signed and unsigned integers and for floating point arithmetic. For this design course, only a subset of the original instructions, the signed integer arithmetic, was implemented. On the other hand, we extended our processor model by some features which had been derived (and much simplified) from the SPARC architecture definition [3]:

interrupt- and exception handling, three different operation modes (supervisor, user, error), and one additional addressing mode. This Architecture has been modelled in VHDL on behavioural level and as a synthesizable RTL description which comprises several different design alternatives for some components. We called our architecture *DLXS* (where *S* stands for 'Stuttgart' version).

3 A complete system as test bench

A complete system including the socket for the DLXS-processor, supervisor and user RAMs, supervisor and user ROMs, an address-decoder, a clock-generator, and a timer has been developed and put into a VHDL-library. The structure of this system is depicted in the left part of figure 1.

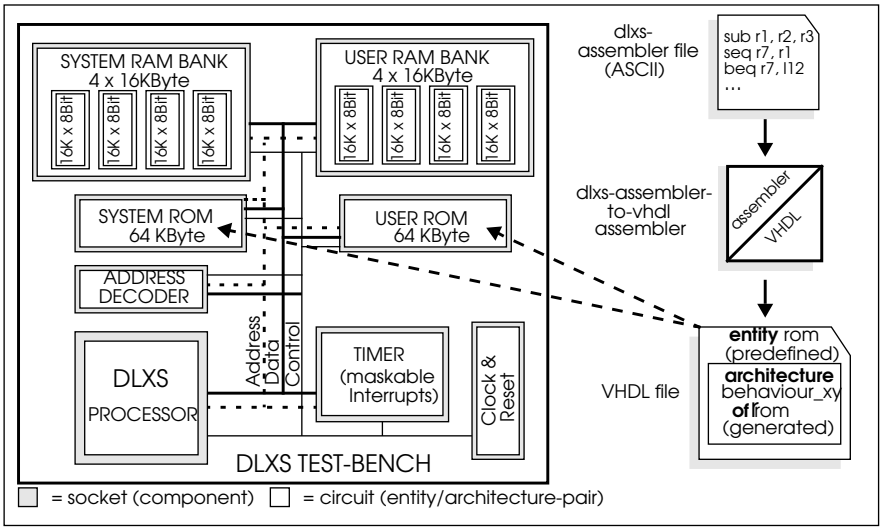

Fig. 1: The DLXS test bench and the generation of user-"PROMS"

Hence, it is possible for the students to "plug" their own model of the DLXS into the test system just by changing the top-level configuration file and to validate it by simulating the entire system.

For the efficient simulation of their processor, the participants can use a specially adapted assembler which allows for the automatic generation of a VHDL architecture-description of the user ROM from the DLXS-assembler source code. Thus, for each feature of their DLXS-model, they can write a small assembler program, e.g. for the validation of the addition instructions of the processor. Afterwards, they just have to generate the corresponding ROM-architecture and simulate the entire test bench

which executes this program. This procedure is shown in the right part of figure 1.

4 Course contents

Coming up from the given specification and some predefined parts, the course participants develop, validate, and finally synthesize their own processor model. The complete design flow as it is applied in this course, is depicted in figure 2.

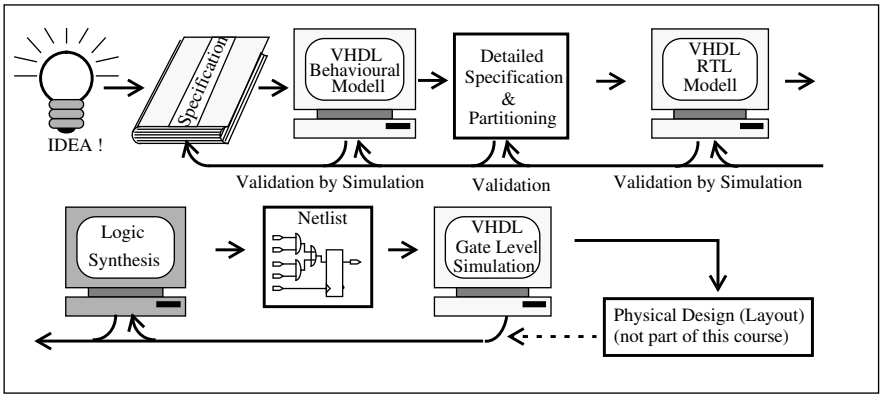

Fig. 2: Applied Top-Down Design Methodology

The software used in our course consists of state-of-the-art simulation- and synthesis tools (SYNOPSYS) which had been purchased via the EUROCHIP initiative. The course starts with a short introduction into VHDL, presented within three one-hour lectures, and in the VHDL-analyzer and simulation software by exercising some smaller design examples, e.g. a RS-flipflop. These examples are chosen so that the three basic VHDL modelling styles - behavioural, dataflow, and structural - are demonstrated.

The next task is the analysis of a given behavioural model of the DLXS within a complete test-bench system (figure 2) in order to explore the different features of the simulation software and to understand the validation concept. This behavioural description exists in several versions which represent different stages of completion. The following exercise consists of the step-by-step development of an RTL model of the processor, especially the datapath, with respect to later synthesis.

After successful simulation and validation of this architectural model, the participants are introduced to synthesis and the corresponding software tools. The following task comprises the synthesis of a gate-level netlist from their descriptions using a proprietary library of the IMS. During this phase, special emphasis is laid on demonstrating the performance but also the limits of synthesis, e.g. the students are held to

model and synthesize several different versions of the 32-bit ALU.

The final task is the gate-level simulation of their DLXS model within the test bench. Although it is not required from the students, the way to the final layout is demonstrated by the instructor in the last lecture. The course finishes with a short visit of the ASIC production line of the IMS.

5 Synthesis results

Using a 1.2 um gate-forest cell library of the IMS, the DLXS can be implemented in a circuit of approximately 60.000 transistors. The maximal clocking frequency for our implemtation lies between 15 and 20 MHz (before layout).

6 Conclusion and future extensions

After the four months, the participants are neither experts in VHDL nor in synthesis based design, but they dispose of a good overview on the mentioned topics and know which problems can arise and how to avoid them. Furthermore, they are familiar with a simulation and synthesis system which can be seen as a quasi standard in industry. Finally, this course leads the students to consolidate their knowledge in the field of computer architecture.

Future modifications of our processor could include the extension by a five stage pipeline for which the processor had been conceived originally. This had not been implemented yet into the DLXS for reasons of better comprehension.

The course material - written in english - including processor specification, description of the predefined test bench, the exercises, and all source codes will be made electronically available.

References:

[1] J. Hennessy, D. Patterson: Computer Architecture A Quantitative Approach, Morgan Kaufman Publ., 1990

[2] P. Bazarghan-Sabet et al. : *A tutorial for advanced VLSI-Course*, 5. Eurochip Workshop on VLSI Design Training, 1994

[3] Sparc International Inc: *The SPARC Architecture Manual (Version 8)*, Prentice Hall, 1992

[4] M. Gumm: VLSI Design Course: *VHDL-Modelling And Synthesis Of The DLXS RISC Processor*, Course Material, University of Stuttgart, IPVR-ISE, 1995 (available via ftp at *ftp.informatik.uni-stuttgart.de*, directory *pub/vhdl/design_course*)

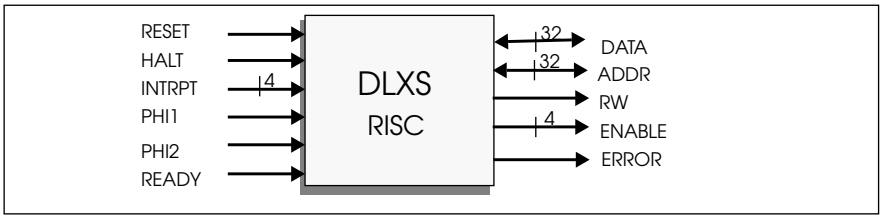

Fig. 3: The DLXS processor