A Web-Based Laboratory for Digital Signal Processing

doi:10.3991/ijoe.v6s1.1382

P. Buschiazzo, D. Leoncini, R. Zunino and A.M. Scapolla
University of Genoa, Genoa, Italy

***Abstract*—This paper presents a web-based laboratory to practice with embedded systems and in particular with Digital Signal Processing. The laboratory has been developed using a service oriented architecture to guarantee flexibility and interoperability with different electronic devices. Moreover it provides end users with a web based interface directly accessible through Internet and employees a wide range of modern Web 2.0 technologies. This approach allows creating complex web applications avoiding to install any plug-in on users' browsers. The paper presents how the remote laboratory can be used in an e-learning scenario and details how it can be integrated in feasible educational path based on common learning platforms such as Moodle.**

***Index Terms*—Digital Signal Processing (DSP), Remote Laboratory, Service Oriented Architectures.**

I. INTRODUCTION

The rapid evolution of Web technologies, applied to sharing lab equipments and educational contents across multiple institutions, has led to a significant increase of remote laboratories. They cover different fields of science and engineering education and allow end users (students and trainees) to access experimental setups remotely via the Internet. This paper presents a remote laboratory focused on Digital Signal Processing (DSP) which has been implemented at the Department of Biophysical and Electronics Engineering (DIBE) of the University of Genoa [1, 2]. The target of this laboratory is not new in the context of remote laboratories (see Section 5 for an outline of existing laboratories on DSP), but it offers an innovative graphical interface which supports a wide range of experiments. The lab targets students' practice on writing and testing programs on the DSP board for speech and image processing, data encryption and compression, automotive control and spectral analysis. A web-based software development environment allows students creating and building C applications that can be remotely executed and monitored.

Students are able to access the laboratory through a web interface and concurrent accesses to the same device are automatically managed by a proper scheduling module.

The laboratory has been developed using AJAX (Asynchronous Javascript and XML) technology and a service oriented approach. This optimizes the performance and facilitates the extension of the experimental setup. At the moment the test bed consists of the Analog Devices BF533 Ez-KiT Lite development board [3] and provides a cost effective support to courses on digital embedded systems. Since the web based interface is designed as a general purpose development environment, it can be easily adapted to support different DSP architectures. The paper is structured as follows: Section 2 presents the laboratory architecture; Section 3 focuses on the software developing environment and explains how AJAX technologies enhance the user experience and provide interactive access to the remote workbench; Section 4 describes how the remote laboratory system can be integrated in a course delivered by the Moodle LMS [4]; Section 5 reports related works. Conclusions and future works end the paper.

II. LABORATORY ARCHITECTURE

The laboratory architecture (see Fig. 1) is organized in three layers: the resources, the server and the user interface. The resources are the real electronic devices and software packages (e.g. compilers) used to run the lab activities. The lab server is the software module which interacts with these resources and manages the incoming requests from end-users. Finally, the user interface allows clients to communicate with the lab server and to operate remotely. The DSP lab user interface is structured as a software development environment where users can develop and build C applications, and monitor their execution on the remote development board.

This section gives an overview of the whole system and details interactions and data exchange between the lab system components.

The resource layer contains instrumentation and third party software packages that compose the test bench. In particular, the DSP lab uses an Analog Devices BF533 Ez-KiT Lite development board and its software development environment, VisualDSP++ [5].

Since at this layer we can have heterogeneous resources using ad-hoc communication protocols and medium, the interaction between each device and the lab server is mediated by software applications, named controllers [6, 7]. A controller can be directly hosted on a device, which is fitted out with a complete operating system or it can be deployed on an intermediate node, such as a common personal computer, that is directly connected to the real devices. Each controller exposes the same general purpose web service interface to the lab server. It provides the methods to send data and commands to devices and to read data from them. As an example, in our laboratory the DSP development board is connected to a PC which runs a controller. The web service interface of this controller allows to send an executable application to the development board and to acquire the board status and the console output as shown in Fig. 1.

Thanks to the controller application, virtual devices, such as the DSP compiler, are managed exactly as the physical devices. The node where the software development kit has been installed runs a controller that mediates the requests to the software development package. The controller interface allows to send the source program to the compiler and to read the output of the compilation process and the code created by the build process.

The low level communication between controller and real devices is implemented by a plug-in approach. Each device or type of devices requires a specific plug-in. Plug-ins are dynamic-link libraries in Windows OS, or shared objects in Linux OS, composed by a simple set of C functions that interact with the devices using low level primitives and specific protocols. Since the controller application can manage several different plug-ins at the same time, different kind of devices can be made available through a single instance of a controller application. The plug-in system and the general purpose interface of the controller application allow to set up distributed test benches where heterogeneous instruments are connected to each other through a network (usually the local network of the laboratory). Moreover, experimental setups can be updated or reconfigured without changing the software of the lab server layer: the devices can be moved from a specific controller instance to another and new devices or software packages can be added with minimal effort.

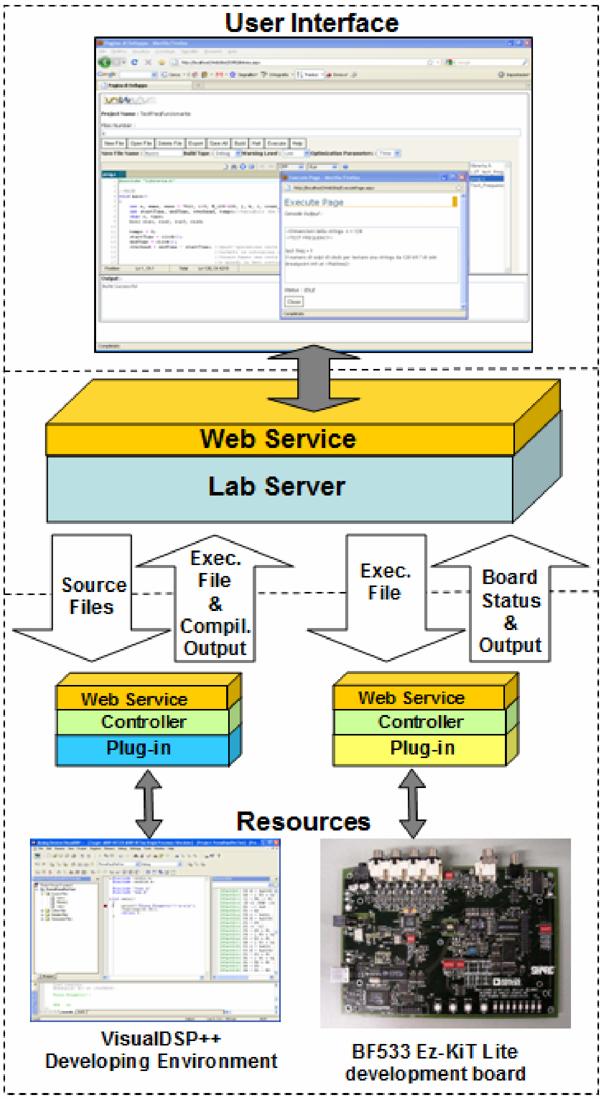


Figure 1. The remote laboratory architecture

The lab server is the core of the architecture, because it manages the communications with end-users and real resources. It receives commands to execute specific actions on the experimental setup by the user interfaces, it interacts with the controllers of the devices under test and forwards the data acquired from the test bench back to users. Each user interface contains a well defined set of actions, i.e. to configure instrumentation or to acquire data from a device. The interaction between the lab server and the user interface is mediated by a simple web service which allows submitting and monitoring actions.

The lab server maps every request to a specific action which contacts the controller in charge of the devices that are involved in the experiment and returns an output.

Another important feature of the lab server is the management of concurrent access to the same experimental setup by multiple users. The scheduling mechanism is strengthened by a proper locking system that tracks the devices in use. When an action is requested to the lab server the scheduler checks if the target devices are available. If they are already in use, the action is queued and it will be processed when the devices will become available. An estimation of the number of concurrent requests that we are going to face, let us consider that the scheduling mechanism should guarantee the students the possibility of working without perceiving significant delays.

III. THE LABORATORY USER INTERFACE

Accessing the lab test bench does not require any particular application or plug-in installed in the users' computers except for a web browser and an Internet connection. The lab user interface is organized as a software development environment where students can write C applications for the Blackfin 533 processor.

The laboratory can be used in two different ways: the user can create a new project and write source code from scratch or load an existing project from a repository. The former approach is intended to provide a blank environment where students or teachers can develop their own applications from scratch and test them on the development board. The latter one aims at supporting inexperienced students by providing examples or draft versions of the programs. Fig. 2 shows as an example a list of pre-loaded experiments inserted into the laboratory interface ranging between simplest codes and complex projects. Students can use these pre-loaded projects to make practice with theoretical subjects presented during the lectures. Moreover, these samples could be reused as starting points to develop other DSP applications . Obviously there is also the possibility to save a project and to continue the work later. The user interface is based on AJAX technology. AJAX minimizes the communication between end user and lab server and allows asking data to the lab server in background without interfering with user interface. Each page can have multiple components and each one can be asynchronously reloaded so that the user perceives the application to be faster and more reactive.

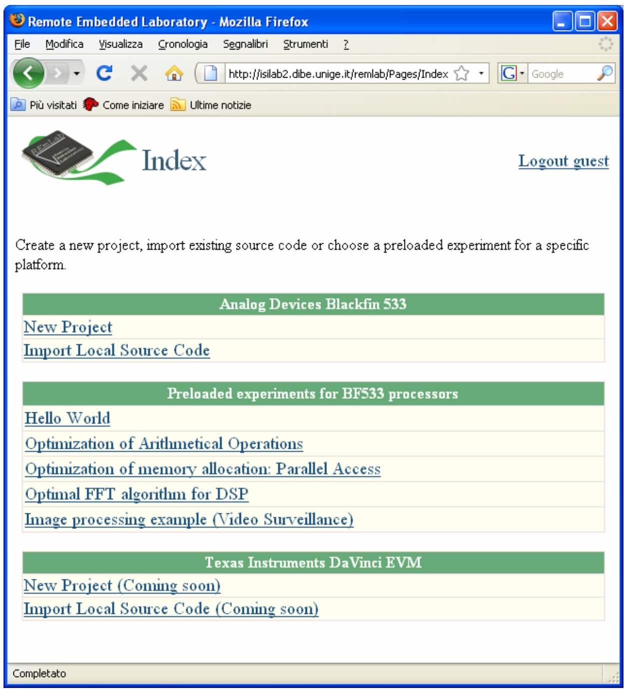
Up to now, the user interface is focused on a specific type of processor, the Blackfin 533, but the whole architecture can be easily adapted to support new platforms. Other kinds of development boards can be added to the current workbench writing low level controls, specific for the new boards, while the user interface shouldn't change significantly.

Fig. 3 reports a snapshot of this interface. The source code takes up the central part and it is displayed inside tabbed panels with a convenient syntax highlighting.

A toolbar offers the most common functionalities available in such environments, such as building and executing the code. The building process output is reported in the lower part of the page. At execution time the console output of the development board is reported in a dedicated popup window that is opened when the program execution starts. The console output is generated as plain text and it is constantly updated for the whole execution of the program.

Each page of the user interface is composed by modular components. Considering the page shown in Fig. 3, it is possible to identify many components: the toolbar, the tabbed source code editor, the compiler output console and the execution popup window. Other components can be easily added to provide new functionalities for end users. For example a component could show the screen of the oscilloscope that analyzes the DSP output data at execution time. DSP systems are used for image processing and, in this case, the process output is a matrix of pixel values. Even though displaying such kind of output as plain text is still possible (i.e. as a matrix of pixel values), it could be convenient to show the result using an image viewer component, as depicted in Fig. 4. This feature considerably improves the application usability and allows the creation of complex projects focused on image processing.

We have already planned to add new viewer components to the current user interface in order to show the internal memory and the registers of the DSP.


Remote Embedded Laboratory - Mozilla Firefox
File Modifica Visualizza Cronologia Segnalibri Strumenti ?
http://isilab2.dibe.unige.it/remlab/Pages/Index
Google
Più visitati Come iniziare Ultime notizie
Completato

Index

Logout guest

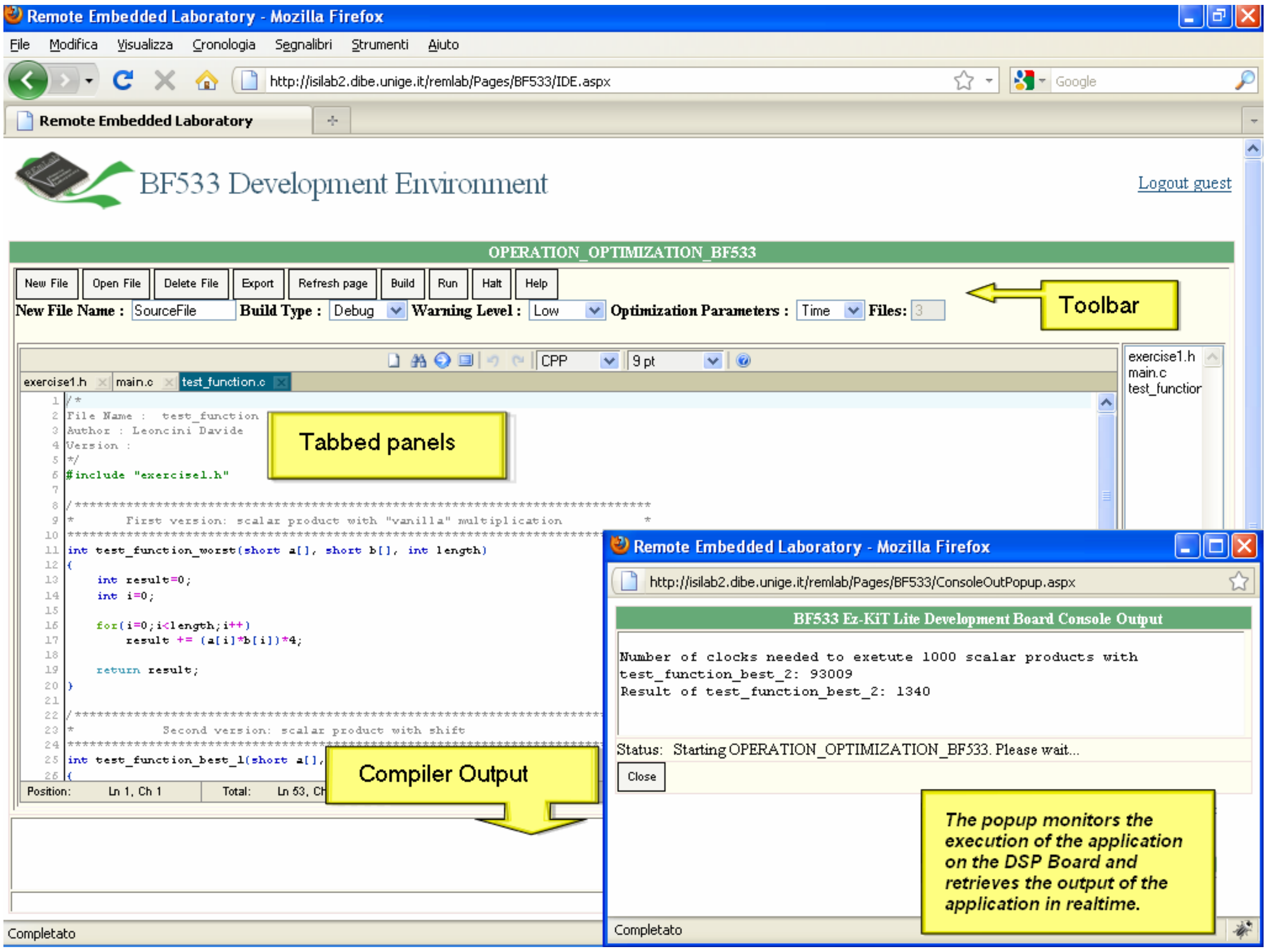
Create a new project, import existing source code or choose a preloaded experiment for a specific platform.

Analog Devices Blackfin 533
New Project
Import Local Source Code

Preloaded experiments for BF533 processors
Hello World
Optimization of Arithmetical Operations
Optimization of memory allocation: Parallel Access
Optimal FFT algorithm for DSP
Image processing example (Video Surveillance)

Texas Instruments DaVinci EVM
New Project (Coming soon)
Import Local Source Code (Coming soon)

Figure 2. List of preloaded experiments in the DSP lab


Remote Embedded Laboratory - Mozilla Firefox
File Modifica Visualizza Cronologia Segnalibri Strumenti Aiuto
http://isilab2.dibe.unige.it/remlab/Pages/BF533/IDE.aspx
Google
Remote Embedded Laboratory
Logout guest
New File
Open File
Delete File
Export
Refresh page
Build
Run
Halt
Help
Toolbar
CPP
9 pt
exercise1.h
main.c
test_function.c
Tabbed panels
Position: Ln 1, Ch 1
Total: Ln 53, Ch
Compiler Output
Completato
Close
The popup monitors the execution of the application on the DSP Board and retrieves the output of the application in realtime.

BF533 Development Environment

OPERATION_OPTIMIZATION_BF533

New File Name : SourceFile Build Type : Debug Warning Level : Low Optimization Parameters : Time Files: 3

```
/*
File Name :  test_function
Author : Leoncini Davide
Version :
*/
#include "exercise1.h"

/*********************************************************************************
*        First version: scalar product with "vanilla" multiplication            *
*********************************************************************************
int test_function_worst(short a[], short b[], int length)
{
    int result=0;
    int i=0;

    for(i=0;i<length;i++)
        result += (a[i]*b[i])*4;

    return result;
}

/*********************************************************************************
*            Second version: scalar product with shift
*********************************************************************************
int test_function_best_1(short a[],
{
```

Remote Embedded Laboratory - Mozilla Firefox

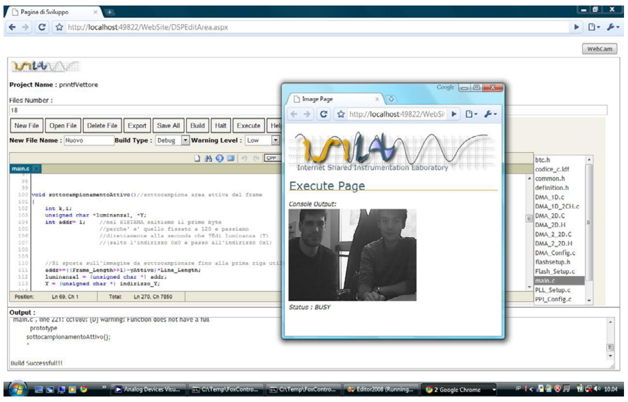
http://isilab2.dibe.unige.it/remlab/Pages/BF533/ConsoleOutPopup.aspx

BF533 Ez-KiT Lite Development Board Console Output

```
Number of clocks needed to exetute 1000 scalar products with
test_function_best_2: 93009
Result of test_function_best_2: 1340
```

Status: Starting OPERATION_OPTIMIZATION_BF533. Please wait...

Figure 3. Snapshot of the user interface


Pagina di Sviluppo
http://localhost:49822/WebSite/DSPEditArea.aspx
WebCam
Internet Shared Instrumentation Laboratory
New File
Open File
Delete File
Export
Save All
Build
Halt
Execute
Image Page
http://localhost:49822/WebSi

Project Name : printfVettore
Files Number : 18
New File Name : Nuovo Build Type : Debug Warning Level : Low

```
void sottocampionamentoAttivo()//sottocampiona area attiva del frame
{
    int k,i;
    unsigned char *luminanza1, *Y;
    int addr= 1;     //nel SISTEMA saltiamo il primo byte
                     //perche' e' quello fissato a 128 e passiamo
                     //direttamente alla seconda che ?Edi luminanza (Y)
                     //(salto l'indirizzo 0x0 e passo all'indirizzo 0x1)


    //Si sposta sull'immagine da sottocampionare fino alla prima riga uti
    addr+=((Frame_Length>>1)-yAttivo)*Line_Length;
    luminanza1 = (unsigned char *) addr;
    Y = (unsigned char *) indirizzo_Y;
```

Position: Ln 69, Ch 1 Total: Ln 270, Ch 7850

btc.h, codice_c.ldf, common.h, definition.h, DMA_1D.c, DMA_1D_2CH.c, DMA_2D.C, DMA_2D.H, DMA_2_2D.C, DMA_2_2D.H, DMA_Config.c, flashsetup.h, Flash_Setup.c, main.c, PLL_Setup.c, PPI_Config.c

Execute Page

Console Output:

Status : BUSY

Output :

```
main.c , line 221: cc1080: {D} warning: Function does not have a full
      prototype
    sottocampionamentoAttivo();
    ^

Build Successful!!!
```

Figure 4. The user interface can use an image viewer for displaying the output of the DSP

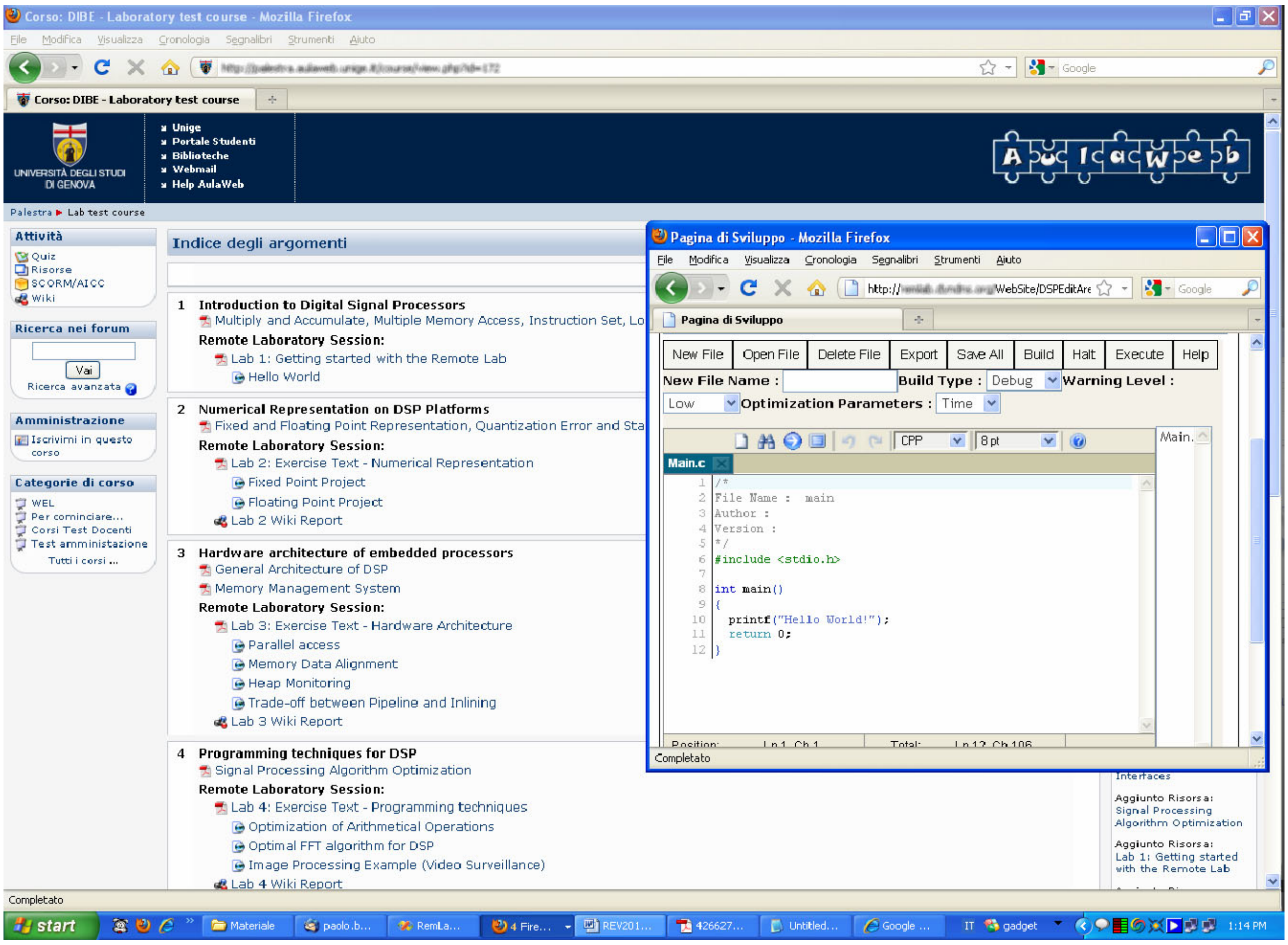
IV. DSP COURSES AND THE LABORATORY

The DSP laboratory can be accessed as a standalone web application but it can be also integrated in existing educational paths based on common learning platforms. This section analyzes how the remote laboratory can be used into a course on embedded electronic systems for the undergraduate students of the Electronic Engineering Faculty of the University of Genoa. The Aulaweb portal (http://www.aulaweb.unige.it/) is the course management system of the University of Genoa and it is based on the Moodle platform.

Fig. 5 sketches out a course which starts introducing fundamental concepts on the architecture of DSP: the Multiply and Accumulate Unit, the Multiple Memory Access, instruction set and numerical representations. More advanced arguments follow.

Each argument is organized into two parts: the first one contains theoretical contents, while the second one proposes experiments to be executed on the DSP lab. These experiments let the students practice on key concepts of the course, i.e. advantages and drawbacks of pipeline and inlining, unrolling operations and code optimization techniques, and allow them to consolidate theory with practical examples on real boards.

Remote laboratory sessions are introduced by documents that explain lab activities and provide guidelines for their execution. They must be read by students before starting the lab work. Links to the DSP lab allow starting operating remotely in the laboratory. Each link points to a specific activity and opens a proper interface in a new web browser window. Usually a startup source code is provided and students must complete, compile and test it online. At the end of the course, students are engaged in solving more complex programs, i.e. the implementation of FFT (Fast Fourier Transform) algorithms and code for video-surveillance which involves the acquisition and elaboration of video frames acquired from a camera connected to the development board. After each laboratory session, students must write out a final report that summarizes the work done and can be used by teachers for final evaluations.



Attività
- Quiz
- Risorse
- SCORM/AICC
- Wiki

Ricerca nei forum

Vai

Ricerca avanzata

Amministrazione
- Iscrivimi in questo corso

Categorie di corso
- WEL
- Per cominciare...
- Corsi Test Docenti
- Test amministazione

Tutti i corsi ...

Indice degli argomenti

1 Introduction to Digital Signal Processors
- Multiply and Accumulate, Multiple Memory Access, Instruction Set, Lo

Remote Laboratory Session:
- Lab 1: Getting started with the Remote Lab
  - Hello World

2 Numerical Representation on DSP Platforms
- Fixed and Floating Point Representation, Quantization Error and Sta

Remote Laboratory Session:
- Lab 2: Exercise Text - Numerical Representation
  - Fixed Point Project
  - Floating Point Project
- Lab 2 Wiki Report

3 Hardware architecture of embedded processors
- General Architecture of DSP
- Memory Management System

Remote Laboratory Session:
- Lab 3: Exercise Text - Hardware Architecture
  - Parallel access
  - Memory Data Alignment
  - Heap Monitoring
  - Trade-off between Pipeline and Inlining
- Lab 3 Wiki Report

4 Programming techniques for DSP
- Signal Processing Algorithm Optimization

Remote Laboratory Session:
- Lab 4: Exercise Text - Programming techniques
  - Optimization of Arithmetical Operations
  - Optimal FFT algorithm for DSP
  - Image Processing Example (Video Surveillance)
- Lab 4 Wiki Report

Interfaces

Aggiunto Risorsa: Signal Processing Algorithm Optimization

Aggiunto Risorsa: Lab 1: Getting started with the Remote Lab

http://[redacted]/WebSite/DSPEditAre

New File Name : Build Type : Debug Warning Level : Low Optimization Parameters : Time

CPP 8 pt

Main.c

```
/*
File Name :  main
Author :
Version :
*/
#include <stdio.h>

int main()
{
  printf("Hello World!");
  return 0;
}
```

Position: Ln 1, Ch 1 Total: Ln 12, Ch 106

Figure 5. Moodle course on embedded electronic systems which uses the remote laboratory

At the end of the course on Embedded Systems of the academic year 2009/2010 students attending the bachelor degree in Electronic Engineering were asked to compile a questionnaire to evaluate the usefulness of the remote laboratory. Students had to answer to four questions, grading the evaluation as poor, adequate or good:

1. Usefulness of the remote laboratory as educational tool.
2. Quality of the graphical interface.
3. Quality of the didactical material.
4. Usability of the remote laboratory

Fig. 6 summarizes the total satisfaction rate which is the ratio between responses evaluated as "good" and all the others. The figure points out a positive feedback from the interviewed students. In fact the remote laboratory is considered a valid tool in an educational context; it has a good graphical interface and the didactical material seems to be helpful for the end users. Even though the usability of the laboratory appears to be adequate, the last question obtained the lowest rank. This result could be justified because of the early stage of the development status and, in the authors' opinion, it must be considered as a great spur to improve the whole system.

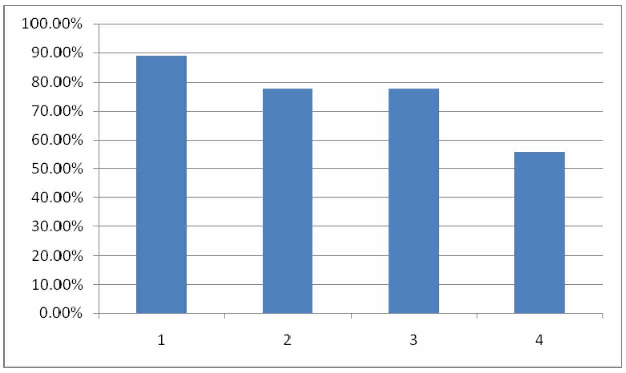


Figure 6. Results of the questionnaire proposed to the students

V. RELATED WORKS

This section points out some implementations that have been set up by other educational institutions to provide remote access to DSP test-bench board or more generally, to embedded systems for didactical purposes.

The University of Deusto developed the WebLab-Deusto [8], a remote laboratory infrastructure which hosts several different experiments. These experiments can be performed by students through a user friendly web interface: they can upload an application for the specific embedded system, use button and switches to control the remote devices and finally they can view the outputs received from a webcam. The WebLab-Deusto offers remote access to three different test benches: the WebLab-CPLD, the WebLab-FPGA and the WebLab-PIC [9, 10].

The DSP remote laboratory of the University of Maribor is presented in [11]. It uses an in-house developed embedded hardware and a software control system based on MATLAB/Simulink and LabVIEW. This remote laboratory enables the users to easily interact with a set of experiments through a friendly LabVIEW user interface. In addition, this remote laboratory includes a booking system, which enables remote users to book experiments in advance.

The eDSPlab [12] is hosted at the Electronic Engineering Department of the University of Seville and is used in an undergraduate microprocessor course. The laboratory uses as device under test a Texas Instruments TMS320C30 digital signal processor connected to a waveform generator and a common oscilloscope. The laboratory instrumentation can be also monitored through a webcam which offers visual real-time feedback of the laboratory workbench.

Another example of remote laboratory focused on DSP is maintained by the Arizona State University which has developed thd RESP (Real-Time Embedded Signal Processing) laboratory [13]. The laboratory offers two different workbenches: the first one is based on a Freescale evaluation module board that includes a DSP56858 chip and the second one is based on the DSK5510 kit from Texas Instruments which is a low-cost standalone development platform that enables users to evaluate and develop applications for the TI C55XX DSP family. This remote laboratory is currently used in a Real-Time DSP course, and the exercises are focused on FFT, FIR (Finite Impulse Response) filters and musical notes synthesis.

Similar examples of DSP remote laboratories are presented in [14] and [15].

The investigation of these labs shows that most of them require installing extra plug-ins or frameworks on the user side. This can cause compatibility problems between the extra software and the end-user environment. Moreover, in presence of firewalls, further problems can arise.

The architecture presented in this paper takes advantage from a completely web based development environment and makes easy to extend the current test bed with new devices, development boards and application contexts.

VI. CONCLUSIONS AND FUTURE WORKS

In this paper we have presented an interactive remote laboratory on Digital Signal Processing that allows developing and executing real C applications in a completely web based environment. The laboratory has been successfully tested by the students during the last course on Embedded Systems 2009/2010 at the Electronic Engineering Faculty of the University of Genoa. The satisfactory result of the questionnaire compiled by the students showed the effectiveness and usefulness of the system but it also revealed aspects which need further improvements such as the usability of the application and the integration of new embedded devices in the laboratory architecture.

So far, the laboratory uses the Analog Devices BF533 Ez-KiT Lite board and its software development environment, but its components are reusable and other test beds can be easily deployed. In the near future the authors plan to extend the lab offer and to add the Texas Instruments DaVinci EVM development board [16] and other common measurement instrumentation, such as oscilloscopes and wave generators.

At present, the remote laboratory manages simultaneous accesses to the test bench and it automatically queues the requests from different users, but the solution appears inadequate when users want to execute applications which lock the instrumentation for a large time slot. In this case asking users to queue for their turn to use the laboratory instrumentation wastes their time. In order to offer interactive experiments which need an exclusive control of the resources for a long time, the authors planned as a future work to support the lab with a booking system which will manage and schedule the access to the laboratory instrumentation. The authors will make use of the iLab Shared Architecture (ISA) [17, 18] which provides a unifying software framework which can support access to a wide variety of online laboratories. ISA allows distributing laboratories across an arbitrary number of locations and provides scheduling and booking mechanisms which allows to coordinate reservations from multiple users for a single lab server. Before accessing the remote laboratory, users must be authenticated by the ISA system and then they must reserve a specific time slot for their experiment sessions. The scheduling system provided by ISA also notifies users if their reservation must be cancelled or changed.

ACKNOWLEDGMENT

The authors would like to thank Danilo Pesce, Niki Fortunato and Federico Idilli for their precious contribution in the prototype development.

REFERENCES

[1] REmLab SourceForge project, http://rem-lab.sourceforge.net/.

[2] REmLab website, http://remlab-esng.dibe.unige.it/.

[3] ADSP-BF533 Blackfin Embedded Processor Datasheet. http://www.analog.com/.

[4] Moodle homepage, http://moodle.org/.

[5] Analog Devices, Inc., “VisualDSP++ 4.5 C/C++ Compiler and Library Manual 4-1 for Blackfin Processors”, April 2006, Part Number 82-000410-03.

[6] P. Buschiazzo, A.M. Scapolla, "A Service Oriented Framework to Set up Flexible and Distributed Data Acquisition and Control Systems", GRIT'09 Workshop, Lyon, France, October 2009.

[7] FoxController SourceForge project, http://foxcontroller.sourceforge.net/.

[8] J. García Zubia, U. Hernández, I. Angulo, P. Orduña, J. Irurzun. “Acceptance, Usability and Usefulness of WebLab-Deusto from the Students Point of View”, International Journal of Online Engineering. Vol. 5., no. 1, 2009. ISSN: 1861-2121.

[9] J. García Zubia, D. López-de-Ipiña, U. Hernández, Pablo Orduña, I. Trueba, “An Approach for WebLabs Analysis”, International Journal of Online Engineering. Special Issue REV 2007, vol. 3., no. 2, 2007. ISSN: 1861-2121.

[10] J. García-Zubia, I. Angulo, U. Hernandez, P. Orduña, “Plug&Play Remote Lab for Microcontrollers: WebLab-DEUSTO-PIC”, 7th European Workshop on Microelectronics Education May 28–30, 2008, BME – Budapest, Hungary.

[11] D. Hercog, B. Gergic, S. Uran, K. Jezernik, “A DSP-Based Remote Control Laboratory,” *Industrial Electronics, IEEE Transactions on* , vol.54, no.6, pp.3057-3068, Dec. 2007. (doi:10.1109/TIE.2007.907009)

[12] S. Gallardo, F. Barrero.; S.L. Toral, M.J. Duran “eDSPlab: A remote-accessed instrumentation laboratory for digital signal processors training based on the Internet,” *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference on* , vol., no., pp.4656-4661, 6-10 Nov. 2006. (doi: 10.1109/IECON.2006.347780)

[13] R. Ferzli, L.J. Karam, “An Online Web-Based Real Time Digital Signal Processing Course,” *Frontiers in Education Conference, 36th Annual* , vol., no., pp.6-11, 27-31 Oct. 2006. (doi: 10.1109/FIE.2006.322627)

[14] Z. Dvir, "Web-Based Remote Digital Signal Processing (DSP) Laboratory Using The Integrated Learning Methodology (ILM)," *Information Technology: Research and Education, 2006. ITRE '06. International Conference on* , vol., no., pp.211-216, 16-19 Oct. 2006. doi: 10.1109/ITRE.2006.381567.

[15] A. Kalantzopoulos, D. Markonis, E. Zigouris, “A Remote Laboratory for Real-Time Digital Image Processing on Embedded Systems”, International Journal of Online Engineering, Vol. 5, No. 4, 2009.

[16] DM355 EVM Technical Reference, April 2008, Number 509905-0001 Rev. E.

[17] MIT iCampus: iLabs Architecture, http://icampus.mit.edu/iLabs/architecture/.

[18] V.J. Harward, J.A. Del Alamo, et al., “The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories”, Proceedings of the IEEE, Vol. 96, No. 6, 2008. (doi:10.1109/JPROC.2008.921607)

AUTHORS

P. Buschiazzo is with the Biophysical and Electronic Engineering Department, University of Genoa, Genoa, Italy (e-mail: paolo.buschiazzo@unige.it).

D. Leoncini is with the Biophysical and Electronic Engineering Department, University of Genoa, Genoa, Italy (e-mail: davide.leoncini@unige.it).

R. Zunino is with the Biophysical and Electronic Engineering Department, University of Genoa, Genoa, Italy (e-mail: rodolfo.zunino@unige.it).

A. M. Scapolla is with the Biophysical and Electronic Engineering Department, University of Genoa, Genoa, Italy (e-mail: scapolla@unige.it).

This article was modified from a presentation at the REV2010 Conference at KTH, Stockholm, Sweden in June 2010. Submitted July 15th, 2010. Published as resubmitted by the authors July 29th, 2010.