

Remote Operation and Control of Computer Engineering Laboratory Experiments

Helmut Bähring Jörg Keller Wolfram Schiffmann
FernUniversität in Hagen
Fakultät für Mathematik und Informatik
58084 Hagen, Germany
{helmut.baehring,joerg.keller,wolfram.schiffmann}@fernuni-hagen.de

Abstract

We present a lab-in-a-box, a metal case needing only power supply and network access, to provide a complete infrastructure to establish a microprocessor laboratory for embedded applications in computer engineering. The laboratory is used in distance teaching and thus all instruments and devices are controlled and observed via the net. Our focus is on flexibility to setup experiments, and on scheduling access to such a laboratory.

1. Introduction

In computer engineering the education in classes should be complemented by laboratory sessions where students can get hands-on experience. A multitude of experiments can be thought of. However, lab instruments are expensive, so that a small number of them has to be shared by many students, who have to be scheduled for lab hours, and experiments' setups can only be changed when all groups have completed the previous experiment. Normally, this can still be handled by scheduling different groups to perform one experiment within one week, before changing the setup for the next week. However, our institution is Germany's distance teaching university, serving German speaking students all over the world. Course materials are available online for registered students (a reason why we cannot make our materials publicly available), but a lab course either requires access to the instruments, i.e. physical presence, or is restricted to simulation experiments. While those still provide some insight, they do not show everything, especially in digital signal processing where real-time requirements have to be obeyed (see e.g. the clock decoding example in Section 4). It is possible albeit difficult to get our students to the campus for a few days, but they would complain, and with reason, if they would have to spend most of their time

waiting for access to the lab instruments.

In order to attack all those problems, we established a microprocessor lab course with internet access, and reported about it at the Workshop on Computer Architecture Education two years ago [2]. Yet, our solution at that time was kind of a prototype. It needed an expensive front-end server to control all the instruments of an experiment, and changing of experiment setups was cumbersome. Also, the scheduling between different student groups was only on a first-come-first-serve basis. In a recent publication [3], we envisioned a so-called lab-in-a-box, i.e. a metal case with only a plug for power supply and a plug for network connection. The case should contain all the necessary functionality to connect the instruments and devices with the network, the power supply should be controllable via the net. The realization of this vision, on which we report here, was enabled by small devices called XPorts that bridge between network and instruments, and additionally contain the applets that enable the remote control via the net. Thus, after programming one XPort for each instrument device, a generic lab infrastructure is established that can be provided in larger numbers for small cost. Change of experiments only requires exchange of instruments, and replugging. With programmable switches, even this can be avoided.

Also, we have coupled the lab with a system for collaborative work, where it is possible to reserve experiments in advance, exchange experiences between students during and after the experiments, and where it is possible to pre-configure experiments by the system, i.e. if a student logs in to perform the next experiment in the sequence, the system can switch on exactly those components that the student needs.

While our experiments are not directed towards microarchitecture, they are oriented towards computer architecture by encompassing the use of different types of processors: microprocessors, digital signal processors, and microcontrollers, so that the students learn about their differences.

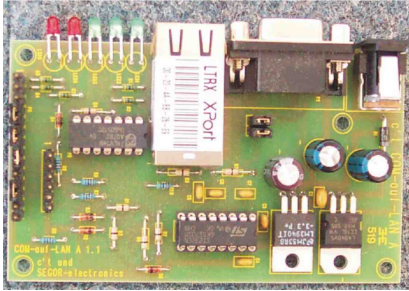


Figure 1. Lantronix's XPort provides a LAN-to-serial bridge and a webserver.

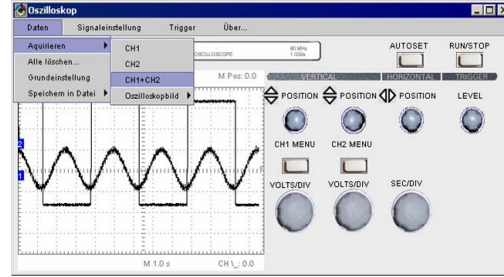


Figure 2. Remote control applet for the Tektronix TDS oscilloscope.

The remainder of this article is organized as follows. In Section 2 we discuss related work. In Section 3, we briefly present the XPort devices that enable encapsulation of the control software and network access to instruments. In Section 4, we report on experimental setups for micro-controllers and digital signal processors. In Section 5, we detail the steering and scheduling of student groups for the laboratory with the help of a tool for collaborative learning. In Section 6, we draw conclusions and give an outlook on future work.

2. Related work

The number of universities that provide remotely controllable laboratories is increasing steadily. While simulations are cheap and easy to establish they can only model a limited degree of realism. A satisfactory simulation of the dynamic behavior of complex systems requires much computing power and even if such computing capability would be available it cannot accomplish real-time quality. Especially, the simulation of embedded computers is very challenging and a simulation of an electron microscope would be still impossible.

Thus, in order to develop and test the real-time behavior of those systems we have to rely on the real hardware instead of simulations. Remote laboratories allow the access to experimental setups that can be operated and controlled 24 hours on 7 days a week. Beside the device that should be investigated, the experimental setups usually comprise several expensive instruments for stimulation of the device and measurement of the device's reaction. The spectrum of remotely controllable instruments can range from a quite low budget oscilloscope up to an extremely expensive electron microscope.

In the recent literature different categories of remotely controllable experiments were described: Analysis of the characteristics of semiconductor devices [8], setup of electronic circuits and measurement of their behavior [6, 1],

physical observations like mechanics experiments [9], or analysis of the characteristics of the upper atmosphere [14], different kind of control systems for studying the parameter setup and analysis of the dynamic behavior of the control loop [10], communication engineering experiments [5, 15], and circuits of digital programmable logic [13].

Beside the description of specific remote lab systems also general distance learning and pedagogical issues were addressed. In [12] a framework for analyzing the effectiveness of remote labs is proposed. [4] mainly focuses on issues that impinge on the specification and design of remote labs. He also identifies the open problems that must be solved to achieve a widespread acceptance of remote labs in education. The main deficits of current remote laboratories concern the lack of a unified appearance of the instruments' remote control panels, the users' accessibility and the support for collaboration among the lab's participants. Only a few papers describe support for user management and toolkits for the experimental setups [8].

In this paper we offer solutions to the deficits cited above. We will demonstrate how traditional instruments can be made network-enabled in a unified fashion. We describe some computer engineering experiments that have been realized by means of a lab-in-a-box approach and we introduce a framework that allows not only for mutual exclusive lab access, but also for advance reservation of experiments, conducting the experiments and a collaborative data interpretation and discussion of the experimental results.

3. Network Interface for Instruments

In order to implement remotely controllable experiments we need network-enabled signal generators and measurement devices. The signal sources are used to excite the device under test (DUT) and its reactions are detected by the measurement devices. Of course, also the DUT setup must be network-enabled.

Unfortunately, most of the traditional laboratory in-

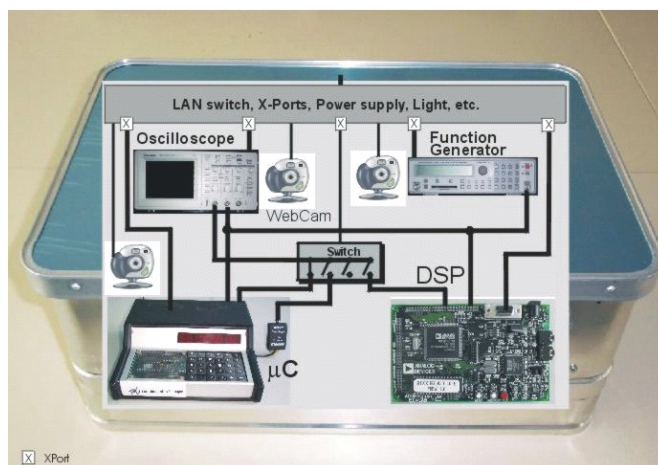


Figure 3. A remote lab-in-a-box for embedded applications.

struments like function generators, oscilloscopes or digital multi-meters are not equipped with a network interface. Instead, usually serial interfaces (RS232) are provided. Therefore, we need a device that can act as a bridge between these two interface standards.

Lantronix's XPort [11] provides an embedded ethernet device server that can easily adapt any electronic device with serial capability. Beside the bridging functionality there is also a built-in webserver. XPort uses a 16 Bit CPU which is clocked at 48 MHz (Figure 1). It is equipped with 256 KByte RAM and 512 KByte flash memory. 384 KByte remain for user data and Java applets while the rest is used by the XPort's operating system and the server applications (bridge and webserver). By means of a device installer or an DHCP-protocol the IP-address can be configured. As soon as the IP-address is assigned other configurations can be done by a web-frontend. Java applets and html pages are uploaded via the TFTP-utility.

By means of instrument-specific applets we can provide internet access to the instrument that is connected to the XPort. Users only need a web-browser and the URL of the Java applet. Figure 2 shows a screenshot of a remote control applet (RCA) for the Tektronix TDS 210 oscilloscope. This applet can easily be adapted to other 2-channel oscilloscopes by changing the control word sentences associated with the generic parameter functions (e.g. how to set channel amplification or set time base). In this way, Java applets for common laboratory instruments can be used as templates to create RCAs for instruments of other producers.

Beside the bridging and webserver capability an XPort also provides three parallel output signals that can be used to switch between different DUTs or experimental setups. By means of a four-way analog multiplexer that needs two

of those signals as control lines, we can choose between up to four configurations that reside in one Lab-in-a-box to be presented in the next section. The third output can be used as a reset signal. Of course, we can arbitrarily scale this switching possibilities by additional XPorts.

One may argue that the XPorts' functions could also be implemented by one powerful PC that is equipped with multiple serial interfaces and that serves many instruments. Even though one can proceed in this way the use of multiple XPorts has several advantages: Due to the fact that they use a flash memory, XPorts can be switched on together with the associated instrument. Shortly after power-on they provide the desired network interface to the instrument. Another advantage is the lower power consumption of the XPorts and the better scaling of the webserver throughput. With respect to energy saving it is useful to switch on remotely controlled experiments only when they are actually needed. This is also much simpler with a dedicated XPort server.

4. A Remote Lab-in-the-Box for Embedded Applications

Although our concepts for remotely controllable experiments are suitable for laboratory courses in a wide range of engineering or natural sciences, we are, due to our background, especially interested in their realization in the domain of computer engineering. In a first step we built a lab-in-a-box, the structure of which is shown in Fig. 3, for use in our laboratory course on applied microprocessors, e.g. microcontrollers (μC) and digital signal processors (DSP).

The box contains two microprocessor systems, namely a microcontroller system with a small keyboard and a small LED display (shown bottom left) and a DSP with a stereo Codec to process audio input signals (bottom right). Output

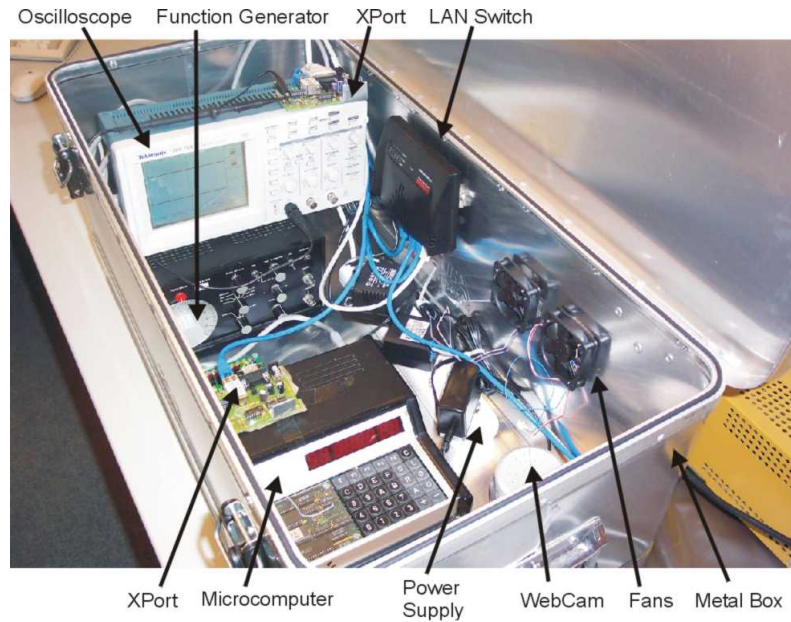


Figure 4. Test installation of the lab-in-a-box.

signals of both systems can be watched on the screen of a digital oscilloscope. A switching unit allows to choose one of four selectable output signals of the μC or DSP. Analog or digital input signals to both microprocessor systems can be delivered by a function generator.

The two instruments, i.e. the oscilloscope and the function generator, as well as the two microprocessor units are connected to the internet by means of the above described XPorts (symbolized by an X in Fig. 3) via a multiport LAN switch. In addition, one of those XPorts is used to operate the above mentioned switching unit, which is a 4-to-1 analog multiplexer, by two of its parallel output signals.

Three webcams, which are directly connected to the internet, allow to directly observe the displays of the two instruments as well as the microcontroller system. This means that the student has the impression of a real experiment, and has the assurance that the web frontends of the instruments and microprocessor units are not merely simulating the experiment. A small lamp inside the box provides enough light for the webcams to show the displays and their environment, so that the user gets a real impression of the experimental setup. This lamp and all the other devices in the box get their electrical energy by a power supply which is controlled by an internet-connected switch. So, to perform an experiment one first has to switch on the light and all the necessary equipment and to switch them off when work is done — just as in a real laboratory. A test installation of the box is depicted in Fig. 4.

The lab-in-a-box allows to do lots of experiments in the

domain of embedded applications. Some of them can be done with the same setup, only by re-adjusting the 4-to-1 switch. Of course, for greater changes between experiments, the lab's tutor has to do some adaptation job, e.g. to exchange some devices, or to connect specific hardware modules to the input/output signals of the microprocessor systems. The microprocessor system consists of a self-developed microcomputer, integrating an 8-bit processor, a keyboard and a seven-segment display. As peripherals it offers two parallel ports, two serial interfaces and three 16-bit timers. A small operating system (monitor program) allows or eases to store programs or data and to control the display, the keyboard and the peripherals. The RS232 Interface is used for communication with a PC or the XPort described above.

Fig. 5 shows in detail how an “augmented” application can be executed with the μC system. Here, a small radio controlled clock unit (DCF-77) is connected to the microprocessor to deliver the exact time of day and date. Students can watch the signals delivered by the clock module on the screen of their PC by means of the webcam. There they see the analog signal received by the antenna of the clock module and the generated digital signal (as shown at the bottom of Figure 5). Furthermore, they can acquire and record this signal on harddisk by means of the applet controlling the oscilloscope.

Let us assume that students have to write a program for the μC to analyze the clock signal and to show time or date, resp., on the display of the μC system. To do this they first

have to work through the course text, which describes the basics for the experiment. Then they can use — locally and perhaps off-line — an integrated software development tool (IDE) and a simulator. Figure 5 shows that the graphical interface of this simulator resembles very much the appearance of the used real microcomputer, so the student can easily operate both. When they believe their program is fault-free and fully operational they submit it to the μC in the lab-in-a-box via internet. The software (a Java applet) for accessing the μC system has also almost the same look as the physical system itself (see left side of Fig. 5). Besides upload or download of programs and data, it allows to remotely control every component of the μC system, including the display and keyboard: all keyboard inputs by the user are transferred to the remote system and all outputs to the display are transmitted back to the controlling applet. When the user operates the clear key C of the graphical interface a special signal is sent to the real microcomputer which is thereby reset into its initial state. Furthermore, the user can watch the display by a webcam — thus realizing that what he sees is not just the result of a simulation. So, when the job of analyzing the RC clock signal is done quite well, the "real" microcomputer as well as the webcam and the Java applet on the PC screen will show the same exact European time, regardless where on earth the user lives.

Fig. 6 shows how experiments using the DSP can be done. One of the most popular and surely easiest experiments in digital signal processing is to write a program for FIR filtering (Finite Impulse Response). The DSP usually has no control functions to perform. Therefore, the applet to control it and its graphical user interface is rather simple: it only allows to reset the DSP, to upload programs to the DSP and start them, to request an interrupt or to download memory regions from the DSP for further inspection. All actions are monitored in one small window shown in the upper right corner of Fig. 6. To support the students in writing their DSP programs they can use a professional software development tool, including again a simulator.

Fig. 6 shows the execution and evaluation of the FIR program running on the DSP. In the upper left corner you can see the graphical user interface of the applet which controls the function generator. Here one can choose a certain form of output signals with its specific parameters (like frequency, impulse width, etc.). The determined input signal and the resulting (filtered) output signal of the DSP can be watched by the webcam attached to the oscilloscope (lower right corner) or on the virtual screen of the applet controlling the oscilloscope (lower left corner). Of course, the latter signal may also be further processed or stored on hard-disk. At the top of the right side you can see the window to select and upload programs from the client PC to the DSP.

As a last example Fig. 6 shows a more fundamental experiment in computer engineering. Its aim is to acquaint

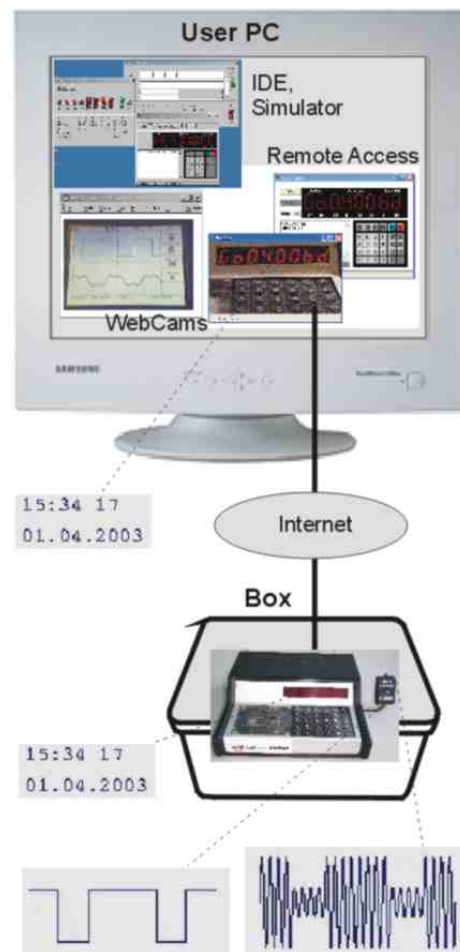


Figure 5. Graphical user interface for a special microcontroller application

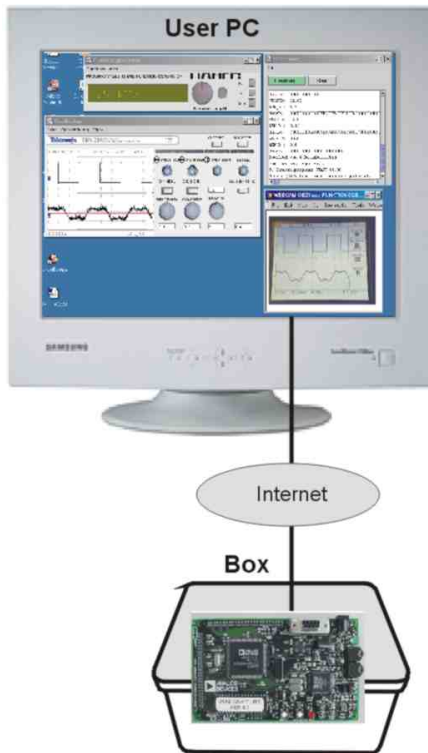


Figure 6. A DSP experiment setup.

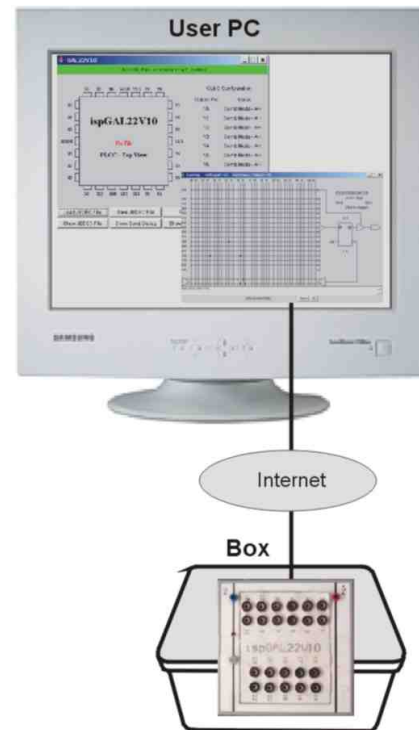


Figure 7. An experiment setup for programming GALs.

the students with the programming and practical use of Programmable Logical Devices (PLD), especially the so-called GALs (Generic Array Logic). The deployed device is an in-system programmable (isp) GAL22V10. It consists of a circuit with up to 22 inputs and 10 outputs that can be used either in combinatorial or registered mode. The students can program this GAL chip by means of a Java application that allows a graphical programming of a matrix with approximately 6000 fuses. As shown in Figure 7 the students select the output pin that should be programmed (left upper corner of 7) and the GAL-editor displays the corresponding part of the fusemap that now can be easily programmed by clicking the intersections between input (or feedback) wires and an AND-term of the output function (right lower corner). After programming of all the output lines the final fusemap can be uploaded via the Internet to the remote lab-in-a-box where it will be used to configure the GAL-chip. Some of its outputs are connected to the inputs of an oscilloscope and can thus be observed by means of a webcam or the virtual oscilloscope shown on the screen of the user PC.

5. Collaborative Web Platform

An indispensable component of a remote lab is the web platform that supports both mutual exclusive access to the

lab-in-a-box and collaboration of student working groups. In the *GridLabs* project supported by the German Ministry of Research and Education (BMBF) we developed a *Collaborative Remote Laboratory* (CRL) server that integrates collaboration support and remote access to real laboratory equipment. To provide an intuitive software interface we use the metaphor of a virtual building that consists of virtual rooms augmented with real experimental equipment [7]. As in real life these augmented virtual reality rooms are connected by floors that can create hierarchical structures. Virtual rooms can only be entered if one holds a valid virtual key. A remote lab for one or more related experiments comprises several virtual rooms for different purposes (see Figure 8):

- main entrance hall,
- administration room,
- communication and support room,
- materials room,
- show room, and
- one or more lab rooms.

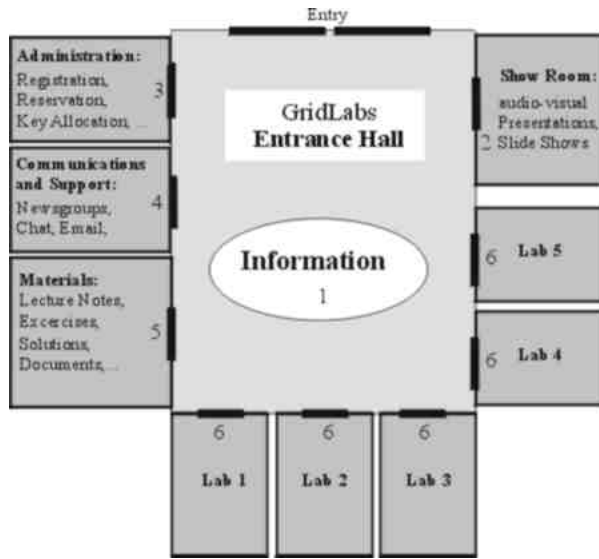


Figure 8. Structure of the CRL web platform.

All the rooms provide facilities for asynchronous and synchronous communication between the lab's participants (threaded email and chat respectively). Moreover, in the communication and support room students can meet tutors to get more qualified help on specific problems. Also in all rooms of the CRL, so called *pages* of shared data can be stored and accessed by everybody who holds a virtual key for that room. Two kind of pages can be distinguished: web-editable content pages (WIKIs) and binary pages (files) for the exchange of data by means of external applications. In contrast to [7] CRL allows not only access to pure virtual rooms but CRL is an extension of CURE that simultaneously supports the access to real remotely controllable devices.

Every student gets a key for the main entrance hall. In the materials room the students find detailed descriptions of the experiments, assignments and user manuals of this laboratory's instruments and DUTs. In the showroom slide shows and tutorial movies explain the essential procedures for remote experimentation. Students can also create private (sub)rooms for storing their individual results of laboratory sessions, which are conducted in one of the lab rooms. In order to collaborate with other students they can make copies of their personal keys and pass copies of it to their colleagues. In this way, the room becomes a (virtual) group room.

In contrast to the rooms described so far lab rooms can only be entered for a limited time period. During this time a mutual exclusive access must be guaranteed. To accomplish this we generate a bunch of virtual keys that are only valid for a given time slot. The time slots are fixed to a given time

period (e.g. 15 minutes each) which can be easily changed by the guidelines of the laboratory's organizer. In the administration room a key rack provides all time-limited keys that are still available. If there are n equivalent lab rooms (actually each one requires a lab-in-a-box) there will also be n keys for each time slot. Thus the operational availability can be easily adapted to the needs by adding new lab-in-a-box modules and corresponding lab rooms.

By taking keys from the rack (in a FCFS-manner) the students reserve a lab room for a specific time slot. Because there is only one key per lab room and time slot mutual exclusive access is guaranteed. The key holder can only act as an experimenter during the corresponding time slot of the key. He can also create and pass observer keys which allow the access of the lab room as an observer but not permit the control of the experimental setup. The results of an experiment will be stored in pages of a group room where they can be discussed within the working group and prepared for submission to their supervisor.

6. Conclusions

We have presented a lab-in-a-box, a flexible infrastructure for setting up experiments to be performed over the internet. The lab-in-a-box is operational and will be used in the next laboratory course, running in the coming fall semester. The experiments there concentrate on embedded systems, i.e. microcontrollers and digital signal processors, however the applicability is much wider. The initial effort to program control applets for all devices and instruments looks much higher than it really was, because after the first applet, we could re-use much of the code, as many instruments are controlled in similar ways, by sending control strings and receiving result strings.

In the future, we would like to incorporate our remote lab in the GridLabs project, which already was envisioned in [3]. In this effort, several remote labs are to be shared among several virtual organisations in order to be able to provide a much larger spectrum of instruments and experiments.

References

- [1] A. Bagnasco, G. Parodi, D. Ponta, and A. Scapolla. A modular and extensible remote electronic laboratory. *International Journal on Online Engineering (iJOE)*, 1(1), 2005.
- [2] H. Bähring, J. Keller, and W. Schiffmann. A combined virtual and remotely accessible microprocessor laboratory. In *Proc. 11th Workshop on Computer Architecture Education (WCAE 2004)*, pages 136–141, June 2004. <http://www4.ncsu.edu/~efg/wcae/2004/>.
- [3] H. Bähring, J. Keller, and W. Schiffmann. Web-based support for computer engineering. In *Proc. 5th Conference*

Virtual University: VU 2005, June 2005. <http://pv.fernuni-hagen.de/docs/keller-vu05.pdf>.

- [4] M. Cooper. Remote laboratories in teaching and learning – issues impinging on widespread adoption in science and engineering education. *International Journal on Online Engineering (iJOE)*, 1(1), 2005.
- [5] S. Das, L. N. Sharma, and A. K. Gogoi. Remote communication engineering experiments through internet. *International Journal on Online Engineering (iJOE)*, 2(1), 2006.
- [6] I. Gustavsson, J. Zackrisson, L. H. A. Akesson, I. Claesson, and T. Lagö. Remote operation and control of traditional laboratory equipment. *International Journal on Online Engineering (iJOE)*, 2(1), 2006.
- [7] J. Haake, T. Schümmer, A. Haake, M. Bourimi, and B. Landgraf. Supporting flexible collaborative distance learning in the cure platform. In *Proceedings of the Hawaii International Conference on System Sciences, HICS '04*. IEEE press, 2004.
- [8] V. Harvard, J. del Alamo, V. Choudhary, K. deLong, J. Hardison, S. Lerman, J. Northridge, D. Talaverna, C. Varadharajan, S. Wang, K. Yehia, and D. Zych. ilab: A scalable architecture for sharing online experiments. In *International Conference on Engineering Education*, Gainesville, 2004.
- [9] D. Kennepohl, J. Baran, M. Connors, K. Quigley, and R. Currie. Remote access to instrumental analysis for distance education in science. *The International Review of Research in Open and Distance Learning*, 6(3), 2005.
- [10] R. Langmann. Web-based remote control by liveconnect. *International Journal on Online Engineering (iJOE)*, 1(1), 2005.
- [11] Lantronix. Xport manuals. <http://www.lantronix.com/device-networking/embedded-device-servers/xport.html>.
- [12] D. Miller and J. Ferreira. Online labs and the marvel experience. *International Journal on Online Engineering (iJOE)*, 1(1), 2005.
- [13] J. Murphy, J. Grout, J. Walsh, and T. Shea. Local and remote laboratory user experimentation access using digital programmable logic. *International Journal on Online Engineering (iJOE)*, 1(1), 2005.
- [14] G. Olson, D. Atkins, R. Clauer, T. Finholt, F. Jahanian, T. Killeen, A. Prakash, and T. Weymouth. Upper atmospheric research collaboratory. *ACM Interactions*, 5(3):48–55, 1998.
- [15] S. Sivakumar and W. Robertson. Development of an effective remote laboratory for online internetworking education. In *Proceedings of the 37th Hawaii International Conference on System Sciences*, 2004.