Hands-on Experience for Teaching Computer Architecture

Henri Delebecque

Hands-on Experience for Teaching Computer Architecture

H. Delebecque*

Abstract

Teaching computer architecture in our School of Electrical Engineering (Supélec) includes both lectures and practical projects. Regarding the practical projects (each about 50 hours of work) three different pedagogical computers have been defined. The first one is very simple: it has only eight instructions and two addressing modes. It emphasizes the very basic concepts of computer architecture, and all our first-year students must design and build it after completing a course on the fundamentals of combinational and sequential logic. The second pedagogical computer has a more complex architecture and all the senior computer science majors have to design and build either a PDP/11-like or a stack-based microprogrammed processor. The third processor our majors have to design and build is a pipelined RISC processor, including their choice of trade-offs between hardware and software as part of their task. The building part of the projects, which have all to end up with a fully working processor, is possible in the rather short time available by using a pedagogical hardware toolkit which has been developed in our lab for that purpose. This toolkit has been designed to allow students to concentrate on the functional characteristics of digital devices by reducing the practical constraints induced by real circuits to a minimum (fan-in and fan-out, short-circuits, wiring, lay-out, etc.). We are currently developing more advanced add-on devices for the toolkit to allow students to explore further concepts of modern architectures, such as the cooperation of processors in multi-processor arrays. More information on the contents of the projects, the reactions of the students and the design of the toolkit are given below.

1. Pedagogical Context

"Computer Architecture" is a two-year general course which is compulsory for all students at Supélec, and includes both lectures and practical sessions. The first year course helps students to assimilate the basic concepts of computer

architecture from the hardware and software points of view. Students become familiar with the usual topics of combinational and sequential logic, and acquire knowledge in high level and assembly programming languages.

The second year course lets students master the fundamental physical entities of computer architecture, namely busses, memory chips and modules, interrupts, and their software counterparts. All students ending their second year at Supélec design and build a pedagogical computer, named OPIP, during a lab session which will be described later.

During their third and last year, all students have to specialize in one of fourteen majors, each in a particular field (e.g. communications systems, power electronics, computer science…). Students who choose to major in computer science will have to master the main fields of modern computer systems, through theoretical lectures and lab work. For that purpose, they have to complete two hardware oriented projects. The first lab-project is devoted to the design and building of a PDP-11-like microprogrammed processor, and the second one is dedicated to a pipelined RISC processor.

2. Pedagogical aims

We have designed the three lab sessions in Computer Science with the four following pedagogical goals in mind. The first two of them are specific to the Computer Architecture domain, and the two others represent general principles of education in our school of engineering

2.1 Students should master the computer's complexity

One of the main purposes of the first lab session is to make students able to design and build a fully working computer within a one-day session. During this session they learn that a computer is neither a "completely closed black box", nor a "magic device", but rather only a complex and comprehensible automaton.

To achieve this goal, we prpose the students to build a very small but realistic computer : the OPIP. It is small since it has only eight instructions, and a 4K-word memory module. However, it remains realistic, because it is based on the same principles as most modern computers.

Students have to build their OPIP using a small set of hardware modules, each devoted to a simple functionality: Arithmetic and Logical Unit, data storage for the memory and register modules, etc. Each module has been designed to remain easily understandable with the theoretical knowledge gained during the lectures in Computer Science. This allows students to keep a clear overall understanding of their computer during the whole building phase, and thus to master easily its behaviour, even in its finest details.

2.2 Lab sessions present architectures of increasing complexity

Computer architecture lectures present computers in increasing order of complexity and the lab sessions follow this progression. Nevertheless, the basic building blocks remain the same, for economical and pedagogical reasons. For this reason, we limit the diversity of the hardware modules, which reduces design and servicing costs and we present our students an homogeneous set of tools, able to synthesize and debug the three kinds of processors with the same paradigm. To achieve these goals, we have focused on modularity and clearness in the design of the hardware toolkit involved in these projects.

2.3 Students should use their theoretical knowledge

During lab sessions, we ask students to apply the notions presented in lectures, such as the interpretation of an instruction, the effect of an addressing mode, the purpose and behaviour of the various registers. These lab sessions are complementary to lectures since students learn to solve particular but real problems with the methods taught in theoretical lectures, which in turn helps them to better understand the concepts.

These lab sessions are a good example of "horizontal" projects, where students have to use concepts acquired in "vertical" lectures (e.g in digital electronics, computer architecture, programming languages).

2.4 A minimal training time

The time devoted to lab sessions in Computer Science during the two first years is limited to about 50 hours, because the global curriculum has to cover the whole scope of electrical

engineering. This time is divided into two modules, one per year. The first one allows students to become familiar with assembly programming languages, whereas the second is the OPIP lab session. To fulfil this requirement, we had to design a toolkit and projects so as to minimize both the training phase, and the time spent by students in auxiliary operations (wiring of modules, loading and downloading the memory, etc).

Lab Material

All these practical sessions use a pedagogical hardware toolkit, called the ESE1000, which has been designed and built at The Computer Science Department. ESE1000 presents modules such as registers, arithmetic and logical unit, or memory in a pedagogical way: inputs and outputs are continuously displayed in hexadecimal, power supply connections are hidden, outputs are protected against short-circuits. Moreover, ESE1000 features modules which are as universal as possible, allowing it to support the three lab sessions mentioned above with only eight kinds of modules.

The Arithmetic and Logic Unit, register and memory modules are common to the three architectures. The ALU module allows sixteen different operations (addition, subtraction, one's or two's complement…). The register module includes a 16-bit wide register (usually devoted to the role of accumulator), and a 16 bit-wide Program Counter. Finally, a 4K-word memory module allows dual access through two independent ports and it has built-in latches for address and incoming data, allowing students to minimize data transfers between the memory and the other modules. There is also a front panel with sockets for Programmable Logic Devices. This panel is able to control the other modules through the equations loaded into PLDs by the students. The OPIP and RISC projects both use a multiphase clock, but the CISC architecture requires a microprogrammed sequencer.

Pedagogical results

The OPIP lab session is proposed to our 360 students, which first design a basic but rather slow processor, and then optimize it to suppress useless phases of instruction interpretation. The OPIP is also given as a one-day vocational training workshop. In both cases, 90% of the students complete

the minimal requirement, and 70% the optimized one. Moreover, former pupils asked to mention one of their favourite lab session very often select the OPIP, even if they dislike hardware. They stress the advantage of remaining physically in contact with the computer they are building, and thus understanding everything that happens.

The success rate is lower for the CISC project. We have observed that the length of this project should better be restrained to the only four afternoons available in a week (the remaining time is reserved for lectures and sport). Often the work which is left for the following week is unsatisfactory, because students seem to lose track of what they were working on. Moreover, the programming environment supplied until now leads to lengthy loading and downloading operations which slow down the debugging phase. To overcome that difficulty, we are currently developing a powerful front-end, based on the X-Windows interface, and supported by workstations (Macintoshes or DEC VaxStations)