Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 30 (2012) 234 – 241

Procedia Engineering

www.elsevier.com/locate/procedia

International Conference on Communication Technology and System Design 2011

A Partial Reconfiguration based Approach for Frequency Synthesis using FPGA

Abhishek Tiwari [a]*

Broadcast and Communiaction Group, Centre for Development of Advanced Computing, Thiruvanathpuram, India

Abstract

In order to obtain clocks needed for high speed, high-density designs, dedicated FPGA clock managers are becoming a necessity. The flexibility and programmability they provide is critical to the scope of applications they can support. Recent FPGA architectures, such as Xilinx Virtex Series, allow for partial and dynamic run-time reconfiguration. The FPGA fabric can modify its configuration data at run-time, enabling substitution of specific portions of an implemented hardware design causing the system to be adapted to the needs of the application.

This Paper outlines a new approach of Digital Frequency Synthesis in conjunction with FPGA clock managers. The proposed implementation carries out the frequency synthesis using Dynamic Reconfiguration Port (DRP) of a DCM primitive through the Reconfigurable module in the Fabric. The design highlights Partial Reconfiguration based design approaches which dynamically reconfigures the clock frequency of a DCM according to the variable needs of the running application. Both the output frequency and phase can be precisely and rapidly manipulated on-the Fly. The suggested architecture is first simulated, implemented, and experimentally verified on a Virtex-5 FPGA board.

Keywords : FPGA ;Digital Clock Manager; Dynamic Reconfiguration Port ;PR flow,Reconfiguration module;Virtex-5;SDR

1. Introduction

With the increasing size and complexity of Field Programmable Gate Arrays (FPGA), clock distribution has become an important issue. It has become increasingly difficult to maintain high clock rates as the complexity and size of circuits implemented constantly grow.

* Abhishek Tiwari. Tel.: +91-9895499836
E-mail address: abhishek@cdac.in

1877-7058 © 2011 Published by Elsevier Ltd.
doi:10.1016/j.proeng.2012.01.856

Clock managers incorporated in FPGAs [1-4], [17] helps to solve clock distribution problems, and to improve their flexibility and functionality.Using the frequency multiplication and division features, designers are able to manipulate the clock rates on- and off-chip, and use it for clock alignment among subsystems that use different clock rates.

An example is waveform based applications such as Software Defined Radio (SDR), where the waveform design engineer may need different frequencies at runtime based on the waveform of the selected communication standard [1].FPGAs have specific features that enable SDR implementation. These include: high-bandwidth memories, embedded DSP Blocks, phase-locked loops (PLL), and high-speed interfaces. Also its ability to perform field upgrades and reconfiguration of waveforms has a large benefit to the military community. The framework also complies to low power design and Dynamic Reconfigurability which has increasingly becoming essential for Software Defined Radio (SDR), Cognitive Radio (CR) and other wireless based applications [4].

2. Digital Clock Manager Circuit

FPGAs with their embedded DCMs solve clock management problems in high-speed design, giving the designer an edge over other alternatives. Whether system-clocking needs frequency synthesis, phase alignment, board deskew, adhering to source synchronous interface standards, clock-data synchronization, clock switching, or all of the above, DCMs make designing simpler, saving development and testing costs. It can implement a clock delay locked loop, a digital frequency synthesizer, digital phase shifter, and a digital spread spectrum as shown in Figure 1(a).The clock management system using DCM enables input clock frequency duplication, multiplication and division and phase shifting. On-chip synchronization is achieved by connecting the CLKFB input to a point on the global clock network driven by CLK0 output of the same DCM. The DCM_ADV Primitive shown in figure 1(b) has access to all DCM features and is used for the proposed design. Xilinx-Virtex-5 used in this implementation provide many on chip DCM circuits that offer zero propagation delay, lowest jitter performance[5-6], [17].

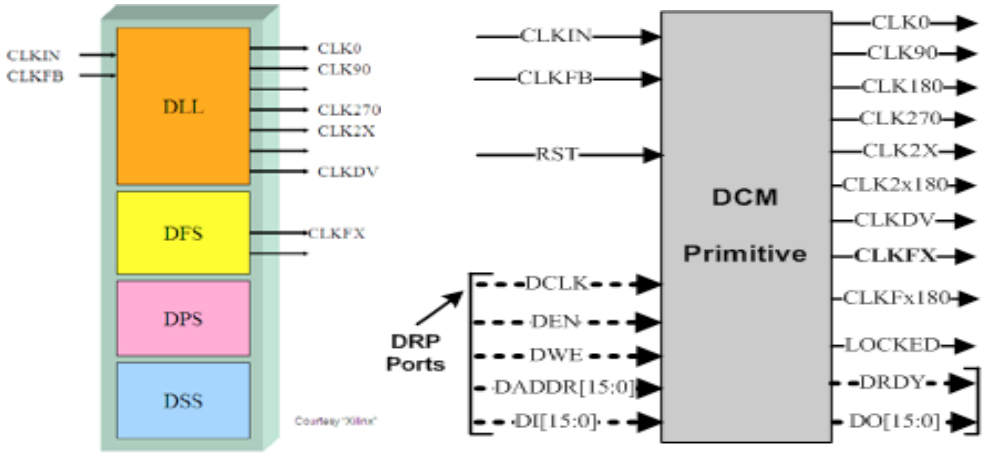

Figure 1 (a) DCM Block; (b) DCM Primitive Block

2.1 Digital Frequency Synthesizer (DFS)

The proposed design exploits DFS feature of the clock manager. In frequency synthesis, the DCM can generate a wide range of output clock frequencies by performing flexible clock multiplication and

division of an input clock [7]. The Frequency Synthesizer provides the most flexible means to multiply, divide, or multiply and divide an input frequency. As shown above in Figure1 (b) the CLKFX and CLKFX180 are outputs in conjunction with the CLKFX_MULTIPLY and CLKFX_DIVIDE attributes provide a frequency synthesizer that can be any multiple or division of CLKIN. CLKFX and CLKIN are in phase every CLKFX_MULTIPLY cycles of CLKFX and every CLKFX_DIVIDE cycles of CLKIN when a feedback is provided to the CLKFB input of the DLL. The frequency of CLKFX is defined by the following equation.

$$CLK\ FX = F_{CLKIN} * CLKFX\_Multiply(M) / CLKFX\_DIVIDE(D) \quad (1)$$

Both the CLKFX and CLKFX180 output can be used simultaneously.CLKFX180 is the CLKFX frequency, shifted 180 with regards to CLKFX. CLKFX and CLKFX180 always have a 50/50 duty cycle.

3. Dynamic Reconfiguration port (DRP)

A method to dynamically change the clock output frequency, of the DCM is through its Dynamic Reconfiguration Port (DRP) [7] as pointed in Figure 1(b). The Dynamic Reconfiguration Ports (DRPs) can update the initial DCM settings without reloading a new bit stream to the FPGA. With Dynamic Reconfiguration, DCM attributes are changeable to select a different phase shift, frequency, or frequency-mode setting from the currently configured settings. The Dynamic reconfiguration port that is an integral part of each functional block which greatly simplifies the process of reconfiguration. This allows the user to dynamically change the clock outputs while the design is running. Frequency, divide values, and multiply values can all be changed dynamically.

4. Partial reconfiguration methodology

Partial reconfiguration (PR) is the ability to reconfigure select areas of an FPGA any time after its initial configuration. Recent evolution in FPGA technology allows the designer to update/reconfigure only a specific part of the internal structure of the FPGA at run-time using a technique known as Partial Dynamic Reconfiguration (PDR) [8-9],[17]. With PDR, it is possible to implement dynamic systems that adapt to run-time needs of the application allowing reuse and reconfiguration of hardware cores (IP blocks). This allows the FPGA to remain operational without comprising the integrity of the applications running on parts of the FPGA that do not undergo reconfiguration. There are two main styles of partial reconfiguration [8]:

- Module Based Partial Reconfiguration: This method is based on Modular Design Flow. The complete design can be divided into modules known as Reconfigurable modules (RM). Reconfigurable modules is swappable logic resides in a region is known as a Reconfigurable Partitions (RP). This method is extensively used in partial reconfiguration of Xilinx devices.
- Difference Based Partial Reconfiguration: This method of partial reconfiguration is accomplished by making a small change to a design, and then by generating a bit stream based on only the differences in the two designs.

5. Proposed Experimental Architecture

The Framework presented in this paper focuses on dynamically reconfiguring the Digital clock managers using Partial reconfiguration technique. In order to implement clock scaling following this

approach, the system must be designed in a specific way. One of the DCM primitive in FPGA had been instantiated in the Static region as it cannot be placed in the Reconfigurable region [3],[4]. The implementations of the same system with different DCM configurations must be generated using Partial Reconfiguration. The logic design solution presented here mainly consists of two components: a DCM module in static region and the DCM controller module in the Reconfigurable region. Figure 2 shows RTL schematics of the overall design components. The proposed architecture for run-time Reconfigurability of DCM had been implemented and verified using Xilinx Virtex-5 board- ML506 with XC5VSX50T- FFG1136 device [10].

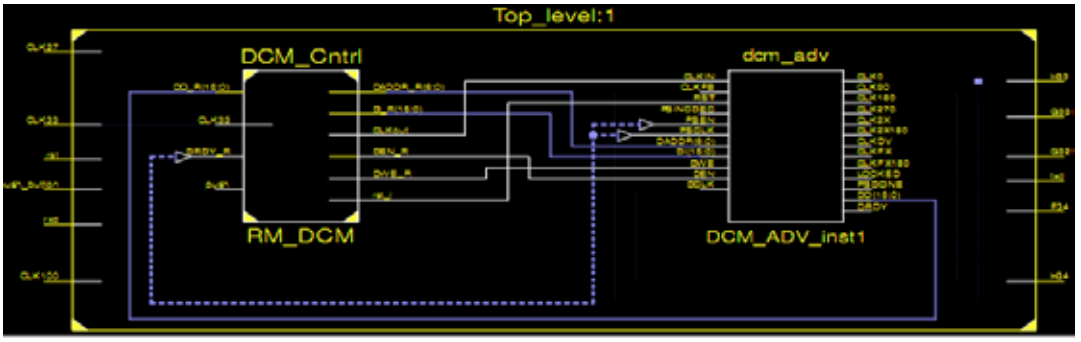

Figure 2 RTL schematic Diagram

The Experimental details are structured as in the following order: Section 5.1 has Module and hierarchy details. Section 5.2 discusses the Design and Simulation results. Section 6 presents Partial Reconfiguration based implementation flow. Section 7 points on the results obtained and its quantitative comparison. Section 8 and 9 categorizes the implementation summary and Future design aspects respectively.

5.1 Module and Hierarchy Details

Figure 3 is a diagram of the hierarchical netlist. Top and DCM are modules in the static region of the design, meaning this logic maintains normal operation while the other modules can be reconfigured [10].

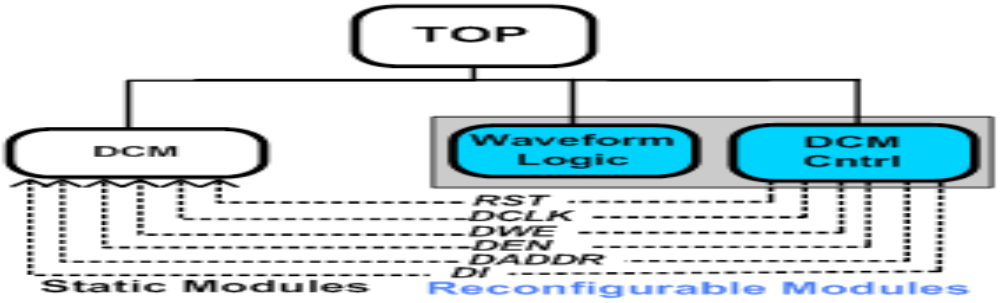

Figure 3 Design Hierarchy Flow

The reconfiguration module DCM_Cntrl controls the reconfiguration of the DCM component placed in the Static Region using DRP port interface. Each block that composed the top design is synthesized separately and then integrated into the Planahead tool from Xilinx. This tool has the particularity to allow

to the designer to realize a modular conception with specific placement constraints, and it allows to generate partial reconfiguration files that are mandatory for partial reconfiguration [9].

5.2 Design and Simulation Efforts

The design in this part combines the two different technologies of the FPGA (a) Dynamic reconfiguration port of DCM and (b) Partial Reconfiguration (PR) flow. The DCM control Logic for reconfigurable Partition has been implemented in VHDL using the State machine based technique for Glitch free and timing accurate operation. To verify the proposed segmented architecture, a behavioral simulation of the proposed DRP control logic was carried out using Xilinx ISE software prior to hardware implementation. A VHDL code describing the circuit is simulated using a Xilinx simulator. Moreover, the Digital clock manager circuit is also validated for design before using it to implement the proposed architecture. The DCM block introduced in Figure 1 is also verified using the available FPGA templates. The design implementation mainly focuses on the Frequency Synthesizer output CLKFX of the DCM.

The multiply and divider values for the clock synthesis can be accomplished in accordance based on the vendors datasheet [7]. To reconfigure the DCM using its DRP port following steps had been performed as part of the simulation result shown in Figure 4.The flow of implementation is as follows:

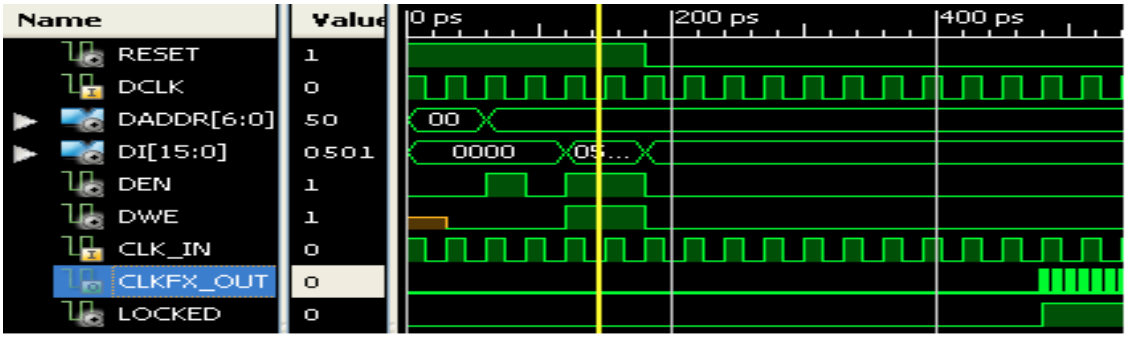

Figure 4 Simulation results of DRP Logical Block

Step 1: The DCM must be held in reset by activating input RESET while changing the M/D values.
Step 2: The rising edge of DCLK signal is the timing reference for all the other port signals.
Step 3: This signal DEN enables read and write port operations. It should only be pulsed for one DCLK cycle.
Step 4: When DWE and DEN is high, it enables a write operation to the port .It should only be pulsed for one DCLK cycle.
Step 5: The value on DADDR bus specifies the individual cell that is written or read on the next cycle of DCLK. The address is presented in the cycle that DEN is active.
Step 6: The value on DI bus is the data that is written to the addressed cell.
Step 7: When LOCKED signal is asserted, indicates that clock is generated and stable.

6. Partial Reconfiguration based Implementation

6.1 Module based Implementation

This proposed novel approach exploits the Partial Reconfiguration property of FPGA. In order to implement clock scaling using this approach, the system must be designed in a specific way. One of the DCM primitive in FPGA had been instantiated in the Static region as it cannot be placed in the Reconfigurable region [12-14]. The implementations of the same system with different DCM configurations must be generated using Partial Reconfiguration.The design is synthesized using Xilinx ISE and floor planning, physical constraints and partial bit files are generated using Plan Ahead tool. Figure 5(a) shows the system sub-divided into static and Reconfigurable partitions based on PR flow [8], [15]. For each Reconfigurable Partitions (RP) multiple bit files will be generated based on the application. The function implemented in Reconfigurable Partitions is modified by downloading one of several partial bit files Frequency_LF.bit, Frequency _HF.bit and Frequency_VHF .bit based on the respective Waveforms application.

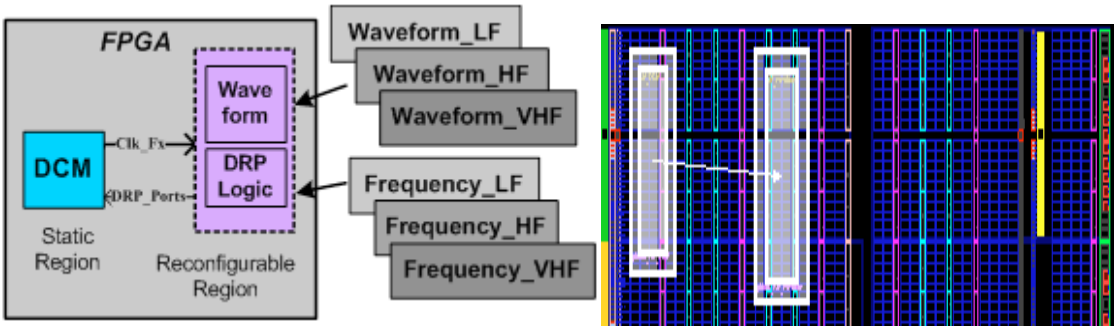

Figure 5 (a) Reconfigurable Partition Blocks; (b) Pblocks of Reconfigurable Partition

Figure 5(b) shows how the physical design reconfigurable blocks (Pblock) has been partitioned in a part of FPGA fabric. The Partial bit files had been generated for each Pblock based on the different configurations [8], [15].

6.2 Difference based Implementation

The Difference Based design flow takes the Full Bit file and updated Native Circuit Description, NCD file as inputs and generates a partial bit streams which only includes the difference between the two designs. The internal clock synthesis registers can be changed using FPGA editor environment and new NCD file has been generated. This design flow is optimal for such designs where the changes are minor and no reconfiguration of complete functions is to be performed. The overview of the routed design can be viewed in FPGA Editor. The Difference Based technique would be more suitable to use since it would extract out the DCM reconfiguration part which would lead to a smaller size of the partial bit streams and very less reconfiguration time. But it is not suitable for a system with many Reconfigurable Modules.

7. Partial Reconfiguration Results

7.1 Observed Reconfiguration Time

A very low configuration time is one of the novel advantages of the proposed implementation flow. For this test system the area of one PRM was found to be 25 KB. Table 1 shows the Partial Reconfiguration Time (PRT) calculated based on the following equation [14]:

$$PRT = PRM\ (Bits) / CCLK(Hz) * BusWidth(Bits) \quad (2)$$

Table 1 Measured Reconfiguration Time

Configuration mode	Max Clock rate	Data width	Configuration Time (PRT)
Select Map	100 Mhz	16 bit	125 usec
JTAG	66 Mhz	1 bit	3030 usec

7.2 Observed Power Reduction

Total Power consumption was measured using XPower Analyzer available with Xilinx ISE Environment. The Fmax for the tested design is 100 MHz. Table 2 compares the measured power dissipation, using the proposed technique and traditional method.

Table 2 Measured Power Consumption at Fmax=100 MHz

On-Chip	Traditional	Proposed
Clocks	2.79	0.24
Logics	0.12	0.00
Signals	0.31	0.00
IOs	68.22	0.00
DCM	68.00	33.49
Quiescent	704.43	703.10
Total	843.87	736.83

A reduction of 107mW of power obtained with the proposed implementation flow. The PR based design and size of the bit file leads to decrease in power consumption.

8. Implementation summary

The DRP Reconfiguration Logic previously discussed has been tested using ML506 Evaluation platform which consists of a Virtex-5 SXT FPGA [10]. However, it can be implemented on any Xilinx FPGA supporting DCM_ADV primitive features. All the logic inside the FPGA was described in generic VHDL code, except for the blocks that are FPGA fabricant dependent, such as the DCM. These blocks were either generated by Xilinx CoreGen tool or instantiating component primitives. The Integrated Software Environment (ISE) is used for design synthesis and simulation. Plan Ahead tool from Xilinx had

been used for functional, post place and route simulation and bit file generation. In order to achieve the high performance required by the timing constraints, FSM based VHDL coding techniques were used in the written code.

9. Future work

The paper validates the partial reconfiguration flow based frequency synthesis using DCM resource but the methodology could be extended to further use of PLL where Frequency, phase, duty cycle and divide values all can be changed dynamically. The future task will be to investigate how dynamic reconfiguration can be exploited for optimized synthesis using FPGA clock networks. The performance Evaluation and the implication of the proposed method is a work in progress.

10. Conclusion

The emergence of new Reconfigurable architectures offers user the opportunity to deploy and implement many exciting applications. The framework here helps in full utilization of the DCM resources already existing on FPGA, which can help in saving the extra hardware needed for the same. However, central to the success of these applications is the design based on hardware and software reuse which is an area of focus in this paper.

References

[1] Steven W. Cox, Joel A. Seely, “Rapid SDR waveform development in FPGAS using DSP builder”, CF-SDR031505-1.0.
[2] Kevin Skey, John Bradley, Karl Wagner, “A Reuse Approach For FPGA-Based SDR Waveforms”, Case No. 06-0796.
[3] Katarina Paulsson,et al., “,Exploitation of Run-Time Partial Reconfiguration for Dynamic Power Management in Xilinx Spartan III-based Systems.”
[4] Altera, FPGA Run-Time Reconfiguration: Two Approaches: A White Paper, WP-01055-1.0, March 2008, ver. 1.0.
[5] Majd Ghazi Batarseh, et al., Window-Masked Segmented Digital ClockManager-FPGA-Based Digital Pulsewidth Modulator Technique, IEEE transactions on power electronics, vol. 24, no. 11, November 2009.
[6] Xilinx, Virtex-5 FPGA User Guide, UG190 (v5.3), www.xilinx.com May 17, 2010, Page 50.
[7] Xilinx, Virtex-5 Configuration Guide UG191, www.xilinx.com, August 20, 2010, Page 106.
[8] Xilinx, Partial Reconfiguration User Guide UG702 (v12.3), October 5, 2010, Page 17& 27&103.
[9] Julien Delorme, et al., “A FPGA partial reconfiguration design approach for cognitive radio based on NoC architecture”, IEEE,2008.
[10] Xilinx, ML505/ML506/ML507 Evaluation Platform, UG347 (v3.0) www.xilinx.com, May 19, 2008.
[11] J. Jones, M. Stettler , “Dynamic Reconfiguration and Incremental Firmware Development in the Xilinx Virtex 5”,2008.
[12] Ross Hymel, Alan D. George , “Evaluating Partial Reconfiguration for Embedded FPGA Applications”, 2004.
[13] P. Sedcole, et al., “Modular dynamic reconfiguration in Virtex FPGAs”, IEE Proceeding on Digit. Tech., Vol. 153, No. 3, May 2006.
[14] Bjorn Osterloh,et al., “Dynamic Partial Reconfiguration in Space Applications”,2009 NASA/ESA Conference on Adaptive Hardware and Systems.
[15] Wang Lie, Wu Feng-yan , “Dynamic partial reconfiguration in FPGAs” ,Third International Symposium on Intelligent Information Technology Application,2009,IEEE.
[16] John Huie, et al., “Synthesizing FPGA Cores For Software-Defined Radio”, CF-SDR031405-1.0.
[17] Ian Brynjolfson and Zeljko Zilic, “FPGA Clock Management for Low Power” , McGill University, 2008.