Realization of Distributed Remote Laboratory and Remote Debug Software for Embedded System

YIN Wei-feng , SUN Rong-gao, WAN Zhong
School of Computer Science and Information Technology
Zhejiang Wanli University
Ningbo, Zhejiang
P.R.China
http://www.zwu.edu.cn

Abstract: - Remote laboratories are becoming widely accepted in universities for providing distance education and for augmenting traditional laboratories. In this paper, A remote laboratory for embedded system is presented, where students at remote sites can perform actual experiments using actual hardware equipment and tools concurrently. Distributed architecture is presented using Web Services technique to share laboratories resource for multi-user. A design of ARM debugging interface based on JTAG is given. This debugging interface has been used for remote debugging to build embedded system remote laboratory. A Integrated learning environment for remote experimentation for embedded system is provided for convenient learning by students self.

Key-Words: -Virtual Laboratory, Remote Laboratory, Embedded System, Web Services, JTAG debug.

1 Introduction

Engineering education necessitates the use of laboratories for measurement, data collection, and analysis and design activities as well as for hands on experience of equipment, physical devices. Local laboratories are the traditional way of doing experimentation. Neither a virtual environment nor remote access can replace its function. But it has disadvantages like fixed time and place, limitation on the number of equipment sets and hence the number of students who can use them.

As the broadband connectivity to the Internet becomes common, Web based e-Learning have come to play an important role in self-learning, where learners are given much flexibility in choosing place and time to study. Even though local laboratories are still the most widely used technique for experimentation, the ideas of remote access to laboratories and virtual laboratories were developed. Online laboratories became a very useful support for practical aspects of teaching methods since the time their technical basis got available world wide. There are two main classes of such online labs: Remote Labs and Virtual Labs.

Virtual laboratory is a simulation environment not to interact with real equipment. Virtual laboratories rely heavily on fast processor, advanced computer graphics and visual programming technology. Virtual instrumentation allows the mimicking of real equipment and can be used in data collection, analysis and interfacing to real equipment. However, it is very expensive and time consuming to mathematically model many system and complex for making the simulation close to real. Moreover, simulations are artificial and they cannot instill the sense of reality in the learner. Remote laboratories are software environments that run experiments by interacting with real devices, which allow remote users to communicate with measurement devices and experiments set up to make experiments on a real system[1].

Being required by industry, embedded systems have become very popular learning courses recently. Experiences in these areas have shown that a complementary interdependent approach, combining theoretical material underpinned by practical exercises, is essential for effective learning [2]. For engineering related distance education courses the use of a web based delivery mechanism is the only realistic method of providing practical hands-on experience, allowing remotely located students to complete laboratory assignments, unconstrained by time or geographical considerations while developing skills in the use of real systems and instrumentation[3]. This paper presents a Distributed Remote Laboratory for Embedded System (DRLES), where students at remote sites can perform actual experiments using actual hardware equipment and

tools concurrently. A GUI client application based web is given to program remotely to embedded system experiment board. Web services technology is using to provide services for multi-user sharing laboratories resource. The paper is organized as follow: Section 2 summaries recent research in the area of remote laboratory, Section 3 introduces the DRLES project, presents the architecture web based for remote laboratory. Section 4 of the paper realizes software for remote debugging. Section 5 gives an integrated learning environment of the DRLES for remote experimentation. Section 6 of this paper summarizes recent research, discusses future direction for further investing.

2 Remote Laboratories

Remote laboratories involve many engineering fields. The first remote laboratories were control engineering and robotics laboratories. Lately, remote laboratories have become more common in other engineering fields.

There are many examples of the remote laboratory systems such as NetLab[4], the R-Lab[1], the I-Lab[5], the PEARL[6] and the WebLab[7][8].

The NetLab allows a real physical system, set in a laboratory, to be remotely controlled from a PC (personal computer) via the Internet using virtual instruments. The users can control remote instrument over the Internet using a web client. The server software of the system is written in LabVIEW programming language. The server processes users' commands and controls the programmable instruments through the IEEE 4888.2 interface also known as the GPIB (General Purpose Interface Bus).

The R-Lab is developed using a Web Application Service (WAS) over a special software module called the Web-Kernel for the electronics lab experiment series. The Web-Kernel provides a common access point for remote experiments. The sever supports a number of analog/digital interface cards as well as allowing access to a number of modern laboratory instrumentation including oscilloscopes and signal generators.

I-Labs (Internet assisted Laboratories) project is a cooperation of the Stanford Center for Innovations in Learning, California, and the Learning Lab Lower Saxony, Germany, within the Wallenberg Global Learning Network. By working with a remote lab students shall learn to program, maintain and supervise remote devices. In the I-Labs project, an educational concept for collaborative, self-directed learning with tutorial assistance in online laboratories was developed.

The PEARL (Practical Experimentation by Accessible Remote Learning) project is aimed at the development of systems enabling students to conduct real-world manufacturing engineering and digital electronic experiments. The PEARL system is based on a 3-tier client/server architecture, Client machines of PEARL users are separated from machines located in the lab. The Lab Servers are directly connected to the experimental equipment and act as controllers. A gateway machine, acting as a web server, connects PEARL clients (running a java enabled browser) to the lab servers. In PEARL implementations a CORBA-based connection is used between web server and lab server. The PEARL architecture developed can accommodate a wide range of (low-level) equipment interfaces. It has been used in computer vision courses for visual inspection and digital image processing, a remote electronics lab for circuit design and microcontroller programming, a wet chemistry module, a biochemistry experiment centered around an electron microscope.

The WebLab is developed and deployed for an online microelectronics characterization laboratory that allows the characterization of transistors and other microelectronic devices in real time through the internet.

AIM [9] is a collaborative project between the Rensselaer Polytechnic Institute and the Norwegian University of Science and Technology where the remote laboratory is used to complement existing courses in semiconductor technology and device characterization.

Other notable remote laboratories involve embedded system. The University of Zagreb provides several virtual laboratories allowing access to microcomputer development boards, dc motors and a monitor/debugger program for courses involving embedded systems [10]. Some projects generally focus on microprocessor based experiments [11][12]. The Relax project [13] investigates the possibility of developing a new business model where high quality remote experiments and support material can be made available to a worldwide market. A more comprehensive, in-depth review of current online remote access laboratories is available here [14]. DIESEL[15] is another notable remote laboratories which presents and demonstrates a client-server architecture for distant access to an collaborative learning environment for remote experimentation for

embedded system.

Most of being mentioned systems above focus on virtual instrument and client-server architecture.

In the context of electronic engineering and embedded systems this would include facilitating remote access to software tools e.g. cross compilers and VHDL synthesis tools to allow the user to program modern experimental equipment e.g. Microcontrollers, Systems on Chip (SoC) and real-time operating systems (RTOS) while allowing the remote control of debugging instrumentation e.g. logic analyzers and oscilloscopes. The remote user can build, debug and test circuits, connect instrumentation to test points on experimental boards as required. The next section introduces and describes the DRLES project (Distributed Remote Laboratory for Embedded System) which is an integrated learning environment for remote experimentation which seeks to address many of the issues raised in this review.

3 Architecture of Distributed Remote Laboratory for Embedded System

Traditionally students attended practices in campus based laboratories at fixed times. This approach restricted access to laboratories resources to traditional working hours which did not meet the needs of students requiring more flexible attendance patterns to meet current lifestyle commitments.

The overall objective of the DRLES project was to develop remote laboratory for Embedded System, where students at remote sites can perform actual experiments using actual hardware equipments and tools concurrently. A distributed architecture allows multi-user access remote laboratory at the same time. The architecture of the DRLES is shown in the Fig.1.

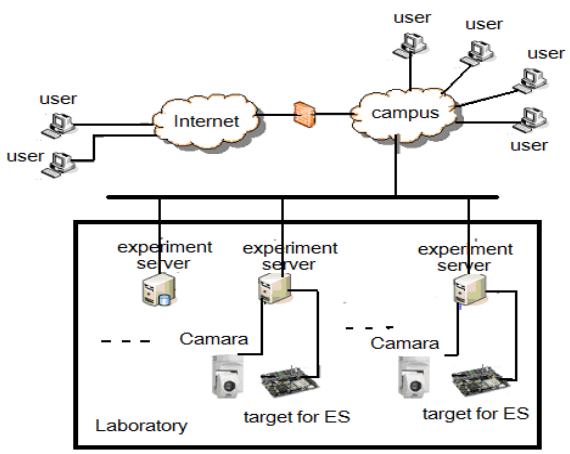

Fig.1 Architecture of DRLES

The system includes some clients, Internet network, management server and Services Providers. The Services Provider consists of experiment server which controls laboratory equipments, experiment board and Video.

Clients as remote users are PCs with access to the Internet. They can be home PCs or Lab PCs allowing access from either remote dial-up or campus laboratories. Their main software need is the ubiquitous web browser. Web based systems are able to entry to service request from local PC's browser, receive collected service results (measurement data) and analyze these data with GUI at local site.

The roles of management server are authentication, election of service, managing measured data and sending back service results to the service requestor.

3.1 Communication Model

Fig.2 shows a four-tier communication model for the system's implements: the presentation layer, the data layer, the business layer and the physical layer.

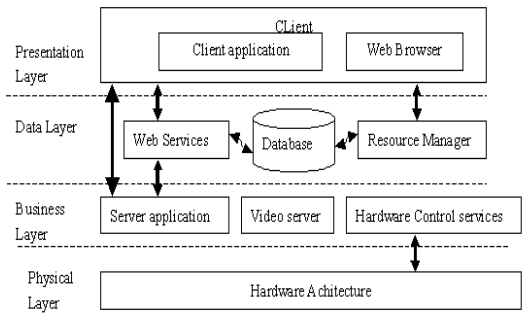

Fig.2 DRLES four tier communication model

The presentation layer is a client application, which is an integrated learning Environment based web. The DRLES client provides the GUI user interface which allows the user to control instrument and download program to embedded system remotely. The data layer provides access to the database. The business layer is implemented in the DLRES server application and provides access and control to the physical layer. The Physical layer consists of all hardware resources (e.g. experiment boards and test instruments).

A web service is used as a gateway between the presentation, business and data layers to allow the client application and server application to access the database. The server application (business layer) responds to commands from the client application by executing the appropriate control programs on the hardware architecture (physical layer) to configure

the embedded circuits, signal routers and instruments while sending commands to the circuit under test.

Fig.3 shows the typical sequence of interactions between the various components of the DRLES software during application initialization and remote experimentation.

During the initialization stages (1-10) the DRLES client attempts to connect to the remote experiment server. The users being confirmed by management server will select the experiment that they wish to carry out to download to the embedded system target board. The remaining interaction (11-15) involve the user working on the experiment.

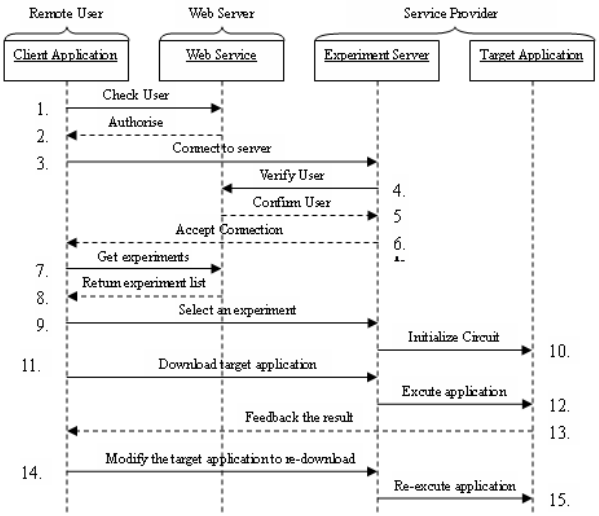

Fig.3 DRLES Client/Server operation

3.2 Web Services

To enable share laboratories resources we propose a service-oriented architecture (SOA) and wrap the function of hardware into Web Services. SOA proposes an architecture for the cooperation of two or more systems via interface-exposed services described in a widely understandable language. Web Services, as the latest technology for distributed applications, provides a new potential to build online experiment systems. The most valuable feature of Web Services for online experiment systems is interoperability. By sending eXtensible Markup Language (XML) based Simple Object Access Protocol (SOAP) message to the remote components, and using Internet protocols, such as Hypertext Transfer Protocol(HTTP) or Blocks Extensible Exchange Protocol(BEEP), Web Services ensures the interoperability of the components on different platforms and/or implemented in different programming languages.

In order to automatically choose the Service Provider with the different kinds of measurement equipments and the target devices, we employed the Web Services Registry Technology. The Architecture is showed in Fig.4. A service provider first registers its services in a UDDI registry server. A service requester searches the registry server and gets all the potential resources. It selects the proper services based on its own criteria. The service requester sends SOAP messages directly to the service provider to invoke the remote service. The service registry specifications define a way to discover information about the Service provider. The specification consists of several related documents that define the Provider ID, The IP Address, Domain, the supported target devices, the measurement equipments and so on. It provides a mechanism for users to dynamically fid out the service providers necessary to perform a scientific experiment.

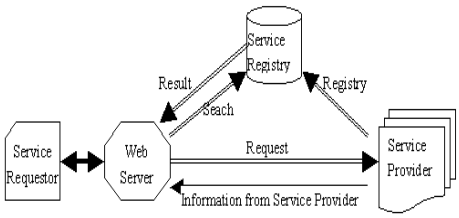

Fig.4 architecture of web services

When multi-user present the request to the web service at the same time, we adopted the Job Management Technology and Shared Resource and Service Management (SRSM) [16] in Management server.

Communication between the client application and the server uses HTTP protocol. Interaction with the web service occurs over HTTP and messages and data are exchanged using the Simple Object Access Protocol (SOAP). In detail: The client sends the request to the web service. The request should contain the ID of the client to identify the session. The web service returns the identifier of the reference. The client always contacts the service using the resource identifier. The experiment is executed and the results are returned to the Web Service. The Web Service records the results in a proper manner and returns the results to the client.

3.3 Service Provider

The Service Provider for an experiment is organized, as a control PC with JTAG interface, a logic analyzer, a Embedded system target unit based ARM9 core with Linux RTOS. Moreover the service provider includes an experiment server. This

experiment server controls laboratory equipments and the JTAG interface to program to a target board. This server is connected the service registry to inform about health checking and observation of what the service provider is being used for process.

The hardware components of a experiment server includes test instrumentation and embedded system boards. The test instruments are configured and controlled from the server using the GPIB protocol while the experimental boards are accessed from the workstation using either RS232 or JTAG connections as required. Each workstation also contains an extensive range of software tools necessary for the design, editing and compiling of embedded software programs.

The server can retrieve measurement data from the equipment and send them back to the client when requested by the user. A Webcam allow users to see the equipment and to monitor the execution of their command. A chat room is provided to allow multiple users to communication with each other to work on experiments collaboratively.

3.4 Shared Resource and Service Management

There are many systems with simply client-server system that usually limit the number of users to the number of actual available resources. In DRLES, we design Shared Resource and Service Management system to perform allocations of the restricted shared services. When many users request the same services at the same time, the job scheduler will processing user requests to prevent conflicts among users. Job management prevents competitions and arranges the service scheduling.

Shared Resource and Service Management system (SRSM) is employed for the job scheduler to process the user requests. SRSM controls service providers which provide the sharable services. It finds out an available service, which satisfies the user's request and instructs the selected service provide to begin an immediate processing. It then returns the service results to the user. Furthermore, SRSM performs other tasks such as management, registration, health checking, and observation of what the service provider is being used for processes.

The job scheduling strategy adopts the first come first service (FCFS) with priority. All users will be allocated the priority according to the identity such as manager, teacher, student, and guest. A priority table will be set in database with the highest priority of manager, the second priority of teacher, then the student, the guest.

4 The Design of Debug Software

In the embedded debug domain, the physical layer usually corresponds to some form of synchronous serial interface that operates at a fraction of the target processor speed. Analogous to the ubiquity of TCP/IP in the networking world, the most common debug interface at the physical level is IEEE 1149.1, otherwise known as JTAG. The Joint Test Action Group (JTAG) is a standardized as the IEEE Std.1149.1 specification for boundary-scan testing. This boundary-scan test (BST) architecture offers the capability to efficiently test components on PCBs with tight lead spacing. Many complex semiconductor devices such as ARM-microprocessors and FPGAs have a interface complied with JTAG's standards. JTAG interface is a primitive hardware interface using the parallel port of PC. JTAG software will be designed to debug or download program by JTAG interface.

4.1 The Architecture of JTAG

JTAG was not conceived as a debug interface originally. Its formal name is Standard Test Access Port and Boundary-Scan Architecture, and it was designed as a mechanism for testing printed circuit boards. The idea was to attach a shift register cell to every signal on the device, chaining the cells together around the periphery (hence boundary scan).

JTAG interface consists of three parts with the TAP controller, scan chain, instruction system shown as Fig.5.

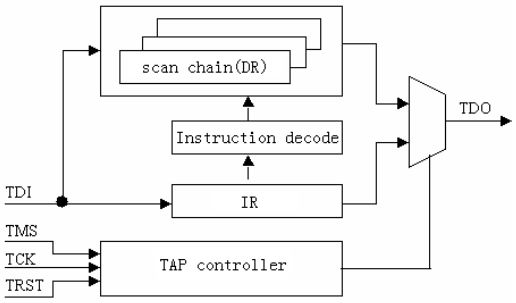

Fig.5 JTAG Achitecture

The Test Access Port, or TAP, is a state machine that cycle through the various TAP states, shifting in data and instructions. There are four dedicated JTAG

signals corresponding to clock (TCK), input (TDI), output (TDO), and mode (machine state) select (TMS). By clocking in commands on TDI via the IR scan path of the TAP controller, a device can be made to send or receive scan data through the DR scan path.

4.1.1 The TAP Controller

The TAP controller is a state machine which is controlled by TMS input signal and TCK clock signal to provide control signal. One selects instruct register or data register, another controls all registers' capture, shift, update and so on operation. The scan chain includes a group data registers that control input and output of core logical.

4.1.2 The Scan Chain

There are three JTAG style scan chains inside ARM-core. These allow testing, debugging and ICEBreaker programming. The scan chains are controlled from a JTAG style TAP (Test Access Port) controller. In addition, support is provided for an optional fourth scan chain. This is intended to be used for an external boundary scan chain around the pads of a packaged device.

The three scan paths are referred to as scan chain 0, 1 and 2. Scan chain 0 allows access to the entire periphery of the ARM core, including the data bus. The scan chain functions allow inter-device testing (EXTEST) and serial testing of the core (INTEST). Scan chain 1 is a subset of the signals that are accessible through scan chain 0. Access to the core's data bus D[31:0], and the BREAKPT signal is available serially. Scan Chain 2 simply allows access to the ICEBreaker registers.

4.1.3 JTAG Public Instruction

There are instruction register and instruction decode logical in which the instructions is shifted in instruction register and then decoded to produce serial of signal with TMS signal when in testing.

Some public instructions are supported in JTAG. The selected scan chain is placed in test mode by the EXTEST instruction. The EXTEST instruction connects the selected scan chain between TDI and TDO. SCAN_N instruction connects the Scan Path Select Register between TDI and TDO. During the CAPTURE-DR state, the fixed value 1000 is loaded into the register. During the SHIFT-DR state, the ID number of the desired scan path is shifted into the scan path select register. In the UPDATEDR state, the scan register of the selected scan chain is connected between TDI and TDO, and remains connected until a subsequent SCAN_N instruction is issued. The selected scan chain is placed in test mode by the INTEST instruction. The INTEST instruction connects the selected scan chain between TDI and TDO. The IDCODE instruction connects the device identification register (or ID register) between TDI and TDO. The BYPASS instruction connects a 1 bit shift register (the BYPASS register) between TDI and TDO. RESTART (0100) instruction is used to restart the processor on exit from debug state. The RESTART instruction connects the bypass register between TDI and TDO and the TAP controller behaves as if the BYPASS instruction had been loaded. The processor will re-synchronize back to the memory system once the RUN-TEST/IDLE state is entered. SAMPLE/PRELOAD instruction is included for production test only, and should never be used.

4.2 The realization of JTAG protocol

The JTAG interface provides excellent visibility into the internal workings of a target, with little to no overhead affecting normal system operation. JTAG signals can be daisy-chained among devices, giving access to any JTAG-enabled devices on a board, or more commonly in the present day to multiple modules on a single piece of silicon. A typical system is shown in Fig.6 which has three parts: the debug host, the protocol converter, the ARM-core such as ARM9TDMI, ARM7TDMI.

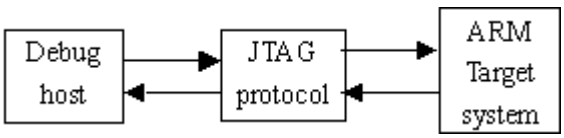

Fig.6 debug system

The debug host is a computer, for example a personal computer, running debugger. The debug host allows the users issue high-level commands such as "set breakpoint at location XX", or "examine the content of memory from 0x0 to 0x100". The protocol converter converts the commands from the debug host to the interface signal for ARM JTAG. The target system is a MCU circuit board based ARM.

Through the TAP the debugger can access all data register and instruction register. There are five interface signals as TCK, TMS, TDI, TDO and TRST, four input signal and one output signal. The TAP can be connected the JTAG interface of development board.

All operations of TAP is driven by the clock signal that The Test Clock Input (TCK) provides for TAP. Test Mode Selection Input (TMS) is used to control the TAP state machine convert. Test Data Input (TDI) is a interface of data input. All data is input by TDI interface bit by bit in serial. Test Data Output (TDO) is a interface of output by which all data is output bit and bit in serial. Test Reset Input (TRST) can reset the TAP controller.

One of the debug interface connects with the JTAG interface of ARM target board, Another connect with PC by parallel port. The JTAG interface circuit is shown as Fig.7.

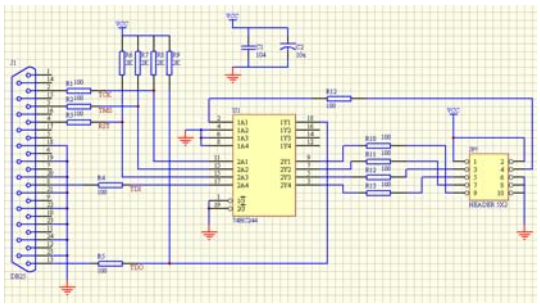

Fig.7 JTAG interface circuit

4.3 The Debugger Software Design

The process of serial test and debug is best explained in conjunction with the JTAG state machine. Fig.8 shows the state transitions that occur in the TAP controller.

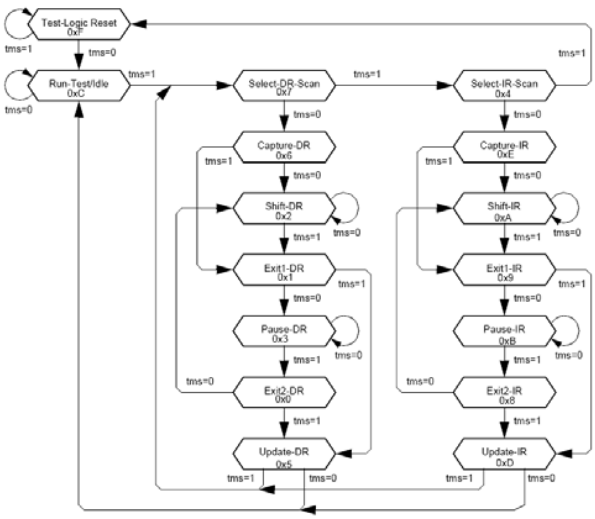

Fig.8 TAP controller state transitions

The procedure access the data register via TAP: a data register accessed is selected by instruction register. The data register will be connected between TDI and TDO. The data is input to the selected data register by TDI and output from the selected data register by TDO.

The software design of JTAG protocol converter is shown as Fig.9.

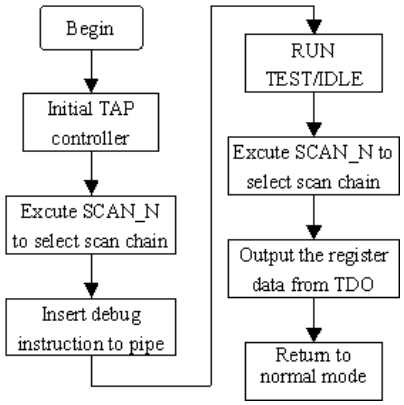

Fig.9 The software frame of Debugger

The procedures are as follow:

(1) Initial TAP controller. Five TCK signals will be produced when TMS is high to force the TAP controller into the Test-Logic-Reset state. Then set the TMS low to produce one TCK signal to force the TAP controller into Run-Test/Idle. In the last, set the TMS high, produce one TCK signal to force the TAP controller into Select-DR-Scan state.

(2) After SCAN_N has been selected as the current instruction, when in SHIFT-DR state, the scan chain select register is selected as the serial path between TDI and TDO. During the CAPTURE-DR state, the value 0b10000 is loaded into this register. This is shifted out during SHIFT-DR, while a new value is shifted in. During the UPDATE-DR state, the value in the register selects a scan chain to become the currently active scan chain. All further instructions such as INTEST then apply to that scan chain.

(3)Insert the debug instruction into instruction pipe line. During debug mode, the instruction can be inserted into pipe instruction of core not to use external data bus. The software control the TAP controller to change the state: SELECT-DR->CAPTURE-IR->SHIFT-DR->UPDATE-DR->inserts the instruction into instruction pipe.

(4) Run RUN-TEST/IDLE. The core processes debug instruction and puts the data on data bus.

(5)Execute SCAN_N to select the scan chain included data bus.

(6)The register data output from TDO in serial. The register state on data bus will be captured and

output from TDO serial.The TAP controller change the state by software as follow: SELECT-DR->CAPTURE-DR->SHIFT-DR->UPDATE-DR.

(7) Return to normal mode. Set TRST low or Set TMS high after five TCK.

5 DRLES Integrated Learning Environment

A Web browser is used as user interface. The Web browser provides a platform for transmitting information as well as an environment to run the client software. A Web Server is the interface between the client and the experiment. The Web itself provides the infrastructure to exchange the necessary information. The main design idea of the system is convenient for user. A Integrated Learning Environment (ILE) based web provides an interface between users and system. Another important aspect is to transport the feeling of a real experiment to the remote user. A video and audio broadcast can provide the remote user with the feeling of being physically present at the location of the real experiment. With the visual feedback, the user supervises the experiment and checks whether the performance of the process is as expected. For system identification purpose and controller evaluation, it is necessary to collect the relevant data of the process. These data have to be stored at the server for download and additional processing by the students. As only one student at a time receives access to an individual experiment, schedules and exclusive access procedures to the experiment are necessary. Users should be able to book laboratory time in advance. They should carry out the whole booking procedure by themselves in order to choose the time most appropriate to their needs.

The Client application includes the ILE uses a distributed software architecture developed using J2EE technology. The Client application consists of some function modules: user login, learning support resources, experimentation operation, visual feedback, and so on. The Login Module is used to authenticate authority of students from database when they input their user account and password. The Learning Support Resources Module provides guide for students to learn the course and do experiments. The Experimentation Operation module provides the function to debug or execute the target application remotely. In chat room students can be talking with each other. The Visual Feedback is used to provide video display for students.

5.1 User login

The database consists of three tables. These are the users table, the schedule table and the logging table. The users table stores user names, password, time quota and administrator flag. If the administrator flag is set, the user is administrator and manages the database. The logging table stores the log messages of the experiment. Messages include logging time and date, type of message and module name.

Students accessing the remote laboratory initially connect to the web server that handles administration and authorization duties and choose an available web services to connect to a experiment server. Each individual experiment server is identical and hosts a range of experimental related hardware and software tools required to carry out practical experiments.

The client application sends the username and password to the web service, which verifies the user's credentials against the database and checks if there is an available service for the current time. If there is an available service the web service will return the IP address of the experiment server to the user requested. The client application then sends a connect request to the server application, again passing the username and password.

The user chooses an experiment and presents a request to web server. The web server finds out an available service, which satisfies the user's request and instructs the selected service provider to begin an immediate processing. It then returns the service result to user.

5.2 Learning Support resources

The DRLES software environment provides a hierarchical tabbed navigation system so that users can quickly navigate through the various features and resources provided by the application. The learning support resources provide the user with the basic assistance needed to begin conducting remote experiment. The resources available to the user include extensive lecture notes and experiment instructions as well as teaching courseware and guidance for the experiments.

A comprehensive help system is available outlining and demonstrating general operation of the remote laboratory and the conduction of experiments. The user can select one from the list of all experiments' item that they wish to carry out. Then the application will send a message to the server to

download a control program to an ARM-based target board to execute the program.

5.3 Visual Feedback

A live video stream for viewing the experiment and an optional audio stream helps to provide a laboratory feeling. To provide visual feedback to the students during experiments where necessary, a series of Webcams have being integrated into web browser with JMF[17] and control applet. The JMF allows users to create web based interactive media applications combined with video streaming capabilities. JMF is an API for incorporating media data types into Java applications and applets. The JMF API is a cross-platform solution written entirely in the Java programming language. Just one single applet is needed to receive the video and audio stream from the server. This applet is stored on the server eliminating the need to install it at the client. A webcam is installed to capture the instantaneous. The result will be feedback with video streaming when the remote users send the program to an ARM-based target board to execute the program. If the executing result is not correct, the remote users can modify the program until the result is correct.

5.4 Experimentation Operation

The Experimentation Operation Module provides three kinds of functions, includes Chat Room, HyperTerminal, and Send Files. The Chat Room is used to process the talking message from students. The HyperTerminal is used to process real-time of commands transmission from students. The Send Files is used to transfer file from students to target boards.

Usually, we want the actions from target board that can be operated more complicated, it requires preprogramming. For example, we first write a program to control 7-segmant LED for display the status of an example, then compile the program become an executable file. Next, we download the executable file into target board. So we can choose the file that we want to transmit and push “select file,” the file selection window will be showed up and path inside the “select file” must be chose. Then, we push “Send” button, the file will begin to transmit data. Under the button has a transmission process bar, it will show transmission status during transmission process. At same time, the file will have already been “FTP” to server. Next, pushing “Download to ARM,” the file will go through server and use the JTAG protocol then be downloaded to target board. Meanwhile, the executable file has already existed inside the target board. Through visual feedback, users will observe the response of the LED on the target board. If the response from target board is not correct, students need to revise program and follow the operation process and repeat the operation again.

6 CONCLUSION

The implementation of a remote access experimental laboratory for embedded systems has being presented. In this paper, we propose using a web services to wrap experiment function. This approach affords resources share for multi-user. An integrated learning environment was developed and the individual components of the integrated learning environment were described. It is convenient for students learning by self and conducting remote experiment. This Future work in this area will concentrate on extending the range of experiments available in the system and on implementing help systems for the more complex experiments detailed in the experimental roadmap. Moreover a generic architecture provides a promising platform from which to develop future remote experiments.

References:

[1] Dervis Z.Deniz, Atilla Bulancak, Gökhan Özcan, a Novel Approach to Remote Laboratories, *33rd Frontiers in Education Conference (FIE2003),* Vol.1, 2003,pp.T3E-8-T3E-12.

[2]W.Wolf, J.Madsen, Embedded systems education for the future, *Proceedings of the IEEE,* vol.88, no.1, 2000, pp.23-30.

[3] J.M.Callaghan, J.Harkin, T.M.McGinnity, P.L.Maguire, Client-server architecture for remote experimentation for embedded system, *The International Journal of Online Engineering,* Vol. 2,No.4(2006), http://www.i-joe.org.

[4] Zorica Nedic, Jan Machotka, Andrew Nafalski, Remote Laboratories Versus Virtual and Real Laboratories, *33rd Frontiers in Education Conference (FIE2003),* Vol.1,2003,pp:T3E-1-T3E-6.

[5] Nils Faltin1, Andreas Böhne, Jörg Tuttas, Bernardo Wagner, Distributed Team Learning in an Internet-Assisted Laboratory, *International Conference on Engineering Education,* August, 18-22, 2002, Manchester, England.

[6] T.Schafer, J.M.Seigneur, A.Donelly, PEARL:A Generic Architecture for Live Experiments in a

Remote Lab, *http://iet.open.ac.uk/pearl/ publications/icsee03.pdf,Outubro.*
[7] J.A.del Alamo, The MIT Microelectronics WebLab: a Web-Enabled Remote Laboratory for Microelectronic Device Characterization. *In NL 2002. http://weblab.mit.edu* .
[8]J.A.del Alamo1, J.Hardison, G.Mishuris, L. Brooks, C.McLean, V.Chan, L.Hui, Educational Experiments with an Online Microelectronics Characterization Laboratory, *International Conference on Engineering Education 2002 (ICEE2002), Manchester, UK*, August 18-22, 2002.
[9] H.Shen, Conducting Laboratory Experiments over the Internet, *IEEE Trans. on Education*, 30, 1999,pp.191-199.
[10] G.Muzak, I.Cavrak, The virtual laboratory project, *Conference Information Technology Interfaces*, 2000, pp:241-246
[11] F.G.Gonzalez-Castono, L.Anido-Rifon, Internet access to real equipment at computer architecture laboratories using the Java/CORBA paradigm, *Computers and Education Journal (Elsevier),* 2001,pp:151-170.
[12] J.Paaso, T.Manninen, Computer supported and controlled learning with the aid of Internet technologies, *Continuing Engineering Education and Lifelong Learning*, 11, no.1/2, 2001, pp:95-106.
[13] F.Bjarne, E.Kjell, T.Malvig, I.Eikaas, Remote Experimentation – New content in Distance learning, *International Conference on Engineering Education*, August 6 – 10, 2001 Oslo, Norway.
[14] MJ.Callaghan, J.Harkin, TM.McGinnity, LP. Maguire, An Internet-based methodology for remotely accessed embedded systems, *IEEE International Conference on Systems, Man and Cybernetics*, October 2002.
[15] J.M.Callaghan, J.Harkin, M.Gueddari, T.M. McGinnoty, P.L.Maguire, Client-server architecture for collaborative remote experimentation, *ICITA 2005.Third internationall Conference*, vol 2, 2005 pp.125-129.
[16] N.Fujii, N.Koike, A time-sharing remote laboratory for hardware design and experiment with shared resources and service management, *ITHET 2005. 6th International Conference on Volume* , Issue , 7-9 July 2005 Page(s): T2B-5 - T2B-10.
[17]Sun Microsystems. Java Media Framework. *http://www.java.sun.com/products/java-media/jmf/*