The Use of Programmable Logic Devices In Computer Architecture Course Laboratories

F. J. Looft, Ph.D. (SM-IEEE)
Department of Electrical and Computer Engineering
Worcester Polytechnic Institute, Worcester, MA 01609

Abstract

This paper addresses the use of programmable logic devices (PLDs) in undergraduate computer architecture course laboratories. It is argued that using these devices provides a mechanism for introducing students to topics that are of critical importance to the competitiveness of industry. Specifically, factors such as the probability of implementing a correct design solution, eliminating the need for a prototype of a circuit through simulation, and the speed with which a solution can be found are factors that by using PLDs one can introduce in a meaningful way to the computer architecture laboratory experience.

1. Introduction

The need to update computer architecture course laboratory facilities regularly provides an opportunity for the reexamination of support and implementation factors relating to general laboratory experiment design. While the choice of hardware and software form the basis for a computer architecture laboratory, the focus of this paper is more pedagogical in nature. Specifically, the issue addressed herein is the inclusion of programmable logic devices (PLDs) as fundamental components of the laboratory experience. Because the use of PLDs transcends a specific HW environment, the following discussion will evolve with minimal reference to a specific microprocessor or PLD architecture and is relatively free from reference to the level of teaching.

2. Programmable Logic Devices

It is convenient to define a PLD as,

"...a digital, user-configurable integrated circuit used to implement custom logic function. PLDs implement any Boolean expression or registered function using the generic logic structures of the devices."[1]

[1] Altera Components Handbook, page-3, 1993.

This definition does not address the underlying complexity of the device, the type of logic functions that can be implemented, the technology of implementation, the speed of operation, or even the reconfigurability of the device. Rather, the focus of this definition is on the *use* of PLDs. That is, the *replacement* of many fixed-function logic devices by one or a relatively few number of custom integrated circuits, and the *method* by which the desired logic operation is described and optimized.

Although it is not the intent of this paper to focus on a particular type of PLD, it is inevitable that when presenting examples one discusses the devices used. While there are several well known manufacturers of programmable logic devices and an equally wide range of device configurations, WPI computer architecture laboratories are based on the use of the Altera family of PLDs.

An example of a *macrocell* characteristic of those developed and employed by Altera, but similar in many ways to those used by other manufacturers, is shown in Figure 1. In such cells, one is usually able to use the AND-OR array with and without the memory element. Further, the output of the cell can be used to drive an output pin and/or used as a feedback signal back to other cells through an internal interconnect matrix.

Interconnects are typically created through the use of a high level *hardware description language* (HDL) that is used to describe the desired logical function. This HDL description is then compiled and translated into a set of interconnections within and between multiple cells.

Why?

The aforementioned logic replacement is an obvious reason for using a PLD, but a few related reasons should be noted. First, there are competitive reasons underlying the logic replacement function. These include a need to,

- reduce hardware and software development time and costs,
- improve design quality and upgradability,
- increase the probability of implementing a correct logical and operational solution,
- reduce or eliminate the need for a prototype that must be debugged for logical correctness.

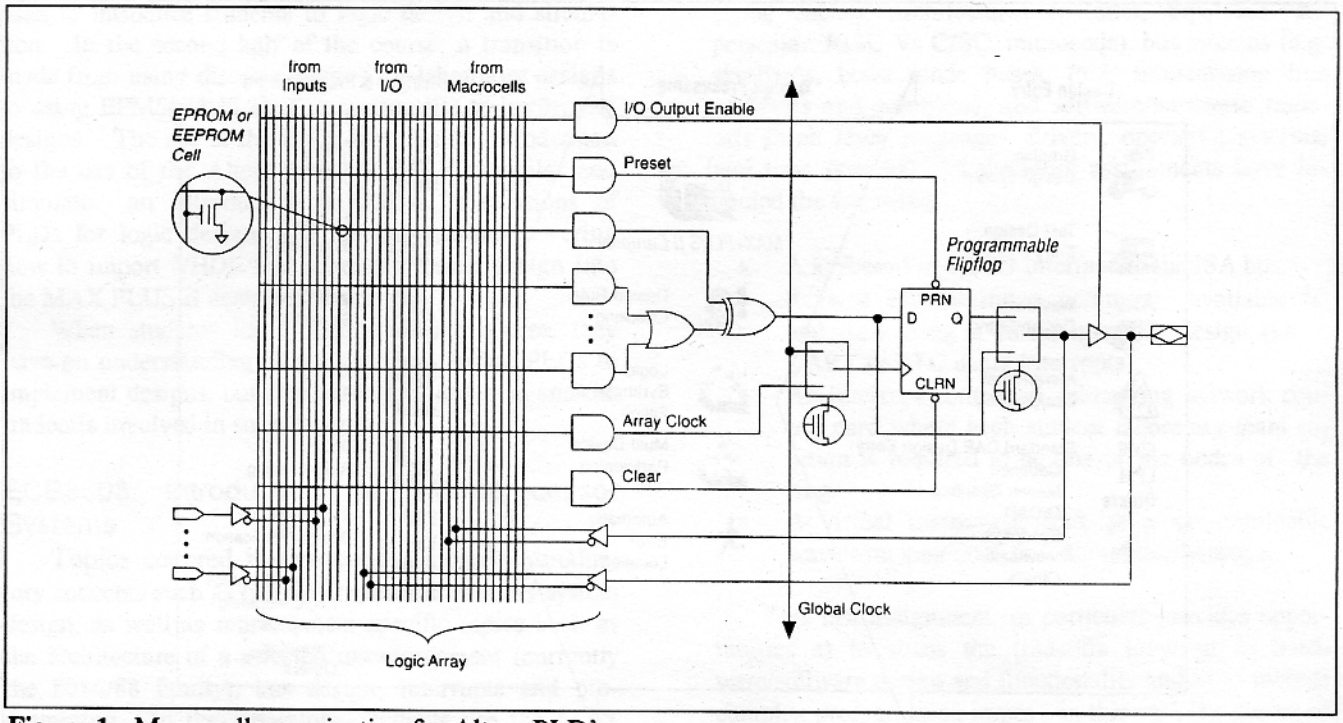

Figure 1. Macrocell organization for Altera PLD's.

These factors provide a basis for both lecture material and for enhancing the laboratory experience, particularly, for example, if students are told they must have a solution they are confident will work as a result of simulation prior to being allowed to actually build the assigned project in the classroom laboratory. Second, by implementing procedures and mechanisms as part of the laboratory assignments that address these goals, there are the added advantages that i) laboratory assignments can be made more realistic since more time can be spent on system design considerations and less on debugging, wiring and other pedantic actions, ii) students will generally enjoy the laboratory exercises more since the assignments will better reflect what they will encounter early in their careers, and iii) more students will successfully complete the assigned laboratory exercises since there will be synthesis and verification tools available that support correct design solutions.

The methodology of design characterization provides similar opportunities to teach, discuss and practice design methodologies that are common in industry. These include,

- a structured design methodology with a standard design format
- a guarantee of proper logical operation as a result of simulation, and
- and tools for testing *operational* limits (max. clock rate, max. propagation delay, etc.).

Of course, virtually any commercial system will satisfy these goals. For example, the Altera MAX+PLUS II design framework depicted in Figure 2 shows that numerous design entry methods (both custom and industry standard) are supported, that the design process can be optimized to meet the needs of the user, and that verification/programming are supported by multiple software modules.

Finally, one should also recognize that there are other factors that are peculiar to the educational environment and should be considered by each college laboratory development committee. These factors might include,

- cost of implementation, upgrades and maintenance,
- ease of teaching and learning,
- level of support from local industry.

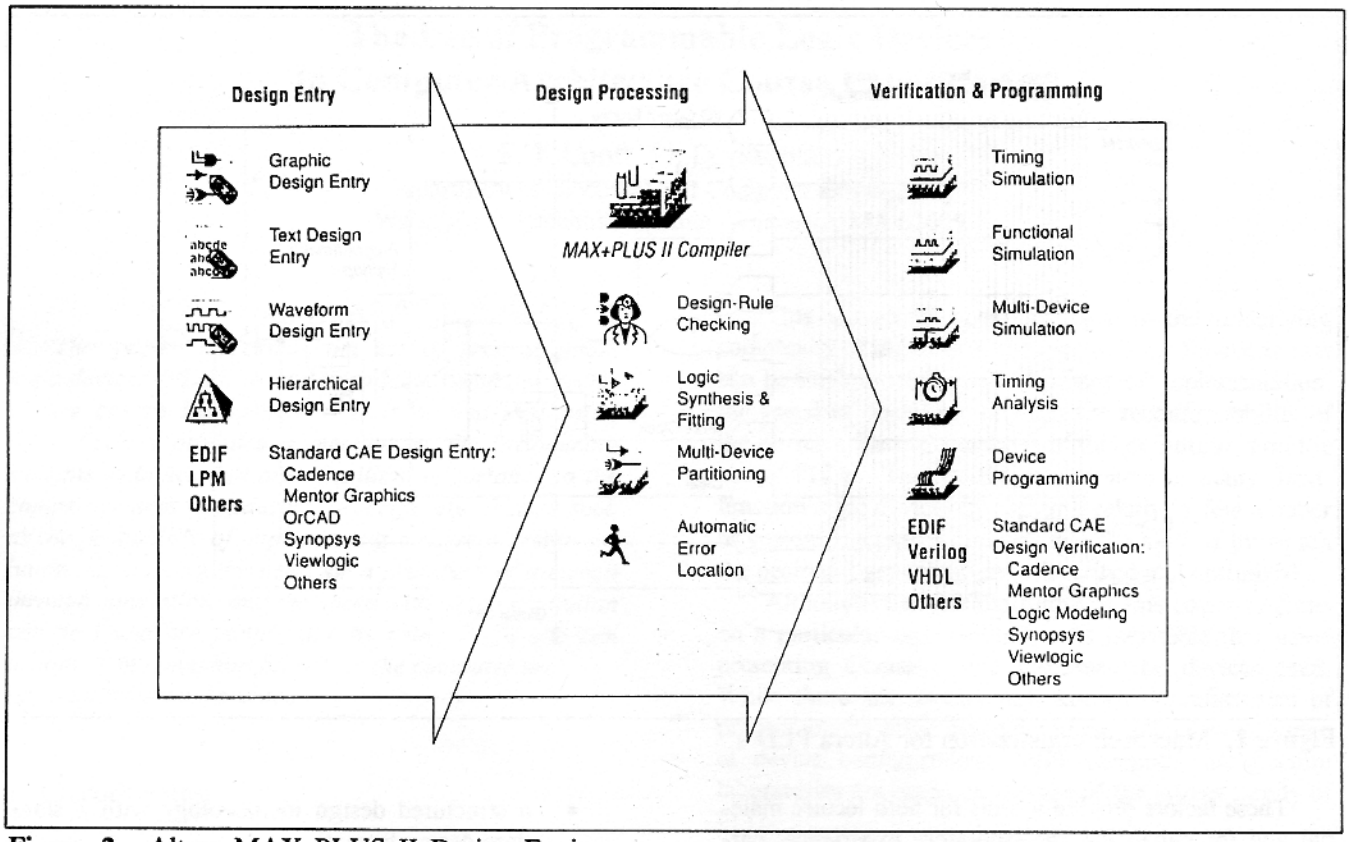

Figure 2. Altera MAX PLUS II Design Environment.

The second factor, that of the time required to teach the PLD design software and the time for students to learn the design framework should not be dismissed lightly. At WPI, this issues is addressed by incorporating lecture time for covering topics related to the design framework (Altera MAX PLUS II), as well as holding weekend, in-lab tutorials where the students have time to try different design entry methods without the pressure of a laboratory assignment deadline and with the support of several teaching assistants as well as the course instructor. While this is a significant time investment, the investment time is capitalized upon by using the same software and enhancing student abilities throughout a three course sequence in the computer architecture area (*below*).

3. Computer Architecture at WPI

PLD Design Environment

The WPI ECE department fully supports the Altera MAX PLUS II design environment. Student's create logical designs using, for example, the Altera Hardware Description Language (AHDL) or ViewLogic VHDL, compile and simulate those designs, and then download the corrected designs to EPLDs[2] of varying complexity[3].

For WPI, the advantage of the MAX PLUS II software is that it runs on an 80486 PC with a SVGA monitor and 16MB of main memory. Our laboratory supports three courses and is equipped with 12 development systems, a laserjet printer, two Altera and one general purpose EPLD programming stations. Devices from 20 to 88 pins can be readily programmed.

Courses

ECE3801 Introduction to Digital Logic Design

Topics covered include basic material such as Boolean algebra and more advanced material such as state machine design. For the first half of the course, the Viewlogic PROSIM design environment is taught and

[2] Erasable, Programmable Logic Device.

[3] Currently we support the EPM5032 and EPM5064 devices.

used to introduce students to logic design and simulation. In the second half of the course, a transition is made from using discrete devices for laboratory designs to using EPM5032 PLDs (32 macrocells) to implement designs. The use of the PLDs includes an introduction to the use of the Altera MAX PLUS II compiler and simulator, an introduction to AHDL, discussions of PLDs for logic design, and tutorial sessions covering how to import VHDL and schematic based design into the MAX PLUS II design environment.

When students have completed this course, they have an understanding of not only how to use PLDs to implement designs, but the reasons for doing so and the tradeoffs involved in such designs.

ECE3803 Introduction to Microprocessor Systems

Topics covered in this course include introductory concepts such as memory types and memory system design, as well as more system specific topics such as the architecture of a selected microprocessor (currently the 8086/88 family), bus design, interrupts and programming. In the laboratory, students use the MAX PLUS II system and EPM5032 EPLDs to implement interface designs of increasing complexity. Sample laboratory problems include the following.

- A parallel output port with address decoding and control logic for a 7-segment display.
- A programmable I/O port with handshaking for data transfer.
- A complete stepper motor controller with control and status registers for mode control.

In the latter part of the course, each student lab team is required to design and build a complete microprocessor system and to design several interfaces to their system. It is at this point that the benefit of using PLDs is clearly apparent. For example, all required "glue logic" such as the address decoders, data bus transceivers, address latches and simple I/O logic can be implemented in one EPLD. Using the EPLD minimizes wiring, wiring errors, logic errors and other associated design problems. In fact, within a week of being assigned the laboratory problem, almost all students have their system working.

ECE4801 Microprocessor System Design

This course follows ECE3803 and covers topics such as advanced memory system design and management (paged mode, synchronous and asynchronous designs, cache), architectures (parallel, pipelined, superscalar, RISC Vs CISC, microcode), bus systems (e.g. standards, burst mode buses, PCI, transmission line problems and solutions), and software/hardware tradeoffs (high level languages, drivers, operating systems, real time systems). Laboratory assignments have included the following.

- A keyboard and LCD interface to the ISA bus.
- Reverse engineering a commonly available IC and then using it in an interface design (i.e. a UART or RTC using interrupts).
- An interrupt controlled, token ring network control card where each student laboratory team solution is required to be one of the nodes of the ring.
- A virtual instrument such as a programmable waveform generator or a digital oscilloscope.

This last assignment, in particular provides opportunities a) to stress the tradeoffs involved in hardware/software design and functionality and b) to include complex programming issues - in this case the design of a virtual instrument screen interface.

Since the assigned laboratories are complex and encompass more material than the traditional hardware design problem assigned in earlier years, the method by which the laboratory assignment is assessed has changed as well. Briefly, it is common to assess when the student laboratory team has,

- completed the system design and assessment of system design tradeoffs,
- completed the design of the EPLD and has successfully simulated and documented the design,
- has demonstrated correct operation of the hardware interface using simple test programs, and
- has completed and demonstrated the laboratory assignment[4].

4. Projects

At WPI every undergraduate is required to complete a project in his/her major area. This project represents

[4] The laboratory programming is based on the use of Borland Turbo C++, Turbo Assembler and Turbo Debugger.

the equivalent effort of three courses, is typically completed in the 4th year, and satisfies ABET's "Capstone Design Experience" requirement. Of course, the Altera MAX PLUS II and Viewlogic PROSIM design environments provide an unparalleled opportunity for project students to implement complex digital systems. Sample projects include the following.

- A performance comparison of direct mapped and a two-way set associative cache architectures with varying line sizes. EPLDs were used in a test board that was designed to provide hit/miss rates. Data was collected from running numerous test programs designed to exercise the boards.
- A complete RISC processor with a six stage pipelined architecture. Each state of the RISC is implemented using one or more EPLDs. Several different design teams, each responsible for one stage of the pipeline, work on this project.
- A programmable controller for a digital security control system.

5. Summary

Although the initial investment costs are potentially high, the costs of not investing in PLD support systems is also high. Foremost among the latter is the fact that without PLDs in the laboratory environment, the computer engineering faculty will be loosing an opportunity to translate into practice key issues that are crucial to not only the competitiveness of industry, but also the value of the engineers that work in those industries. Pedagogically, including PLDs in the computer architecture course laboratory also makes sense . It is our experience that (a) they forestall obsolescence of the laboratory investment, (b) they provide a basis for significantly more meaningful experiments since time can be spent on developing the design, not wiring and debugging, and (c) they provide a basis for capstone design experiences that would, with discrete logic, simply be too complex to implement.