RUN TIME DYNAMIC PARTIAL RECONFIGURATION USING MICROBLAZE SOFT CORE PROCESSOR FOR DSP APPLICATIONS

C. V. Borkute[1], A. Y. Deshmukh[2]

[1]*Research Student, G.H.Raisoni College of EngineeringNagpur, Maharshtra, India,* ***cvborkute@gmail.com***
[2] *Deputy Director & Dean(Planning & Quality Assurance) G.H.Raisoni College of Engineering Nagpur, Maharashtra, and India,* ***aydeshmukh@gmail.com***

Abstract

DSP Application requires a fast computations & flexibility of the design. Partial Reconfiguration (PR) is an advanced technique, which improves the flexibility of FPGAs by allowing portions of a design to be reconfigured at runtime by overwriting parts of the configuration memory. In this paper we are using microblaze soft core processor & ICAP Port to reconfigure the FPGA at runtime. ICAP is accessed through a light-weight custom IP which requires bit stream length, go, and done signal to interface to a system that provides partial bit stream data. The partial bit stream is provided by the processor system by reading the partial bit files from the compact flash card. Our targeted DSP application is matrix multiplication; we are reconfiguring design by changing partial modules at run time. To change the partial bit stream we interfaces a microblaze Soft processor & using a UART interface.ISE13.1 & PlanAhead is used for partial reconfiguration of FPGA .EDK is used for microblaze soft processor design & ICAP Interface .The simulation is done using Chip Scope Logic Analyzer & the complete hardware implementation is done on Xilinx VIRTEX -6 ML605 Platform.

Keywords *— PlanAhead, EDK, Dynamic partial reconfiguration, ICAP, Matrix multiplication, Chipscope pro analysis, DSP application, Microblaze processor*

----------***----------

1. INTRODUCTION

Xilinx partial reconfiguration extends the inherent flexibility of the FPGA by allowing specific regions of the FPGA to be reprogrammed with new functionality while applications continue to run in the remainder of the device. Partial reconfiguration addresses three fundamental needs by enabling [1]

- Reduce cost and/or board space
- Change a design in the field
- Reduce power consumption

The two most prevalent user problems, addressed by partial reconfiguration are:

- Fitting more logic into an existing device
- Fitting a design into a smaller, less expensive device

Historically, designers have spent days, if not weeks, trying new implementation switches, reworking code, and re-engineering solutions to squeeze them into the smallest possible FPGA.

Partial reconfiguration enables these designers to reduce the size of their designs by dynamically time-multiplexing portions of the available hardware resources [2]. The ability to load functions on an as-needed basis also reduces the amount of idle logic, thereby saving additional space. In the past, changing a design in the field required new placement and routing of the design and the delivery of a full configuration file. The engineer also had to shut the system down while making the change. In contrast, when using partial reconfiguration, the designer needs only to place and route the modified function in context with the already-verified remainder of the design, then deliver this new partial image to a system in the field. Moreover, the engineer can dynamically insert new functions while the system is up and running, improving system up-time [3]. Thus, mutually exclusive functions can be plugged into the same space without having to redesign the system or move to a bigger device, Power consumption has become a primary concern for today's designers. Like size and cost, it is a metric with strict limits in most systems. However, as FPGA designs grow in size and complexity, they consume more power. While synthesis and implementation tools coupled with appropriate design techniques can help reduce power consumption, partial reconfiguration implementations can further reduce static and dynamic power. One way to reduce static power is to simply use a smaller device. With partial reconfiguration, designers can essentially time slice the FPGA and run parts of their design independently. The design then requires a much smaller device or fewer devices because not every part of the design is needed 100% of the time. Partial reconfiguration also

has the potential to reduce operating power as well as static power. The SOC implementation [4] is one of the advantage of the partial reconfiguration is the reduced size & complexity as shown in figure1.

Fig 1: Modifying Functionality and Reducing Size using Partial Reconfiguration

This paper describes the complete flow of the DPR (dynamic partial reconfiguration) [5] using ICAP (internal configuration access port), of Microblaze soft processor [6], [7] & verified the changes in the hardware using chi scope pro analyzer at run time. This system is design for the image processing application where matrix multiplication play important role. We have implemented the complete design on Xilinx Virtex 6 FPGA .The simulation results are shown for Matrix multiplication. This partial reconfiguration supports to change the design in run time.

2. RUN-TIME PR WITH ICAP

The ICAP primitive is the hardwired FPGA logic by which the bit stream can be dynamically loaded into the configuration memory. As shown in figure 2, ICAP interfaces to the configuration memory and furthermore provides parallel access ports to programmable resources. During system run-time, a master device (normally an embedded microprocessor) may transmit the partial reconfiguration bit stream from storage devices to ICAP to accomplish the reconfiguration process. The complete ICAP design, in which the ICAP primitive is instantiated, interfaces to the system interconnection fabric to communicate with the processor and memories. Detailed design architectures will be discussed in the next section.

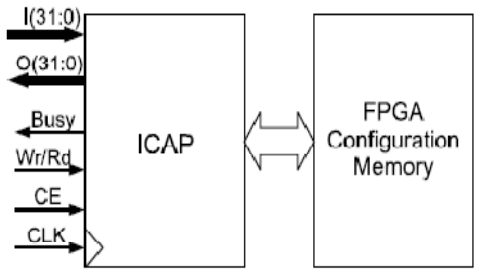

Fig 2: The ICAP primitive on Xilinx FPGAs

3. DESIGN ARCHITECTURE

To implement a design that can be dynamically reconfigurable using a light-weight custom IP that can be used to access ICAP to write partial bit stream. Figure 3 shows a top-level system. The design consists of a reconfigurable user logic (Matrix Multiplication data), having different matrix multiplication inputs. User verifies functionality using HyperTerminal under user application. The dynamic modules are reconfigured using the custom IP having ICAP resource.

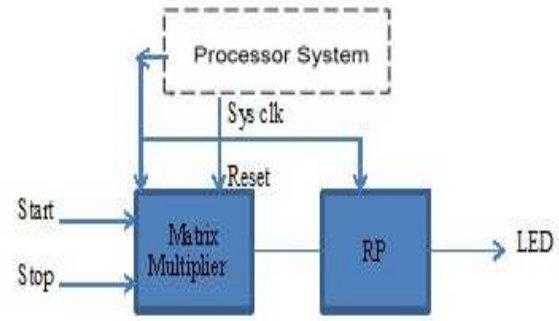

Fig 3: TOP Level system design Architecture

There are two matrix input which are store in BRAM & Can be change using a partial bit stream. Another functionality that is running on board is the LED Counter which runs in two Modes clockwise & anticlockwise depending on the Bit files.

4. SYSTEM DESIGN FLOW

The system design flow Consist of 15 steps to generate the final design files which can be used for the Partial & full FPGA configuration[8]. Figure 4 shows the complete system design flow .Microblaze processor & ICAP Interface can be design through the Xilinx EDK (Embedded design Kit) tool, using Xilinx platform studio .Figure 5 shows the hardware system assembly. The first step in design flow is the system hardware design which can be done using Xilinx platform studio. The next step is to create software project using SDK (Software design Kit) tool. PlanAhead defines a reconfigurable partition & used to define the reconfigurable area & Reconfigurable modules. Chip scope Pro used to debug the system & to monitor the design simulation & signal flow.

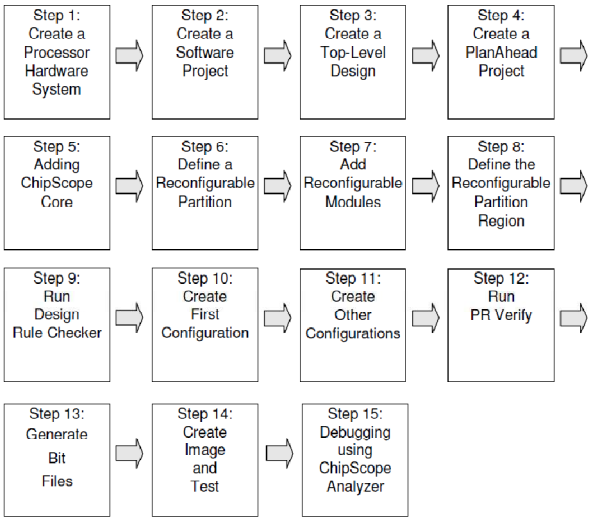

Fig 4: Run time partial reconfiguration System design flow

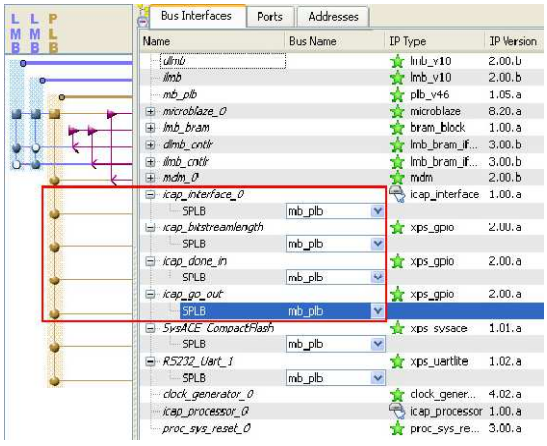

Fig 5: System assembly connections

5. EXPERIMENTAL SETUP:

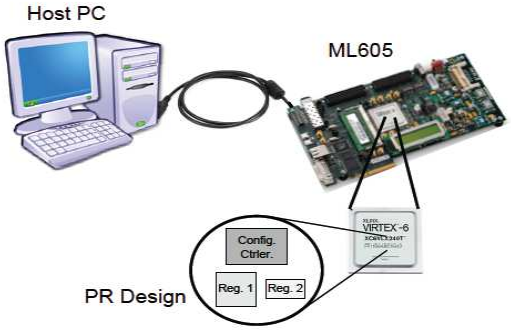

Fig 6: Experimental Test Setup

The ICAP controller design was synthesized using Xilinx ISE 13.2, and was implemented using Xilinx PlanAhead 13.2 The system will be hardware validated by testing it on a Xilinx ML605 evaluation board, which contains a Virtex 6 XC6VLX240T FPGA Figure 6 shows the test setup used to carry out the system design.

6. SIMULATION RESULT

Simulation results for 2X2 Matrix Multiplication is shown in figure 7. The system is reconfigured for the Different matrix input data & run on hardware & run time partial reconfiguration is verified in both simulations as well as in synthesis on hardware. ICAP provides a very good mechanism to change the Partial bits team in runtime .The Input for the changing the bit files is given from hyper terminal of the PC to the microblaze processor which changes the bit stream at run time through ICAP interface. The Chip scope pro debug the program & verified the changes at the time of run time partial reconfiguration of FPGA.

CONCLUSIONS

Dynamic partial reconfiguration provides a very fast reconfiguration of FPGA. Most of the algorithms which are used in DSP, image and video processing, computer graphics and vision and high performance supercomputing applications have matrix operation as the kernel operation. DPR is very useful while handling a time critical application. In this paper we have presented a technique to reconfigure FPGA at runtime to speed up the dsp application.

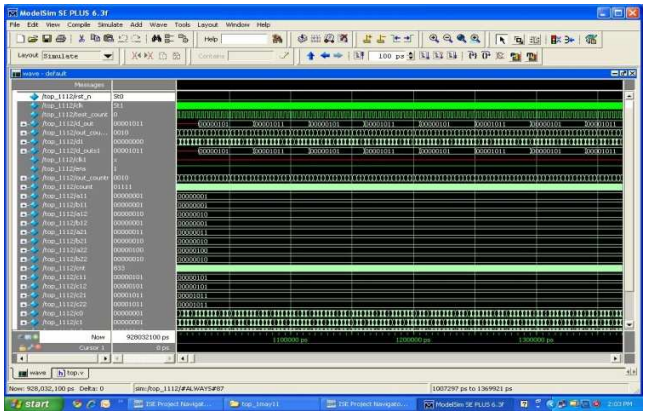

Fig 7: Simulation of 2X2 Matrix Multiplication

REFERENCES

[1]. Kao C. 2005. Benefits of partial reconfiguration. Xcell journal.

[2]. K. Papadimitriou, A. Dollas, and S. Hauk.. 2010. Aneffective framework to evaluate dynamic partial reconfiguration in FPGA systems. Instrumentation and Measurement, IEEE Transactions

[3]. Hanho L. 2007. A self-reconfigurable adaptive FIR filter system on partial reconfiguration platform IEICE transactions on information and systems.
[4]. Griese B., Erik V., Mario P. and Ulrich R., 2004. Hardware support for dynamic reconfiguration in reconfigurable SoC architectures. In Field Programmable Logic and Application.
[5]. Marco D. Santambrogio. Dynamic reconfigurability in embedded system design: a model for the dynamic reconfiguration. Master's thesis, University of Illinois at Chicago, 2004.
[6]. Xilinx Inc, MicroBlaze Processor Reference Guide (August 24, 2004)
[7]. Xilinx Inc, Embedded System Tools Reference Manual, EDK
[8]. Xilinx Inc,Partial reconfiguration user guide UG702 October 5, 2010

BIOGRAPHIES

C. V. Borkute, he has completed his Graduation in Electronics Engineering. He is CEO at Qualitat systems, Pune (India). Perusing Master of Engineering from G. H. Raisoni College of Engineering, Nagpur.He has 10+ years of experience in VLSI & embedded domain. His area of interest is embedded system & VLSI

Dr. A. Y. Deshmukh, he has completed his Ph.D from VNIT Nagpur in 2010. He is currently working as Deputy Director & Dean-Planning & Quality Assurance at G.H.Raisoni College of Engineering Nagpur, India. He has filed 02 Patents.He is also working as Coordinator TEQIP-II (World Bank Assistance Project). He is Technical Committee Member of IEEE Soft Computing, USA. He is also Counselor of IEEE Students Branch. He has to his credit around 50 International Conference and Journal Publications. He has also worked as International Co-Chair for ICETET-08, ICETET-09, ICETET-10, ICETET-11, ICETET-12.He has also worked as Guest Editor for International Journal IJSSST & Reviewer & Session Chair for many conferences.