DYNAMIC PARTIAL RECONFIGURATION TECHNIQUES FOR SOFTWARE DEFINED RADIO HARDWARE IMPLEMENTATION

By

Ahmed Kamaleldin Ahmed Elsayed

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
In
Electronics and Communications Engineering

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2017

DYNAMIC PARTIAL RECONFIGURATION TECHNIQUES FOR SOFTWARE DEFINED RADIO HARDWARE IMPLEMENTATION

By
Ahmed Kamaleldin Ahmed Elsayed

A Thesis Submitted to the
Faculty of Engineering at Cairo University
In Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
In
Electronics and Communications Engineering

Under the Supervision of

Prof. Dr. Ahmed Farouk Shalash

Professor of Electronics and Communications
Department of Electronics and Communications Engineering
Faculty of Engineering, Cairo University

Dr. Hassan Mostafa Hassan

Assistant Professor of Nanoelectronics, Bioelectronics and Optoelectronics
Department of Electronics and Communications Engineering
Faculty of Engineering, Cairo University

Dr. Abdullfattah Mouhamed Obeid

Associate Professor
National Center for Electronics and Communications Research
King Abdulaziz City for Science and Technology, Saudi Arabia

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2017

DYNAMIC PARTIAL RECONFIGURATION TECHNIQUES FOR SOFTWARE DEFINED RADIO HARDWARE IMPLEMENTATION

By
Ahmed Kamaleldin Ahmed Elsayed

A Thesis Submitted to the
Faculty of Engineering at Cairo University
In Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
In
Electronics and Communications Engineering

Approved by the
Examining Committee:

_____
Prof. Dr. Ahmed Farouk Shalash, Thesis Main Advisor

_____
Dr. Hassan Mostafa Hassan, Member

_____
Dr. Abdulfattah Mouhamed Obeid, Member

_____
Prof. Dr. Amin Mohamed Nassar, Internal Examiner

_____
Prof. Dr. El-Sayed Mostafa Saad, External Examiner
(Electronics and Communications Engineering, Helwan University)

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2017

Engineer's Name: Ahmed Kamaleldin Ahmed Elsayed
Date of Birth: 20/05/1990
Nationality: Egyptian
E-mail: Ah.kamal.ahmed@gmail.com
Phone: 01026477642
Address: Elsheikh Zayed City – Giza - Egypt

Registration Date: 01/10/2012
Awarding Date:

Degree: Master of Science
Department: Electronics and Communications Engineering

Supervisors:

Prof. Ahmed Farouk Shalash
Dr. Hassan Mostafa Hassan
Dr. Abdulfattah Mouhamed Obeid

Examiners:

Prof. Dr. Ahmed Farouk Shalash (Thesis main advisor)
Dr. Hassan Mostafa Hassan (Member)
Dr. Abdulfattah Mouhamed Obeid (Member)
Porf. Dr. Amin Mohamed Nassar (Internal examiner)
Porf. Dr. El-Sayed Mostafa Saad (External examiner – Electronics and Communications Engineering, Helwan University)

Title of Thesis:

Dynamic Partial Reconfiguration Techniques for Software Defined Radio Hardware Implementations.

Key Words:
Dynamic Partial Reconfiguration (DPR); Software Defined Radio (SDR); Field Programmable Gate Arrays (FPGA); Reconfigurable Systems.

Summary:

Acknowledgments

This thesis submitted to the faculty of engineering at Cairo University in partial fulfillment of the requirements for the degree of Master of Science in electronics and electrical communications, in cooperation with Opto-Nano-Electronic (ONE) lab headed by Dr.Hassan Mostafa, Cairo University Faculty of Engineering Egypt. ONE Lab is a research lab that is funded by the Egyptian Ministry of Communication and Information Technology, ITIDA, and NTRA.

Firstly, I heartily express my thanks and gratitude to Prof.Dr Ahmed Shalash, Dr.Hassan Mostafa, and Dr. Abdulfattah Obeid (KACST) my master thesis supervisors for their supervision, and for providing great advice and support at every step during this master thesis preparation.

I would also like to thank Eng: Ahmed Sadek (Si-Vision), Islam Osama (Mentor Graphics), Khaled Essam (GUC), and Sherif Hosney (Mentor Graphics) for providing the opportunity to work together on the project of Next Generation FPGA and Dynamic Partial Reconfiguration systems implementation and for the pleasant working environment and for the nice moments together.

Last but absolutely not least, I wish to extend my deepest and most sincere thanks and gratitude to my family for their support throughout my study years.

June 2017, Egypt
Ahmed Kamaleldin

Table of Contents

List of Tables

List of Figures

Nomenclature

Abbreviation	Description
3G	Third Mobile Generation.
ADC	Analog to Digital Converter.
ASIC	Application Specific Integrated Circuit
AXI	Advanced eXtensible Interface
BPSK	Binary Phase Shift Key
BRAM	Block Random Access Memory
CLB	Configurable Logic Block
CR	Cognitive Radio
CRC	Cyclic Redundancy Check
DAC	Digital to Analog Converter
DDR	Double Data Rate
DMA	Direct Memory Access
DPR	Dynamic Partial Reconfiguration
DSP	Digital Signal Processing
FEC	Forward Error Correction
FIFO	First Input First Output
FPGA	Field Programmable Gate Arrays
FSM	Finite State Machine
GPP	General Purpose Processor
HDL	Hardware Description Language
ICAP	Internal Configuration Access Port
IFFT	Inverse Fast Fourier Transform

ILA	Integrated Logic Analyzer
JTAG	Joint test Action Group
LTE	Long Term Evolution
LUT	Look Up Table
OFDM	Orthogonal Frequency Division Multiplexing
PC	Personal Computer
PCAP	Processor Configuration Access Port
PL	Programmable Logic
PLL	Phase Locked Loop
PR	Partial Reconfiguration
PRC	Partial Reconfiguration Controller
PS	Processing System
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Key
RM	Reconfigurable Module
RR	Reconfigurable Region
RTL	Register Transfer Level
Rx/Tx	Receiver/Transmitter
SC-FDMA	Single carrier-Frequency Division Multiple Access
SDR	Software Defined Radio
SoC	System on Chip
TTI	Transmission Time Interval
UART	Universal Asynchronous Receiver/Transmitter
WCDMA	Wideband Code Division Multiple Access
XST	Xilinx Synthesis Technology

Abstract

The Software Defined Radio is a communication system designed so that the physical layer can be implemented by means of software (software-defined). The Next generation wireless systems are software defined / hardware reconfigurable systems characterized by supporting multiple wireless standards with a growing needs for high data rate, reconfigurable hardware optimized platforms and low power consumption for longer battery operation time and to accommodate any new communication standards.

FPGA is a promising reconfigurable hardware platform to implement the software defined radio by using the Dynamic Partial Reconfiguration techniques to achieve the requirements of fast reconfigurability, hardware optimization, and low power consumption.

In this thesis, dynamic partial reconfiguration implementation of a software defined radio that supports multiple communication standards is designed and tested. Different partial reconfiguration implementation techniques are used to reduce the switching time between different communication standards and protocols. For hardware optimization and low power consumption, an automated partitioning algorithm is proposed and applied to find the optimum resources allocation of the different reconfigurable parts of the software-defined system over the FPGA. Therefore, this thesis offers a novel reconfigurable hardware implementation of software defined radio using dynamic partial reconfiguration and is compared to other work in the literature.

Chapter 1 : Introduction

The last two decades witnessed an exponential growth in the connectivity between people as a consequence of the emerging of wireless communication systems. There has been tackling efforts from researchers and industrial communities in developing and upgrading of the wireless communication systems to satisfy the increasing number of users and nodes, and the high data rate required to enable a various form of transmission data among people e.g. voice, video,... Consequently, modern wireless communication systems are characterized by supporting a number of different wireless communication standards that are proportional to the number of services and wireless technology generations. Different wireless communications standards are fixed wireless standards that use fixed portions (fixed bandwidth) of the radio frequency spectrum. The wireless standards are nod used simultaneously at the same time that leads to inefficient utilization of the radio frequency spectrum

To solve the problem of inefficient utilization of radio spectrum, the concept of Cognitive Radio (CR) was proposed by Joseph Mitola [1-2]. CR dynamically configures the wireless terminal to utilize the available radio spectrum that is not used by other terminals to avoid the possibility of interfering between wireless channels. CR is able to adapt its parameters such as transmit power, coding rate, bandwidth and center frequency to enhance the utilization of radio spectrum in a changing environment. CR is an efficient approach to managing the radio frequency in a contested environment and it can be developed and implemented using Software Defined Radio (SDR) technique [1-2].

Modern wireless terminal is a Multi-Standard Communication System (MSCS) that operates in multiple frequency bands with different wireless standards. Two major challenges are facing the implementation of MSCS [4]: 1) the radio spectrum utilization and 2) the hardware utilization. For hardware utilization, each wireless standard has its own hardware resources, therefore a high density of hardware physical resources are required and will be augmented proportionally with the number of wireless standards supported by the MSCS, high power consumption and low battery life. Upgrading the MSCS with new wireless technologies requires the modification and redesigning of the hardware platform to be able to accommodate new wireless technologies.

These two major problems, the inefficient use of radio spectrum and the waste in hardware physical resources lead the researchers and the industrial communities to find a new way or a new paradigm for hardware implementation reusing the same set of physical resources to operate the new wireless technology with the old technology. This concept is defined by Reconfigurable Computing (RC) architectures [5] where the reconfigurable hardware platform offers the performance and energy efficiency of hardware and the flexibility and reprogrammability of software. RC performing computations with spatially programmable architectures, like Field Programmable Gate Arrays (FPGAs). Using RC architectures in the hardware implementation of MSCS provides rapid configuration or handover between different communication standards with benefits of minimization of hardware resources utilization, low power consumption and the ability of system scalability by upgrading the MSCS with new wireless standards.

Figure 1.1 shows the main blocks of a simple communication system. The communication system is composed of: 1) a Digital Signal Processing (DSP) unit is responsible for signal receiving or transmitting over the channel. Transmitter and receiver for a wireless standard are considered as DSP blocks, in this thesis the implementation of SDR system is concentrated on the implementation of DSP blocks of the system. 2) DAC/ADC blocks, digital to analog and analog to digital converters are used to convert the signal from the digital domain to the analog domain or vice versa according to the Tx/Rx direction of the signal. 3) RF Front End block contains the Low Noise Amplifier (LNA), filters and Power Amplifiers (PA), this block is out of the scope of this thesis.

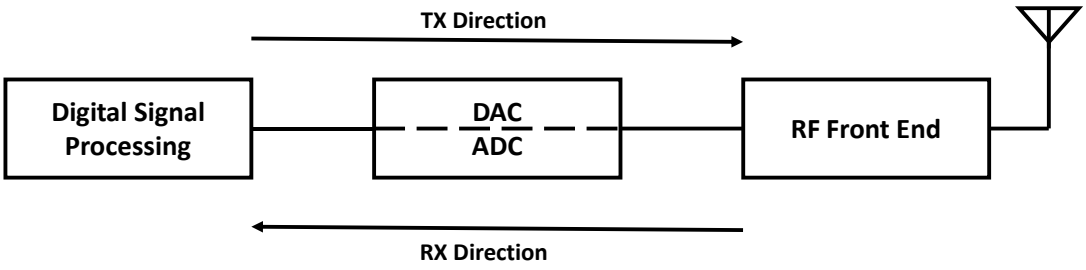


Figure 1.1: Simple Communication System

SDR proposes a paradigm shift from fixed dedicated digital hardware radio platforms to a reprogrammable digital hardware platform [3]. Wireless standards are implemented via software updates, hence modern MSCS is software defined over reconfigurable hardware platform. The SDR system is generally based on the regular communication system shown in Figure 1.1. The motivation of the SDR system came from the using of the same physical layer blocks that have the same functionality with different wireless communication standards (GSM, UMTS, LTE, etc…). Figure 1.2 shows a typical implementation of a SDR system. The system consists of a memory unit to store the different waveforms that represent different information that is sent from the source terminal to the destination terminal. Each waveform represent a unique wireless standard supported by the SDR system. The DSP unit contains the reconfigurable physical layer blocks which is reconfigured by a controller or a set of control unites to sense the radio spectrum and load the selected waveform from the memory storage.

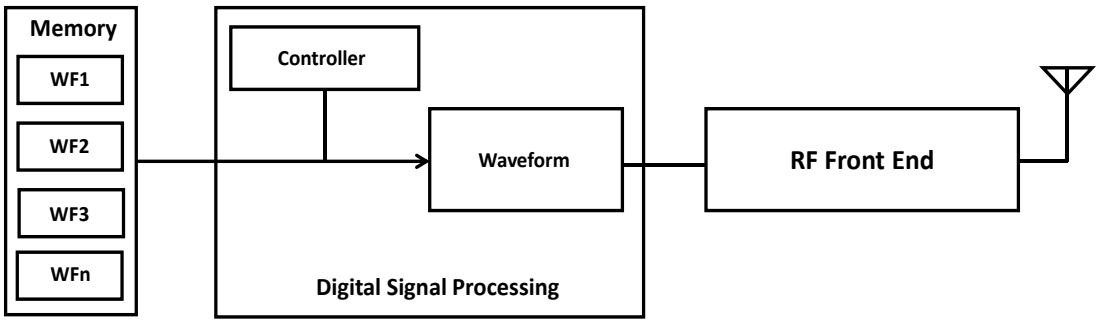


Figure 1.2: Typical SDR System

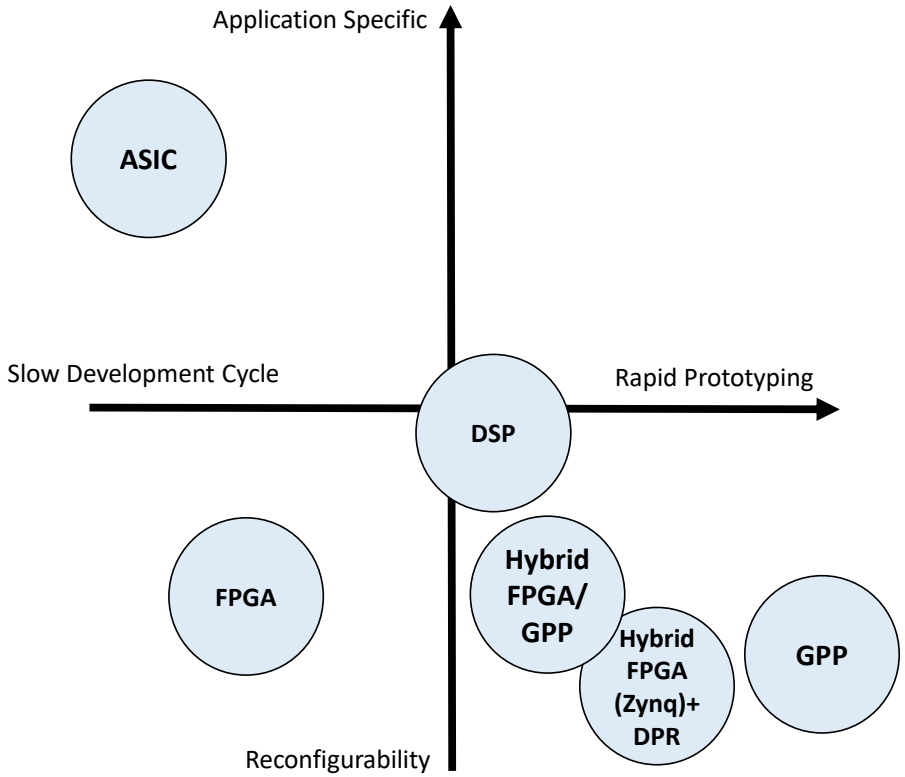


Figure 1.3: Trade-off between Different SDR Hardware Platforms [3]

The digital processing part of SDR system shown in Figure1.2 can be implemented on different hardware platforms such as An Application-Specific Integrated Circuit (ASIC), General Purpose Processors (GPP), DSP and FPGAs. Next generation of SDR system or generally the MSCS terminals need a hardware platforms that is characterized by [3]: 1) hardware reconfigurability for low cost adaptation with the new wireless technologies 2) Rapid prototyping and real time experimental testing of physical layers blocks 3) Fast development cycle or short design cycle 4) Adopt Hardware/Software co-design approach to implement a part of the SDR design by means of hardware and a part by means of software routines for better performance and for optimum hardware resources utilization. Figure 1.3 shows the trade-off between reconfigurability and design time for various hardware platforms suitable for the hardware implementation of SDR.

In the recent years, The FPGAs capabilities are developed and enhanced to be more flexible and runtime reconfigurable [5] by introducing the concept of Dynamic Partial Reconfiguration (DPR). DPR allows the FPGA to be reconfigured during runtime by the reconfiguration of a specific part on the FPGA without shutting down the rest of FPGA. DPR pushes the FPGAs to become a promise reconfigurable hardware platform with a high level of flexibility that allows it to be used as the target hardware platform for the implementation of SDR. As shown in Figure 3.1 applying DPR on the FPGA platform increase the reconfigurabilty of the FPGA to be more reconfigurable than traditional software programmable platforms like the DSP and GPPs. DPR offers the benefits of low power consumption and the efficient hardware resources utilization for the SDR system. In this thesis, a novel hardware reconfigurable implementation for SDR will be demonstrated targeting a hybrid FPGA (Xilinx Zynq) that support the DPR technology feature.

1.1. Problem Domain and Context

This thesis focuses on the dynamic partial reconfiguration (DPR) implementation of a software-defined radio system that supports multiple wireless communication standards and able to accommodate next generation of wireless standards without the need to change the hardware platform. In recent years, the using of adaptive reconfigurable hardware platforms e.g. (FPGA+DPR) have been increased to overcome the limitation of fixed hardware platforms like GPPs and DSPs in the implementation of SDR physical layer blocks [8-12]. The limitations are the increasing value of power consumption and the largest silicon area resulting from the growing needs for high data rate and large number of wireless standards that should be supported by the mobile terminal.

The development of a reconfigurable hardware platform for SDR system requires a fast switching time between wireless standards, in other words reducing the time of handover between different wireless standard is an essential target. Also, resources allocation and reconfigurable partitions scheme of the system have a direct impact on the total power consumption and the total silicon area of the reconfigurable hardware platform [7].

Therefore, DPR implementation for SDR system should take into account the reduction of time of reconfiguration for fast handover between wireless standard and a cost effective partitioning scheme to reduce the power consumption and the efficient use of hardware resources.

1.2. Thesis Objectives

This thesis explores the techniques of DPR hardware implementation for SDR systems on FPGA platform. The main objectives of this thesis are:

1. Demonstrate the dynamic partial reconfiguration implementation techniques and determine the design metrics that influence the performance of the implementation.

2. Develop techniques and design-time optimization architectures to reduce the reconfiguration time of the DPR process using the hybrid Xilinx Zynq FPGA.

3. Provide essential design guidelines for the DPR-SDR designers to choose the suitable DPR implementation techniques for their system requirements.

4. Implementation of high-speed DPR multi-standard convolutional encoder.

5. Develop techniques and tools to automate the partitioning step required by the DPR implementation flow to find the optimum partitioning scheme of a specific design to enhance the performance of the DPR implementation.

6. Full implementation of a DPR SDR system that supports multiple wireless standards with a high-speed reconfiguration time, low power consumption, and optimized resources cost.

1.3. Organization of the Thesis

The thesis presents a DPR implementation techniques to design a reconfigurable hardware platform for SDR system. The thesis is organized as follows.

Chapter 2 presents the high-speed DPR implementation for SDR system. In this chapter a brief overview about the Xilinx Zynq hybrid FPGA is introduced, the internal architecture of Xilinx Zynq SoC is presented including the ARM Cortex A9 hardcore processor and the Xilinx 7 series programmable logic part. A detailed description of the dynamic partial reconfiguration technique and how this technique is provided by Xilinx Zynq FPGA. Also, four types of partial reconfiguration controllers are presented and evaluated to achieve a high-speed DPR. Finally, a multi-standard convolutional encoder is used as a case study in order to evaluate the impact of the four PR controllers on the DPR implementation of SDR system.

Chapter 3 presents a cost-effective automatic partitioning scheme for DPR implementation. In this chapter, an overview of the trivial DPR Xilinx design flow is presented and the drawbacks of the trivial flow in the step of design partitioning and resources allocation are clarified. A modified DPR design flow is proposed to increase the efficiency of partitioning step and resources allocation by applying a modified open source automatic partitioning algorithm to find a set of sub-optimal solutions for low-cost partitioning schemes. The partitioning algorithm generates a number of sub-optimal solutions that are evaluated according to best DPR implementation performance based on minimum reconfiguration time and minimum partitions area.

Chapter 4 presents a novel DPR hardware implementation for SDR system. The proposed reconfigurable SDR system supports three wireless standards (3G, WIFI, and LTE). A full transmitter communication chain is adopted as SDR system. Two different DPR implementations are proposed for the reconfigurable hardware platform. First DPR implementation is based on a single partition region where all the communication blocks of the active standard are allocated in a single partition, and in the case of switching from a wireless standard to another, the full partition should be reconfigured. The second implementation is a multi-partitions region DPR implementation based on finding an optimum set of partition regions using partitioning algorithm proposed in Chapter 3. In second DPR implementation switching between wireless standards does not require the reconfiguration of all defined partition regions in the system. Comparisons between the two implementations are presented in this Chapter based on the total area utilized by static and dynamic parts of the system, the time of reconfiguration or the timing required for handover and the total power consumption of the system. Also, an efficient test environment on Xilinx Zynq FPGA is developed to test the functionality and measure the performance of the two proposed DPR implementations

Finally, Chapter 5 contains the summary of achievements of the thesis. In addition, Chapter 5 provides potential directions for future work.

Chapter 2 : High-Speed Dynamic Partial Reconfiguration Implementation For SDR Systems

2.1. Introduction

Modern Wireless communication systems support multiple wireless standards using an adaptive reconfigurable wireless terminal that adopts dynamic communication chains based on the SDR concept [2]. Microprocessors are often used for SDR implementations to provide the required flexibility by a set of software routines. Obviously, microprocessors are not a suitable hardware platform for the high data rate and low power constraints required by the baseband signal processing [1, 4]. Recently FPGA runtime Dynamic Partial Reconfiguration (DPR) has been widely used to provide the hardware flexibility for SDR implementation reusing the same hardware resources on the FPGA [15, 16].

This Chapter gives an introduction to the hybrid FPGA device and the internal architecture of Xilinx Zynq 7000 which is used in this thesis as the FPGA device for SDR hardware implementation. Also. An introduction to the runtime DPR technique and its advantages. Also, This Chapter investigates experimentally the high-speed DPR implementation for SDR using different DPR Partial Reconfiguration (PR) controllers taking into account the impact of reconfiguration throughput or the reconfiguration time, and power consumption on the implemented system performance. A multi-standard convolutional encoder design is implemented as a case study on the SDR DPR based implementation.

2.2. Hybrid Xilinx ZynQ FPGA Overview

FPGA is a reconfigurable logic device that can be reconfigured or reprogrammed to execute certain application reusing the same hardware resources. Over the past three decades, the FPGAs have been developed to become more complex and containing several types of reconfigurable resources [5]. Xilinx FPGA is one of the most popular FPGA devices in the market. Xilinx FPGAs are SRAM-based FPGA technology that is composed of two distinct layers: the configuration memory layer and the hardware logic layer. The configuration memory layer stores the FPGA configuration setup also called configuration bitstream that contains all the required information to reprogram the FPGA for a certain logic structure. The hardware logic layer contains the hardware resources of the FPGA, including Lookup Tables (LUTs), Block RAM (BRAM) and Digital Signal Processing (DSP) blocks

Hybrid Xilinx Zynq FPGA is a new generation of All Programmable System on Chip (SoC) [36]. Xilinx Zynq combines a dual-core ARM Cortex A9 processor with a 7-series Xilinx Artix FPGA [35]. The FPGA part of the Zynq device is called Programmable Logic (PL) and the other part that contains the ARM processor and the related peripheral blocks is called Processing System (PS). The PS side communicates with the PL side through the Advanced eXtensible Interface (AXI) which provides a high bandwidth and

low latency connections between the two sides of the Zynq device. Figure 2.1 shows the architecture of the Xilinx Zynq FPGA and the location of the PL and PS sides. Also, it is shown that the PS side contains a DDR controller for external DDR memory, SD-Card and UART controllers and I/O peripheral controller. Xilinx Zynq FPGA provides more level of flexibility by allowing runtime DPR to change the functionality of the reconfigurable resources on the FPGA during runtime [17]. In this thesis, Xilinx Zynq FPGA device XC7z020clg484-1 is selected for the DPR hardware implementation of SDR system.

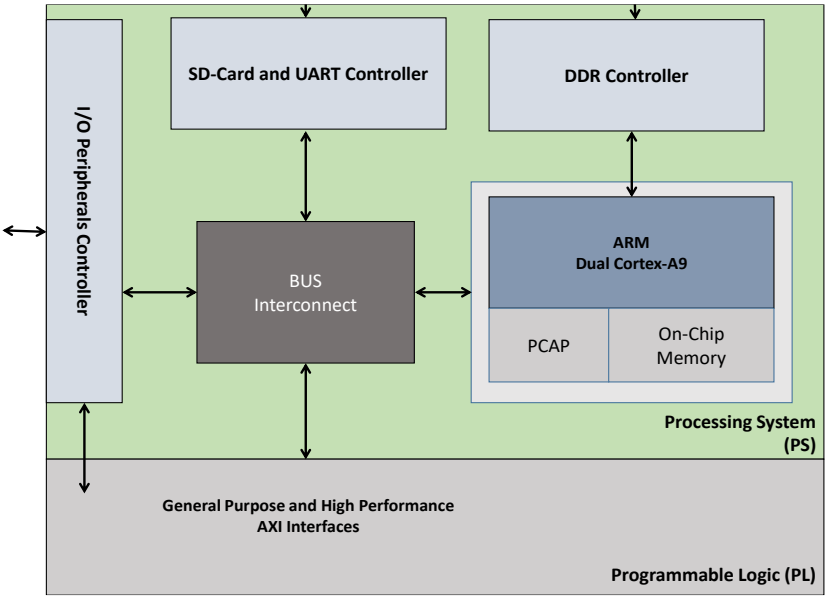


Figure 2.1: Xilinx Zynq FPGA Architecture

2.2.1. Zynq Programmable Logic (PL) part

The PL part of the Zynq device is shown in Figure 2.2. The architecture of Zynq FPGA is based on the Xilinx 7-series FPGA architecture. The PL part is a traditional FPGA logic fabric. The PL part architecture is divided into six clock regions. Each clock region contains tiles of reconfigurable resources. The resources tiles are organized in columns or blocks spanned vertically along the clock region. A column contains a single type of resources: CLB, BRAM, DSP or Input/output blocks (IOB) for interfacing.

2.2.1.1. Configurable Logic Blocks (CLB)

CLB is the main programmable logic resources on the FPGA that is used for implementing logic circuits on the FPGA [37]. CLB is located in a two-dimensional array on the PL part and connected to similar resources type other CLBs via programmable interconnects. Each CLB contains two slices connected to a switch matrix to switch between them as shown in Figure 2.3. Slice is a sub-block inside the CLB that contains logical resources for implementing combinational and sequential logic circuits. A single slice is composed of 4 six input LUTs, 8 Flip-Flops (FFs) and multiplexers. LUT is the core element of the CLB which is reprogrammed to implement logic functions up to six input or be used as RAM/ROM. LUTs are combined together to form a larger logic function, memories or shift registers as required. FFs is a sequential element that can be used to implement a latch. Xilinx Zynq XC7z020clg484-1 has in total 6650 CLB blocks, 53200 LUTs, 106400 FFs and 200 IOBs.

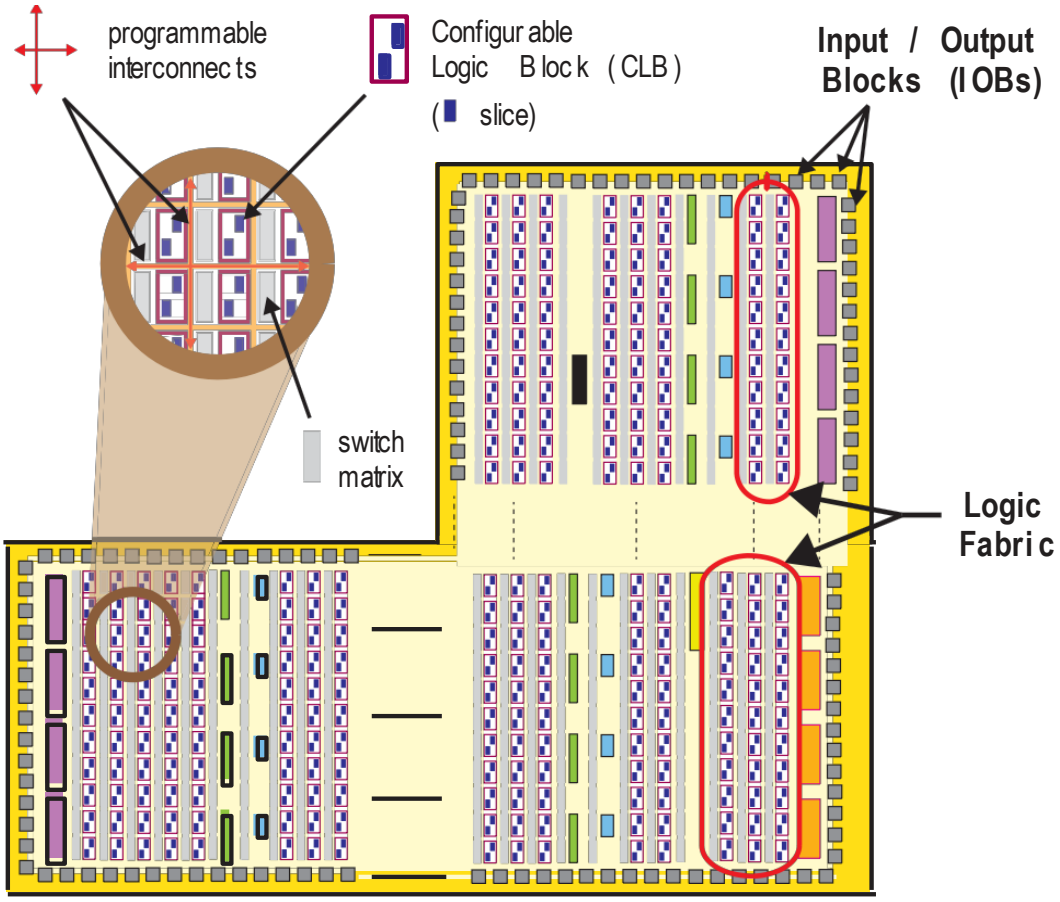


Figure 2.2: The Programmable Logic part of The Zynq FPGA Device [42]

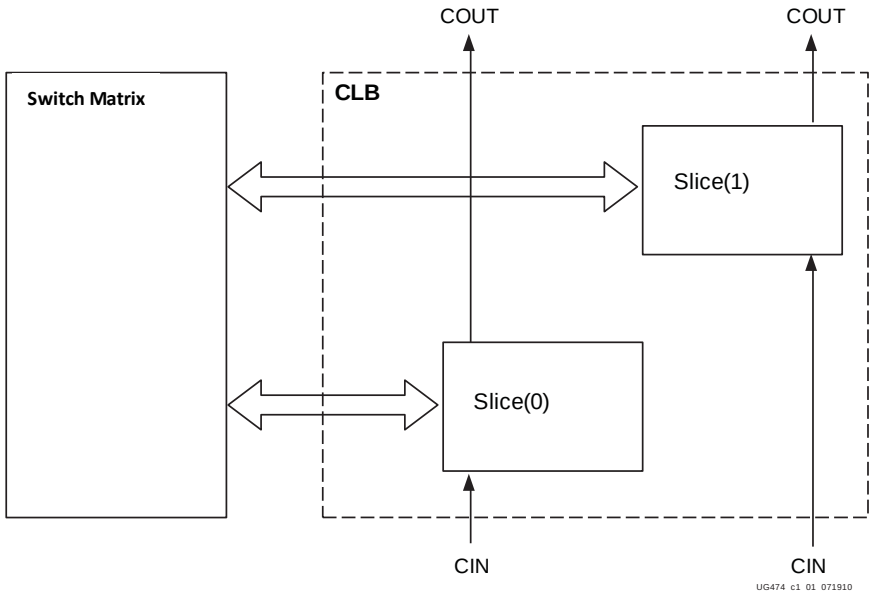


Figure 2.3: CLB Routing Matrix in Xilinx 7-Series FPGA [37]

2.2.1.2. BRAM and DSP Blocks

In addition to the CLBs on the PL fabric, there are two specific components: BRAM for memory requirements and DSP48E1 for high-speed arithmetic operations as shown in Figure 2.4. BRAMs in Zynq FPGA are equivalent to BRAM blocks in Xilinx 7-series FPGA [38]. Each BRAM block can store up to 36Kbit of information. It could be configured to be one block of 36Kbit or two independent blocks each one has 18Kbit of information. BRAMs are used to implement RAM, ROM, and FIFOs in the PL fabric. Using BRAM has a significant improvement on the resources utilization by implementing larger capacity of memories in small physical elements, the alternative is using distributed RAM which is constructed by using a large number of LUTs that spanned over a larger area on the PL fabric. DSP48E1 is a special slice for implementing high-speed arithmetic operations [39]. DSP48E1 consists of pre-adder/subtractor, multiplier, and post-adder/subtractor. Various complex computations especially floating-point arithmetic blocks could be implemented using DSP48E1 slices. Xilinx Zynq XC7z020clg484-1 has in total 140 BRAM blocks of 36Kbit and 120 DSP48E1 slices.

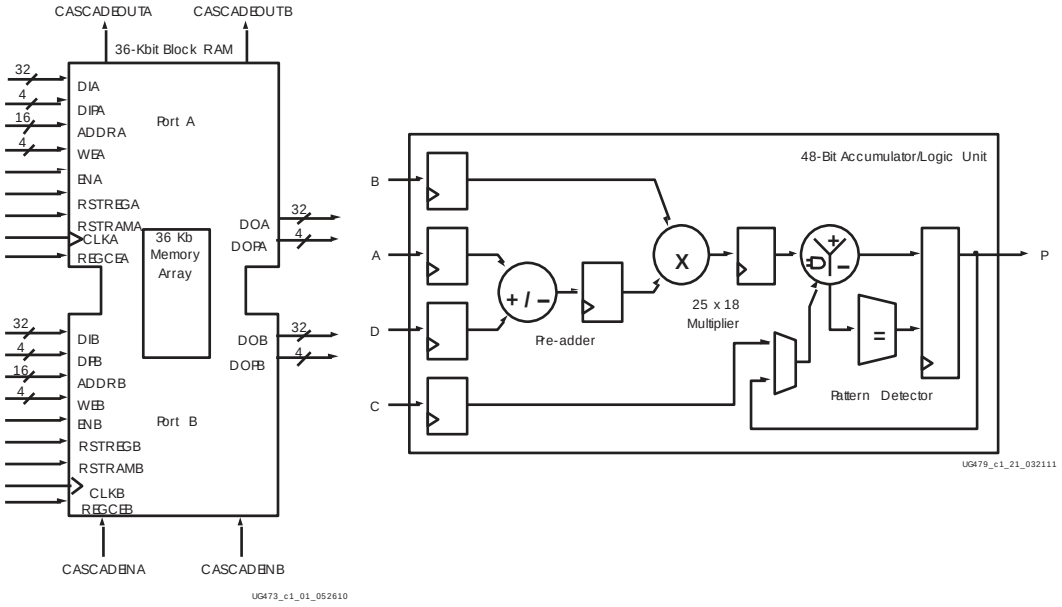


Figure 2.4: Block RAM (Left) [38] and DSP48E1 Block (Right) [39] in Xilinx 7-Series FPGA

2.2.2. Zynq Processing System Processing System (PS) Part

All Zynq devices have the same architecture, and all of them contain a PS part [36, 42]. The PS part contains as shown in **Figure 2.5**, an Application Processor Unit (APU), I/O peripheral interfaces, memory interfaces and controllers, interconnect circuits, and clock generators. The APU consists of two ARM Cortex A9 processors, each core is associated with a Media Processing Engine (MPE), Floating Point Unit (FPU) and Memory Management Unit (MMU) plus data and instruction caches memory. The PS part has different interfaces between the PS and PL and the PS and external components.

The communication between the PS and the external components like UART, SD-Card, and USB is achieved via Multiplexed Input/output (MIO). In Zynq FPGA all the

external components are linked to the PS unit and there is no direct pass from the PL to the external components. So for the logic on PL to be communicated with the external components communication interfaces should exist between the PL and PS. AXI interconnects and interfaces are forming the bridge between the PL and PS through general purpose 32-bit ports and high-performance 64-bit ports on the PS side. The PS unit is controlled and configured through software codes running on the ARM processor usually C or C++ programming languages are running on a standalone operating system on the ARM processor. PS unit has an important role in debugging and testing of the implemented logic on the PL side. Throughout the thesis, the PS unit is used to deliver external input data samples to the PL and capturing output from the PL to an external memory to test the functionality of the implemented SDR communication blocks.

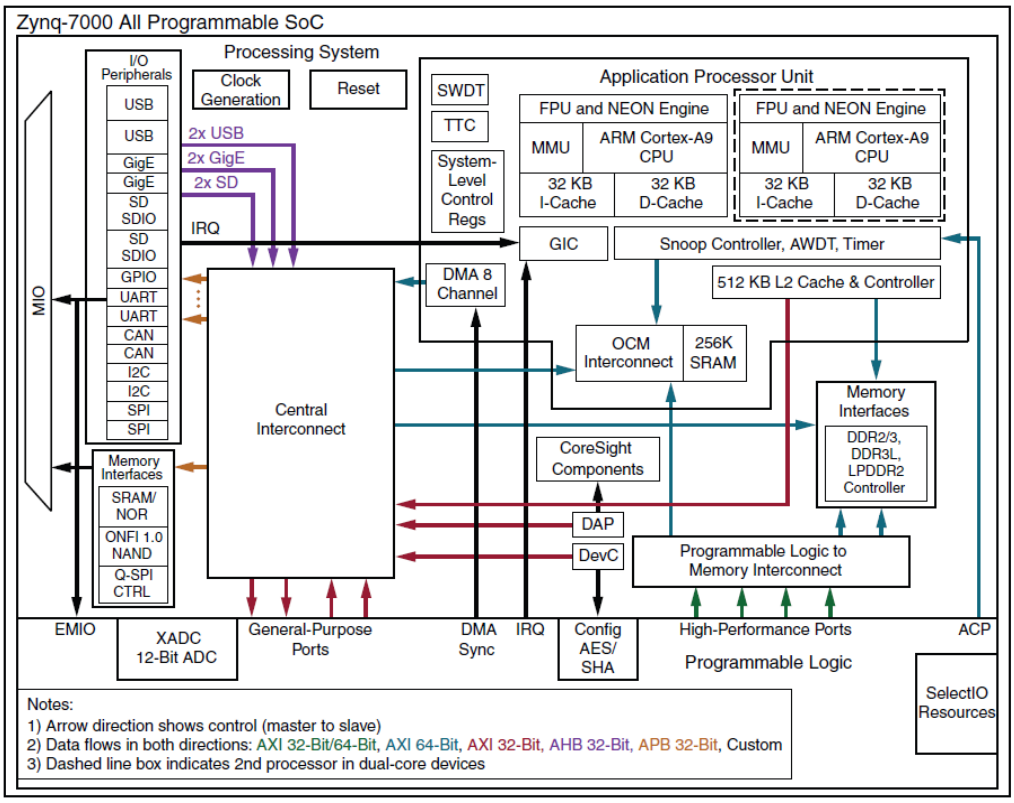


Figure 2.5: The Processing System Part of The Zynq FPGA Device [36]

2.3. Dynamic Partial Reconfiguration

DPR is a feature of SRAM-FPGAs that offers the advantage of flexibility to reconfigure a part of FPGA during runtime reusing the same hardware resources [7] Xilinx DPR design flow requires the partitioning of the design into a static part and a dynamic part [34] as shown in Figure 2.6. The dynamic part contains the reconfigurable modules (RM) of the system, while the static part contains the static modules that are not be affected during the reconfiguration. The dynamic part contains multiple Reconfigurable Regionss (RRs), each RR has a set of RMs which can be swapped during runtime without interruption. A partial bitstream is generated for each RM to be mapped into a specific RR during reconfiguration. Partial bitstreams are loaded from a nonvolatile memory to the FPGA configuration memory through dedicated configuration interfaces.

DPR are classified according to the configuration modes as internal or external reconfiguration techniques, based on the reconfiguration is handled internally within the FPGA or by an external device like a PC or another FPGA. Xilinx 7-series FPGAs have two internal configuration interfaces to the FPGA configuration memory [37]: (i) The Internal Configuration Access Port (ICAP) that is physically located on the FPGA fabric. (ii) Processor Configuration Access Port (PCAP) only available for the Xilinx 7-series Zynq FPGA equipped with a hard macro ARM processor. Also, three external configuration interfaces are used through the serial configuration ports: JTAG, Serial mode, and Select-Map. In this thesis, only the internal configuration modes are used for DPR to achieve a high speed of configuration.

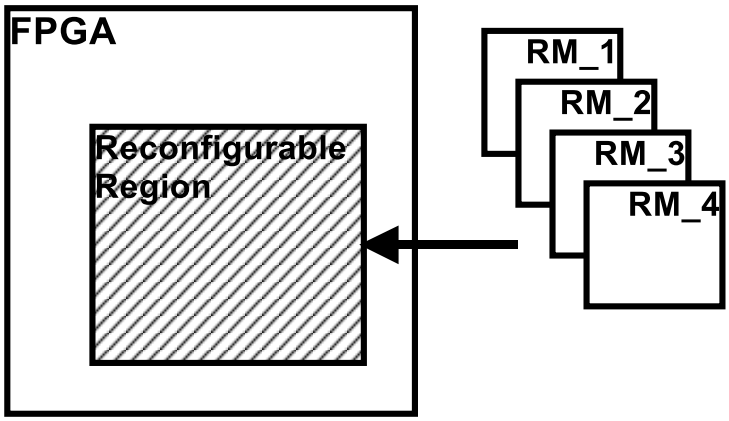


Figure 2.6: Dynamic Partial Reconfiguration in SRAM-FPGAS.

2.3.1. Configuration Modes

DPR can be performed by loading RMs partial bitstreams to the FPGA configuration memory. Accessing the configuration memory is done through various FPGA configuration modes or configuration ports [35]. Configuration modes are categorized by the type of configuration interface used to access the configuration memory. Table 2-1 shows the different configuration modes for Zynq FPGA.

2.3.1.1. External Modes

External configuration modes use external FPGA interfaces to load the partial bit files to the FPGA configuration memory. JTAG is the only external configuration port for Zynq FPGA. The partial bitstreams are transferred from an external storage source, for example, the PC through the JTAG serial interface to the configuration memory. The data rate of the JTAG configuration interface is limited to 8.25 MB/S and not suitable for real-time application like the SDR system [18] and required an external PR controller like CPU or another FPGA to control the reconfiguration process.

2.3.1.2. Internal Modes

Internal configuration modes use internal FPGA interfaces to load the partial bit files to the FPGA configuration memory. Two internal configuration modes are used in Xilinx Zynq FPGA. 1) ICAP configuration mode is based on the ICAP hard macro 32-bit configuration port primitive located on the PL side to access the configuration memory

with a theoretical data rate of 400 MB/S. 2) PCAP configuration mode is based on the PCAP 32-bit configuration port in the PS side controlled by the ARM processor to access the configuration memory with a data rate of 400 MB/S

Table 2-1: Zynq FPGA Configuration Modes

Configuration Mode	Type	Max Clock	Data Width	Max Bandwidth
ICAP	Internal	100 MHZ	32-bit	400 MB/S
PCAP	Internal	100 MHZ	32-bit	400 MB/S
JTAG	External	66 MHZ	1-bit	8.25 MB/S

2.3.2. Advantage and Disadvantage of DPR

The main advantages of the reconfigurable systems are:

Resources utilization: In traditional design implementation most of the hardware resources are not used at a time till it is activated to operate for a certain period of time. Using reconfigurable hardware and DPR will increase the resource utilization by only implemented the active part of the design in the required time and time multiplexing the resources between the design hardware modules according to activity schedule.

Scalability: Using reconfigurable hardware will allow upgrading system to accommodate newly defined tasks to handle the growth in technology and features. It also facilitates the deploying of bug fixing in hardware which will decrease the cost of redeploying new hardware and increase the time to market for the products.

Reusability: Reusing the resources for different design implementations, where a system can be customized for adaptability.

Power reduction: is considered the most important item, where power consumed for the system although most of the parts are not working. In the Integrated Circuits (IC) design, the static power is consumed by the device although it is not active. FPGA reconfiguration helps in delaying the implementation of a specific part until its time of operation, which will decrease the consumed power over time and though the battery lifetime.

Area: Instead of implementing a full system horizontally which consume area, System can be optimized by vertical implementation idea which is programming in space and time. Where a stack of blocks are stored and loaded at the time of operation. This will save the area used by the same blocks in the horizontal design.

On the contrary, there are some disadvantages for the DPR and they are improving by research such as:

Latency: Latency increased by the time overhead add by the reconfiguration time [19]. It could be improved by using high-speed PR controller to speed up the reconfiguration time.

Memory: As blocks will be stored, in what is called vertical design approach in this thesis, more memory is needed for storing the different implementations until the time of operation. As the storage sizes are increasing this item is improved. For example, 5 files of few kilobytes contain the new reconfiguration can be stored on gigabytes of attached storage device. Reconfiguration files can be stored on servers and accessed through the network as the network accessing are improving by time.

2.4. Dynamic Partial Reconfiguration Controller

Partial Reconfiguration (PR) controllers are proposed by researchers to control the DPR and to enhance the reconfiguration speed and maximize the reconfiguration throughput [19]. PR controller provides the interface for transferring partial bitstream data to the FPGA internal configuration port (i.e., ICAP or PCAP) from an external or internal memory with a high data throughput as shown in Figure 2.7. Moreover, some PR controller architectures have the capabilities of monitoring the system performance by measuring the reconfiguration time and determine the status of RMs.

The time of reconfiguration is a key factor in the design of PR controller as it measures how fast the controller can handle the reconfiguration process. A lot of PR controllers are used in DPR whether they are conventional controllers provided by the FPGA vendors or novel controllers developed to carry out the DPR to be more efficient and reducing the reconfiguration time. PR controller is implemented by using a Finite State Machine (FSM) or a dedicated custom processor to control and manage the DPR.

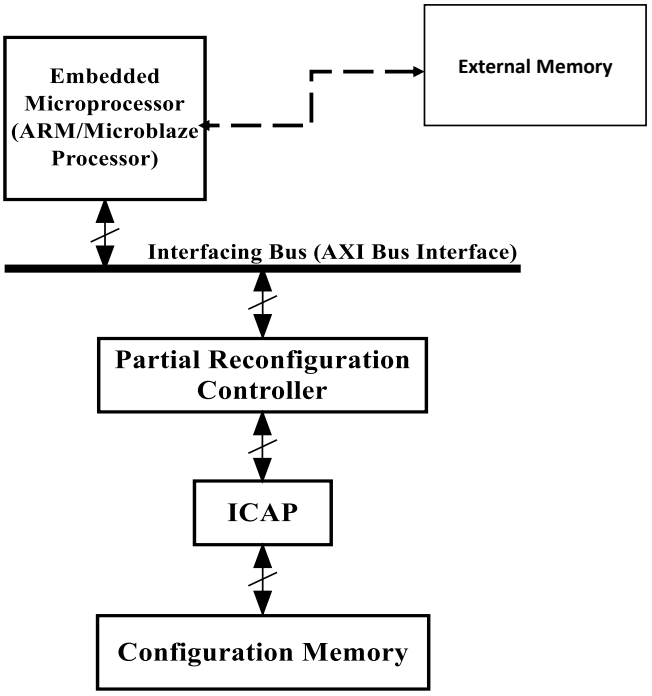


Figure 2.7: Partial Reconfiguration Controller in DPR System

2.4.1. Time of Reconfiguration

Reconfiguration time (or reconfiguration throughput) is a critical parameter in DPR designs which affects the whole system performance implemented on the FPGA. Reconfiguration time depends on the dimension of RP and the generated partial bitstream data size. Also, the memory configuration setups used for data transfer increase the reconfiguration time overhead. Many contributions are proposed in the literature to improve the reconfiguration time and enhance the feasibility of DPR.

In [20] authors minimizing the runtime DPR overheads by using streaming Direct Memory Access (DMA) engines and a bitstream size compression technique for achieving a high reconfiguration throughput. Different versions of open source high-speed ICAP controllers are presented in [21-22] that significantly improve the reconfiguration throughput. Therefore, these controllers are studied and evaluated in this chapter with the other conventional controllers to study the implementation of a high-speed DPR-based SDR system.

Figure 2.8 shows different types of internal PR controllers supported by Xilinx Zynq FPGA and they are described in the next sub-sections as follows.

2.4.2. Xilinx ICAP Controller (AXI-HWICAP)

Xilinx provides several IP cores for interfacing the Xilinx's ICAP primitive with the user system design. ICAP Controllers enable an embedded microprocessor such as Microblaze or ARM processors to access the configuration memory. ICAP is a Xilinx FPGA hard macro that provides direct access to the configuration memory both in read and write modes [35]. The ratio of the ICAP interface data width to the configuration memory is 8, 16 or 32 bit wide in Xilinx 7-series. The ICAP provides a maximum theoretical reconfiguration throughput equal to 400 MB/S at a data width of 32 bits and a clock frequency of 100MHZ.

Practically, the reconfiguration throughput that has been measured in real time applications is less than the theoretical ICAP throughput due to the reconfiguration overhead added to the DPR at the system level design. XPSHWICAP and AXI-HWICAP [40] are two ICAP Controllers designed for Processor Local Bus (PLB) and AXI buses interfaces which are connected to the buses as a slave peripherals.

In this thesis, only AXI-HWICAP shown in Figure 2.9 is considered because it is the only Xilinx ICAP controller IP core supported by the Xilinx Zunq FPGA and the AXI bus interface. During DPR, the partial bitstream data are buffered from an external memory to a Write/Read FIFOs inside the core. An internal state machine observes the occupancy status of the FIFOs and continuously supplies partial bitstream data to the ICAP and then to the configuration memory. These controllers give a low reconfiguration throughput that's far below the theoretical ICAP throughput. The limitation comes from the low data rate of partial bitstream sent to the ICAP while using the microprocessor for fetching partial bitstreams data from an external memory to the controller FIFOs.

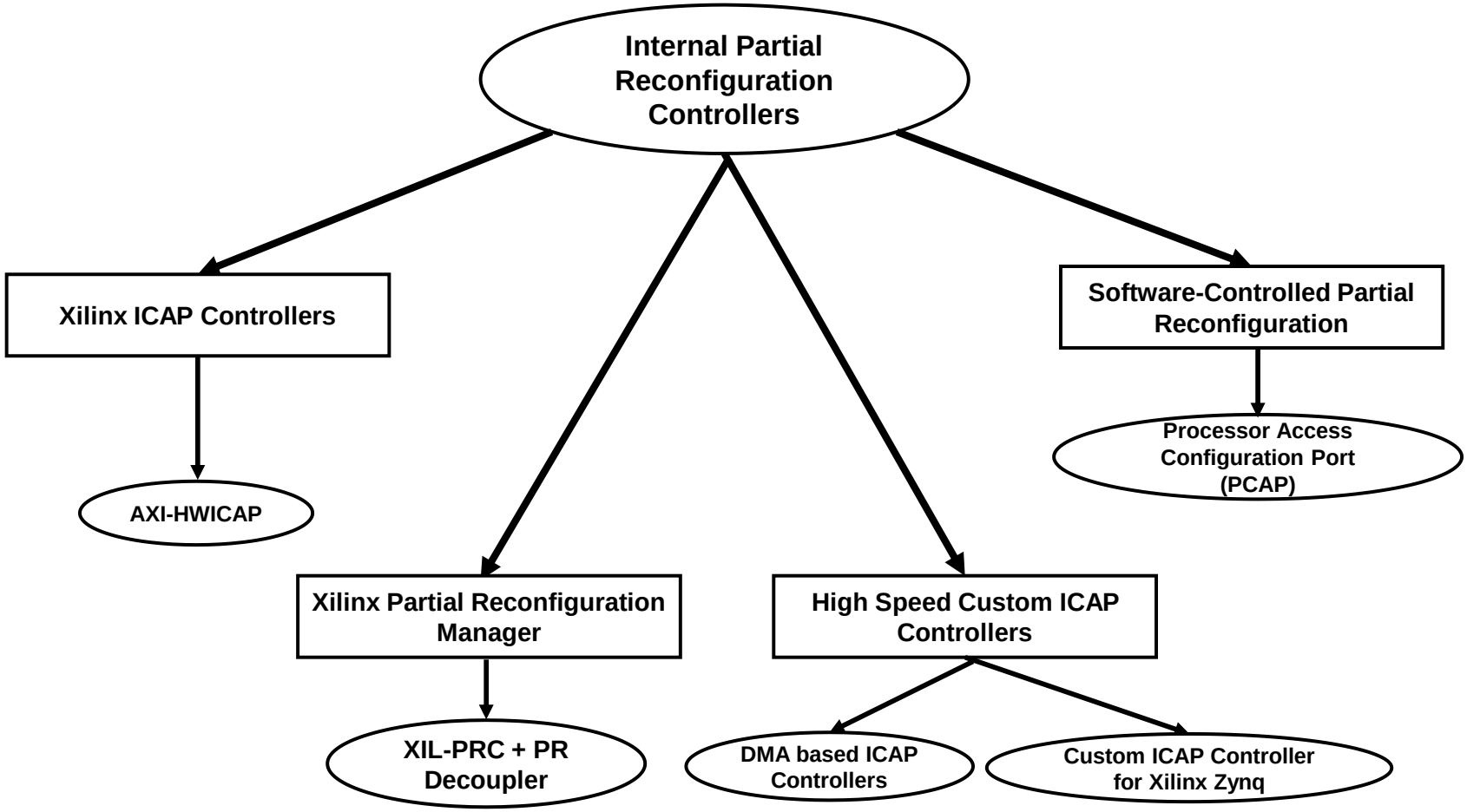


Figure 2.8: Types of Internal Partial Reconfiguration Controllers

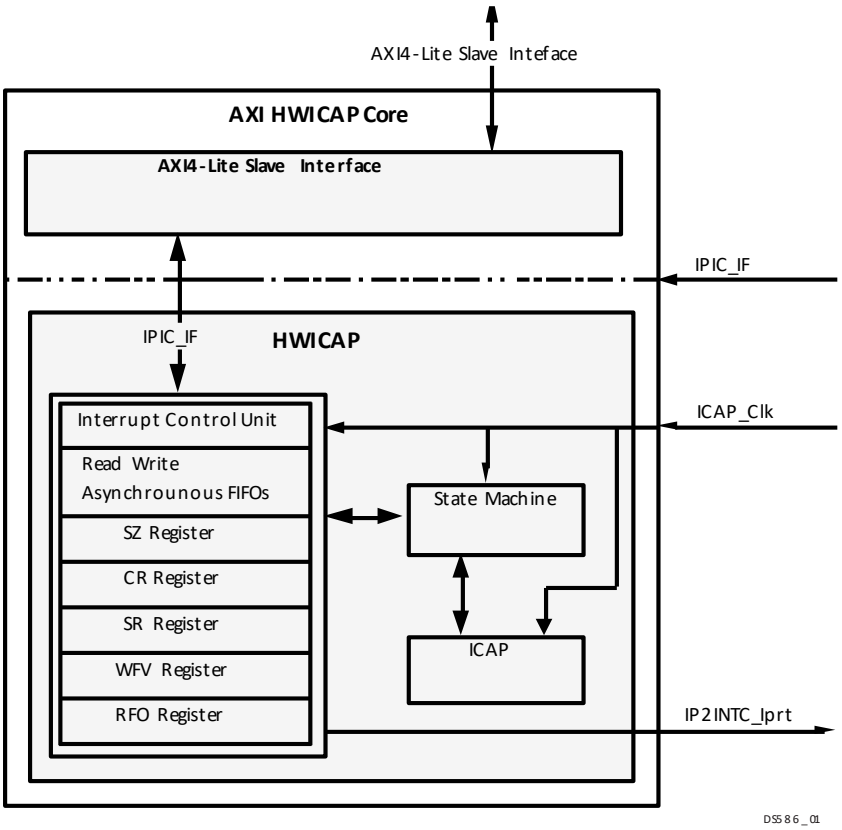


Figure 2.9: AXI- HWICAP Partial Reconfiguration Controller [40]

2.4.3. Custom DMA Based ICAP Controller

Various Open Source ICAP Controllers are proposed by researchers [21-25] to improve the reconfiguration time while using the ICAP through an embedded microprocessor. Moving partial bitstream data to the ICAP using a microprocessor is inefficient and reduces the feasibility of DPR. In [24], an open source ICAP controller is proposed, reaching a reconfiguration throughput of 399.80MB/S. The controller fetches the partial bitstream data from a DDR using the DMA controller to an asynchronous FIFO connected to the ICAP and then to the configuration memory. A custom reconfiguration controller for Xilinx Zynq FPGA (ZYCAP) is presented in [25], the design is shown in Figure 2.10. ZYCAP achieves a reconfiguration throughput of 382 MB/S. ZYCAP is a DMA based AXI-HWICAP controller equipped with two AXI slave bus interfaces connected to the ARM processor. The AXI4 bus interface is used by the DMA to receive partial bitstream data from DDR memory and the AXI-Lite bus interface for control signals.

2.4.4. Software-Controlled Partial Reconfiguration

Xilinx Zynq FPGA provides the potential to implement a software-controlled (S-C) DPR through the processing system (PS) device configuration (DevC)/PCAP interface [44]. This scheme does not require any programmable logic (PL) resources during DPR as shown in Figure 2.11. The DevC unit is controlled by the ARM processor to select the internal configuration interface to be PCAP or ICAP according to the user system design.

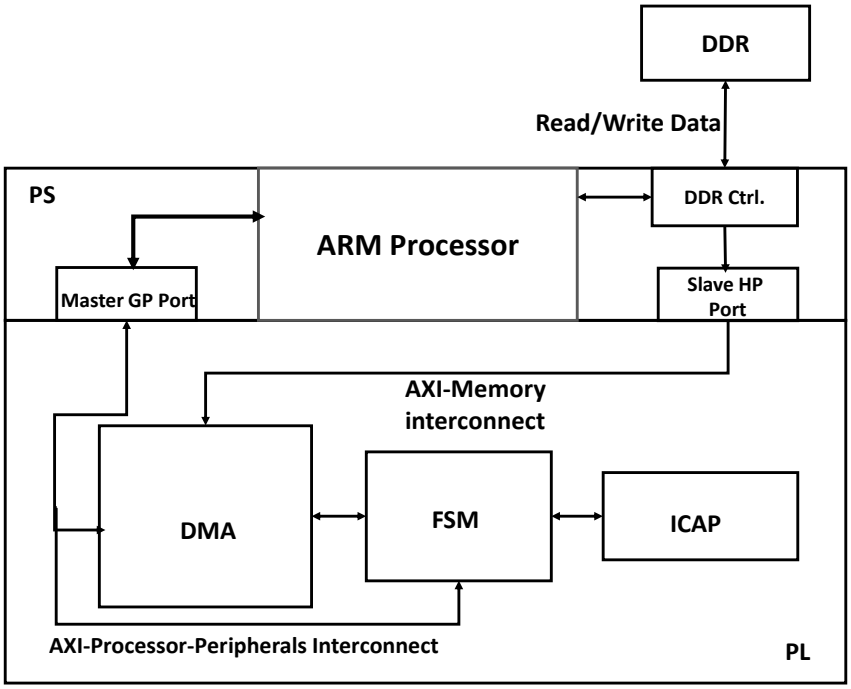


Figure 2.10: Custom DMA Based ICAP Controller

PCAP interface contains AXI-PCAP Bridge to access the configuration memory in the PL side from the PS side. In S-C PR, the ARM controls the DPR using an internal DMA engine to transfer partial bitstream data from an external memory to PCAP interface and then to the configuration memory on the PL side. The PCAP has a 32-bit wide data interface that operates at a clock frequency of 100 MHZ, achieving a theoretical reconfiguration throughput of 400 MB/S similar to the theoretical reconfiguration throughput achieved by the ICAP. The drawback of that scheme is that it blocks the ARM during DPR and prevents it from doing other software tasks.

2.4.5. Xilinx Partial Reconfiguration Manager

The Xilinx Partial Reconfiguration Controller (Xil-PRC) core shown in Figure 2.12 provides management functions for self-controlling partially reconfigurable designs, the core is equipped with AXI4-Lite bus interface [41]. Xil-PRC is released for enclosed systems where all the design RMs are known to the controller. Xil-PRC consists of a 32 Virtual Socket Managers (VSM) and each VSM can contain up to 128 RMs. The Virtual Socket is a term that used to refer to a RP that is managed by the Xil-PRC. VSMs are connected to a fetch path that fetches the partial bitstream data from an external memory to the ICAP without passing by the processor. Each VSM operates independently of the others and has its own AXI4-Stream interface for partial bitstream data transfer. An external trigger is sent to a VSM to select the required RM to be loaded into the RP, each RM has its own trigger number. Xil-PRC accesses the external memory using an AXI External Memory Controller (axi_emc) or a Memory Interface Generator (MIG). Xil-PRC does not depend on the processor to fetch partial bitstreams from an external memory, which leads to a high reconfiguration throughput close to the theoretical ICAP throughput.

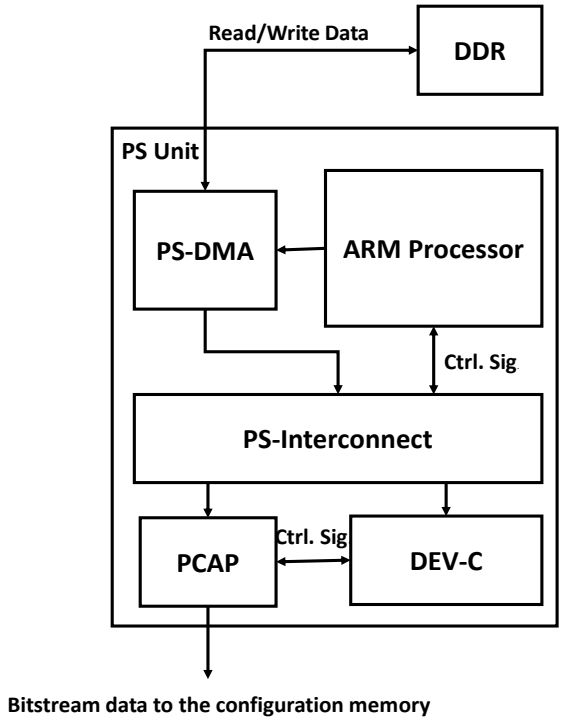


Figure 2.11: Software Controlled Partial Reconfiguration

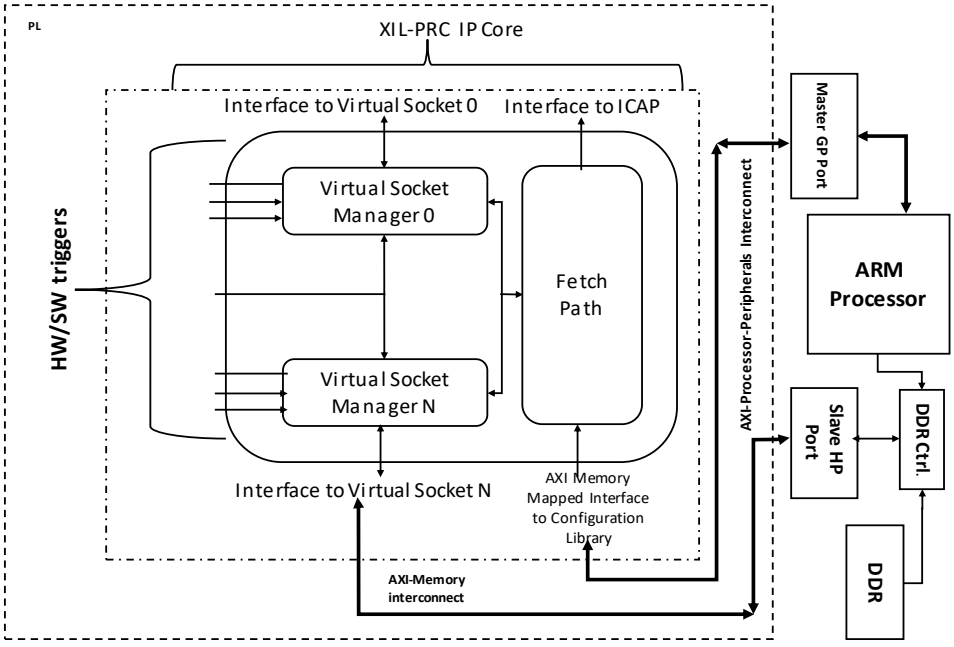


Figure 2.12: Xilinx Partial Reconfiguration Manager/Controller (Xil-PRC [41])

Figure 2.13 shows the reconfiguration time of the four types of PR controllers discussed in this section with various reconfigurable partition regions sizes. The reconfiguration time is accurately calculated using AXI timer and ARM timer to count

the number of clock cycles required to complete the DPR process. Xil-PRC and DMA-based ICAP controller (ZYCAP) achieve the minimum reconfiguration time thanks to the use of DMA to speed up the transfer data rate between the DDR and the ICAP interface to the configuration memory. AXI slave memory mapped AXI-HWICAP is the worst PR controller in term of reconfiguration time due to the use of internal ARM cache memory for data transfer between the DDR and the ICAP interface to the configuration memory.

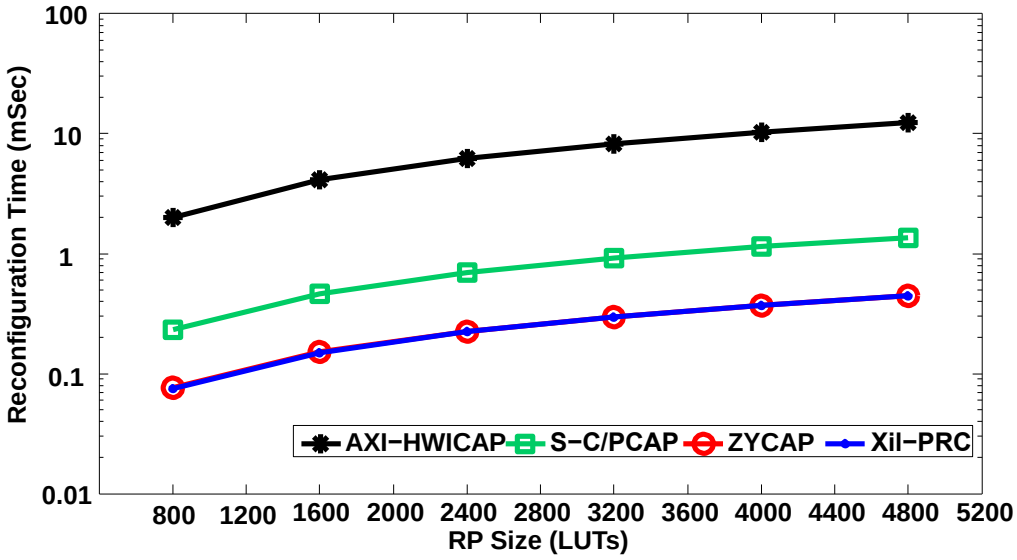


Figure 2.13: Reconfiguration Time for Different PR Controllers

2.5. Multi-Standard Convolutional Encoder for SDR system (A Case Study)

In order to analyze the impact of DPR techniques on the implementation of SDR, a multi-standard convolutional encoder is used as a case study as shown in Figure 2.14. Convolutional encoders are one of the Forward Error Correction (FEC) coding schemes widely used in wireless communication systems to reduce the effect of noisy channels.

Three main parameters are used for describing convolutional codes(n, k, l), where n represents the number of output bits, k represents the number of input bits and l represents the constraint length or the number of shift registers. The Convolutional encoder rate is. In our experiment, seven encoder schemes are used as a benchmark from (2G, 3G, WIFI, and LTE) communication standards. DPR is used to switch between the seven encoders RMs at runtime reusing the same RR on the FPGA.

2.5.1. 2G Convolutional Encoder

The 2G is the second generation of the mobile communication. It is based on the Global System for Mobile Communications (GSM) standardized by the European Telecommunications Standards Institute (ETSI). The convolutional encoder is used in channel coding block of the 2G standard [45].

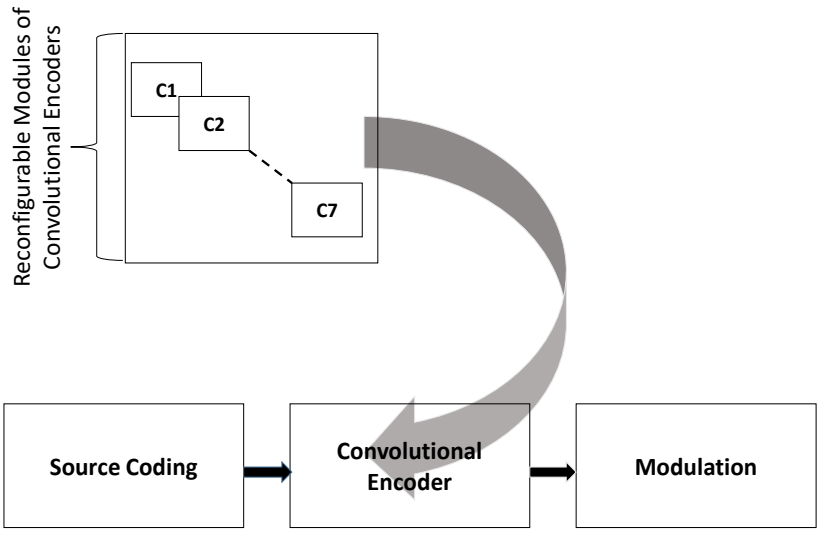


Figure 2.14: Multi-Standard Convolutional Encoder

2.5.2. 3G Convolutional Encoder

The 3G is the wireless mobile third generation standard. 3G uses the Wideband Code Division Multiple Access (WCDMA) radio technology to offer greater spectral efficiency and data rate up to 384 kbps in uplink and downlink and simultaneous voice and data. The 3G uses the convolutional encoders in the channel coding of the different channels [46]. Figure 2.15 shows the block diagram for the convolutional encoders used in the 3G mobile system. The 3G standard uses a zero biting, where the initial state of the shift registers should be zero and also the final state should return back to zero.

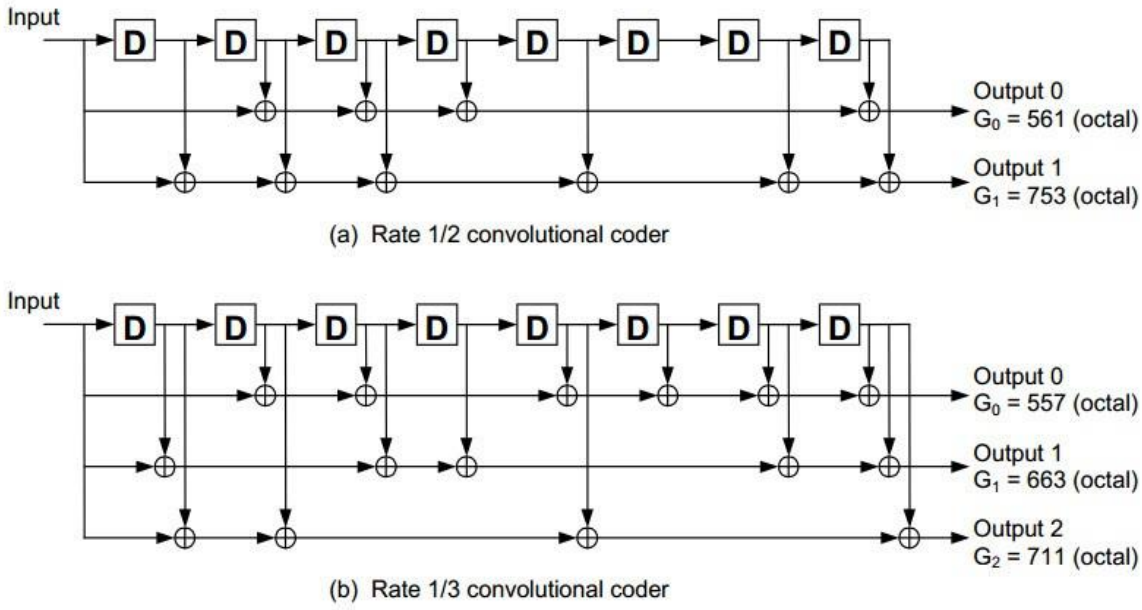


Figure 2.15: 3G Convolutional Encoder [46]

2.5.3. LTE Convolutional Encoder

Long Term Evolution (LTE) or 4G technology is the fourth generation in wireless mobile standards. The LTE uplink transmitter uses Single Carrier Frequency Division Multiple Accesses (SC-FDMA) for its lower peak-to-average power ratio (PAPR) results in a power cost reduction in transmitter terminal. LTE supports six different channel bandwidths from 1.4 to 20 MHZ in both frequency and time division duplex (FDD and TDD). Figure 2.16 shows the convolutional encoder used in the LTE [47], and Figure 2.17 shows the turbo encoder that is based on the convolutional encoder [47]. In the LTE standard, the tail bit is used where the initial and final states of the shift registers should be the same.

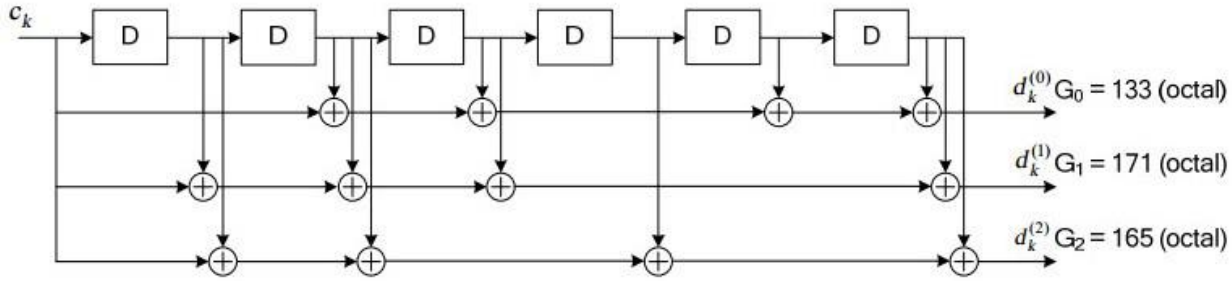


Figure 2.16: LTE Convolutional Encoder [47]

2.5.4. WIFI Convolutional Encoder

WIFI is a fixed wireless communication standard defined by Institute of Electrical and Electronic Engineers (IEEE) and is known as IEEE 802.11 standard. There are several specifications in the 802.11 family each has different channel characteristics as well as different data rates, modulation techniques and ranges of operation. The convolutional encoder shown in Figure 2.18 is used in 802.11 a / g standards [48]. It will be implemented in the thesis demonstration for the SDR system using DPR

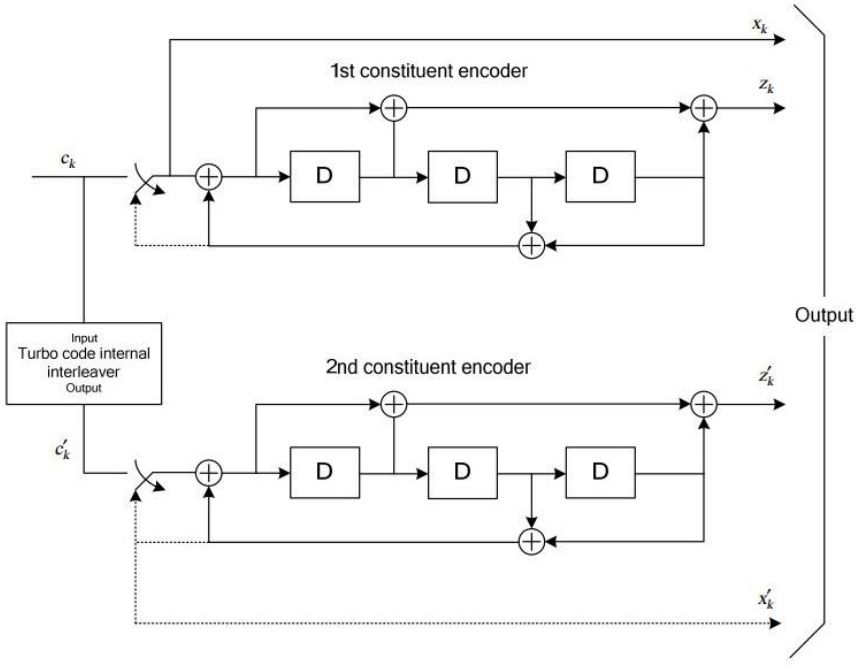


Figure 2.17: LTE Turbo Encoder [47]

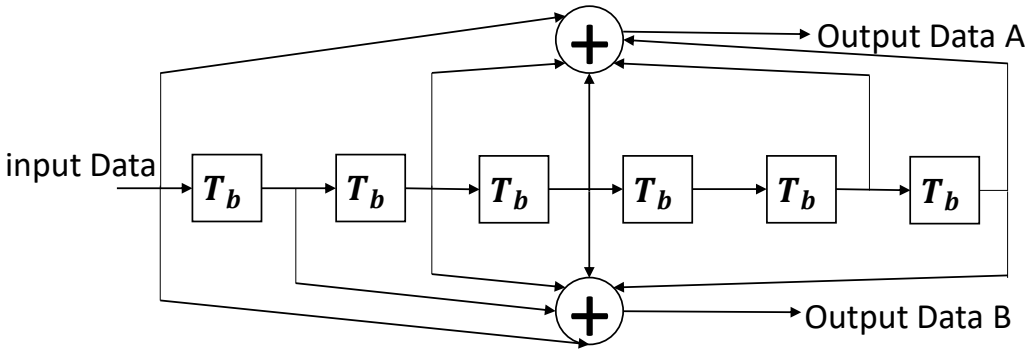


Figure 2.18: WIFI Convolutional Encoder [48]

2.5.5. Multi-Standard Convolutional Encoder DPR Implementation

Convolutional encoders are widely used in the wireless communication systems, such as 2G, 3G, LTE, and WIFI. Table 2-2 shows the characteristics of the different convolutional encoders used in 2G, 3G, LTE, and WIFI technologies. Different convolutional encoders reflect different hardware combinations of RMs.

Table 2-2: Convolutional Encoder RMs

Conv. Encoder	C1	C2	C3	C4	C5	C6	C7
System	2G	2G	2G	3G	3G	LTE	WIFI
Rate(K/n)	1/2	1/3	1/2	1/2	1/3	1/3	1/2
Constr. Length (l)	5	7	5	9	9	9	7

2.5.5.1. System Implementation and Setup

The system has been implemented using Xilinx Zynq XC7Z020LG484-1 FPGA and tested with a ZC702 board [43]. The DPR flow has been carried out using Xilinx Vivado tool. The complete system is developed as shown in Figure 2.18. The system design is controlled by the ARM Cortex-A9 microprocessor on the PS side, the ARM processor communicates with the PL using the AXI bus interface. The static part of the system consists of the PR controller, the ICAP for internal configuration, the AXI bus connections, and the static blocks in the wireless communication chain. The dynamic part contains a single RP for the reconfigurable encoder. The encoder has 7 RMs (C1-C7), each module is synthesized in 37 LUTs. A single encoder RM is active at a time. Encoders RMs (C1-C7) partial bitstreams data are stored on SD flash memory to switching between them during DPR by replacing the existing encoder RM with another RM on the RP. The system is connected to an external PC using UART interface to interact with the ARM processor through software codes.

2.5.5.2. Reconfiguration Time and Throughput

In this experiment, the reconfiguration time is computed by measuring the number of clock cycles required to complete the DPR task after a switching decision is issued by the ARM processor. The reconfiguration time is measured by AXI-Timer implemented on the PL side. Figure 2.20 (b) shows the reconfiguration throughput of the four PR controllers used for SDR implementation. SDR designs using Xil-PRC and ZYCAP achieve an average reconfiguration throughput of {396.5, 392 MB/S} approximately equal 98 % of the ICAP throughput.

2.5.5.3. Resources Utilization and Power Consumption

The resource utilization for the four PR controllers used in SDR designs is shown in Figure 2.20 (a). Xil-PRC utilizes 4X LUTs as compared to the AXI-HWICAP, while ZYCAP utilizes 556 LUTs approximately less than 2X of AXI-HWICAP LUTs.
Figure 2.20 (c) shows the average estimated power consumption of the four PR controllers used in the SDR designs. ZYCAP consumes the minimum power in comparison with the other PR controllers implemented on the PL side.

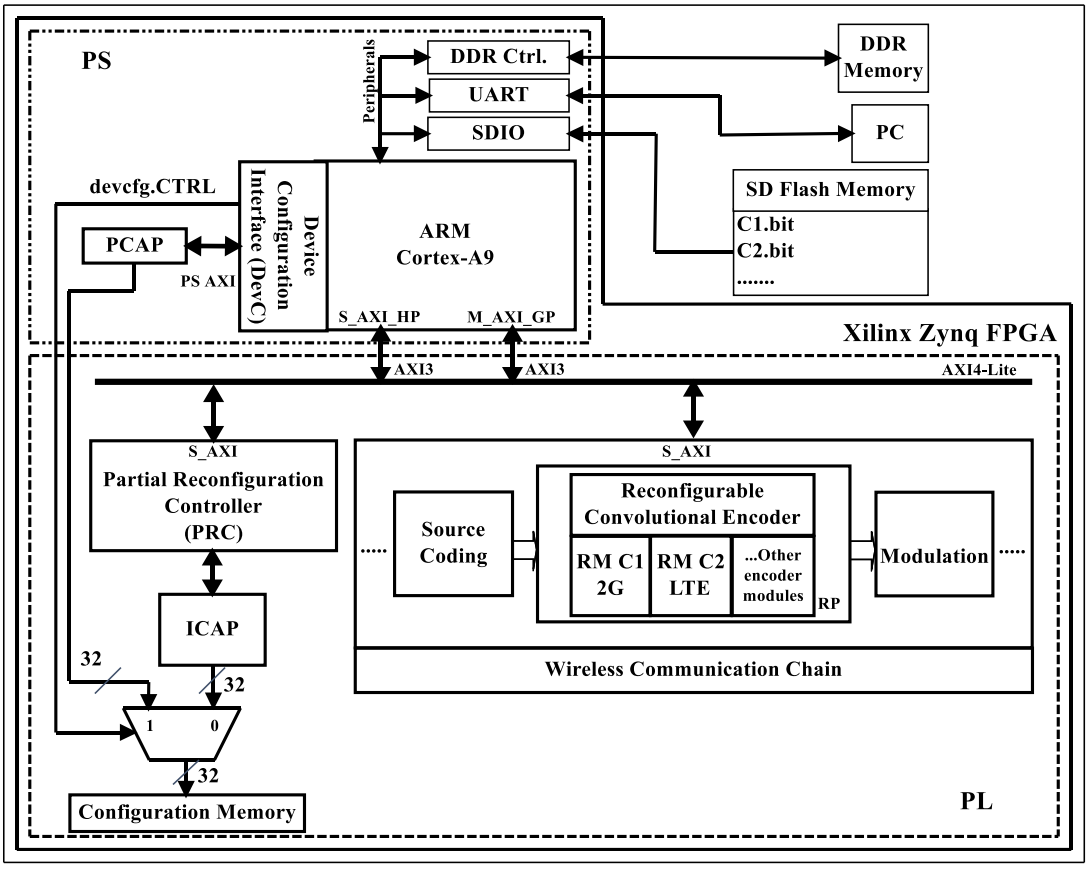


Figure 2.19: Multi-Standard Convolutional Encoder DPR system Overview

2.5.5.4. Design Insights and Recommendations

Xil-PRC and ZYCAP will be always recommended for SDR designs that require high reconfiguration speed as shown in Figure 2.13. Low power DPR-based SDR designs that use S-C / PCAP do not require any resources on the PL side for DPR.

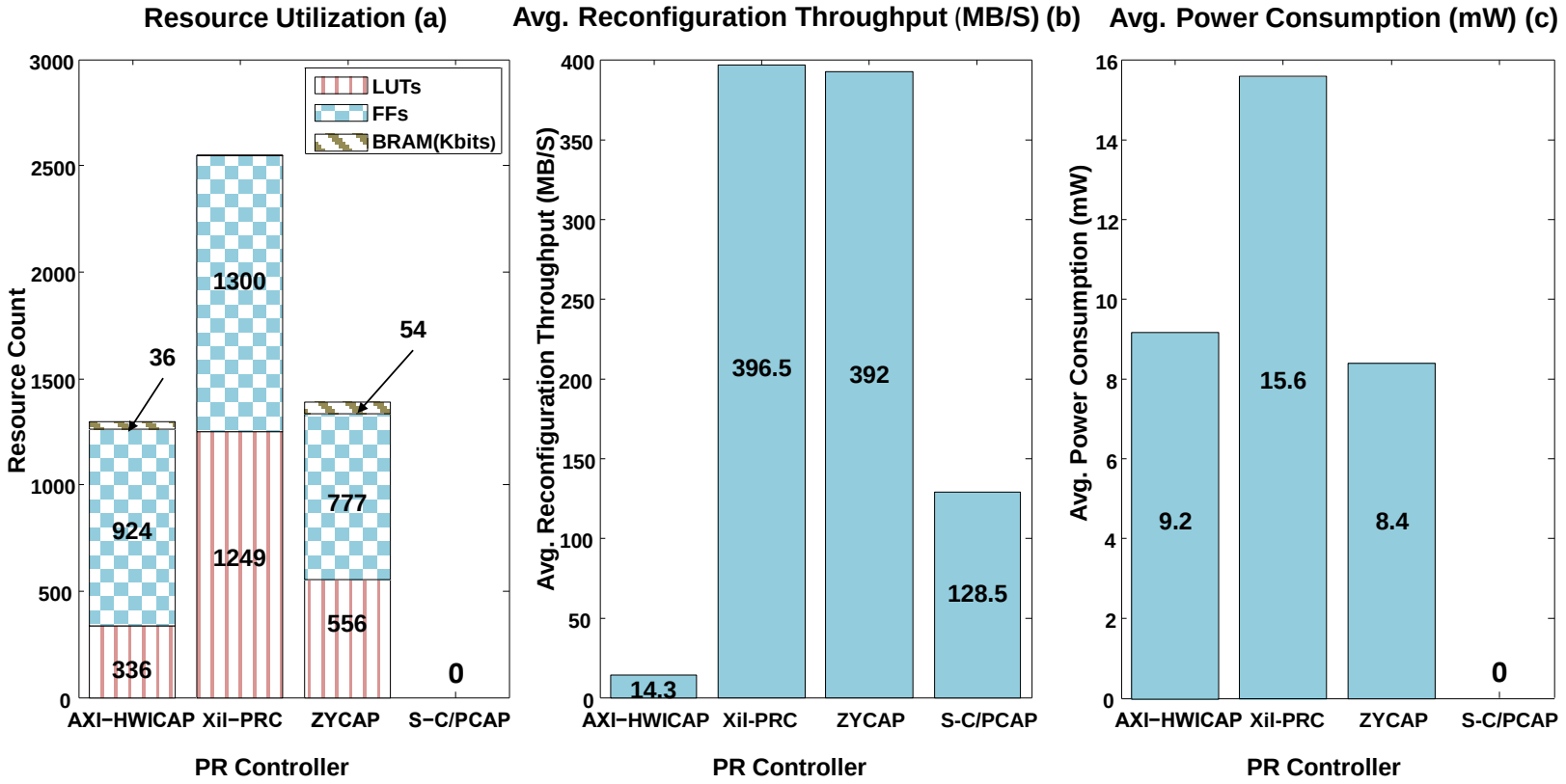


Figure 2.20: PR Controller Performance Evaluation

Therefore, the power consumption is the power consumed by the ARM during the reconfiguration as shown in Figure 2.20 (a, c), but the drawback of this scheme is that the ARM processor is blocked from doing other software tasks during reconfiguration time. Xil-PRC achieves the highest reconfiguration throughput at the expense of a large resource utilization. Meanwhile, ZYCAP provides an acceptable reconfiguration throughput with a reduction in power consumption and resource utilization making it a good choice for DPR-based SDR designs.

2.6. Summary

In this chapter, Xilinx Zynq FPGA and DPR technique are introduced. Also, four partial reconfiguration controllers are presented and used to implement a high-speed reconfigurable SDR system targeting a Xilinx Zynq FPGA. A reconfigurable convolutional encoder is implemented as a benchmark to evaluate the performance of the four partial reconfiguration controllers. This experiment provides essential design guidelines for the DPR-SDR designers to choose the suitable PR controller based on their system requirements.

Chapter 3 : A Cost-Effective Automatic Partitioning Scheme For Dynamic Partial Reconfiguration Implementation On FPGA

3.1. Introduction

The DPR system consists of a number of reconfigurable modules (RMs). Each RM has a number of modes called RM mode that is changed in run time according to the DPR system operating modes. These RMs are physically implemented on specific locations on the FPGA defined by partially reconfigurable regions to isolate the reconfigurable parts of the system from the other non-reconfigurable parts.

Partitioning is the designing process to determine the number and size of reconfigurable regions for a DPR system according to the number and size of RM modes and the system configurations modes. When designing DPR systems, it is required to reduce the partitioning cost of the total reconfigurable regions area used by RMs and the reconfiguration time.

Reconfiguration time is an important factor in DPR system design which determines the required time to reconfigure the system from an operating configuration mode to another in run time. In present vendor DPR design tools the designer manually specifies the number and size of reconfigurable regions and the RM allocation to each reconfigurable region and hence the granularity of reconfiguration. Inefficient partitioning and RM allocation to the RRs can result in large unutilized resources area on the FPGA floorplan and a long reconfiguration time [26].

Therefore, efficient partitioning schemes should provide a reduction in the reconfiguration time and a high utilization of the area of the reconfigurable regions. The inter-connections or communication interfaces between the reconfigurable regions and the static part of the design are fixed during the reconfiguration because the routing an inter-connections take places in the static part of the design.

This chapter provides a cost-effective automatic partitioning scheme for an efficient DPR design flow to reduce the total partitions reconfiguration time and increase the utilization of the total partitions area. An improved open source heuristic partitioning algorithm [27] based on hierarchical clustering algorithm is used to determine the best partitioning scheme for a given DPR system. The method provides the optimum arrangement (numbers and sizes) of reconfigurable regions and the optimum RM allocation to the proposed reconfigurable regions. Also, a circuit routing communication architecture is implemented to establish dynamic routes between the reconfigurable regions itself and between the reconfigurable regions and the static design at the time of reconfiguration. For performance analysis, a number of DPR designs are used as benchmarks to evaluate the performance of the proposed partitioning scheme based on the total reconfiguration time of the DPR design and the total proposed partitions area.

3.2. Dynamic Partial Reconfiguration Design Flow

Currently, Xilinx and Intel (Altera) are the main FPGA devices vendors on the market. They provide a series of FPGA families that support the DPR design flow through their software tools. In this thesis, we consider only the Xilinx DPR design flow through Xilinx FPGA design software: Xilinx Vivado Design Suite or Xilinx PlanAhead tool.

In this section, an overview of the traditional DPR design flow is discussed. A proposed modified DPR design flow is presented to overcome the inefficiency of the traditional DPR design flow.

3.2.1. Xilinx DPR Design Flow

Xilinx DPR design flow [34] requires the dividing of the DPR system into static logic blocks and reconfigurable logic blocks. A DPR system consists of a number of operating modes known as configuration modes and each configuration mode contains a number of reconfigurable modules modes (RM) which are changed from a configuration mode to another at runtime. Figure 3.1 shows an example of a DPR system with four modes of configuration (Conf.1, Conf.2, Conf.3, and Conf.4).The configuration has a four reconfigurable module (A, B, C, D) with three modes: A1, A2, A3; B1, B2, B3; C1, C2, C3; D1, D2, D3.

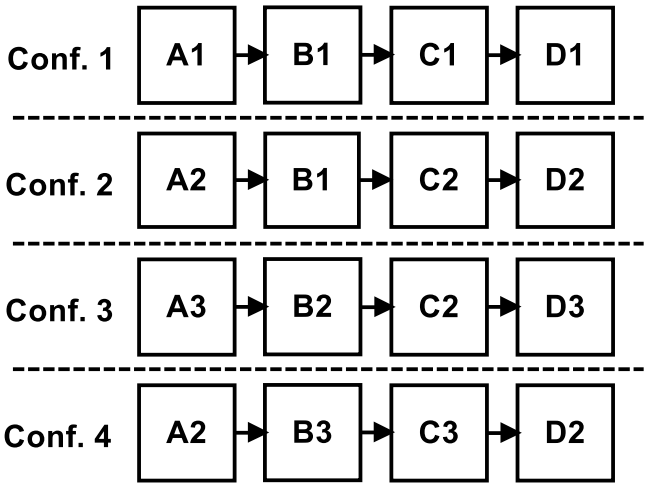


Figure 3.1: An example of DPR design with four modes of configuration and four reconfigurable modules per configuration (A, B, C, and D).

The DPR design flow is conducted by Xilinx tools through the following steps:

3.2.1.1. Synthesize The Reconfigurable Modules (RMs)

Xilinx Synthesis Technology Tool (XST) is used to synthesize all the RMs modes in all given DPR design configurations. The RMs modes are given to the synthesis tool as a set of HDL files to determine the numbers and types of resource requirements on the FPGA such as (LUTs, BRAM, and DSP blocks).

The output from the synthesis step is:

1) The design netlists e.g. (.NGC[1], .DCP[2]) files.
2) All the RM modes resource requirements (Slices, BRAM, DSP).

3.2.1.2. Define The Reconfigurable Regions (Partitioning)

Reconfigurable Region (RR) is a defined dynamically reconfigured area on the FPGA device to allocate the different RM modes for each configuration during runtime. The RR includes different types of resources according to the resource requirements of its allocated RM modes during all possible configurations. Xilinx DPR design flow lets the designers specify the sizes and numbers of the required RRs to all the RMs for all possible configurations manually without any kind of design automation or optimization techniques.

Two approaches are considered for manual partitioning [29]. The first approach, known as the single region partitioning approach which requires the allocation of all the system RMs into a single large size RR that holds the most resources required by the largest configuration in the system as shown in Figure 3.2. Each time a switching occurs from a configuration to another the whole area of the RR need to be reconfigured even in the cases of configurations that require fewer resources than the total RR area. Since the total RR area is reconfigured every time a new system configuration mode is loaded regardless of the number of resources required for each configuration, the reconfiguration time is fixed and equal to the reconfiguration time of the whole RR.

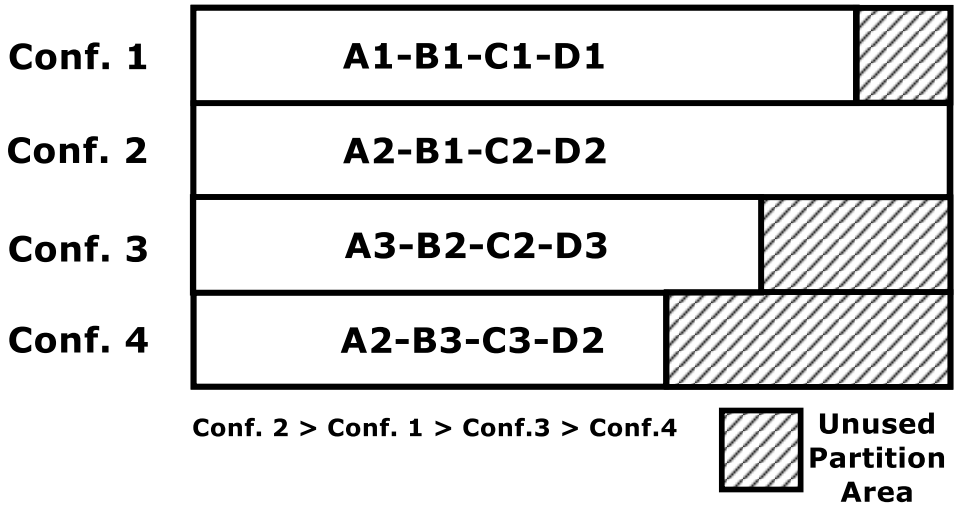


Figure 3.2: An example of a single partition region partitioning approach

Each configuration physically consumes the same constant RR area on the FPGA. As a consequence of this approach, the total required storage size of partial bitstreams on the memory is increased and it is equal to the largest configuration partial bitstream size multiplied by the number of system configuration modes. This approach has a significant advantage on area consumption since the single partition region area is equal to the

[1]Netlist Generated Constraints file supported by Xilinx ISE design tool.

[2] Design Check Point file contains the netlist of the design supported by Xilinx Vivado Design suit.

minimum required reconfigurable partition area for a DPR design which is the area of the largest configuration mode in the DPR design.

The second approach is the modular way for partitioning by assigning each RM into a single separate RR that could be reconfigured from one RM mode to another mode at runtime, known as single module per region partitioning approach. The number of RRs is equal to the number of RMs of the DPR system and the size of each RR is the size of the largest RM mode assigned to this RR. For example, in Figure 3.1 there are four RRs as the system consists of four RMs. The sizes of RRs are equal to the sizes of the largest mode of each RM in the system as in Figure 3.3 (RR1: A2; RR2: B1; RR3: C1; RR4: D1).

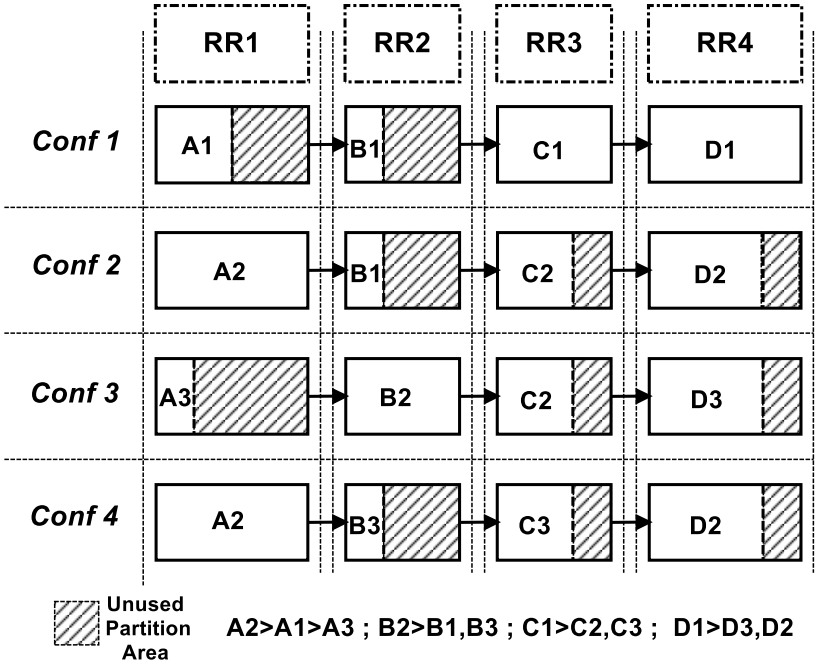


Figure 3.3: An example of one module per region partitioning approach

This approach offers a less reconfiguration time compared to the one single partition region approach since the switching from a configuration mode to another requires the reconfiguration of multiple small RRs for each RM that is changed without the need to reconfigure the whole RRs as in single region partitioning approach. The one module per region approach is the worst area-efficient due to the total RRs size is equal to the sum of the largest mode size for each RM in the system as shown in Figure 3.3. Therefore the percentage of area utilization is the lowest compared to any other partitioning approaches.

3.2.1.3. Floorplanning

The floorplanning step is done manually by the designer by arranging the set of RRs with the static partition into rectangular shapes on the FPGA floorplan. The inefficient distribution of RRs and static partition on the FPGA floorplan results to:

1) Inefficient utilization of the FPGA physical resource types.

2) Long wiring connections between RRs results in long input/output propagation delay between RRs.

The designer should be aware about the target FPGA physical architecture for efficient RR floorplanning. Manual floorplanning consumes a long design time and post place and route functional and timing simulations are needed to ensure that the floorplanned reconfigurable blocks met the required design constraints.

3.2.1.4. Implementation (Place and Route)

Implementation or (Place and Route) is done automatically by the tool using the design netlists and the (area, time and floorplanning) constraints generated from the above steps to map the design on the FPGA device.

Xilinx tool generates a list of implementation results for each design configuration known as the full configuration implementation result and a number of partial implementation results for each partial block (RMs) in the design.

3.2.1.5. Bitstream Generation

Finally, a full and partial bitstream sets are generated for different configurations and partial blocks (RMs) in the design. The FPGA target is initially configured by a full bitstream (A complete configuration image) and for runtime reconfiguration, partial bitstreams of the required RMs are loaded from an external memory.

3.2.2. Proposed DPR Design Flow

The Proposed DPR design flow is based on the regular Xilinx DPR design flow except for the step of partitioning as shown in Figure 3.4. The regular DPR design flow is modified by automating the partitioning step using a modified clustering partitioning algorithm proposed by [27,32].The partitioning algorithm tries to determine the optimum partitioning scheme by finding the optimum sizes and numbers of RRs for a given DPR system reducing the total reconfiguration time and the total RRs area.

In this proposed DPR design flow, the automatic partitioning algorithm assigns different modes from different RMs to the optimum RR to save the total partitions area and reduces the RMs reconfiguration time. As a consequence, the inter-connections between the RRs and the communication interfaces with the static region should be changed from a configuration to another [33].

For example, as shown in Figure 3.5, in conf.1, the output of RR1 (B1) is connected to the input of RR3 (C1) after switching to the conf.2 the output of RR1 (B1) should be connected to the input of RR2 (C2) to maintain the functionality of the DPR system. A proposed circuit routing switch is used to handle that problem by re-routing the input/output signals between the RRs during the reconfiguration. The circuit routing switch is a custom reconfigurable architecture implemented in the static region with a variable numbers and widths of input/output ports. The proposed tool introduces the term of ***" Virtual Flexible Reconfigurable Partition "*** by Combining the partitioning algorithm and the circuit routing switch together to virtually change the partition size during reconfiguration to fit with the RMs modes sizes as a solution to save the DPR design area as shown by the red dashed circles in Figure 3.5 for RM (A).

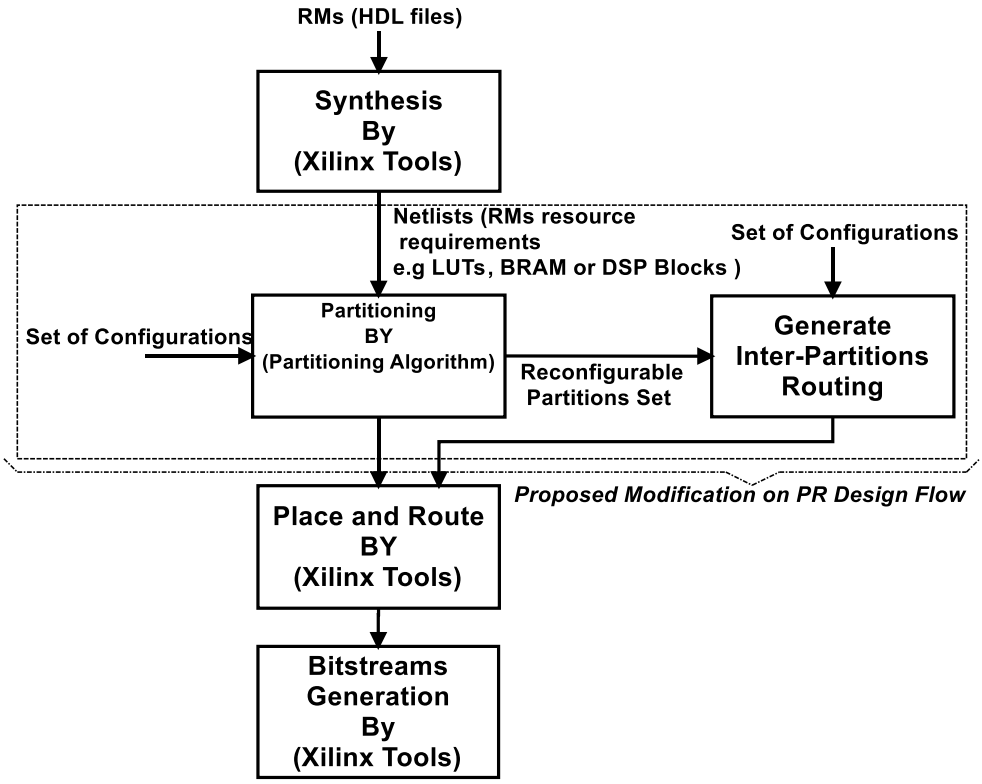


Figure 3.4: Dynamic Partial Reconfiguration design flow with the proposed partitioning tool flow.

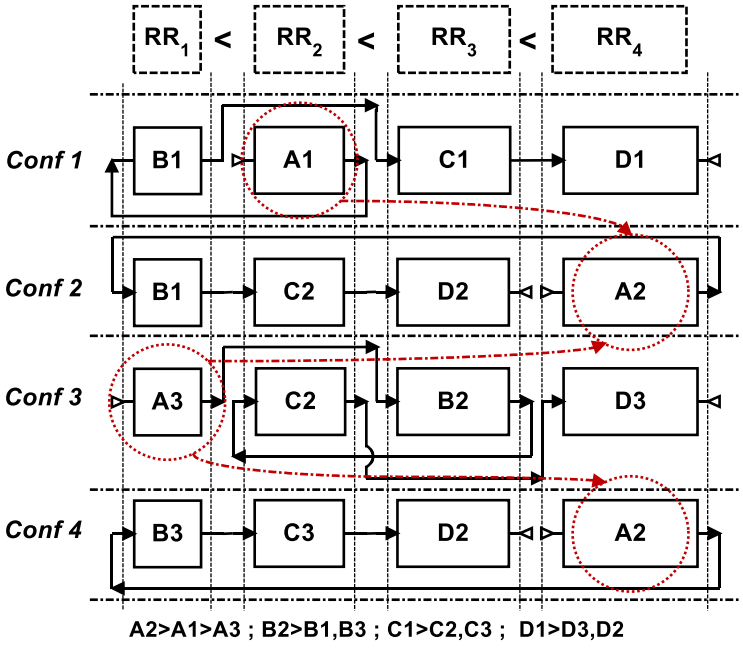


Figure 3.5: A Proposed Partitioning example where different modes belong to different RMs share the same RR and re-routing is required between RRs. The red dashed circles shows the concept of virtual flexible reconfigurable partition

3.3. Problem Formulation

Efficient DPR design implementation requires an efficient RR allocation or an efficient partitioning technique to minimize the total time of reconfiguration and the total used RR area. This section formulates the DPR partitioning requirements and the FPGA physical constraints that impact the partitioning process.

3.3.1. DPR Partitioning Requirements and Mathematical Formulation

Reconfiguration time and the total partitions area are the key requirements for any DPR applications in our case the SDR system. The SDR system should have a fast switching time between different wireless standard supported by the system and optimum hardware resource utilization for power and area saving.

The problem of partitioning can be described as follows:

1. Minimize total reconfiguration time.
2. Minimize total partitions area (Minimize total resource requirements).

The partitioning step design constraints:

1. All RRs of the design are fitting in the given FPGA device.
2. All required RMs in the design should have an allocated RR.
3. All the design configurations should be implemented.
4. The number of RRs should be greater than one (Case of the single region) and not exceed the number of RMs.
5. The minimum total partitions area of the design is the maximum area required by the largest configuration in the design.

The partitioning problem can be formulated mathematically to be solved and optimized using several types of partitioning algorithms [26]. The variables used in the mathematical formulation are shown in Table 3-1.

Table 3-1: Variable used in DPR Partitioning mathematical formulation

Variable	Description
$\boldsymbol{N}$	Total number of reconfigurable modules
$\boldsymbol{C}$	Set of design configuration
$\boldsymbol{R_{m,u}}$	Number of resource for module m in mode u
$\boldsymbol{A_{p,c}}$	Area of partition P in configuration c
$\boldsymbol{T_p}$	Reconfiguration time for partition P
$\boldsymbol{T_c}$	Reconfiguration time for configuration c
$\boldsymbol{T_{total}}$	Total reconfiguration time of the system
$\boldsymbol{A_{total}}$	Total partitions area

The maximum number of resources for module m is equal the maximum number of resources used among the different modes of module m. For example, in Figure 3.1 module modes A1, A2, A3 are three modes of reconfigurable module A.

$$R_{m\,MAX} = MAX_u\,(R_{mu}) \quad (1)$$

Different modes of module m are allocated to a reconfigurable region partition with area $A_{p,c}$ in configuration c , in some cases when applying partitioning algorithm multiple modes of different modules can be merged into a single partition region. The area of the partition region is the summation of all modules in region p.

$$A_{p,c} = \sum_{u,m} R_{u,m} \quad c = 1,2,\ldots.C \quad (2)$$

For example in Figure 3.3, $A_{1,1} = Area\ of\ (A1), A_{1,2} = Area\ of\ (A2)$, and in Figure 3.1 for single region partition $A_{1,1} = Area\ of\ (A1 + B1 + C1 + D1), A_{1,2} = Area\ of\ (A2 + B1 + C2 + D2)$

Reconfigurable partition area is the maximum partition area required for a set of modules modes among the different configurations

$$A_p = MAX_c\,(A_{p,c}) \quad (3)$$

Total partitions area is the summation of all reconfigurable partitions in the DPR design.

$$A_{total} = \sum_p A_p \quad (4)$$

Time of reconfiguration is calculated by the time spent to reconfigure the partition region reconfigurable frames number f . Time of configuration of a reconfigurable partition is given by:

$$T_p = \frac{A_p}{f} * (reconfiguration\ time\ of\ one\ frame) \quad (5)$$

Each configuration in the design consists of multiple partitions the reconfiguration time of configuration is given by:

$$T_c = \sum_p T_p \quad (6)$$

Total reconfiguration time is the summation of all design configurations reconfiguration time as given by:

$$T_{total} = \sum_c T_c \qquad c = 1,2,3\ldots.C \quad (7)$$

The objective is to improve the DPR system performance by minimizing the total partitions area and minimizing the total reconfiguration time.

$$Minimise\ (A_{total}) \quad (8)$$

$$Minimise\ (T_{total}) \quad (9)$$

3.3.2. FPGA Partitioning Physical Constraints

In 7-series Xilinx FPGAs there are several physical constraints on the size and shape of RR [34, 35]. These constraints can result in an inefficient resource utilization over the FPGA floorplan. Therefore, these constraints affect negatively the total reconfiguration time and total partitions area of a DPR design and the whole performance of the system.

As shown in Figure 3.6, the RR shape should be a rectangular shape that extends vertically or horizontally over the rows and columns within the same clock region of the FPGA. In the FPGA floorplan the columns span by the entire device height, the column contains a number of resource blocks of the same type e.g (CLB, BRAM or DSP) blocks. Also, the FPGA contains columns of interconnections blocks to connect resource columns with each other. The intersection between a column and a row is called a tile: CLB tiles, BRAM tiles, and DSP tiles. In 7-series Xilinx FPGA one CLB tile contains 50 CLB block each CLB block contains 2 Slices each Slice is 4 LUTs, one BRAM tile contains 10 BRAM block of 36Kbit and one DSP tile contains 10 DSP blocks.
One tile is the minimum reconfigurable size for 7-series Xilinx FPGA. The tile has a rectangular shape of one-row height by one column wide. The tile is a 36 reconfigurable frame of 101X32 bits.

Therefore, the minimum partition size is the size of one tile (50 CLB or 10 BRAM or 10 DSP). The partition size is equal an integer number multiplied by the tile size. For example: 1, 2, ... N tiles.

Also, the edge of partitions should be a resource block not allowed to make an interconnection block a partition edge. The interfaces between the static part and partitions called partition pins take some resources (Bus Macros) inside the defined partitions and should be taken into consideration while determining the partition size.

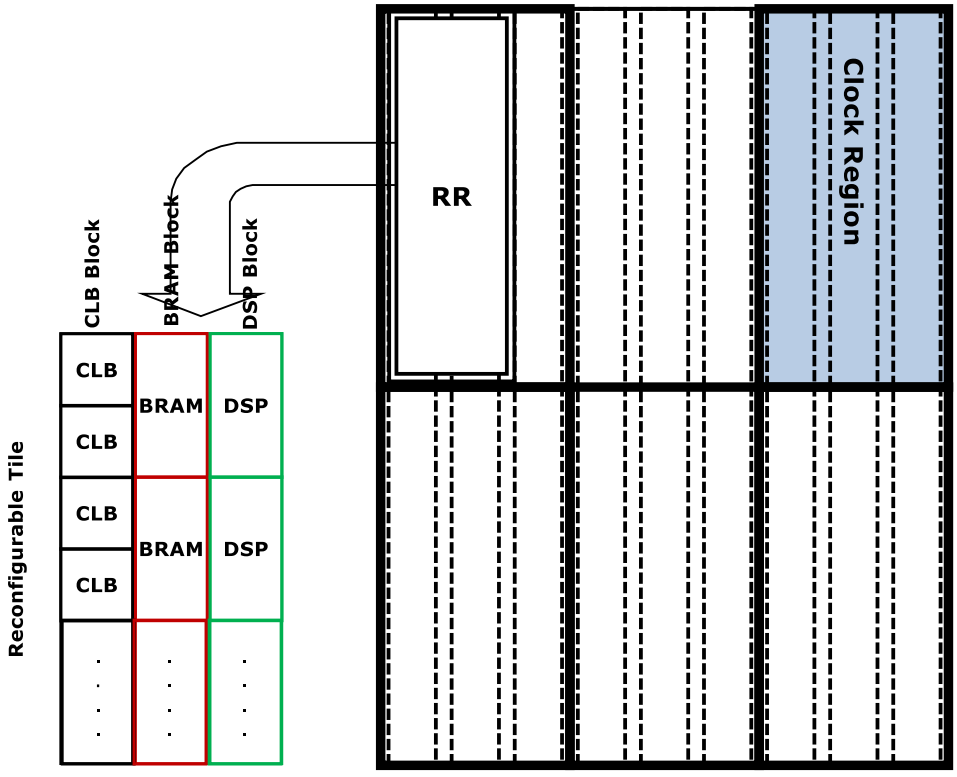


Figure 3.6: Xilinx 7-series reconfigurable partition physical constraints

3.4. Automatic Partitioning Algorithm

Several works propose methodologies and techniques to solve the partitioning problem or to develop an automated tool to optimize the reconfigurable region allocation for partial reconfiguration (PR) designs. Many types of optimization and task partitioning algorithms are taken from other research trends to find an efficient solution for the PR partitioning case. In [26] an optimization method based on integer linear programming (ILP) is proposed for automatically determine the optimal partitioning solutions with the minimum reconfiguration time and minimum partitions area. The PR system is represented by a number of mathematical equations to be solved. A reasonable improvement in area and reconfiguration time is achieved.

An automated partitioning tool is proposed by [27] based on a modified hierarchical clustering algorithm. The tool finds the optimal partitioning schemes with the minimum reconfiguration time and no optimization is done for the area of reconfigurable regions. This partitioning tool reduces the reconfiguration time by a significant percentage factor but at the same time, the reconfiguration area increased. CoPR is a PR design tool flow introduced in [28] for the design of the adaptive reconfigurable system. The CoPR tool includes: 1) automated partitioning scheme 2) Partition interface generator 3) automated floorplanning algorithm. The tool accepts a high-level description of an adaptive system and implements it on the FPGA automatically.

A simulated annealing based algorithm is proposed by [29] to determine modules allocation to reconfigurable regions based on minimizing the partitions area. The reconfiguration time is not be considered in that algorithm. An automated Hardware/Software (HW/SW) PR partitioning tool flow for HW/SW co-design systems is introduced in [30] by allocating tasks or reconfigurable modules to a reconfigurable partition region on the FPGA or to be executed by a microprocessor. In [31], a run time temporal partitioning is proposed to reduce the reconfiguration time. An inter-module communication is implemented to re-communicate the reconfigurable partition regions during reconfiguration to enable temporal partition assembly.

In this chapter, an open source modified hierarchical clustering algorithm proposed in [27, 31] is used for the automatic partitioning phase in the proposed DPR tool flow. In this section, two proposed partitioning solution are studied based on the reduction in total time of reconfiguration and total partitions area.

3.4.1. Modified Hierarchical Clustering Algorithm

A modified hierarchical clustering algorithm [31] with an agglomerative strategy is used for partitioning process. First step is to check for the implementation feasibility by comparing the FPGA device resources number with the resources required to implement the largest configuration of the DPR system. Hence, the minimum possible area required for a DPR system is the area of the largest configuration of the DPR system. If the implementation is feasible on the selected FPGA device, a connectivity matrix of $N \times M$ is generated with a number of rows (N) represent the number of configurations and a number of column (M) represent the number of reconfigurable modules in all the modes of configurations. An element (i, j) in the matrix with a value 1 represents a reconfigurable module mode j in configuration i as shown in the following example.

For the example design in Section 3.2.1 Figure 3.1:

$$Conf.1: \quad A_1 \rightarrow B_1 \rightarrow C_1 \rightarrow D_1$$

$$Conf.2: \quad A_2 \rightarrow B_1 \rightarrow C_2 \rightarrow D_2$$

$$Conf.3: \quad A_3 \rightarrow B_2 \rightarrow C_1 \rightarrow D_3$$

$$Conf.4: \quad A_2 \rightarrow B_3 \rightarrow C_3 \rightarrow D_2$$

The connectivity matrix will be:

$$\begin{array}{c} \\ Conf.1 \\ Conf.2 \\ Conf.3 \\ Conf.4 \end{array} \begin{array}{c} (A_1 \ A_2 \ A_3 \ B_1 \ B_2 \ B_3 \ C_1 \ C_2 \ C_3 \ D_1 \ D_2 \ D_3) \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

The connectivity matrix is used to determine two weights variables for the clustering partitioning optimization algorithm. The $node\ weight$ of reconfigurable mode is the number of times that module appears in the possible configurations and it is equal the sum of the required module column in the connectivity matrix. For module A_1 the node weight is equal 1 and for B_1 is equal 2. Second weight is the $edge\ weight$ $(W_{i,j})$ which is the number of times a two module occur concurrently in the possible configurations. For module A_1, B_1 the edge weight is 1 and A_2, D_2 the edge weight is 2. All the node and edge weights are calculated from the connectivity matrix to build a graph network as shown in Figure 3.7.

The algorithm first checks for a complete subgraph in the network where every pair of distinct vertices is connected by a unique edge. Initially, all the nodes are not connected and the number of edges, $K = 0$. The algorithm iterates and at every iteration the two nodes with the highest edge weight are linked. Highest edge weight indicate that the two nodes represent two modules occur concurrently more frequently in the given system configurations and should be merged in the same partition region. This is shown in Figure 3.7 (a, b) a sub-graph of $K = 1$ and sub-graph of $K = 4$ for $onf.1$.

Each sub-graph represent a base partition which is the set of module clusters that can be used to determine the final partitions set. The frequency of a single base partition occurrence in a configuration is called $requency\ weight$. Frequency weight is the smallest edge weight in a sub-graph. For example in Figure 3.7 (b) the frequency weight of a sub-graph $\{A_1, B_1, C_1, D_1\}$ is 1 and in Figure 3.7 (c) for sub-graph $\{A_2, B_2, C_3, D_2\}$ is 2. The algorithm iterates until all the possible links are connected and the graph network is completed. The result of this step is a set of base partitions generated from all sub-graphs of the clusters network. Table 3-2 lists all the base partitions for the example used here.

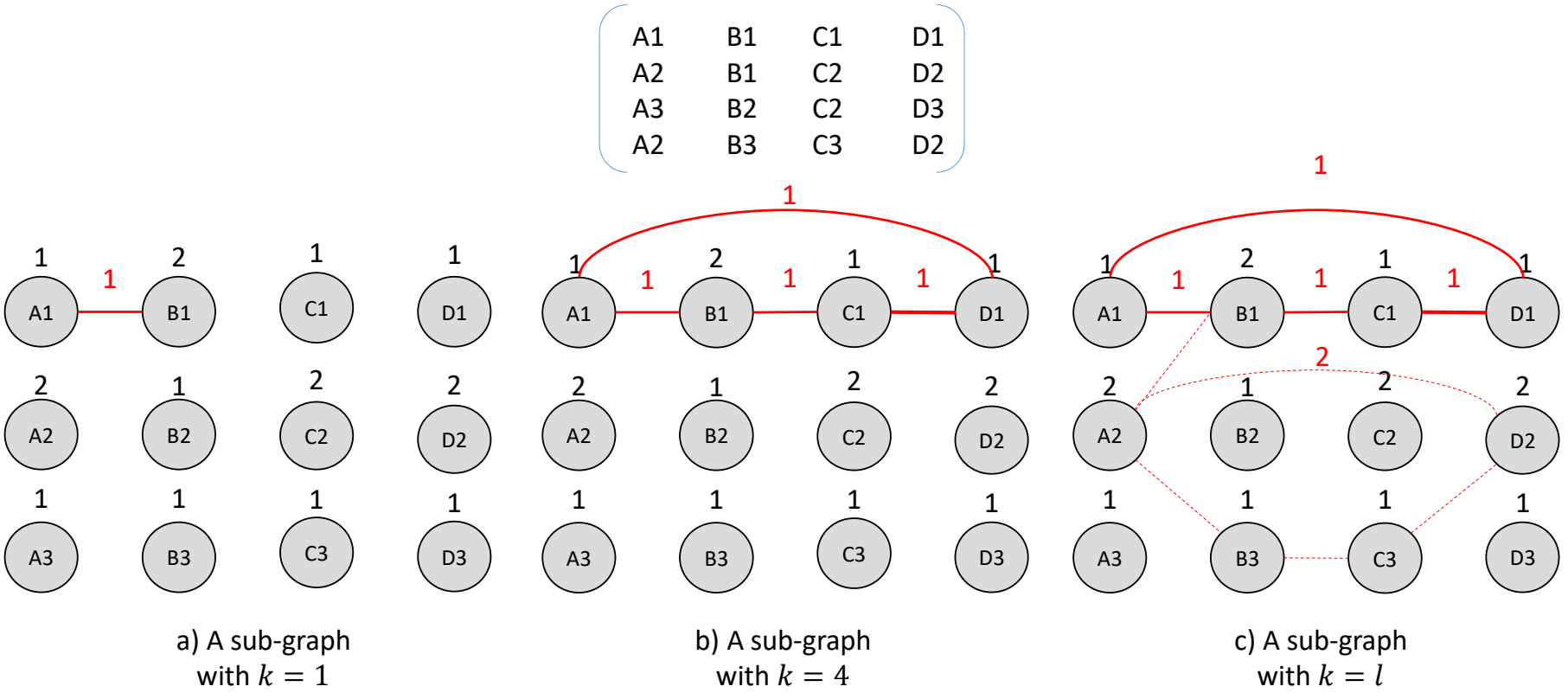


Figure 3.7: Example of Connectivity Graph Network

Table 3-2: Base Partitions

Base Partition	Freq Weight	Base Partition	Freq Weight	Base Partition	Freq Weight
$\{A_1\}$	**1**	$\{A_2, C_2\}$	**1**	$\{A_1, B_1, D_1\}$	**1**
$\{A_3\}$	**1**	$\{A_2, D_2\}$	**2**	$\{D_1, B_1, C_1\}$	**1**
$\{B_2\}$	**1**	$\{B_1, C_2\}$	**1**	$\{A_2, B_1, C_2\}$	**1**
$\{B_3\}$	**1**	$\{B_1, D_2\}$	**1**	$\{D_2, B_1, C_2\}$	**1**
$\{C_1\}$	**1**	$\{C_2, D_2\}$	**1**	$\{A_2, B_1, D_2\}$	**1**
$\{C_3\}$	**1**	$\{A_3, B_2\}$	**1**	$\{A_3, B_2, C_2\}$	**1**
$\{D_1\}$	**1**	$\{A_3, C_2\}$	**1**	$\{A_3, C_2, D_3\}$	**1**
$\{D_3\}$	**1**	$\{A_3, D_3\}$	**1**	$\{B_2, C_2, D_3\}$	**1**
$\{A_2\}$	**2**	$\{B_2, C_2\}$	**1**	$\{A_2, B_3, C_3\}$	**1**
$\{B_1\}$	**2**	$\{B_2, D_3\}$	**1**	$\{A_2, C_3, D_2\}$	**1**
$\{C_2\}$	**2**	$\{C_2, D_3\}$	**1**	$\{B_3, C_3, D_2\}$	**1**
$\{D_2\}$	**2**	$\{A_2, B_3\}$	**1**	$\{A_1, C_1, D_1\}$	**1**
$\{A_1, B_1\}$	**1**	$\{A_2, C_3\}$	**1**	$\{A_2, C_2, D_2\}$	**1**
$\{A_1, C_1\}$	**1**	$\{B_3, C_3\}$	**1**	$\{A_1, B_1, C_1, D_1\}$	**1**
$\{A_1, D_1\}$	**1**	$\{B_3, D_2\}$	**1**	$\{A_2, B_1, C_2, D_2\}$	**1**
$\{C_1, B_1\}$	**1**	$\{C_3, D_2\}$	**1**	$\{A_3, B_2, C_2, D_3\}$	**1**
$\{D_1, B_1\}$	**1**	$\{C_1, D_2\}$	**1**	$\{A_2, B_3, C_3, D_2\}$	**1**
$\{A_2, B_1\}$	**1**	$\{A_1, B_1, C_1\}$	**1**		

Once base partitions are generated as in Table 3-2, a covering algorithm is used to select *candidate base partitions* for optimum partitioning. The objective is to minimize the reconfiguration time and total partitions area. So, base partitions are prioritized based on the number of modules, if two base partitions have the same number of modules they are arranged in ascending order of their frequency weight values. Therefore candidate base partitions are selected from the highest base partitions frequency weights.

For each configuration represent a row in the connectivity matrix the corresponding modules in the selected base partition for example base partition $\{A_2, D_2\}$ in $conf.4$ the corresponding location of $\{A_2, D_2\}$ in the matrix are set to zero.

$$(0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0)$$

Each base partitions in the table are selected and compared to the connectivity matrix until all element of the matrix become zero. A candidate base partition is selected if the modules of a selected base partitions can cover all the possible configuration. For example a candidate partition set of base partitions:

$$(\{A_1\}, \{A_2\}, \{A_3\}, \{C_1, B_1\}, \{B_2, C_2\}, \{B_3, C_3\}, \{D_1\}, \{D_2\}. \{D_3\})$$

Can cover all the possible configurations of the system.

Next step, the algorithm search for *a compatible set of partitions* from candidate partitions set. Two partitions or more are compatible if the modules contained in them do not co-occur or contradict in any configuration of the system, in other words, each partition in the compatible partitions set should contains different modules from different configurations. For example $\{C_1, B_1\}, \{B_2, C_2\}, \{B_3, C_3\}$ are compatible partitions while $\{A_1, B_1\}$, $\{C_1, B_1\}$ are not compatible.

Region allocation starts by allocating each compatible base partitions set from the selected candidate partition set in a reconfigurable region. Minimizing reconfiguration time required the calculation of the cost function of each compatible base partitions in candidate partition set to choose the compatible partition of the minimum cost function. Cost function of assigning two base partitions or more of a compatible base partitions set is calculated by the number of reconfigurable frames required by the maximum base partition area from the selected candidate base partitions as

$$A_r = \max(A_{base\ 1}, A_{base\ 2}, \ldots\ldots, A_{base\ n}) \qquad (10)$$

Where, A_r is the reconfigurable region area calculated by the number of reconfigurable frames and $A_{base\ n}$ is the area of base partition n.

Total reconfiguration time is the sum of all possible configuration transition in the system. In the worst case, total reconfiguration time is the sum of transitions from all configurations to all other configurations. Reconfiguration time $tcon_r$ for a reconfigurable region assigned to a candidate partitions set is proportional to the reconfigurable region area.

$$tcon_r \propto A_r \qquad (11)$$

Total reconfiguration time is calculated as:

$$t_{total} = \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} tcon_{i,j} \qquad j > i \qquad (12)$$

Where c is the total number of configurations and $tcon_{i,j}$ is the reconfiguration time to switch from configuration i to configuration j.

$$tcon_{i,j} = \sum_{r=1}^{N} d_{i,j} \times tcon_r \qquad (13)$$

Where N is the total number of reconfigurable region per configuration and $d_{i,j}$ is the probability of transition from configuration i to configuration j which is equal to 1 in case of transition and 0 in no transition.

To compute all the reconfiguration time of all compatible base partitions sets the algorithm iterates by assigning new compatible base partitions to a region and calculate the required resources number. If all possible compatible partitions from a selected candidate partitions are done the algorithm select a new candidate partition and assign new compatible base partitions and calculates their reconfiguration time. The algorithm iterates until all candidate partitions are finished and no more candidate partitions are possible.

Finally, the covering algorithm select the compatible base partitions sets based on the lowest reconfiguration time cost or lowest resources requirements. Figure 3.8 shows a flow chart of the steps of partitioning algorithm. Modified clustering algorithm find a sub-optimal solution whereas a defined configuration transition is required as an input to the algorithm from the beginning and if all possible combination of transitions were considered the problem of partitioning would become an N-P hard problem.

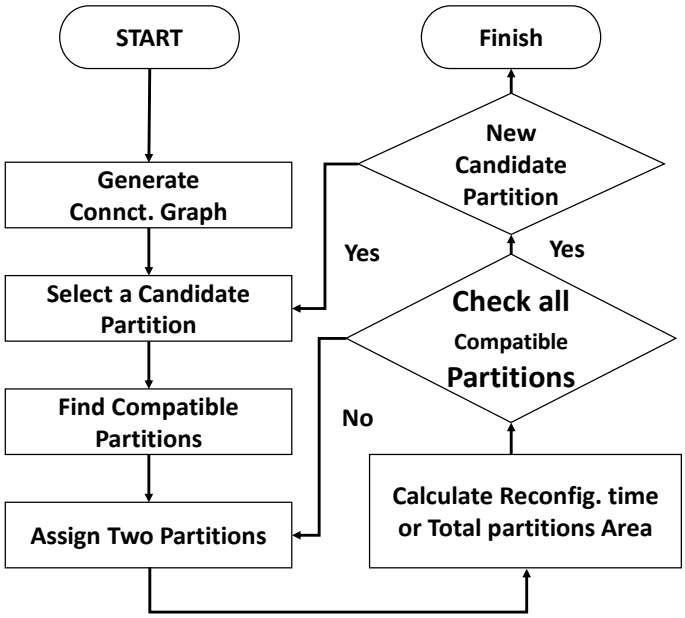


Figure 3.8: Partitioning Algorithm Flow Chart

3.4.2. Merging of Consecutive Reconfigurable Modules

The partitioning algorithm gives two kinds of sub-optimal solutions based on the selection of candidate base partitions and the type of merging of reconfigurable modules in the base partitions included in the selected base partitions. First sub-optimal solution (Solution-One) is the merging of consecutive reconfigurable modules in the same base partition. Figure 3.9 shows an example of this solution. In ***(Solution-One)*** selection of base partitions that contain merged modules reduces the total reconfiguration time of the system compared with the selections of base partition with one module in the partition. Excluding the base partitions of merged non-consecutive modules from selection could facilitate the routing process between modules during implementation.

3.4.3. Merging of Non-Consecutive Reconfigurable Modules

In some cases of DPR designs, Solution-One minimizes the total reconfiguration time by assigning a number of large reconfigurable modules in static partitions as an optimum solution to reduce the reconfiguration time. Therefore, the total area is increased due to some of the reconfigurable modules are not always active in all possible configurations which lead to a reservation of unutilized area during runtime. To minimize the total area and at the same time minimize the total reconfiguration time, the selection of candidate partitions set includes the base partitions that contain mergeable non-consecutive reconfigurable modules ***(Solution-Two)*** for example {A2-C2} shown in Figure 3.10. ***(Solution-Two)*** takes a long runtime compared to ***(Solution-One)*** since the number of iterations is increased to find the optimal compatible partitions set from the candidate partitions set which is augmented by adding the new mergeable partitions in the set.

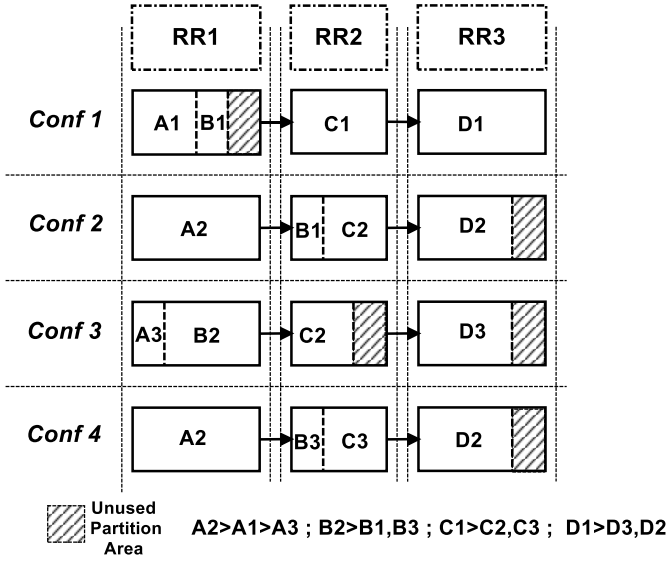


Figure 3.9: Merging of Consecutive RMS *(Solution-One)*

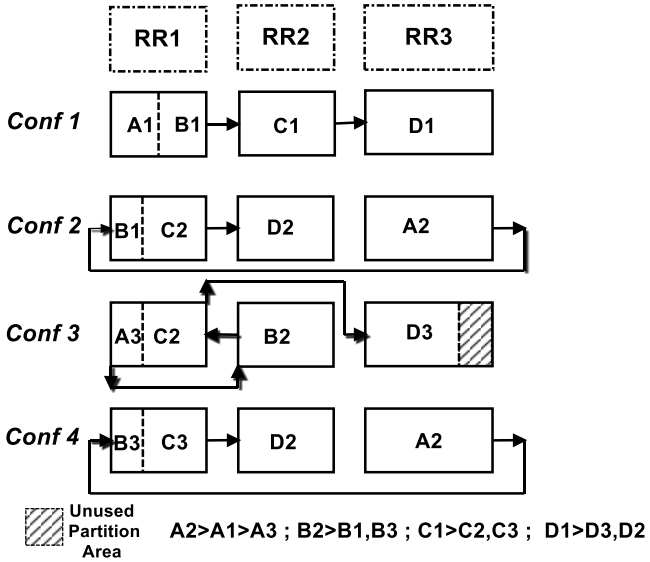


Figure 3.10: Merging of Non-Consecutive RMs *(Solution-Two)*

3.5. Dynamic Interconnection For Reconfigurable Modules

Routing or communications between RRs are modified in the proposed partitioning algorithms since the optimal partitioning solution is a set of partitions that different modes from different RMs share the same RR in different configurations as shown in Figure 3.5. A rerouting technique between RRs is needed to maintain the validation of the design data flow. A proposed routing switch is shown in Figure 3.11. The routing switch is implemented with a number of multiplexers and decoders in the static region of the DPR design. The switch input ports are connected to all RR output ports and the switch output ports are connected to all RR input ports. A finite state machine (FSM) is

designed to control the routing switch by sending selection signals to the switch to change the routes between input ports and output ports in the time of reconfiguration. The routing switch has a map of all RM modes locations for every configuration in the DPR system.

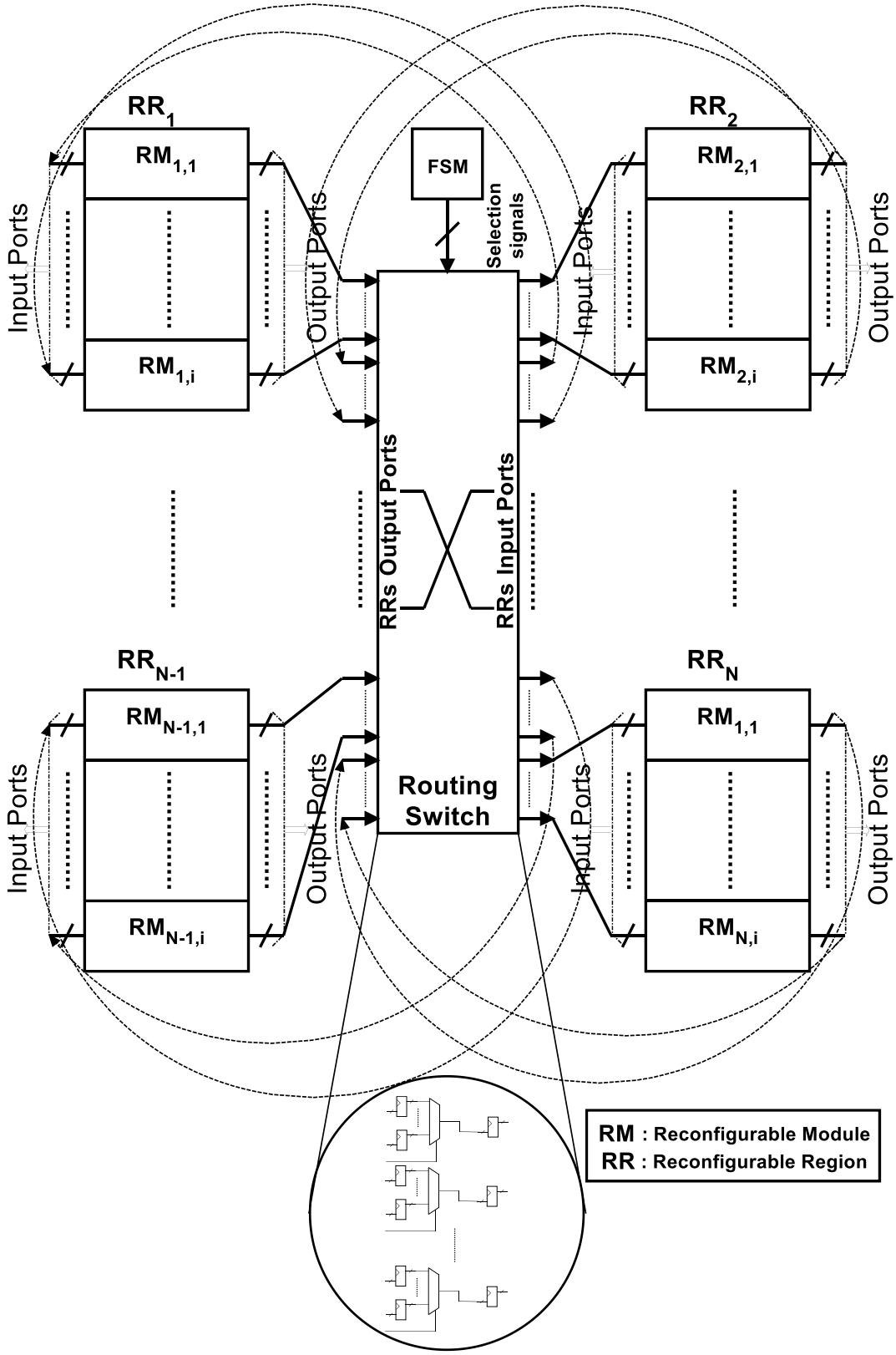


Figure 3.11: Circuit routing switch For Partitions Interconnections

3.6. Performance Evaluation of Different Partitioning Schemes

The objective of this section is to measure the performance of the four partitioning methods described in the previous sections. Specifically, the solutions provided by the partitioning algorithm that gives the possibility of merging non-consecutive RM modes in one RR. The performance analysis is based on 1) the total reconfiguration time 2) The total partitions area used by the DPR system.

A number of DPR designs are required as benchmarks to evaluate the performance of partitioning methods. Unfortunately, a few number of DPR designs are available in the literature. Hence 16 synthetic DPR designs are generated to test the partitioning algorithms. The 16 designs are generated as shown in Table 3-3. Design configurations are generated randomly in each DPR design until every RM mode in the design is utilized.

Figure 3.12 shows a comparison of the total reconfiguration time for single module per region scheme, single region scheme, partitioning algorithm ***(Solution-One)*** and partitioning algorithm ***(Solution-Two)***. Total reconfiguration time of the single module per region partitioning scheme is the highest since the switching between configurations requires the reconfiguration of all RM reconfigurable regions, and for that method, the RR size for each RM is equal to the size of the largest mode belong to that RM. Therefore, the total partitions size to be reconfigured is equal to the summation of the largest mode size for each RM in the design.

The total reconfiguration time for partitioning algorithm ***(Solution-Two)*** is better than partitioning algorithm ***(Solution-One)*** in all DPR designs cases. Since the optimum solution of merging some of the non-consecutive RM modes which are in the same configuration into a single partition saves the total reconfigured area and correspondingly the time for reconfiguration is reduced. Moreover, from

Figure 3.12 (c, d) when the number of RMs per configuration decreases, the total reconfiguration time for the two partitioning algorithm solutions is approximately the same since the merging of non-consecutive RM modes in that design cases does not lead to an impressive reduction in the total reconfigured partitions area.

The two Partitioning algorithm solutions improve the total reconfiguration time by 30% to 40% compared to the traditional partitioning schemes. Figure 3.13 shows a comparison of the total partitions area for the four partitioning methods. The single region partitioning method is the least partitions area among the other methods since the minimum partitions area of a DPR design is the area of the largest configuration of the DPR design. Partitioning algorithm ***(Solution-One)*** consumes a total partitions area more than the single region scheme in all the DPR design cases.

The total partitions area is reduced by allowing partitioning algorithm to select the optimum solutions of merging some non-consecutive RM modes in the same configuration into the same RR as in the partitioning algorithm ***(Solution-Two)***. In addition, from Figure 3.13 (c, d) at design cases with a small number of RMs the total partitions area obtained by the proposed algorithm is approximately equal the minimum partitions area that a DPR designs could reach.

Table 3-3: Random Generated DPR Designs

Design	Configurations	Design	Configurations	Design	Configurations	Design	Configurations
1	A1-B3-C3-D4-E3-F4	**5**	A1-B1-C1-D3-E1	**9**	A4-B1-C1-D2	**13**	A4-B4-C4
	A4-B1-C1-D4-E1-F2		A2-B4-C2-D4-E3		A1-B2-C4-D1		A1-B1-C2
	A2-B3-C1-D4-E4-F3		A2-B3-C3-D1-E1		A2-B3-C4-D4		A3-B3-C4
	A2-B1-C3-D1-E1-F1		A4-B1-C4-D4-E4		A2-B2-C4-D3		A2-B2-C4
	A2-B3-C4-D2-E1-F2		A1-B2-C1-D2-E2		A3-B4-C3-D4		A2-B3-C4
	A3-B4-C2-D2-E4-F3		A3-B3-C3-D3-E2		A2-B4-C2-D3		A3-B2-C3
2	A2-B3-C1-D3-E2-F1	**6**	A2-B2-C4-D3-E3	**10**	A4-B1-C2-D1	**14**	A4-B2-C4
	A2-B3-C4-D4-E4-F4		A1-B2-C3-D2-E2		A1-B2-C2-D2		A1-B3-C3
	A4-B4-C3-D2-E1-F3		A2-B4-C2-D3-E2		A4-B2-C3-D4		A3-B2-C3
	A4-B2-C4-D1-E2-F2		A3-B4-C3-D4-E1		A3-B3-C1-D3		A4-B1-C1
	A1-B3-C2-D4-E2-F3		A4-B1-C2-D1-E4		A4-B3-C4-D4		A4-B2-C2
3	A4-B2-C3-D2-E4-F3	**7**	A4-B3-C3-D3-E4	**11**	A2-B4-C4-D1	**15**	A1-B2-C1
	A2-B1-C2-D4-E1-F1		A3-B3-C4-D3-E2		A1-B1-C3-D4		A3-B4-C2
	A2-B4-C2-D3-E2-F1		A4-B2-C3-D3-E3		A3-B2-C3-D3		A3-B3-C3
	A3-B4-C2-D1-E1-F3		A1-B1-C2-D4-E3		A2-B2-C4-D3		A2-B2-C4
4	A3-B1-C4-D3-E1-F2	**8**	A3-B4-C2-D3-E1	**12**	A3-B2-C4-D4	**16**	A4-B3-C4
	A3-B4-C3-D2-E3-F4		A4-B3-C1-D3-E2		A4-B2-C1-D3		A4-B1-C4
	A1-B2-C2-D3-E4-F2		A1-B1-C1-D1-E3		A4-B1-C1-D4		A1-B3-C1

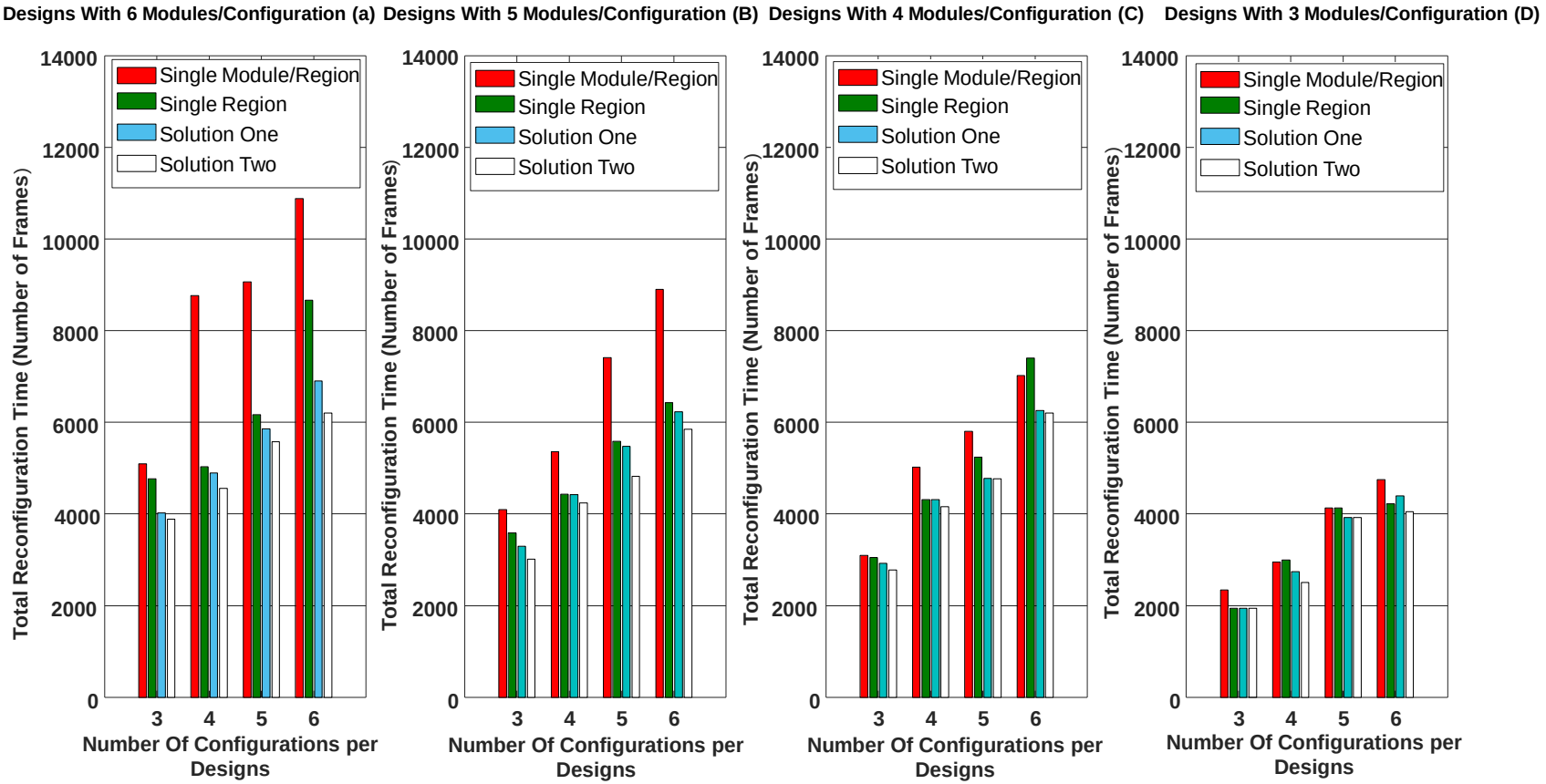


Figure 3.12: Total reconfiguration time for the four partitioning methods according to DPR designs with (a) 6 RMs,(b) 5 RMs,(c) 4 RMs,(d)3 RMs per configuration.

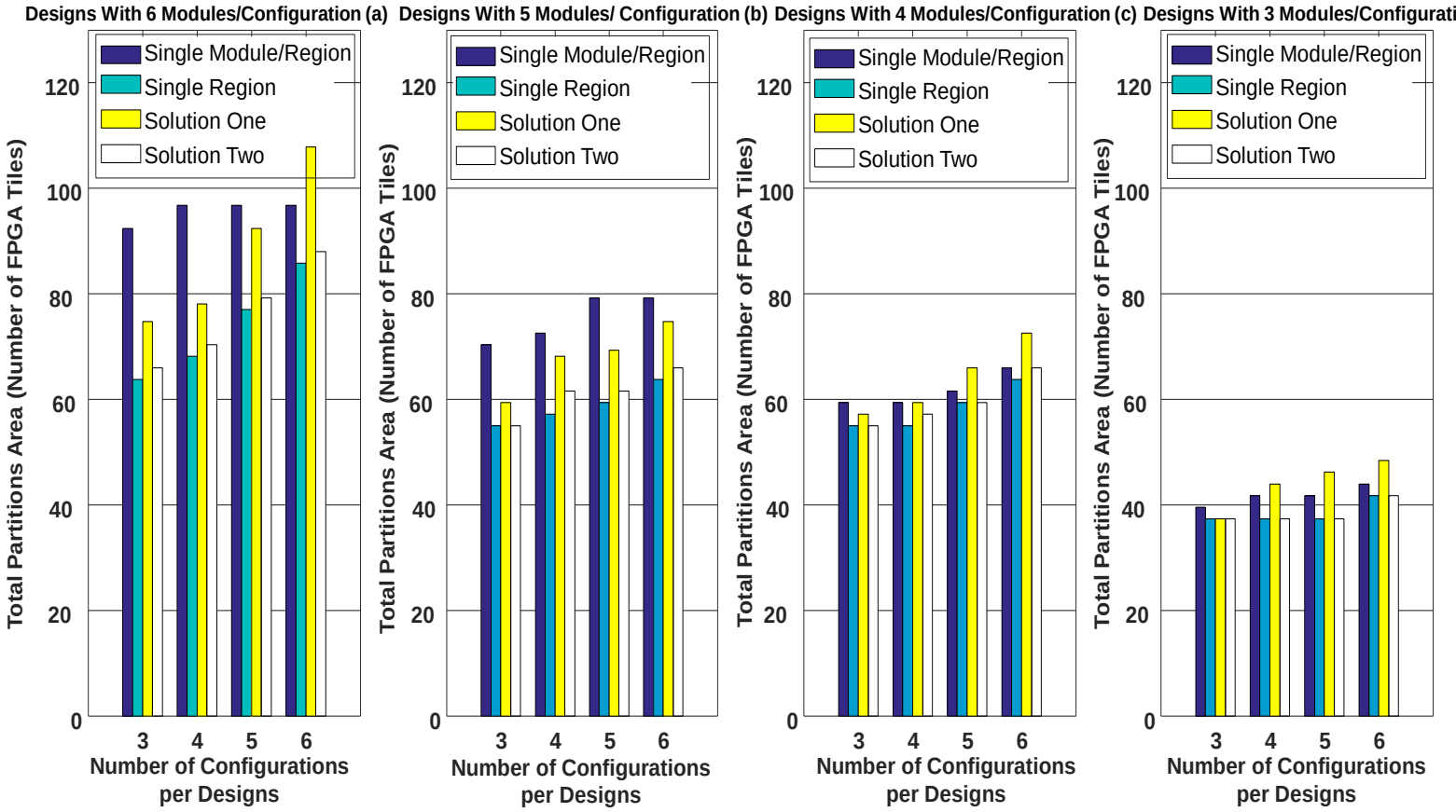


Figure 3.13: Total partitions area for the four partitioning methods according to DPR designs with (a) 6 RMs, (b) 5 RMs, (c) 4 RMs, (d) 3 RMs per configuration.

The percentage of improvement of the partitioning algorithm ***(Solution-Two)*** compared to ***(Solution-One)*** is shown in Figure 3.14. The partitioning algorithm ***(Solution-Two)*** improves the total reconfiguration time in average by ~5% and improves the total partitions area in average by ~10% compared to ***(Solution-One)***. As the number of RMs increase in DPR design, the partitioning algorithm ***(Solution-Two)*** reaches a significant improvement percentage of ~ 20% over ***(Solution-One)***. For example, by applying the partitioning algorithm on a DPR design with 8 RMs and 10 configurations the total reconfiguration time is improved by a 21% and the total partitions area is improved by 16% compared to traditional methods. It's clear that partitioning algorithm offers a significant improvement by reducing the cost of partitioning of DPR designs with a large number of RMs and configurations.

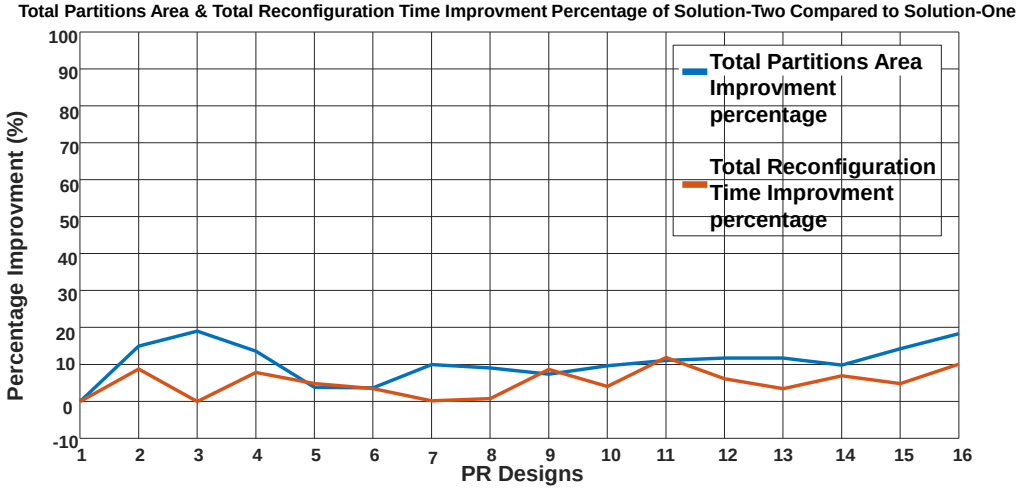


Figure 3.14: Percentage of performance improvement of the Solution-Two compared to Solution-One.

3.7. Summary

The techniques of partitioning and the allocation of RMs to RRs have a direct impact on the performance of DPR systems. A performance evaluation of four partitioning techniques is presented in this chapter. Reconfiguration time and total reconfigurable area are the main parameters to evaluate a partitioning scheme. It is demonstrated that manual techniques of partitioning are not suitable for time critical DPR systems. A modified DPR tool flow is proposed in this chapter to improve the time of reconfiguration by 40% compared to the traditional flow and at the same time reducing the design area by virtually changing the RRs size during reconfiguration to be fit with the size of active RMs using a routing switch to reconnect the RRs according to the active configuration mode.

Chapter 4 : Reconfigurable Hardware Platform for SDR System

4.1. Introduction

This chapter proposes a novel reconfigurable hardware platform for an SDR system that supports multiple wireless communication standards. The reconfigurable multi-standard SDR hardware platform is implemented using DPR techniques on Xilinx Zynq hybrid FPGA. In this chapter, three different wireless communication transmitter chains (3G, WIFI, and LTE) are supported by the proposed reconfigurable SDR system.

Two DPR implementation techniques are proposed in this chapter. The first implementation is done using the single partition region partitioning approach, the second implementation is done by applying the partitioning algorithm described in Chapter 3. The two proposed DPR SDR implementations are designed and evaluated using the ZC702 Xilinx Zynq development board [43].

A performance evaluation between the proposed implementations is done based on the total time of reconfiguration, total power consumption and the total partitions area of the three modes of configuration of the proposed reconfigurable SDR system. The FPGA reconfiguration is done internally through the ICAP using the high throughput Xilinx partial reconfiguration controller (Xil-PRC).

4.2. Multi-Standard SDR System Transmitter Chain

3G, WIFI, and LTE are widely used wireless communications standards. Implementing all these standards in parallel as each standard has its own hardware resources results in larger silicon area and higher power consumption. Commercial available SDR system platforms are reprogrammed by software routines that run on fixed hardware platforms like General-Purpose Processors (GPPs) or Digital Signal Processors (DSPs). Obviously, GPPs and DSPs are not suitable reconfigurable hardware platforms for the high data rate and low power constraints required by the baseband signal processing for an SDR System. FPGAs can support high data rates and high bandwidth for current and next generations' wireless communication standards with a reduction in power consumption and reasonable hardware resources utilization. Runtime DPR offers the needed hardware flexibility to reconfigure the SDR system from a wireless standard to another reusing the same FPGA hardware resources.

Several previous works have discussed the advantages of applying DPR on the implementation of a reconfigurable hardware platform for SDR. McDonald [15] presented a full reprogrammed SDR physical layer implementation on Virtex-4 the presented system has a high reconfiguration time overhead and the throughput of the system is low compared to the new wireless standards requirements. Also in [12], a high-speed reconfiguration time dynamic cognitive radios implementation is presented using the Xilinx Zynq hybrid FPGA. The proposed system uses the FPGA Programmable Logic (PL) to implement the baseband and the ARM processor for the MAC layer.

In [13-14] a multi-standard DPR SDR implementation is proposed to emphasize the feasibility of DPR on the implementation of SDR. The proposed system does not consider a fully SDR terminal (transmitter/receiver) and targeting an old Xilinx FPGA series (Virtex-5)

4.2.1. Proposed SDR System Overview

The proposed SDR system is designed on reconfigurable hardware platform (FPGA) using DPR technique. Different communication chains are defined by dynamically reconfiguring a set of reconfigurable region on the FPGA floorplan to change the hardware functionality reusing the same hardware resources as shown in Figure 4.1. Different standards communication blocks (RMs) are stored on an external memory to be loaded on runtime in case of the system is switching between different communication standards.

Also, a processing system unit takes care of:

1. Communication blocks data processing
2. FPGA reconfiguration for switching between different communication standards
3. Memory management and passing input data to the SDR system and capturing output data (testing and evaluation).

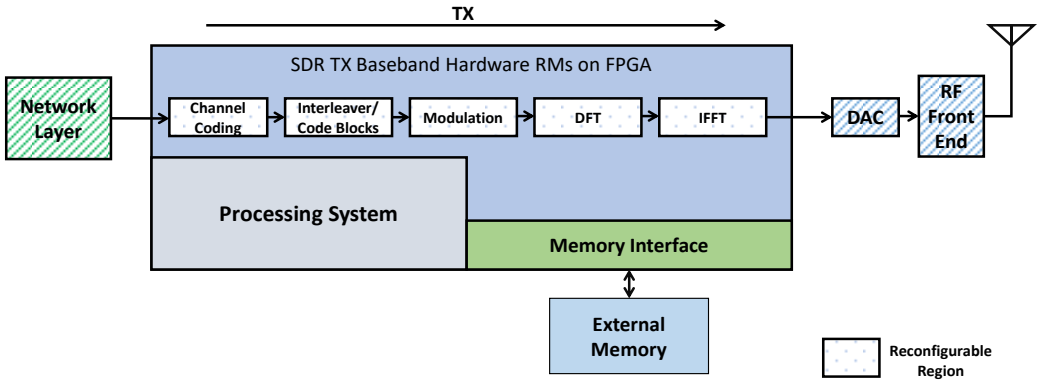


Figure 4.1: Multi-Standard SDR System TX Chain

4.2.2. Supported Communication Standards

Digital Wireless communication standards are almost similar in basic communication blocks. The simple physical communication channel is formed of channel coding blocks that are used to overcome the channel circumstances on the transmitted or received data. The second main blocks are the interleavers and code blocks that are used for error correction and minimize the effect of burst errors introduced in transmission. Modulation blocks are commonly used in communication channels where a carrier signal adapted in a special way to carry the data signal over the medium to reach its destination. Spreading and Scrambling blocks are widely used for purpose of

increasing the bandwidth of the signal and avoid the interference between different channels.

A three communication TX chains (3G, WIFI, and LTE) are supported by the proposed SDR system. Where the communication blocks have different specs according to the mode of operation for each TX chain.

4.2.2.1. 3G Full Transmitter Chain

The 3G is the third generation wireless mobile standard. 3G uses the Wideband Code Division Multiple Access (WCDMA) radio technology to offer greater spectral efficiency and data rate up to 384 kbps in uplink and downlink for simultaneous voice and data. Figure 4.2 shows the physical communication blocks of the 3G TX communication chain supported by the proposed SDR system [49-51].

The 3G TX chain consists of the following blocks:

1. CRC (Cyclic Redundancy Check) Attachment:

CRC process is provided on transport blocks for error detection in which the entire block is used to calculate the CRC parity bits for each transport block. The size of the CRC used in the proposed implementation is 8 bits.

2. Code Block Segmentation:

Segmentation of the bit sequence from transport block after CRC attachment is performed on this block. Code block segmentation fragment a large transport block into smaller code blocks before the channel encoder.

3. Convolutional Encoder:

In the 3G data is encoded using FEC convolutional encoder or turbo encoder. A convolutional encoder is used in the proposed implementation with a constraint length of 9 and coding rates of 1/2.

4. Code block concatenation:

After the channel coding for each code block, the encoded blocks are serially concatenated according to of their output index from the encoder.

5. Interleavers Blocks:

Interleavers blocks contains Radio frame equalizer, first interleaver block, radio frame segmentation and second interleaver. Interleaving is a way to re-arrange data in a non-contiguous way to make it stand burst errors. Interleaving period used in this implementation or the Transmission Time Interval (TTI) is equal 10 ms.

6. Spreading and Scrambling Blocks:

Spreading and Scrambling are applied to the physical channels. They consist of two operations.

- Channelization operation (Spreading): increase the bandwidth of the signal using fully orthogonal codes called channelization codes to not interfere with each other.
- Scrambling operation: Scrambling code is applied to the spread signal and it doesn’t affect the signal bandwidth. It’s used to separate between

different users in uplink and the receiver can retrieve the original data by using the same scrambler sequence.

7. Modulation Mapper:

Different phase and amplitude modulation techniques are used in 3G standard like Binary Phase Shift Key (BPSK), Quadrature Amplitude Modulation (QAM). The proposed implementation uses BPSK modulation scheme for the modulation process.

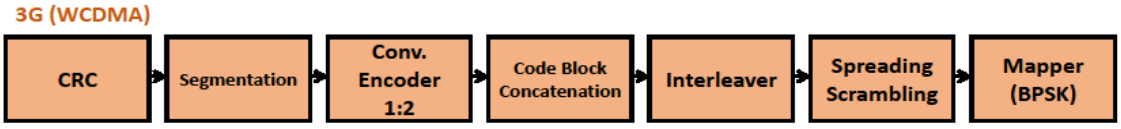


Figure 4.2: 3G TX Chain

4.2.2.2. WIFI Full Transmitter Chain

WIFI is a fixed wireless communication chain based on the IEEE 802.11 a standard. It allows the connection to the internet wirelessly. The physical layer of WIFI uses Orthogonal Frequency Division Multiplexing (OFDM) as a modulation scheme to reduce the interference effect. WIFI technology uses the 5 GHZ frequency spectrum and can process data up to 54 Mbps. Figure 4.3 shows the physical communication blocks of the WIFI TX communication chain supported by the proposed SDR system [48].

The WIFI TX chain consists of the following blocks:

1. Scrambler Block

Scrambler block is used to randomize the data and generate data patterns for purpose of synchronization

2. Convolutional Encoder Block

For channel coding convolutional encoder of rates 1/2, 1/3 or 3/4 corresponding to the desired data rate with a constraint length of 7. The encoder is followed by parallel to serial block to transmit the encoded bits to the puncture.

3. Puncturer Block

Puncturer block is used to generate additional rates from a single convolutional code. The basic idea behind puncturing is increasing the rate of the code by inserting a dummy zero instead of a number of encoder output bits. The proposed SDR system supports the two types of puncturers in wifi standard: (2/3) and (3/4) according to the data rate.

4. Interleaver Block

All encoded data bits shall be interleaved by a block interleaver with a block size corresponding to the number of bits in a single OFDM symbol.

5. Modulation Mapper

In 802.11a (BPSK, QPSK) and (16-QAM, 64-QAM) modulation techniques are used according to the desired data rate. The proposed SDR system supports only the (BPSK and QPSK) modulation scheme.

6. IFFT Modulation

Inverse fast Fourier transform (IFFT) is used for modulation process, as specified by the IEEE 802.11a standard, 64-point IFFT is used with symbol duration of 4 us and 20 MHz operation mode.

7. Preamble

In IEEE 802.11a, The Preamble block is used for synchronization purpose. Two preambles are used for short and long symbols.

Figure 4.3: WIFI TX Chain

4.2.2.3. LTE Full Transmitter Chain

LTE is the fourth generation wireless mobile standards. LTE uplink transmitter uses Single Carrier Frequency Division Multiple Accesses (SC-FDMA) for its lower peak-to-average power ratio (PAPR) results in a power cost reduction in transmitter terminal. LTE supports six different channel bandwidths from 1.4 to 20 MHZ in both frequency and time division duplex (FDD and TDD). The resources allocation in LTE is based on resource block concept. Figure 4.4 shows the physical communication blocks of the 3G TX communication chain supported by the proposed SDR system [51-55].

The LTE TX chain consists of the following blocks:

1. CRC Block

The CRC block in LTE is the same as in 3G. The size of the CRC used in the proposed implementation is 24 bits.

2. Code Segmentation Block

Segmentation of the bit sequence from transport block after CRC attachment is performed on this block. Code block segmentation fragment a large transport block into smaller code blocks before the channel encoder.

3. Turbo Encoder Block

Turbo encoder is a Parallel Concatenated Convolutional Code (PCCC) with two 8-state constituent encoders and one turbo code internal interleaver. The coding rate of turbo encoder is 1/3.

4. Rate Matching and Interleaver Blocks

This block consists of:

- Rate matching block for turbo coded transport channels.
- Three sub-block interleavers for the three output bits of the turbo decoder.
- Bit collection block to collect the output from the three sub-block interleavers.
- Bit selection block is used to remove the dummy bits from the bit collection output.

5. Code Block Concatenation

After the channel coding, the encoded blocks are serially concatenated according of their output index from the encoder.

6. Scrambler Block

Scrambler is used to randomize the bits, prevent long sequences to keep synchronization.

7. Modulation (Mapper)

In LTE (BPSK, QPSK) and (16-QAM, 64-QAM, 256-QAM) modulation techniques are used according to the desired data rate. The proposed SDR system supports only the (QPSK) modulation scheme.

8. SC-FDMA Block

SC-FDMA is like Orthogonal Frequency Division Multiple Access (OFDMA) with Discrete Fourier Transform (DFT) and N-IFFT. M-point DFT and N-point IFFT is selected such that M is less than M. LTE supports different N-point IFFT sizes for different frequencies. The proposed SDR system supports only 64-point DFT and 256-point IFFT

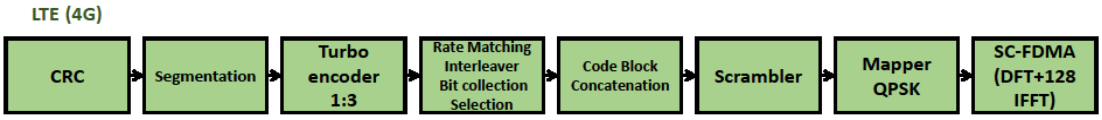


Figure 4.4: LTE TX Chain

4.3. Design and Implementation of SDR TX Chain

The proposed design of the reconfigurable hardware SDR system is switching between three wireless transmitter chains (3G, WIFI, and LTE). Using DPR hardware implementation techniques targeting a 7-series Xilinx Zynq hybrid FPGA. The design consists of a number of reconfigurable modules (RMs) for each transmitter chain. RMs represent the different communication blocks required for each TX chain as mentioned in the previous section described the communication blocks. Each standard has a unique set of RMs that is not common among the three standards. In the proposed reconfigurable hardware SDR implementation RMs are allocated to Reconfigurable Regions (RRs) on the FPGA floorplan. The SDR system uses DPR to switch between the three transmitter chains during runtime. Table 4-1 shows the different RMs for each transmitter chain (the three configuration modes of the reconfigurable SDR system).

The system is implemented using Xilinx Zynq XC7Z020LG484-1 hybrid FPGA on ZC702 evaluation board [43]. The design flow is based on the regular Xilinx DPR design flow.

There are two implemented DPR design approaches. A first trivial implementation is to consider only one large RR that holds the most hardware resources required by the largest configuration. Then, dynamically reconfigure this region with a wireless standard.

This is known as single region partitioning. The second approach is to split this RR into several ones. Therefore, this will reduce the reconfiguration time and power consumption, and in addition, reduce reconfigurable hardware. The two design parameters now are 1) number of RRs and 2) how to split different blocks among these RRs optimally.

Xilinx Vivado 2015.2 synthesis tool is used to synthesize the communication blocks of each standard (TX chain RMs). Each wireless standard has its own RMs with different type and number of resources, as shown in Figure 4.5. The communication blocks are given to the synthesis tool as a set of HDL files to determine the numbers and types of resource requirements on the FPGA. RMs synthesis is a preliminary step before DPR implementation and resources allocations

4.3.1. SDR System Setup

For SDR implementation using DPR techniques a setup environment (static part of the system) is required to:

1) Control and manage the input/output data from and to the SDR system.
2) Control and monitor the FPGA reconfiguration during the DPR process.
3) Memory management for loading partial bitstream from an external memory to the FPGA configuration memory.
4) Evaluate the SDR system performance by measure the total reconfiguration time and the power consumption for each configuration mode.

Xilinx Vivado 2015.2 design suite is used to build the setup environment or the system block design for the SDR system DPR implementation. Figure 4.6 shows the complete system block design diagram as it appears in the tool GUI. The system block design consists of Processing System (PS) and Programmable Logic (PL). The PS unit is shown by the red box in Figure 4.6. PS unit consists of the ARM Cortex A9 processor, I/O peripheral interfaces, DDR Memory controller and clock generator.

The PS unit is generally used for:

- Transfer the Partial bit files of the different RMs stored in the SD-Card to the onboard DDR memory using DMA-PS. Each partial bitstream has a specific length of data bytes and a specific range of addresses in the DDR memory. The partial bitstreams data are stored in the DDR to be loaded faster to the FPGA configuration memory during the reconfiguration process.
- PS unit is used as the clock sources of the system (AXI-BUS, I/O Peripherals), clock frequencies are generated from the ARM PLL or I/O PLL.
- Table 4-2 shows the clock frequencies required for the system and the input/output data rates of the three TX chains.
- PS unit controls the AXI-Bus and responsible of signals handshaking between system design blocks.
- PS unit is used to read input data required for SDR testing from the SD-Card to the DDR memory.

Table 4-1: SDR System Configuration Modes and Reconfigurable Modules

Confi	RR_1	RR_2	RR_3	RR_4	RR_5	RR_6	RR_7	RR_8
3G	CRC-8bit	3G-Segmentation	3G_Conv.Encoder (1:2)	3G_Code Block Concatenation	3G_Inter leaver	3G_Spreading Scrambling	Mapper (BPSK)	-
WIFI	WIFI-Scrambler	WIFI-Conv_Encoder (1:2)	WIFI_Interleaver	Mapper (BPSK)	64-Point IFFT	Preamble	-	-
LTE	CRC-24bit	LTE_ Segmentation	Turbo_Encoder (1:3)	Rate Matching+Inter leaver	LTE-Code Block Concaten ation	LTE_ Scrambler	Mapper (QPSK)	SC_FDMA (64-Point DFT + 256 Point IFFT

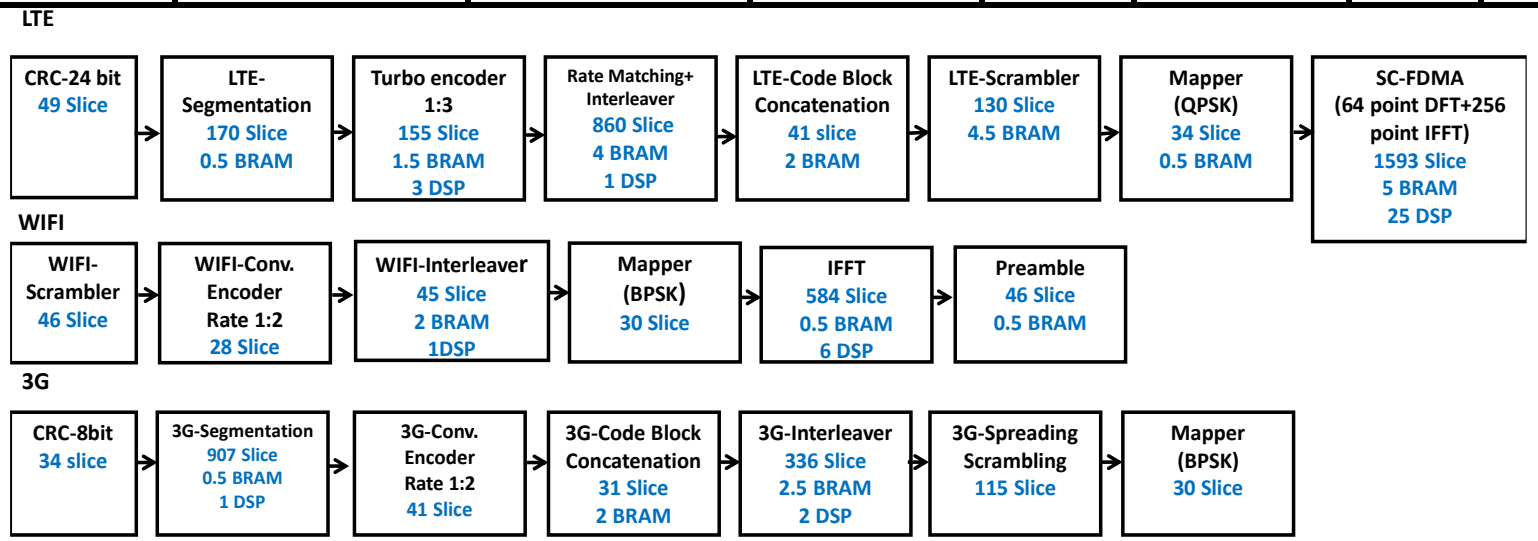


Figure 4.5: SDR RMs Resources Requirements for the Three Transmitter Chains

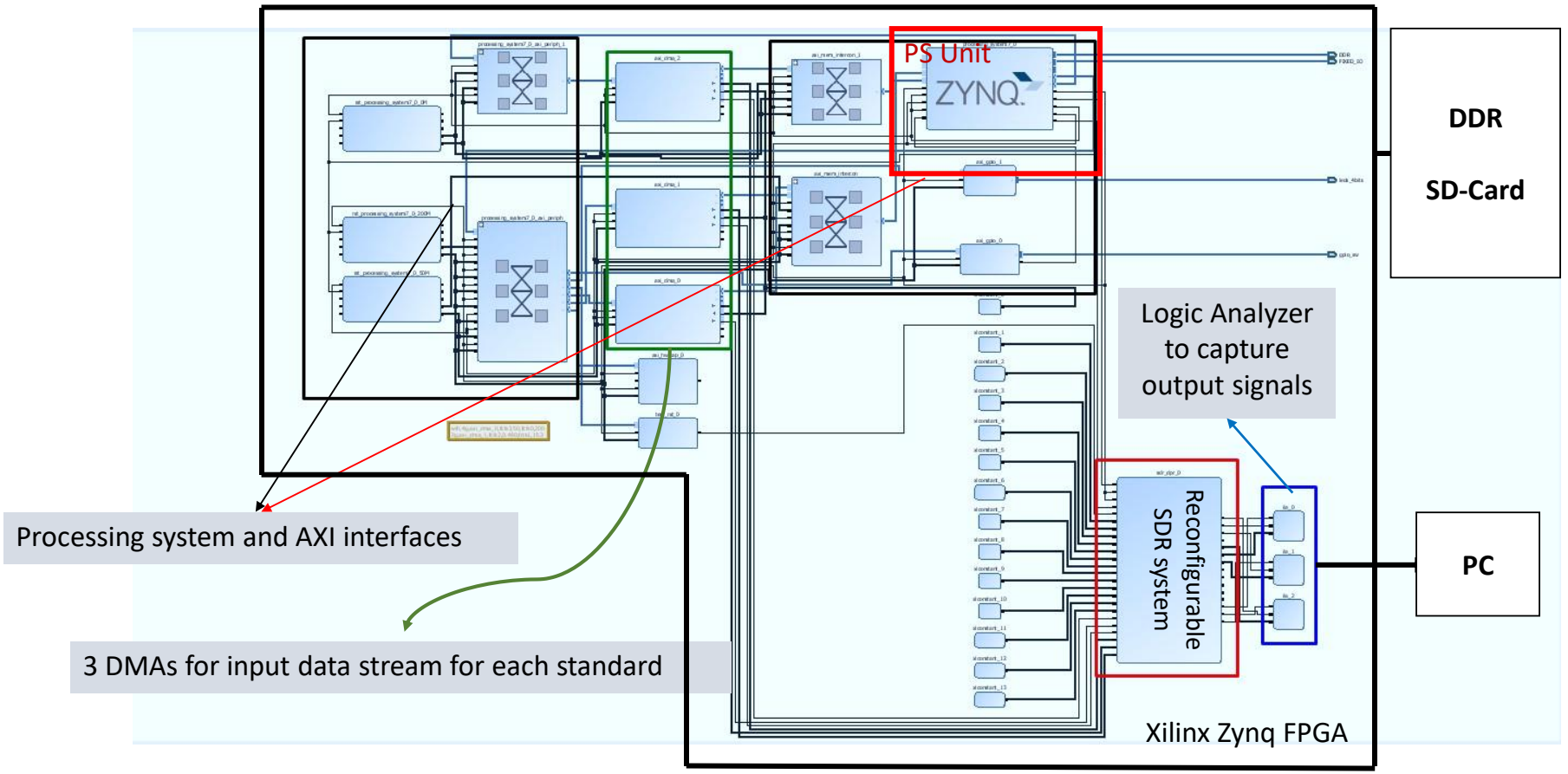


Figure 4.6: SDR System Block Design using Xilinx Vivado Design Suit

Table 4-2: Reconfigurable SDR System Clock Frequencies

	ARM Processor	PRC and ICAP	System AXI-Bus + I/O Peripherals	3G		WIFI		LTE	
Clock Frequency	666.666 MHZ	100 MHZ	200 MHZ	Input CLK	Output CLK	Input CLK	Output CLK	Input CLK	Output CLK
				478 KHZ	3.82 KHZ	50 MHZ	25 MHZ	50 MHZ	50 MHZ

The PL part contains:

- Reconfigurable SDR hardware block contains the RMs of the TX chains. The RMs are a set of black boxes in the system block design before the process of reconfiguration. During reconfiguration, a number of RMs black boxes are reconfigured and filled with the required RMs logic design according to the selected configuration (active standards).
- Partial reconfiguration controller takes place in the system block design as an IP core with two types of interfaces:
  i. AXI slave interface to communicate with the ARM processor for controlling and monitoring of the reconfiguration process.
  ii. AXI memory streaming interface to transfer the partial bitstreams data from the DDR to the ICAP and then to the FPGA configuration memory.

  PR controller and ICAP are running with a clock frequency of 100 MHZ.
- Three input streaming DMAs are used to transfer the input data from the DDR memory to the SDR TX chain RMs. Each configuration mode (active standard) has its DMA with a transfer rate equal to the input clock of the active standard. For example, if the 3G is the active standard the DMA transfer rate is equal 478 KHZ as in Table 4-2.
- Three debug hub cores for on-chip integrated logic analyzer (ILA) to capture the output signals of the SDR system. Each communication standard supported by the SDR system has its own debug core that is running with a clock frequency equal to the output clock of the specific standard. The captured output signals are sent to the PC via UART to be analyzed.

4.3.2. Single Partition Region DPR Implementation for SDR System

First DPR implementation for the reconfigurable hardware SDR system is the single partition DPR implementation denoted as (***First DPR Implementation***). Single partition DPR design implementation is based on the trivial Xilinx DPR design flow [] discussed in chapter 3. This section presents the DPR implementation design of the SDR system using a single partition region for SDR RMs allocation. Also, the results of the floorplanning area required for the dynamic parts and the static part of the system, the results of the total reconfiguration time required to switch between the three different communication standards of the SDR system, the memory size needed to store the RMs partial bitstreams and finally the power consumption values of the system at every design configuration. The static part of the single partition region SDR system is shown in Figure 4.7. The static part of the system consists of the hardware blocks required for controlling and monitoring the reconfigurable partition region of the system.

4.3.2.1. Total Partitions Area on The FPGA

Table 4-3 summarizes the FPGA required resources for each configuration mode in the SDR system shown in Figure 4.5 and Table 4-1. It shows the number of Slices, BRAM and DSP used by the three configuration modes of the SDR system, the floorplanned area of the single partition region, the size of partial bitstream file for each configuration and the utilization of the three different configurations in the floorplanned partition region. The single partition region area should be large enough to allocate the largest configuration are in the SDR system. Therefore the single partition region is allocated on the FPGA floorplan with a suitable dimension to hold the LTE configuration mode which is the largest configuration mode area in the SDR system. Figure 4.8 shows the three FPGA floorplanned designs of the three different configuration modes, the location and size of the single partition region are fixed during reconfiguration and equal to 3300 Slice, 30 BRAM, and 120 DSP. The single partition region spanned over three FPGA clock regions with a length of three reconfigurable tiles of 150 CLB and a width of 20 blocks. From Table 4-3 and Figure 4.8, the three configuration modes of the system share the same single partition region resources and all the configuration modes of the system have the same partial bitstream size of 723 KB which is proportional to the single partition region size on the floorplan. For the utilization of single partition region, it is shown that the WiFi configuration mode has the worst utilization percentage of 23.6 %. Therefore, single partition region DPR implementation is not a cost effective DPR implementation for SDR system. Table 4-4 shows the total resources cost of the SDR system, the static part is the resources required by the DMAs, AXI-bus, partial reconfiguration controllers and the interconnections between the system blocks. Dynamic part is the single partition region that holds the three configuration modes of the system.

Table 4-3: Single Partition Region Resources Cost

Configuration	**Resources Required**			**Single Partition Area**			**Utilization (%)**	**Partial BitSize (KB)**
	Slice	BRAM	DSP	Slice	BRAM	DSP		
3G	1494	5	3	3300	30	120	41.5	723
WIFI	779	3	7				26.5	723
LTE	3032	18	29				95.8	723

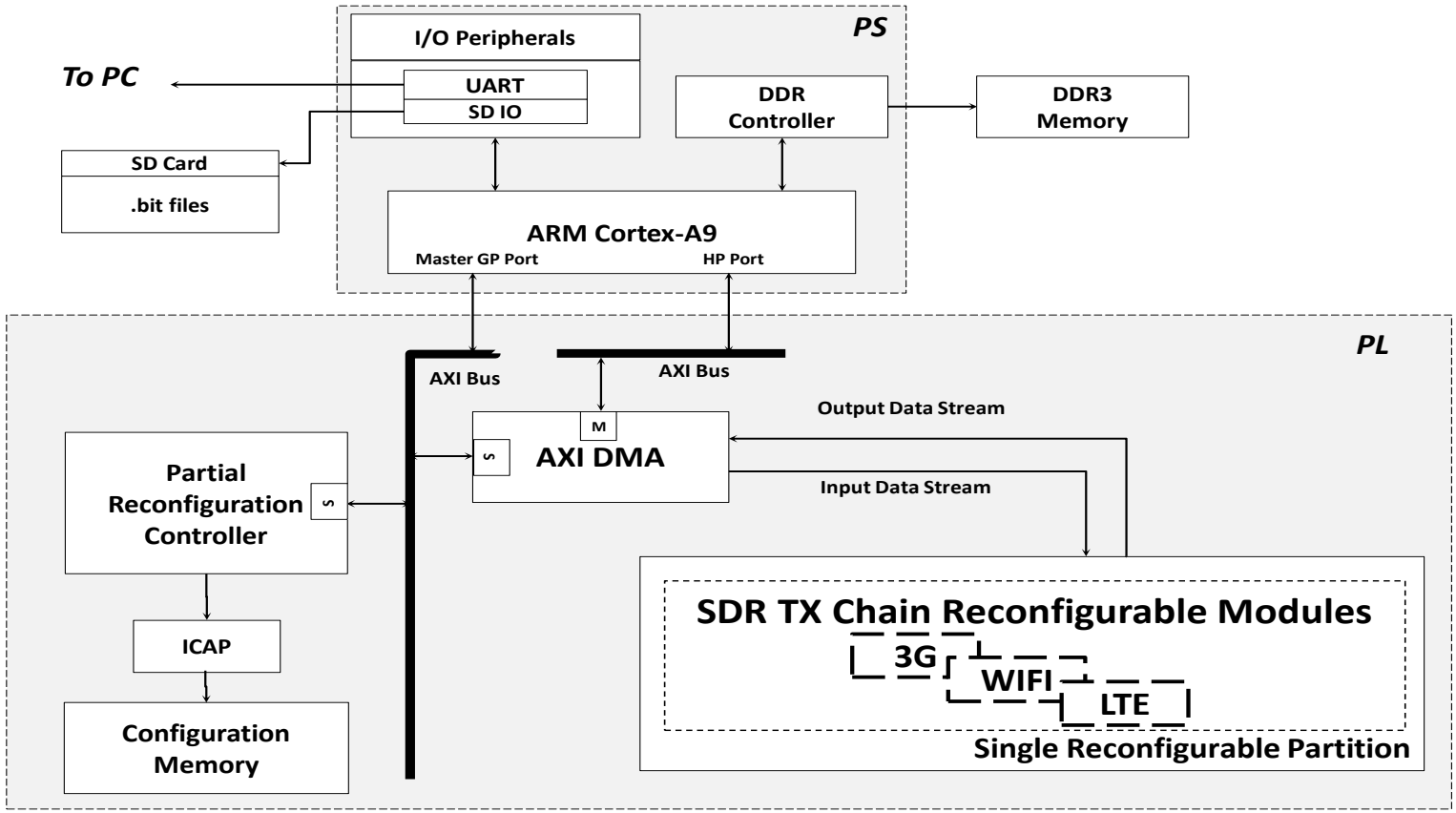
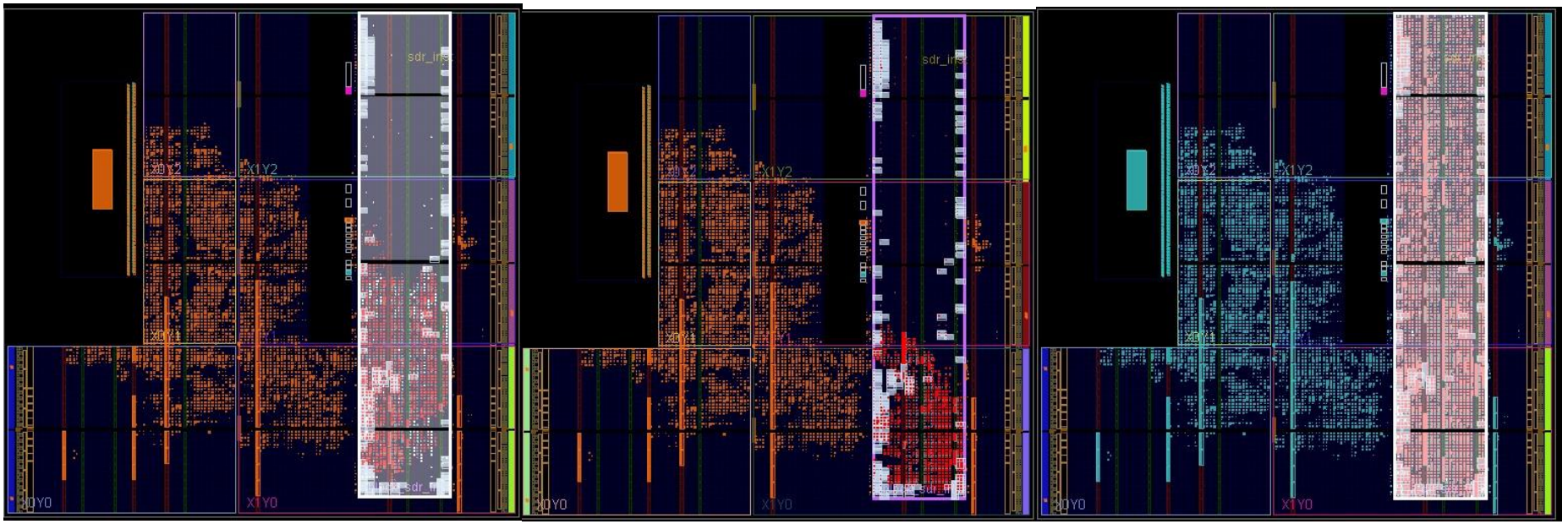


Figure 4.7: SDR System overview of single partition Region DPR Implementation



(a) 3G (b) WIFI (c) LTE

Figure 4.8: (a) 3G, (b) WIFI, and (c) LTE Configuration Floorplam Design of DPR Implementation Using Single Partition Region

Table 4-4: Single Partition Region SDR Implementation Total Resources Cost

SDR Design Parts	Resources		
	Slice	BRAM	DSP
Static Part	3485	41	0
Dynamic Part	3300	30	120
Total Area	6785	71	120

4.3.2.2. Total Reconfiguration Time Measurements

Reconfiguration time is the time needed for reconfiguring a partition region in a DPR design. In single partition region DPR implementation only one partition region needed to be reconfigured in runtime to switch from a configuration to another in the reconfigurable SDR system. Table 4-5 shows the reconfiguration time of the single partition region using three types of PR controllers running at a clock of 100 MHZ. It is shown that using Xil-PRC on the DPR process gives a high-speed dynamic reconfiguration compared to the AXI-HWICAP and PCAP. For the rest of results calculation and performance evaluation, the Xil-PRC is used as the PR controller for the SDR system. Reconfiguration times are measured on board using the ARM processor timer at runtime.

Total reconfiguration time of the SDR system is the summation of the reconfiguration time of every configuration mode in the system.

The total reconfiguration time using (Xil-PRC) = 3G reconfiguration time + WIFI reconfiguration time + LTE reconfiguration time = 3*1.86ms =5.58ms

Table 4-5: Single Partition Region Time of Reconfiguration

PR Controller at 100MHZ	AXI-HWICAP	PCAP	Xil-PRC
Time of Reconfiguration of Single Partition (ms)	**51.7**	**5.75**	**1.86**

4.3.2.3. Power Consumption Measurements

Table 4-6 shows the PL total power consumption of the three configuration modes of the SDR system. The power consumption is measured using Xilinx vivado power estimator tool on the programmable logic of the FPGA, the power consumed by the PS (ARM processor) is excluded from the power analysis. The PL total power consumption for every configuration include the power of the static part of the system plus the dynamic part. The Power consumption of the static part has a fixed value with every configuration modes of the system and equal to 13 mw. LTE configuration mode consumes the maximum power of the system as shown in Table 4-6 since the LTE configuration mode is the largest configuration area and in the same time the highest data rate of the SDR system with a maximum clock frequency of 50 MHZ. The 3G configuration mode consume the minimum power of the system since the 3G configuration mode has an

output data rate of 3.82 MHZ and input data rate of 0.48 MHZ. On the Zynq FPGA platform, the PS unit (ARM + I/O peripherals interfaces)consumes a power of 1.533 w which is a fixed power consumption value over the three configuration modes of the SDR system and not related to the power consumption on the PL side of the Zynq FPGA

Table 4-6: Single Partition Region DPR Implementation PL Total Power Consumption

Configuration	PL Total Power Consumption
3G	20 mw
WIFI	26 mw
LTE	81 mw

4.3.3. Multi-Partitions regions DPR Implementation for SDR System

Second DPR implementation for the reconfigurable hardware SDR system is the multi partitions regions DPR implementation denoted as (***Second DPR Implementation***). Multi partitions regions DPR design implementation is based on the proposed Xilinx DPR design flow discussed in Chapter 3 by applying clustering partitioning algorithm to find the optimum partitions number and sizes for RMs allocation in order to reduce the total reconfiguration time of the system, minimize the total partitions area overhead and increase the FPGA resources utilization for all system configuration modes. This section presents the DPR implementation design of the SDR system using multi partitions regions for TX chains RMs allocation. Also, the results of the floorplanning area required for the dynamic parts and the static part of the system, the results of total reconfiguration time required to switch between the three different communication standards of the SDR system, the memory size needed to store the RMs partial bitstreams and finally the power consumption values of the system at every design configuration.

Figure 4.9 shows the partitioning scheme determined by the modified clustering partitioning algorithm. The partitioning optimum solution is based on reducing the total time of reconfiguration and reduce the total partitions area of the given three configuration modes of the SDR system (3G, WIFI, and LTE). The partitioning scheme solution consists of 4 RRs with sizes equal to the sizes of the largest merged RMs allocated to them. Therefore, RR_1 size >= (LTE {Code Concatenation block}) RM size, RR_2 size >= (LTE {Scrambler + Mapper} RMs size), RR_3size >= (3G {Segmentation + Mapper} RMs size) and R_4 size >= (LTE {IFFT+DFT} RMs size). From Figure 4.9 it is shown that 3G and WIFI configuration modes use only three RRs (RR_1, RR_2, and RR_3) to reduce the partition area overhead results from the using of a single partition region in first DPR implementation.

The static part of the multi partitions regions SDR system is shown in Figure 4.10. The static part of the system consists of the hardware blocks required for controlling and monitoring the reconfigurable partitions regions of the system and the routing switch used to reroute the interconnection between the RRs and reroute the interconnections between the static part and dynamic part during runtime.

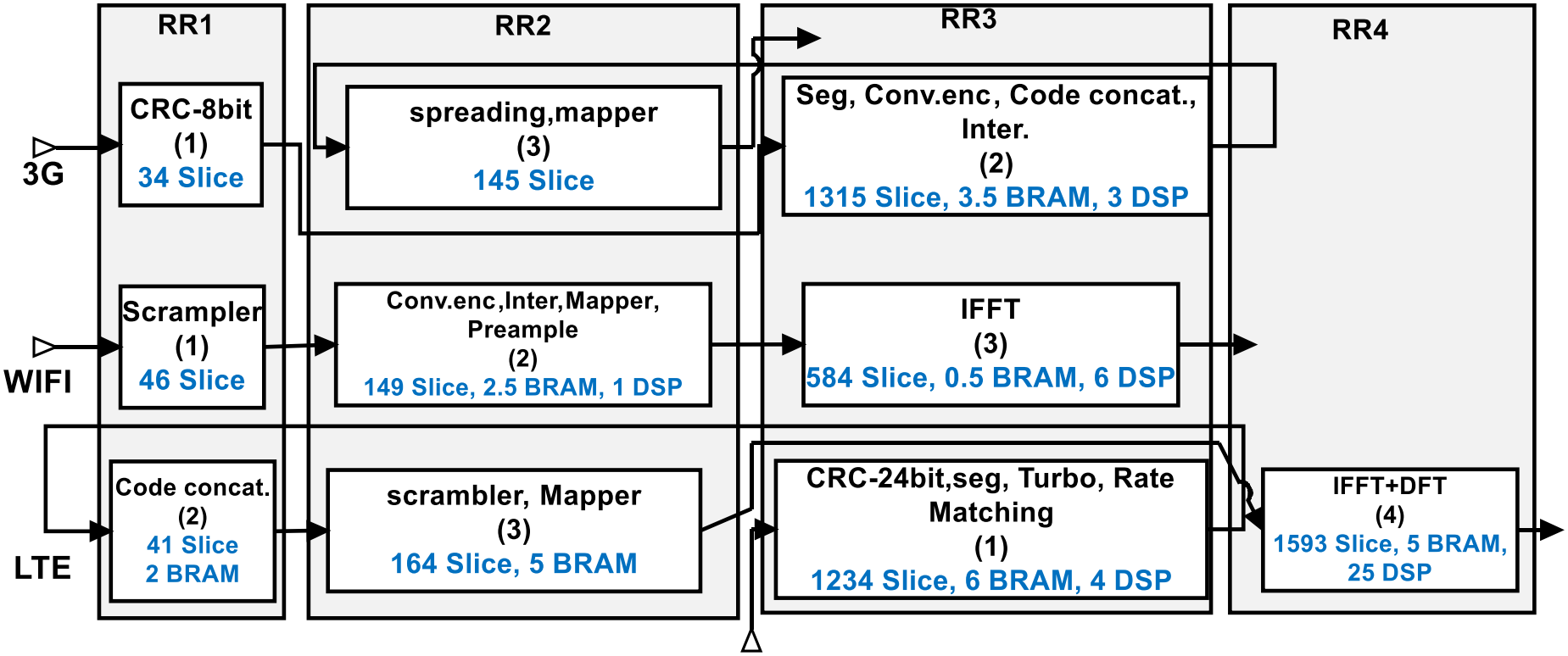


Figure 4.9: SDR RMs Partitions Using Modified Clustering Partitioning algorithm

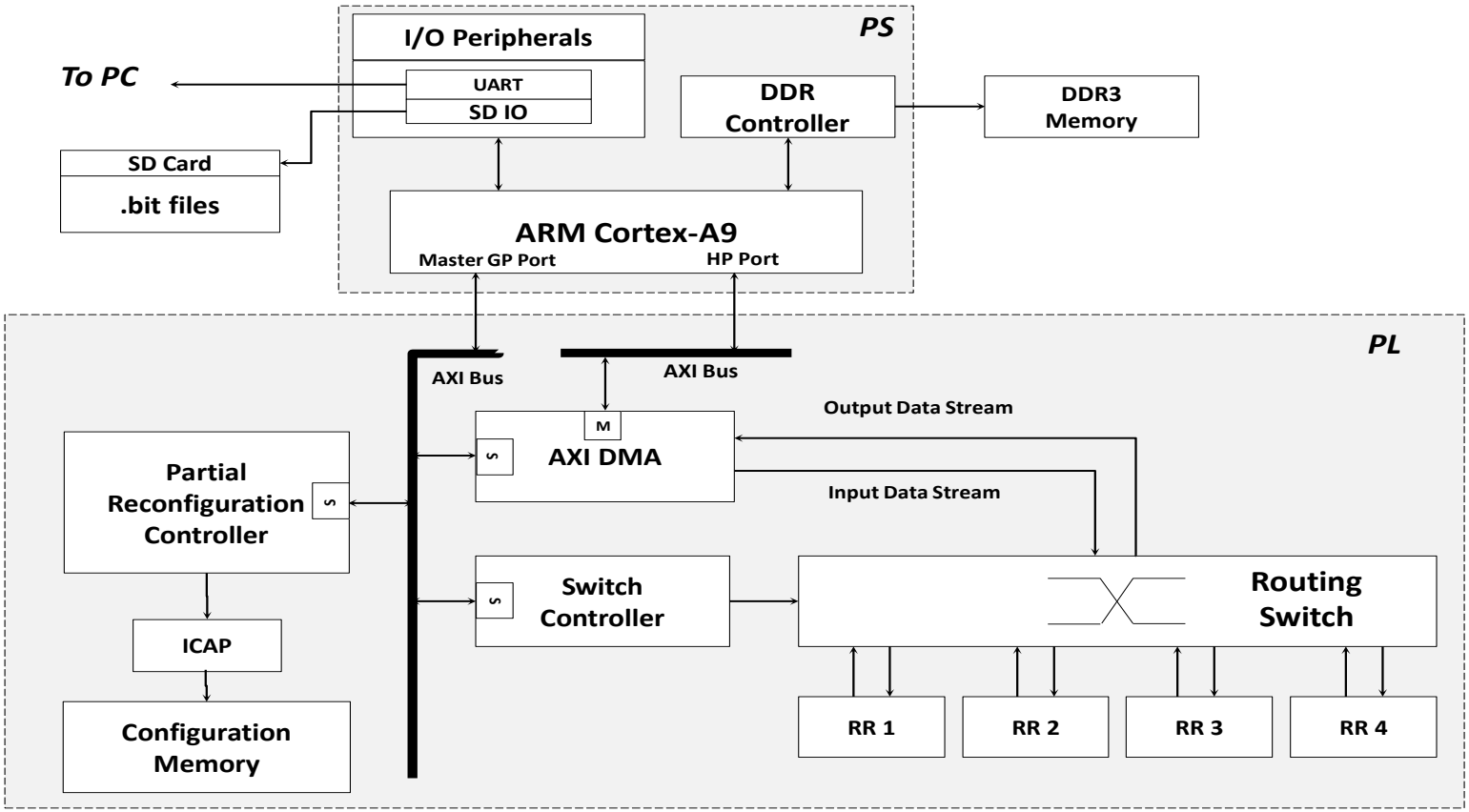


Figure 4.10: SDR System Overview of Multi-Partitions Regions DPR Implementation

4.3.3.1. Total Partitions Area on The FPGA

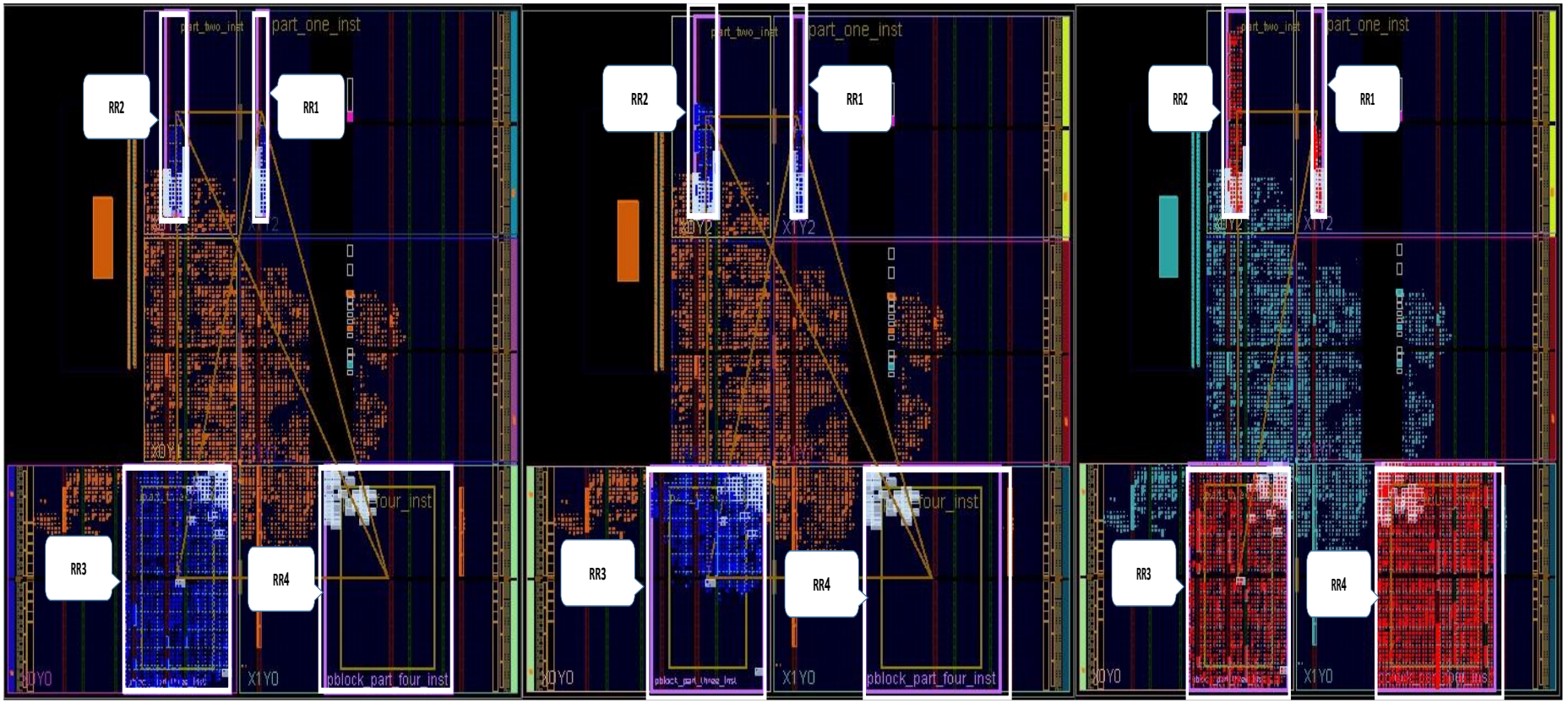
Table 4-7 summarizes the FPGA required resources for each configuration mode in the SDR system shown in Figure 4.9 . It shows the number of Slices, BRAM and DSP used by the three configuration modes of the SDR system, the floorplanned area required for of the four partitions regions, the size of partial bitstream file for each RR and the utilization of the three different configurations in the four RRs. Table 4-8 shows the total resources cost of the SDR system, the static part, and the dynamic parts. Figure 4.10 shows the three FPGA floorplanned designs of the three different configuration modes, the location, and size of the four partitions regions are fixed during the reconfiguration.

Table 4-7: Multi-Partitions Regions Resources Cost

RR (Partial Bitstream Size KB)		3G			WIFI			LTE		
		Required Resources	Floorplanned	Util (%)	Required Resources	Floorplanned	Util (%)	Required Resources	Floorplanned	Util (%)
RR_1	Slice	34	100	45	46	100	52	41	100	42
77	BRAM	0	10		0	10		2	10	
KB	DSP	0	0		0	0		0	0	
RR_2	Slice	145	200	46	149	200	49	164	200	89
103	BRAM	0	10		2.5	10		5	10	
KB	DSP	0	20		1	20		0	20	
RR_3	Slice	1315	1300	93	584	1300	45	1234	1300	95.5
320	BRAM	3.5	20		0.5	20		6	20	
KB	DSP	3	20		6	20		4	20	
RR_4	Slice	0	1600	0 Not Used	0	1600	0 Not Used	1593	1600	99
324	BRAM	0	10		0	10		5	10	
KB	DSP	0	40		0	40		25	40	
Total RRs	Slice	1494	1600	93	779	1600	49	3032	3200	95
	BRAM	3.5	35		3	35		18	45	
	DSP	3	50		7	50		29	90	

Table 4-8: Multi-Partitions Regions SDR Implementation Total Resources Cost

SDR Design Parts	Resources		
	Slice	BRAM	DSP
Static Part	3501	41	0
Dynamic Parts	3200	45	90
Total Area	6701	86	90



(a) 3G (b) WIFI (c) LTE

Figure 4.11: (a) 3G, (b) WIFI, and (c) LTE Configuration Floorplan Design of DPR Implementation Using Partitioning Algorithm

4.3.3.2. Total Reconfiguration Time Measurements

Reconfiguration time is the time needed for reconfiguring a partition region in a DPR design. In multi partitions regions DPR implementation in contrast with single partition DPR implementation, a number of partition regions should be reconfigured in runtime to switch from a configuration to another in the reconfigurable SDR system. Table 4-9 shows the reconfiguration time of the four partition regions required for implementation using three types of PR controllers running at a clock of 100 MHZ. It is shown that using Xil-PRC on the DPR process gives a high-speed dynamic reconfiguration compared to the AXI-HWICAP and PCAP. For the rest of results calculation and performance evaluation, the Xil-PRC is used as the PR controller for the SDR system. Total reconfiguration time of the SDR system is the summation of the reconfiguration time of every configuration mode in the system. Accurate Reconfiguration times are measured on board using the ARM processor timer at runtime.

Table 4-10 shows the reconfiguration time of three SDR TX chains (3G, WIFI, and LTE), the reconfiguration time or the switching time of a TX chain is the summation of all the partition region reconfiguration time needed to implement the full configuration.
The switching time of the 3G configuration mode = Reconfiguration Time of (RR_1+RR_2+RR_3) = 0.2ms+0.26ms+0.8ms = 1.26ms.
The switching time of the WIFI configuration mode = Reconfiguration Time of (RR_1+RR_2+RR_3) = 0.2ms+0.26ms+0.8ms = 1.26ms.
The switching time of the LTE configuration mode = Reconfiguration Time of (RR_1+RR_2+RR_3+RR_4) = 0.2ms+0.26ms+0.8ms+0.82 = 2.08ms.

Total reconfiguration time of the SDR system is the summation of the reconfiguration time of every configuration mode in the system.

The total reconfiguration time using (Xil-PRC) = 3G reconfiguration time + WIFI reconfiguration time + LTE reconfiguration time = 1.26ms+1.26ms+2.08ms = 4.6ms

Table 4-9: Multi-Partitions Regions Time of Reconfiguration

PR Controller at 100MHZ	AXI-HWICAP	PCAP	Xil-PRC
Time of Reconfiguration of RR_1 (ms)	5.38	0.6	0.2
Time of Reconfiguration of RR_2 (ms)	7.2	0.8	0.26
Time of Reconfiguration of RR_3 (ms)	22.37	2.5	0.8
Time of Reconfiguration of RR_4 (ms)	22.65	2.52	0.82

Table 4-10: SDR TX Chains Switching Time in Multi-Partitions Regions DPR Implementation

		PR Controller at 100MHZ		
		AXI-HWICAP	PCAP	Xil-PRC
Time of Reconfiguration of a full TX Chain (ms)	3G	35 ms	3.9 ms	1.26 ms
	WIF	35 ms	3.9 ms	1.26 ms
	LTE	57.6 ms	6.4 ms	2.08 ms

4.3.3.3. Power Consumption Measurements

Table 4-11 shows the PL total power consumption of the three configuration modes of the SDR system in multi-partitions regions DPR implementation. The power consumption is measured using Xilinx vivado power estimator tool on the programmable logic of the FPGA, the power consumed by the PS (ARM processor) is excluded from the power analysis. The PL total power consumption for every configuration include the power of the static part of the system plus the dynamic part. The Power consumption of the static part has a fixed value with every configuration modes of the system and equal to 9 mw. LTE configuration mode consumes the maximum power of the system as shown in Table 4-11 since the LTE configuration mode is the largest configuration area and in the same time the highest data rate of the SDR system with a maximum clock frequency of 50 MHZ. The 3G configuration mode consume the minimum power of the system since the 3G configuration mode has an output data rate of 3.82 MHZ and input data rate of 0.48 MHZ. On the Zynq FPGA platform, the PS unit (ARM + I/O peripherals interfaces)consumes a power of 1.533 w which is a fixed power consumption value over the three configuration modes of the SDR system and not related to the power consumption on the PL side of the Zynq FPGA

Table 4-11: Multi-Partitions Regions DPR Implementation PL Total Power Consumption

Configuration	PL Total Power Consumption
3G	18 mw
WIFI	22 mw
LTE	74 mw

4.4. Performance Evaluation

The performance evaluation comparison between the two proposed DPR implementations for the reconfigurable SDR system is done based on the total partitions area, the total time of reconfiguration and the total estimated power consumption of the design as shown in Figure 4.12 . The total partitions area is calculated by the number of reconfigurable frames. Reconfigurable frames are related to the FPGA tile CLB tile equal 36 reconfigurable frames, BRAM tile equal 28 reconfigurable frames and DSP tile equal 28 reconfigurable frames []. Reconfigurable frames is used as normalized unit for the three types of resources on the FPGA to calculate the partitions area Figure 4.12 (a) shows a comparison between the total number of reconfigurable frames of the single partition and multi-partition DPR implementations, the total area decreases by ~4.5% in case of multi-partition implementation, although the two implementations have the same required hardware resources as shown in Figure 4.5 and Figure 4.9. In case of single

partition DPR implementation the size of the partition region should be at least equal to the largest configuration size and therefore the single partition region is theoretically the minimum partitioning scheme area over any other partitioning schemes that could be used for a DPR design. But Floorplanning and the FPGA architecture added some physical constraints on the location and dimension of the partition region. This is due to the fact that during floorplanning, using a part of an FPGA column will make the whole column reserved even if it is not totally used see Figure 4.8. To overcome this problem in the multi-partitions implementation the partition region selection on the floorplan is done vertically or column wise inside the same FPGA clock region to eliminate the number of wasted columns as shown in Figure 4.11.

An 8.6% reduction in the total partitions area of multi-partition regions DPR implementation results in a reduction in maximum power consumption compared to the single partition region DPR implementation as shown in Figure 4.12 (c). It is noticed that the total reconfiguration time decreases by ~16% in case of using the partitioning algorithm in multi-partition DPR implementation. Since the switching between some designs configurations does not require the reconfiguration of all RRs. It is noticed that the fourth RR is only used by LTE, leading to a reduction in the total reconfiguration time. Also, choosing locations of RRs in floorplanning plays an effective role in reducing the time of reconfiguration. So, the designer should be aware of the FPGA floorplan

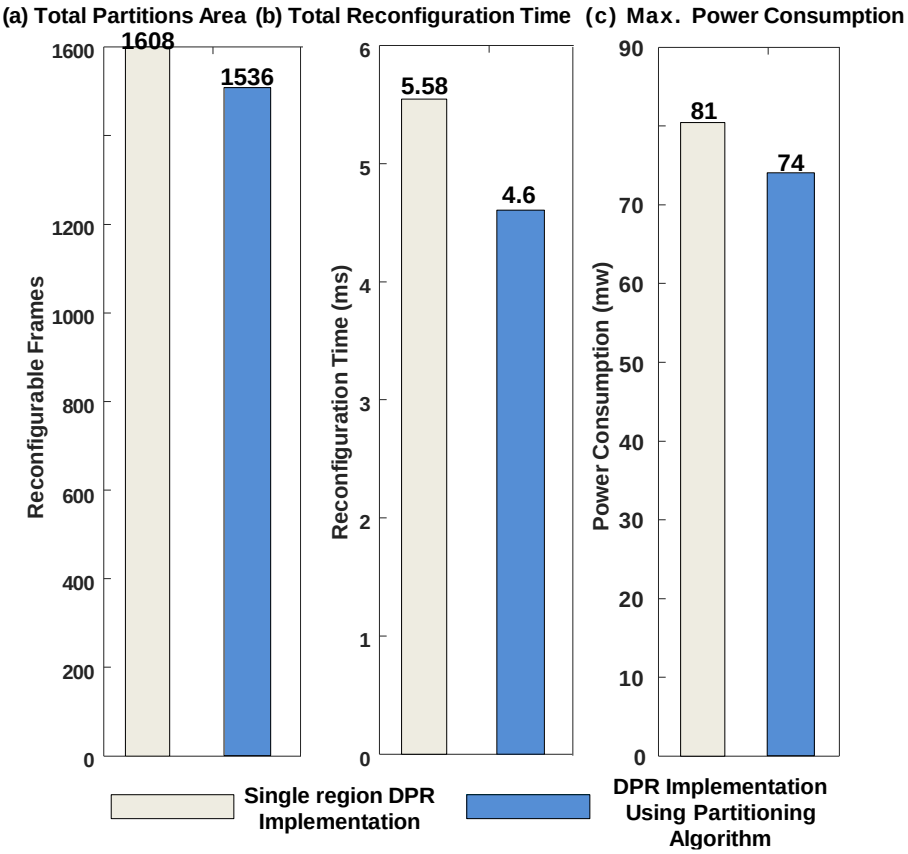


Figure 4.12: Performance Evaluation of the Two Proposed SDR Implementations

4.5. Summary

Hardware reusability is a novel approach in implementing the SDR systems. The DPR on an FPGA platform is an up-and-coming technique to implement reconfigurable SDR chains. In this section, a proposed reconfigurable hardware platform for SDR system is implemented, the proposed SDR system supports three wireless communication transmitter chains for 3G, WIFI, and LTE. Two DPR implementation approaches are considered using single partition region and multi- partitions regions on the FPGA floorplan. It is marked that multi-partitions regions approach has better performance in terms of reconfiguration time, reconfigurable area and power consumption.

Chapter 5 : Conclusion and Suggestions for Future Work

The objective of this thesis is the design and implementation of a DPR hardware platform for SDR systems targeting Xilinx Zynq hybrid FPGAs. This objective is met by:

1. Investigating the dynamic partial reconfiguration technique (*see* Chapter 2) by define the concept of dynamic partial reconfiguration and discuss the advantages and disadvantages of DPR implementation. Also, four partial reconfiguration controllers are presented and used to implement a high-speed reconfigurable SDR system targeting a Xilinx Zynq FPGA. A reconfigurable multi-standards convolutional encoder is implemented as a benchmark to evaluate the performance of the four partial reconfiguration controllers used in the implementation of high-speed DPR implementation. The multi-standards convolutional encoder supports (2G, 3G, WIFI, and LTE) wireless standards. It is shown in Chapter 2 that Xil-PRC gives a high reconfiguration throughput of 396.5 MB/S which is nearly the theoretical value of the ICAP data rate of 400MB/s. Also, Chapter 2 provides design guidelines for the DPR-SDR designers to choose the suitable PR controller based on their system requirements.

2. The techniques of partitioning and the allocation of RMs to RRs have a direct impact on the performance of DPR implementation (*see* Chapter 3). A performance evaluation of four partitioning techniques is presented in this thesis. Reconfiguration time and total reconfigurable area are the main parameters to evaluate a partitioning scheme. It is demonstrated that manual techniques of partitioning are not suitable for time critical DPR systems like SDR. A modified DPR tool flow is proposed in this chapter to improve the time of reconfiguration by 40% compared to the traditional flow and at the same time reducing the design area by virtually changing the RRs size during reconfiguration to be fit with the size of active RMs. Open source partitioning algorithm is used to find the optimum set of partitions. Two kinds of solutions are generated by the algorithm depend on the type of merging between RMs in the same configuration. Also, a routing switch is required to reconnect the RRs each time a new set of configurations are loaded.

3. Hardware reusability is a novel approach in implementing the SDR systems. The DPR on an FPGA platform is an up-and-coming technique to implement reconfigurable SDR chains. Chapter 4 proposed a reconfigurable hardware platform for SDR system based on DPR implementation techniques, the proposed SDR system supports three wireless communication transmitter chains for 3G, WIFI, and LTE. Two DPR implementation approaches are considered using single partition region and multi- partitions regions on the FPGA floorplan. It is marked that multi-partitions regions approach has better performance in terms of reconfiguration time, reconfigurable area and total power consumption.

5.1. Suggestions for Future Work

1. Find the possible similarities between the communication chains blocks to reduce the number of RMs of the system and therefore the number of RRs that should be reconfigured will be reduced during handover between different communication standards.

2. Implementation of a full reconfigurable hardware SDR system that supports the receiver chain (RX) with the transmitter chain.

3. Augment the communication standards supported by the reconfigurable SDR system. For example support the Bluetooth and ZigBee wireless standards.

4. Complete the automation of all steps of DPR design flow. In this thesis a proposed automatic partitioning scheme is developed to find a set of optimum partitions solutions, next design steps are the generation of partitions top wrappers and partitions floorplanning which is done manually by the designer. Automation of partitions wrapper generation and floorplan will reduce the DPR implementation design time and increase the efficiency of the reconfigurable system.

5. DPR hardware verification is more challenging compared to traditional FPGA designs verification due to the change of the hardware function for each new active configuration. Developing a run time DPR hardware verification methodology increases the controllability and observability of the DPR process.

6. High-Level Synthesis (HLS) tool can be used to generate hardware modules (RTL) from a high-level language model e.g.C, C++. Integration of DPR design flow with the HLS tools will accelerate the DPR design time. Xilinx introduces a new software tool for Hardware/ Software co-design implementation (SDSOC) that uses HLS technique and DPR design flow for the purpose of developing a reliable and efficient reconfigurable system.

References

1. J. Mitola and G. Q. Maguire, "Cognitive radio: making software radios more personal," in IEEE Personal Communications, vol. 6, no. 4, pp. 13-18, Aug 1999.
2. J. Mitola, “Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio,” 2000.
3. G. Sklivanitis, A. Gannon, S. N. Batalama and D. A. Pados, "Addressing next-generation wireless challenges with commercial software-defined radio platforms," in IEEE Communications Magazine, vol. 54, no. 1, pp. 59-67, January 2016
4. J. Delahaye, G. Gogniat, C. Roland and P. Bomel, "Software radio and dynamic reconfiguration on a DSP/FPGA platform", Frequenz, vol. 58, no. 5-6, pp. 152-159, 2003.
5. R. Tessier, K. Pocek and A. DeHon, "Reconfigurable Computing Architectures," in Proceedings of the IEEE, vol. 103, no. 3, pp. 332-354, March 2015.
6. S. M. Trimberger, "Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology," in Proceedings of the IEEE, vol. 103, no. 3, pp. 318-331, March 2015.
7. D. Koch, Partial Reconfiguration on FPGAs: Architectures, Tools and Applications, vol. 153. Springer Science & Business Media, 2012 .
8. K. Arun Kumar, “A low power implementation of psk modems in fpga with reconfigurable filter and digital nco using pr for sdr and cr applications,” in International Conference on Green Technologies (ICGT), pp. 192–197,IEEE, 2012.
9. K. Arun Kumar, “Fpga implementation of psk modems using partial reconfiguration for sdr and cr applications,” in India Conference (INDICON),2012 Annual IEEE, pp. 205–209, IEEE, 2012.
10. K. Arun Kumar, “Fpga implementation of qam modems using pr for reconfigurable wireless radios,” in Emerging Research Areas and 2013 International Conference on Microelectronics, Communications and Renewable Energy (AICERA/ICMiCR), 2013 Annual International Conference on, pp. 1–6, IEEE, 2013.
11. M. Hentati, A. Nafkha, X. Zhang, P. Leray, J. F. Nezan, and M. Abid, “The study of the impact of architecture design on cognitive radio,” in Proceedings of the 8th IEEE International Multi-Conference on Systems, Signals & Devices (SSD), p. CD, 2011.
12. S. Shreejith, B. Banarjee, K. Vipin and S. Fahmy, "Dynamic cognitive radios on the Xilinx Zynq hybrid FPGA", in International Conference on Cognitive Radio Oriented Wireless Networks, 2015, pp. 427-437.
13. Sadek, A., Mostafa, H., Nassar, A., Ismail, Y."Towards The Implementation Of Multi-Band Multi-Standard Software-Defined Radio Using Dynamic Partial Reconfiguration". International Journal of Communication Systems,15 June 2017.
14. A. Sadek, H. Mostafa and A. Nassar, "Dynamic channel coding reconfiguration in Software Defined Radio," 2015 27th International Conference on Microelectronics (ICM), Casablanca, 2015, pp. 13-16.

15. E. McDonald, "Runtime FPGA partial reconfiguration", in 2008 IEEE Aerospace Conference, 2008, pp. 1-7.

16. Leray, Pierre, Xiguang Wu, and Jacques Palicot. "Cognitive Radio Management Benefiting From Flexible Reconfiguration". Proceedings of the Fourth International Conference on Telecommunications and Remote Sensing, 15 June 2015.

17. J. C. Rodríguez and K. F. Ackermann, "Leveraging partial dynamic reconfiguration on Zynq SoC FPGAs," 2014 9th International Symposium on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), Montpellier,2014, pp. 1-6.

18. A. Hassan, R. Ahmed, H. Mostafa, H. A. H. Fahmy and A. Hussien, "Performance evaluation of dynamic partial reconfiguration techniques for software defined radio implementation on FPGA," 2015 IEEE International Conference on Electronics, Circuits, and Systems (ICECS), Cairo, 2015, pp. 183-186.

19. K. Papadimitriou, A. Anyfantis and A. Dollas, "An Effective Framework to Evaluate Dynamic Partial Reconfiguration in FPGA Systems," in IEEE Transactions on Instrumentation and Measurement, vol. 59, no. 6, pp. 1642-1651, June 2010.

20. S. Liu, R. Pittman, A. Forin and J. Gaudiot, "Minimizing the runtime partial reconfiguration overheads in reconfigurable systems", The Journal of Supercomputing, vol. 61, no. 3, pp. 894-911, 2012.

21. S. Bhandari et al., "High Speed Dynamic Partial Reconfiguration for Real Time Multimedia Signal Processing," 2012 15th Euromicro Conference on Digital System Design, Izmir, 2012, pp. 319-326.

22. S. Di Carlo, P. Prinetto, P. Trotta and J. Andersson, "A portable open-source controller for safe Dynamic Partial Reconfiguration on Xilinx FPGAs," 2015 25th International Conference on Field Programmable Logic and Applications (FPL), London, 2015, pp. 1-4.

23. M. Liu, W. Kuehn, Z. Lu and A. Jantsch, "Run-time Partial Reconfiguration speed investigation and architectural design space exploration," 2009 International Conference on Field Programmable Logic and Applications, Prague, 2009, pp. 498-502.

24. K. Vipin and S. A. Fahmy, "A high speed open source controller for FPGA Partial Reconfiguration," Field-Programmable Technology (FPT), 2012 International Conference on, Seoul, 2012, pp. 61-66.

25. K. Vipin and S. A. Fahmy, "ZyCAP: Efficient Partial Reconfiguration Management on the Xilinx Zynq," in IEEE Embedded Systems Letters, vol. 6, no. 3, pp. 41-44, Sept. 2014.

26. K. Vipin and S. A. Fahmy, "Efficient region allocation for adaptive partial reconfiguration," 2011 International Conference on Field-Programmable Technology, New Delhi, 2011, pp. 1-6.

27. K. Vipin and S. A. Fahmy, "Automated Partitioning for Partial Reconfiguration Design of Adaptive Systems," 2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops, and Phd Forum, Cambridge, MA, 2013, pp. 172- 181.

28. K. Vipin and S. A. Fahmy, "Automated Partial Reconfiguration Design for Adaptive Systems with CoPR for Zynq," 2014 IEEE 22nd Annual international Symposium on Field-Programmable Custom Computing Machines, Boston, MA, 2014, pp. 202-205.
29. A. Montone, M. Santambrogio, D. Sciuto and S. Memik, "Placement and Floorplanning in Dynamically Reconfigurable FPGAs", ACM Transactions on Reconfigurable Technology and Systems, vol. 3, no. 4, pp. 1-34, 2010.
30. S. Yousuf and A. Gordon-Ross, "An Automated Hardware/Software Co- Design Flow for Partially Reconfigurable FPGAs," 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Pittsburgh, PA, 2016, pp. 30-35.
31. A. Jara-Berrocal and A. Gordon-Ross, "Runtime Temporal Partitioning Assembly to Reduce FPGA Reconfiguration Time," 2009 International Conference on Reconfigurable Computing and FPGAs, Quintana Roo, 2009, pp. 374-379.
32. M. Srinivas and C. K. Mohan, "Efficient clustering approach using incremental and hierarchical clustering methods," The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, 2010, pp. 1- 7.
33. C. Bobda and A. Ahmadinia, "Dynamic interconnection of reconfigurable modules on reconfigurable devices," in IEEE Design & Test of Computers, vol. 22, no. 5, pp. 443-451, Sept.-Oct. 2005.
34. Xilinx Inc. “Partial Reconfiguration User Guide UG909" v2016.1, April 2016.
35. Xilinx Inc.” 7 Series FPGAs Configuration User Guide UG470” v1.11, September 2016.
36. Xilinx Inc. “Zynq-7000 All Programmable SoC Technical Reference Manual UG585” v1.11, September 2016.
37. Xilinx Inc. “7 Series FPGAs Configurable Logic Block UG474” v1.8, September 2016.
38. Xilinx Inc. “7 Series FPGAs Memory Resources UG473” v1.12, September 2016.
39. Xilinx Inc. “7 Series DSP48E1 Slice UG479” v1.9, September 2016.
40. Xilinx Inc. “AXI HWICAP PG134” v3.0, October 2016.
41. Xilinx Inc. “Partial Reconfiguration Controller PG193” v1.1, October 2016.
42. Louise H. Crockett, Ross A. Elliot, Martin A. Enderwitz, and Robert W. Stewart. 2014. The Zynq Book: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 all Programmable Soc. Strathclyde Academic Media, UK.
43. Xilinx Inc. “ZC702 Evaluation Board for the Zynq-7000 XC7Z020 All Programmable SoC UG850”, September 2015.
44. C. Kohn “Partial Reconfiguration of a Hardware Accelerator on Zynq-7000 All Programmable SoC Devices XAPP1159”, Xilinx Inc.Application Notes, 2013.
45. “Digital cellular telecommunications system (Phase 2+); Channel coding,” 3GPP TS 45.003 version 12.1.0 Release 12.
46. “Universal Mobile Telecommunications System (UMTS); Multiplexing and channel coding (FDD),” 3GPP TS 25.212 version 12.0.0 Release 12.

47. “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding,” 3GPP TS 36.212 version 12.2.0 Release 12.
48. “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications,” IEEE Computer Society LAN MAN Standards Committee and others.
49. “Universal Mobile Telecommunications System (UMTS); Physical layer – general description,” 3GPP TS 25.201 version 12.0.0 Release 12.
50. “Universal Mobile Telecommunications System (UMTS); Physical channels and mapping of transport channels onto physical channels (FDD),” 3GPP TS 25.211 version 12.1.0 Release 12.
51. “Universal Mobile Telecommunications System (UMTS); Spreading and modulation (FDD),” 3GPP TS 25.213 version 12.0.0 Release 12.
52. “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception,” 3GPP TS 36.101 version 12.9.0 Release 12.
53. “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); LTE physical layer, General description,” 3GPP TS 36.201 version 12.0.0 Release 12.
54. “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation,” 3GPP TS 36.211 version 12.5.0 Release 12.
55. “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures,” 3GPP TS 36.213 version 12.5.0 Release 12.

Appendix A: List of Publications

1. ELdin, A. K., A. Mohamed, A. Nagy, Y. Gamal, A. Shalash, Y. Ismail, and H. Mostafa, "Design Guidelines for the High-Speed Dynamic Partial Reconfiguration Based Software Defined Radio Implementations on Xilinx Zynq FPGA", International Symposium on Circuits and Systems (ISCAS 2017), Baltimore, USA, IEEE, May 2017. DOI: 978-1-4673-6853-7
2. ELdin, A. K., K. Mohamed, A.Shalash, A.M.Obeid, Y. Ismail, and H. Mostafa, "A Reconfigurable Hardware Platform Implementation for Software Defined Radio using Dynamic Partial Reconfiguration on Xilinx Zynq FPGA", IEEE International Midwest Symposium on Circuits and Systems (MWCAS 2017), Boston, MA, USA, IEEE, In Press

الملخص

نظام الراديو المعرف برمجياً هو نظام اتصـــالات راديو مصـــمم لكى تكون فيه الطبقة المادية قابلة للتنفيذ برمجياً (معرفة برمجياً). إن الجيل القادم من النظم اللاســلكية هو عبارة عن نظم معرفة برمجياً / متغيرة فيزيائياً حيث تتميز بدعم العديد من معايير الاتصـــالات اللاســلكية التى تحتاج الى المزيد من معدلات البيانات العالية، منصات فيزيائية مناسبة الحجم قابلة للتغير، و قليلة استهلاك الطاقة.

يعتبر حقل مصفوفات البوابات المنطقية منصة فيزيائية متغيرة واعدة لتنفيذ النظام المعرف برمجياً باستخدام تقنيات إعادة التشـــكيل الجزئى الديناميكى لتحقيق اعلى معدل إعادة تشـــكيل و تقليل عدد الموارد الفيزيائية المستخدمة وتقليل استهلاك الطاقة.

فى هذه الأطروحة، يتم تصـــميم و تنفيذ و إختبار إعادة التشـــكيل الجزئي الديناميكي لنظام الراديو المعرف برمجيا الذي يدعم العديد من معايير الاتصـالات. العديد من تقنيات إعادة التشكيل الجزئي الديناميكي تم إستخدامه لتقليل زمن التحويل بين معايير الاتصـــالات المختلفة لتحقيق اعلى معدلات بيانات. ولتقليل عدد الموارد الفيزيائية والطاقة المســـتهلكة تم اقتراح وتطبيق خوارزميات لإيجاد أفضـــل توزيع لعدد الموارد الفيزيائية للاجزاء المعاد تشـكيلها لنظام الراديو المعرف برمجياً على حقل مصـــفوفات البوابات المنطقية. لذا، تقدم هذه الرســـالة تصـــميم مبتكر لتنفيذ نظام راديو معرف برمجياً باســـتخدام تقنيات إعادة التشـــكيل الجزئي الديناميكي وتقييمه مع الأعمال ذات الصلة.

مهندس: أحمد كمال الدين أحمد السيد

تاريخ الميلاد: 20/5/1990

الجنسية: مصرى

تاريخ التسجيل: 1/10/2012

تاريخ المنح:

القسم: هندسة الإلكترونيات والإتصالات الكهربية

الدرجة: ماجستير العلوم

المشرفون:

أ.د. أحمد فاروق شلش

د. حسن مصطفى حسن

د. عبد الفتاح محمد عبيد

الممتحنون:

أ.د. أحمد فاروق شلش (المشرف الرئيسي)

د. حسن مصطفى حسن (عضو)

د. عبد الفتاح محمد عبيد (عضو)

أ.د أمين محمد نصار (الممتحن الداخلي)

أ.د. السيد مصطفى سعد (الممتحن الخارجي – أستاذ بهندسة حلوان)

عنوان الرسالة:

تقنيات تنفيذ إعادة التشكيل الجزئي الديناميكى لنظام الراديو المعرف برمجيا

الكلمات الدالة:

إعادة التشكيل الجزئي الديناميكى، نظام الراديو المعرف برمجيا، مصفوفات البوابات المنطقية القابلة للبرمجة، النظم متغيرة التشّكل.

ملخص الرسالة:

.....

.....

.....

.....

.....

.....

.....

تقنيات تنفيذ إعادة التشكيل الجزئي الديناميكى لنظام الراديو المعرف برمجياً

اعداد
أحمد كمال الدين أحمد السيد

رسالة مقدمة إلى كلية الهندسة - جامعة القاهرة
كجزء من متطلبات الحصول على درجة ماجستيرالعلوم
في
هندسة الإلكترونيات والإتصالات الكهربية

يعتمد من لجنة الممتحنين:

الاستاذ الدكتور: أحمد فاروق شلش	المشرف الرئيسى
الاستاذ الدكتور: حسن مصطفي حسن	عضو
الاستاذ الدكتور: عبد الفتاح محمد عبيد	عضو
الاستاذ الدكتور: أمين محمد نصار	الممتحن الداخلي
الاستاذ الدكتور: السيد مصطفى سعد (أستاذ بهندسة حلوان)	الممتحن الخارجي

كليــة الهندســة - جامعــة القاهـرة
الجيـزة - جمهوريـة مصـر العربيــة
2017

تقنيات تنفيذ إعادة التشكيل الجزئي الديناميكى لنظام الراديو المعرف برمجياً

اعداد
أحمد كمال الدين أحمد السيد

رسالة مقدمة إلى كلية الهندسة - جامعة القاهرة
كجزء من متطلبات الحصول على درجة ماجستيرالعلوم
في
هندسة الالكترونيات والاتصالات الكهربية

تحت اشراف

أحمد فاروق شلش
أستاذ دكتور بقسم هندسة
الإلكترونيات والإتصالات
الكهربية
كلية الهندسة ــ جامعة القاهرة

حسن مصطفى حسن
دكتور بقسم هندسة
الإلكترونيات والإتصالات
الكهربية
كلية الهندسة ــ جامعة القاهرة

عبد الفتاح محمد عبيد
دكتور بقسم
الإلكترونيات والإتصالات الكهربية
مدينة الملك عبد العزيز للعلوم والتكنولوجيا ـ المملكة العربية السعودية

كليــة الهندســة - جامعــة القاهـرة
الجيـزة - جمهوريـة مصـر العربيــة
2017

تقنيات تنفيذ إعادة التشكيل الجزئي الديناميكى لنظام الراديو المعرف برمجياً

اعداد

أحمد كمال الدين أحمد السيد

رسالة مقدمة إلى كلية الهندسة - جامعة القاهرة
كجزء من متطلبات الحصول على درجة ماجستير العلوم
في
هندسة الالكترونيات والاتصالات الكهربية

كليــة الهندســة - جامعــة القاهــرة
الجيـزة - جمهوريـة مصـر العربيــة
2017