A Combined Virtual and Remotely Accessible Microprocessor Laboratory*

Helmut Bähring Jörg Keller Wolfram Schiffmann
FernUniversität Hagen
FB Informatik
58084 Hagen, Germany
{helmut.baehring|joerg.keller|wolfram.schiffmann}@fernuni-hagen.de

Abstract

We present a microprocessor lab that is accessible remotely, i.e. students can control the hardware in the lab from their computer at home. At the same time the lab also provides the features of a virtual lab, i.e. students conduct experiments on simulators. Both modes of the lab are run under a common user interface. This lab is suited especially for distance education, as students can develop and test their code offline, while still conducting their experiments on real hardware. This lab is thought to replace a traditional lab course where students had to be present on campus for one week fulltime, which is quite difficult to realize in distance education.

1 Introduction

FernUniversität is Germany's Distance Teaching University, serving also Austria and Switzerland and German speaking students in other parts of the world. The Computer Science Department offers study programmes leading to Bachelor, Master, and PhD degrees, respectively. Course texts are made available to the students in electronic form, preferably pdf, via a portal where students can access the courses they are enroled in. The course pages also provide study materials in addition to course texts, such as applets, link lists, multimedia enhancements, and the like. Additionally, paper copies of the course texts are sent via regular mail to the students. Assignments are submitted either in paper by mail or electronically, preferably via a system called WebAssign [9] that provides a web interface to the students and achieves automatic distribution of assignments to correctors and notification of submitting students about results and solutions when the assignments are corrected. WebAssign also allows to add modules for automatic grading of submissions, for simple cases such as multiple choice, but even for the submission of circuit schematics [5]. Students can ask questions or seek help over a number of communication channels: telephone, telefax, email, newsgroups, and (partly) video conferences over the Internet. Our method of teaching allows students to follow their study programme at any place and at any time. As more than 85% of our students work, this is a necessity in order to give these students a chance of success, as regular attendance of meetings at fixed times (even via Internet), is difficult or impossible for a majority of them.

Yet, the German computer science curriculum requires laboratory courses as part of the undergraduate study programme. For the reasons mentioned above, laboratory sessions on-campus are difficult to realize, even if they are compressed to a single one-week session close to the end of the semester, where that session is preceded by studies and preparations at home. This calls for virtual laboratories. While a virtual programming lab can be achieved with usage of the internet and tools for student collaboration, a virtual microprocessor lab proves more difficult.

First, while a number of simulator tools are available to create a virtual lab, they necessitate the installation of software on the student's computer. Yet, FernUniversität's students access the Internet frequently from their workplace, where they often do not have rights to install software. This would call for a lab with browser access, where all software is available as applets. However, students at home suffer from the amount of online time necessary in this case, and all students would suffer from the applet download times. Second, simulators never give a complete picture. Especially in applications where signals are to be checked or produced in a manner that includes a certain real-time behavior, a simulator is clearly not sufficient. While this can be cured by a remote lab, where a physical device is in the lab at FernUniversität and can be programmed, run and observed remotely, this again incurs long online times. Additionally, in order to serve a large number of students, the necessary amount of hardware installation would be enormous. We overcome these contradicting requirements and constraints by establishing a microprocessor lab that is both remotely accessible and virtual.

*This Research was partly funded by FernUniversität's Innovation Fund 2003.

Laboratory experiments are indispensable parts of most universities' engineering education programs. During the last decade different kinds of remote labs were developed. They can be divided into three categories: 1. electronic devices, 2. control engineering and robotics laboratories, and 3. digital logic and microprocessor systems. In the first category, the main focus is to measure the electrical characteristics of semiconductor devices. A good overview of current remote laboratories of this kind can be found in [4, 8]. In [1] remote laboratories for electrical and mechanical engineering are described. In [10] a control engineering laboratory is proposed where the students have to design, implement and test a discrete controller on a real plant that can be remotely accessed. Examples of remote laboratories for digital logic and microprocessor systems can be found in [6, 7]. In this paper we propose a new microprocessor lab that provides not only remote access but also an integrated simulation facility to prepare experiments while not being connected to the internet. To conduct the experiments remotely, the students can use the *same* graphical interface as used for the simulation. We are not aware of any other lab that tries to combine both modes under a single user interface. Also, our approach is in contrast to known online labs as it tries to reduce online time. Hence, we believe our approach to be novel.

The remainder of the paper is organized as follows. In Section 2, we detail the requirements and constraints of a microprocessor lab in distance education. In Section 3, we describe the implementation of our lab. In Section 4, we present some case studies of how our installed infrastructure will actually be used in lab courses. In Section 5, we give a conclusion and an outlook on further applications and activities.

2 Requirements and constraints for a virtual and remote microprocessor lab

If students want to get familiar with microprocessor programming and deployment, they must have access to a software development environment and a target system. Our computer engineering lab course[1] is divided into three phases. First of all, the students work through a course text that describes the basics for the experiments. For instance, the processors to be programmed are described, and the pitfalls and fallacies of assembler programming are explained. In the second phase they use development tools to practice machine language programming. The development system consists mainly of an editor and an assembler for compiling machine programs into the microprocessor's object code. These tools — cross-assemblers in particular — are widely and freely available, and are either installed locally at the students' computer or accessed remotely. In the third phase the students test their programs on a target system. For this purpose, we had to develop an appropriate target system that operates and looks like the real microprocessor system which is used in our laboratory on campus.

The target system should provide not only a simulator but also serve as a remote control for a real microprocessor system in Hagen, so that the execution of students' programs can be observed in real-time. In the simulator mode the students can use the target system as a *virtual* laboratory for testing the object code at home. Although they can test the functionality of their programs in this phase by simulation, they are not able to either check the real-time behavior, nor can they control external hardware that is connected to a real microprocessor system. In order to provide these facilities the students should be able to connect via the Internet to a real target microprocessor system located at the lab on the Hagen campus, i.e. they use a *remote* laboratory. By switching into the remote control mode the students will be able to remotely control an experimental setup in the lab, consisting of the processor itself and some additional hardware. By means of additional measurement devices that can also be controlled remotely the students can conduct experiments with external hardware (e.g. a traffic light panel). In addition to the software tools, web cams provide live pictures of how the setup in the lab behaves, e.g. which lights are on or off.

Students thus develop their programs locally and test them on a simulator. If they do not have the rights to install software on the computer they are using, they can also use a development system on a computer in Hagen via the Internet. When a student is confident that his program works, then he accesses the remote lab, uploads his program to the target system, and conducts his experiment on real hardware. If an error occurs or the real-time behavior differs from what is expected or required, he corrects the program until he is satisfied. In the end, he submits his results and his program via the WebAssign system to the correctors for grading. In this manner, the virtual lab mode saves valuable online time for many students, as they can develop at home in an offline mode. This also distinguishes our approach from online labs.

Despite the two modes of operation, students should not be required to learn two sets of user interfaces, as this would incur a lot of additional learning time. Hence, the virtual lab and the remote lab should share a common user interface. For example, a student should be aware whether he is running code on a simulator or on real hardware, but the controls for starting the program run should be the same.

[1] In the German curriculum, a basic computer engineering education is part of the computer science programme.

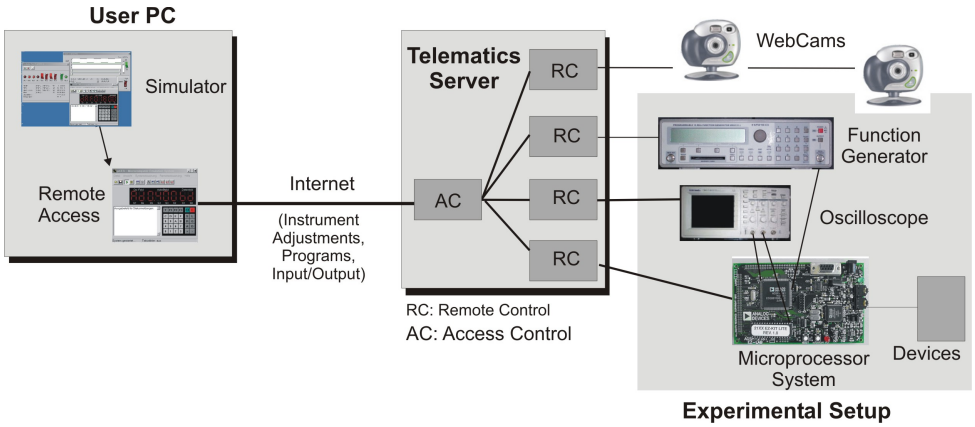

Figure 1: Structure of the microprocessor lab.

3 Implementation of the microprocessor lab

The remote laboratory is realized by a client-server approach. Students who want to use the remote lab have to download a number of client programs that serve as remote controls for the microprocessor system and the measurement equipment, such as an oscilloscope or a web cam. These clients connect to server processes that run on a server computer (named telematics server) in the laboratory at Hagen. Some of those clients also have simulator functionality (e.g. the microprocessor client), so students can seamlessly switch between the virtual lab and the remote lab. The server computer in turn is connected to the hardware devices in the lab: microprocessor card, oscilloscope, web cam, and so on. The microprocessor card itself is connected to some additional hardware such as a traffic light kit. The server also provides a central installation of the development software and the simulator for students who are not allowed to install software on their local computer. Those students may access the server either via a remote shell or session, or web-based via VNC [11]. Figure 1 outlines the scenario of our microprocessor lab.

To accomplish more complex experiments with attached devices, such as a radio controlled clock (see next section), the simulator establishes a TCP/IP-connection via the internet to the server at our laboratory, thus allowing students to switch seamlessly to the remote lab. After authentication, the user gets access to the remote control software of different components of the experimental setup. These components include one or more oscilloscopes, function generators and mainly the microprocessor system hardware. Attached to this system will be different devices, e.g. the above mentioned radio controlled clock or a model of a traffic light.

The user is able to upload his programs to the remote microprocessor system via the user interface and to start them there. He can adjust the already mentioned instruments remotely, perform different measurements, and gets back the results. To give him the possibility of observing the experiments, and thus the feeling of being in the laboratory room, all instruments and devices — but also the whole laboratory — are accessible by controllable web cams.

The client programs are preferably programmed in Java. The software is programmed in two parts: one part that is independent of the particular device for which the client is the frontend, and one part which realizes the particular device's frontend. The former part can be re-used for all devices, the latter part is programmed in a way that realization of an additional device can be done with the least possible effort. Using Java also had the advantage of being more independent from the type of computer the students have locally, than with any other realization. Figure 2 depicts the microprocessor user interface in a close-up.

Of course, the implementation of the lab is not restricted to the particular microprocessor card. There are also cards with a microcontroller and a digital signal processor, respectively, that can be accessed in the same manner. Also several other experiments can be realized, see Section 5.

4 Case studies

In this section we want to demonstrate how the virtual and remote lab can be used by the students. When the students have worked through the course text, which describes the basics for the experiments, they can conduct the experiments. In all cases they use the microcomputer simulator depicted in Figure 2. Note, that the graphical interface of this simulator can also used as a remote control during the remote laboratory experiment. By means of a pull down menu the user can

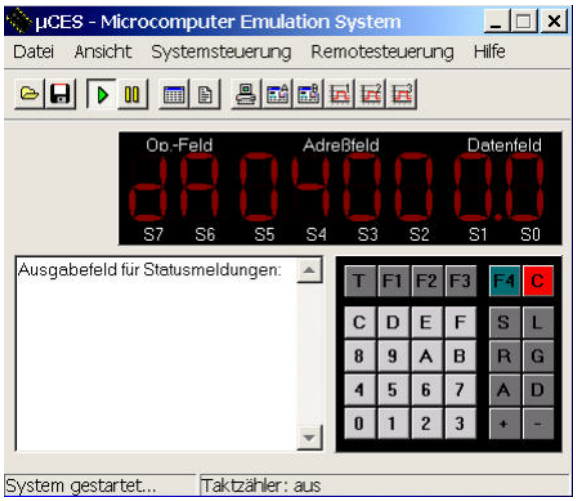

Figure 2: Microcomputer simulator with integrated remote control.

switch between the two modes. For the remote mode the IP of the remote lab server must be entered.

4.1 Decimal counter

In this experiment the students have to write a assembler program for a decimal counter that is incremented every second. At first, the experiment should be conducted by means of the microcomputer simulator. The students enter, edit and assemble the program on their home PC and load it into the simulator. Then the object code is launched and a four digit starting value is entered via the keyboard. The counter will be started by pressing the character '+' and the current counter value will be displayed on the LED display. If the experiment has been successfully conducted in the virtual laboratory, the students can switch to the remote laboratory by using the corresponding pull down menu. They enter the server IP and will get connected to a real microcomputer system located in the computer engineering laboratory at Hagen (see Figure 3). Now, the program runs on the real microcomputer while the display can be observed by means of a web cam.

4.2 Radio controlled clock signal

The objective in this experiment consists of writing a program that should emulate the signal of a radio controlled clock signal. While the development cycle is identical to that for the decimal counter experiment, we need an additional measurement device here to watch the generated signal. For this purpose we implemented a client-server program suite that allows to remotely control an oscilloscope at Hagen and to display the measured curves on the student's PC (see Figure 4).

5 Conclusions and outlook

We have realized a microprocessor lab course that serves the requirements of students in distance education: minimization of online time, avoidance of downloads and local installations if possible, and access to real hardware. At the same time, we fulfilled our curricular requirements: performing experiments complex enough to teach the students low-level programming of microprocessor hardware. This is accomplished by two modes of operation: virtual lab and remote lab. The switching between these modes is seamless, relieving the students from having to learn several user interfaces for the same device. Our own software development has been reduced to a minimum by re-use wherever possible. The achievements so far have resulted in a shortening of the on-campus phase from one week to two days for the next offering the lab course in spring 2005.

There are further experiments, for which virtual and remote versions have been developed, and which will complete the virtualization of the microprocessor lab in the future. These are: digital logic, where a GAL can be programmed remotely, a digital signal processor, which is used to implement a filter for a signal which is generated by a function generator, and a microcontroller which is programmed for a control task. The function generator can be controlled remotely in the same manner as the oscilloscope. The digital signal processor and the microcontroller can, in principle, be accessed in the same way as the microprocessor. However, the simulator for the digital signal processor was not freely available in Java, but provided with the development software for a particular operating system. Also, the user interface for the card itself is separate from the development software and not freely available. Hence, we did not find a way so far to provide students with a seamless switch between virtual and remote modes for this experiment. Furthermore, the control of the hardware necessitates a remote session or use of VNC.

The (partial) virtualization of the microprocessor lab continues our transformation to a web-based computer engineering education, as set out in [2], which started by integrating a circuit design tool, submission of assignments via the Internet, and semi-automatic assessment and grading of assignments with student circuits in a beginners course [5]. The integration of the simple scalar tool set [3] into a course on advanced computer architecture is still underway.

We still have to accomplish a kind of reservation scheme for the physical devices. Currently, the remote lab is allocated on a first-come-first-serve basis, which is not the optimal choice at times close to deadlines for assignments. We also plan to extend this type of lab access to other computer engineering lab courses.

Figure 3: Real microcomputer at Hagen university.

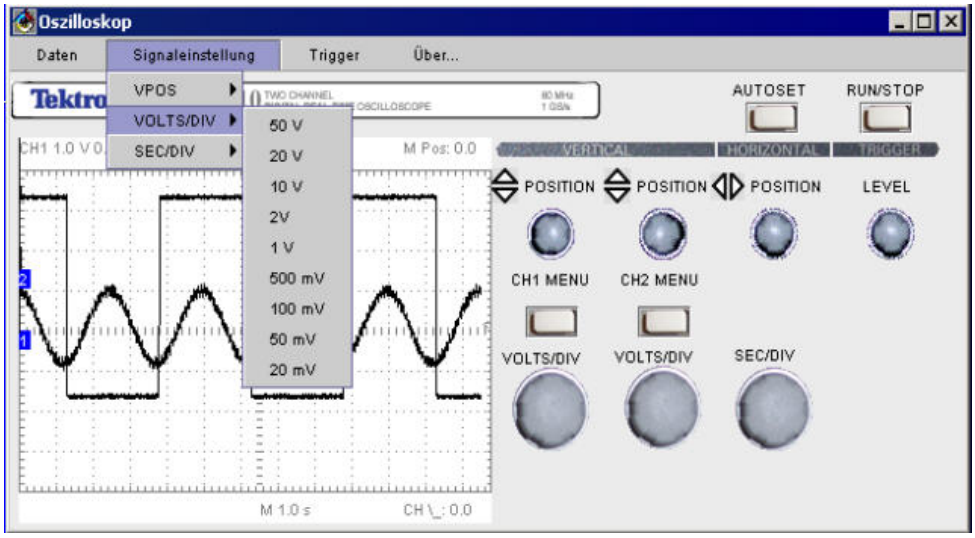

Figure 4: Remote control for the oscilloscope.

Acknowledgements

The research presented here was partially funded by FernUniversität's Innovation Fund 2003. We thank Bernhard Fechner for implementing part of the web interfaces for the lab devices.

References

[1] B. Alhalabi, D. Marcovitz, K. Hamza, S. Hsu. Remote Labs: An innovative leap in engineering distance education. IFAC, 2000.

[2] H. Bähring, J. Keller, W. Schiffmann. Deployment of New Media at FernUniversität Hagen. In Informationstechnik und Technische Informatik, vol. 43 no. 4, 2001, pp. 215-218.

[3] D. Burger, T. M. Austin. The simple scalar tool set. Tech. Rep. TR-1342. Computer Science Dept., Univ. of Wisconsin, Madison, 1997.

[4] T. A. Fjeldly, M. S. Shur. Lab on the Web: Running Real Electronics Experiments via the Internet, Wiley-VCH, 2003.

[5] U. Hönig, J. Keller, W. Schiffmann. Web-Based Exercises in Computer Engineering. In Proc. International Conference on Networked e-learning for European Universities, Granada, Nov. 2003.

[6] S.-J. Hsieh, P. Y. Hsieh, D. Zhang. Web-based simulations and intelligent tutoring system for programmable logic controller. ASSE/IEEE Frontiers in Education Conference, Session T3E, IEEE 2003.

[7] Y. Ko, T. M. Duman, A. Spanias. On-line laboratory for communication systems using J-DSP. ASSE/IEEE Frontiers in Education Conference, Session T3E, IEEE 2003.

[8] Z. Nedic, J. Machotka, A. Nafalski. Remote laboratories versus virtual and real laboratories. ASSE/IEEE Frontiers in Education Conference, Session T3E, IEEE 2003.

[9] H. W. Six, G. Ströhlein, J. Voss. Evaluation of WebAssign. In 20th World Conference on Open Learning and Distance Education (ICDE Congress). Düsseldorf, April 2001.

[10] J. Tuttas, B. Wagner. Distributed online laboratories. Intern. Conference on Engineering Education, 2001.

[11] RealVNC. `http://www.realvnc.com/`.