Overview of the URI ICED Protosys Laboratory

Augustus K. Uht
Dept. of Electrical and Computer Engineering
University of Rhode Island

May 23, 2000

1 Introduction

The URI ICED (Integrated Computer Engineering Design curriculum) multi-year project entails the design, construction, test and evaluation of a complete computer system including CPU, memory interface, bus, compiler and network interface. This paper gives an overview of the ICED Protosys Laboratory; the paper is primarily targeted for the ICED student. The key elements of the Protosys hardware and software are identified and described.

Once you grasp the overall picture and components of the Protosys, you will be ready to dive in to the ICED tutorials and labs and do the detailed design work. We recommend that from time to time you refer back to this paper to refresh yourself with ICED's big picture.

2 ICED Laboratory Tools

The overall laboratory goal is to have multiple students' computers communicating with each other on an Ethernet-like network. You will be able to see a vertical slice through the design levels of the machine, including source code, assembly code, machine code, digital signals on the computer bus and analog pictures of particular digital signals in the system.

The vast majority of the ICED laboratory experiences are built around the design or use of a CPU within an FPGA. An FPGA consists of many uniform logic cells wired together via an interconnection network. The *configuration* of the FPGA consists of the specific logic functions realized by the cells and the signal routing and connections. This information is held on the FPGA in static RAM. FPGA's can be reconfigured an unlimited number of times.

2.1 A Sketch of the ICED Prototype System Laboratory

A schematic block diagram of the physical components of the lab is given in Figure 1. Referring to the figure, we now go through the system, describing the major components; bear in mind that the goal is for you to physically realize an entire computer system. Note that the "custom" hardware described below has already been built for you.

A Virtual Computer Corp. (VCC) EVC1 card containing a Xilinx FPGA is housed in the chassis of a Sun Ultra 1 and is connected to the Sun's SBUS I/O bus. An external connection to the EVC1 is made through the back panel of the Sun via a custom connector/daughtercard assembly called the *d-card*. The d-card connects to another custom card, similar to a motherboard, called the *sys-card*. The sys-card contains your prototype's bus and supporting hardware, including the prototype's main memory. In turn, the sys-card is connected through short cables to a generic protoboard, which holds the ICED system interface logic that you must design. The sys-card also provides for a standard BNC (Bayonet Nut Couple connector) tap onto the prototype network, ICEDnet. Lastly, the HP logic analyzer is used to test, debug, measure and otherwise investigate the prototype. The logic analyzer has 64 data input channels, viewable as state or timing information, as well as analog signal and assembly code viewing options.

This paper is excerpted from [1], [2] & [3], with modifications.
For more ICED information , see the ICED home page: http://www.ele.uri.edu/iced

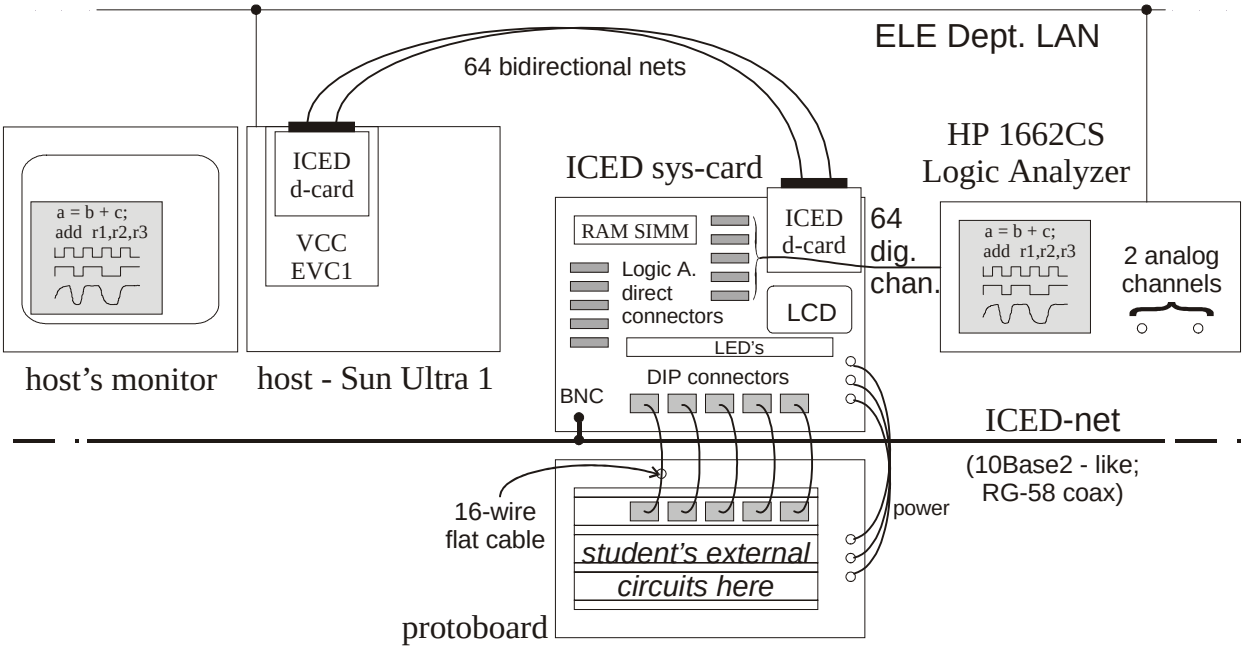

Figure 1. *ICED prototype system and lab station. Your CPU is in the FPGA on the VCC EVC1. The sys-card and d-cards were designed and built at URI for your use in the Protosys. The logic analyzer is completely viewable-on and controllable-from the host.*

2.2 VCC EVC1 FPGA Platform

The major component of the EVC1 (see Figure 2) is the Xilinx XC4020E-2 FPGA, nominally equivalent to 20,000 simple gates. The FPGA is large enough to hold a simple 32-bit processor. 32-bit pipelined CPU's with forwarding can also be accommodated at the cost of extra time to route the FPGA's internal interconnections.

The EVC1 has many useful features. First, there is a general-purpose interface from the host Sun's SBUS to the FPGA. In fact, part of the interface extends to within the FPGA, and is logically connected to your designs via the EDA tools. Further, many VCC-provided C software utilities are available to you for such purposes as downloading design files (configuring the FPGA), changing the frequency of a separate clock oscillator (the POM; see the figure), general communication with the FPGA, and resetting and testing the EVC1.

External connections are made to the EVC1 via its standard daughtercard connectors. These provide access to general I/O pins on the FPGA, configurable as input, output, or bidirectional. An ICED d-card containing SCSI-type transceivers plugs into the EVC1 and sends 64 directionally-configurable signals to a d-card piggybacked on the sys-card. The directions of the signals leaving the EVC1's d-card are set by programmable outputs on the FPGA. On the sys-card, the signals are also directionally-configurable, this time by jumpers or logic on your protoboard.

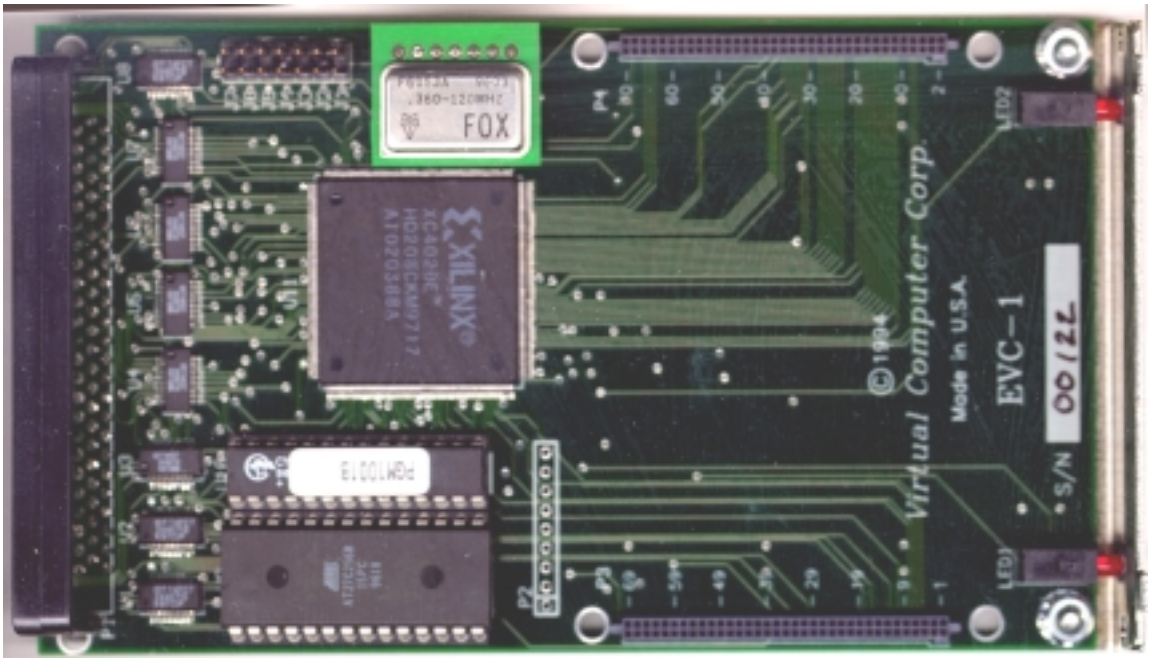

Figure 2. *Virtual Computer Corporation EVC1, shown about full size. From left to right, the main components are: Sun SBUS connector (on obverse, beneath handle), interface and FPGA configuration logic, Xilinx XC4020E-2 20,000 gate-equivalent FPGA, POM (Programmable Oscillator Module - marked "FOX"), and space and connectors (P3 and P4) for a daughtercard (ICED d-card). A retooled back plate (not shown) holds two connectors providing external access to the d-card, EVC1 and hence the FPGA.*

2.3 ICED Protosys Laboratory Station

A picture of the overall ICED hardware laboratory setup as realized is shown in Figure 3. The different pieces of equipment are as follows:

1. Host computer display
2. Host computer - with cover off
3. Logic analyzer - 64 digital inputs, 2 analog inputs
4. Logic analyzer digital probe - 16 inputs
5. EVC1 card with FPGA and ICED daughter-card (*d-card*)
6. ICED system-card and chassis (*sys-card*)
7. ICED computer bus - *IBUS* - connects d-card on EVC1 with d-card on sys-card
8. Protoboard - holds memory, network and LCD to CPU interface logic
9. ICED network - *ICEDnet*

The Protosys equipment is used as follows. The host computer runs the equivalent of a monitor program, allowing it to control the ICED computer's bus (IBUS) and CPU. With these capabilities the host is used to download code and data to the ICED computer's memory and start, stop and single-cycle the ICED computer. The host is also used independently of the Protosys to run the Mentor Graphics and Xilinx EDA tools to enter the CPU and other digital designs and configure them for the FPGA. The configuration data is downloaded to the FPGA to make it become an ICED CPU. Software development is also done on the host, including compiler and monitor development.

The custom hardware consists of the d-cards and the sys-card. The d-cards connect the FPGA (ICED CPU) to the sys-card. The sys-card is essentially a realization of the ICED computer bus and the ICED computer-proper with multiple built-in probing points and great interconnection flexibility. The protoboard connects to the sys-card and is used to hold the student-designed interface logic connecting the ICED computer's main memory and I/O devices to the IBUS. GAL re-programmable logic devices (Generic Array Logic) and standard SSI and MSI IC's are made available to you for this hardware.

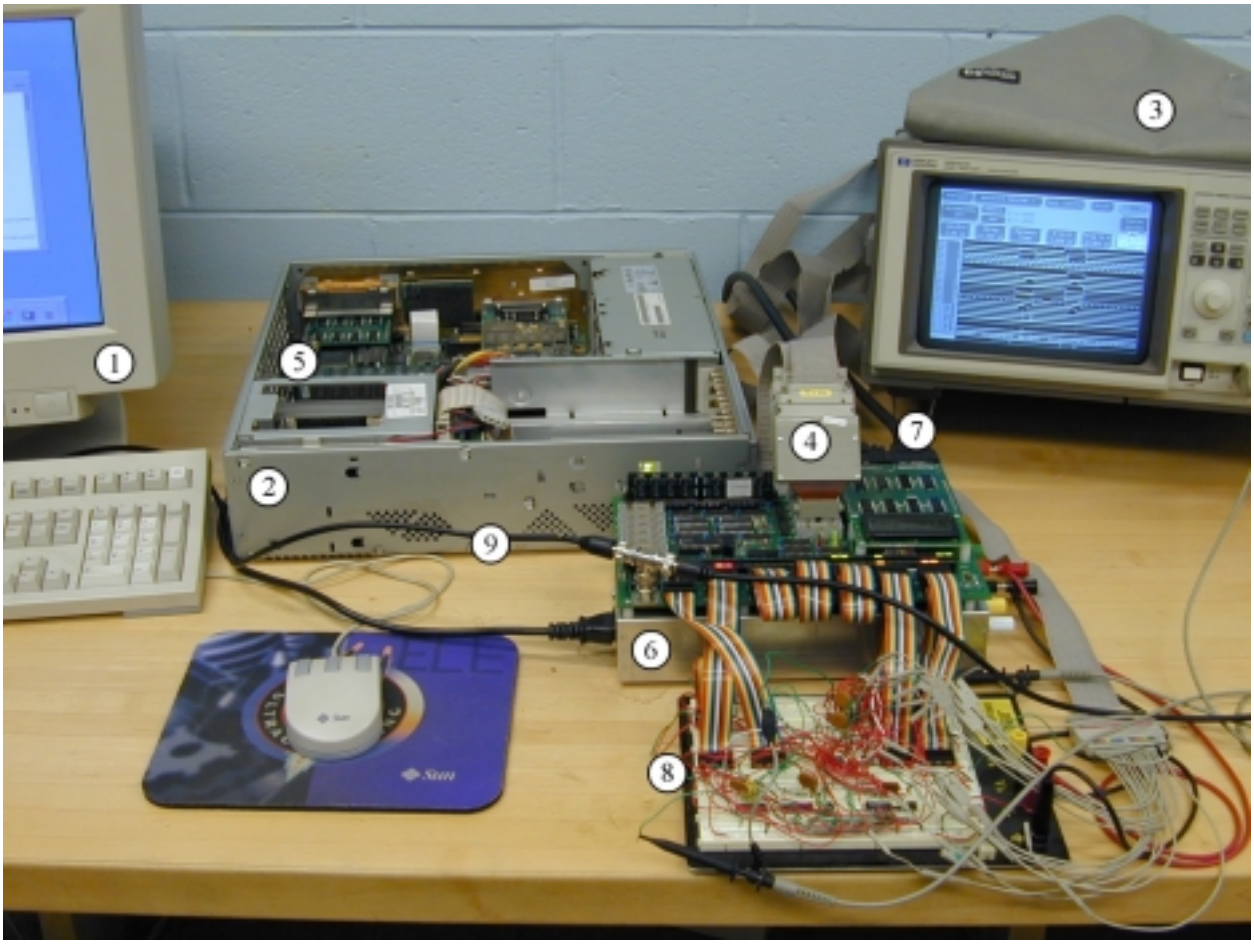

Figure 3. *ICED Lab Station.* Photo by Laurette Bradley.

The logic analyzer connects directly to the sys-card through special connectors, reducing the wiring drudgery for the student. The analyzer is used both to debug the ICED computer and to visualize its operation at multiple levels of abstraction. There is an X-window interface and Internet interface on the logic analyzer allowing the analyzer to be viewed and controlled from the host's display. In the future this may also allow some form of remote access and open up distance learning opportunities with the Protosys station.

With this compact yet sophisticated setup you are able to make interdependent software and hardware changes and readily see their effect in a real system.

2.4 Protosys Custom Hardware Design Goals

Our goals in designing the physical architecture of the Protosys custom hardware were to minimize your wiring drudgery while maximizing design flexibility and pedagogical impact. We also desired to use standardized components to reduce our costs and to give you additional exposure to real-world systems. From the systems perspective we wanted you to focus on the functional interaction of all of the system components. Lastly, it was also desirable that the system be somewhat bullet-proof.

We achieved these goals as follows. The 32-bit data bus and 24-bit address bus of the IBUS may be connected to the memory directly via several DIP connectors, bypassing the protoboard. There are address multiplexers for the memory, also reducing the wiring headache. Special connectors are provided to allow easy multiple connections (16) to be made to the HP Logic Analyzer in most cases. We use a standard dynamic RAM card. All of the control

signals are made available to you, and you have to do something reasonable with them for the system to work. You must design all of the interfaces: IBUS to memory, IBUS to LCD display, IBUS to ICEDnet, etc. Although the 64 bidirectional signals connecting the d-cards are nominally configured as the standard ICED IBUS, they may mostly be arbitrarily reconfigured to suit a student's particular requirements. The d-card IBUS transceivers we selected are able to withstand driving each other in opposite directions, although this should be avoided. They also have built-in thermal protection shutdown circuitry.

We chose a standard dynamic RAM SIMM (Single Inline Memory Module) to give you exposure to the system and interface problems and principles of memory refresh. The SIMM interface design also exposes you to RAS/CAS (row-column) memory device selection. While these features may not be issues for users of today's high-end systems (RAMBUS), they certainly are issues for the memory system designers themselves.

A 10Base2 Ethernet transceiver is used for the physical layer of ICEDnet. It may seem that this is an antiquated standard to follow, but it has pedagogical advantages. You can directly probe the ICEDnet and see actual data collisions, that is an actual Carrier Sense Multiple Access/Collision Detect (CSMA/CD) protocol in action; this is not possible with the more recent 10BaseT standard. While CSMA/CD is not commonly used in today's LAN's, with switching and point-to-point connections becoming the norm, the basic concept of resource contention in a communication medium is still very important, e.g., wireless systems.

2.5 ICED sys-card

The ICED sys-card is the equivalent of a motherboard for the ICED computer, except that the CPU is remotely located in the host. While locating the CPU away from its memory is normally not done, in this case it allowed us to use a basic COTS (Commercial Off-The-Shelf) FPGA board, the EVC1, with its associated software and host interface support. It will also serve to hammer home to you the implications of a relatively slow memory system, a common characteristic of modern computer systems.

The ICED sys-card is shown in Figure 4. It has the following items and connected components:

1. Host computer
2. d-card
3. IBUS connectors - two groups of 32-bidirectional signals each - from host computer
4. ICED computer dynamic RAM - 8 MB standard SIMM
5. sys-card to protoboard connectors - thirteen 16-pin DIP (Dual Inline Package) connectors
6. IBUS signal indicator LED's - 32 green (top) for data, 24 yellow (lower-right) for address and 8 red (lower-left) for control
7. Logic analyzer quick connectors - thirteen 16-signal connectors
8. Logic analyzer digital signal probe - 16 signals
9. ICEDnet connector - 10Base2 physical layer standard
10. LCD display - 16 characters by 2 rows, general purpose
11. DC power outputs - +5, +12, -5, -12 V. - only +5 currently used, for protoboard
12. EVC1 status display - four LED's

While all of this hardware itself resides on the sys-card, all of the data and control connections are brought off of the sys-card to the protoboard. Several DIP connectors are used to communicate with the student's circuits on the protoboard. All of the d-card signals (after the transceivers) go to the DIP connectors, as do all of the signals from the SIMM and the transceiver configuration signals. The sys-card and d-card are custom printed-circuit boards, URI designed. The d-cards were assembled externally, the sys-cards at URI. All of the testing is being done at URI.

2.6 Protoboard and Interface Logic

The protoboard is a generic circuit-prototyping board, composed of several push-in connection arrays. The DIP connectors from the sys-card plug into one or more of these arrays. There is one protoboard per student group, kept by the group for the duration. The board holds the interface and memory control circuits, and also acts as a

patchboard. It is only attached to the sys-card while the group is actively using the laboratory station during a laboratory session.

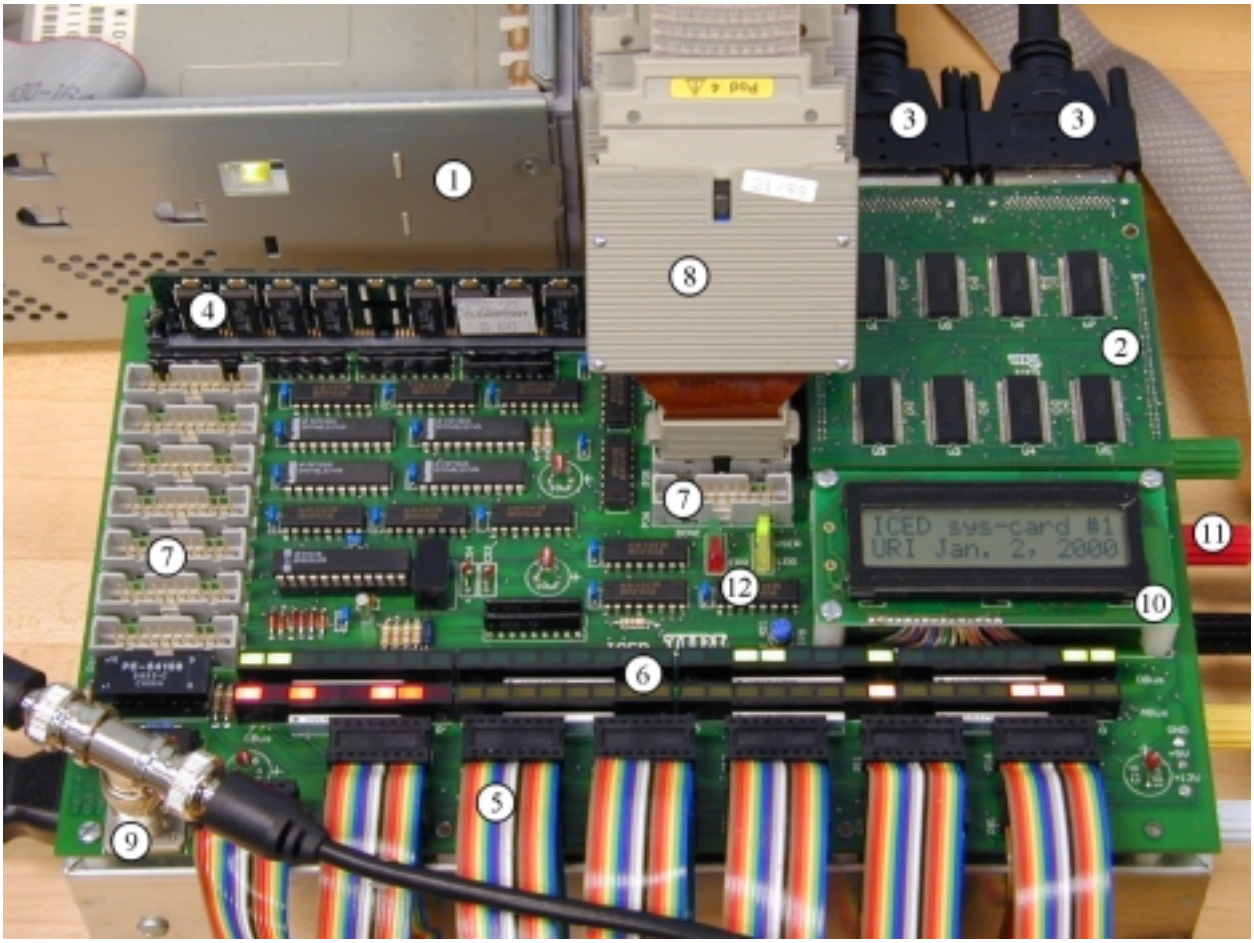

Figure 4. *ICED sys-card.* Photo by Laurette Bradley.

2.7 Logic Analyzer

Each ICED lab station has its own high-speed HP 1662CS logic analyzer containing 64 data channels and an integrated 2 channel digitizing oscilloscope. Each analyzer is viewable-on (X-window) and controllable-from its companion Sun host via the department's LAN. The LAN is also used to download disassembly information to the analyzer, as well as to connect the analyzer to a software analyzer running on the host.

2.8 Summary and Comments

The ICED Protosys laboratory station provides a great deal of flexibility and hardware/software interaction exposure to you. While the trend in the world seems to be to primarily focus on simulation, the Protosys requires you to do real hardware and software design, construction, debug and evaluation.

We will have some interesting capabilities with the lab equipment, once all of the software tools have been installed. In particular, it will be possible to correlate an analog view of a signal with a digital view of the signal, with its particular machine/assembly instruction, and finally with the corresponding original high-level source instruction of

the test program. This will give you the ability to see the effect of a design change at one of these levels on all of the levels, and thereby see the effects of hardware/software tradeoffs in an extremely concrete way.

3 ICED Software Tools and Aids

3.1 EDA Tools

A major element of the ICED experience is a large-scale exposure to modern industrial EDA software tools. They must be used for complex digital designs; it is impossible to realize such designs without using their automation, simulation and verification capabilities before fabrication. Therefore, you must be able to use such tools effectively.

We use commercial software donated by Mentor Graphics and Xilinx. You will follow a typical design flow through the tools[1] [http://www.ele.uri.edu/iced/protosys/eda]. Although these are complex tools, and sometimes tricky to learn, they are used throughout the curriculum, so your learning time is spread over many courses. It is not essential to learn multiple competing vendors' tools, nor is it helpful. Once you have used one schematic capture tool, learning a new one is relatively easy. The situation is similar to programmers learning a new language: one imperative language is about the same as any other, conceptually. By only using one set of tools we are able to standardize the department on the one set and you won't waste time learning extra detailed information.

3.2 Other Software

ICED also uses software design aids. In particular, a disassembler and a symbol utility are used within the logic analyzer to interpret and annotate the prototype's bus operation. Individual Assembly instructions can then be recognized on the bus.

A retargetable assembler called `icasm` has been written to accommodate students' different instruction sets. One version of `icasm` exists for the canned CPU instruction set (IcpuED). You may use `icasm` both directly or to process the output of your own compiler.

A large component of ICED is the writing of a complete compiler for a simple high-level language for your prototype computer. Using special HP software running on the host, and using information generated by the compiler during an application's compilation run (a symbol table), the correspondence between high-level source code and machine instructions is displayed to the student on both the host and the logic analyzer.

4 Reality: ICED Operation and Challenges

All of the design work can be done on any of the department's or the College of Engineering's Sun workstations or Dell Windows NT PC's; a full lab station is needed only at the end of a lab for physical debug and cost and performance investigations.

There are problems with a multi-term/year project: you may get out of sync due to a leave of absence, part-time study, student exchange program participation, etc.; your partner may drop out; or, your group may fall far behind. In order to handle these situations, a canned design is available to you to restart with at the beginning of each core course, should you desire or need to start fresh. As an example, a pre-designed CPU is available at the beginning of the Senior year compiler design course.

We have been seeing a disturbing trend in our students' approach to hardware design. In the old days, making a design error could be close to fatal for a student's project, since much of the hardware was built with handwired small- and medium-scale IC's. (Of course, this is still true for custom chip design.) Therefore students discovered, correctly, that they needed to spend a lot of time thinking about and checking their designs before putting hand to wire. More recently, simulation helped to verify the design.

But now the hardware itself is programmable! If a mistake is made, no problem, it's easy to change: just hack in a fix, reroute and reconfigure. It's easy. That's the problem. These days students have the tendency to approach their hardware design tasks like many approach software design: sit down at the terminal and start hacking with little thought having been given to the task at hand. Although the bad designs thus created do not waste hardware, they certainly do waste the students' time as well as being BAD and BUGGY designs. It is an area of concern; we don't want hardware to start crashing at the same rate as many software packages. Fortunately, consumers seem to have little tolerance for buggy hardware; the problem may correct itself. We still tell the students that the first thing to do is sit down with paper and pencil and keep drawing designs and waveforms and analyzing them until they're sure they have it right; only then go to the EDA tool and start entering the design. Well, some of them listen; more do the second time.

5 Summary, Status and Conclusions

For those of you desiring hardware and software computer engineering design skills, as well as the underlying theoretical knowledge to create complex, realistic digital systems, the new URI curriculum offers a unique experience.

ICED is a work in progress. To date all of the custom hardware has been designed and built; testing is underway. Some tasks remain in setting up the laboratory software. We have begun an evaluation process of the curriculum that so far has had encouraging results[1]. Our evaluations will continue and we will continue to publish them as the results become available.

The URI ICED approach is one solution to solving the competing demands on computer engineering curricula. The focus is learning time-honored principles, reinforced by learning timely skills. The latter will help you step right into industry and be productive.

The ICED curriculum is a novel attempt to bring the problems and opportunities of a long-term project into the undergraduate curriculum, allowing you to engineer complex digital systems from both software and hardware components, and to comprehend the structure of these systems, from networks to gate-level logic in a CPU. Many aspects of computer architecture are investigated, including Instruction Set Architecture, microarchitecture, system architecture, and network architecture. Hardware/software tradeoffs are studied throughout the curriculum, the old-fashioned way: *'You build it.'*

Acknowledgements

We are indebted to Carlo Tognina for his excellent and dedicated work on the ICED hardware.

ICED is supported by many institutions and companies. Please see http://www.ele.uri.edu/iced for a current and complete list.

References

[1] A. K. Uht, J.-C. Lo, Y. Sun, J. C. Daly and J. Kowalski, "Building Real Computer Systems", *IEEE Micro*, May-June 2000, pp. 48-56, in press.
[2] A. K. Uht and Y. Sun, "The Laboratory Environment of the URI Integrated Computer Engineering Design (ICED) Curriculum", in *Proceedings of the 1998 Frontiers in Education Conference*, ASEE and IEEE, Tempe, Arizona, November 1998.
[3] A. K. Uht, "The URI Integrated Computer Engineering Design (ICED) Curriculum: Progress Report", in *Proceedings of the ASEE 2000 Annual Conference*, St. Louis, Missouri, to appear June 2000.