Available online at www.sciencedirect.com
ScienceDirect
Computers & Education 49 (2007) 891–915
COMPUTERS & EDUCATION
www.elsevier.com/locate/compedu

Applying a constructivist and collaborative methodological approach in engineering education

Lorenzo Moreno *, Carina Gonzalez, Ivan Castilla, Evelio Gonzalez, Jose Sigut

Escuela Tecnica Superior de Ingenieria Informatica, Universidad de La Leguna, 38200 La Laguna, Tenerife, Spain

Received 9 September 2005; received in revised form 18 November 2005; accepted 15 December 2005

Abstract

In this paper, a methodological educational proposal based on constructivism and collaborative learning theories is described. The suggested approach has been successfully applied to a subject entitled "Computer Architecture and Engineering" in a Computer Science degree in the University of La Laguna in Spain.

This methodology is supported by two tools: the Moodle platform as a collaboration framework among students and teachers and a free Instruction Level Parallelism (ILP) processor simulator called SIMDE, developed by the authors to promote the experience and help the understanding of superscalar and VLIW processors.

This work is described showing how the constructivist and collaborative approaches have been applied and how the activities have been structured temporarily in phases. This educational proposal has been validated and improved with the feedback of the students during two academic years.

Furthermore, the methodological procedure is also suitable to be used not only in subjects with contents which require the understanding of dynamic situations but also in subjects with other requirements.

Keywords: Educational technology; Collaborative learning; Constructivism; Simulators; E-learning; Computer architecture

* Corresponding author.
E-mail address: lmoreno@ull.es (L. Moreno).

0360-1315/$ - see front matter
doi:10.1016/j.compedu.2005.12.004

1. Introduction

A fundamental topic in teaching Computer Architecture is the ILP (Instruction Level Parallelism) subject. Concepts about ILP need the understanding of dynamic situations and implicit parallelism in the different functional units of processor. Therefore, a difficult challenge to be solved by the teacher is to show how the execution process of instructions is carried out.

Furthermore, another key point to be taken into account is the previous knowledge of the student and his misconceptions. Both things are essential for every constructivist approach. Studying these difficulties can help in the design of an effective learning process. Their background, i.e. the foundations upon which the students will construct their models, will clearly affect the way in which the new knowledge is assimilated. The instructor needs to know what to pull down, if necessary. Thus, it is necessary to carry out an exploratory learning, exploring their own ideas and beliefs about ILP processors. On the other hand, it is essential to identify their learning difficulties as a way to understand their cognitive process.

In previous courses, the students have been in contact with several aspects referred to Computer Architecture. They are familiar with concepts such as cache, float units, registers, etc., but they do not dominate concepts such as out of order execution, ILP, branch prediction, etc. From their programming experience, in both high and low level languages, a number of students seem to have adopted, among others, the following misconceptions:

(a) The processor cannot start to execute an instruction until it has not finished executing the preceding one.
(b) The effects of every processed instruction cannot be reversed, in other words, every instruction that enters into the processor is always executed.
(c) Although several instructions can be executed in a parallel way, there is only one "flow" of execution.
(d) The order of execution policy in every processor is similar to the execution flow tested in a Borland-like debugger.

From their experience for several years in teaching activities, the authors have identified three possible sources of learning difficulties when students face ILP processing:

(a) Parallelism in instruction execution.
(b) Out of order execution.
(c) A good number of parameters to define: cache levels, predictive strategies, Translation Lookaside Buffer (TLB), Branch Target Buffer (BTB), number of positions in the reorder buffer, number of functional units, number of reserve stations, etc.

These issues have been shown to be difficult to understand from the reading of a text book since they are too complex to be represented in students' mind. Even in highly acclaimed reference texts like "*Computer Architecture: A Quantitative Approach*" by Hennesy and Patterson (2003), each example depends on specific simplifications that make more difficult to put all concepts together. It is clear that a more visual way of representation (for example, simulators) is desired.