Rapid Prototyping Using Field Programmable Logic Devices

THE AUTHOR DESCRIBES AN UNDERGRADUATE COMPUTER ENGINEERING CURRICULUM USING A RAPID PROTOTYPING APPROACH TO SIMULATE, SYNTHESIZE, AND IMPLEMENT DIGITAL SYSTEM AND COMPUTER ARCHITECTURES.

James O. Hamblen
Georgia Institute of Technology

Traditionally, undergraduates in electrical and computer engineering study the design and implementation of a simple computer and then develop their own designs. In recent years, computer design courses have for the most part taken a simulation-only approach. Rapid prototyping techniques and a new generation of large field-programmable logic devices (FPLDs) enabled an educational approach that combines modeling with hardware description languages (HDLs), extensive simulation, synthesis, and final verification on a hardware prototype.

Rapid prototyping using an FPLD

Ideally, after design and extensive simulation, students would implement and test their new computer architecture on a custom VLSI device. For most classes, cost and fabrication time delays prevent this approach as a viable option.

An attractive alternative for schools is to develop a prototype of the new design using FPLDs. This approach provides fast design cycles and cuts costs because the FPLD hardware is reusable. Industry widely used this option to prototype ASICs and some of the recent microprocessors from companies such as Intel and AMD. The larger designs require a hardware emulator device containing numerous interconnected FPLDs. Recent generations of larger FPLDs contain 10,000 to 1,000,000 gates. For smaller designs, such as those by students, it is now possible to implement the design on one FPLD. Typically, an FPLD-based prototype will have a clock rate 5 to 100 times slower than a fully custom VLSI device. This approach requires a computer-aided design (CAD) tool that supports FPLD development and a small FPLD board.

CAD tools

Commercial digital design CAD tools have made tremendous advances in recent years, and many are used in digital design, computer architecture, and VLSI courses. Many schools now use commercial CAD tool that most CAD tool vendors, such as Cadence, Mentor, and Synopsys, provide at a greatly reduced cost—ranging from a few thousand dollars a year to free. Many of the educational site licenses do not include the bundled third-party tools and preclude the use of this software on student-owned PCs. While commercial Unix workstation-based CAD tools offer many advantages, few schools have the vast quantities of high-end workstations needed to train and support large numbers of undergraduate students and only use these workstations in the more advanced courses.

Student versions of CAD tools for PCs from FPLD vendors, such as Altera and Xilinx, are available free or at a nominal cost. Many digital design and computer architecture texts now

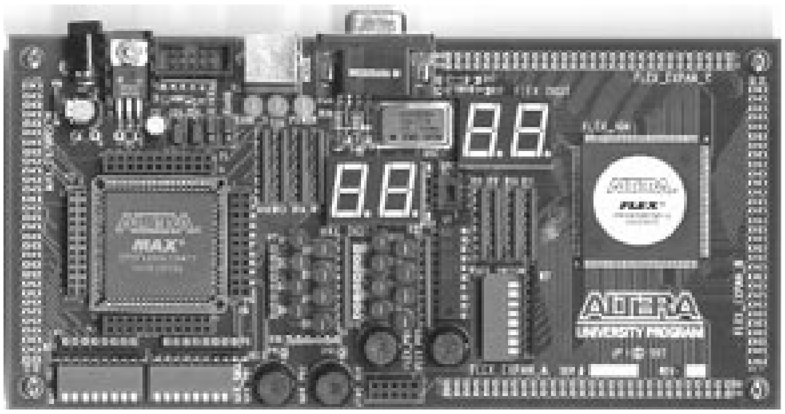


Figure 1. Altera UP1 student CPLD board.

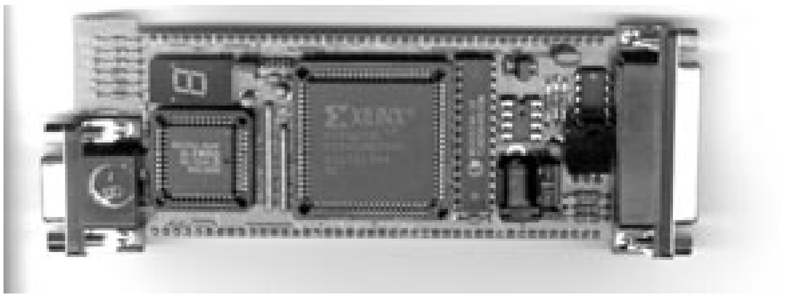


Figure 2. Xilinx XS40 student FPGA board.

include sections on VHDL and provide CD-ROMs with student version CAD tools. Currently, these student versions include schematic capture, VHDL, Verilog, functional and timing simulation, synthesis, place and route, and programming capabilities in the range of 5,000 to 20,000 gates. The majority of electrical and computer engineering students own a PC that can run the student versions as an alternative to university-supplied computers. For larger designs, schools can provide commercial versions of these tools—through donations or at nominal fees—for university-owned PCs.

FPLD development boards

Advanced high-pin-count packaging makes it difficult for schools to directly use large FPLD chips. For ease of use, the FPLDs must be mounted on a small multilayer printed-circuit board with switches, LEDs, and power and I/O connectors. FPLDs include both field-programmable gate arrays (FPGAs) and complex programmable logic devices (CPLDs). Altera and Xilinx each have an active program to support universities that includes special low-cost—about the same as a contemporary textbook—FPLD development boards to support designs developed with student version CAD tools. No expensive programming hardware is required since the logic device is programmed via the PC's parallel port connection. These development boards make it possible to provide students with a low-cost hardware prototyping capability of several thousand gates.

Figure 1 shows the Altera University Program 1 (UP1) CPLD education board.[1] This board includes both a MAX EPM7128S 2,500-gate, P-term EEPROM-based CPLD, located on the left side, and a FLEX EPF10K20 20,000-gate, SRAM-based CPLD with 12 Kbits of internal RAM on the right. In addition to LEDs, seven segment displays, and switches, it has a PS/2 mouse or keyboard connector, and a video graphics adapter (VGA) connector that connects to FPLD pins.

To use VGA or PS/2 I/O, users must design an interface by using logic in the FLEX CPLD. Headers can be mounted around the edge of the board for expansion. For larger designs, a 70,000-gate FLEX 10K70 can be soldered onto the surface in place of the 10K20.

Xilinx offers two low-cost education boards, the XS40 and XS95.[2] These boards have I/O pins that fit into a traditional 0.1-inch protoboard. The XS95 contains an XC95108 CPLD. The XS40, shown in Figure 2, contains a 5,000-gate XC4005XL FPGA. It also has a seven-segment LED, 32 Kbytes of SRAM, and an 8051 microcontroller. An optional motherboard, the XSTEND, for the XS board adds VGA and PS/2 connectors, audio input and output, and a prototyping area. A 10,000-gate XC4010 chip can be used in place of the 4005 for larger designs.

Xilinx's larger education board is based on the new Virtex FPGA. This board contains a 50,000- to 800,000-gate Virtex FPGA, a stereo codec, a PS/2 keyboard and mouse port, a USB port, a VGA with a RAMDAC, two banks of 512k × 16 SRAM, a 16-Mbit flash, a video decoder, and an Ethernet port. The pricing on this larger board depends on the FPGA size and is currently in the same range as a PC. Given the higher cost, a school needs to purchase this board for student use in a traditional laboratory setting.

New paradigm for student laboratory projects

Georgia Tech recently converted from the quarter system to the semester system. This forced the redesign of our courses and presented an ideal opportunity to include computer-based enhancements, such as Web-based testing and multimedia course material, CAD tools, and new device technology.

Traditionally, engineering curricula cover theory first and then applications. Many faculty members feel this approach does not appeal to the current generation of engineering students, may promote a lower retention rate, and does not help motivate students. Easy-to-use CAD tools and widespread computer availability now support an alternative pedagogical approach that first demonstrates interesting applications and then, in subsequent courses, discusses theory in more depth. The digital logic and computer architecture courses provide an opportunity for such a methodology.

We, that is, the faculty, redesigned courses and developed new laboratory projects for the digital logic, computer architecture, and senior computer engineering design courses at Georgia Tech.[3] Early design examples use schematic capture and library components. We use VHDL for more complex designs after a short introduction to VHDL-based synthesis.[3,4] VHDL is currently more widely used in schools than Verilog, in part because Verilog was a proprietary product for several years.

This approach more accurately reflects contemporary practice in industry than the more traditional transistor-transistor logic (TTL) protoboard-based laboratory courses used at schools for the last two decades. With modern logic synthesis tools and large FPLDs, students need more advanced designs to present challenging laboratory projects. Rather than limiting projects to a few TTL chips that will fit on a small protoboard, it is possible to create designs containing tens of thousands of gates using the student version CAD tools and FPLD boards. Student laboratory projects can now implement entire digital systems and small computers.

We developed a number of interesting and challenging laboratory projects for new courses involving state machines with video output, serial communications, video games and graphics, simple computers, keyboard and mouse interfaces, robotics, and pipelined RISC processor cores. For most designs, we used the Altera student version CAD tools and UP1 CPLD board, which were available several months earlier than the Xilinx products. Since the designs are written in VHDL, they can be ported to other programmable logic boards or CAD tools.

Intellectual property cores

One problem area for schools is the limited availability of IP cores. IP cores are widely used in industry to increase productivity, but only a few are available at a reasonable cost for educational use. To use IP cores on a routine basis, universities need special educational licensing or more public domain IP cores. At the 1999 Microelectronic Systems Education Conference (http://microsys7.engr.utk.edu/~mse99/), several speakers suggested that schools should try to develop a Web-based IP core depository to support IP education.

I developed several small IP cores to support use of the Altera UP1 board's I/O features. They include a clock prescalar, switch debouncer, single pulser, seven-segment decoder, VGA video sync generator, PS/2 keyboard input, PS/2 mouse input, and a video character generator ROM. Students use the CAD tools, FPLD boards, and IP cores in our required three-course sequence of undergraduate computer engineering laboratories.

Digital logic laboratory assignments

Our first digital logic laboratory now starts with a single gate and ends with a simple computer implemented on a CPLD that runs assembly language programs and controls a robot. Students still use a standard protoboard and TTL chips for the first two labs to help them gain an understanding of the basic digital logic devices, chips, and pins, and how these can connect to build more complex devices. The subsequent laboratory projects are simulated and then implemented on the FPLD board. We developed two tutorials on the CAD tool environment, an overview of programmable logic, and a design library with several easy-to-use input and output functions to help students get started quickly. Since the CAD tool automatically calculates timing delays, students examine timing issues in early assignments—helping to demystify them.

The third laboratory has a counter that is clocked by push-button input. Students see

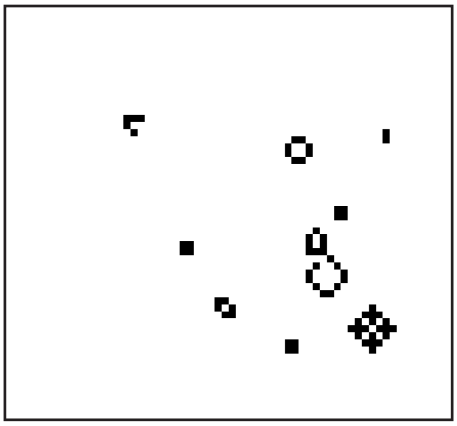


Figure 3. Graphics mode VGA output generated by CPLD in Conway's life game.

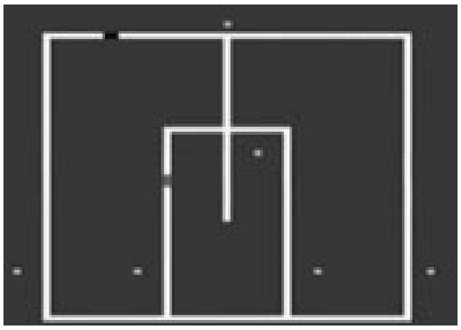


Figure 4. Video output from train simulation in state machine laboratory assignment.

the simulation work correctly, but on the FPLD board-switch contact bounce causes incorrect operation. This provides a real-world example of why hardware verification is needed even after detailed timing simulations.

Figure 3 shows a video output generated directly by the CPLD board. It is running Conway's life game, a form of cellular automata. This is used as a state machine and memory laboratory assignment in the digital logic class. Students are given the video hardware display module, see sidebar, and then they develop the state machine to update generations in pixel memory using the game's rules.

In another digital logic lab, students use the VGA graphics code to develop a train system animation and simulation. Students write a state machine to control the two trains. Outputs control track switches. Inputs from several track sensors detect trains. The state machine must avoid collisions and implement a required travel path. If the trains collide, they stop and the screen flashes. Dipswitches control the speed of the trains. Each semester, instructors change the required path of the trains. Figure 4 shows the video output display of trains, tracks, sensors, and switches.

In the next two laboratory assignments, students receive a simple 16-bit computer model

VGA video generation using an FPLD

The limited I/O features and data displayed on the CPLD and FPGA educational demo boards can pose a problem for complex designs. A video signal can enable display of additional data. Users can generate a VGA video output signal that displays graphics or textual data using hardware inside the CPLD or FPGA.[1] This requires only five signals or pins, two sync signals, and three RGB color signals. A simple resistor and diode circuit converts TTL logic levels to the analog RGB signals. This circuit and a VGA connector come installed on the Altera UP1 board and the Xilinx XSTEND board.

As seen in Figure A, a 25-MHz clock, which is the 640 × 480-VGA pixel data rate, drives counters that generate the horizontal and vertical sync signals. Outputs from these counters also generate the pixel row and column addresses, which connect to a pixel RAM for graphics data or a character generation ROM when displaying text.

The required RAM or ROM is also synthesized inside the FPLD chip. Without additional external memory devices, this ROM or RAM space is limited to only a few thousand bits, so users must take care to conserve the limited video memory available.

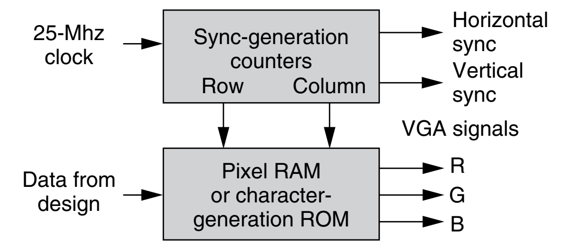


Figure A. FPLD generation of a VGA video signal.

References

1. J. Hamblen, "Using Large CPLDs and FPGAs for Prototyping and VGA Video Display Generation in Computer Architecture Design Laboratories," *Proc. Workshop on Computer Architecture Education (WCAE-4) in the IEEE Computer Society Technical Committee on Computer Architecture Newsletter*, July 1999, pp. 12-14.

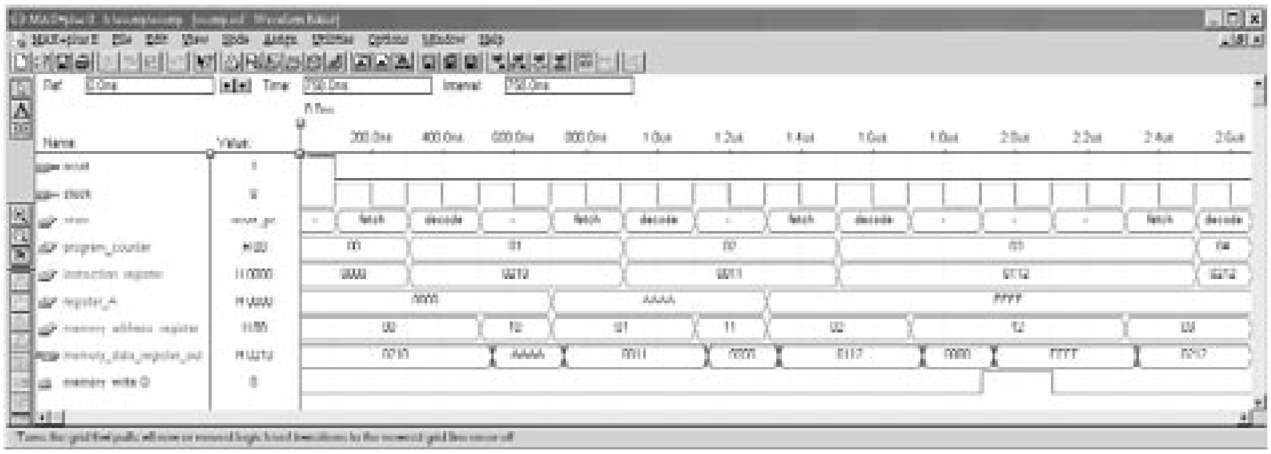


Figure 5. Timing simulation of VHDL-based computer model.

written in VHDL. The computer has a multicycle fetch, decode, and execute sequence and a single accumulator. It provides 256 to 738 words of 16-bit memory using the CPLD's internal 12-Kbit RAM. The entire design for the computer including memory requires only three pages of VHDL code. Students extend the instruction set by adding new instructions and additional registers. They then test the new instructions by running timing simulations and by developing a prototype on the CPLD board. After running test programs that are hand coded in machine language to verify the new instructions function correctly, students package the design as an IP core for reuse in later assignments. A full gate-level timing simulation of the computer design, shown in Figure 5, runs easily on a PC with the student version CAD tools.

For the more complex designs, such as a computer, a few LEDs and switches do not provide enough debug information. In a laboratory environment, students would normally use a logic analyzer to capture additional data. These devices are expensive, take several minutes to attach, and can require a significant number of normally scarce chip pins. Since students would typically work at home on these low-cost boards, we added a VHDL-based state-mode logic analyzer to each design. They can display data from the logic analyzer by generating a VGA video signal directly in the FPLD. Students can attach their PC's VGA monitor to the FPLD board for use as a debugging tool. The class provides students with a video IP core to assist in the hardware verification process. The video IP core displays the value of the computer's major bus signals and registers in hexadecimal.

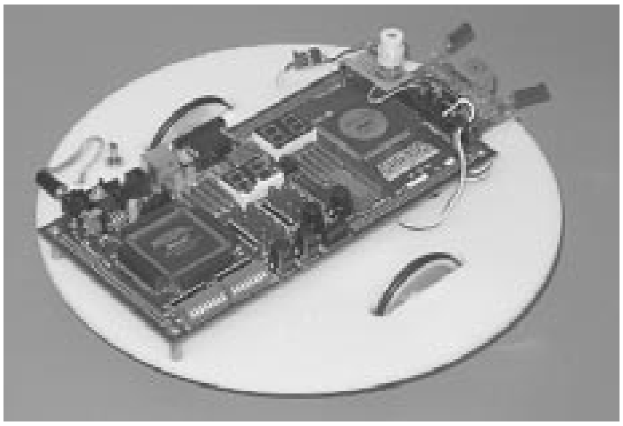


Figure 6. Robot controlled by a CLPD board.

In the last two laboratory assignments, students use their computer IP core to control the small mobile robot shown in Figure 6. The robot uses low-cost, commercially available, radio-controlled modeling parts. The total cost of the robot is about the same as a textbook. Students write assembly language code to monitor sensor inputs and control pulse-width-modulation servo drive motors. Sensors include infrared and sonar distance-measuring devices and a low-cost, Hall effect, magnetic-compass module used in automobiles.

A student-developed meta-assembler, similar to software widely used for bit-slice designs, lets students develop longer programs. The meta-assembler outputs machine code in a format directly supported by the CAD tool. This

Figure 7. Meta-assembler output for a student's instruction set.

enables automatic FPLD memory initialization during device programming and configuration, making it easier to run assembly language programs in both simulations and on the FPLD board. Figure 7 shows a sample output from the meta-assembler.

In the final laboratory assignment, a team design project uses the robots and the computer IP core. Recent design projects included obstacle avoidance, a robot dance contest, infrared communication between robots, and robots following a path programmed using a standard infrared TV remote control.

Computer architecture laboratory assignments

In our introductory computer architecture class, students study a MIPS 32-bit RISC processor design from the widely used Patterson and Hennessy text.[5] Figure 8 shows the

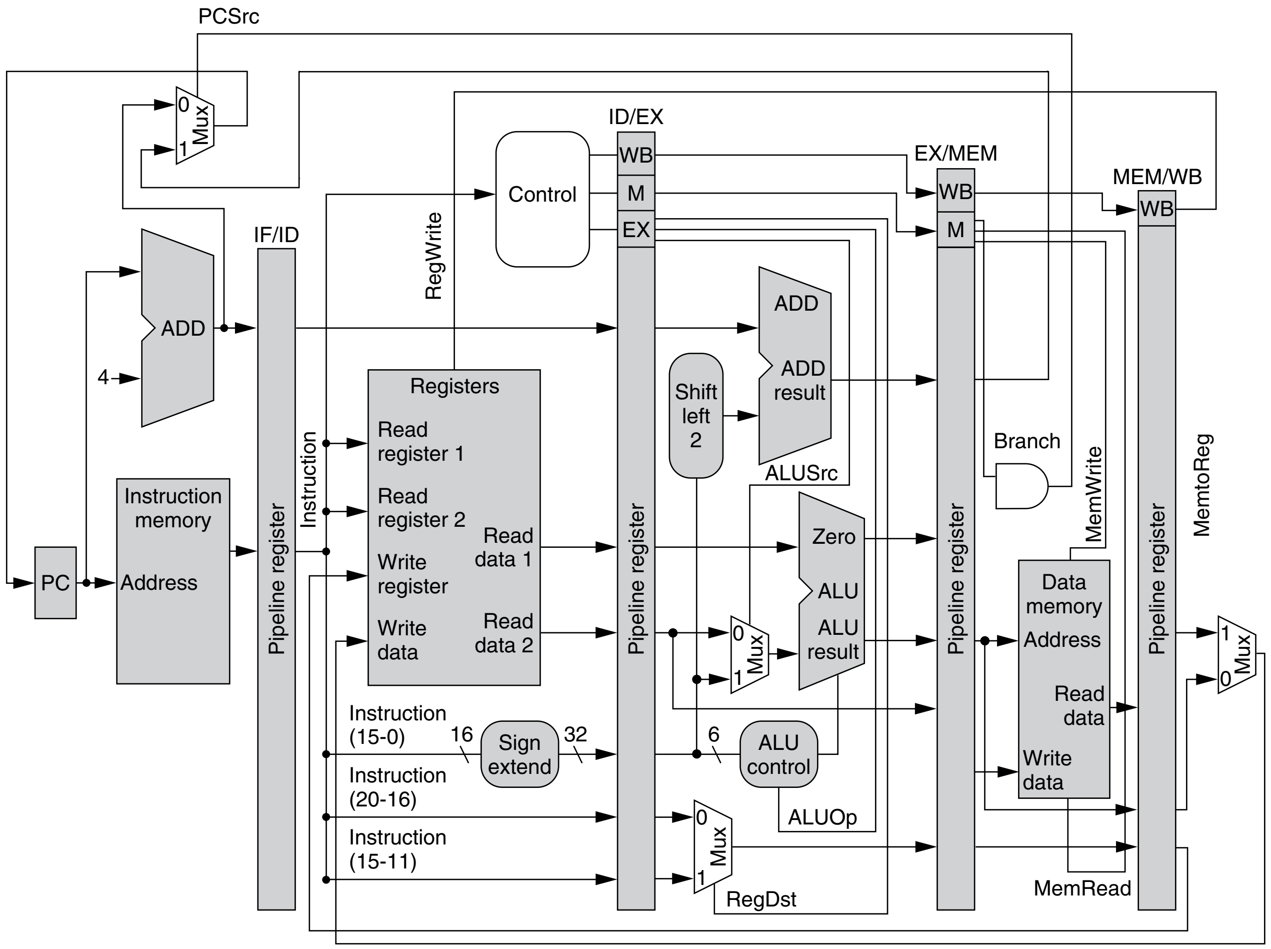


Figure 8. Pipelined MIPS processor core.

MIPS design. Concurrently, students run simulations on a one-clock-cycle MIPS model written in VHDL.

In a series of five laboratory assignments, students expand the MIPS model by adding instructions, pipelining, forwarding, and branch flushing. In an earlier version of this course, we developed the MIPS VHDL synthesis model and then ported it to student version CAD tools.[6] In this laboratory, as in the others, students debug simulations prior to programming the CPLD device, since it provides a faster development cycle. Figure 9 shows a gate-level timing simulation of the MIPS model run on the student version CAD tool.

Figure 10 shows the video output from a MIPS processor core used in the computer architecture class. This VHDL-based design contains a small character generator ROM with an 8 × 8 font for character display. An IP core provides students with the video display hardware module and logic analyzer code.

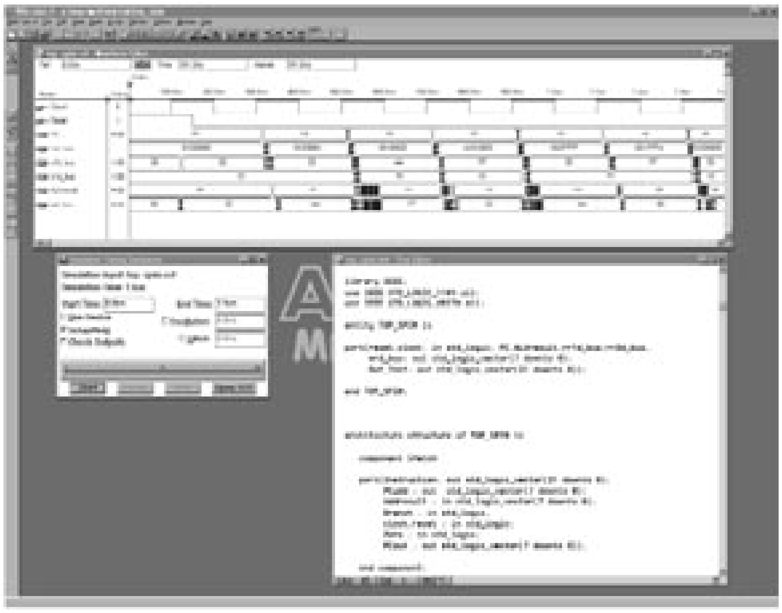
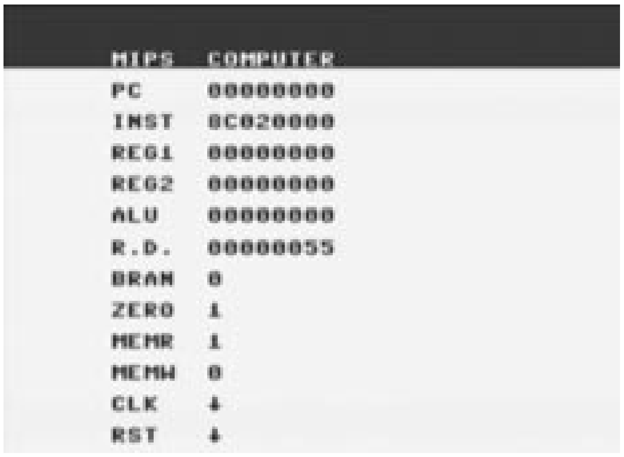


Figure 9. MIPS processor core timing simulation.



```
MIPS  COMPUTER
PC    00000000
INST  0C020000
REG1  00000000
REG2  00000000
ALU   00000000
R.D.  00000055
BRAN  0
ZERO  1
MEMR  1
MEMW  0
CLK   1
RST   1
```

Figure 10. VGA output from internal CPLD logic analyzer in the MIPS processor model.

Computer engineering senior design laboratory

We took the same approach of using the VHDL-based CAD tool and a CPLD board to develop an updated version of our computer engineering senior design laboratory. Students work together in teams of two to four on a four-month computer-design project to develop hardware and software for a pipelined RISC processor of their own design. Students in this course are already familiar with digital design, FPLDs, VHDL modeling, simulation, synthesis, assembly language, C programming, and computer architecture from earlier required coursework, although the first few groups had not previously used the CAD tools or FPLD boards.

Previously, this course used commercial Unix workstation versions of the Synopsys CAD tools and a 30,000-gate Zycad/Inca hardware emulator.[7] The course also required significant CPU time on several Unix workstations for a design cycle, and the limited availability of the workstations had an impact on many of the designs.

The Altera UP1 board contains a 20,000-gate CPLD chip: the FLEX 10K20. We desoldered and replaced the 240-pin, quad-flat-pack surface-mount chip with a pin-compatible FLEX 10K70, which contains 70,000 gates and more internal RAM. This lets students work with larger designs in the senior design class and still use the low-cost student board.

As shown in Figure 11, students use VHDL-based logic synthesis and the 70,000-gate FPLD to develop a working prototype. Students configure a meta-assembler to produce machine language test programs. The meta-assembler lets the user define instruction formats and opcodes. For their processor design, students use lcc, a retargetable ANSI C compiler that automatically generates a code generator, lburg, to develop a cross com-

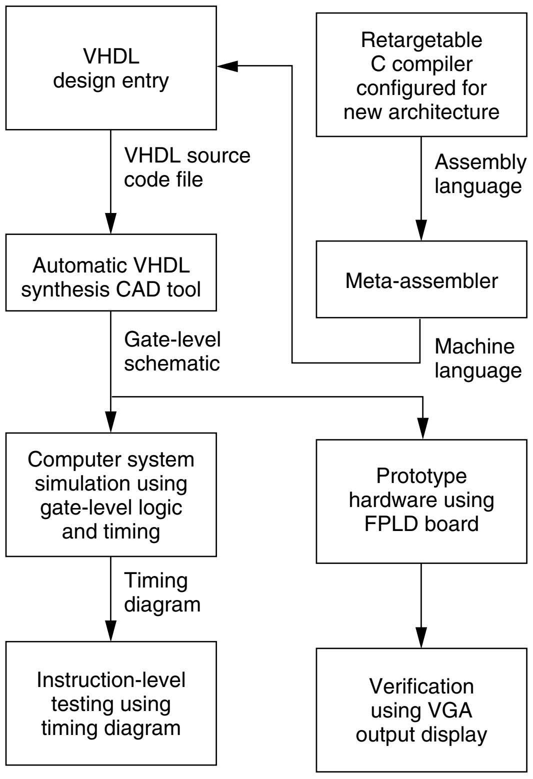

VHDL design entry
VHDL source code file
Automatic VHDL synthesis CAD tool
Gate-level schematic
Computer system simulation using gate-level logic and timing
Timing diagram
Instruction-level testing using timing diagram
Retargetable C compiler configured for new architecture
Assembly language
Meta-assembler
Machine language
Prototype hardware using FPLD board
Verification using VGA output display

Figure 11. Design process and associated CAD tools.

piler.[8] Students load object code into a ROM inside the CPLD.

After VHDL synthesis, students implement the resulting schematic or netlist on the CPLD device using a commercial version of Altera's Max Plus II tools. Student version CAD tools only support the smaller FPLDs originally supplied on the student FPLD board. Automatic generation of the data files to produce the prototype systems requires 5 to 20 minutes of computer time for each large design. Design download time to the FPGA/CPLD board is a few seconds. Prototype digital designs with up to 70,000 gates are possible with typical clock rates in the 8- to 12.5-MHz range. To aid hardware verification, students modify the video display hardware modules and the logic analyzer code—developed for the introductory computer architecture classes—to monitor and debug their new processor designs.

Students have implemented five- to eight-stage pipelined RISC processor cores with MIPS, Alpha, DLX, Sparc, and PARISC-like instruction sets. A few students have implemented superscalar designs, but most are too large to fit onto the 70,000-gate device. To enable faster design cycles and simulation times, students implement a subset of the register file and reduce the data path width to 8 bits for early simulations. They do not implement floating-point and memory management instructions. The designs required 42,000 to 61,000 gates and students successfully executed a bubble-sort program on the CPLD board at an 8- to 12.5-MHz clock rate.

We evaluate designs using a number of factors including clock rate, gate count, and benchmark program execution time. To prevent students from starting too late, we divided the design task into a series of project milestones. At each milestone, a presentation is required from each design team. The presentation also serves as a design review, and the instructor provides feedback to correct any major problems.

In the following academic quarter, students add a PS/2 keyboard input interface, a VGA-based terminal for character output, and integer multiply and divide hardware. They also implement the full register file, widen the data path, and retarget a C compiler for their processor design.

For the final exam, students receive two C programs that they have to compile and execute correctly on their processor without manual intervention. If an error is found, the groups can explain and fix the error to reduce the points deducted from their test score. The vast majority of groups successfully run the compiled C code on their processor. About half of the groups find a minor hardware or software bug that they diagnose and repair during the final exam period. Common problems include a previously undetected forwarding bug in the pipeline, an incorrect instruction from the assembler, errors in compiler retargeting and code generation, and minor hardware errors in a single instruction's execute phase.

Using the new PC-based student versions of CAD tools on CD-ROMs and low-cost CPLD and FPGA boards, undergraduate students can design, simulate, and develop working prototypes of complex digital and computer systems as a routine part of their laboratory coursework. Students can use their own PCs to run relatively large problems. Students detect most design problems in the functional and timing simulations; but it is still worthwhile to develop a hardware prototype especially with the support provided by the newer CAD tools. After design entry and simulation, these tools require little additional time or effort for the final hardware prototyping and verification step. Feedback from students is very positive, and they are strongly motivated by seeing their designs run on actual hardware rather than only a simulation.

We are currently working to add Web-based streaming-video lectures to support each of the laboratory projects. In the senior design class, the size of the FLPD limits the more complex student designs and will benefit by upgrading to larger FPLDs when the next generation of development boards become available. MICRO

References

1. *Altera University Program Design Laboratory Package User Guide*, Altera Corporation, San Jose, Calif., 1997; http://www.altera.com/.
2. D. Van den Bout, *The Practical Xilinx Designer Lab Book*, Prentice-Hall, Upper Saddle River, N.J., 1999; http://vig.prenhall.com/acadbook/0,2581,0130205869,00.html.
3. J. Hamblen and M. Furman, *Rapid Prototyping of Digital Systems*, Kluwer Academic Publishers, Norwell, Mass, 2000; http://www.ece.gatech.edu/~hamblen/book/book.htm.
4. D.J. Smith, *HDL Chip Design*, Doone Publications, Madison, Ala., 1996; http://www.doone.com/hdl_chip_des.html.
5. D. Patterson, and J. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, Morgan Kaufmann, San Mateo, Calif., 1994; http://www.mkp.com.
6. J. Hamblen, "Using VHDL Based Modeling, Synthesis, and Simulation in an Introductory Computer Architecture Laboratory," *Int'l J. Electrical Eng. Education*, Vol. 33, No. 3, July 1996, pp. 251-260.
7. J. Hamblen et al., "An Undergraduate Computer Engineering Rapid Systems Prototyping Design Laboratory," *IEEE Trans. on Education*, Vol. 42, No. 1, Feb. 1999, pp. 8-14.
8. C. Fraser and D. Hanson, *A Retargetable C Compiler: Design and Implementation*, Benjamin/Cummings Publishing, Redwood City, Calif., 1995; http://www.cs.princeton.edu/software/lcc.

James O. Hamblen is an associate professor in electrical and computer engineering at the Georgia Institute of Technology. His current research interests include rapid prototyping, high-speed parallel and VLSI computer architectures, computer-aided design, and reconfigurable computing. Hamblen received a PhD in electrical engineering from Georgia Tech, an MSEE from Purdue University, and a BEE from Georgia Tech. Prior to earning his PhD, he worked as a systems analyst for Texas Instruments in Austin, Texas, and as a senior engineer for Martin Marietta in Denver, Colorado.

Direct questions about this article to the author at Georgia Institute of Technology, School of ECE-0250, 777 Atlantic Drive, Atlanta, GA, 30332-0250; hamblen@ece.gatech.edu.