DEFINING A CURRICULUM IN COMPUTER ARCHITECTURE: A VIEW ON THE ACTIVITIES OF THE JEP0449 WORKING PARTY ON ADVANCED COMPUTER ARCHITECTURE

António de Brito Ferrari*

This paper describes the rationale for the Advanced Computer Architecture course proposals of the JEP0449 TEMPUS project. The discussion on the consequences of the recent developments in processor architectures for the curriculum contents provides the background for the presentation of the Working Party proposals. These are described in some detail. Finally the integration of the new courses into the Computer Engineering curricula at Wroclaw and Zielona Gora is examined.

1. The Recent Evolution of Computer Architecture and its Consequences for Teaching

The term *architecture* as applied to computers was precisely defined for the first time by G.Amdahl as "the attributes of a system as seen by a programmer" (Bolton, 1990). By defining *architecture* as the basic functional interface between the user and the hardware, Amdahl introduced a new conceptual level in the analysis of digital computers, distinct from the Boolean Logic and Digital System levels.

It has been the notion of the autonomy of the programming model of a machine, as given by its Instruction Set, from its hardware structure that made possible the existence of "computer families" based on a single architecture whose implementations along the years were able to put into profit the technology advancements to achieve significant and steady improvements in performance. IBM's S/360-370, DEC's PDP-11 and VAX, have been some of the most successful cases of this architectural standardisation.

The evolution of microprocessors from the late 1970s, once the technology was there to allow for the integration of a 16-bit CPU into a single chip, followed a similar pattern, with Motorola, National and even Intel keeping to their proprietary architectures during a time span where the scale of integration increased by more than one order of magnitude.

In such a stable architectural environment, a small number of processor architectures dominated the market, with the S/370 architecture, shared by IBM and the various "plug-compatible" manufacturers, having 80% or 90% of the mainframe market, VAX in a clear leading position in minicomputer systems and Intel and

* Departamento de Electrónica e Telecomunicações, Universidade de Aveiro, 3800 Aveiro, Portugal

Motorola sharing between them the PC and workstation markets. On the other hand, the processors made good use of the increasing scale of integration, not only in microprocessors but also in mainframe-class processors where gate arrays became by far the dominant technology.

The organisation of the curriculum in Computer Architecture for Computer Science and Engineering degrees has traditionally, at least since the ACM curriculum proposals of 1978 (ACM Curriculum Committee, 1979), been organised around two courses: the first oriented towards functional aspects, with a detailed study of an instruction set and assembly programming, and the second dealing with structural aspects. The latest ACM/IEEE joint curriculum proposals (ACM/IEEE-CS Joint Curriculum Task Force, 1991), although made in terms of Knowledge Units, instead of courses, remain compatible with the ACM 1978 model in what concerns the Computer Architecture area.

The implications for teaching of a situation where a small number of architectures completely dominated the market, were an increased emphasis on the functional aspects, with a lot of attention being given to the familiarisation of students with one of the leading architectures. In engineering courses that meant usually either a iX86 or M680X0-based course, oriented towards the detailed study of the instruction set and programming in assembly language. The course on computation structures, and in particular the laboratory linked to it, was increasingly faced with the problem that actual processor implementations by the industry were no longer using standard, off-the-shelf, components. Hence a laboratory that relied on the use of bit-slices and MSI parts to build a microprogrammed processor no longer reflected the technological state-of-the-art. On the other hand, to base a laboratory on computer structures on the use of VLSI technology was fraught with difficulties: not only was it difficult to design the components and assemble a system within the timing constraints; "realistic" processor architectures were also too complex to be implemented in a short time span, especially by students with what was at best a very limited VLSI design experience. These constraints originated the move made by a number of universities to emphasize the functional content of the Computer Architecture curriculum, at the expense of the study of computer structures.

In recent years however the stable world of processor architectures has been shattered by the commercial success enjoyed by RISC. This success, closely related to the movement towards the adoption of Open Systems, challenges the principles on which the developments of the two preceding decades had been based, by:

- putting into question the degree of independence of the instruction set definition from the efficiency of hardware implementations, calling for a close link between architecture and VLSI technology
- assigning a renewed importance to the measurement and analysis of instruction set use by asserting that the market is again open to new processor architectures, significantly different from the ones that dominated the last decades
- devaluing the requirement for human-friendliness in instruction set design, pointing that, as a consequence of the pervasive use of high-level languages,

instruction sets should be good targets for compilers, although not necessarily so for programmers

- calling for a close link between processor architecture and compiler technology

Some of the most successful among the new architectures, namely SPARC and MIPS, have their origins in university environments, the former an evolution of the Berkeley RISC (Patterson and Sequin, 1982), that coined the term and started the revolution, the latter directly from Stanford University (Hennessy et al, 1983; Horowitz et al, 1987). It was probably the first time since the 1950s that university teaching and research in computer architecture had such a direct impact on the computer industry through its mainline products - CPUs for conventional, single-processor, systems.

It is our view that the main consequence of this evolution for the curriculum is the need to put a new emphasis on the principles of processor architecture design, within a framework that allows for the quantitative analysis of different alternatives. This runs contrary to two well established approaches to teaching computer architecture:

1. to base the computer architecture courses on a specific processor family. In electrical engineering degrees, due to the market dominance of the IBM PC and its compatibles, programs based on the Intel processor family have been the most common.
2. to follow a case study approach, where different architectures are described and contrasted within a loose framework where no proper analysis tools are available, and hence no quantitative data is obtainable. This approach has been mainly popular in the 1970s and early 1980s.

A second line of evolution of computer systems is the increasing importance of parallel architectures, that moved from the domain of special-purpose systems for specific application niches to become a mainstream commercial solution for high-performance scientific computation. While symmetric multiprocessing with a small number of processors is now the rule for high-end servers, massively parallel systems (MPPs) appear likely to become the standard solution for the next generation of supercomputers. On the other hand powerful and comparatively cheap multiprocessor systems based on the Transputer increasingly appear as a cost-effective solution for a large number of applications in different areas.

The development of parallel processing technology is likely to continue at a fast pace in the foreseeable future, accompanied by a fast expansion of the market for it. Hence rather than regarding Parallel Architectures as a specialised topic not belonging to the core subjects that must be covered in a university degree program in Computer Science and Engineering, it should be considered part of that core, as a subject where every computer engineer should have a solid background.

2. The Working Party Proposals

2.1. Outline of the Proposals

Although the specific task of the Working Party was to design the advanced-level courses on Computer Architecture, the pre-requisites for such courses made it necessary to issue recommendations on the basic-level courses. The existence of two such courses has been proposed:

* *Computer Organisation and Assembly Programming - 45 lectures + 45-60 h Lab.*

* *Interfacing and Peripherals - 30 lectures + 45-60 h Lab.*

The first course should treat all of the topics recommended for course CS-3 of the ACM Curriculum'78 (ACM Curriculum Committee, 1979). Interfacing and Peripherals should deepen the knowledge on the organisation of the input/output system and make students familiar with programming under interrupts. The laboratory part of such a course is particularly important, with the students being exposed to realistic interfacing experiments.

To cover the more advanced topics two further courses are proposed:

* *Computer Architecture - 30 h Lectures + 30 h Lab.*

* *Parallel Architectures - 30 h Lectures + 30 h Lab.*

For these courses a first course on Digital Systems and *Computer Organisation and Assembly Programming* are pre-requisites . *Compilers* and *Operating Systems* should also be given adequate coverage, through a first-level course on Operating Systems, and either a course on compilers or an intermediate-level course on High-Level Language Programming that gives attention to the compilation process, including code generation.

The Digital Systems programs are under the competence of the Working Party on "Logic Design, ASICs and CAD", and the recommendations being issued by it match well with the Computer Architecture requirements.

2.2. Course Programs

The two courses proposed address the areas of single processor systems design and parallel architectures. Although there are few explicit links between the two courses, it is advisable that the students have already done the course on *Computer Architecture* when attending the *Parallel Architectures* course.

Computer Architecture - 30 h Lectures + 30 h Lab.

Objectives:

To give state-of-the-art knowledge on processor design. Two main aspects are covered in order to achieve that objective:

1. the analysis of instruction sets and their use by high-level language compilers
2. the principal techniques used in processor implementation

Strategy:

It is proposed that the Hennessy and Patterson textbook (1990) is used as the main reference for the course. The book assumes only a quite basic background, progressing fast to cover fairly advanced topics. Although the authors have made some of the more important contributions to RISC architectures, the approach is well balanced, using extensive measurements of instruction use to evaluate instruction sets.

The software that is provided to support laboratories, including source code for various benchmarks, a simulator and a C compiler for the DLX RISC architecture defined in the book, will be the main basis for the 30 hours of laboratories that are part of the proposal. The VLSI design of a processor, or some of its main building blocks, could be envisaged once EUROCHIP membership is available to central and East European countries.

List of Topics:

1. Performance Measurement
2. Instruction Set Analysis and Design:
3. The requirements of high-level language compilers. RISC.
4. Processor implementation techniques. Microprogramming.
5. Pipelining.
6. Memory hierarchy. Cache. Architectural support for Virtual Memory.
7. Superscalar and Very Long Instruction Word (VLIW) architectures.

Parallel Architectures - 30 h Lectures + 30 h Lab.

Objectives:

To describe the various types and topologies of parallel computer systems, with particular emphasis on distributed memory MIMD architectures.

Strategy:

Distributed Memory multiprocessor systems appear early in the program so that the laboratory sessions, which are to be based on the use of transputers, can start early.

List of Topics:

1. Classification of Computer Structures: SISD, SIMD, MIMD.
2. Distributed memory MIMD architectures
2.1. Transputer-based. Occam.
2.2. Hypercubes
3. Shared memory MIMD architectures. Case studies: Sequent, Alliant
4. Multiprocessor Algorithms.
5. Vector Processors. Numerical Algorithms. Vectorizing compilers.
6. Array Processors. DAP. Connection Machine.
7. Data Flow architectures.

2.3. Integration into the Curricula at Wroclaw and Zielona Gora

Wroclaw Technical University and Zielona Gora Higher College of Engineering are running 5 year degree courses in Computer Engineering. The analysis of the existing curricula revealed that the amount of time available for courses on Computer Architecture, summarised in Table 1, was:

Wroclaw: 165 hours Lectures 150 hours Laboratories 45 hours Exercises

Zielona Gora: 120 hours Lectures 180 hours Laboratories

The number of hours available was considered by the working party as sufficient to build a sound curriculum that includes advanced topics. This made the implementation of the recommendations to be issued quite a lot easier by making it unnecessary to interfere with courses in other areas.

3. Conclusions

The *Computer Architecture* and the *Parallel Architectures* courses proposed are part of the restructured curricula, currently under implementation. As a result of the upgrading of teaching laboratories, another activity of TEMPUS JEP0449, the equipment necessary to support the laboratory component of the advanced courses is now for the most part installed at the Polish institutions.

The recent opening of EUROCHIP membership to the TEMPUS countries opens up new possibilities for the Computer Architecture curriculum, through its linkage to VLSI design, that could be explored along the lines described in (Marriott and Ferrari, 1992), putting an increased emphasis on the design of state-of-art processors.

Table 1. Existing courses in the area of Computer Architecture

	Course	Semester	Lectures	Lab.	Exercises
Wroclaw	Comp Arch	4, 5	45	30	15
	Peripherals	6, 7	60	30	-
	Arch. & Prog. of μC	5,6	60	90	30
Z.Gora	μP Progr.	5	15	30	-
	Comp Arch	6, 7, 8, 9	105	120 + 30	-

Acknowledgements

Thanks are due to the other members of the Working Party on Advanced Computer Architecture of TEMPUS JEP0449 for their contributions to the proposals described in this paper, although the views expressed are the sole responsibility of the author.

References

ACM Curriculum Committee on Computer Science (1979): *Curriculum'78: Recommendations for the undergraduate program in computer science.*- Comm. ACM, v.22, No.3, pp.147-166.

ACM/IEEE-CS Joint Curriculum Task Force (1991): *Computing Curricula 91.*- IEEE Press.

Amdahl G. (1967): *Validity of the single processor approach to achieving large scale computing capabilities.*- Proc. AFIPS Spring Joint Computer Conf., pp.483-485.

Hennessy J. et al. (1983): *Design of a High Performance VLSI Processor.-* Proc. Third Caltech Conference on VLSI, March, pp.33-54.

Horowitz M. et al. (1987): *MIPS-X: A 20 MIPS peak, 32-bit microprocessor with on-chip cache.-* IEEE Journal of Solid-State Circuits, v. SC-22, No.5, pp.790-799.

Marriott A.P. and Ferrari A.B. (1992): *ARICH experience in teaching VLSI Design and Computer Architecture.-* Proc. 3rd EUROCHIP Workshop on VLSI Design Training, Grenoble, France, pp.260-267.

Patterson D.A. and Sequin C.H. (1982): *A VLSI RISC.-* IEEE Computer, v.15, no.9 , pp.8-22.

Patterson D.A. and Hennessy J. (1990): *Computer Architecture: a Quantitative Approach,* Morgan Kauffman.