Dynamic partial reconfiguration in FPGAs

Wang Lie, Wu Feng-yan
Dept. of Computer Science & Electronic Information
Guangxi University
Nanning,China
dzxh_wfy@163.com

***Abstract*—Dynamic parital reconfigurable FPGAs offer new design space with a variety of benefits: reduce the configuration time and save memory as the partial reconfiguration files (bitstreams) are smaller than full ones. This paper introduces a simple reconfigurable system and focuses on the advantages of the newest dynamic partial reconfiguration design flow.**

Keywords-FPGA; Dynamic Partial Reconfiguration(DPR) ; Early Access Partial Reconfiguration(EAPR)

I. INTRODUCTION

Since 80's the Field Programmable Gate Array (FPGA) market growing rapidly with varied of application in different industries.Their great advantage is their flexibility that arises from their programmable nature as compared to systems using application specific integrated circuits (ASICs).There is a new concept evolving in FPGA industry.The so-called dynamic partial reconfiguration(DPR) can be exploited in many application fields,for instance to fulfill space requirements in small portable systems, to create a system-on-a-chip with a very high level of flexibility, to realize adaptive hardware algorithms,and so on.

Section 2 describes the definition of dynamic partial reconfiguration and two different methods are compared.The advantages of the newest design flow is analysed in Sect.3,the proposed DPR system is also presented in this section.Section 4 describes the implementation design flow of the experimental system on the FPGA platform.In Sect.5,the results of the experiment is presented.Finally,the conclusion is given in Sect.6.

II. DYNAMIC PARTIAL RECONFIGURATION

Partial reconfiguration(PR) is the ability to reconfigure select areas of an FPGA any time after its initial configuration.

A. Two groups of PR

From the functionality of the design, partial reconfiguration can be divided into two groups: dynamic partial reconfiguration(DPR) and static partial reconfiguration[1]. Dynamic partial reconfiguration,which is illustrated in figure 1, also known as active partial reconfiguration, permits to change a part of the device while the rest of an FPGA is still running. In static partial reconfiguration the device is not active during the reconfiguration process. In other words, while the partial data is sent into the FPGA, the rest of the device is stopped and brought up after the configuration is completed, as shown in figure 2.

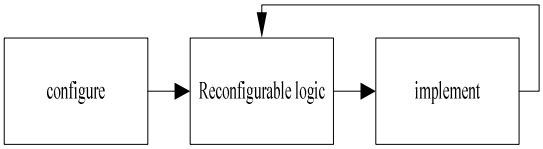

Figure 1. Dynamic partial reconfiguration

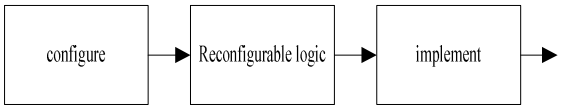

Figure 2. Static partial reconfiguration

Dynamic partial reconfiguration is carried out to allow the FPGA to adapt to changing hardware algorithms, improve fault tolerance and resource utilization, to enhance performance or to reduce power consumption. DPR is especially valuable where devices operate in a mission-critical environment that cannot be disrupted while some subsystems are being redefined.

DPR is not supported on all FPGAs.The Xilinx FPGAs (i.e. Virtex series) are some of the few devices on the market allowing dynamic partial reconfiguration.

B. Several styles of DPR

Xilinx inc. suggests in [2] two basic styles of dynamic reconfiguration on a single FPGA: the Difference-based partial reconfiguration and the module-based partial reconfiguration.

Difference-based partial reconfiguration can be used when a small change is made to the design. It is especially useful in case of changing Look-Up Table (LUT) equations or dedicated memory blocks content. The partial bitstream contains only information about differences between the current design structure (that resides in the FPGA) and the new content of an FPGA. Switching the configuration of a

978-0-7695-3859-4/09 $26.00 © 2009 IEEE
DOI 10.1109/IITA.2009.334

module from one implementation to another is very quick, as the bitstream differences can be extremely smaller than the entire device bitstream.

Module-based partial reconfiguration uses modular design concepts to reconfigure large blocks of logic. The distinct portions of the design to be reconfigured are known as reconfigurable modules. Because specific properties and specific layout criteria must be met with respect to a reconfigurable module, any FPGA design intending to use partial reconfiguration must be planned and laid out with that in mind.

In March of 2006, Xilinx introduced the early access partial reconfiguration(EAPR)design flow along with the introduction of slice based bus macros.

III. EARLY ACCESS PARTIAL RECONfiGURATION

Designers familiar with XAPP290,partial reconfiguration for Virtex-II devices, will find two key differences between the modular-based design PR flow and the EAPR flow[3,6,7].

A. Advantages of EAPR

1) The requirement described in PR for Virtex-II Devices for whole-column PR regions is removed. The EAPR flow now allows for PR regions of any rectangular size in Virtex-4.

The smallest unit of configuration memory that can be read or written is a "frame"[4], which spans the entire height of the device (including I/O blocks) and a fraction of one column in Virtex-II and Virtex-II Pro. So the area for PRM has four slice minimum width and always the full height of the target device.

The latest generation of Virtex FPGAs, the Virtex-4 family, marks a significant change in layout over previous devices. The configuration architecture is still frame-based, but a frame spans 16 rows of configurable logic blocks (CLBs) rather than the full device height[5]. The design rules of new partial reconfiguration can support more flexible than previous one.

2) The bus macro is based on slice not tri-buffer.This enhancement dramatically improves timing performance and simplifies the process of building a PR design.

Slice-based bus macros provide means of locking the routing between PRMs and the base design, making the PRM pin compatible with the base design. As a result all connections between the PRM and the base design must pass through the bus macros, except clock signal. Hardwired slice based bus macro guarantees that routing channels between modules remain unchanged, avoiding contentions inside the FPGA and keeping correct inter-module connection even though each time partial reconfiguration is performed. The partial reconfiguration design method provides various types of bus macro in terms of device, direction and synchronicity. Bus macro must places on the reconfigurable module boundary.

B. Design based on EAPR

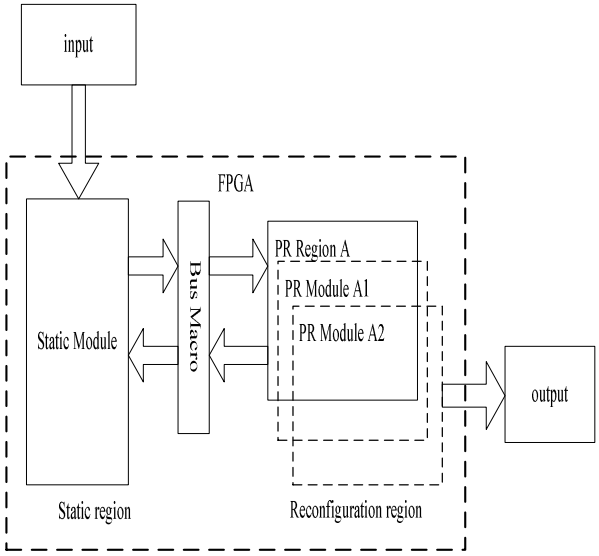

Figure 3. Architecture of the design

Static module is the design remains in operation during the PR process.Partial Reconfiguration Module(PRM) is the design module that can be swapped in and out of the device on the fly,multiple PRMs can be defined for a specific region.Partial Reconfiguration Region(PRR) is the part of the FPGA that is set aside for partial reconfigurable modules.More than one PRRs can be set on the fly.Bus macros(BM) are pre-placed,pre-routed macros.All non-global signals between the PRMs and the static modules must be wired through BMs.

C. Description of the design in this paper

This paper presents a DPR system using the EAPR flow.In this system, a Gray counter and a Johnson counter are reconfigurated dynamically on a Xilinx Virtex4-FX12 FPGA chip. The output part is four leds,which are used to demonstrate the counter.The proposed EAPR-based system architecture is shown in Fig.3.This system consists of a led control module and two partial reconfiguration modules(PRM A1, PRM A2)which are placed on the same partial reconfiguration region. The led control module which enable the leds is a part of the static module.

IV. DESIGN FLOW

Each of the steps of the early access partial reconfiguration design flow is outlined by Figure 4 and described in detail below[3,6].

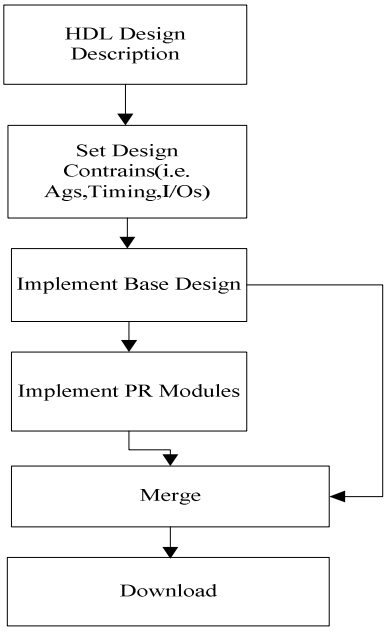

Figure 4. EAPR design flow

A. Hardware Description Language (HDL) design and synthesis

The initial steps in the EAPR design flow are similar to the initial steps in the standard modular design flow.

1) Top-Level design

The top-level module that does not contain any logic only contains I/O instantiations, clock primitives, static module instantiations, partial reconfiguration module instantiations, and signal declarations. In addition, the top-level module must define bus macros.The based design and each PRM must be connected through the bus macro.

2) Base design

The static modules contain logic that will remain constant during reconfiguration. This step is same with traditional HDL design method,but the static modules cannot contain any clock or reset-related primitives.In the proposed design,the control manager is implemented by the based design.

3) PRMs design

Similar to the static modules, the partial reconfiguration modules must also not contain global clock signals, but may use those from the top-level module.When designing multiple reconfigurable modules to utilize the same reconfigurable area, the component name and port configuration of each module must match the reconfigurable module instantiation located in the top-level module. The proposed system has two PRMs which are Gray counter and Johnson counter.

B. Set Design Constraints

After the HDL design description and synthesis ,the next step is to set design constraints.Design constraints include the area group,reconfiguration mode,timing constraint and location constraints.The area group constraint specifies which modules in the top-level module are static and which are reconfigurable.Each module instantiated by the top-level module is assigned to a group. The reconfiguration mode constraint is also only applied to the reconfigurable group, which specifies that the group is reconfigurable. Location constraints must be set for every pin, clocking primitive, and bus macro in the top-level design.Bus macros are located so that they straddle the boundary between the PR region and the base design.

C. Implement Base Design

Before the static modules are implemented, the top-level is translated to ensure that the constraints file has been properly created. The information generated by implementing the base design is used for PRM implementation phase. The base design implementation follows three steps: i.e., translate, map and Place & Route(PAR).

D. Implement PRMs

After the base design is implemented,each PRM must be implemented separately and follows base design implementation steps: translate,map,and PAR.

E. Merge

The final implementation phase is the merge phase.During the merge phase, a complete design is built from the base design and each PRM. In this step,many partial bitstreams for each PRM and initial full bitstreams are created to configure the FPGA.

V. RESULTS

Firstly,the full bitstream is downloaded to initial the device,then,the partial bitstreams are downloaded when the FPGA is running,meanwhile,the leds light along with the choosed partial bitstreams.The full bitstream is 582KB,while the partil bitstream is 12KB for Gray counter and 12KB for Johnson counter.Since the size of the bitstream is directly proportional to the number of resources being configured[8],partial reconfiguration utilizes a smaller bitstream than a full bitstream for the FPGA. The direct benefit is less space needed for storing the necessary configurations for operation.As reconfiguration times are highly dependent on the size and organization of the PRRs, an additional benefit is that the reconfiguration time is shorter.

VI. CONCLUSION

In this paper,we have illustrate the clear advantage of EAPR flow over the XAPP290.Then,we design a experiment based on EAPR to proof that this method can decrease the size of bitstreams obviously.

Recently,Xilinx presented a new tool called PlanAhead to support the EAPR design flow[9,10], this software is the first graphical environment for partial reconfiguration. Using PlanAhead design tools as a platform for partial reconfiguration applications can greatly simplify the complexities of the dynamic operating environment,

allowing a single device to operate in applications that previously required multiple FPGAs.

ACKNOWLEDGMENT

We gratefully acknowledges the financial support given by the Natural Science Foundation of Guangxi Province,China.

REFERENCES

[1] Matthew G.Parris. Optimizing Dynamic Logic Realizations For Partial Reconfiguration Of Field Programmable Gate Arrays. B.S. University of Louisville .2008.

[2] Xilinx,Inc.XAPP290:Two Flows for Partial Reconfiguration:Module Based or Difference Based [EB/OL].http :/ / www. xilinx. com,2004.

[3] Chang-Seok Choi,Hanho Lee. A Self-Reconfigurable Adaptive FIR Filter System on Partial Reconfiguration Platform. IEICE TRANS. INF. & SYST. vol 90, 2006,pp1932-1938.

[4] Virtex-4 Configuration Guide,Xilinx,Inc. http://www.xilinx.com/support/documentation/user_guides/ug071.pdf

[5] P.Sedcole, B.Blodget, T. Becker, J. Anderson, P. Lysaght, Modular dynamic reconfiguration in Virtex FPGAs .Computers and Digital Techniques, IEE Proceedings , London,vol 153,issue:3.,2006,pp157-164.

[6] Xilinx,Inc.UG208:Early Access Partial Reconfiguration User Guide [EB/OL]. http :/ / www. xilinx. Com.2006.

[7] K.G.Nezami, P.W.Stephens, S.D.Walker, Handel-C Implementation of Early-Access Partial-Reconfiguration for Software Defined Radio. Wireless Communications and Networking Conference, 2008. IEEE. ,2008 ,pp1103 – 1108.

[8] E.J.McDonald, Runtime FPGA Partial Reconfiguration. Aerospace Conference, 2008 IEEE,Los Angeles,2008,pp1-7.

[9] N. Dorairaj, E. Shiflet, and M. Goosman, “PlanAhead software as a platform for partial reconfiguration,” Xcell Journal, no. 55, pp. 68–71, 2005.

[10] Xilinx,Inc .Partial Reconfiguration Design with PlanAhead. http:// www. xilinx. Com. 2008.