ESE 1000 A hardware toolkit for teaching computer architecture

C Bocage, H Delebecque, O Friedel, C Timsit

ESE_{1000}
A hardware toolkit for teaching computer architecture

C. Bocage, H. Delebecque, O. Friedel, C. Timsit

Abstract
We have developed a modular toolkit for hands-on experimentation which enables students to build and test a wide variety of computer architectures. This toolkit is based on a decomposition into three hierarchical layers: a layer of the CPU's internal modules (e.g. an ALU), a layer of internal processor modules (e.g. a cache module), and one of processor modules (e.g. a 68000 motherboard). This toolkit allows experimentation on a complex multiprocessor array as easily as experimentation on a simple pedagogical computer, with the same user interface and minimal changes in the environment. The toolkit has been designed to be driven either locally, through a pedagogical and powerful front panel, or through a remote host (such as any workstation supporting X Window and Motif, or a Macintosh).

Context
"Introduction to Computer Engineering" is a two-year general course compulsory for all students at the Ecole Supérieure d'Electricté and taught at the Department of Computer Science. The first year course covers the basic concepts, both at the hardware level (Boolean algebra, combinational design, sequential design, arithmetic and logic units), and in software design (high level and assembly programming languages). The second year course is on advanced architectural aspects of computer science, integrating software and hardware design in a unified presentation. A lab session associated with this course allows students to design and build a 16-bit pedagogical computer (called OPIP) using our hardware modular toolkit, hereafter called ESE_{1000}. Supelec students can choose to major in computer science in their third and last year of studies during which they master the larger field of modern computer architecture.

Pedagogical Requirements
The ESE_{1000} has been designed with the pedagogical objective of supplying our students with convenient building blocks and debugging tools for practical experiments. The hardware toolkit has been preferred to its software counterpart for two reasons: firstly, the students work with real hardware, with all its inherent physical constraints (timing, fan-out, characteristics dispersion), and secondly because up to now,

software simulators that take such factors into account have been professional products, which make large demands on computer time and require extensive training for their use. It's feared that the difficulty of running such a product would divert the student's attention from the goal of the experiment by putting too much emphasis on the software aspects.

User Interface

For the OPIP experiment, students use the ESE_{1000} in its least expensive stand-alone configuration**.** It offers a pedagogical front panel, which allows the display of all the necessary modules (two 16-bit registers, an ALU, a 4K word memory module, a multi-phase clock, an instruction register and discrete NAND gates), and their control via user-accessible inputs.

The advanced architectures that students experiment with in their third year, involve complex modules which require remote control and analysis capabilities. This defines the extended configuration of the $ESE_{1000,}$ which includes a workstation or any personal computer able to offer a good human/machine interface. This interface is the main link between the students and the computers they are trying to design, and it must therefore provide the representation of every information exchanged with the ESE_{1000} in the most pedagogical and transparent way. This ability to display the internal state of every module without distortion is a key point of the ESE_{1000} since this allows the student's attention not to be diverted by technological considerations.

The pedagogical aims

The OPIP lab session is organized to emphasize the fundamentals of computer architecture, through the design of a very simple but complete processor, starting from gates and other very simple modules. The student has to design the sequencer, using the timing signals provided by the multi-phase clock, and the information he picks from the data path. The student can monitor the execution of particular instructions either step-by-step, analyzing each information transfer, or instruction-by-instruction when the processor begins to work properly.

The ESE_{1000} can work in two different modes to allow the student to start and/or stop the processor at will. In the first and normal mode, the student's hand-wired logic circuitry controls all the modules, and the multi-phase clock delivers its signals continuously. In the second mode, called the debugging mode, the student can interact with every module via the ESE_{1000} pilot processor, which is able to sample the content of each module and to display it on the front panel. Moreover, in order to allow a workstation to sample the state of the ESE_{1000} , the pilot processor can interpret its commands as well.

During the advanced experiments, students apply the concept of microprogramming, and discover its advantages and drawbacks by building various processors on the same hardware base. Moreover, they can investigate the design of a well-know industrial computer, namely a PDP-11, as well as other, more specialized processors, like a Lisp processor. The extended configuration also includes addressable registers and a microprogramming module, composed of an AMD 2910 and 4K of 112-bit word memory.

A three-layer hierarchical design

The ESE_{1000} has been designed around three hierarchical levels, each of which corresponds to a bus. The first and most internal bus is divided into five 16-bit wide buses, and a 192-bit wide state/control bus. Three of the 16-bit wide buses are called argument buses and the two others result buses, based on their use by the ALU module. The state/control bus conveys every signal coming from, or going to, a module plugged into the internal bus. The meaning of each state/control signal is not predefined and is determined by the module.that uses it; therefore two modules requesting the same signal for different uses can't be plugged in at the same time. The amorphous nature of this bus is the key to the ESE_{1000}'s versatility. The internal bus requires a double-europe sized backplane, and plays the role of the on-chip bus of commercial microprocessors.

The second bus, called the external bus, is the main link between the internal bus and the outside world. In some advanced experiments, it can be used as a processor bus, connecting the processor to its central memory module and peripheral devices. The internal and external buses exchange information through a gateway. By default, the processor built around the internal bus is the master of the external bus, but it may, in some cases, have to support DMA and allow the pilot processor to access the memory modules..

The third and more widely accessible bus is a well-known industrial standard, the NuBus, designed by Texas Instruments. Its main purpose is to allow the design of multiprocessor arrays, by connecting several external buses together through special interface cards.

Related work

A previous hardware toolkit, called the ESE_{10}, developed at the Computer Science Department in 1968, was in use until last June. The OPIP lab session described above was originally designed for this machine. The microprogramming module was added in 1973, allowing the development of the advanced projects, but also revealing the limitations of the ESE_{10}. The $ESE_{10}$was based on 16-bit wide modules connected together by students using 16 different wires, each one

conveying a bit, whereas the ESE_{1000} uses three-state 16-bit wide outputs, connected to general purpose buses. With the ESE_{10}, the time spent by students in wiring the data path of their processors increased dramatically as experiments became more complex leading the students to focus their attention on multiple point-to-point connections, instead of considering whole computer design. Moreover, the DTL technology used in the ESE_{10}, chosen for its good noise tolerance and its robustness, is no longer supported by integrated circuit manufacturers.
Its excellent modularity and the versatility of its internal bus make ESE_{1000} an open and expandable toolkit for experimenting with computer architectures. The differences between our approach, and those described elsewhere [Morgan.88], is the ability of the ESE_{1000} to support a broad variety of architectures, ranging from a simple 16-bit processor to complex multiprocessor configurations. There are other versatile toolkits which allow an equally wide range of experiments [Cutler & Ecker.87]. However these are strictly software simulators with their own specific properties and they serve a purpose which is different from our ESE_{1000} as has already been pointed out above.

Conclusion

It is still too early to draw general conclusions on its effectiveness. However, the students who have used it have greatly appreciated the clearness and precision of the tools developed for on-line analysis and debugging.

References

[Cutler & Ecker.87] Michael Cutler and Richerd R. Eckert
A Microprogrammed Computer Simulator
IEEE Transactions on Education Vol 30 n° 3 August 1987

[Morgan .88] Stephen M. Morgan
A simple Instructional Computer
IEEE Transactions on Education Vol 31 N° 2 May 1988