Online FPGA laboratory for interactive digital design

Authors:

Joar Rusten, UniK – University Graduate Center, Norwegian University of Science and Technology, N-2027 Kjeller, Norway, joarolai@unik.no
Sigbjørn Kolberg, UniK – University Graduate Center, Norwegian University of Science and Technology, N-2027 Kjeller, Norway, kolberg@unik.no

Abstract *— Recent advances in digital electronics technology have had a huge impact on the learning environment for students enrolled in digital design courses. Up u ntil now, Field Programmable Gate Arrays (FPGA) have been utilized on small circuit -boards that incorporate buttons and small interactive features. The increased availability of computer -hosted boards with large FPGA devices enables large numbers of studen ts to utilize the power of configuring real devices through a remote laboratory. Remote laboratories in other disciplines have been subject to development for several years and have proven effective in making hardware resources available to more students. Our laboratory is based upon Web Services and Java technology to maximize the flexibility of hosting applications and give access to interfaces at different levels. A higher level Remote Method Invocation (RMI) interface for JAVA is developed to take care of the user interaction with the system. A learning module called One Instruction Computer Laboratory (OICLab) presents a user interface comparable to those of common digital simulators. A specially designed simulator allows users to compare simulation an d physical execution of real time execution of the OIC. A further development of learning modules in digital design will aid students in their understanding of how high level languages are transformed into logical operations.*

Index Terms *— Field programma ble gate array, FPGA, Remote Laboratory, Web Services, digital design*

INTRODUCTION

In universities around the world, digital design and computer architecture are important ingredients in electrical engineering curricula. Courses offered in these disciplines often utilize simulators and small embedded, electronic boards that offer hands-on testing with the designs created. As the discipline is often highly focused on design of applications, laboratories are commonly used to enhance training and reinforce subject matter.

Electronics laboratories based on experiments performed directly on laboratory hardware can in some situations offer too little flexibility for both students and teachers. This is usually caused by limited hardware capacity and the trade-off with equipment quality. By introducing remote laboratories based on international web standards, limited amounts of high-quality laboratory hardware can be utilized by several students at the same time, minimizing the demand for dedicated laboratory facilities. Several institutions engaged in electronic disciplines have had good experience on the usage of remote laboratories [1]. In such a scenario, students engaged in laboratory exercises will be totally independent of both location and timetable, and are therefore free to perform unlimited amount of experiments wherever and whenever they want. Remote laboratories which implement standardized Web Services interfaces can easily be traded between educational institutions, enabling rapid expansion of laboratory exercises and also introduce new relations between participants, both students and teachers, as well as the developers of these remote laboratories [2]. When using remote laboratories, teachers can easily integrate and demonstrate laboratory exercises directly in lectures performed in traditional lecture rooms or even teach these exercises via remote lectures to a global audience.

The creation of a special computational unit called “One Instruction Computer” (OIC) has been developed as a learning module for the laboratory, OICLab [3]. As the OIC requires additional infrastructure to be useful, a framework consisting of a compiler, a simulator and an emulator has been created to support the dissemination of this particular application. The project output has important educational benefits related to microprocessor design. These properties can be utilized in courses that are taught in both pure digital design courses and computer architecture classes.

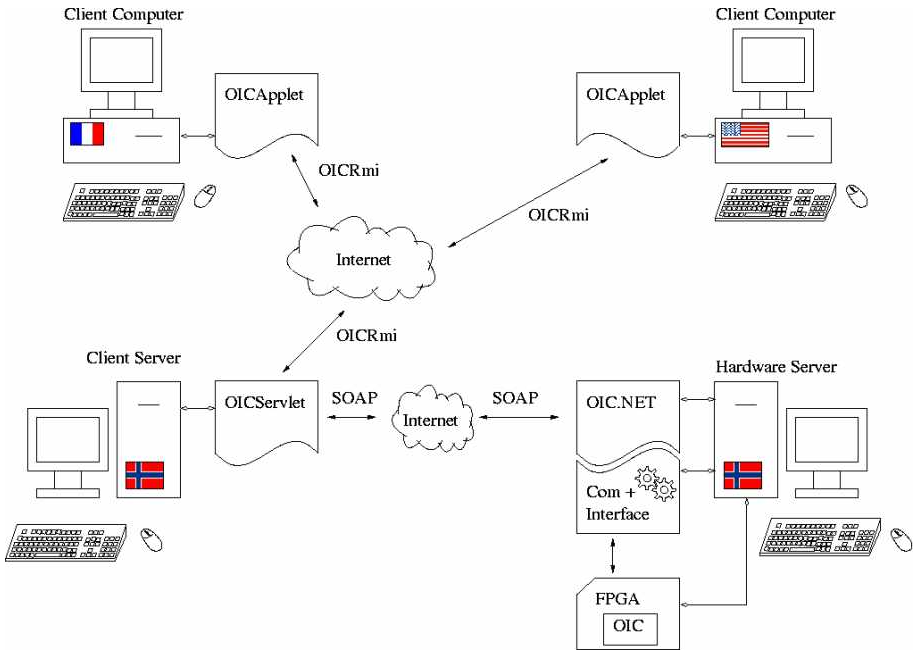
OIC – THE LEARNING MODULE

Our proposed learning module is based on a special microprocessor that is implemented on an FPGA. The microprocessor has only one instruction; and is therefore named OIC. The basic functionality of the instruction is subtraction and is called SABON- "Subtract And Branch On Negative". Because of the simple instruction set, this application is especially well suited for learning and teaching of basic microprocessor design.

OIC SYSTEM

The OICLab laboratory is composed of three main components, one or several Client computers, the Client server, and the Hardware server. See Figure 1. A user accesses the laboratory from a Client computer through a special Java Applet called OICApplet that is implemented in a standardized HTML web page. This application is linked to the Client server through a Java servlet called OICServlet that holds several system specific software-developing tools for the OIC processor. One of these tools is further connected to the Hardware server, indirectly connecting the user to the laboratory hardware, and the OIC processor installed there.

FIGURE. 1
STRUCTURE OF OICLAB



OICApplet is the user interface of OICLab, which is incorporated in a web page to optimize flexibility for remote users. The laboratory is organized as a workspace similar to that offered by Microsoft® Windows®, where programs can be initiated from a special menu and executed in separate windows that can be rearranged and manipulated by the user. OICApplet is currently offering the following programs: a file manager, a text editor with a built inn assembler, a simulator, and an emulator. Each user of OICLab is assigned a user account on the Client server, where a dedicated home directory is created for each user. This directory holds laboratory specific files created and generated by the user. These files can then be remotely manipulated from OICApplet by utilizing the file manager program. No programs located in OICApplet, including the file manager, is capable of manipulating or even access files located on the Client computer, thereby eliminating the possibility that OICApplet can harm the users local computer. OICApplet communicates with OICServlet located at the Client server via OICRmi, which is a system specific communication scheme based on HTTP, where remote methods located in OICServlet can be invoked from OICApplet with serializable objects as arguments.

The OIC processor is programmed with special OIC assembler programs that can be created and assembled with the help of the text editor found in OICApplet. Files edited by this text editor will reside in the user home directory on the Client

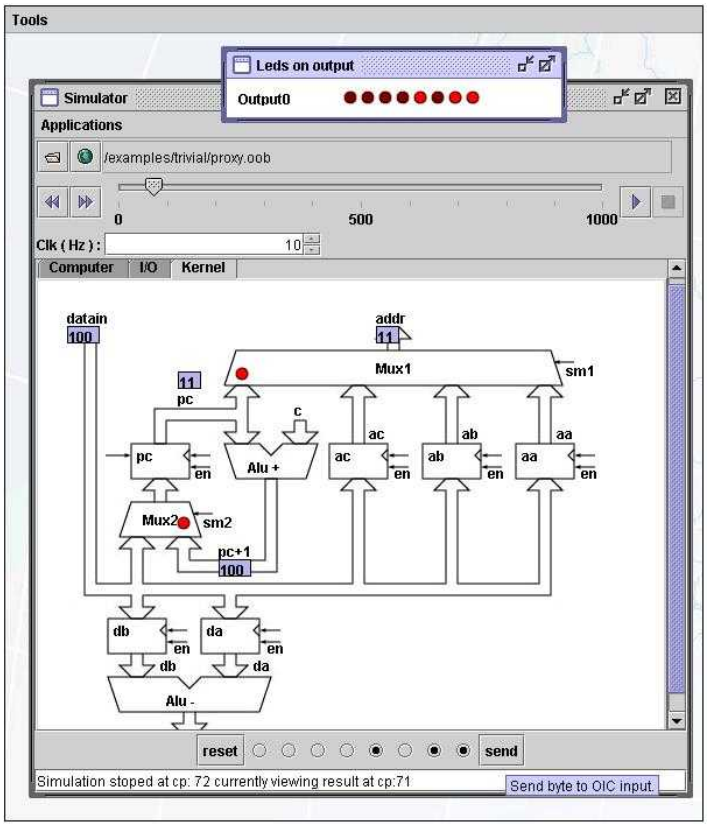
server, where they can be retrieved and utilized by the OIC assembler, and other development tools. Information gathered during assembling is sent back to OICApplet and displayed in the text editor where the source files are being edited. Executable files generated by the assembler will be saved in the user home directory where other development tools can access them.

The Hardware server contains an interface for the FPGA developing board and the OIC processor. This interface is composed of a native Microsoft® Windows® COM+ component implemented by a C# SOAP Web Service named OIC.NET, see Figure 1. OIC.NET offers RMI for altering the behavior of the OIC processor, e.g. methods for altering the content of the OIC memory and for asserting external control signals to the processor, for example reset signals. These methods are then invoked from OICServlet, indirectly connecting the Client computer to the Hardware server and the OIC processor. With the proposed remote access architecture, we eliminate the need for several hardware instances by introducing a scheduling algorithm to access the hardware.

OIC SIMULATOR

Simulation is performed in two stages based on an executable file selected in the internal file manager of the simulator program. The first stage of the simulation is performed on the Client server in OICServlet where the content of the selected executable file is organized in a virtual memory container, which is shipped back to the Client computer. The second stage of the simulation is performed in the simulator program on the Client computer, which visualizes the behavior of the processor according to content of the virtual memory container. Different parts of the internal structure of the processor can be selected in the simulator, and further examined in either continuous or manually stepped execution mode. Interactive features found in the simulator program enable users to manually change the data representation of the internal data signals and buses in the OIC processor. Additional information regarding the behavior and structure of the processor and its internal components can also be accessed interactively from the simulator program. The OIC processor is designed with several dedicated input and output ports, which can be accessed by OIC programs. This functionality is also included in the simulator program to enable the user to interact with these ports through simulations. See Figure 2.

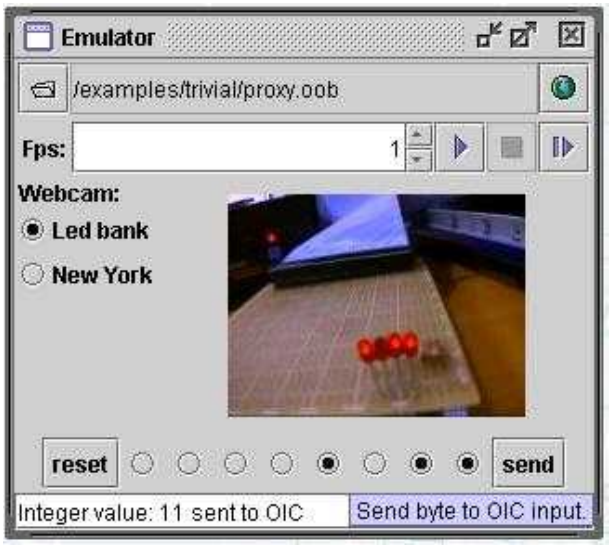
FIGURE. 2
SIMULATOR PROGRAM



OIC EMULATOR

The emulator program is a visual and interactive link between the user and the hardware implementation of the OIC processor. Executable OIC programs located in the user home directory on the Client server are loaded on commands from the emulator program into the memory of the OIC processor via OIC.NET. A simple hardware application, which consists of four light emitting diodes, is connected to one of the output ports of the OIC processor. A standard web camera installed in the laboratory provides the user with real time visual information about the behavior of this hardware application, where images from the laboratory are displayed directly in the emulator program. See Figure 3.

FIGURE. 3
EMULATOR PROGRAM . FOUR LIGHT EMITTING DIODES CONNECTED TO THE OIC PROCESSOR



OIC PROCESSOR

The OIC processor implemented in this laboratory is based on a processor model used by the department of Computer Science and Engineering of Penn State University [4]. This processor is, in theory, capable of performing any mathematical and computational tasks known from other processors enriched with much bigger instruction sets. However, by choosing the correct instruction, all other instructions can, to some extent, be regarded as superfluous. This processor performs subtraction followed by a conditional branch based on the outcome of the subtraction. The SABON instruction is specified in the following way in OIC assembler language.

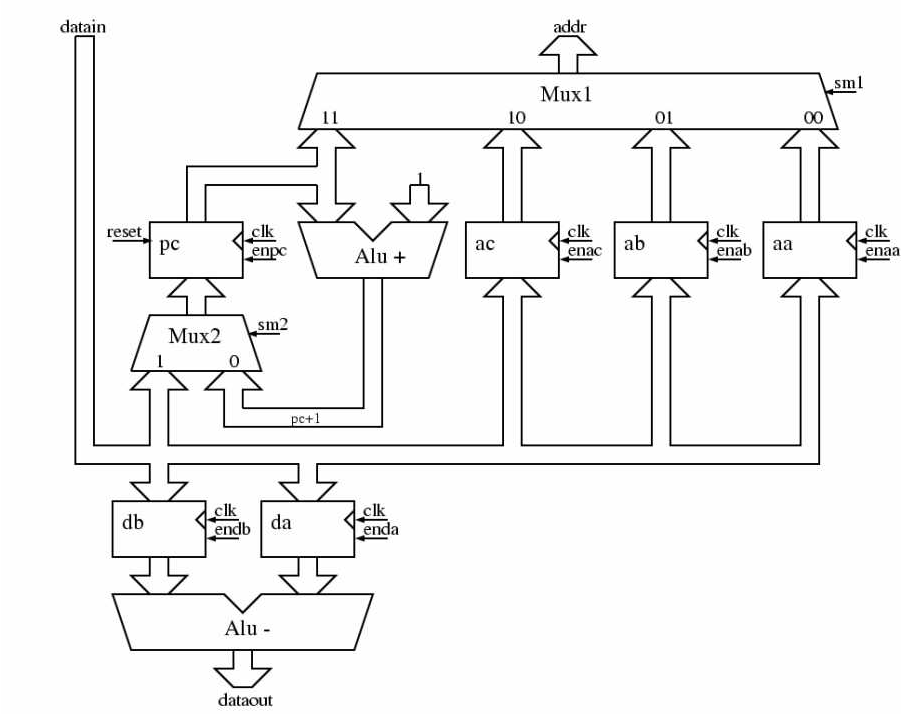
name: sabon a, b, c, branch ; memory(c) = memory(a) - memory(b)

The SABON instruction takes four arguments, a, b, c and branch, in addition to a name attribute. Comments can be located directly in the assembler code, separated from the executable code fragments by semicolons. The OIC processor implemented in this laboratory is only capable of performing immediate operations on memory locations specified in the instruction, since it lacks both working registers as well as register-access instructions. A SABON instruction is interpreted as explained in the comment found in the example program line above, where the branch argument must coincide with the name-attribute of the instruction to where the program should branch if subtraction give a negative result. If the result of the subtraction is positive, the processor will advance to the next instruction in the program, discharging the branch argument. This feature can be utilized to perform complex mathematical and computational tasks by organizing several SABON instructions in macros, which in turn can be used in other macros to build more and more complex tasks.

OIC PROCESSOR ARCHITECTURE

The OIC processor kernel implements a relatively simple architecture, and makes use of well known electronic building blocks found in FPGA circuits, i.e. registers, multiplexers, and simple arithmetical units. Figure 4 shows the internal structure of the processor kernel. The kernel is connected to the memory via an address buss, addr, and two data buses, datain and dataout. The three first arguments in a SABON instruction are sequentially loaded from the memory into registers: aa, ab and ac, respectively. The content of these register are then utilized by the kernel when accessing data variables found on these memory locations. The contents of the data variables found on addresses aa and ab are loaded into registers da and db, respectively, where they are used by Alu- to perform the subtraction part of the SABON instruction. The result from the subtraction is presented on the dataout bus and stored in the memory location specified by register ac. After the subtraction part is completed, the kernel will load the last argument, the branch argument, into the kernel, and further store it in the program counter register, pc, if the subtraction gave a negative result. If the subtraction gave a positive result, the kernel will update the program counter with the address of the next instruction in the program, and discharge the branch argument. More detailed information about the OIC processor, the surrounding laboratory, and its developing tools, can be found in [5].

FIGURE. 4
THE KERNEL OF THE OIC PROCESSOR



OIC LABORATORY USAGE

Student using this laboratory will gain insight in basic microprocessor operation through a simple microprocessor model that is based on the same principles as more complex sequential microprocessors. Because this processor have limited functionality, students are forced to show inventiveness and skills to be able to design effective microcode.

DISCUSSION AND SUMMARY

As FPGA technology advances, the possibilities to build more sophisticated remote laboratories are enhanced. Earlier, laboratories using FPGA devices have been accessible for only hands-on experimentation. The devices have also been relatively expensive. The cost benefits of sharing such hardware resources through the Internet are large and the laboratories are made accessible for everyone with an Internet connection.

The learning benefit from traditional hands-on laboratory work using a soldering iron to build logical circuits is in many institutions considered invaluable. However, with larger designs, this approach is impractical. We argue that the remote laboratories provide as much knowledge of the digital design's behavior as the traditional hands-on laboratories. The presentation techniques and framework that can be built around remote laboratories give students a more rigid learning environment. Although the "presence" feeling of holding the physical circuit cannot be replaced, the educational benefits obtained trough graphical means may be just as strong.

ACKNOWLEDGEMENT

This work was supported by a grant (eMerge project [6]) from the European Community within the framework of the Socrates/Minerva program. We also acknowledge the previous support from the Nordic Council of Ministers through the Nordunet2 program.

REFERENCES

[1] Fjeldly, T, A, and Shur, M, S, *LAB on the WEB, Running Real Electronics Experiments via the Internet* , John Wiley & Sons, New York, NY 2003.

[2] Kolberg, S, and Fjeldly, T, A, "Internet Laboratory with Web Services Accessibility", *Advances in Technology -Based Education: Towards a Knowledge -Based Society, Proc. 2nd Int. Conf. on Multimedia ICTs in Education (m -ICTE2003)* , Badajoz, Spain, Dec. 2003), A. M. Vilas, J. A. M. Gonzalez, and J. M. Gonzalez, Editors, Vol. 3, pp. 1700-1704.

[3] OICLab, Department of Electronics and Photonics, University Graduate Center at Kjeller, URL: http://www.lab-on-web.com/oic/

[4] Penn State University, Department of Computer Science and Engineering, Computer System Architecture, The One Instruction Set Computer, URL: http://www.cse.psu.edu/~cg331/

[5] Rusten, J, O, "OIC , One Instruction Computer, Subtract And Branch On Negative", Master Thesis in Electronics and Telecommunications, University Graduate Center at Kjeller / Norwegian University of Science and Technology, 2004

[6] eMerge: a European Community project within the Socrates/Minerva program, see J. MARTINEZ, et al., "eMerge, a European Educational Network for Dissemination of Online Laboratory Experiments", *Int. Conf. on Engineering Education (ICEE 2003)* , Valencia, Spain, July 2003, paper no. 3171.