Remote FPGA Lab With Interactive Control and Visualisation Interface

Fearghal Morgan, Seamus Cawley, Frank Callaly, Shane Agnew, Patrick Rocke, Martin O'Halloran,
Nina Drozd, Krzysztof Kępa, Brian Mc Ginley
Electrical & Electronic Engineering, National University of Ireland, Galway, Ireland
e-mail: fearghal.morgan@nuigalway.ie

***Abstract*—This paper describes a scalable and extendable Remote Field Programmable Gate Array Laboratory (RemoteFPGA) which can be used to enhance the learning of digital systems and FPGA applications. The web-based console provides an always-on, real-time, interactive control and visualisation interface to/from a bank of remote FPGAs. A Xilinx ISE project template enables integration of user HDL-based designs to execute on the RemoteFPGA. Host-FPGA communication is supported using a register-based interface. Users can create real-time, interactive and visual demonstrators of digital systems components. The paper presents a demonstrator for a Finite State Machine (FSM) application, and illustrates the use of web-based control and visualisation for enhanced learning of FSM behaviour. The paper also presents a case study of the use of RemoteFPGA in undergraduate teaching.**

Keywords-Remote FPGA laboratory, digital systems education, reconfigurable computing education, web-based training

I. Introduction

The goal of this research is to enhance the learning of digital systems and FPGA applications. The paper describes an always-on, scalable and extendable Remote Field Programmable Gate Array Laboratory (RemoteFPGA) [1]. RemoteFPGA includes a novel web-based, real-time, interactive control and visualisation interface to the FPGA hardware. Fig. 1 illustrates two FPGA development systems, included within the RemoteFPGA array. Each FPGA module is paired to an associated webcam for streaming real-time images of each active FPGA. FPGA resources are attached via a USB hub to the host server. The server allocates FPGA resource access to a single authorised requester. The system currently supports remote configuration and data transfers to/from Digilent Nexys 2 (-1200 and -500) FPGA modules, and can be scaled and extended to support a range of FPGA technologies. The architecture allows straightforward addition or removal of supported FPGA modules and individual FPGA webcams. The paper describes the internal USB and register-based FPGA interface architecture, and the architecture of the web application server.

The novel RemoteFPGA console enables users to upload a diagrammatic view of their design (at any level of the design hierarchy) and to overlay a range of interactive input-output icons as a User Interface (UI) on top of the graphic. This facilitates real-time interaction with signals within the user's FPGA design. UI switch setting changes on the remote monitor are mirrored within the FPGA input Control and Status Register (CSR), to affect the behaviour of the FPGA hardware. Changes to selected monitored FPGA signals are reflected on the UI graphic through changing UI icon states. The paper presents a demonstrator for a Finite State Machine (FSM) application, and illustrates the use of web-based control and visualisation for enhanced learning of FSM behaviour. The RemoteFPGA has supported a digital design workshop course. Individual user access and activity is monitored using secure account-based access, to provide information on the learning process and the student activity. The paper presents user activity statistics.

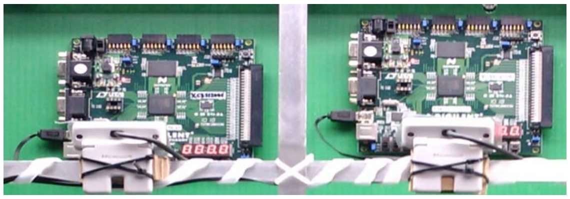


Figure 1. Two FPGA development systems and webcams, included within the RemoteFPGA system.

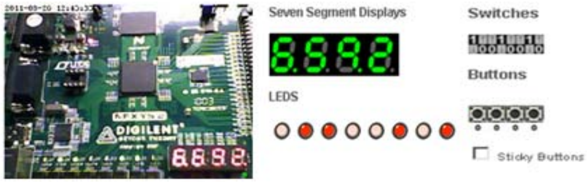


Figure 2. Remote FPGA Mirrored Nexys2 User Interface

A Xilinx ISE project template is provided to enable the integration of user HDL-based designs to execute on the RemoteFPGA. This includes a coreCSR module to manage and monitor the FPGA module-specific interface such as switch and button inputs, LEDs and seven segment display status. Students can create interactive demonstrators of their own project assignments, to better illustrate their understanding and achievement.

Fig. 2 illustrates the top level RemoteFPGA interactive monitor display for the Digilent Xilinx Nexys2. The streamed video window provides a real-time view of the selected FPGA module, and the operation of its display devices, which provide a visual, though limited, guide to the FPGA dynamic system behaviour. The coreCSRs mirror the UI switch settings and display interface. The UI update rate for core and user CSRs is 1-10 Hz depending on the network speed and latency. As with a local FPGA hardware development system, it is not practical to closely monitor fast changing display values. The streamed FPGA image updates at a rate of once per second. Even with normal network latencies, this is adequate for practical visualisation of one-second rate changes in the FPGA display devices.

Fig. 3 illustrates the RemoteFPGA system architecture. This includes the FPGA, webcam and webpage,

978-0-7695-4529-5/11 $26.00 © 2011 IEEE
DOI 10.1109/FPL.2011.98

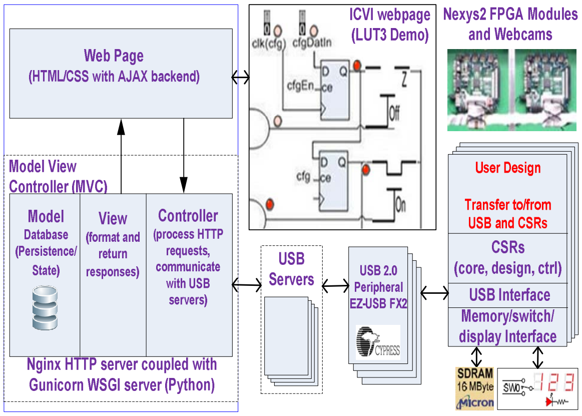


Figure 3. Top level block FPGA block diagram

application server, web server, and USB server for each active session. The USB server provides the link between the web application and the USB port. The system executes on a dual core Intel system, supporting seven FPGAs and associated webcam, concurrently.

The structure of the paper is as follows: Section 2 reviews previous work in the development of remote FPGA laboratories. Section 3 describes the FPGA architecture developed to support the RemoteFPGA. Section 4 describes the application server and web server architectures, and the process of FPGA configuration and data transfer. Section 5 presents a RemoteFPGA demonstrator. Section 6 presents a case study of the application of the RemoteFPGA in undergraduate teaching. Section 7 concludes the paper and proposes future work.

II. Related Work

This section reviews previous work in the development of remote FPGA laboratories. Lowe [2] highlights the incentives for developing remote laboratories including reducing cost, facilitating inter-institutional resource sharing, security, reliability, flexibility and convenience, enriching the student interaction, and pedagogic aspects of remote laboratories. The use of technologies such as Asynchronous Javascript and XML (AJAX) simplify remote lab architectures. With remote labs, more rather than less experimentation by students becomes possible, and remote labs make possible student exposure to systems which might not have otherwise been afforded them. Trevelyan [3] reports that students using remote access laboratories operated equipment for much more time than in conventional laboratory classes, and learning outcomes seemed to be significantly improved as a result.

The authors have provided a summary review [4] of reported work and reported remote FPGA laboratories. Many remote laboratories comprise analogue expansion modules and support a range of laboratories including analogue and digital components. Many use a LabVIEW interface and Virtual Network Computing (VNC) to a pool of remote test and measurement equipment (function generators, oscilloscope etc), to control and monitor the remote PC/FPGA. A number of reported remote labs support a wide and complex range of experiments, including a number of FPGA experiments. While effective, the requirement of third party software such as LabVIEW for data acquisition and instrument control, increases system cost and complexity. Soares [5] reports a remote Altera FPGA lab for introductory digital systems courses, which uses the Altera Quartus in-system memory content editor to access and control/display the state of FPGA module switches, without requiring additional hardware or third party software. The RemoteFPGA novel web-based, real-time, interactive control and visualisation interface to the FPGA hardware provides a unique platform for interactive digital systems training. RemoteFPGA does not require additional third party software or hardware.

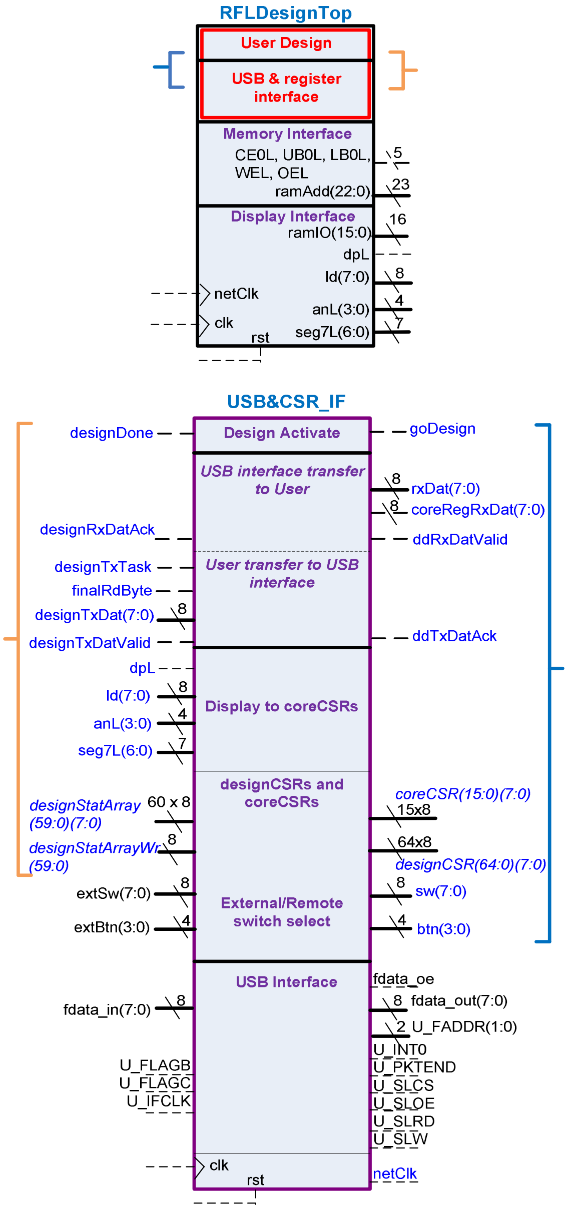


Figure 4. Top level block FPGA block diagram

III. FPGA ARCHITECTURE

This section describes the FPGA architecture developed to support the RemoteFPGA. Fig. 4 illustrates the FPGA top level system block diagram. The USB&CSR_IF subsystem includes the host-FPGA (USB) interface, 16 core CSRs and 64 design CSRs. Users can therefore select up to 64-bytes of design data for control or monitoring of the FPGA behaviour. The RFLDesignTop subsystem includes the user design, along with the interfaces to/from USB&CSR_IF, SDRAM and displays/switches. RFLDesignTop selects between external or CSR-based switch inputs, and stores FPGA display outputs to coreCSRs. User designs can be integrated into the RemoteFPGA HDL hierarchy using the RFLDesignTop VHDL template. RFLDesignTop also defines the signals selected as user interactive input controls or monitors to be overlaid on the interactive monitor diagram. The user is required to connect these signals to their FPGA design top level HDL before integrating their HDL within the RemoteFPGA top level HDL model.

Complex applications can be implemented using host-based command and data sequences. For example, a read/write SDRAM memory console interface is included in the RemoteFPGA to support block SDRAM data transfers.

IV. REMOTEFPGA WEB APPLICATION

This section describes the RemoteFPGA web application server architecture (Fig. 3) and functionality, and the process of FPGA configuration and data transfer. The operation of the interactive monitor is also outlined. Fig. 3 illustrates the web application server and one USB server, FPGA and webcam for each active session. The web application associates an FPGA/webcam pair with a user session, and creates and manages associated information within a database. The database provides persistent storage and state information for user accounts, uploaded bitstreams, system block diagrams and FPGAs. A USB server is activated for each FPGA/webcam session. Each USB server executes on the PC to which the FPGAs and webcams are connected, and provides the link between the web server and the specific Cypress EZ-USB FX USB 2.0 peripheral. The web application configures the FPGA with the requested configuration bitstream and the USB server manages data reads/writes from/to the USB interface module implemented on the FPGA.

A Model View Controller (MVC) framework supports the GUI application, FPGA/webcam resource management, and configuration and data transfers. The MVC model element provides the persistent data store (state management). The RemoteFPGA webpage requests data to/from the USB server and uses http streaming to send and receive real-time data to/from the FPGA. Webpage javascript supports the addition, placement and update of items on a predefined system block diagram to provide the console interface.

The FPGA configuration steps, executed by the web application, are as follows:

- EZ-USB firmware is loaded to enable USB JTAG configuration.
- Xilinx ISE IMPACT is used to convert the FPGA configuration bitstream to SVF format suitable for JTAG configuration.
- SVF configuration data is programmed to the FPGA over JTAG via EZ-USB.
- EZ-USB firmware is loaded to enable USB communication to the FPGA. A USB server is activated, connects to the EZ-ESB device and listens for a connection from the web application.

Subsequent data transfers between the RemoteFPGA web page and the FPGA use the USB server via a socket connection from the web application. Access to the RemoteFPGA is provided using a secure account system. One logged in, a user requests access to an FPGA from a list of available devices, e.g, Xilinx Spartan-3E XC3S1200E or XC3S500E device. The system automatically allocates an FPGA/webcam pair to the user, and removes this resource from the available list. FPGA configurations are performed sequentially to balance the CPU load. Configuration typically takes several seconds and queued users are automatically advised.

The RemoteFPGA system includes a library of reference design bitstreams which all users can use to configure an FPGA. Selection of a reference demonstration automatically configures the FPGA and opens the associated interactive system diagram and user interface to the user.

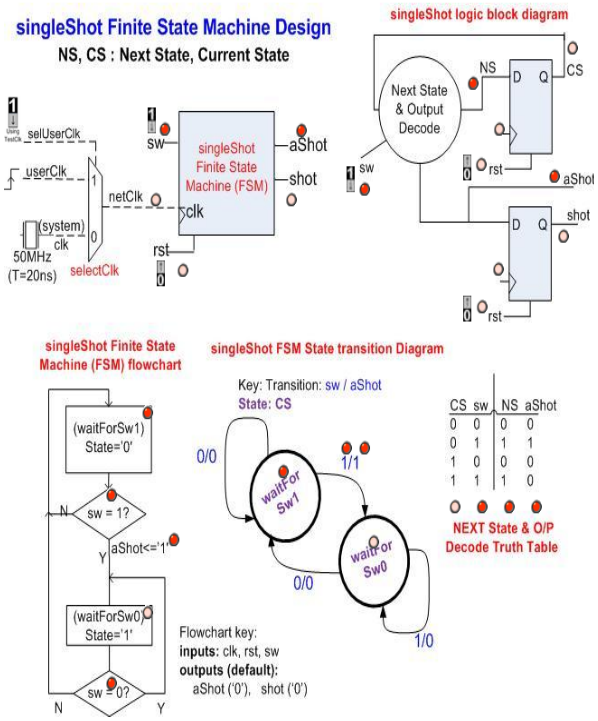
V. REMOTEFPGA DESIGN DEMONSTRATOR

This section presents a RemoteFPGA demonstrator for a single shot Finite State Machine (FSM) (Fig. 5) and illustrates the novel use of web-based control and visualisation for enhanced learning of FSM behaviour. The singleShot asserts signal aShot for a maximum of one clock period on assertion of input signal sw. aShot is an unregistered output (Mealy state machine). The interactive graphic (Fig. 5) includes a range of elements typically used to describe FSM behaviour. These include the singleShot FSM component, flowchart, state transition diagram, system RTL block diagram and truth table. Each element is animated during FSM operation on the remoteFPGA hardware. The user can interact in real-time with the FSM, implemented FPGA hardware design, while observing (in context) the behaviour of signals (at any level of the design hierarchy). The RemoteFPGA system also includes a clock gating function within the CoreCSR subsystem which enables interactive user control of clock activity to aid interaction with the design implemented on the FPGA.

Feedback from students has indicated improved understanding of FSM behaviour and the relationship between the various digital system behaviour description formats, compared to that obtained from static text-book-based descriptions.

VI. CASE STUDY: APPLICATION OF THE REMOTEFPGA IN UNDERGRADUATE TEACHING

This section illustrates how the RemoteFPGA can provide information on the learning process and student activity. Fig. 6 illustrates course participant activity on the RemoteFPGA during three weeks of usage on a digital design and implementation workshop course. Fig. 6 displays the number of logins (which automatically


singleShot Finite State Machine Design
NS, CS : Next State, Current State
singleShot logic block diagram
selUserClk
userClk
netClk
(system) clk
50MHz (T=20ns)
selectClk
1
0
sw
rst
clk
singleShot Finite State Machine (FSM)
aShot
shot
Next State & Output Decode
NS
CS
D
Q
rst
aShot
shot
singleShot Finite State Machine (FSM) flowchart
(waitForSw1) State='0'
sw = 1?
aShot<='1'
(waitForSw0) State='1'
sw = 0?
N
Y
singleShot FSM State transition Diagram
Key: Transition: sw / aShot
State: CS
0/0
1/1
waitFor Sw1
waitFor Sw0
0/0
1/0
Flowchart key:
inputs: clk, rst, sw
outputs (default):
aShot ('0'), shot ('0')
NEXT State & O/P Decode Truth Table

CS	sw	NS	aShot
0	0	0	0
0	1	1	1
1	0	0	0
1	1	1	0

Figure 5. Console interface demonstration of a Finite State Machine

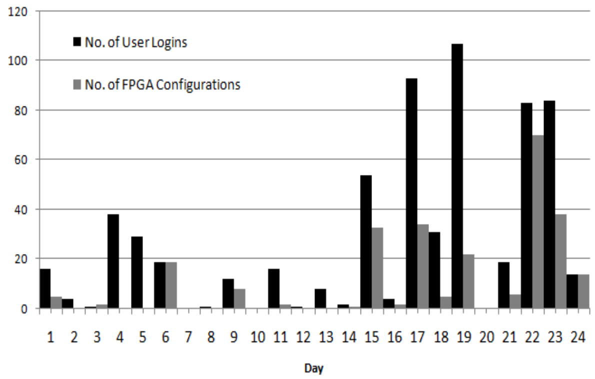


Figure 6. Participant activity. Number of logins, and number of participant-specific FPGA configurations as course progresses (note: a demonstration FPGA design is automatically configured on login).

configure a demonstration FPGA design and related interactive console), and the number of participant-specific FPGA configurations. The workshop requires the implementation of several FPGA hardware subsystems. Participants capture an integrated series of VHDL modules [6], verify design behaviour using the Xilinx ISim simulator and synthesise the design, before configuring the Xilinx Spartan-3E FPGA on the remote Digilent Nexys 2 FPGA development system. Since FPGA configuration and testing occupies a relatively small amount of the VHDL capture-to-FPGA implementation process, remote sharing of five FPGA modules has proven effective for a group of 20 participants, working in pairs. No manual FPGA hardware setup is required.

The course assignment extended over a period of approximately three weeks. Fig. 6 illustrates the increasing usage of the RemoteFPGA system as the assignment progresses. Fig. 6 highlights the increasing configuration of users' own designs as the course progresses. Results indicate more frequent access to the FPGA than in previous workshops which used bench-based FPGA development systems. Participants have benefitted in particular from the interactive control and visualisation of demonstrators and user designs. Feedback from students has indicated enhanced understanding of digital systems behaviour, compared to that obtained from static text-book-based descriptions. The RemoteFPGA system could also be used to monitor both under-activity, and over-activity of course participants, where over-activity may suggest an over-reliance on an inefficient design-by-implementation methodology, rather than a comprehensive simulation-based test strategy prior to FPGA implementation.

VII. CONCLUSIONS AND FUTURE WORK

This paper describes a scalable and extendable Remote FPGA system, used to enhance the learning of digital systems and FPGA applications. The RemoteFPGA web-based remote console provides a novel real-time, interactive control and visualisation interface to/from a bank of remote FPGAs. A Xilinx ISE project template enables integration of HDL-based user designs. The application of RemoteFPGA on a digital design workshop course has verified its efficacy in supporting an interactive learn-by-doing methodology. Extending support to other Xilinx FPGA modules, and Altera FPGA modules is in progress.

ACKNOWLEDGMENT

Supported by the Irish Research Council for Science, Engineering and Technology (IRCSET), International Centre for Graduate Education in Micro and Nano Engineering (ICGEE), NUI Galway Millennium Research Fund, NUI Galway Centre for Excellence in Teaching & Learning, and Xilinx University Programme.

REFERENCES

[1] RemoteFPGA website, remoteFPGA.com.

[2] Lowe, D, "Evolving Remote Laboratory Architectures to Leverage Emerging Internet Technologies", *IEEE Transactions on Learning Technologies*, vol 2, Issue 4, Aug 2009, pp 289-294.

[3] James Trevelyan, "Lessons Learned from 10 Years Experience with Remote Laboratories", *International Conference on Engineering Education and Research"Progress Through Partnership*", 2004, pp 687-697.

[4] Fearghal Morgam, Seamus Cawley, "Enhancing Learning of Digital Systems using a Remote FPGA Lab", *6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC 2011)*, June 2011.

[5] Joao Soares, Jorge Lobo, "A Remote FPGA Laboratory for Digital Design Students", *7th Portuguese Meeting on Reconfigurable Systems (REC 2011), Feb 2011,* pp 95-98.

[6] Fearghal Morgan, Patrick Rocke, Martin O' Halloran, "Applied VHDL Training Methodology, EDA Framework and Hardware Implementation Platform", *International Conference on Reconfigurable Computing and FPGAs* pp.8 pp.-19, 28-30 Sept. 2005.