DIFFERENCE BASED PARTIAL RECONFIGURATION

R.V. Kshirsagar and S. Sharma
Priyadarshini College of Engineering, Nagpur, India
ravi_kshirsagar@yahoo.com
Priyadarshini College of Engineering, Nagpur, India
sanjeevietb@rediffmail.com

ABSTRACT

In this paper, we address the main aspects of Difference based partial reconfiguration on the Xilinx FPGAs. Difference-based partial reconfiguration is useful for making small on-the-fly changes to design parameters such as logic equations, filter parameters, and I/O standards. Partial reconfiguration is the prerequisite of reconfigurable computing, as it allows time-sharing of physical resources for the execution of multiple design modules. Moreover, partial reconfigurable modules can be swapped in or out on the fly from the operating environment control while other modules in the base design continue functioning without incurring any system downtime. This, results in dramatically increase in speed and functionality of FPGA based system.

KEYWORDS: *FPGA, reconfiguration, difference based partial reconfiguration.*

1. Introduction

Recent FGPA devices allow its partial reconfiguration, while the rest remains active and working; this is called dynamic partial reconfiguration. This functionality allows for a very wide range of computational units to be used on a single device. This capability allows the use of smaller devices, that are cheaper and with a lower static power consumption, and the implementation of more complex computational structures that would otherwise not fit in the device.

Difference-based partial reconfiguration can be used when a small change is made to the design. It is especially useful in case of changing LUT equations or dedicated memory blocks content. The partial bit-stream contains only information about differences between the current design structure (that resides in the FPGA) and the new content of an FPGA. There are two ways of difference-based reconfiguration known as a front-end and back-end. The first one is based on the modification of the design in the hardware description languages (HDLs). It is clear that such a solution requires full repeating of the synthesis and implementation processes. The back-end difference-based partial reconfiguration permits to make changes at the implementation stage of the prototyping flow. Therefore there is no need for re-synthesis of the design. The usage of both methods (either front-end or back-end) leads to creation of a partial bit-stream that can be used for a partial reconfiguration of the FPGA.

Normally, reconfiguring an FPGA requires it to be held in reset while an external controller reloads a design onto it. Partial reconfiguration allows for critical parts of the design to continue operating while a controller either on the FPGA or off of it loads a partial design into a reconfigurable module. Partial reconfiguration also can be used to save space for multiple designs by only storing the partial designs that change between designs. In this paper we have implemented two examples. Firstly, we have implemented a full adder and then reconfigured it with full subtractor. Then another example was shifting the leds right and then reconfiguring the leds to shift left.

2. Difference-based partial reconfiguration

The main objective for difference-based partial reconfiguration is allowing small design changes. After the changes are made, the BitGen program is used to produce a bitstream that only programs the differences between the original design and the new one. Depending on the changes, this partial bitstream can be much smaller than the original bitstream. The software can load these bitstreams

quickly and easily. All that is required is an understanding of how to make logic changes using the *FPGA_Editor* application, and the pertinent options to select in BitGen.it.

2.1 Making Small Design Changes Using FPGA_Editor

While there are a myriad of different types of changes that can be made to an FPGA design, in this application i.e. for adder first we have to change LUT programming using FPGA Editor. While it is possible to change routing information, this is not recommended due to the possibility of internal contention during reconfiguration.
We have to change LUT Equations in order to make new adder_test.ncd files for the reconfigurable circuit. The smallest logical element that can be selected is the Slice. First, the block must be viewed. An individual slice is selected by hand and the particular block is edited using the Block Editor toolbar. The attributes are changed using LUT equations. Once these changes have been made, the new adder_test.ncd is used with the BitGen application to generate a difference-based bitstream.

2.2 Creating Difference-Based Partial Reconfiguration Bitstreams

A difference-based partial reconfiguration bitstream can be created with the BitGen utility using the -r switch. This switch produces a bitstream that contains only the differences between the input .ncd file and the original bit file.

2.3 Using Bitstreams and Programming the FPGA

Partial reconfiguration supports serial JTAG programming options. The Xilinx configuration application, iMPACT, can be used in conjunction with any Xilinx download cable to interface to target device(s) for configuration. Alternatively, the user can create board-level functions to control device configuration. Because partial bitstreams are indistinguishable from full bitstreams, end users must be careful to correctly sequence the application of these partial bitstreams to the target devices. The iMPACT software can identify a partial bitstream but cannot determine if it is being applied in the correct sequence order. When downloading a device using a partial bitstream, iMPACT software displays a message indicating that a partial bitstream is being used and that care should be taken to ensure correct sequencing. Beyond that, the end-user configuration experience with partial bitstreams is identical to that of full bit streams.

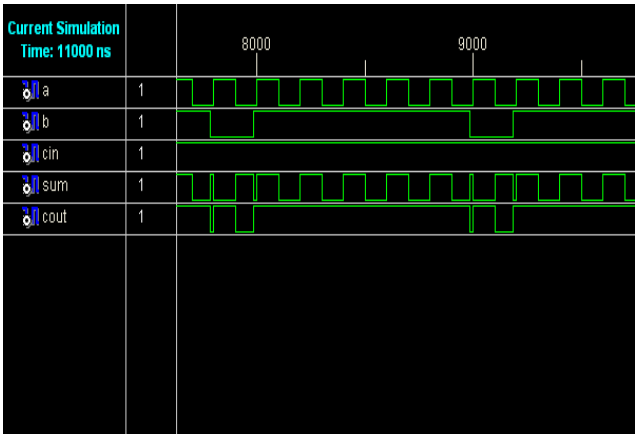

Fig. 1 Simulation of an Adder circuit

Initially the code of the full adder has been written in VHDL and simulated as Fig.1indicates and similarly the code of up counter has been written and simulated as Fig.2indicates.As the program has been simulated it is been downloaded on to FPGA. Further the same adder program can be changed to subtractor by making changes in the equation with the help of FPGA Editor as shown in Fig. 3.The bit

stream is generated and while running the adder program on FPGA the subtractor bit stream is loaded and now the output of subtactor is achieved.

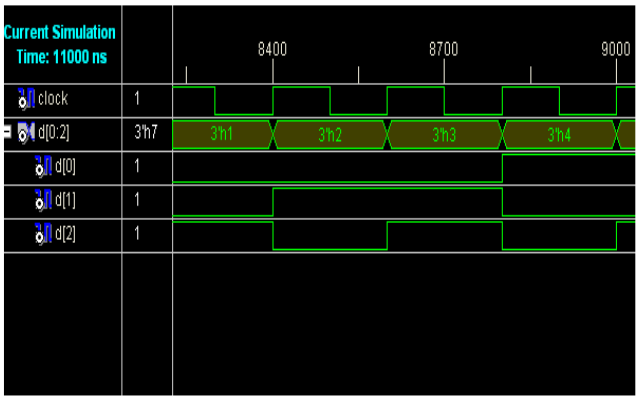

Fig. 2 Simulation of an UP counter

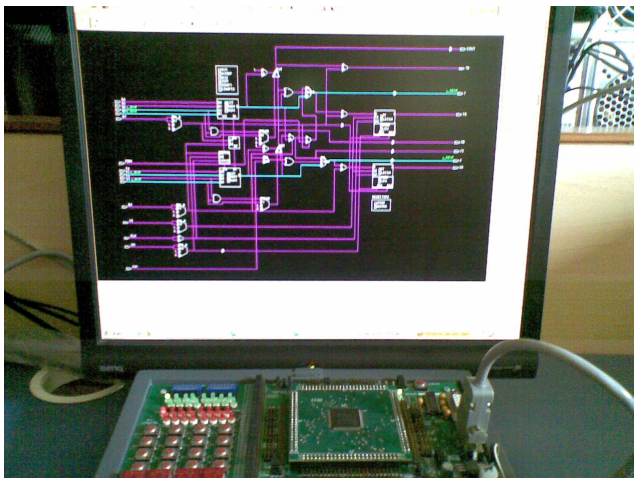

Fig. 3 Implementation on Spartan2s30 FPGA

3. Conclusions

In this paper we have discussed about the implementation flow for Difference Based Partial Reconfiguration. Full Adder/Subtractor and Up/Down counter has been modeled in VHDL and implemented on Xilinx Spartan2Sxcv30tq144 FPGA board with difference based reconfiguration. This in turn reduces the time by rearranging the equation on FPGA Editor and generating the bits to be loaded on to the chip. In future we are going to implement these concepts for making a smart reconfigurable computing system. Thus we have successfully implemented the difference-based reconfiguration using the two examples that is for Full Adder with Full Subtractor and Up Down counter using the Spartan2Sxcv30tq144.

References

[1] Two Flows for Partial Reconfiguration: Module Based or Difference Based, Xilinx Application Note XAPP290 (v2.0) December 3, 2007.

[2] S. Wong, S. Vassiliadis, and S. Cotofana, "Future directions of (programmable and reconfigurable) embedded processors," in Embedded Processor Design Challenges, Workshop on Systems, Architectures, Modeling and Simulation—SAMOS 2002.

[3] R. Krueger, "Using High Security Features in Virtex-II Series FPGAs". Xilinx Application Note XAPP766, version 1.0, Xilinx, Inc. July 2004.

[4] B. Blodget, P. James-Roxby, E. Keller, S. McMillian and P. Sundararajan. A Self-reconfiguring Platform. In Proceedings of the International Conference on Field Programmable Logic, Lisbon, Portugal, Sept.2003.

[5] T. Kean. Secure Configuration of Field Programmable Gate Arrays. In proceeding of 11th International Conference on Field-Programmable Logic and Applications, FPL'2001. Belfast, United Kingdom.

[6] T Wollinger, J. Guajardo, C. Paar, "Security on FPGAs: State-of-the-Art Implementations and Attacks," in ACM Transactions on Embedded Computing Systems, Vol. 3, No. 3, August 2004, Pages 534–574.

[7] K. Bondalapati and V. Prasanna. "Reconfigurable Computing systems," in Proc. IEEE, vol. 90, no.7, pp.1201-1217, July 2002.

[8] Tiago de Albuquerque Reis, Antonio Augusto Frohlich " Operating System Support for Difference-Based Partial Hardware Reconfiguration " RSP '09 Proc. 2009 IEEE/IFIP International Symposium on Rapid System Prototyping Pages 75-80 Publisher IEEE Computer Society Washington, DC, USA ©2009

[9] Salih Bayar and Arda Yurdakul, "Self-Reconfiguration on Spartan-III FPGAs with Compressed Partial Bitstreams via a Parallel Configuration Access Port (cPCAP) Core, " pdf PRIME 2008 - 4th Conference on Ph.D. Research in Microelectronics and Electronics , 22-25 June 2008, Istanbul, Turkey.

[10] Paulsson K, Hübner M, Bayar S and Becker J (2008) Exploitation of run-time partial reconfiguration for dynamic power management in Xilinx spartan III-based systems. In: Proc. Intl. Conf. on Field Programmable Logic & Appln., (FPL 2008). Sept. 8-10. pp: 699-700.

Authors

Ravindra V. Kshirsagar (FIETE,LMISTE) is presently working as Professor and Head of the Department of Electronics Engineering , Priyadarshini College of Engineering.He is also the chairman of Board of Studies(Electronics Engg.) of R.T.M.,N.U., Nagpur. He has done his B.E.(E&TC) in 1984 from Govt. Engg. College, Jabalpur. He completed his M.Tech. (Electronics Engg.) and Ph.D. from VNIT,Nagpur. He has a vast teaching experience of 20 years and 2 years of industry experience. He has published many research papers in national and international conferences. He is a fellow member of IETE and LMISTE .Also he was Ex - IEEE member. His special field of interest includes Reconfigurable Computing, VLSI Design, Fault tolerance and DFT.

Sanjeev Sharma received his B.Tech degree from IET, Barielly, he received his M.Tech degree from RKNEC, RTMNU, India, he received his PG Diploma in VLSI from CDAC, now he is a Ph.D candidate of RTMNU, he has over 6 years of industrial experience, working at CDAC, now he is a professor at PCE, His overall 10 Papers published in international and national conference on selected areas in communication and embedded systems, his research interests include artificial intelligence, wireless communication, embedded systems and soft-core processor.