The URI Integrated Computer Engineering Design (ICED) Curriculum: Progress Report

Augustus K. Uht
University of Rhode Island

Abstract

The University of Rhode Island started the ICED curriculum in the Fall of 1997. The key feature of ICED is a substantial 2-3 year long project tying together important but normally disjoint computer engineering concepts across the major. The students learn how to make critical hardware/software tradeoffs with long-term implications. Courses in processor design, compiler design and networks are required, and tied together through the major project. This keeps students motivated in novel ways: they enjoy discovering the implications of decisions made in one area, across all other areas over the span of years.

This paper reports on the status and progress of ICED after two years of startup operation. Some custom hardware was required for the curriculum; it has now been designed, a prototype for one lab station has been built, and the hardware for the other stations is under construction. In our original NSF grant we noted the need for additional funds to build this hardware and otherwise equip more lab stations; we applied for and received these funds from the Champlin Foundations of Rhode Island in 1998/9. The students have received the new curriculum enthusiastically and have learned much. It has also been a learning experience for the faculty involved. We are immediately plowing what we learn back into the design of ICED and its core courses, in order to fully achieve and enhance our curricular goals. The paper includes descriptions and documentation of all of the above.

I. Introduction

Traditional computer engineering curricula include unconnected coursework with isolated project work leaving students to the erroneous conclusion that hardware and software design are separate and exclusionary activities, with little effect on each other. This is particularly unsatisfactory since complex embedded-systems designs are being used more and more in the “real world”.

The Integrated Computer Engineering Design or ICED curriculum[1] at the University of Rhode Island seeks to remedy this by teaching students to make key hardware/software tradeoff design decisions. This is achieved with the following ICED components: 1) a 2-3 year long project spanning six or more hardware and software courses; 2) required coursework including advanced software topics such as compiler design, as well as a full slate of hardware courses; 3) the use of modern commercial EDA (Electronic Design Automation) tools[2]; and 4) custom hardware[3] and software to enhance the laboratory experience of the students.

ICED was begun in 1997 with funding from the National Science Foundation. In 1999 further funding was obtained from the Champlin Foundations of Rhode Island. This paper is a status report on the development of ICED. Briefly, almost all of the COTS (commercial-off-the-shelf) hardware and software has been obtained and setup, the custom hardware has been designed with prototypes built and with the remaining units under construction, student response has been positive and the students have been learning a lot. The instructors have also been learning from the experience.

The remainder of this paper is organized as follows. In Section II details of the current state of the laboratory equipment are given, in Section III the students' experience with ICED is briefly reviewed, the instructors' impressions and the lessons learned to date are presented in Section IV, and we conclude in Section V.

II. Laboratory Equipment Status

The current multi-year ICED project is the design, construction, debug and evaluation of a complete computer system, including compiler, central processing unit, memory interface and network interface. The overall goal is to have multiple students' computers communicating with each other on an Ethernet-like network. The student 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 laboratory setup was designed with these objectives in mind.

The preliminary architecture of the laboratory equipment has been presented earlier[3]; this was before the custom hardware had been built. When ICED was first started we obtained most of the COTS hardware needed, including EVC1 circuit boards from Virtual Computer Corporation. The EVC1 contains a Field Programmable Gate Array (FPGA) from Xilinx Corporation. FPGA's consist of logic elements whose functions and interconnections may be reconfigured an unlimited number of times. The FPGA on the EVC1 is used as the basic hardware element for the students' hardware designs. The FPGA is large enough to hold a simple 32-bit processor.

A. ICED Protosys Laboratory Station

To date all of the custom hardware has been designed and the prototypes built. A picture of the overall ICED hardware laboratory setup as realized is shown in Figure 1. 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*

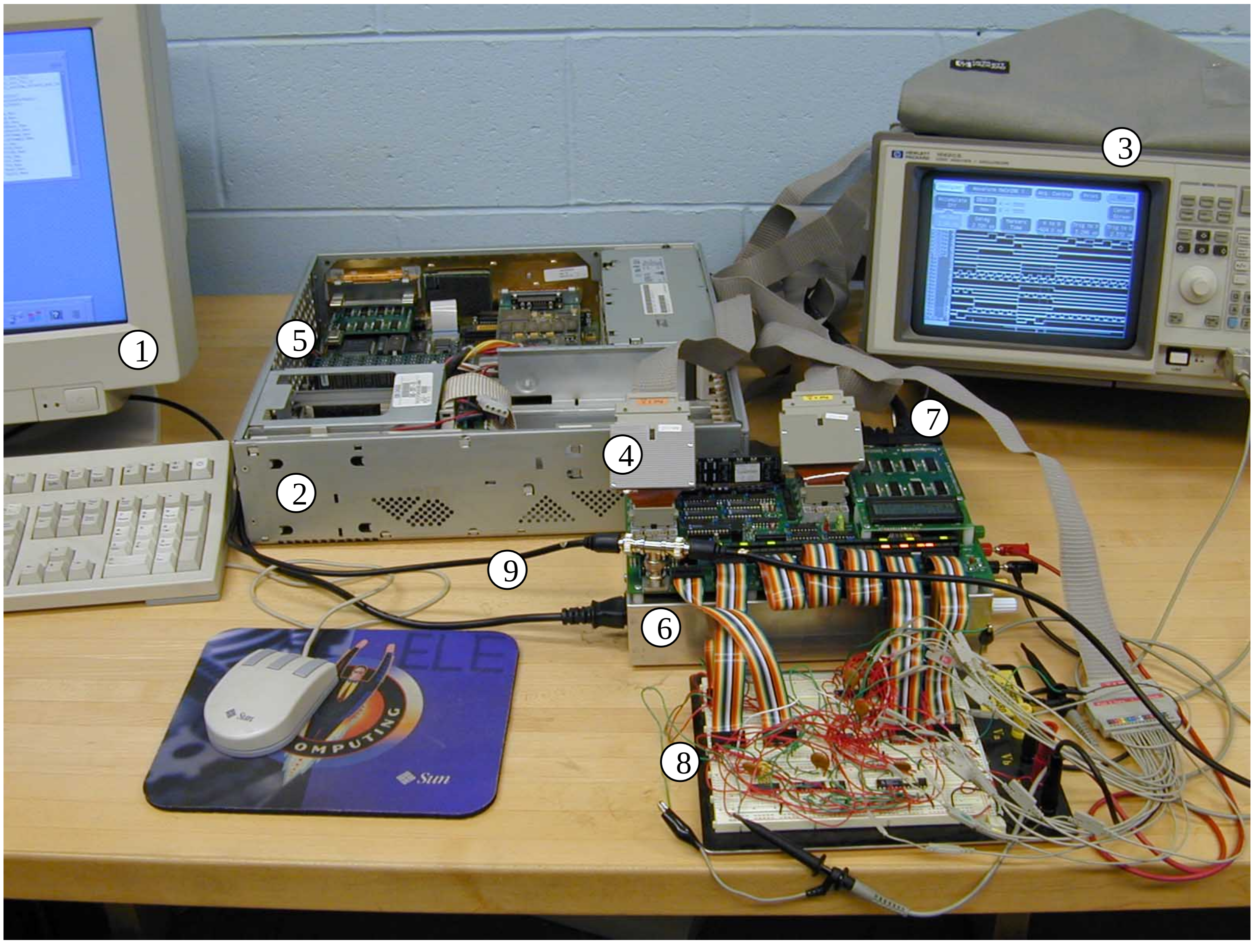

Figure 1. ICED Protosys Laboratory Station.

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. Simple programmable logic is made available to the student for this hardware.

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 students are able to readily make interdependent software and hardware changes and see their effect in a real system right away.

B. 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 FPGA board, the EVC1, with its associated software and host interface support. It will also serve to hammer home to the student the implications of a slow memory system.

The ICED sys-card is shown in Figure 2. 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
5. sys-card to protoboard connectors - thirteen 16-pin DIP 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
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

The sys-card contains the connections to the IBUS (from the ICED CPU in the host), the ICED computer's main memory, the ICEDnet (with a transceiver) and a general-purpose LCD display device. 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. As mentioned earlier, in order for the computer to work the student must add the interface or "glue" logic between the IBUS, the main memory and all of the I/O devices. In order to reduce the wiring overhead to the student, at least in the early stages, the IBUS data and address lines can be connected to the memory via jumper cables without going through the protoboard. The IBUS control lines, however, must be generated or driven from the protoboard. Per-signal LED's are provided on the sys-card for every IBUS signal, to further aid in system debug.

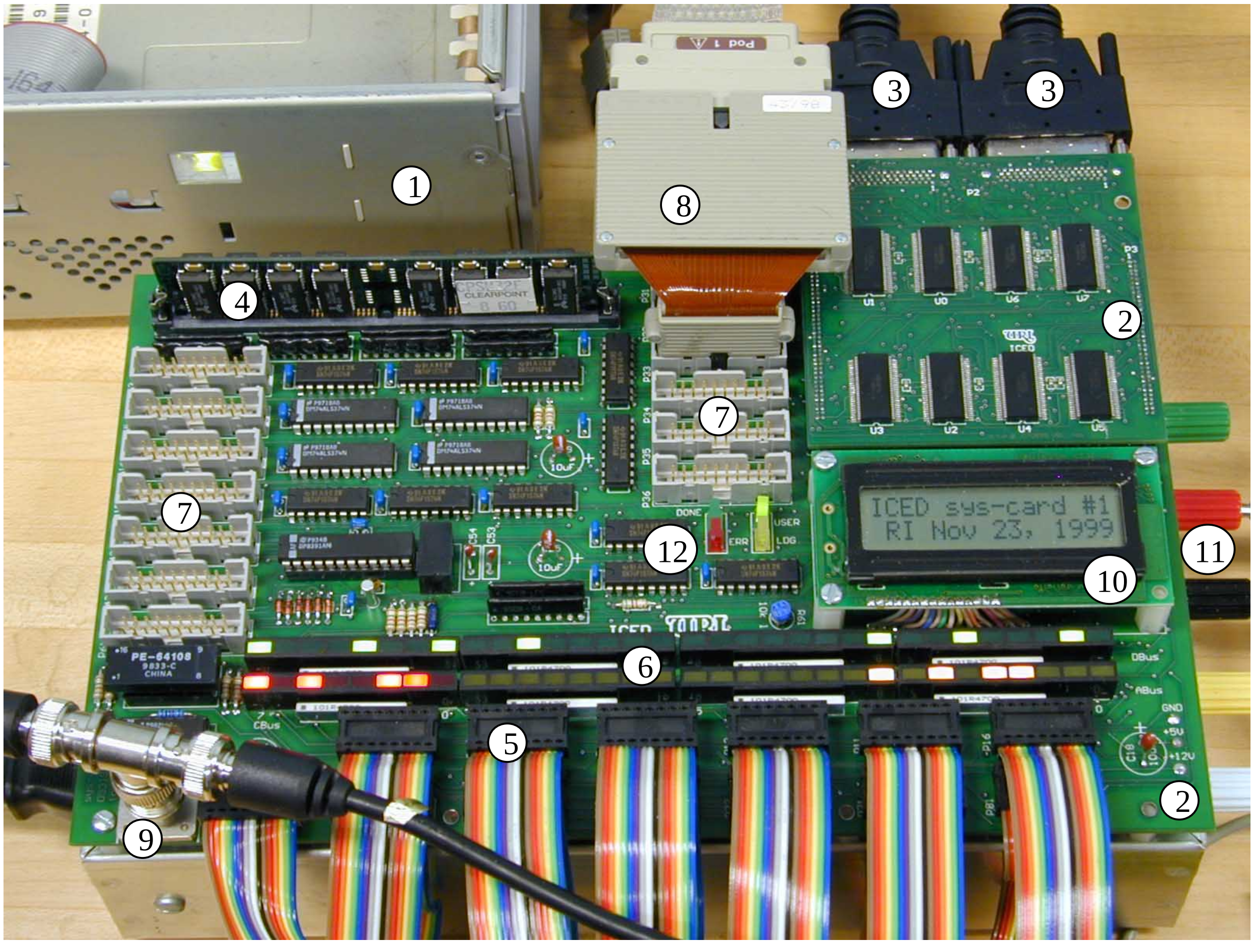

Figure 2. ICED sys-card.

C. Summary

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

III. Students' Responses and Evaluations

The students have enthusiastically embraced the ICED curriculum. In departmental exit interviews conducted by the URI College of Engineering last year, the following comments were received in response to the question: "What would you recommend the University do to improve the quality of the undergraduate education?":

> "[The] Curriculum change [to ICED] was good…."
> "The ICED curriculum is an incredibly good idea."
> "Keep the ICED curriculum."

The students have also rated many of the ICED core course highly in the standard URI "Student Evaluations of Teaching" surveys conducted at the end of every semester.

On the down side, there was a general complaint of “too much work” as well as a general frustration with the “gotchas” present in the EDA design tools, a major cause of wasted time. Other comments from the exit interviews cited above were:

> (“[The] Curriculum change [to ICED] was good….”) “….but we were guinea pigs.”
> (“The ICED curriculum is an incredibly good idea.”) “If it becomes too central,
> the program would loose [its] broad appeal.”

In prior work[3] we made a preliminary study of the effect of the ICED curriculum on students’ hardware/software tradeoff-making abilities and on their computer engineering abilities in general. The data indicated that ICED students had statistically-significant better abilities than non-ICED students in both cases. This study is on-going and further results will be reported later.

IV. Instructors’ Impressions - Lessons Learned

A. Student Issues

The students have had a great deal of difficulty completing the projects, not so much because of the difficulty of the material but because of the complexities of the EDA tools. While the EDA tools are absolutely necessary to get the job done and are usually fun to use, often work is wasted due to strange bugs in a tool. This causes great frustration for the students, as well as putting great pressure on their limited available study time. With such large projects it is critical that the instructor and TA be well-versed in these “gotchas” and be able to warn the students about them.

Fortunately, the EDA tools have improved markedly over the last couple of years. In some cases we had been using the tools shortly after their first release, when software bugs are more likely. On the other hand, the problem does not go away completely because every new feature and every bug-fix has the often-realized potential for introducing new bugs and “gotchas”. A more lasting solution may be to ensure that all faculty and TA’s go through the entire design tool cycle once a year, after new versions of the tools have been installed and before the school year begins.

Another EDA aspect is the learning curve for the EDA tools. In order to do basic designs all the way through to the FPGA level it is necessary to learn several complex tools and associated skills. This year we reduced the number of new tools learned in the first heavy design core course, and also relied on the students’ grasp of a tool learned in a prior year. The other new tools necessary to be learned will be postponed one term. It remains to be seen if this solution is effective.

From a pedagogical perspective, given the time requirements of the large projects it is paramount that the basics of digital design and computer operation be grasped early on. While the following may be obvious to many teachers, it is important to restate that it is better to learn a few ideas well than many not at all.

B. Equipment and Equipment Development Funding

Designing and building custom hardware for curricula is very rewarding for the instructor and its use is hopefully rewarding for the students (they have not yet used the custom hardware part of the

Protosys stations). However, it is incredibly time-consuming. We had few funds for TA support for the construction of the hardware and are barely making ends meet. This situation is being remedied by the new system at NSF in which personnel support funds may also be requested at the same time and in the same grant as equipment funds.

In retrospect, rather than starting up the new curriculum immediately and performing just-in-time coursework and equipment preparation it perhaps would have better to have spent a year in preparation before actually letting the students loose on the material. At the time I felt it imperative to update the curriculum as soon as possible; I was perhaps over-zealous.

V. Conclusions

The ICED program at URI is well underway. The equipment is either in hand or being constructed. In spite of the warts, URI students and instructors have found the new ICED curriculum extremely rewarding and exhilarating. The students and instructors are learning much. And we don't even have it all working yet.

Acknowledgements
This work was partially supported by the National Science Foundation through grant DUE-9751215, in part by a grant from the Champlin Foundations of Rhode Island, in part through matching funds from the Office of the Provost at URI, in part through hardware donations from Clearpoint Co. and Virtual Computer Corporation, and in part through software donations from Mentor Graphics Corporation, Xilinx Corporation and Virtual Computer Corporation. We are greatly indebted to all of our supporters.
The ICED curriculum is the product of many people in the Department of Electrical and Computer Engineering, the Department of Computer Science and Statistics, and the Instructional Development Program at the University of Rhode Island.

Bibliography

1. Uht, A. K. The Integrated Computer Engineering Design (ICED) Curriculum. In the *IEEE Technical Committee on Computer Architecture Newsletter*, IEEE, February, 1999.
2. Uht, A. K. Use of CAD Tools in the Integrated Computer Engineering Design (ICED) Curriculum. In the *Proceedings of the 14th International Conference of the Mentor Graphics Users' Group*. International Mentor Graphics Users' Group, Portland, Oregon, October, 1997.
3. Uht, A. K. & Sun, Y. The Laboratory Environment of the URI Integrated Computer Engineering Design (ICED) Curriculum. In the *Proceedings of the 1998 Frontiers in Education Conference*, ASEE & IEEE, Phoenix, Arizona, November, 1998.

AUGUSTUS K. UHT
Augustus K. Uht is an Associate Professor at the University of Rhode Island and is currently on sabbatical leave as a Visiting Scholar at Northeastern University. He received his B.S. and M.Eng.(Elect.) degrees from Cornell University in 1977 and 1978, and his Ph.D. from Carnegie Mellon University in 1985, all in electrical and computer engineering. From 1978 to 1982 Prof. Uht was an engineer at International Business Machines Corporation in East Fishkill, N.Y. He is a licensed Professional Engineer in New York and Pennsylvania. Prof. Uht received the URI 1998 Aurelio Lucci Award for Faculty Excellence in Electrical Engineering. He was a URI Faculty Teaching Fellow in 1995-1996.