International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 4, No. 3, November 2015, pp. 201~208
ISSN: 2089-4864

Dynamic Partial Reconfiguration with FIR Filter Application

Noopur Astik, Priti Shahane
Department of Electronics and Telecommunication Engineering, Symbiosis International University, India

Article Info

Article history:

Received May 2, 2015
Revised Aug 10, 2015
Accepted Aug 28, 2015

Keyword:

DPR
FIR filter implementation
System Generator
Hardware co-simulation
Black Box

ABSTRACT

Dynamic partial reconfiguration has evolved as a very prominent state of art for efficient area utilization of *Field Programmable Gate Array* (FPGA) as well as significant reduction in its overall power consumption when properly used to lessen the idle logic on FPGA. It provides desired results even as the computational complexity increases in the field of Digital Signal Processing. This paper explains Dynamic Partial Reconfiguration (DPR) with an example of Finite Impulse response (FIR) filter of order 10. Initially RTL coding for Direct Form FIR structure is written in Verilog in fixed point format for low pass and high pass filter modules using ISE Design suite. Functioning of the both the modules is verified individually through hardware co-simulation on ZYBO (Zynq Board) from Digilent using Black Box from System Generator. Finally dynamic partial reconfigurable FIR filter with low pass and high pass as reconfigurable modules is implemented on ZYBO using Plan Ahead tool. Final comparison of resource utilization with and without DPR is presented.

Corresponding Author:

Noopur Astik,
Department of Electronic and Telecommunication Engineering,
Symbiosis International University,
Symbiosis Institute of Technology (SIT), Lavale, Pune- Maharashtra 412115, India
Email: NoopurAstik@gmail.com or noopur.astik@sitpune.edu.in

1. INTRODUCTION

Evolution of reconfigurable device bridged the gap between the two extremities, general-purpose processor and specialized ASIC while offering several benefits for the implementation of digital circuits in the vast and fast growing field of System Integration.

Modern *Field Programmable Gate Array* available by leading vendors is a Static RAM based reconfigurable device and it therefore outmatches *Application Specific Integrated Circuit* in terms of cost-effectiveness and flexible binding time of the device function [1].

Although Moore's law has successfully sustained and miniaturization of transistor continues since the commencement of Integrated circuit (IC) era, *Field Programmable Gate Array* industry claims for innovations beyond the law. In order to quench the demands of higher bandwidth, capacity and reduction in power consumption, FPGA vendors are looking for alternatives that outpace Moore's law. Partial reconfiguration (PR) feature of FPGA turns out to be one such state of the art in this industry. With the advent of SRAM based FPGA, reconfigurable hardware has gained prominent space between the extremes of general purpose processor and specialized ASIC. SRAM cells provide unlimited reconfiguration and multiple modules operate simultaneously. PR enhances reconfiguration by allowing only specific regions of FPGA to be reprogrammed with new functionality such that the remaining of the device continues to run without any need to reset.

Dynamic Partial reconfiguration (DPR) is practicable design solution useful to many sectors such as aerospace, defense, mobile communication, etc. Apart from the industrial applications it is also utilized to implement various complex algorithms in Digital Signal Processing as well as Digital Image Processing. In order to analyze and comprehend DPR in the field of Digital Signal Processing, it is illustrated here to implement FIR filter since filters are signal conditioners and form a crucial part of signal processing. Although *microprocessors* are the traditional platform for implementing DSP applications until recently

available accelerated hardware. As cost and time to market are some of the primary assets of FPGA, designers therefore select the device for variety of applications including signal processing. And FPGA along with DPR is suitable for every application requiring high parallel processing rather than restricting to clock driven performance.

This paper is organized into 3 sections: - 1) Partial Reconfiguration 2) FIR filter design and hardware co-simulation in system generator 3) Result analysis of implementation of filter modules using DPR.

2. PARTIAL RECONFIGURATION

The concept of partial reconfiguration originated from reconfigurable computing which has been in existence since 1960 when Gerald Estrin proposed the theory of reconfigurable hardware. Partial reconfiguration allows portion of array to change instructions rather than disturbing the functioning of entire array. When this change is implemented at run- time it is termed as dynamic. It thus reduces the reload time and is useful when all functions are not required simultaneously working on the FPGA fabric.

2.1 Approach and Tools

Today almost all FPGA vendors provide support for partial reconfiguration and till now three main approaches have been followed. Different vendors provide PR supporting tools, however the tools and techniques discussed here are Xilinx based unless mentioned.

a) Difference based PR- It is used when a small change is made to the design like modifying a Look-Up Table (LUT) function or memory content. The partial bit-stream contains only information about the differences between the current design structure (that resides in the FPGA) and the new content of an FPGA. The appropriate tool to achieve it is command line tool Bitgen and FPGA editor.[2]
b) Modular PR- The method was introduced to allow designers to work on different modules of the same project in parallel. Eventually it is an approach by Xilinx for partial reconfiguration with many modules working under a top-module. Modules occupy the full height of the device with no inter-modules resources being shared. Bus macros are used in order to facilitate inter module communication. Bus macro is a defined signal path to cross over partial reconfiguration boundary. [3] Synthesis and implementation tools and Xilinx XFLOW tool are used.
c) Early Access Partial Reconfiguration (EAPR) - It is similar to *Modular PR*. The designer sets up a top level design that instantiates global resources, I/O-pins and static and partial reconfigurable modules. It uses slice bus macro rather than tri-state buffer (TBUF). TBUF connect PR region to a common bus but the effectiveness of TBUFs was compromised by their fixed locations [4] and provided less flexibility along with the risk of interference between reconfigurable modules, hence replaced by *slice-based* bus macros. Latter comprise of 2-LUTs per bit signal crossing reconfigurable modules [5] and provide a fixed interface between static and reconfigurable modules in the design. All signals except global signals between PRM and static module are routed through bus macro.

Although recent advancement from Xilinx introduced PlanAhead tool which do not require busmacro instantiation any longer since they are hard macro with two slices needed in the design. Partition pins are the improvised solution over bus macro. It requires single LUT per bit signal and auto inserted at each RM boundary. It is also called as proxy-LUT. [6] The same approach of PlanAhead flow has been used to implement FIR filter.

2.2 Partial Configuration Methiods

FPGA can be configured using several configuration modes and during reconfiguration SelectMap, JTAG or ICAP port is selected depending on the reconfiguration bit file transfer being external or internal.

a) SelectMap mode- It is a parallel mode of external configuration. In this mode an external host is required, such as a microprocessor or microcontroller to load byte wide configuration data into the FPGA device from any non-volatile memory. It is a faster mode of configuration and can be used for Spartan series which lack internal configuration port. [7]
b) Internal Configuration Access Port (ICAP) - ICAP is a subset version of the SelectMAP interface that allows access to the configuration registers of the FPGA. The ICAP enables the self reconfiguration feature of the FPGA with parallel data transfer. ICAP interfaces are used only for partial reconfigurations and not for initial configuration. [8] Partial bitstreams are loaded into the FPGA with the help of an ICAP, which is usually controlled by an embedded processor. Hence reconfiguration time is directly proportional to total reconfigurable region.

c) JTAG mode- It is a simple serial configuration mode. The JTAG interface requires programming cable and the bit file is downloaded using iMPACT. It supports hardware debugging and can program SPI flash. [9]

3. FIR FILTER IMPLEMENTAION

A filter separates a desired signal from any unwanted disturbances and basic filters are frequency selective filters. The design of FIR filter includes desired frequency response and coefficient calculation for the FIR filter. [10] Coefficients of filter are calculated here with Hamming window using FDA tool box in MatLab. Filter specifications for Direct Form FIR low pass (LPF) and high pass filters (HPF) are as follows:-

LPF: - Sampling frequency =22000Hz
Cut-off frequency = 9000Hz
Filter order= 10

HPF: -Sampling frequency =48000Hz
Cut-off frequency = 10800Hz
Filter order= 10

3.1. Fixed Point Format

Real data types are not synthesizable in Verilog. Real numbers are therefore converted to fixed point format in order to write RTL Verilog code for Direct Form FIR filter structure without using any IP cores for data type conversion. The fixed point format employed is Q[QI].[QF] format where QI is number of integer bits and QF is number of fractional bits [11]. Range and resolution for signed integer is calculated as follows:

$$-2^{QI-1} \leq \alpha \leq (2^{QI-1} - 2^{-QF}) | \epsilon = 2^{-QF} \quad (3.1)$$

Coefficients are represented in Q1.7 format and range from -1 to 0.992185 with the resolution of 0.0078125 when calculated using equation 3.1. 8-bit input data is assumed to be in the form of Q5.3 and thus in range of -16 to 15.875 with the resolution of 0.125.

Multiplication of Q1.7 and Q5.3 format results in Q6.10 format thus 16-bit output is taken.

3.2. RTL using Verilog Code

RTL for Direct Form FIR low pass module and high pass module is generated using Xilinx ISE 14.6 for implementation on ZYBO xc7z010-1clg400. Data input is 8-bit and data output is 16-bit. Additional output net 'ld' shown in figure 1 is connected to LED and is off when low pass is implemented and ON in case of high pass to mark the correct working of filters during dynamic swap of modules. Thus total IOBs are 27 for each filter module

Figure 1. In-out schematic of low pass & high pass FIR filter modules

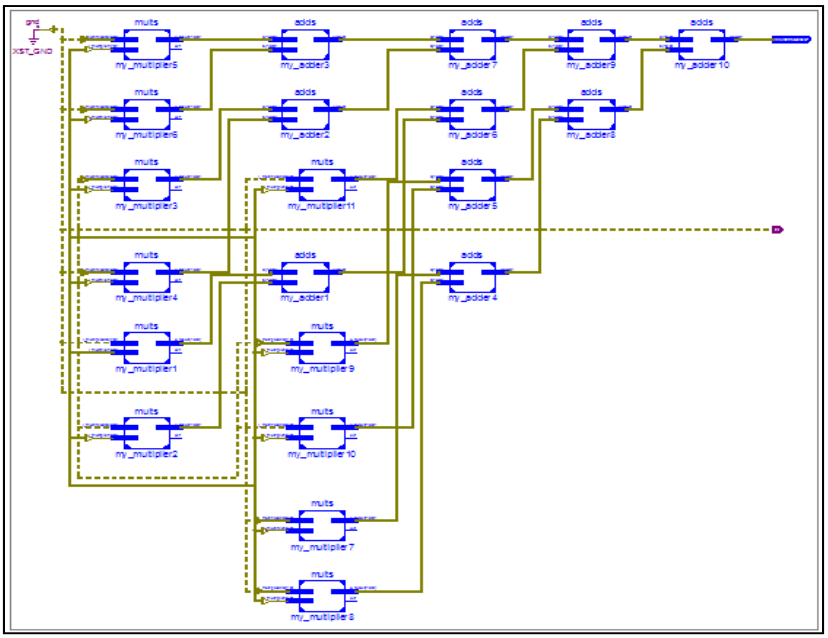


Figure 2. RTL schematic of Direct Form FIR filter

	Lowpass			Highpass		
Logic Utilization	Used	Available	Utilization	Used	Available	Utilization
Number of Slice Registers	113	35200	0%	101	35200	0%
Number of Slice LUTs	1966	17600	11%	1912	17600	10%
Number of fully used LUT-FF pairs	48	2031	2%	43	1970	2%
Number of bonded IOBs	27	100	27%	27	100	27%
Number of BUFG/BUFGCTRLs	1	32	3%	1	32	3%

Figure 3. ISE 14.6 design summary of both the individual filter modules (without DPR)

3.3. Filter Verfication Using System Generator

System Generator is a DSP design tool from Xilinx that enables the use of the MathWorks model-based Simulink design environment for FPGA design. [12] System generator provides Xilinx blocksets in Simulink to generate test vectors.

The FIR filter Verilog modules are verified using Black Box from the Xilinx blockset. Black Box block provides a way to incorporate VHDL or Verilog code into the Simulink model. [13] While dragging the Black Box into the Simulink model, it prompts to assign necessary HDL file to it. All necessary inputs and outputs are defined through Gateway In and Gateway Out FPGA boundary blocks as shown in figure 4 below.

Black Box uses ISE simulator mode and below model shown is a representation for both low pass and high pass where respective Verilog file is assigned to Black Box. Figure 4 also includes hardware co-simulation block of ZYBO through JTAG download. During Simulink run, results are compiled in hardware and displayed back on Simulink sink resource (scope). Input signal is chirp signal with target frequency 10^7Hz and initial 100Hz with target time 0.0001. Clock period is $1/(5 * 10^6)$. Simulation is run for 0.01s. Since simulation uses high frequencies all parameters are set for better visual scope graph.

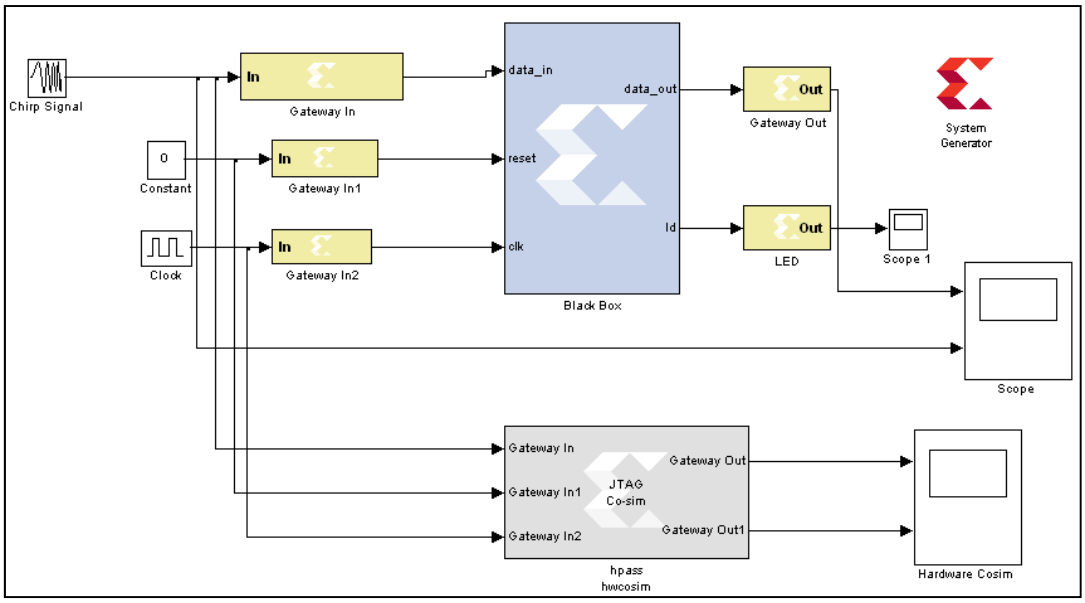


Figure 4. Design verification of FIR filter with order 10 using Black Box in SysGen

Time domain output of scope is shown in below plots for low pass as well high pass. However output of Scope1 as well as from Gateway Out1 of hwcosim block is high for HPF and low for LPF since it is the output given to LED at the time of DPR implementation.

Figure 5 shows chirp signal input and LPF and HPF outputs observed on Scope block.

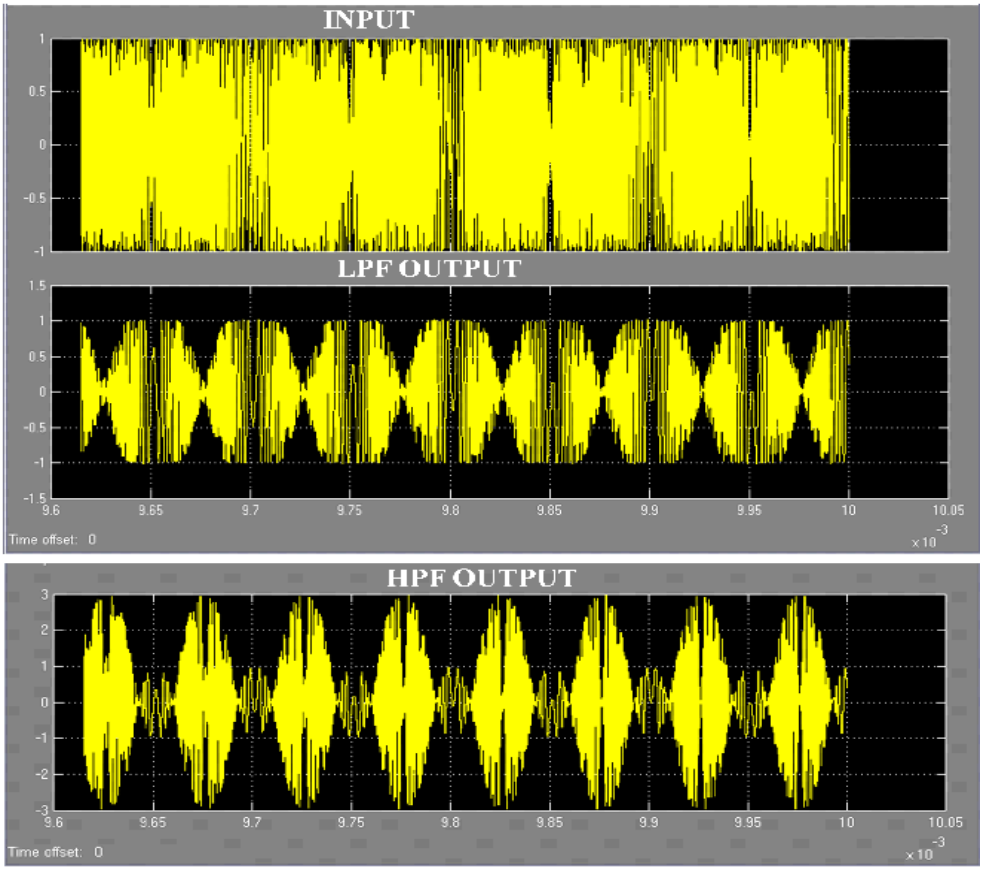


Figure 5. Scope output for Low pass module and High pass module

Figure 6 gives the compact output of hardware co-simulation performed on ZYBO for low pass and high pass modules. The Gateway Out is filtered signal and Gateway Out1 is again LED output which is high when HPF is running and low while LPF is running. Below hardware co-simulation results are found similar to Simulink simulation output shown in figure 5.

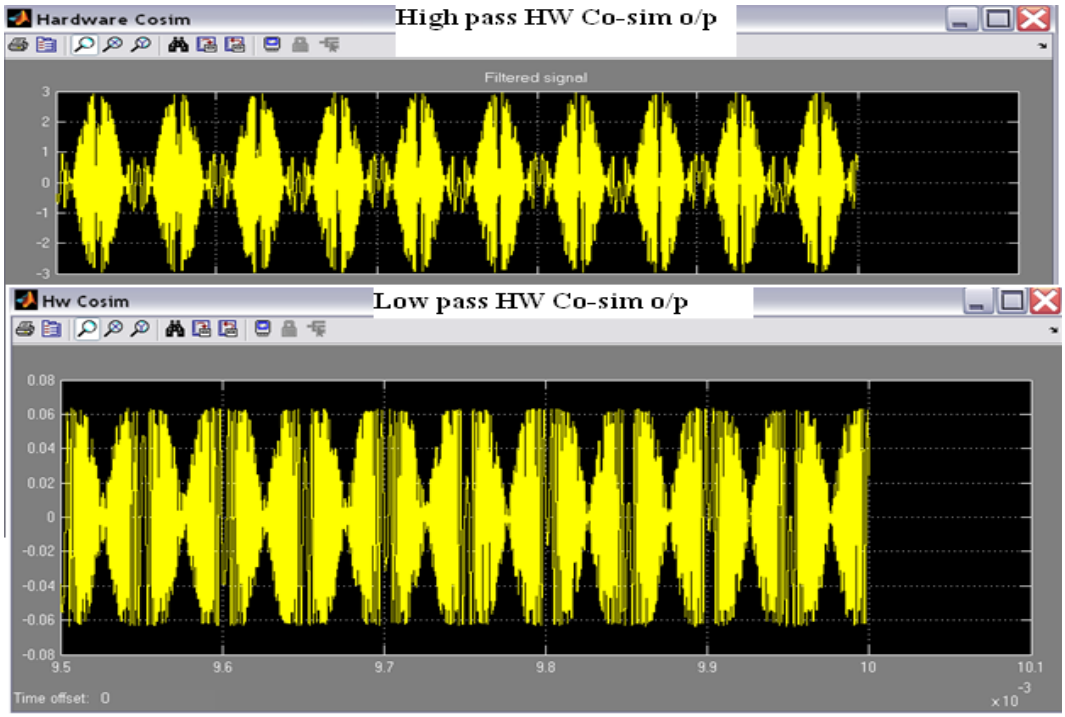


Figure 6. Hardware Cosim Output from Gateway Out using ZYBO

3.4. FIR Filter Implementation Using Dynamic Partial Reconfiguration in PlanAhead Tool

PlanAhead tool from Xilinx provides access to partial reconfiguration (PR) design from Hardware Description Language (HDL) synthesis. It helps create full and partial configuration bit files by allowing designer to define reconfigurable modules. [14] Figure 7 is the block diagram depicting reconfigurable modules (RM) of FIR filter.

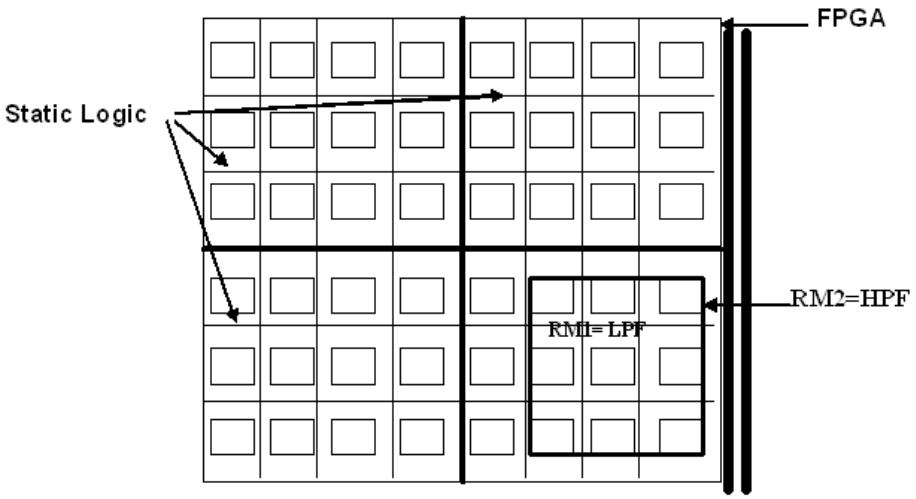
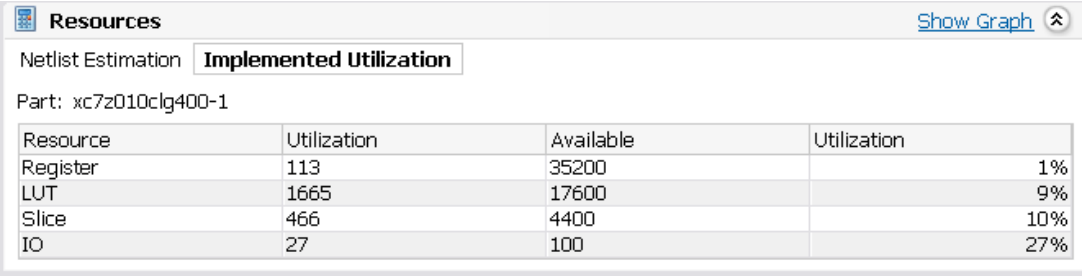


Figure 7. Block Diagram for DPR of FIR filter

For FIR filter two reconfigurable modules are created: - LPF and HPF with a separate top module in Verilog that contains only IO instantiations, clock primitives and PRM instantiations along with the necessary user constraints defined under UCF file. In brief, PlanAhead flow for DPR begins with HDL description, design constraints, implementation of base design and PR modules, creating reconfigurable partitions, floorplanning, generating bit file and finally downloading it to the FPGA board. [14]



Resource	Utilization	Available	Utilization
Register	113	35200	1%
LUT	1665	17600	9%
Slice	466	4400	10%
IO	27	100	27%

Figure 8. Combined resource usage of FIR filter implementation using DPR

3.5. POWER ANALYSIS

Xpower Analysis tool (XPA) is used for the analysis of power consumed by individual low pass and high pass modules and to analyze power consumption in DPR project of filters. Table 1 is a comparison of dynamic and static power estimation by XPA tool for individual LPF and HPF filter design as well combined design of filters without DPR and with DPR.

Table 1. Power Analysis

Filter Design \ Supply Power	Total (W)	Dynamic (W)	Quiescent (W)
LPF only	0.103	0.00	0.103
HPF only	0.103	0.00	0.103
Combined LPF & HPF without DPR	0.161	0.061	0.100
filter with DPR (configuration 1: LPF)	0.152	0.053	0.100
Filter with DPR (configuration 2:HPF)	0.149	0.049	0.100

4. RESULT ANALYSIS

FIR filter modules for the filter order of 10 are verified with and without dynamic partial reconfiguration in the above sections. In Table 2 with collated resource utilization data, it is observed that total IOBs required to implement both the modules simultaneously are 44 (excluding the common clock & reset signals) and certainly more than 1665 LUTs required under DPR project.

Table 2. Resource utilization

Resource	Low pass filter only	High pass filter only	LPF+ HPF without DPR	LPF+HPF using DPR
Slice registers	113	101	188	113
Slice LUTs	1966	1912	3883	1665
IOBs	27	27	44	27

The efficiency in case of DSP applications can be increased further with utilization of DSP blocks which are not used at present.

5. CONCLUSION

A precise study of dynamic partial reconfiguration is presented covering its significant aspects. The design statistics of a simple FIR filter application with filter order of 10 and remaining specification chosen for the demonstration purpose do prove that dynamic partial reconfiguration can function as powerful state of

art when extended to higher order filters with higher sampling rates. When exercised with suitable applications where not all functionalities are required at a time and such that they can time share resources, design may then be optimized using DPR with proper knowledge of reconfiguration overhead and frequency of switching required in the application. The reconfiguration can further be extended to self reconfiguration by using ICAP controller or using an external processor. However in general sense, selection of FPGA, ASIC or GPP is purely based on application and is linked to major parameters like costs, tool availability and performance.

REFERENCES

[1] Andr´e DeHon, "Reconfigurable Architectures for General-Purpose Computing", research at *the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology*, 1996
[2] Emi Ato, Difference Based Partial Reconfiguration, *Xilinx Inc. XAP290*
[3] Partial Reconfiguration, Development System Reference Guide, *Xilinx Inc.*
[4] Patrick L., Brandon B., Jeff M., "Enhanced Architectures, Design Methodologies And Cad Tools For Dynamic Reconfiguration Of Xilinx FPGAs", *IEEE*, 2006
[5] Dirk Koch,Jim Torresen and Christian Beckhoff, "Zero Logic Overhead Integration of Partially Reconfigurable Module." University of Oslo
[6] Partial Reconfiguration User Guide, *Xilinx Inc. UG702*,2010
[7] Mike Peattie, "Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode,"*Xilinx Inc. XAPP502*
[8] "Virtex Series Configuration Architecture User Guide,"*Xilinx Inc. XAPP151*
[9] "7 series FPGAs Configuration" user guide, *Xilinx Inc.UG470,* 2013
[10] Steven W. Smith, Digital Signal Processing- *A practical guide for Engineers and Scientists*
[11] Oberstar Erick L.," Fixed Point Representation & Fractional Math", August 2007
[12] "System Generator for DSP," Getting started guide, *Xilinx Inc.UG639*
[13] System Generator for DSP," User guide, *Xilinx Inc.UG640*, 2012
[14] Partial Reconfiguration, PlanAhead Design Tool, *Xilinx Inc. UG743*, 2012