SOFTWARE DEFINED RADIO USING DYNAMIC PARTIAL RECONFIGURATION

By

Ahmed Mohamed Sadek Mabrouk

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

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2016

SOFTWARE DEFINED RADIO USING DYNAMIC PARTIAL RECONFIGURATION

By

Ahmed Mohamed Sadek Mabrouk

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

Under the Supervision of

Prof. Amin Nassar
Professor
Electronics and Communications Department
Faculty of Engineering, Cairo University

Dr. Hassan Mostafa
Assistant Professor
Electronics and Communications Engineering Department
Faculty of Engineering, Cairo University

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2016

SOFTWARE DEFINED RADIO USING DYNAMIC PARTIAL RECONFIGURATION

By

Ahmed Mohamed Sadek Mabrouk

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

Approved by the Examining Committee:

Prof. Amin Nassar, Thesis Main Advisor

Prof. Third E. Name, Thesis Advisor

Prof. Second S. Name, Internal Examiner

Prof. First S. Name, External Examiner
(Some Faculty, Some University)

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2016

Engineer's Name:	Ahmed Mohamed Sadek Mabrouk
Date of Birth:	07/06/1983
Nationality:	Egyptian
E-mail:	ahmd.sadk@gmail.com
Phone:	+201000765843
Address:	Electronics and Communications Department, Cairo University, Giza 12613, Egypt
Registration Date:	01/10/2010
Awarding Date:	dd/mm/yyyy
Degree:	Master of Science
Department:	Electronics and Communications

Insert photo here

Supervisors:

Prof. Amin Nassar
Dr. Hassan Mostafa

Examiners:

Prof. Amin Nassar	(Thesis Main Advisor)
Prof. Third E. Name	(Thesis Advisor)
Prof. Second S. Name	(Internal Examiner)
Prof. First S. Name	(External Examiner) (Some Faculty, Some University)

Title of Thesis:

Software Defined Radio using Dynamic Partial Reconfiguration

Key Words:
Software Defined Radio; Dynamic Partial Reconfiguration; FPGA; Xilinx-5; Cognitive Radio; Reconfigurable Computing; 2G; 3G; LTE; WIFI

Summary:
A novel design is proposed for implementing different communication chains using adaptable single chain and results shows an improvement in area and power consumption. This proposed work shows the advantages of using FPGA feature, Dynamic Partial Reconfiguration (DPR), in the implementation of Software Defined Radio (SDR) System that can switch among different communication standards such as 2G, 3G, LTE, and WIFI. The SDR system is being hardware reconfigured real time to simulate Multi-Standard / Multi-Mode communication systems, where the reconfiguration is being held on FPGA partially and dynamically while the FPGA is fully functioning.

Acknowledgments

Alhamdulillah, all praises and gratitude to Allah, the almighty, for all his blessings and support me with the strength and health to complete this thesis. I would like to thank my academic advisors Prof. Amin Nassar and Dr. Hassan Mustafa for supervising and encouraging me during my thesis. Also, I would like to thank Prof. Serag El-Din Habib for providing me with Xilinx kit for Hardware implementation and testing. I would like to thank Ahmed Atteya, Sharad Sinha and Hasan Baig for their time and long support in transferring knowledge to me.

Dedication

To my mother (Wafaa) soul, my father (Mohamed), my wife (Hoda), my son (Abd ALLAH), family (Nehal, Nisma, Mohamed) and nephews (Logy, Jana, Malek, Maryem) who supported me to complete this work.

Table of Contents

Acknowledgments **i**

Dedication **ii**

Table of Contents **iii**

List of Tables **vi**

List of Figures **vii**

List of Nomenclature **ix**

Abstract **xi**

1 Introduction **1**

1.1 Motivation 1
1.2 Communication system 2
1.3 Software Defined Radio (SDR) 3
1.3.1 Terms used with SDR 4
1.3.2 Benefits and costs of SDR 4
1.3.3 SDR platforms 5
1.4 FPGA Dynamic Partial Reconfiguration (DPR) 5
1.5 Idea of research 7
1.6 Organization of the thesis 7

2 Dynamic Partial Reconfiguration **8**

2.1 FPGA overview 8
2.1.1 Xilinx Virtex-5 10
2.1.2 Configurable Logic Blocks (CLBs) 10
2.1.3 DSP and Block RAM 11
2.1.4 MicroBlaze softcore processor 13
2.2 FPGA reconfiguration 13
2.2.1 Advantages and disadvantages of reconfiguration 14
2.2.2 Reconfigurable FPGAs 15
2.2.3 Reconfiguration time overhead 15
2.2.4 DPR terms 17
2.3 Partial reconfiguration factors 18
2.3.1 Configuration mode 18
2.3.2 Reconfigurable module style based 20

2.3.3 Configuration memory array types . . . 21
2.3.4 Type of reconfiguration . . . 21
2.4 Advanced topics on partial reconfiguration . . . 22
2.4.1 Reconfigurable partition style . . . 22
2.4.2 Connection constraints . . . 22
2.4.3 Configuration memory layers (3D-FPGAs) . . . 23
2.4.4 DPR in time domain (4D) . . . 23
2.5 Summary . . . 24

3 Channel coding DPR implementation using MicroBlaze 26
3.1 Forward error correction . . . 26
3.1.1 Convolutional encoders . . . 26
3.1.2 2G convolutional encoder . . . 27
3.1.3 3G convolutional encoder . . . 27
3.1.4 4G convolutional encoder . . . 28
3.1.5 WIFI convolutional encoder . . . 29
3.2 Convolutional encoder summary . . . 30
3.3 Lab setup . . . 30
3.4 General Encoder Module (GEM) . . . 31
3.5 Single-Loaded Encoder Module (SLEM) . . . 32
3.6 Results for the two systems . . . 33
3.6.1 Area occupied on the FPGA . . . 33
3.6.2 Memory needed . . . 35
3.6.3 Power estimation . . . 35
3.6.4 Time overhead . . . 36
3.6.5 PTF triangle . . . 36
3.7 Conclusion . . . 37

4 SDR chain implementation 38
4.1 Communication standards similarities . . . 38
4.2 Implemented design . . . 39
4.3 General system . . . 42
4.4 Lab setup . . . 45
4.5 System results . . . 45
4.5.1 Area occupied on the FPGA . . . 51
4.5.2 Memory needed . . . 52
4.5.3 Power estimation . . . 52
4.5.4 Time overhead . . . 52
4.6 Conclusion . . . 53

5 Conclusion and Future work 54
5.1 Future work will include . . . 55

List of Publications 56

References 57

A Xilinx Design Flow **60**
A.1 Design Flow 60
A.1.1 FPGA design flow 60
A.1.2 ISE Suit for developing Xilinx FPGA 61
A.1.3 Xilinx DPR Design Flow 62

Arabic Abstract **ا**

List of Tables

1.1 OSI model 3

2.1 FPGA configuration modes[19] 18

3.1 Convolutional encoders used in different communications systems 30
3.2 Area utilization for SLEM and GEM 35
3.3 Memory needed for SLEM and GEM 35
3.4 Power consumption 36
3.5 Configuration and reconfiguration time 36

4.1 SLM design runs list 46
4.2 SLM selected design runs 46
4.3 Design area summary 51
4.4 Number of LUTs used per design run versus general design 52
4.5 Memory needed for SLM and GCSM 52
4.6 Power consumed in the SDR chain 53

List of Figures

1.1 Simple communication system 3
1.2 Ideal SDR system 4
1.3 Partial reconfiguration 6
1.4 Value vs volume for ASIC and FPGA 6

2.1 FPGA internal construction 9
2.2 Softcore and Hardcore processor 11
2.3 CLB routing matrix in Virtex-5 12
2.4 CLB Row and Column relationship in Virtex-5 12
2.5 MicroBlaze block diagram 13
2.6 FPGA configuration types 16
2.7 FPGA configuration time overhead 17
2.8 FPGA configuration modes 19
2.9 Reconfigurable module style based 20
2.10 Configuration memory array types 21
2.11 Reconfigurable partition styles 22
2.12 Connection constrains between static and dynamic regions 23
2.13 Configuration memory layers 23
2.14 DPR in time 24
2.15 DPR factors 25

3.1 Convolutional encoder window 27
3.2 Convolutional encoder example 27
3.3 Convolutional encoder used in 3G 28
3.4 Convolutional encoder used in LTE 28
3.5 Turbo encoder used in LTE 29
3.6 Convolutional encoder used in WIFI 802.11 a/g 29
3.7 Convolutional encoder lab setup 31
3.8 GEM design, switching is done using multiplexer 32
3.9 SLEM design, encoder loaded during FPGA run-time 33
3.10 GEM and SLEM floorplan 34
3.11 PTF triangle 37

4.1 Different communication chains 40
4.2 3G,4G and WIFI communication chain 41
4.3 Reconfigurable SDR chain 41
4.4 Schematic of the SDR chain 41

4.5 SDR chain floorplanning using PlanAhead 43
4.6 Full system 44
4.7 SDR chain lab setup 45
4.11 GSCM design 46
4.8 DR_1 for 3G loaded modules in the SDR chain 47
4.9 DR_2 for WIFI loaded modules in the SDR chain 48
4.10 DR_3 for LTE loaded modules in the SDR chain 49
4.12 GCSM floorplaning 50

A.1 FPGA design flow 61
A.2 Xilinx FPGA design flow 62
A.3 Xilinx DPR Design Flow 64

List of Nomenclature

Abbreviation	Description
2G	Second Mobile Generation.
3G	Third Mobile Generation.
3GPP	3rd Generation Partnership Project.
ADC	Analog to Digital Converter.
ASIC	Application Specific Integrated Circuit.
CF	Compact Flash.
CMOS	Complementary Metal Oxide Semiconductor.
CR	Cognitive Radio.
DAC	Digital to Analog Converter.
DPR	Dynamic Partial Reconfiguration.
EDK	Embedded Development Kit.
FEC	Forward Error Correction.
FPGA	Field Programming Gate Array.
GEM	General Encoder Module.
GSM	Global System for Mobile communications.
ICAP	Internal Configuration Access Port.
IEEE	Institute of Electrical and Electronic Engineers.
ISE	Integrated Synthesis Environment.
JTAG	Joint Test Action Group.
LTE	Long Term Evolution.
LUT	Look Up Table.
OFDM	Orthogonal Frequency Division Multiplexing .
PBS	Partial Bit Stream.
PLB	Programmable Logic Block.
PRM	Partially Reconfigurable Modules.
PRR	Partial Reconfigurable Region.

SDK	Software Development Kit.
SDR	Software Defined Radio.
SLEM	Single Loaded Encoder Module.
SoC	System on Chip.
SRAM	Static Random Access Memory.
UMTS	Universal Mobile Telecommunications System.
WCDMA	Wideband Code Division Multiple Access.
XPA	Xilinx Power Analyzer.
XPS	Xilinx Platform Studio.
XST	Xilinx Synthesis Technology.

Abstract

The Software Defined Radio (SDR) is a communication system designed with its physical layer that can be implemented as software blocks. Unlike the normal radio transceivers, where communication blocks are built in a fixed environment and optimized performance to process certain waveform. The SDR software blocks can be easily configured and tolerated using software to process many waveforms. As the flexibility in the digital front-end reconfiguration increases, this allows SDR to be achievable. Where the same set of hardware can operate Multi-Standard Communication Systems (MSCS). The benefits of the SDR increase when the hardware reconfiguration is flexible to perform dynamic and real-time reconfiguration. In other words, the hardware flexibility allows the SDR dynamic system to implement different waveforms and takes place real-time without the need to switch off the system. The hardware reconfiguration concept exists from decades and passes through many phases. Field Programmable Gate Array (FPGA) is considered one of the best solutions for implementing reconfigurable hardware. Where in this thesis the FPGA new technique, Dynamic Partial Reconfiguration (DPR), is used in a novel design to allow a hardware switching among different wireless communication standards. Using DPR in SDR combines the hardware performance and software flexibility to perform the SDR.

In this work, SDR using DPR is presented in two parts. The first part serves as a proof of concept for switching among different convolutional encoders used in different communication standards and different modes of operation per each standard. Where embedded systems and System on Chip (SoC) concepts are adopted to perform the dynamic and real-time switching. In this part the DPR concept is verified, where convolutional encoders used in different modes of operation of 2G, 3G, LTE and WIFI are realized and tested in two different ways. The first implementation is realizing the full encoders on the same chip at the same time, named as General Encoder Module (GEM) design. In the second implementation, a switching is performed by adopting SoC concepts and FPGA DPR technique, named as Single Loaded Encoder Module (SLEM) design. In SLEM design, only one encoder is loaded per time on the demand of being used. A comparison between the two designs shows improvement in size by 67% and in power by 64% on using DPR technique in the SLEM implementation compared to the GEM implementation. On the contrary, a negligible latency and small extra memory size are required in the SLEM implementation.

In the second part of the thesis, the DPR concept is verified for SDR through implementing a preliminary chain which is used in the transmitter of the different wireless communication standards such as 3G, LTE and WIFI. A real-time reconfiguration of the digital communication blocks for these standards is done using DPR technique on the FPGA. The reconfiguration is done and tested among different modes of operation for the 3G, LTE and WIFI, and shows how the hardware blocks can be completely changed by loading the different modules on the demand of operation for a specific standard. A comparison is done with static communication system consists of a single mode of operation per each standard.

As a conclusion, the Single Load Module (SLM) concept by using DPR for the SDR reconfigurable system allows compact system design for the limited hardware resources and extend battery life in the continuous upgrading and variety of communication standards. This also extends the radio network resources optimization using the SDR. This thesis work is implemented and tested on Xilinx Virtex 5 kit XUPV5-LX110T.

Chapter 1

Introduction

This Chapter introduces the idea of the Software Defined Radio (SDR) and provides an overview of the communication system blocks. It also discusses the choice of the FPGA in the SDR system, and how the FPGA is adopted by using the technique of Dynamic Partial Reconfiguration (DPR) in the SDR system realization. At the end of this chapter, the thesis organization is listed.

1.1 Motivation

In the last two decades, there has been tackling efforts from both the technological and industrial fields on how to increase the connectivity among people. The communication standards are being developed and upgraded to satisfy the speed and the time to handle connectivity among the users whose number is increasing with time. Consequently, this leads to the existence of different communication standards, but as a drawback the radio frequency spectrum is not utilized in an efficient way [1, 2], where the communication bands are not used simultaneously at the same time. The research in the radio spectrum utilization leads to two approaches to solve this problem, the first approach is the Intelligent Antenna (IA) which is an antenna array technology that uses spatial beamforming and signal processing algorithms to cancel interference and reuse of the space resources[3]. IA depends on the Dirty Paper Coding (DPC) technique. The second approach is the Cognitive Radio (CR) which dynamically configures the user terminals, to utilize the radio spectrum that is not used, depending on the available wireless channels detected without interfering with the other users. In other words, CR is considered a way of managing the radio spectrum in an efficient way and it can be developed using the SDR technique[4, 5].

Generally, In a Multi-Standard Communication System (MSCS) there exist two major problems, the utilization of the radio spectrum pointed to in the previous paragraph and utilization of hardware. As each standard has its own transceiver this leads to high cost, large area, high power consumption and low battery life. In the same time, the development of the central base stations and the users' devices changes tremendously to adapt to the new technologies and support the old ones. Developing hardware, upgrading and redistributing costs money and effort. These two major problems, unutilized radio spectrum and waste in hardware, lead to start searching to find a new way of reusing (reconfiguring) the same set of hardware to operate the old and new technologies. The utilization of the radio resources and the physical hardware resources can be done by offloading data transmitted between the different communication systems, and in the same time reconfiguring the hardware resources or reordering them to switch from a standard to another.

The SDR is a way of radio system implementation using software, which is used to form different waveforms. These waveforms allow the system to switch among different communication standards. The motivation of the SDR came from the existence of some physical layer blocks has the same functionality in the different communication systems like (GSM, UMTS, LTE, etc…). Note that, these standards are not used at the same time which allows their hardware resources and radio spectrum resources to be used in a more efficient way. Also, the switching among the different waveforms should be dynamic, more or less in real time. The DPR is a technique used in the Field Programmable Gate Array (FPGA), which allows hardware real time reconfigurable computing system. The DPR can be adopted using its capability of dynamically changing and partially configured, to implement real time SDR system.

1.2 Communication system

Figure 1.1 shows the main blocks in a modern communication system. It is composed of a Digital Signal Processing (DSP) unit, digital and analog converters (Digital to Analog Converter (DAC), Analog to Digital Converter (ADC)), RF front end and antenna. Because of achieving high data rates by processing communication signal digitally using software, which is more easily to develop, distribute and upgrade, the digital transceivers penetrates the traditional analog transceivers by pushing the digital and analog converters towards the antenna and pulling the communication systems more to software design on a given hardware. However, this work will concentrate on the DSP block whereas a brief description for each block is presented as follows:

Digital Signal Processing Block: In the transmitter, this block is responsible for signal adaptation to be sent over a channel. Signal adaptation includes encryption, error correction coding schemes, modulation and further more. Whereas in the receiver this block is responsible for extracting the original information sent, by reconstructing the signal using demodulation, decoding and decryption. This block increases the flexibility of the radio development.

DAC/ADC Blocks: Analog and digital converters used to transfer the signal between the analog domain and digital domain. Using ADC, the received signal is being digitized to be processed digitally using the DSP block. The digital representation depends on the sampling rate that leads to some information loss. While the DAC is reconstructing the signal to nearly the original one.

RF Front End Block: It is the classical block that contains the Low Noise Amplifier (LNA), filters and Power Amplifiers (PA). Where this block is the most challenging block in the SDR development.

Antenna: generally the antenna is a passive device used to capture the electromagnetic waves from the surrounding media, and converts it to an electrical signal. The antenna design complexity varies from a single antenna to multiple antenna arrays. Where the smart antenna is an antenna array that uses the signal processing algorithms to locate the direction of signal arrival. And the reconfigurable antenna is capable of changing its frequency for adaptable systems.

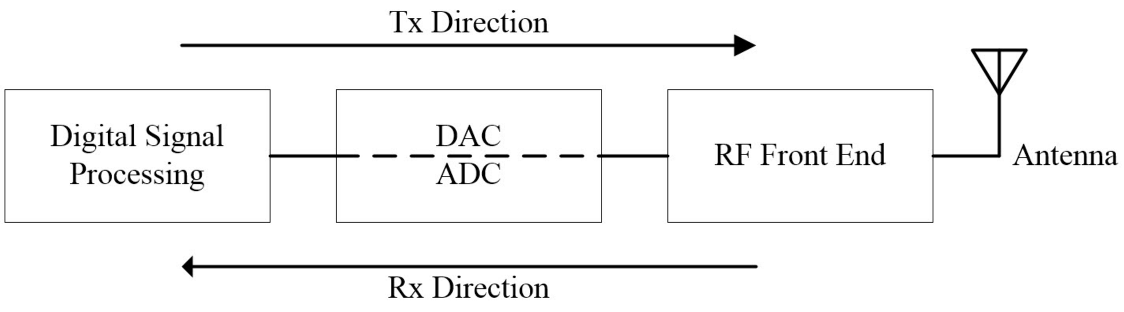


Figure 1.1: Simple communication system

1.3 Software Defined Radio (SDR)

The daily usage of communication standards is increasing. Phone calls, accessing the internet, sharing data and controlling devices are examples of modern communication usage. The devices held these standards vary in shape, functionality and the way of usage like mobile phones, wireless routers, smart chips, smart metering and even more. Although it is not easy to invent a generic device that can do everything, but it is achievable to manage the way of communication between them all. Zooming into this big picture to find adaptable communicating device to communicate the language (communication standard) of the other device. This adaptation is easier to be done through software defined modules, where these modules can change functionality by using the software. The SDR term defined by wireless innovation forum (formerly SDR forum) as "Radio in which some or all of the physical layer functions are Software Defined". The physical layer is the lowest layer in the Open Systems Interconnection (OSI) seven-layer model shown in Table 1.1. Within this layer, Radio Frequency (RF), Intermediate Frequency (IF) or baseband signals are being processed in addition to data encoding / decoding, modulation / demodulation and signal adaptation techniques.

Table 1.1: OSI model

Layer	Number	Name	Functionality	Data Type
Host	7	Application	Machine-User interfacing	Data
	6	Presentation	Encryption and compression	Data
	5	Session	Authentication and permissions	Data
	4	Transport	End-to-end connection and error control	Segment
Media	3	Network	Routing and logical addressing	Packet
	2	Data Link	Error detection and physical addressing	Frame
	1	Physical	Physical medium and signal processing	Bit stream

1.3.1 Terms used with SDR

Digital transceivers: It is transceivers, where the DAC and ADC are attached to the antenna, allowing all baseband to be processed digitally. The SDR finds its way accompanying with the digital transceiver, to easily process the signal through digital blocks in software modules which are easy to be developed and updated. Figure simplifies the ideal block diagram of the digital transceivers.

Cognitive Radio: It has been investigated that radio spectrum is not utilized efficiently. Most of the communication standards exist today are not used at same time, with different usage distribution over time. The communication systems have to be developed to be more aware of its surroundings and channel environment. From here the definition of Cognitive Radio is the Radio that can adapt itself to the surrounding environment and take actions to operate with the best performance. The SDR is utilized by the CR to manage the radio spectrum.

Waveform: The information that is sent from a source terminal had to be adopted to reach its destination with fewer errors, taking into consideration all the noise sources exist due to medium conditions. The waveform is a set of transformations applied to the information at the source to be sent and at the destination to be recovered. Figure 1.2 shows ideal SDR transceiver. A controller used to load waveform from n waveforms saved in memory storage.

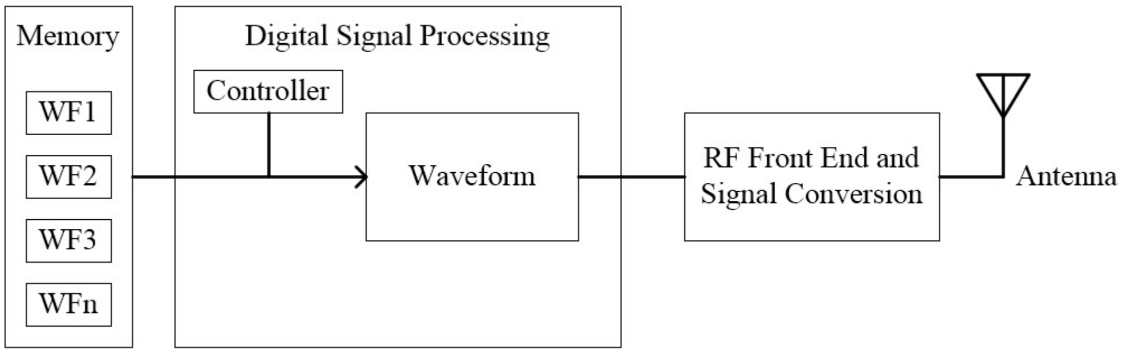


Figure 1.2: Ideal SDR system

1.3.2 Benefits and costs of SDR

There are many benefits of using SDR that it can be used in different industries and applications, hereby listing some of this advantages:

Seamless connectivity: Using SDR allows communication systems to communicate with different standards.

Adaptability: Communication systems can be adapted to the available radio spectrum and frequency reuse that helps in CR for radio resources optimization.

Updateability: As software defined, the communication modules can be easily updated and upgraded to allow new technology standards and modes of operation.

Costless: SDR operates different waveforms using the same hardware resources.

Reusability: The software modules implemented for a given standard for a product can be used with another product. That decreases the time and effort for building the same modules in other devices.

Remote upgrading: Modules can be loaded and upgraded remotely without returning back to the lab.

The flexibility of the SDR comes with some disadvantages which are not related to the idea itself but it is related to the design complexity. It takes too much time and effort from engineers to develop different waveforms that can be adopted with the same set of hardware.

1.3.3 SDR platforms

The digital signal processing part shown in Figure 1.2 can be carried on different hardware platforms such as General Purpose Processor (GPP), Digital Signal Processors (DSP) and Field Programmable Gate Array (FPGA). GPP is a microprocessor that is optimized for a powerful computing but consumes more power, it can be used in laboratories for research purpose. DSP is a microprocessor that is less power consumption than GPP but its development is more difficult than the GPP, it is used in most of the cellular terminals and basestations. FPGA is a microchip that can be configured by the user for a certain purpose, which makes it the best solution for implementing hardware blocks without unused logic gates, i.e. an 8-bit multiplier can be implemented in FPGA while in GPP 32-bit multiplier will be used for the same 8-bit data. Modern FPGAs has lower power consumption and can support up to 28 Gbps transceivers. It is also used in many SDR products that handle radio signals with bandwidth up to 50 MHz.

1.4 FPGA Dynamic Partial Reconfiguration (DPR)

One of the FPGA capabilities that is developed in the last decade is the DPR. This technique allows the FPGA to be configured partially and dynamically without switching off the system. This high flexibility of the FPGA allows it to be used in the hardware realization for SDR implementation. DPR helps in cost and resources reduction on the FPGA chip, providing a flexibility where a system can dynamically be configured without shutting it down. Moreover, this reconfiguration is done to a specific part instead of entire chip reconfiguration. Figure 1.3 illustrates a simple idea behind the DPR technique used in the modern FPGAs. Figure 1.3.a shows the full configuration of the FPGA that an application consumes more area. Figure 1.3.b shows that the size of the application can be reduced by using DPR technique. i.e, if this application has different blocks not used at the same time so these modules can be time multiplexed. Each module can be loaded to function for a certain period of time then another module to be loaded. Figure 1.3.c shows

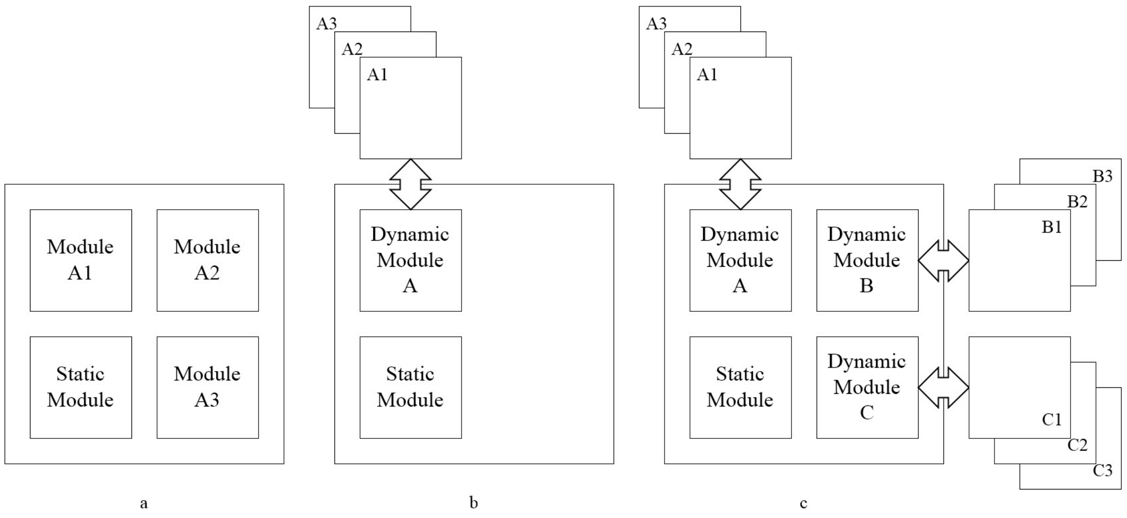


Figure 1.3: (a) Shows full FPGA configuration; (b) DPR technique to realize same system; (c) Shows how the FPGA size increased theoretically

that using DPR increases the size of the FPGA theoretically to realize more applications than regular FPGA configuration, this leads to more utilization of the FPGA resources. By applying the concept of the Partial Reconfiguration in the Software Defined Radio, It will result in a full reconfigurable wireless system which will be demonstrated through the thesis. This concept may also be generalized to other fields of study.

Another benefit of the FPGA run-time reconfiguration, that it extends the FPGA market compared to the ASIC market share. Figure 1.4 shows that the FPGA units used in market shifted to higher volumes domain. The reduction in the FPGA values is due to extending the FPGA functionality through reconfiguration.

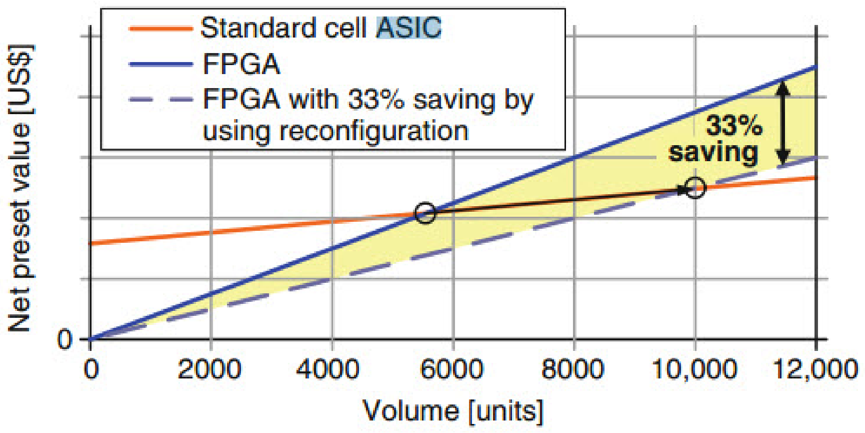


Figure 1.4: Value vs volume for ASIC and FPGA [6]

1.5 Idea of research

2G, 3G, LTE and WIFI are widely used communication standards. Implementing all of these standards on the same chip consumes area and power for each standard. Using a dynamic and reconfigurable systems reduces the chip area and power consumption. In chapter 3, a simple communication channel adaptation that switches among different channel coding schemes is presented. In chapter 4, applying the same concept to other blocks result in a completely reconfigurable system. A design of modulation chains using the DPR technique has been proposed in [8, 7, 9]. In [10], different implementations of channel coding schemes were compared. The hardware switching will be triggered during the handover between two different communication systems [11]. However, the handover takes tens of milliseconds while the hardware reconfiguration takes less time. On the other hand, many studies have been done on offloading the data traffic between cellular systems, e.g. 3G, and a fixed wireless systems, e.g. WIFI [12, 13].

1.6 Organization of the thesis

The thesis presents a way of implementing SDR using FPGA DPR technique. The following chapters are constructed as follows:

Chapter 2 presents the FPGA DPR technique. In this chapter a brief overview about the FPGA is introduced, the internal construction of the FPGA is presented for the Xilinx Virtex 5 FPGA and MicroBlaze softcore processor by Xilinx. A detailed description for the Dynamic and Partial Reconfiguration technique and how this technique is provided by Xilinx FPGA. The factors affect the DPR technique and an overview of the advanced factors.

Chapter 3 shows two different designs for convolutional encoders used in different communication systems 2G, 3G, LTE and WIFI. The first design is using full implementation technique where all encoders are implemented and exist on the chip while in the second design a DPR technique is used. In the DPR technique, the encoders are loaded on demand and exist on external storage device. DPR technique is done using internal configuration port to configure the FPGA fabric. A comparison between the two designs is presented in the chapter sessions. An embedded system is implemented on the FPGA kit and is utilized to control both designs.

Chapter 4 shows a full communication chain is adopted as SDR system, to perform the preliminary main communication blocks in different standards 3G, 4G and Wifi. The Jtag is used as an external port to configure the FPGA fabric. It shows another way of implementing the DPR using Jtag. A comparison is made between DPR design and a portion of the normal design where all hardware of the different systems exist at same time.

Chapter 5 drives the conclusion and future research to be done in the SDR, and the DPR. And how this new technique opens the door in front of massive research, in the hardware optimization for SDR.

Chapter 2

Dynamic Partial Reconfiguration

This chapter gives an introduction to the Field Programmable Gate Array (FPGA). With an overview of the internal construction of Xilinx Virtex-5, which is used in the thesis, and softcore processor MicroBlaze, which is Xilinx IP, used in the internal FPGA configuration. Reconfigurable computing terminology is presented and highlighted for the FPGA. Dynamic Partial Reconfiguration (DPR) technique is detailed explained and a complete reference to the partial reconfiguration is being illustrated in this chapter to support those who will use this technique.

2.1 FPGA overview

The FPGA is an Integrated Circuit (IC) that is electrically programmed to execute a certain application. It initially has no functionality to operate before it is programmed. FPGA is formed from a combination of transistors that are connected in a specific way. Applying an external voltage to these transistors it will operate certain functionality. This combination of transistors called Look Up Tables (LUTs). Each group of LUTs forms a Programmable Logic Blocks (PLB). These PLB blocks have been developed through many years. Recent FPGAs has different types of PLB functionality such as memory blocks that can store data for internal operations, multipliers for complex arithmetic operations, and general PLBs that is used to implement general functions from simple 2-bit adder to a complete microprocessor unit. The internal heterogeneity of FPGA PLBs is shown in figure 2.1. The FPGA internal routing consists of wires and programmable switches that allow the connections among the PLBs, memory blocks, multipliers and I/O ports. These connections are developed for best data routing and latency, sometimes with different characteristics varies from the shortest path to the fastest one. Also, there is a dedicated network of connections that takes care of clock distribution and reset signals for achieving low skew.

The LUT size, the FPGA core component, is measured by its number of inputs such as an LUT has 3 inputs will be named as 3-LUT. The number of LUTs in the PLB may be of equal size or mixture of different sizes. A study is made in [14] shows that heterogeneous mixture of LUTs (3-LUT, 4-LUT, 5-LUT, 6-LUT) is performed more efficiently than a homogeneous 3-LUT based FPGA. While the latter acquires less size than the heterogeneous structure. There are three different major techniques used to

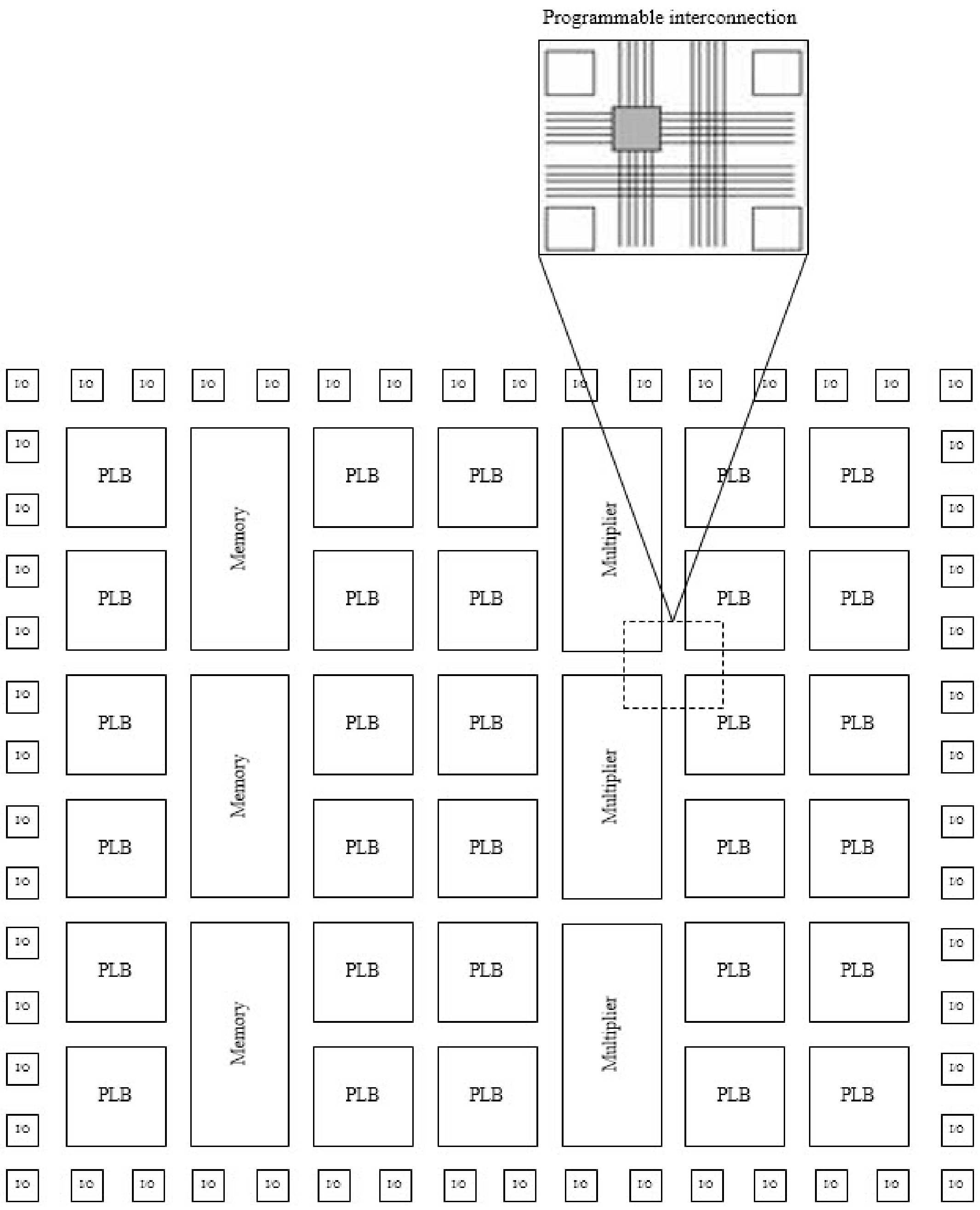


Figure 2.1: FPGA internal construction

program the FPGA LUTs. Anti-Fuse, Flash and SRAM programming technologies. The advantages of the Anti-Fuse and Flash over the SRAM, they are non-volatile and occupies a small area. While the SRAM is easily reprogrammed and use the standard CMOS process technology. Although SRAM has become the dominant approach to program the FPGA LUTs because the advantages it provides, but till now there is no technique that can combine the best of them all.

Current FPGAs has IP blocks, these IPs are standard libraries which are optimized and developed to facilitate the FPGA development. An engineer can drag and drop certain functionality instead of building the new block from scratch. IPs like accumulators, bus interfaces, encoders … etc. The microprocessors are considered one of the important IP core. There are two types of microprocessors, softcore and hardcore. The softcore processor like MicroBlaze by Xilinx is implemented using the FPGA logic gates. The hardcore processor like PowerPC by IBM is fabricated in the core of the IC of the FPGA chip and connected to FPGA fabric as shown in Figure 2.2. The main concern of the softcore processor is its limitation in speed, around 200 MHz, also, it takes many resources on the FPGA. Where there are some advantages of using softcore processor like modifying it for specific requirements, customizing instructions and multiple core system. On the other hand, using hardcore processor can achieve higher processing speeds more than 1GHz. Hence, the hardcore processor has its own fabric in the FPGA chip it doesn't occupy resources on the FPGA fabric which allows the full usage for the FPGA. The disadvantage of the hardcore is its fixed architecture that can't be modified. Zynq series by Xilinx is a perfect example of the current SoC chips, it combines ARM dual-core or quad-core microprocessor in a processing system (PS) with Xilinx FPGA fabric as a Programmable Logic (PL).

2.1.1 Xilinx Virtex-5

An example to the FPGA that is generally introduced in section 2.1, Xilinx FPGA Virtex-5-XC5VLX110T that is used in the thesis. It is FPGA chip from Xilinx Virtex-5 series. Xilinx is a major FPGA vendor of market share 50%.

2.1.2 Configurable Logic Blocks (CLBs)

The Configurable Logic Blocks (CLBs) are the main programmable logic resources in Xilinx FPGAs. The CLBs are general PLBs that is used for implementing sequential and combinational circuits on Xilinx FPGA. The XC5VLX110T has in total CLB array of 160 x 54 (Rows x Columns). In Virtex-5 series each CLB contains two slices and a switching matrix is used to switch between them as shown in figure 2.3. The 54 CLB columns contain 108 slices, where it is an important note that, on using the DPR technique a complete CLB is taken in the constraint boundaries. In other words, you cannot split the CLB while reconfiguring the FPGA. Each slice has four 6-LUTs, four flip-flops, carry-logic and multiplexers, to provide logic, arithmetic and ROM functions. Slice heterogeneity exists in Xilinx Virtex-5 that allows more area and time optimization. Some slices are different in their internal construction providing distributed RAM and 32-bit

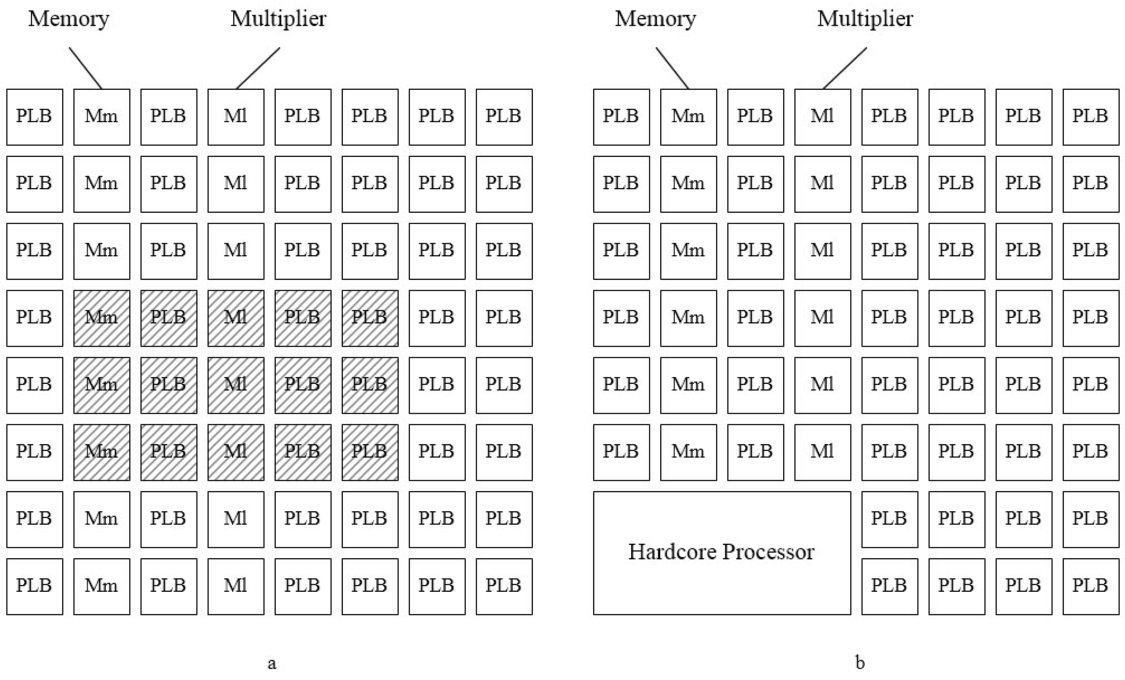


Figure 2.2: Softcore and Hardcore processor (a) Shaded part represent the implementation of softcore processor on the FPGA logic it acquires some of the available resources like PLBs, memory and multiplier blocks; (b) Hardcore processor fabricated beside the FPGA fabric

shift registers beside the main slice functions which are called SLICEM, where the normal slices denoted as SLICEL. The SLICEM is distributed in the FPGA, where some CLBs contain 2 SLICEL and other CLBs contain 1 SLICEM and 1 SLICEL. On mapping the design to the FPGA logic the type of slices should be considered, Xilinx tool PlanAhead gives an estimation for the needed slices in the design.The chip XC5VLX110T has in total 69120 6-LUTs, 69120 flip-flops, 1120 Kb distributed ram and 560 shift registers [27].The slices are numbered per rows and columns as in figure 2.4. The columns are numbered from right to left using X symbol while the rows are numbered from bottom to up using Y symbol. This numbering is important for placing and routing the design while creating the user constraint file (UCF), either manually or using PlanAhead tool. The cause of using slice numbering, that the slice numbering is written continuously from SLICE_X0Y0 to SLICE_X107Y159. There are CLBs numbers not exist because they are replaced by DSPs or Block RAMs or Input / Output (I/O) banks [27].

2.1.3 DSP and Block RAM

DSPs and Block RAM are specific PLBs exist in the architecture of Xilinx families. They are not like the CLBs that are used for general purpose implementations as described in the later paragraphs. DSPs blocks are used to implement complex arithmetic operations, they are optimized architecturally to produce same functionality can be implemented on huge numbers of CLBs. Virtex-5 include DSP48E slice which is considered DSP IP provided by Xilinx. The Block RAM is used to offer internal RAM blocks in the FPGA,

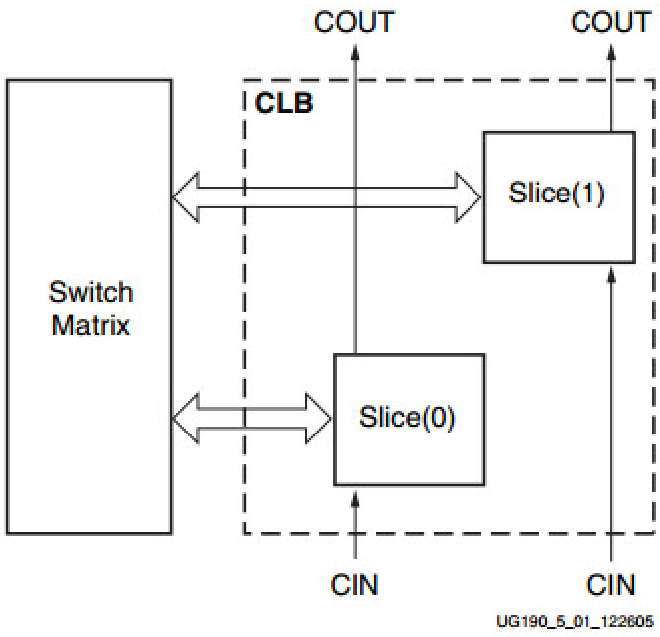


Figure 2.3: CLB routing matrix in Virtex-5[27]

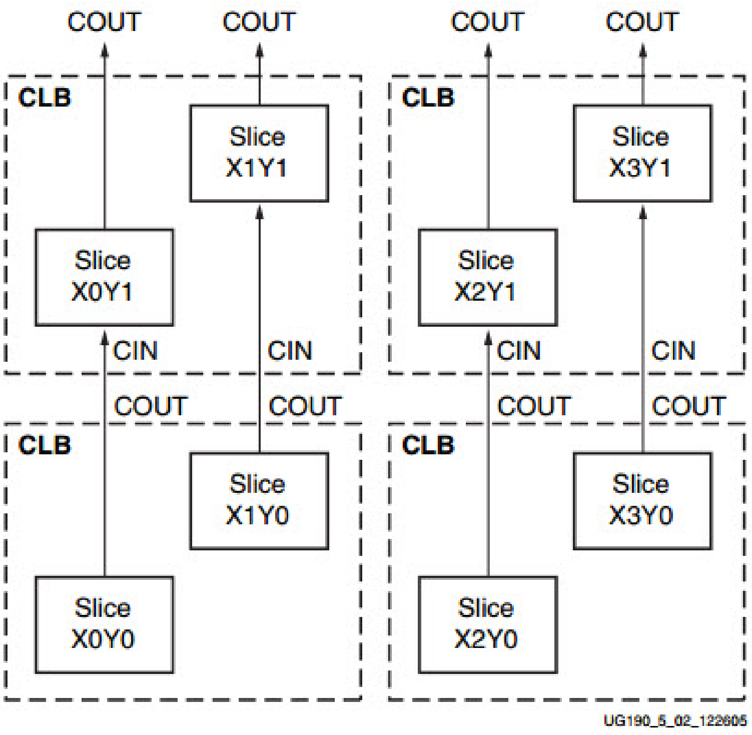


Figure 2.4: CLB Row and Column relationship in Virtex-5[27]

they are optimized for memory storage instead of harvesting the distributed CLBs memory. To clarify this, assume the need for implementing a processing unit using CLBs, this will consume a huge number of LUTs that will consume many CLBs. On using DSPs blocks, this will reduce the CLBs usage. For implementing memory, there are 2 ways, the first using distributed RAM this will consume many CLBs and the second way is using the optimized Block RAM this will save CLBs that can be used for another function.

2.1.4 MicroBlaze softcore processor

MicroBlaze is an embedded softcore microprocessor. It is a reduced instruction set computer (RISC) based architecture. MicroBlaze is optimized for implementation in Xilinx FPGAs families using a portion of the available resources on the FPGA. figure 2.5 shows the internal construction for the MicroBlaze [28].

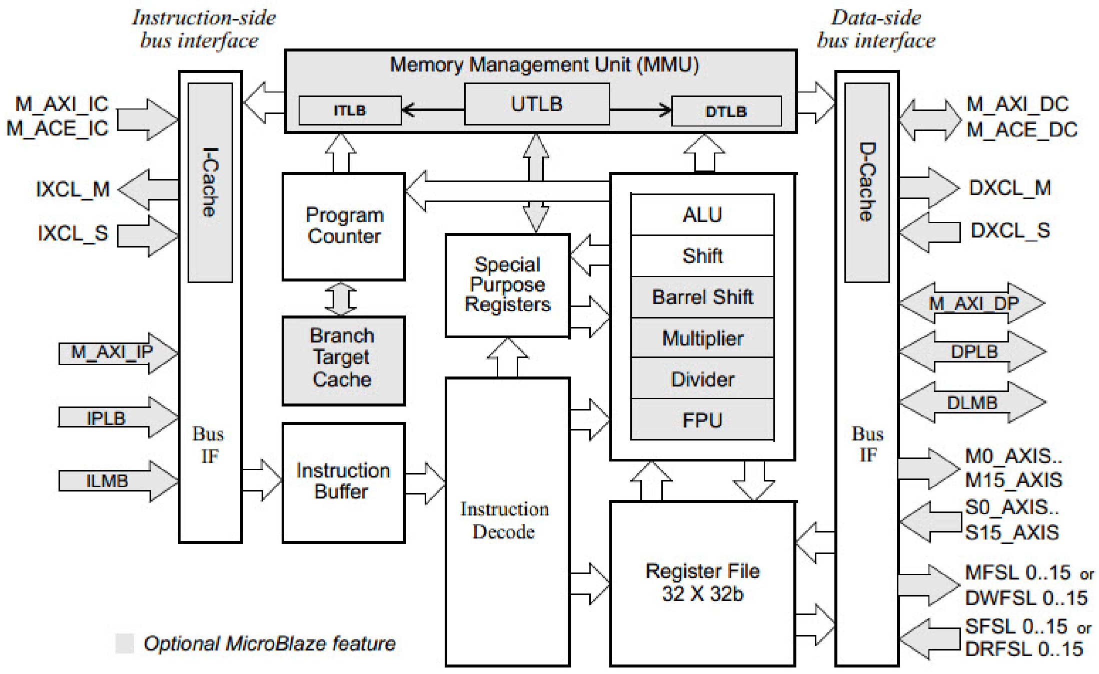


Figure 2.5: MicroBlaze block diagram[28]

2.2 FPGA reconfiguration

Reconfigurable Computing (RC) term first appears in 1960's when G. Estrin proposed the idea of "fixed plus variable structure" [15, 16]. His idea considers a fixed hardware processor controls a variable "reconfigurable" hardware arrays. The reconfigurable hardware is configured to deal with a specific task for a certain time, after this time, it will be released to be reconfigured again for another task. This resulted in hybrid Hardware-Software computer structure, which combines the software flexibility and hardware efficiency. The FPGAs flexibility allows it to be considered the best solution for the RC. Connecting it to a processing unit will achieve the desired work. Modern FPGAs chip combining FPGAs Programmable Logic (PL) and Processing System (PS) in one chip for best performance and latency like Altera Stratix 10 and Xilinx Zynq 7000.

2.2.1 Advantages and disadvantages of reconfiguration

The main advantages of the reconfigurable systems are:

Resources utilization: In ordinary design most of the hardware not used till it is triggered for operation for a certain time, then back to sleep mode waiting for another trigger. Using reconfigurable hardware will increase the resource utilization by omitting the unused part till its time of operation releasing resources to be used for other activity.

Scalability (Updatability): Using reconfigurable hardware will allow upgrading system to accommodate newly defined tasks to handle the growing 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 (Customizability): 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 is some disadvantages for the reconfigurable systems and they are improving by research such as:

Latency: Latency increased by the time needed for the reconfiguration. This item improved by the new approach of run-time reconfiguration.

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.

Scheduling complexity: Defining the time of operation for certain configuration then reconfigure device for another operation is the difficult and the important thing in the reconfiguring devices. Where Engineer has to take care for the scheduling time for loading the device part in the exact time to have the valid data.

2.2.2 Reconfigurable FPGAs

FPGA-based systems are divided into two categories, parameterized and reconfigurable. Figure 2.6 summarize the FPGA types.

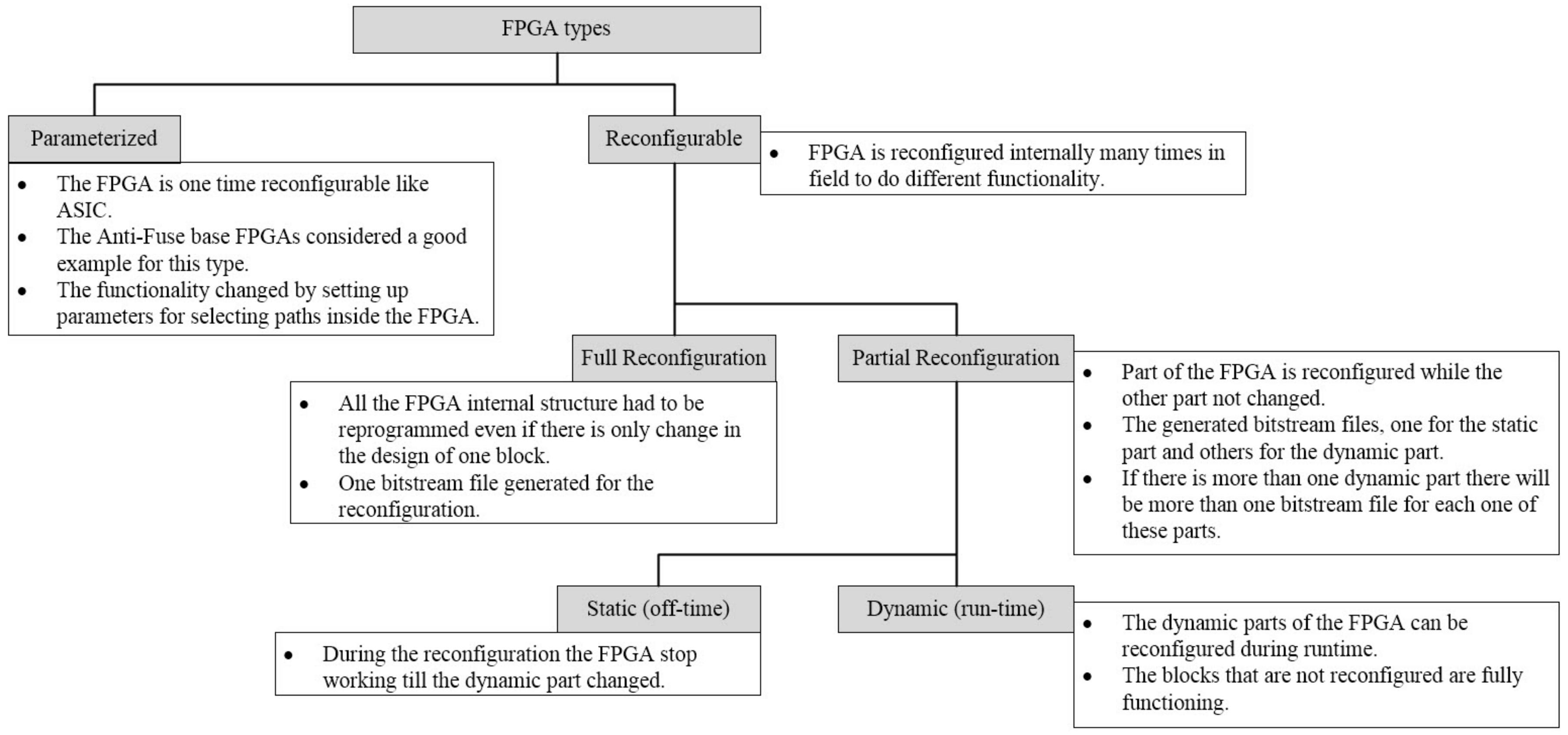
Parameterization: In this category a full image is loaded to the FPGA to run a certain application. This image does not change during run-time but new values can be set for some registers to change the mode of operation of the application, this approach is called parameterization [17]. This type is not considered a true reconfiguration because no hardware blocks swapped, but the implemented hardware is optimized and multiplexed to operate more than one function using select lines. It is more like ASIC where no new connections are added after the deployment.

FPGA reconfigurability: This category is considered a true reconfiguration, where the image of the FPGA is newly loaded and swap is occurred between different bitstreams to execute different applications. This type can be Full Reconfiguration (FR) or Partial Reconfiguration (PR). In the FR type, the downloaded bitstream will configure the whole FPGA to perform a new task. There is only one bitstream file that is loaded to the FPGA that contains all the design and the defined connections. While in the PR type a part of the FPGA is reconfigured while the other part not changed. The PR type has more than one bitstream is to be loaded on the FPGA, a static bitstream and many dynamic bitstreams for one or more dynamic partitions. In the static region, no change in the connections occurs on the FPGA that is why the static region might include the memory controller, the softcore processor, or the internal configuration port. In the dynamic part, the bitstream can be updated with different connections (bitstreams) and loaded in the FPGA.

In the PR, the dynamic part can be changed during off-time or during run-time. The off-time mode the FPGA stop working during this time till the dynamic part is replaced with the new bitstream then the system continues to work. The disadvantage of this type that a latency overhead is added. This disadvantage improved by the second type which is run-time reconfiguration or dynamic configuration. In the dynamic reconfiguration, the dynamic part is configured with new bitstream while the FPGA is running. This dynamic reconfiguration adds no latency for the FPGA operation.

2.2.3 Reconfiguration time overhead

In the parameterized FPGA configuration, the configuration takes place in the power on time. As shown in figure 2.7.a after the first time configuration the device start its operation without any other overhead time added. Figure 2.7.b shows the reconfiguration time of a reconfigurable FPGA. This figure shows the full reconfiguration where there is an overhead time for the second reconfiguration is added equally to the time of the first configuration because the full image is loaded. Figure 2.7.c shows that using partial reconfiguration the overhead time of the FPGA reconfiguration is improved because



FPGA types
Parameterized
The FPGA is one time reconfigurable like ASIC.
The Anti-Fuse base FPGAs considered a good example for this type.
The functionality changed by setting up parameters for selecting paths inside the FPGA.
Reconfigurable
FPGA is reconfigured internally many times in field to do different functionality.
Full Reconfiguration
All the FPGA internal structure had to be reprogrammed even if there is only change in the design of one block.
One bitstream file generated for the reconfiguration.
Partial Reconfiguration
Part of the FPGA is reconfigured while the other part not changed.
The generated bitstream files, one for the static part and others for the dynamic part.
If there is more than one dynamic part there will be more than one bitstream file for each one of these parts.
Static (off-time)
During the reconfiguration the FPGA stop working till the dynamic part changed.
Dynamic (run-time)
The dynamic parts of the FPGA can be reconfigured during runtime.
The blocks that are not reconfigured are fully functioning.

Figure 2.6: FPGA configuration types

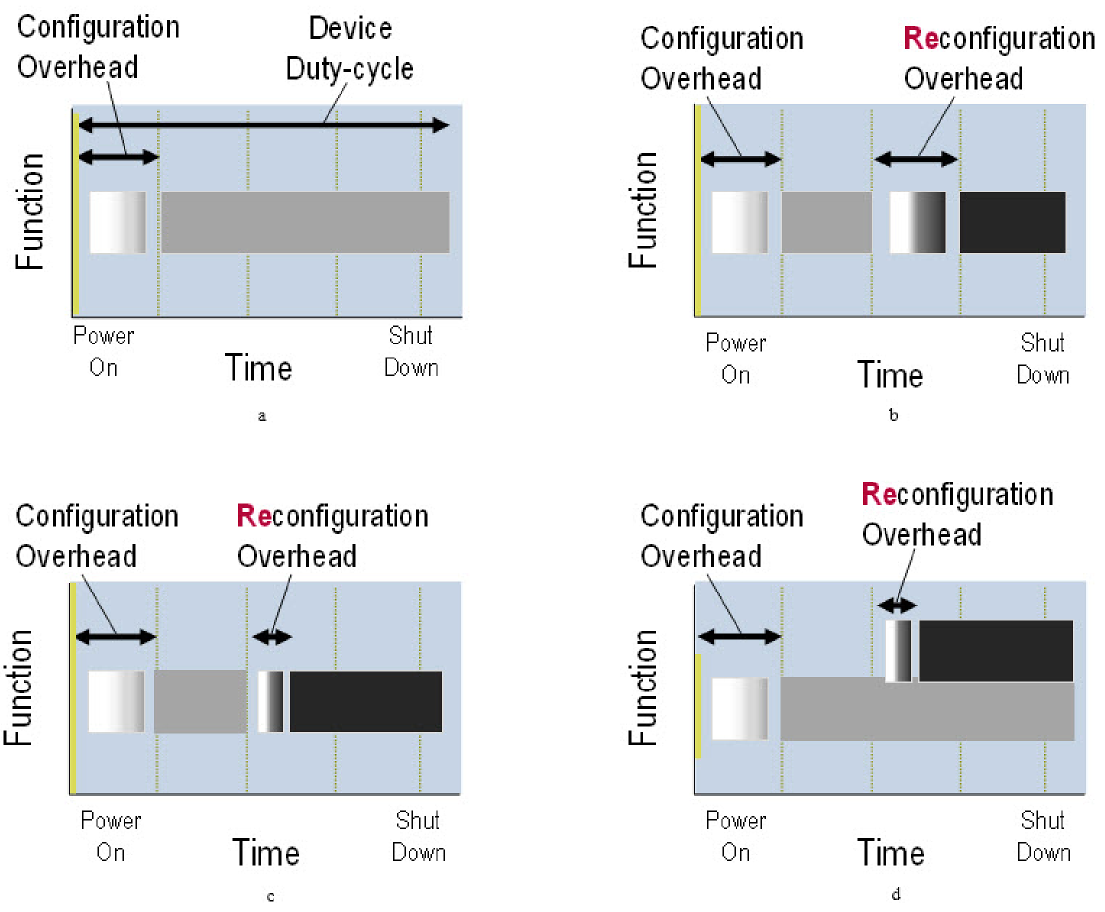


Figure 2.7: FPGA configuration time overhead [18]

only the dynamic part is reconfigured while the static part is not changed. So the reconfiguration time overhead is less than the initial configuration of the FPGA. But the static (off-time) reconfiguration has a problem of suspending the device operation while the configuration of the dynamic part takes place. In 2.7.d DPR technique is used to reconfigure the dynamic part while the FPGA is operating. There is no switch off occurs to the FPGA, so the reconfiguration overhead time is negligible.

2.2.4 DPR terms

Reconfigurable Partition (RP) is the region of the FPGA logic core that will be reconfigured, each RP can be reconfigured with one or more Reconfigurable Module (RM) that swapping occurs among them.

Reconfigurable Module (RM) is the module that contains the application to be run. It is designed using HDL or using netlist.

2.3 Partial reconfiguration factors

Partial Reconfiguration depends on many factors, which is affected by the available tools, FPGA technology, the way of configuration, and the size of configuration changes.

2.3.1 Configuration mode

The first factor is the configuration mode or how the FPGA binary's, responsible for the FPGA configuration, is loaded on the FPGA. There are two ways for loading the configuration RM data to the FPGA RP, which depends on the way of reaching the configuration plane, i.e. is the FPGA reconfigured by external or internal processing unit. The internal processing unit that controls the FPGA reconfiguration can be either softcore processor as shown in figure 2.8.a or hardcore processor as shown in figure 2.8.b. The FPGA bitstream binaries are stored on an attached memory card to the kit and a memory controller will be responsible for accessing the memory to get the configuration bitstream. For the external processing system, the FPGA will be controlled by PC, FPGA, GPP or another controller. The external controller has its own way to fetch the configuration RM data from external memory. An example is shown in figure 2.8.c for a PC reconfigure FPGA RPs through JTAG.

Internal configuration: Using Internal Configuration Access Port (ICAP) where internal processing system, softcore processor like MicroBlaze or hardcore processor like PowerPc, is responsible for reconfiguring the FPGA. ICAP is a SelectMap-like protocol to access the internal configuration memory [19]. The ICAP had to be implemented in the static region with the memory controller and the softcore processor as shown in figures 2.8.a and 2.8.b.

External configuration: Using external controllers like CPU, DSP or another FPGA as master. In the external reconfiguration, the reconfigured FPGA is used in slave mode and the connection between the two devices, the external controller and the FPGA, is carried out through either the Serial mode, JTAG or SelectMap protocols [19]. Figure 2.8.c shows an example for external configuration through JTAG.

Table 2.1 shows the different configuration modes and their speeds for Xilinx FPGA Virtex-5 [19]. The different configuration modes have different bandwidths and clock speed.

Table 2.1: FPGA configuration modes[19]

Configuration Mode	Type	Max Clock	Data Width	Max Bandwidth Bps (bps/8)
ICAP	Internal	100 MHz	32-bit	400 MBps
SelectMap	External	100 MHz	32-bit	400 MBps
Serial Mode	External	100 MHz	1-bit	12.5 MBps
JTAG	External	66 MHz	1-bit	8.25 MBps

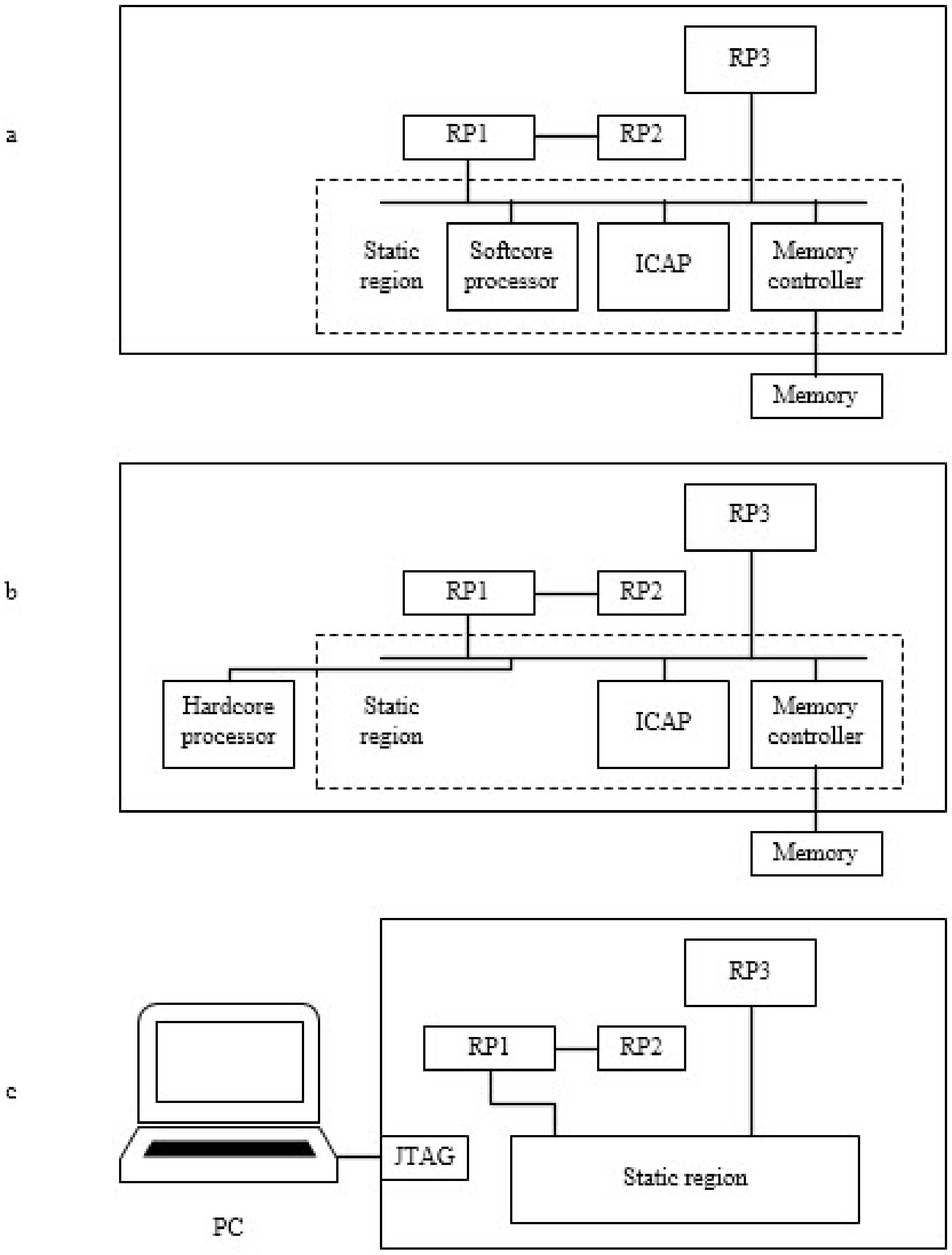


Figure 2.8: FPGA configuration modes; (a) Softcore processor reconfigure RP through ICAP; (b) Hardcore processor reconfigure RP through ICAP; (c) PC reconfigure RP through JTAG the RM stored on the HDD of the PC

2.3.2 Reconfigurable module style based

The RM style based factor is depending on the size of the part to be reconfigured in the FPGA and if the changes among the different RMs is huge or little. Upon which the needed reconfiguration might be either difference based or module based.

Difference Based: Used for small design changes to edit the connections of few LUTs [20]. The bitstream file, to be loaded on the FPGA, contains the difference between the different implementations, as shown in figure 2.9.a. Although this type is used for small design changes but it is more complex and take more time to develop. The designer should have a good understanding of the FPGA internal structure because it needs to do some manual connections for the LUTs.

Module Based: Used for large design changes by replacing a complete block with a new one[21], as shown in figure 2.9.b. This is easier in implementation than the difference based. The problem may face this type is the I/O connections that may differ from block to another.

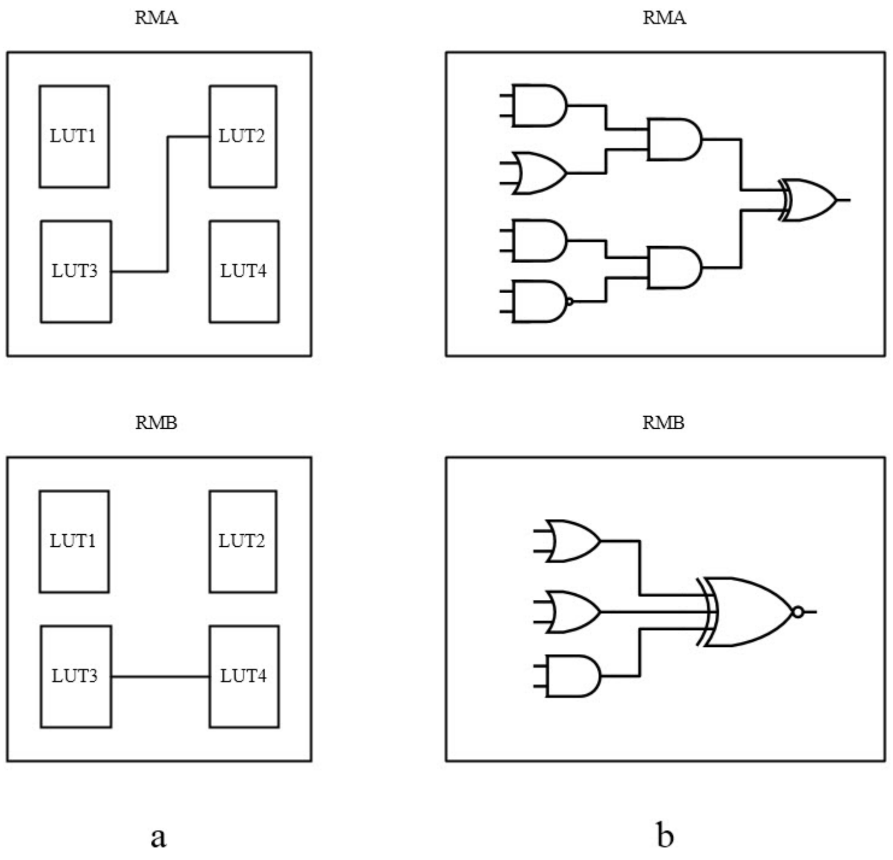


Figure 2.9: Reconfigurable module style based (a) Difference Based reconfiguration the connections between LUTs has minor update between different RMs (b) Module Based reconfiguration each RM has different internal modules and construction

2.3.3 Configuration memory array types

The internal FPGA structure and the technology used to build the FPGA blocks affect the way of the FPGA configuration and partitioning. The FPGA configuration consists of Configuration Memory (CM) array (configuration layer) that takes care of accessing the logic gates (CMOS Layer) in other words the way of activating the programmable logic used in a certain design. The way of partitioning the FPGA depends on how the configuration layer configures the logic gates. There are different FPGA CM ways to access the logic layer:

1-D: In this FPGA structure, the configuration layer access a complete FPGA array column to partition and reconfigure it with the new bitstream. This exists in old FPGAs like in Virtex-II. Figure 2.10.a shows that a complete D1 column is reconfigured with D2 column. [24, 23, 22]

2-D: In modern FPGA structures the configuration layer access the logic plane like a memory, where some cells can be accessed by row and column. This type exists in recent FPGAs families like in Virtex 4, 5, 6 and 7. The partition of the FPGA to be reconfigured can take square or rectangle shape. Figure 2.10.b shows that D1 block is reconfigured with D2 block.[25]

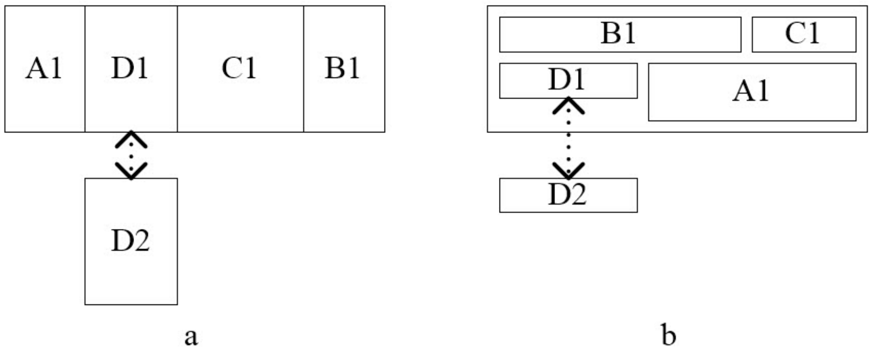


Figure 2.10: Configuration memory array types; (a) 1-D; (b) 2-D

2.3.4 Type of reconfiguration

The different ways of reconfiguring the FPGA is described in Figure 2.6 whereas the time overhead shown in Figure 2.7. This section highlights the related partial reconfiguration techniques which are the Static (off-time) and Dynamic (run-time).

Static (off-time) reconfiguration: In this type, part of the FPGA is reconfigured while suspending the work of the FPGA. For example, if the chip has a communication system implemented on it, the communication system will be halted during this type of reconfiguration.

Dynamic (run-time) reconfiguration: Run-time reconfiguration during the normal FPGA operation. This benefits communication system in the previously given example to continue functioning while reloading new image for a specific block.

2.4 Advanced topics on partial reconfiguration

2.4.1 Reconfigurable partition style

There are three types of the reconfiguration styles depends on how many RM used in the same RP. The island-style is the commonly used and exist by the FPGA vendors for its simplicity and applicability. In this type, the RMs has to be multiplexed in time in the same place but only one is loaded at a time. If the RM used not fully fill the RP there will be wasted LUTs that is not used. Internal fragmentation for the RP is proposed for more resources utilization, this fragmentation is either one-dimensional (slot-style) or two-dimensional (grid-style). In the slot-style, the multiple RMs can share the same RP adjacently while in the grid-style the multiple RMs can be fitted more in the two-dimensional space of the RP. The different RP styles [6] are illustrated in the figure 2.11, where m1, m2, m3 and m4 are different RM modules. Island-style in figure 2.11-a 2 RMs implemented each per RP slot, where there is an unused area. In figure 2.11-b Slot-style is used showing that area utilized where m1 and m2 fits with fewer widths and higher than in figure 2.11-a allowing m3 to be implemented. Grid-style in figure 2.11-c relocation to m3 allows m4 to be implemented and more area is utilized.

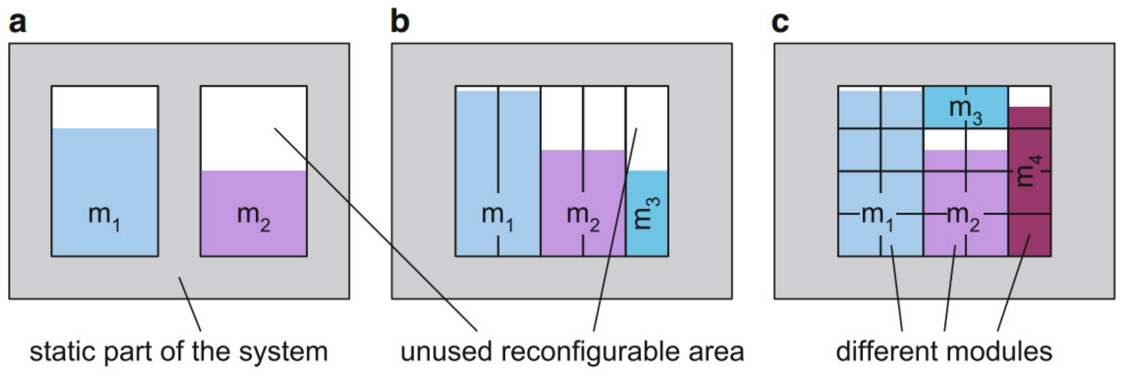


Figure 2.11: Reconfigurable partition styles [6]. (a) Island-style (b) Slot-style (c) Grid-Style

2.4.2 Connection constraints

The connection between the static part and dynamic part is a sensitive region as the static region should have all the available connections to communicate with the dynamic part. i.e. The connections on the dynamic part had to be less than or equal to that on the static part. The connection between the static and dynamic part has three types bus macro, proxy logic and blocker macro [6]. The bus macro is the old method for connecting both parts, in this method extra LUTs are used on both sides for every connection [29]. The proxy logic method is using extra LUT called proxy logic in the dynamic part corresponding to every connection with the static part [21]. This type considered a slice-based bus macro as the number of LUTs nearly halved. In [30] new type of connection is proposed called blocker macro, where PR link wire is defined for both sides. A tunnel is formed in the blocker macro from static region to dynamic region and another tunnel is formed from dynamic region to static region, where each interfacing signal is bounded by PR link wire. Figure 2.12 illustrates the different types of connections between the RP region and the static region, where the RP has two RM modules (NAND and OR functions) to replace each other.

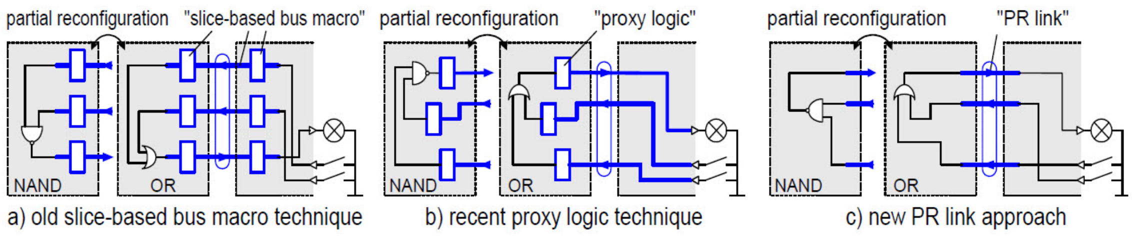


Figure 2.12: Connection constrains between static and dynamic regions[6]. (a) Bus macro; (b) Proxy logic; (c) Blocker macro

2.4.3 Configuration memory layers (3D-FPGAs)

In the 3D FPGAs, The FPGA consists of more than one configuration plane that accesses the CMOS layer. These stacked configuration planes have a switch layer between them to manage the accessing to the CMOS layer. This architecture improves the logic density, delay, and the dynamic power consumption for the conventional 2D-FPGA. Figure 2.13.a shows the conventional 2D FPGA while figure 2.13.b shows stacked configuration plane in 3D FPGA.

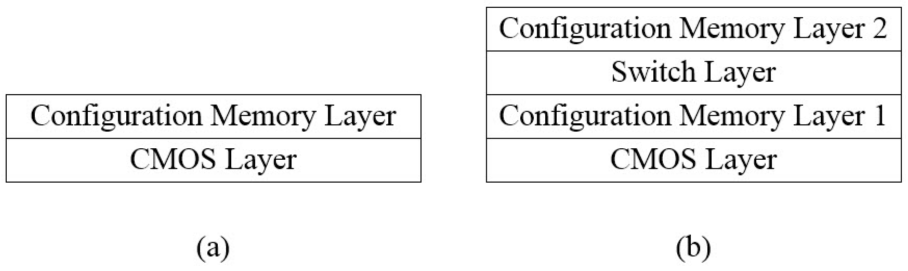

Configuration Memory Layer
CMOS Layer
(a)
Configuration Memory Layer 2
Switch Layer
Configuration Memory Layer 1
CMOS Layer
(b)

Figure 2.13: Configuration memory layers. (a) Conventional 2D FPGA; (b) Stacked 3D FPGA

2.4.4 DPR in time domain (4D)

The RP can be reconfigured with different RMs over time to run a single application. i.e. an application to be divided into parts, the first part to configure RP then released, then second part reconfigure the same RP. The data flow between the different blocks handled through memory. Figure 2.14.a shows an application divided into two blocks, figure 2.14.b shows that RM1 is loaded to RP1 and after certain time RM2 is loaded to continue the data between the two blocks is stored in memory before the RM1 released then retrieved after RM2 is loaded. This allows the conventional 2D FPGA to be reconfigured in both space and time. The advanced architecture of FPGAs is proposed by Tabula for time-multiplexed FPGA architecture named as Spacetime FPGAs [26]. Where the FPGA has different configuration memory layers denoted as a fold. The fold can be imagined as a virtual CMOS layer. Using the previous example in the Spacetime shown in 2.14.c, the RM1 is loaded to RP1 in fold 1 after released RM2 to be loaded and reconfigure PR2 in fold 2. Note that there is a shared memory between the two layers that the data out from the first stage can transfer to the next stage.

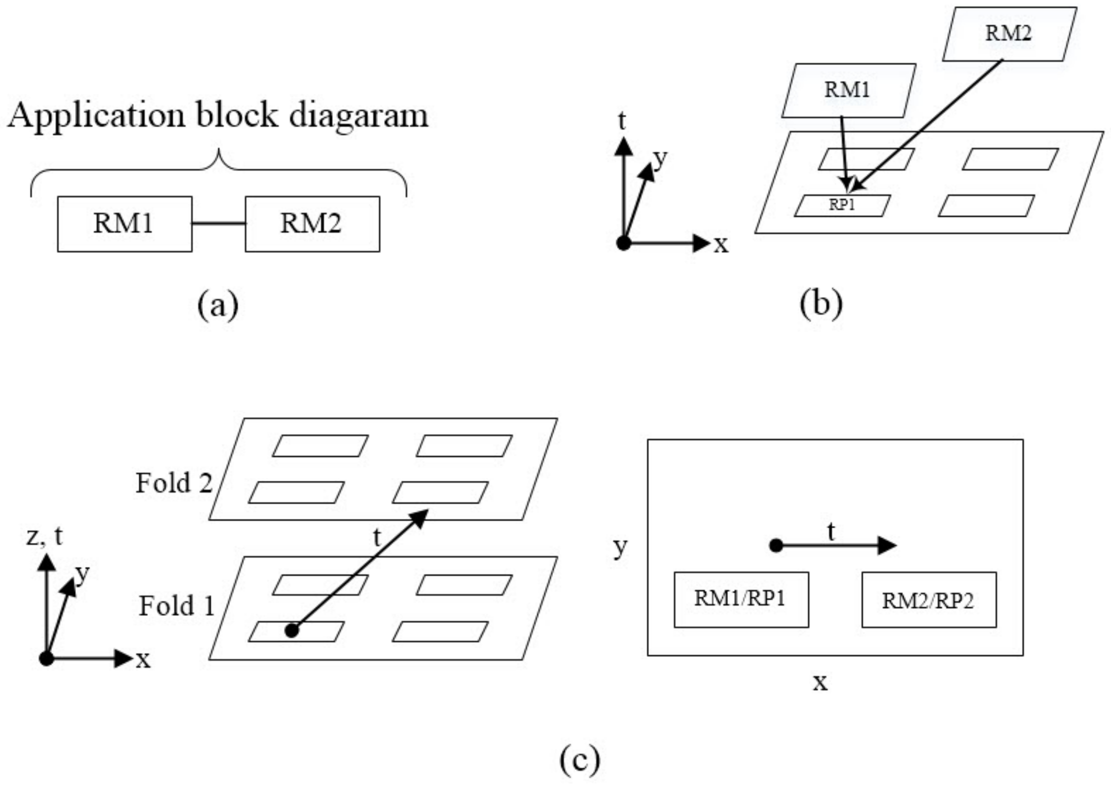


Figure 2.14: DPR in time

2.5 Summary

Partial Reconfiguration is a powerful hardware configuration technique that opens the doors in front of real time reconfiguring systems, figure 2.15 summarizing the DPR factors. Using partial reconfiguration, engineers can create separate configuration files for different communication systems waveforms and load them when needed. This will be shown in the next chapters and how this useful technique for being used in software defined radio.

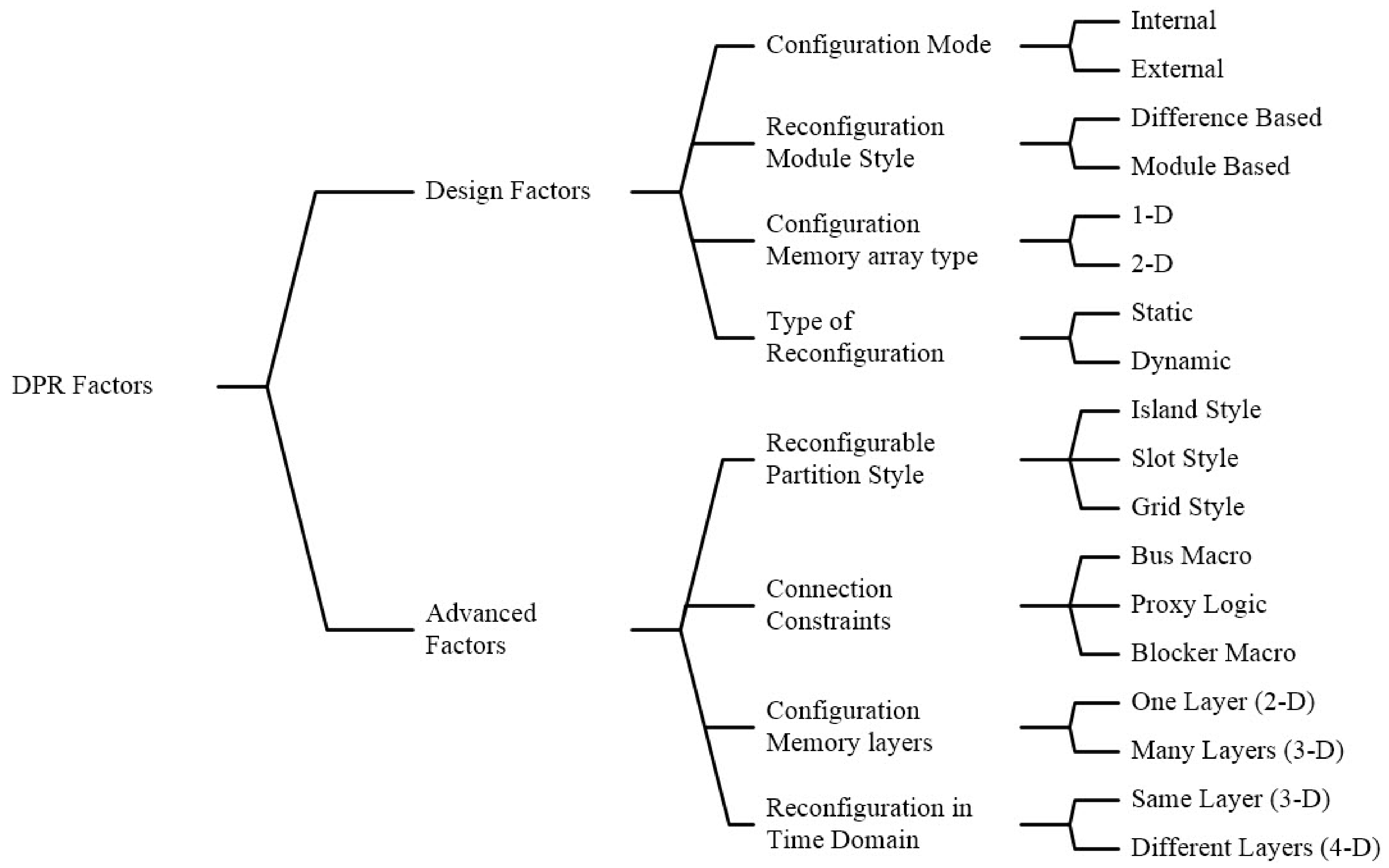


Figure 2.15: DPR factors

Chapter 3

Channel coding DPR implementation using MicroBlaze

This chapter presents two different design techniques for convolutional encoders modules used in the SDR system. The two designs are General Encoder Module (GEM) and Single-Loaded Encoder Module (SLEM). The implemented designs are used to switch among the convolutional encoders used in the different communication standards 2G, 3G, LTE and WIFI with two different concepts. The two designs are implemented and tested on XUPV5-LX110T evaluation kit.

3.1 Forward error correction

Forward Error Correction (FEC) is a channel coding technique used to improve the reliability of the data that is sent through communication channels. It is a digital technique in which a redundancy bits is added to the sent stream of data bits to decrease the effect of the noisy channel on the sent data. The redundancy bits can be the same as the sent data bits or a combination of the sent data through a known function.

3.1.1 Convolutional encoders

Convolutional encoders are one of the FEC coding schemes. They are used to add parity symbols for the data sent over a communication channel. The new frames of data are formed by a combination of the sent data as shown in figure 3.1.

The combination is done through a polynomial function has three main parameters, that are used in the convolutional encoders [N, K, L], where N is the number of inputs, K is the number of outputs and L is the constraint length which indicates the number of memory elements used. The convolutional encoder rate is N/K. The different polynomial generator can be done using different memory element connection. The code rate can be controlled through a puncturing technique to downgrade the rate. The example shown in figure 3.2 has parameters N = 1, K = 2 and L = 3. The rate in this example is equal to 1/2 and the polynomial generators are G_0= 1, 0, 1 and G_1= 1, 1, 0.

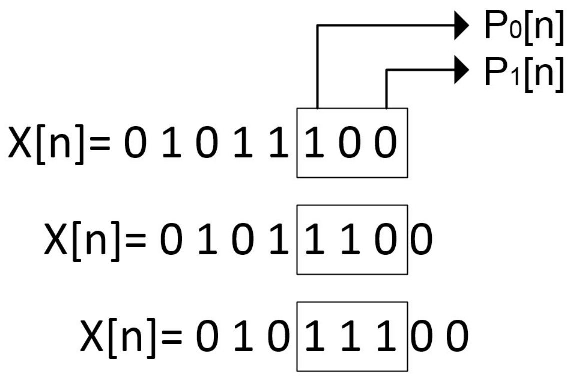


Figure 3.1: Convolutional encoder window

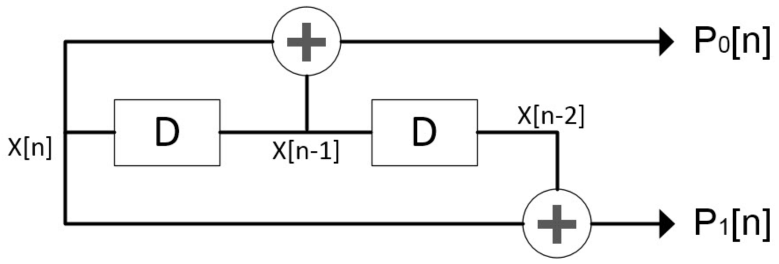


Figure 3.2: Convolutional encoder example

The output of the convolutional encoder corresponding to the values in 3.1 will be:

$P_0[0] = (1 \text{ xor } 0) = 1$, $P_1[0] = (1 \text{ xor } 0) = 1$

$P_0[1] = (1 \text{ xor } 1) = 0$, $P_1[1] = (1 \text{ xor } 0) = 1$

$P_0[2] = (1 \text{ xor } 1) = 0$, $P_1[2] = (1 \text{ xor } 1) = 0$

...

and so on.

3.1.2 2G convolutional encoder

The 2G is the second generation of the mobile communication. It is based on the Global System for Mobile Communications (GSM) developed the European Telecommunications Standards Institute (ETSI). The convolutional encoder used in different channels of the 2G standard [31].

3.1.3 3G convolutional encoder

The 3G is a short term for the third generation of the mobile wireless that provides a transfer data rate starting from 200 Kbit/s. The 3G standard is released by the 3rd Generation Partnership Project (3GPP) community with common names like UMTS and

WCDMA. The 3G uses the convolutional encoders in the channel coding in of the different channels [32]. Figure 3.3 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 zeros and also the final state should return back to zeros. The proposed DPR design shown in this chapter shows that the switching between different modes of operation for the 3G can be done using the DPR technique. Which means that the hardware optimization can be done to the same standard.

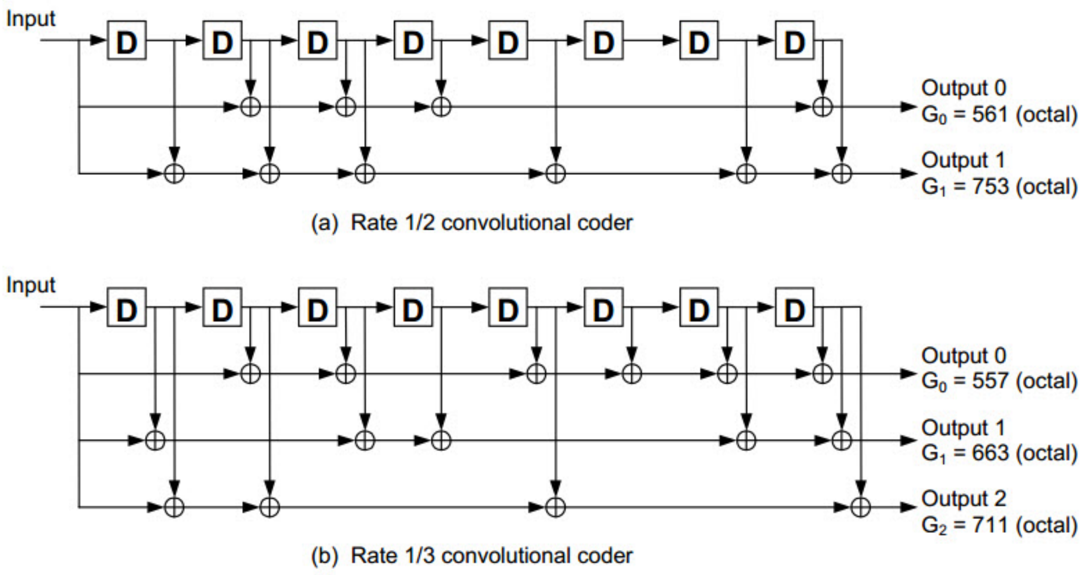
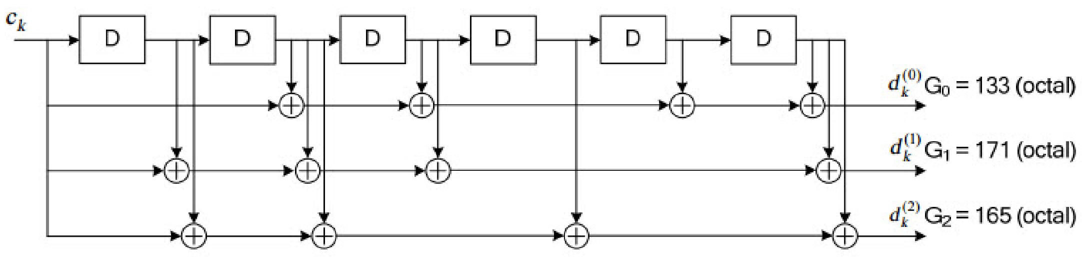


Figure 3.3: Convolutional encoder used in 3G [32]

3.1.4 4G convolutional encoder

4G is the latest deployed communication standard, which allows higher data rates up to 1GBit/sec. The 4G is released by the 3GPP community and widely known as Long Term Evolution (LTE). Figures 3.4 shows the convolutional encoder used in the LTE [33], and figure 3.5 shows the turbo encoder that is based on the convolutional encoder [33]. In the 4G standard, the tail biting is used where the initial and final states of the shift registers should be the same.



c_k
D
$d_k^{(0)}$ G_0 = 133 (octal)
$d_k^{(1)}$ G_1 = 171 (octal)
$d_k^{(2)}$ G_2 = 165 (octal)

Figure 3.4: Convolutional encoder used in LTE [33]

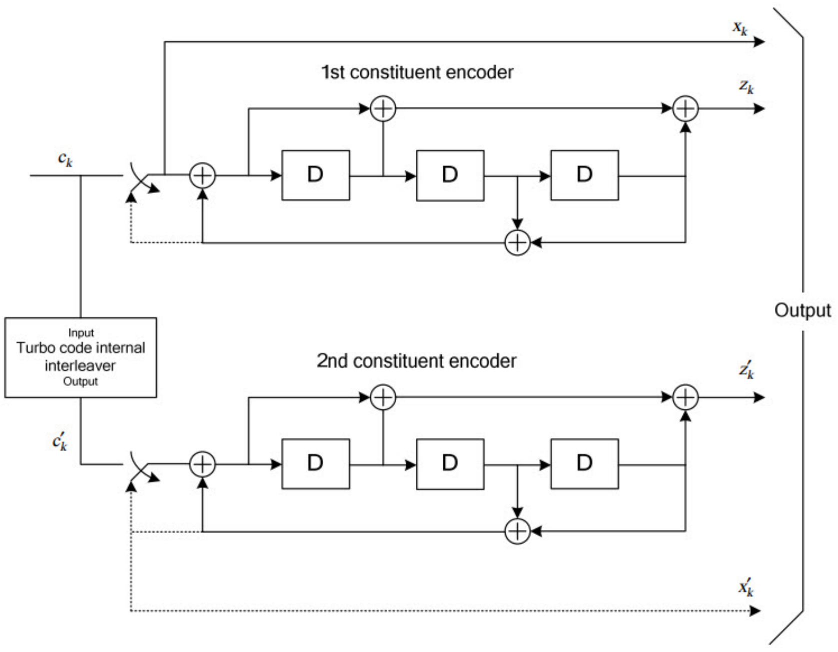


Figure 3.5: Turbo encoder used in LTE [33]

3.1.5 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 suit. IEEE 802.11 standard defines an air interface between a user terminal and a base station or among many devices. 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 3.6 is used in 802.11 a/g standards [34]. It will be implemented in the thesis demonstration for the SDR system using DPR.

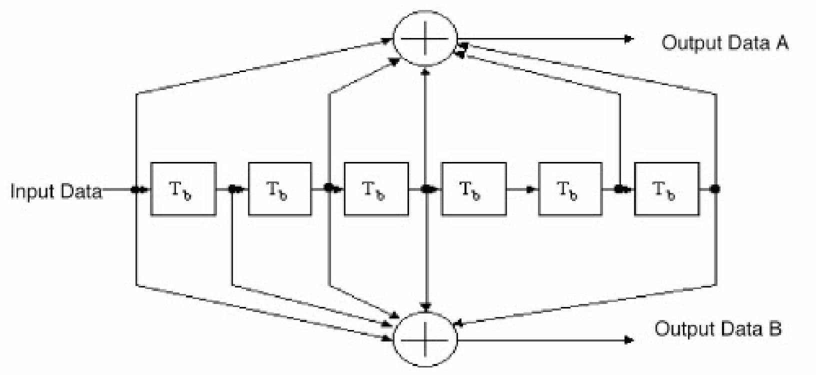


Figure 3.6: Convolutional encoder used in WIFI 802.11 a/g [34]

3.2 Convolutional encoder summary

Convolutional encoders are widely used in the wireless communication systems, such as 2G, 3G, LTE and WIFI. Table 3.1 shows the characteristics of the different convolutional encoders used in 2G, 3G, LTE and WIFI technologies. Different convolutional encoders reflect different hardware combinations of the memory elements. The differences in the hardware mean different realization that consumes area if each channel is fully implemented alone. From this point of view, the thesis serves in combining these hardware channels in the best way using DPR technique. Using the same concept will be applied to all modules may have the same characteristics and serve in the same position in the communication chain as will be discussed in chapter 4.

Table 3.1: Convolutional encoders used in different communications systems

Convolutional Encoders	System	Channel	Rate	Constraint Length	Generator polynomials (Octal)
C1	2G	TCH/FR Speech	1/2	5	G0 = 31 G1 = 33
C2	2G	TCH/HR Speech	1/3	7	G4 = 155 G5 = 123 G6 = 137
C3	2G	Data	1/3	5	G1 = 33 G2 = 25 G3 = 37
C4	3G	BCH, PCH, RACH, DCH, FACH	1/2	9	G0 = 561 G1 = 753
C5	3G	DCH, FACH	1/3	9	G0 = 557 G1 = 663 G2 = 711
C6	LTE	BCH, DCI, UCI	1/3	7	G0 = 133 G1 = 171 G2 = 165
C7	WIFI 802.11 a,g	OFDM channel	1/2	7	G0 = 133 G1 = 171

3.3 Lab setup

The Lab setup is shown in Figure 3.7 consists of:

PC: Is used for communicating with the evaluation kit, where a simple software interface is implemented to control the FPGA kit functional operation. The software is used for entering a number to be decoded and chose among different encoders as will be discussed later. The PC connected to the evaluation kit through a serial cable.

FPGA kit: The XUPV5-LX110T kit has Xilinx Virtex-5-XC5VLX110T FPGA, more details about this FPGA is discussed in chapter 2. The bitstreams of the different encoders and different designs are deployed on this FPGA. Softcore processor MicroBlaze is implemented on the FPGA fabric to control the loading of the bitstreams from the Compact Flash to configure and reconfigure the logic gates.

Compact Flash (CF): Is attached to the kit, which stores the GEM bitstream and the SLEM partial and static bitstreams.

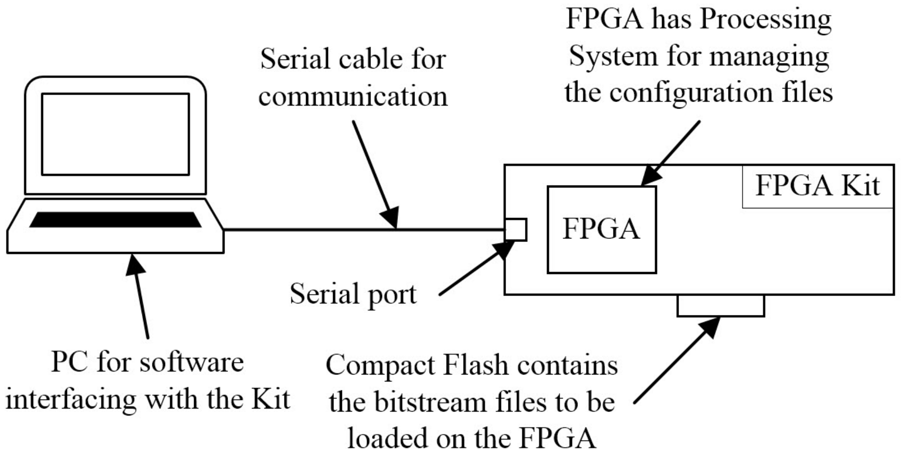

Serial cable for communication
FPGA has Processing System for managing the configuration files
FPGA Kit
FPGA
Serial port
PC for software interfacing with the Kit
Compact Flash contains the bitstream files to be loaded on the FPGA

Figure 3.7: Convolutional encoder lab setup

3.4 General Encoder Module (GEM)

The General Encoder Module (GEM) shown in figure 3.8, is a design where all the convolutional encoders are implemented and exist on the FPGA. The GEM design is formed of one part which is static, i.e. no hardware connection or realization changes in the FPGA during the run-time. A multiplexer is used to switch among the different encoders which are preloaded in the GEM.bit module. The VHDL code is implemented in one file that goes through the normal digital design process described in 2. This design is formed from two parts the processing system and the convolutional module. The processing system is softcore processing system implemented on the FPGA programmable logic. It contains softcore MicroBlaze processor, UART, ICAP and System ACE. The softcore MicroBlaze processor is used for managing the software interface with the PC and manages the values sent to the multiplexer for proper switching among the encoders. The UART IP used to allow the PC communication with the Kit through the serial port. ICAP is IP module used to configure the internal FPGA programmable logic from the internal softcore MicroBlaze processor. System ACE module is used to load the bit files to the FPGA from the Compact Flash.

On turning on the system, the System ACE load the GEM.bit file from the Compact Flash and the ICAP access the reconfiguration memory to program the logic gates. Then a serial communication between the PC and the kit is initiated on resting the kit (press the reset key). A terminal emulator “Tera Term” software on the PC is used, that allows the user to choose among the different ready implemented convolutional encoders. Then the user to insert the number to be encoded and the encoded number return back to the display.

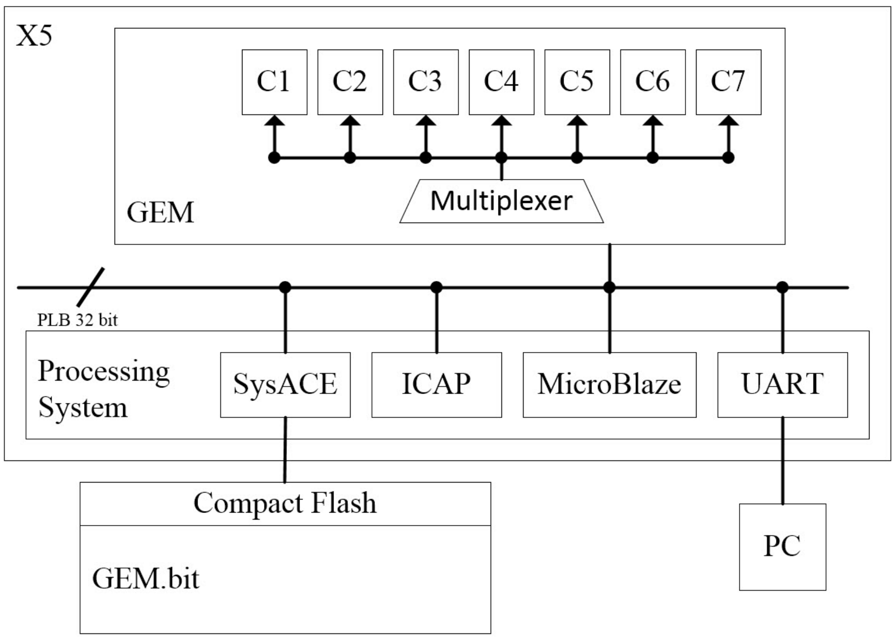


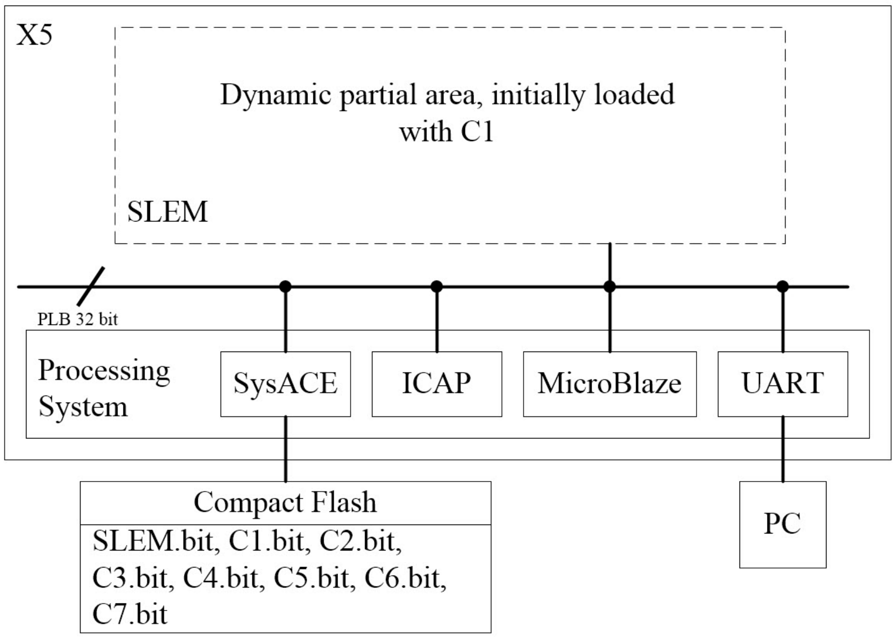
Figure 3.8: GEM design, switching is done using multiplexer

3.5 Single-Loaded Encoder Module (SLEM)

The second design using the PDR technique is named as Single-Loaded Encoder Module (SLEM). Where the switching is done real time among the images stored on the Compact Flash and loaded to the FPGA programmable logic at run-time. In this design, the FPGA is partitioned into a static region and Reconfigurable Partition (RP) which is a dynamic region that can be changed during FPGA run-time. Each encoder is implemented in a single file and is used as a Reconfigurable Module (RM). Each RM has its own netlist that is loaded in the RP partition real time.

In the SLEM implementation that is shown in figure 3.9, the static region contains the processing system which contains softcore MicroBlaze processor, UART, ICAP and System ACE. And a part of the FPGA is programmed to be dynamically reconfigurable using the PlanAhead tool. The MicroBlaze in this design is used for software interface with the PC and is responsible for loading the partial bitstream file to the dynamic reconfigurable part. The MicroBlaze is to communicate with the Compact Flash through the System ACE and load the configuration partial bit stream files (C1.bit – C7.bit) to the RP region through the ICAP. The partial bitstreams files are stored initially on a Compact Flash and loaded on demand and the only chosen encoder will be loaded to the FPGA. The PC is connected to the FPGA kit through a serial port.

On turning on the system the System ACE load the SLEM.bit file from the Compact Flash and the ICAP access the reconfiguration memory to program the logic gates. The SLEM file contains the static design and an initially loaded encoder for the reconfigurable region. Then a serial communication between the PC and the kit is initiated on resting the kit (press the reset key). A terminal emulator “Tera Term” software on the PC is used, that allows the user to choose among the different implemented convolutional encoders



X5
Dynamic partial area, initially loaded with C1
SLEM
PLB 32 bit
Processing System
SysACE
ICAP
MicroBlaze
UART
Compact Flash
SLEM.bit, C1.bit, C2.bit, C3.bit, C4.bit, C5.bit, C6.bit, C7.bit
PC

Figure 3.9: SLEM design, encoder loaded during FPGA run-time

that will be loaded on demand. Then the user to insert the number to be encoded and the encoded number return back to the display. The user will not feel any difference in the user interface.

3.6 Results for the two systems

The trade-off between the two designs is shown in the following points, where SLEM shows a reduction in area and power consumed while GEM has a small benefit in time and memory. The floorplan of the GEM and the SLEM designs is shown in figure 3.10.

3.6.1 Area occupied on the FPGA

In each design a partition of 800 LUTs is assigned to the implementation of SLEM encoder and GEM encoders, using Xilinx PlanAhead tool, so it can easily clarify the utilization percentage of each design in these 800 LUTs. Table 3.2 shows the number of LUTs used for both designs and their utilization percentage. In SLEM design, the maximum area consumed per convolutional encoder is 38 LUT. While the consumed area for the GEM, where all the convolutional encoders exist, is 258 LUTs without any optimization. After optimization, the GEM size became 113 LUTs which is even more than the SLEM design. On using DPR in SLEM design the encoders area improved by a factor of 67% compared to the full implementation of GEM optimized design. Equation 3.1 shows how the improvement factor is calculated.

$$Improvement\ in\ area = 1 - \frac{SLEM\ number\ of\ LUTs}{GEM_{optimized}\ number\ of\ LUTs} \times 100\% \tag{3.1}$$

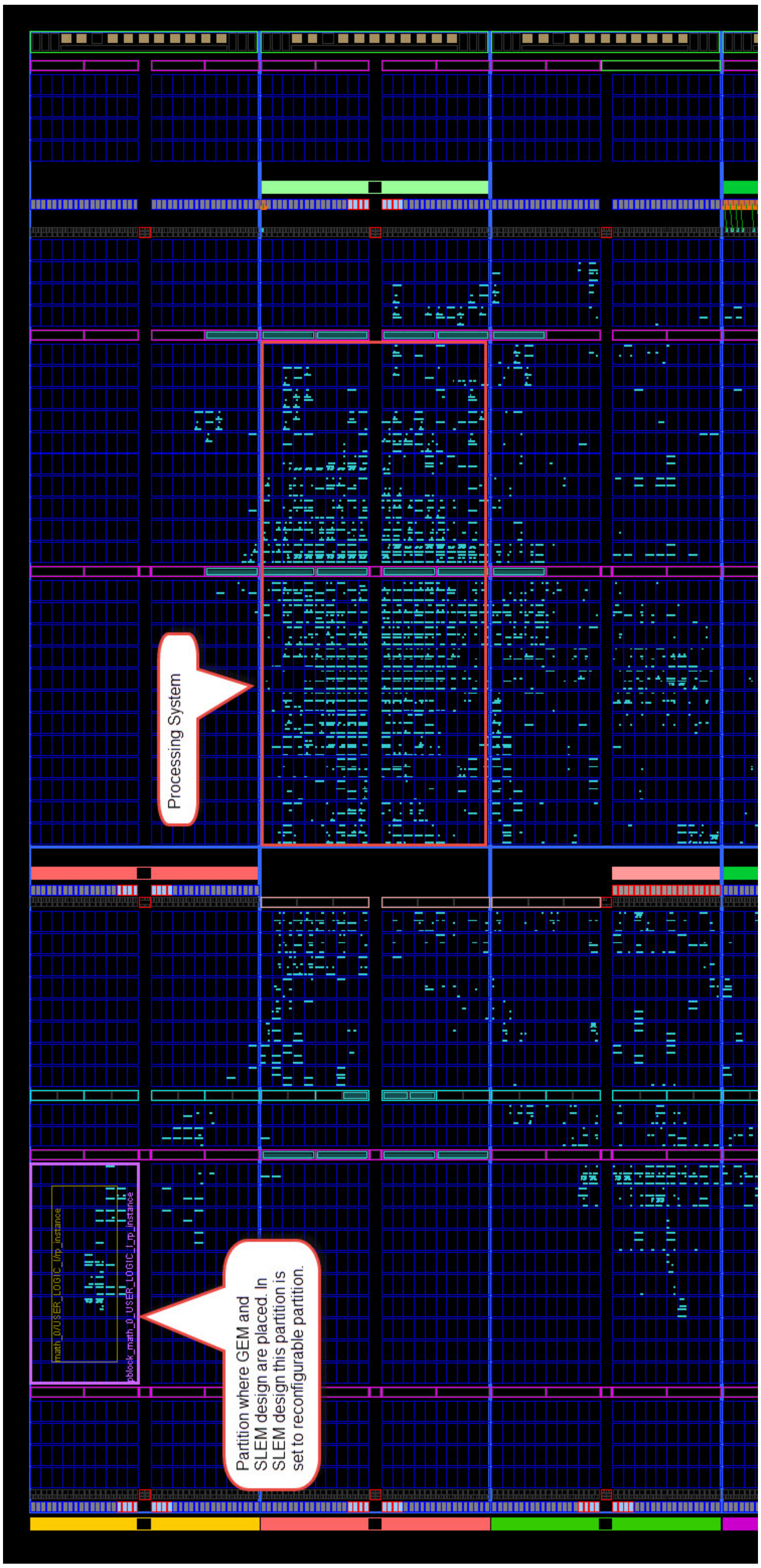

Processing System
Partition where GEM and SLEM design are placed. In SLEM design this partition is set to reconfigurable partition.
math_0/USER_LOGIC_I/rp_instance
pblock_math_0_USER_LOGIC_I_rp_instance

Figure 3.10: GEM and SLEM floorplan

Table 3.2: Area utilization for SLEM and GEM

Encoders	SLEM							GEM	GEM
	C1	C2	C3	C4	C5	C6	C7	No Optimization	Optimized
No. of LUTs	36	37	37	37	38	37	36	258	113
Utilization (%)	4.5	4.6	4.6	4.6	4.8	4.6	4.5	32.3	14.1

3.6.2 Memory needed

The initial file size of the GEM and SLEM designs are nearly the same size 3.8 MB. The GEM design configuration file contains the Processing System (MicroBlaze, ICAP, UART, SySACE) and the encoders. The SLEM initial configuration file contains the Processing System (MicroBlaze, ICAP, UART, SySACE) and one encoder. As the Processing System acquires larger area than the encoder part the few LUTs difference between GEM configuration and SLEM initial configuration is ignored. The SLEM design needs more memory storage than the GEM design because of the partial bitstream files of the individual convolutional encoders each of size 60 KB. Hence, for the designed 7 convolutional encoders SLEM needs 420 KB more memory, which is considered a slight increase in memory. Note that the generated file size depends on the number of LUTs, so the formed 60KB file size depends on the number of LUTs in the dynamic partition, not the implemented design, i.e. the 60KB file size is equivalent to the 800 LUTs not the 38 LUTs of the maximum encoder size in SLEM. Table 3.3 summarizing the memory needed for both designs.

Table 3.3: Memory needed for SLEM and GEM

SLEM	GEM
Initial configuration bitstream File Size = 3.8 MB Reconfigurable bitstream file sizes = 7 x 60 KB = 420 KB	Configuration bitstream File Size = 3.8 MB
Total Size = 4.2 MB	Total Size = 3.8 MB

3.6.3 Power estimation

Xilinx Power Analyzer tool (XPA) is used to estimate the power consumption in the convolutional encoder module. The SLEM design shows improvement in the module logic power consumption over GEM design for different MicroBlaze operating frequencies, as shown in table 3.4. The power improvement percent is calculated according to equation 3.2. GEM design occupies more LUTs that consumes more logic power. While SLEM design, only the needed encoder will be loaded, this consumes less power and less number of LUTs compared to the full implementation. Whereas increasing the frequency of the MicroBlaze from 50 MHz to 100 MHz increases the consumed power in both designs, but the SLEM design shows less power consumption than GEM design for all applied frequencies. The power improvement percent is between from 56% and 64% for different MicroBlaze frequencies of operation. On continue increasing the frequency of operation, it is indicated that the power improvement starts to decrease once again.

Which shows that the optimized value for the SoC design using MicroBlaze is around 75 MHz.

$$Improvement\ in\ power = 1 - \frac{Power\ consumed\ in\ SLEM}{Power\ consumed\ in\ GEM_{optimized}} \times 100\% \tag{3.2}$$

Table 3.4: Power consumption

Operating frequency	SLEM power consumption	GEM power consumption	Power improvement
50 MHz	0.08 mW	0.18 mW	56 %
75 MHz	0.12 mW	0.33 mW	64 %
100 MHz	0.17 mW	0.44 mW	61 %

3.6.4 Time overhead

The initial configuration time in both designs GEM and SLEM are the same because the initial bitstream file has the same size 3.8 MB for both, table 3.3. In the GEM, design there is no time overhead added because all encoders exist while in the SLEM design partial bitstream files of size 60 KB is generated for each encoder, consequently a reconfiguration time overhead is added in the SLEM design. Table 3.5 shows the configuration and reconfiguration time for both designs, and how they are decreased by increasing the MicroBlaze frequency. Equation 3.3 shows the theoretical configuration and reconfiguration time used in Table 3.5. The maximum throughput is obtained for ICAP from table 2.1.

$$Theoretical\ configuration\ time = \frac{Bitstream\ file\ size}{Max\ throughput} \tag{3.3}$$

Table 3.5: Configuration and reconfiguration time

Design	SLEM (C1 – C7)	GEM
Operating frequency 50 MHz		
Configuration time (ms)	19	19
Reconfiguration time (ms)	0.30	0
Operating frequency 75 MHz		
Configuration time (ms)	12.7	12.7
Reconfiguration time (ms)	0.20	0
Operating frequency 100 MHz		
Configuration time (ms)	9.5	9.5
Reconfiguration time (ms)	0.15	0

3.6.5 PTF triangle

Power, Time and Frequency triangle (PTF) is one of the important factors in choosing compromised designs. Figure 3.11 shows the trade-off between the power consumption and the reconfiguration time for a partial bit stream of file size 60 KB. It shows that on operating MicroBlaze with a frequency of 100 MHz decreases the reconfiguration time, and the module logic power consumption increases. While decreasing the MicroBlaze frequency to 50 MHz increases the reconfiguration time and decreases the power consumed in the module logic. The optimized value of MicroBlaze frequency of operation is around 75 MHz, where the power consumed is between 0.1 mw and 0.12 mw. Also, the reconfiguration time needed for reconfiguring the Partial Blocks is between 0.2 msec and 0.25 msec.

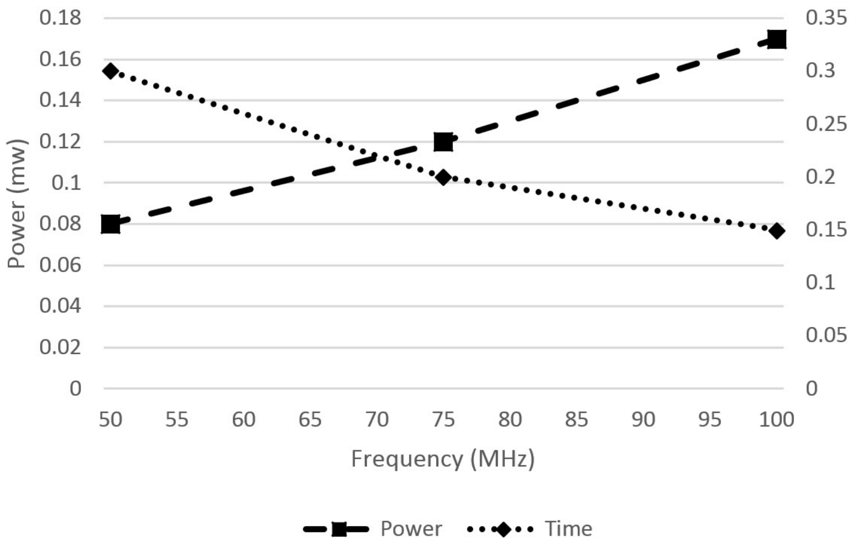


Figure 3.11: PTF triangle shows the trade-off between the power consumption in module logic and the reconfiguration time needed to download new module using DPR in SLEM design for different MicroBlaze operating frequencies.

3.7 Conclusion

This chapter shows the benefits of using the DPR technique in SDR system to implement a single block. Implementing a library of different encoders and switching among them reduces the system complexity and makes it handy and real time upgradable. Using DPR technique saves more power and area on silicon with high percentages with a slight increase in time overhead and memory storage. The next chapter will include simplifying the communication system chain and generalizing the concept of DPR for other blocks.

Chapter 4

SDR chain implementation

This chapter proposes an implementation for an SDR chain for a multi-standard communication system using DPR technique. In this chapter, the different standards are simplified to operate using the preliminary blocks of each communication standard. A comparison is done between simple DPR and normal systems. A different way of DPR using external controller PC is presented in this chapter. The configuration and the reconfiguration are done through JTAG.

4.1 Communication standards similarities

The simple communication channel is formed of channel coding that is used to overcome channel circumstances to retrieve the sent data with accepted accuracy. The second main block is the modulation where a carrier signal adapted in a special way to carry the data signal over the medium to reach its destination. The different communication standards use these main blocks with other blocks depend on specifications of the standard like reaching a high capacity of users with high data rate transfer overcoming the channel circumstances. A simplified communication standards are presented in figure 4.1 for the main blocks in the 2G, 3G, 4G and WIFI standards. Where the presented blocks had different specs according to the mode of operation and channel used for each standard.

The 2G is the wireless mobile second generation standard, it is originally based on the Global System for Mobile Communication (GSM). In the 2G, the physical layer mainly composed of channel coding using convolutional encoder then modulated with Gaussian minimum-shift keying (GMSK) [35, 36, 31, 37].

The 3G is the wireless mobile third generation standard, it uses Wideband Code Division Multiple Access (WCDMA) radio access technology to offer greater spectral efficiency and bandwidth than predecessor 2G. In 3G the data is encoded using FEC convolutional encoders or Turbo encoders, then it is spread on the channel bandwidth affording better channel resistance to noise and interference, and then passes through a different phase and amplitude modulation techniques like Binary Phase Shift Keying (BPSK) and Quadrature Amplitude Modulation (QAM) [38, 39, 32, 40].

WIFI is a fixed wireless communication standard. It allows the electronic devices to exchange data or connect wirelessly to the internet. The WIFI physical layer uses Orthogonal Frequency Division Multiplexing (OFDM) as a modulation format. OFDM is a modulation scheme where orthogonal multi-carriers closely spaced are used to carry data. Each carrier is modulated using conventional modulation techniques such as BPSK, QAM, 16-QAM, etc. The Inverse Fast Fourier Transform (IFFT) in the transmitter side is adopted for the carrier orthogonality purpose. Channel coding in WIFI is done using convolutional encoders or Low Density Parity Check code (LDPC) [34].

Long Term Evolution (LTE) or 4G is the fourth generation in wireless mobile standards. In the LTE uplink transmitter, Single Carrier Frequency Division Multiple Accesses (SC-FDMA) is used for its lower peak-to-average power ratio (PAPR) which in consequence benefits in mobile terminal transmit power efficiency and the power amplifier cost reduction. SC-FDMA is like the Orthogonal Frequency Division Multiple Accesses (OFDMA) with Discrete Fourier Transform (DFT) step is added before layer mapping and N-IFFT, the M-point DFT is selected such that M to be less than N. LTE supports different N-point IFFT sizes for different frequencies [41] and different conventional modulation techniques. LTE uses FEC convolutional encoders and Turbo encoders for channel coding [41, 42, 43, 33, 44].

In 2G and 3G the main blocks are channel coding and modulation. While in the LTE uplink transmitter uses the SC-FDMA channel has 2 different main components DFT and IFFT, and in the WIFI depends on the OFDM channel that has an IFFT block. In Wifi and LTE, a mapper is used for mapping frequency points onto the data subcarriers for the OFDM symbol that is assigned to a user. Also, a cyclic prefix (guard insertion) is added to eliminate the Inter Symbol Interference (ISI) between symbols. The puncturing and interleaving stage in different communication standards can be considered as a part of the channel coding.

4.2 Implemented design

The implemented design proposed in the SDR system is switching among 3G, LTE and WIFI, it is shown in figure 4.2. A selected blocks from 4.1 are the channel coding, modulation, DFT and IFFT. The chosen channel coding for the different systems is the convolutional encoder. For the 3G two different implementations selected, the first with (rate 1/2, constraint length 9) and the second with (rate 1/3, constraint length 9), the LTE convolutional encoder to be implemented with (rate 1/3, length 7) and the WIFI convolutional encoder with (rate 1/2, length 7). The different standards support either BPSK, QPSK or 16-QAM as a conventional digital modulation. In the 3G where it uses the WCDMA to spread data over the channel there is no need for DFT and IFFT blocks, so filler is added. The filler is an empty block added to the chain when no specific block is added. In the LTE User Equipment (UE) transmitter side an M-Point DFT and N-Point IFFT are used as discussed in the previous session, with selected values for M and N are 64 and 256 respectively. In the WIFI chain, a 64-point IFFT used while DFT block is replaced with a filler.

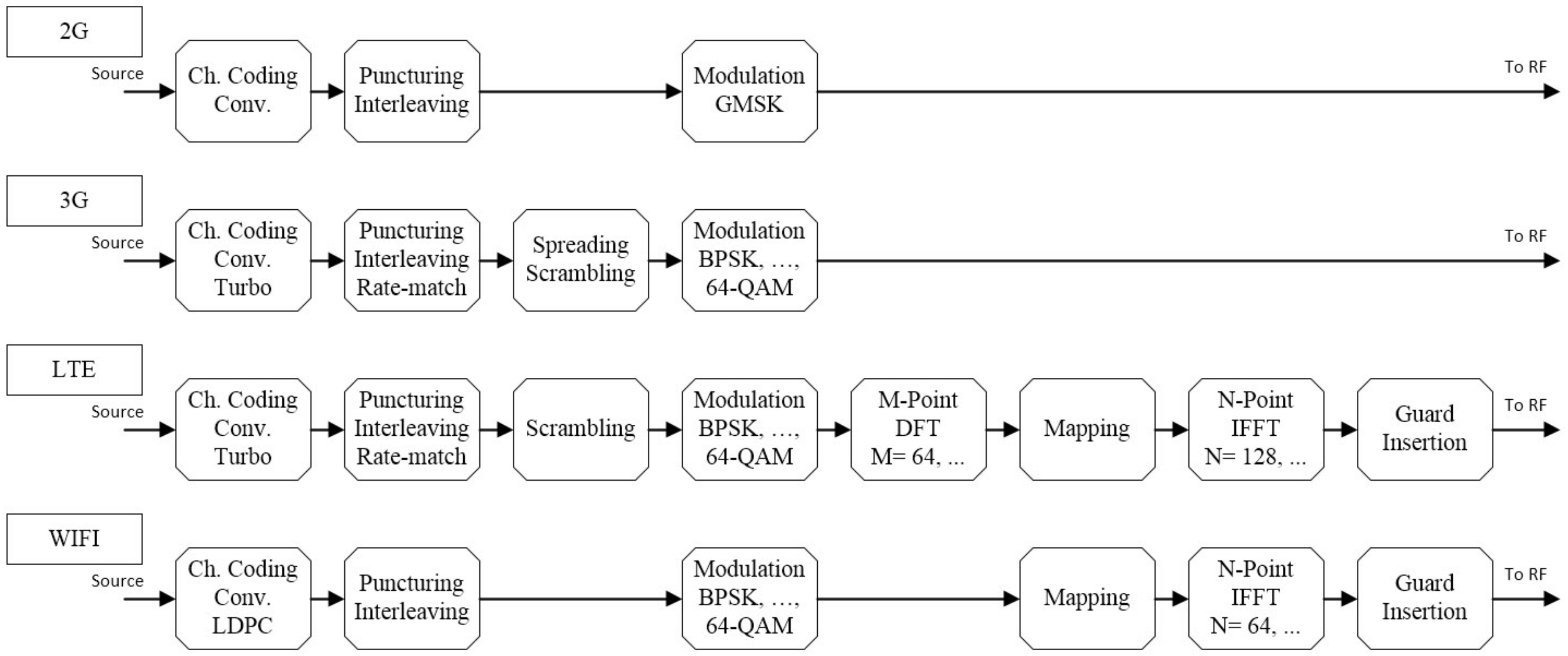


Figure 4.1: Different communication chains

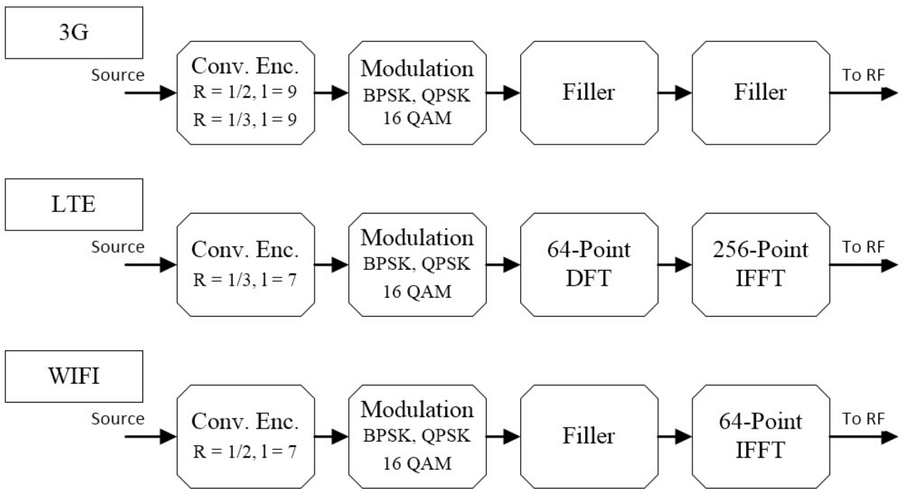


Figure 4.2: 3G,4G and WIFI communication chain

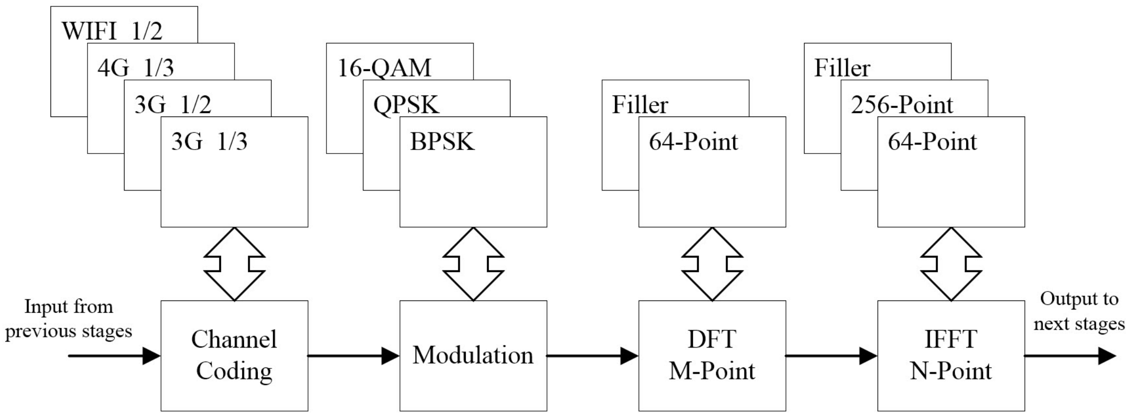


Figure 4.3: Reconfigurable SDR chain

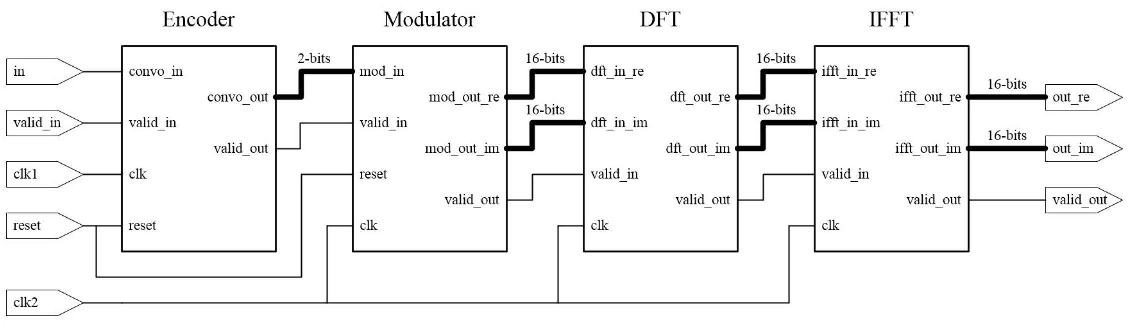


Figure 4.4: Schematic of the SDR chain

The reconfigurable chain implemented on the Xilinx FPGA is shown in figure 4.3. It is composed of 4 successive blocks corresponding to that shown in figure 4.2. The chain is composed of channel coding, modulation, DFT and IFFT. The schematic of the design is shown in figure 4.4 and the floor planned design from PlanAhead tool is shown in figure 4.5.

4.3 General system

The communication system generally introduced in chapter 1. A more detailed system is presented showing more focus on internal system construction in figure 4.6. The system consists of different blocks some of them are implemented on hardware layer and other to be implemented on software layer. These blocks are:

1. Analog domain where the signal is received and converted using ADC / DAC to/from Digital domain.
2. The Baseband processing held on FPGA where hardware modules to be reconfigured for a certain channel.
3. Network Layer for packets reception and processing. It is implemented as software modules.

The processing system takes care of:

1. Software modules data processing.
2. FPGA reconfiguration for hardware modules.
3. Memory management and passing data between different system blocks hardware and software.

The memory device contains:

1. Firmware for the connected devices.
2. The operating system files which contains related applications.
3. Reconfigurable hardware modules.

The Research focus on the yellow parts that include the transmitter chain, FPGA reconfiguration and Reconfigurable modules.

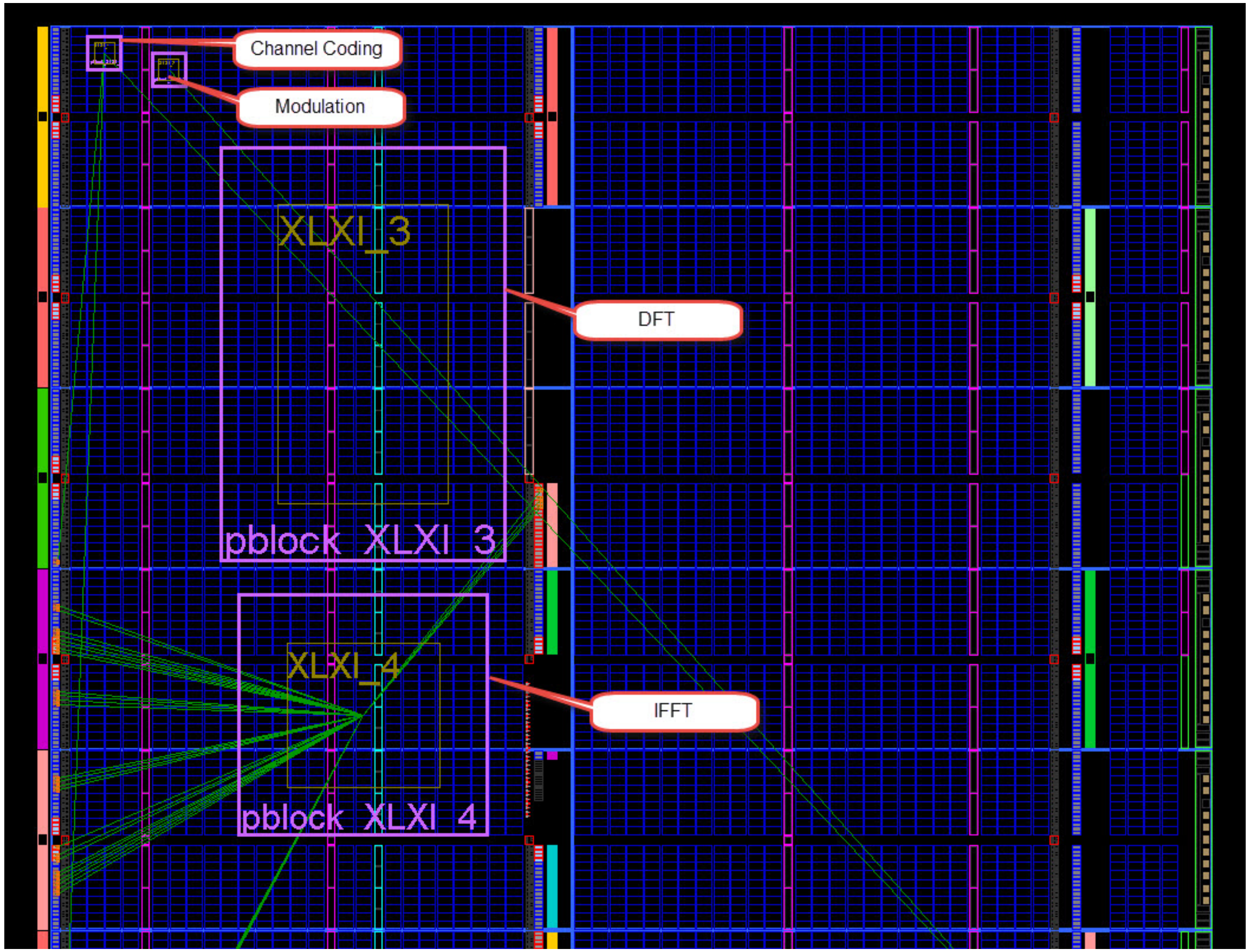


Figure 4.5: SDR chain floorplanning using PlanAhead

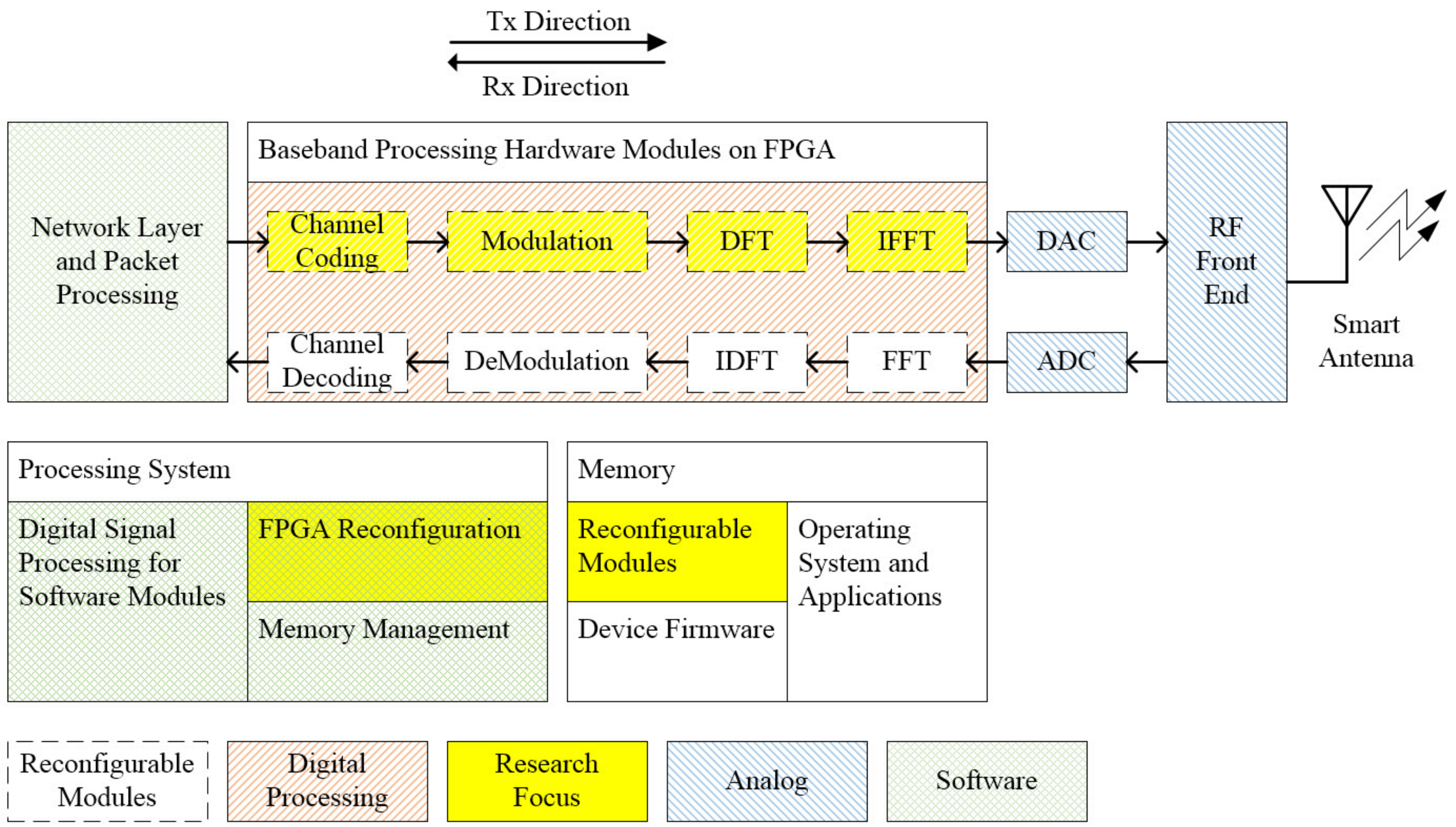


Figure 4.6: Full system

4.4 Lab setup

The Lab setup is shown in Figure 4.7 consists of:

PC: Is the responsible for managing the reconfiguration bitstream files that are stored on its Hard Disk Drive (HDD). In this connection mode setup, the reconfiguration is done through the Jtag.

FPGA kit: The kit has the Virtex-5-XC5VLX110T FPGA, more details about this FPGA is discussed in 2. The VHDL codes for the different communication blocks are deployed on this FPGA.

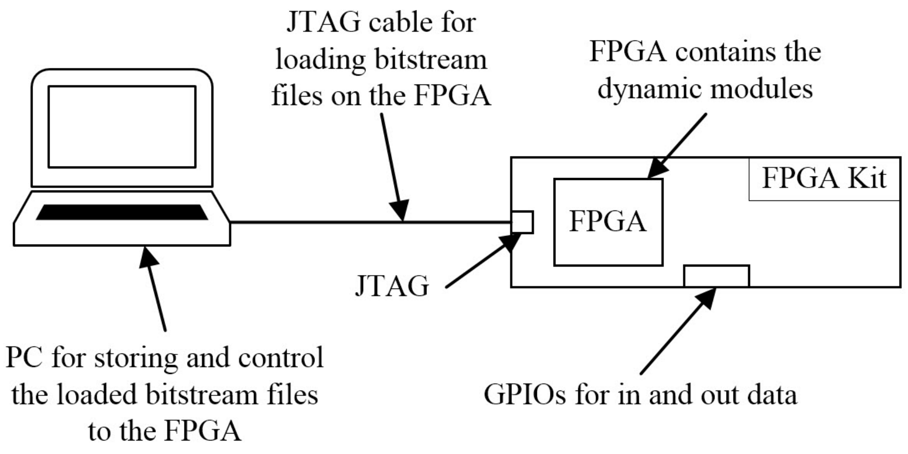

JTAG cable for loading bitstream files on the FPGA
FPGA contains the dynamic modules
FPGA Kit
FPGA
JTAG
PC for storing and control the loaded bitstream files to the FPGA
GPIOs for in and out data

Figure 4.7: SDR chain lab setup

4.5 System results

The current section presents the results of the SDR chain using DPR denoted as Single-Loaded Module (SLM), in terms of area, memory, estimated power ccnsumption and time overhead compared to General Communication System Module (GCSM), where three simple communication standards exist at same time. In SLM design, there are 12 test cases can cover the implemented design shown in figures 4.2 and 4.3 for the different combinations per communication standard listed in table 4.1. Three design runs are chosen to cover a mode of operation in the different communication standards. The first design run is for the 3G communication standard with a convolutional encoder of rate half, BPSK modulation. The second design run is for the WIFI with a convolutional encoder of rate half, QPSK modulation and 64-Point IFFT. The third design run is for the LTE with a convolutional encoder of rate third, 16-QAM modulation, 64-Point DFT and 256-Point IFFT. The design runs are shown in table 4.2. The floorplan of the three design runs DR_1, DR_2 and DR_3 are shown in figures 4.8, 4.9 and 4.10 respectively. In the GCSM, the chosen design runs in the SLM table 4.2 are implemented for simplicity while in the real world each communication standard should be implemented with all its blocks. The GSCM design and its floorplan are shown in figures 4.11 and 4.12 respectively.

Table 4.1: SLM design runs list

Design Run	Standard	Encoder	Modulation	DFT	IFFT
DR_1	3G	3G_half	BPSK	Filler	Filler
	3G	3G_half	QPSK	Filler	Filler
	3G	3G_half	16-QAM	Filler	Filler
	3G	3G_third	BPSK	Filler	Filler
	3G	3G_third	QPSK	Filler	Filler
	3G	3G_third	16-QAM	Filler	Filler
	WIFI	WIFI_half	BPSK	Filler	64-IFFT
DR_2	WIFI	WIFI_half	QPSK	Filler	64-IFFT
	WIFI	WIFI_half	16-QAM	Filler	64-IFFT
	LTE	LTE_third	BPSK	64-DFT	256-IFFT
	LTE	LTE_third	QPSK	64-DFT	256-IFFT
DR_3	LTE	LTE_third	16-QAM	64-DFT	256-IFFT

Table 4.2: SLM selected design runs

Design Run	Standard	Encoder	Modulation	DFT	IFFT
DR_1	3G	3G_half	BPSK	Filler	Filler
DR_2	WIFI	WIFI_half	QPSK	Filler	64-IFFT
DR_3	LTE	LTE_third	16-QAM	64-DFT	256-IFFT

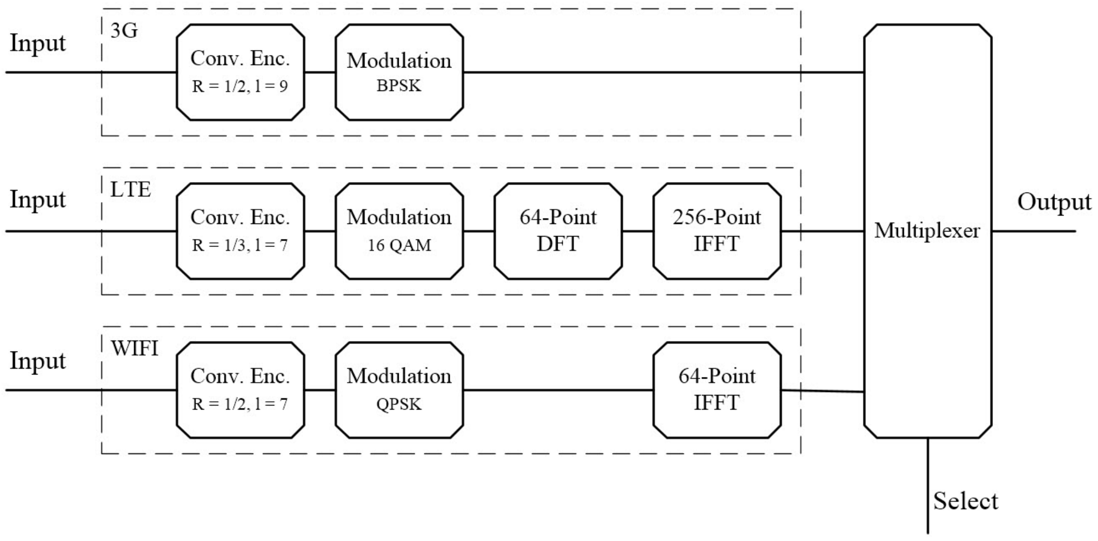


Figure 4.11: GSCM design

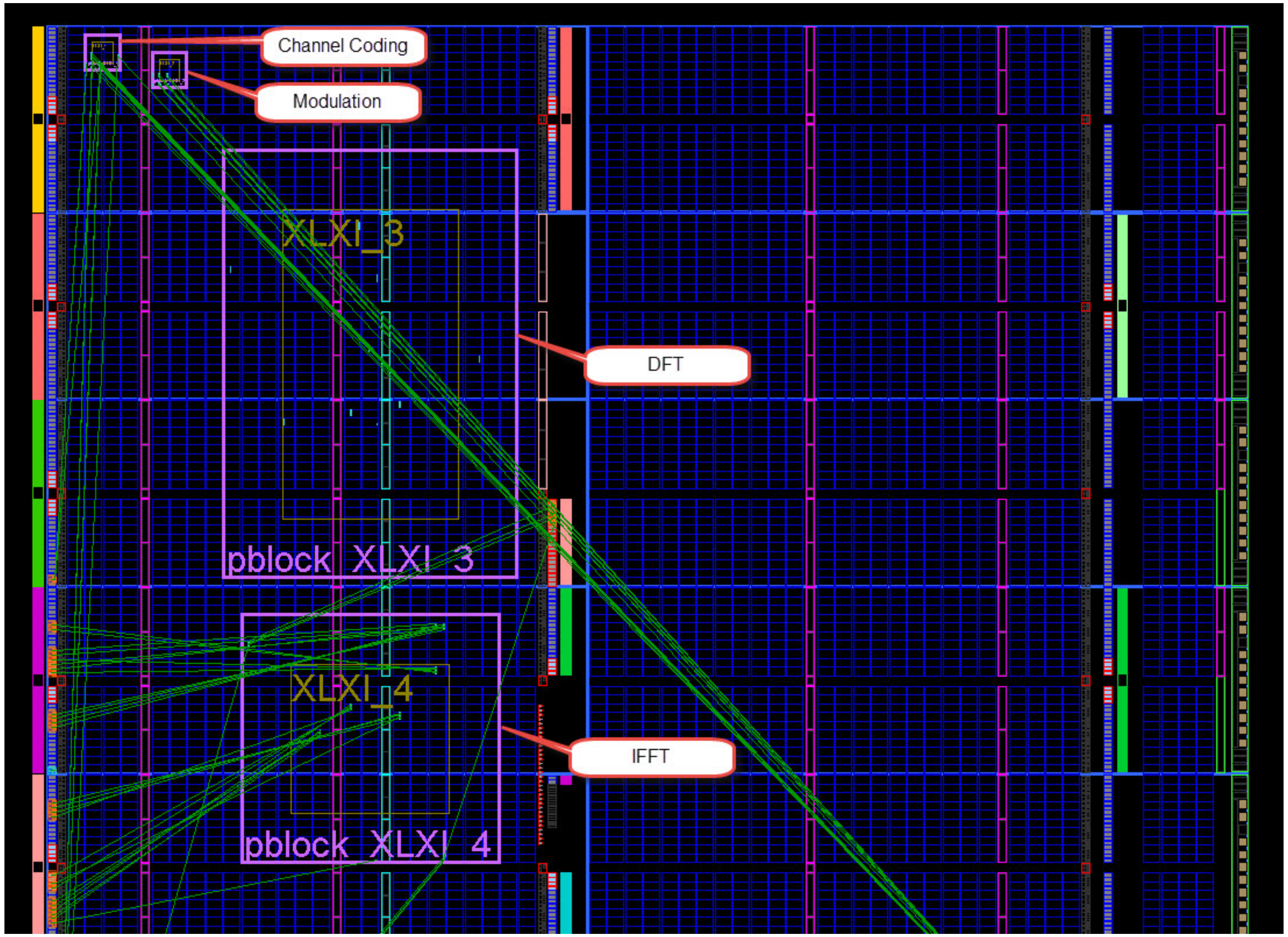


Figure 4.8: DR_1 for 3G loaded modules in the SDR chain

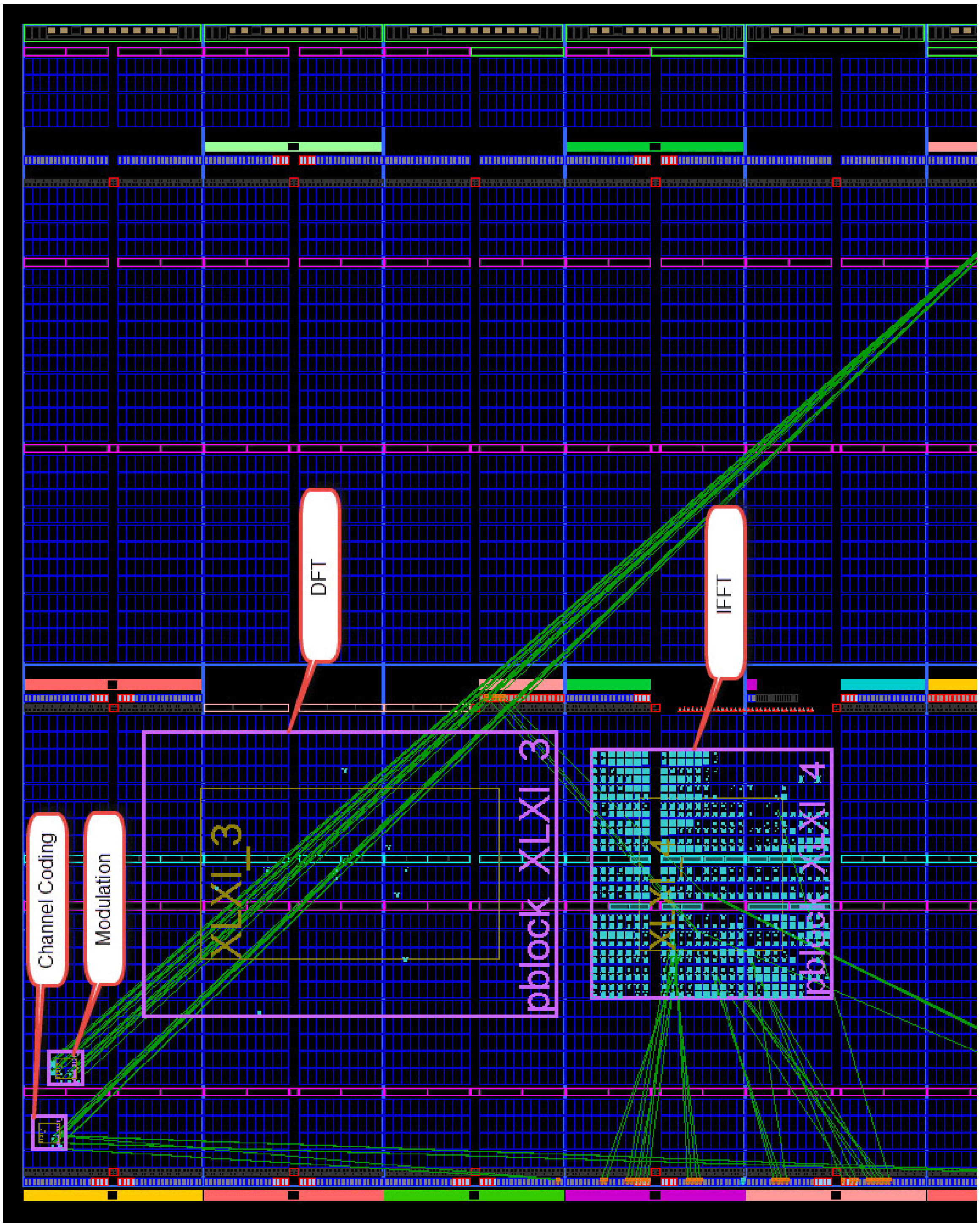


Figure 4.9: DR_2 for WIFI loaded modules in the SDR chain

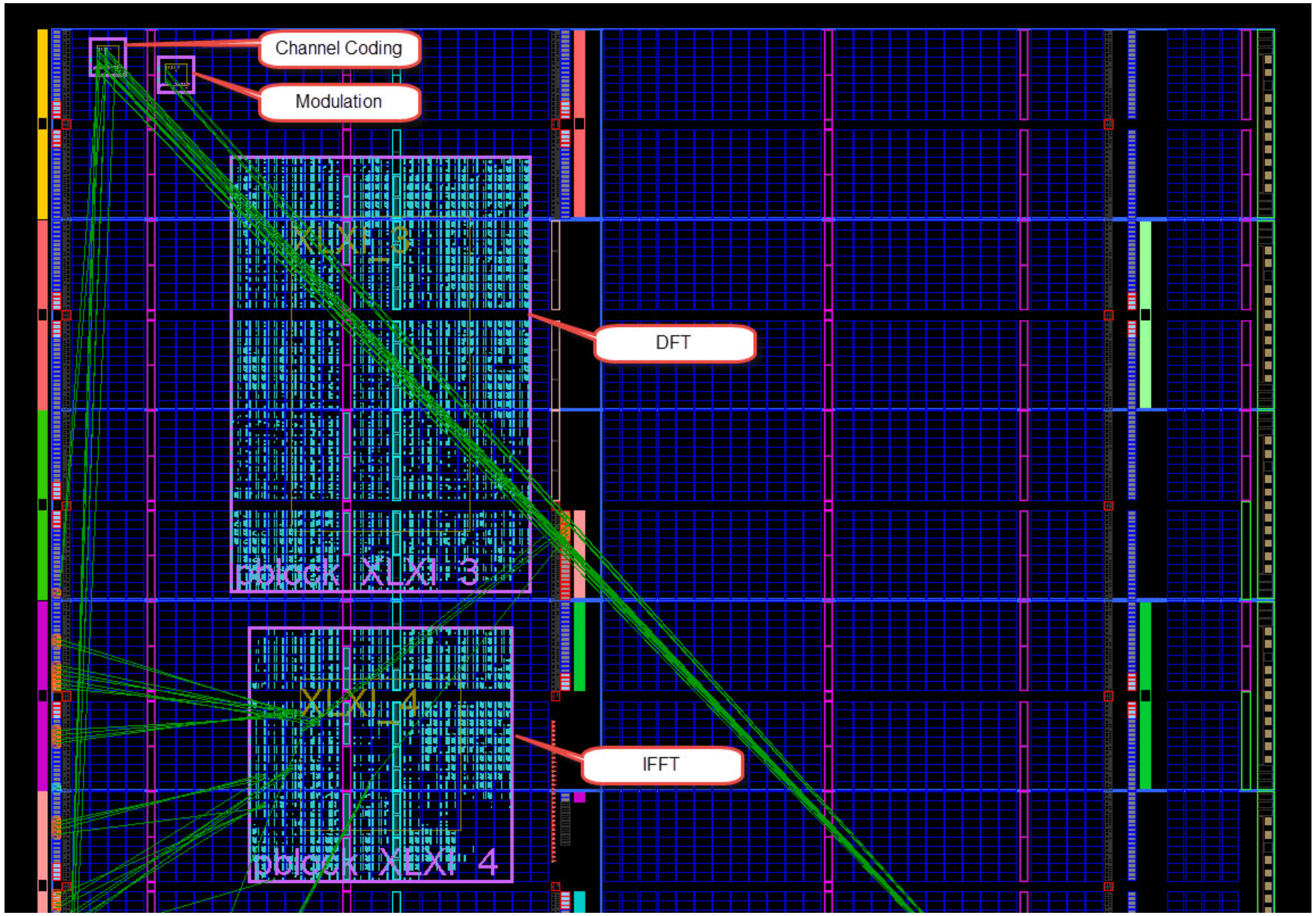


Figure 4.10: DR_3 for LTE loaded modules in the SDR chain

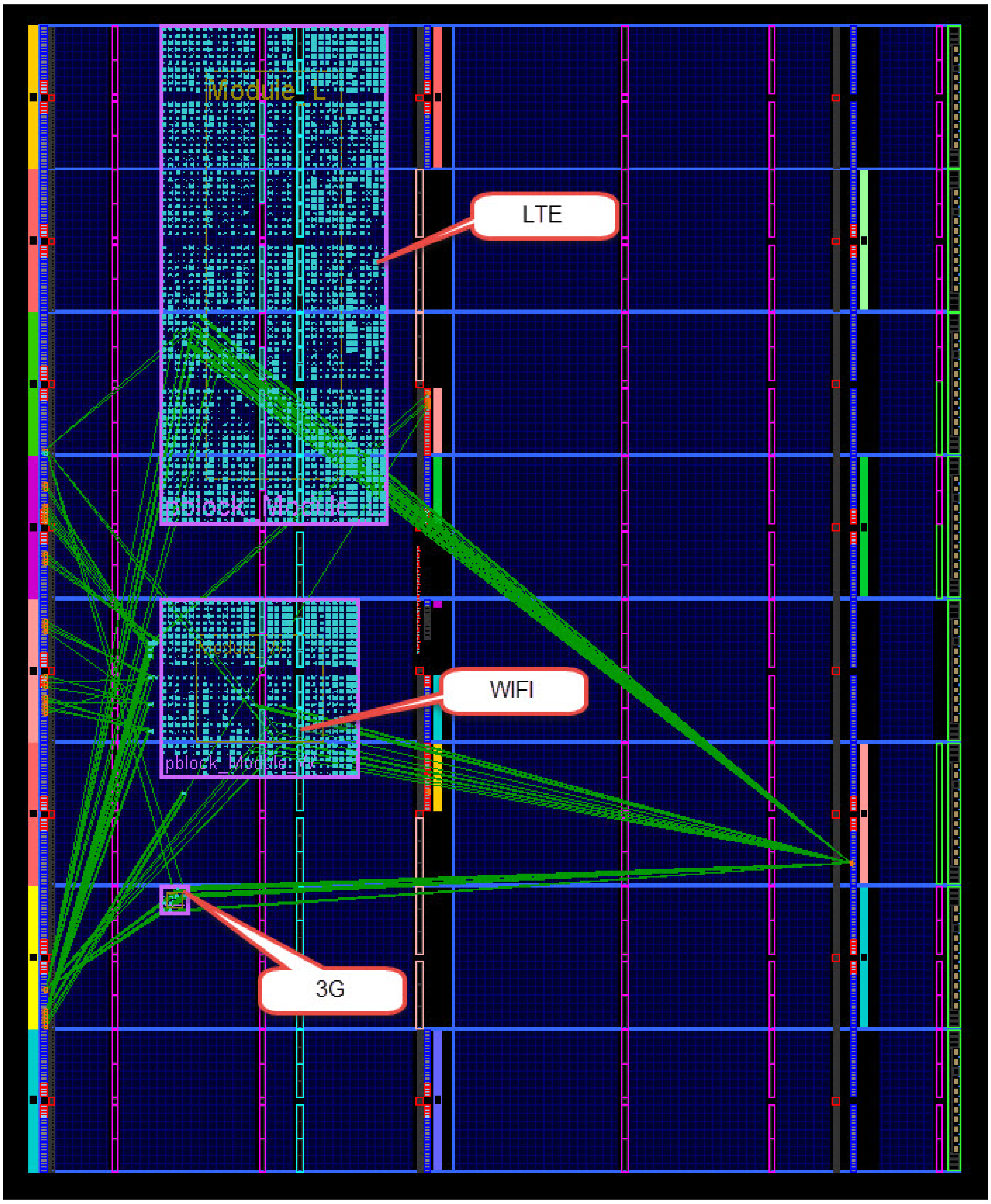


Figure 4.12: GCSM floorplaning

4.5.1 Area occupied on the FPGA

Table 4.3 summarizes the SLM system implementation shown in figures 4.2 and 4.3. It shows the number of LUTs used by each block, the utilization in the floorplanned partition and the generated bitstream files size. The channel coding module is floorpanned in 64 LUTs on the FPGA where the acquired number of LUTs differs from design to another, i.e. the 3G convolutional encoder with rate half acquires 3 LUTs out of the 64 LUTs, with a utilization 5% and the same for the other blocks. The floorplanned LUTs should be big enough to carry the largest module implementation which is the 3G convolutional encoder with rate third in the case of convolutional encoders. Note that the PlanAhead tool has some routing restrictions that needs to enlarge the area of the planned block.The generated file size 12.3KB for the channel coding block depends on the number of the floorplanned LUTs not the acquired LUTs. The 64 floorplanned LUTs generate 12.3 KB file size in channel coding block and modulation block while the 5520 floorplanned LUTs of the DFT generates a file size of 350 KB and the 2808 floorplanned LUTs of the IFFT generates a file size of 210 KB. Although the filler block doesn't acquire physical LUTs but the file generated according to the floorplanned LUTs but on the other hand, the filler block doesn't consume power as there is no LUTs acquired.

Table 4.3: Design area summary

Block type	Block definition	Number of LUTs	Floorplanned LUTs	Utilization (%)	Size (KB)
Channel Coding	3G Convo 1/2	3	64	5	12.3
	3G Convo 1/3	11	64	18	12.3
	4G Convo 1/3	9	64	15	12.3
	WIFI Convo 1/2	2	64	4	12.3
Modulation	BPSK	3	64	5	12.3
	QPSK	4	64	7	12.3
	16-QAM	9	64	15	12.3
DFT	64-Point DFT	4674	5520	85	350
	Filler	0	5520	0	350
IFFT	64-Point IFFT	1811	2808	65	210
	256-Point IFFT	1871	2808	67	210
	Filler	0	2808	0	210

Using DPR in the SDR chain decreases the number of physical LUTs used on the FPGA. Table 4.4 summarizes the area consumed in the three selected design runs using DPR versus the general SDR chain implementation GCSM. On using DPR, because only the used blocks are loaded, the maximum number of LUTs in the case of LTE design run (DR_3) is 6563 LUTs whereas the minimum in the 3G case (DR_1) is 6 LUTs. In the GCSM design without using DPR, the number of LUTs consumed is 6620 LUTs after XST tool optimization for the selected communication blocks per each standard shown in 4.11, where all blocks exist on the FPGA at the same time. This value of 6620 LUTs increases when all the other communication blocks are implemented for the different standards.

Table 4.4: Number of LUTs used per design run versus general design

Design Run	Standard	No. of LUTs
DR_1	3G - standard	6
DR_2	WIFI - standard	1817
DR_3	LTE - standard	6563
GCSM	Shown in figure 4.11	6620

4.5.2 Memory needed

In the SLM design, the initial configured chain size is 3.8 MB which is corresponding to the static part of the FPGA and the RP regions, note that there is an initial communication block loaded on the FPGA corresponding to each RP region. As shown in table 4.3, the needed memory for the SLM design is 5.2 MB which is equal to the initial configuration chain size and the size of each image for all the communication blocks. In the GCSM design, the bitstream file size is equal to 3.8 MB. Table 4.5 shows the summarized memory needed for both designs.

Memory needed for SLM Design = Initial configured chain size + sum (Number of communication blocks x Size of RM per block) = 3.8 MB + 1.4 MB = 5.2 MB

Table 4.5: Memory needed for SLM and GCSM

SLM	GCSM
Initial configuration bitstream File Size = 3.8 MB Reconfigurable bitstream file sizes = 1.4 MB	Configuration bitstream File Size = 3.8 MB
Total Size = 5.2 MB	Total Size = 3.8 MB

4.5.3 Power estimation

Using DPR in the SDR chain decreases the number of physical LUTs used on the FPGA when compared to GCSM, consequently, the power consumed decreases. Table 4.6 shows the consumed power in the SLM and GCSM designs. In the SLM design, the maximum power consumed in the FPGA logic occurs when the LTE design (DR_3) is loaded while the minimum power consumed in the FPGA logic occurs in the 3G design (DR_1). Where in the GSCM design, the consumed power of the communication system shown in 4.11 is 171 mW, where this value increases when the other communication blocks, for each standard, are implemented. The power consumed in the logic gates is estimated using Xilinx Power Analyzer (XPA) tool.

4.5.4 Time overhead

In the SLM design, the reconfiguration time is the time needed for reconfiguring a communication block in the SDR chain. The Maximum Theoretical Reconfiguration

Table 4.6: Power consumed in the SDR chain

Design Run	Standard	Logic power consumed
DR_1	3G - standard	0.24 mW
DR_2	WIFI - standard	42.49 mW
DR_3	LTE - standard	128.8 mW
GCSM	Shown in figure 4.11	171.47 mW

Time (MTRT) is equal to the summation of all reconfiguration time needed if all blocks are being reconfigured. Where the maximum delay of the data to be processed through the chain from the entry point till it is valid data at the output is equal to the MTRT in addition to the processing time per each block. In the GCSM, there is no time overhead added after the initial configuration because all standards communication blocks exist on the chip.

MTRT in SLM = Summation of Block sizes / configuration mode throughput using JTAG = (12.3 + 12.3 + 350 + 210) KB / 66 MHz = 8.65 ms

4.6 Conclusion

Generalizing the DPR concept in the different communication systems blocks leads to compact design in size and power with small memory and time overhead added. Where only the needed system will be loaded while others are stored in external memory. While full system implementation consumes more power and area without any additional memory and time overhead.

Chapter 5

Conclusion and Future work

The objective of this thesis is to design, simulate, and implement DPR system for SDR on FPGAs. This objective is met by investigating and modeling on two different steps. The first step by implementing DPR system for convolutional encoders used in different communication standards 2G, 3G, LTE and WIFI. Where the convolutional encoders initially not exist on the chip but stored in external memory and loaded on demand. This DPR design for the convolutional encoder is compared to conventional convolutional encoder system, where all encoders exist on the same chip. They are compared with respect to area, power, latency and memory. Softcore MicroBlaze processor is used with different frequencies to show the different aspects of the work. The results show that DPR implementation consumes less power and area when compared to the normal design. Whereas the normal design has less memory and latency. With the continuous upgrading in memory, it will not be a problem to store files of Kbits in memory of Gbits. The reconfiguration files can be stored on an external server and loaded through the network while handover takes place among different communication systems. On the other hand, the reconfiguration time can be held during the time taken to handover between two different communication standards.

In the second part of the thesis, ideal communication chains for 3G, LTE and WIFI are implemented using DPR technique where swapping occurs among different blocks for implemented encoders, modulation, FFT and DFT used in these standards. This produces a reconfigurable system that can adapt different communication standards. Using DPR shows an improvement in area and power consumption with fewer extra memory and latency when compared to the normal static implementation.

As a conclusion, DPR is a flexible and efficient way of realizing SDR in a Cognitive Radio system. Implementing a library of different encoding and modulation schemes as well as DFT and IFFT with different sizes and switching among them reduces the system complexity and makes it handy and real time upgradable. These designs of both parts are implemented on Xilinx FPGA kit XUPV5-LX110T.

5.1 Future work will include

1. Simplifying the communication system chain and generalizing the concept of DPR for other blocks.
2. Choosing the optimal values for the power, reconfiguration time and frequency in the DPR design.
3. How to get more benefits using DPR within broadcasting in different communication channels.
4. Generate more libraries of communication blocks to be loaded.
5. Increase number of standards using the DPR, WIMAX can be added.
6. Apply the same concept to the low power energy communication standards like 802.15.4 like ZigBee and Bluetooth.
7. This would help in offloading between cellular systems and fixed wireless systems such as WIFI, not only on the network level but also on the hardware level.

List of Publications

Published:

[1] A. Sadek, H. Mostafa, and A. Nassar, "On the Use of Dynamic Partial Reconfiguration for MultiBand MultiStandard Software Defined Radio," (Cairo Egypt), IEEE, In Press.

[2] A. Sadek, H. Mostafa, and A. Nassar, "Dynamic channel coding reconfiguration in software defined radio," (Casablanca, Morocco), IEEE, In Press.

References

[1] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, “NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey,” *Computer Networks*, vol. 50, pp. 2127–2159, 2006.

[2] T. Yücek and H. Arslan, “A survey of spectrum sensing algorithms for cognitive radio applications,” *Communications Surveys & Tutorials, IEEE*, vol. 11, pp. 116–130, 2009.

[3] A. A. Bletsas, *Intelligent antenna sharing in cooperative diversity wireless networks*. PhD thesis, Citeseer, 2005.

[4] J. Mitola III and G. Q. Maguire Jr, “Cognitive radio: making software radios more personal,” *Personal Communications, IEEE*, vol. 6, pp. 13–18, 1999.

[5] J. Mitola, “Cognitive Radio—An Integrated Agent Architecture for Software Defined Radio,” 2000.

[6] D. Koch, *Partial Reconfiguration on FPGAs: Architectures, Tools and Applications*, vol. 153. Springer Science & Business Media, 2012.

[7] 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 *Green Technologies (ICGT), 2012 International Conference on*, pp. 192–197, IEEE, 2012.

[8] K. Arun Kumar, “Fpga implementation of psk modems using partial re-configuration for sdr and cr applications,” in *India Conference (INDICON), 2012 Annual IEEE*, pp. 205–209, IEEE, 2012.

[9] 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.

[10] 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.

[11] T. Janevski, *Traffic analysis and design of wireless IP networks*. Artech House, 2003.

[12] A. Balasubramanian, R. Mahajan, and A. Venkataramani, “Augmenting mobile 3g using wifi,” in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pp. 209–222, ACM, 2010.

[13] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, “Mobile data offloading: how much can wifi deliver?,” in *Proceedings of the 6th International COnference*, p. 26, ACM, 2010.

[14] I. Kuon, R. Tessier, and J. Rose, “Fpga architecture: Survey and challenges,” *Foundations and Trends in Electronic Design Automation*, vol. 2, no. 2, pp. 135–253, 2008.

[15] G. Estrin, “Organization of computer systems: the fixed plus variable structure computer,” in *Papers presented at the May 3-5, 1960, western joint IRE-AIEE-ACM computer conference*, pp. 33–40, ACM, 1960.

[16] G. Estrin, “Reconfigurable computer origins: the ucla fixed-plus-variable (f+ v) structure computer,” *IEEE Annals of the History of Computing*, no. 4, pp. 3–9, 2002.

[17] H. Harada, Y. KAMIO, and M. FUJISE, “Multimode software radio system by parameter controlled and telecommunication component block embedded digital signal processing hardware,” *IEICE transactions on communications*, vol. 83, no. 6, pp. 1217–1228, 2000.

[18] X. , Inc., “Xilinx partial reconfiguration training,”

[19] X. , Inc., “Xilinx ug191 virtex-5 fpga configuration user guide (v3.11),” October 2012.

[20] E. Eto, “Xilinx xapp290 difference-based partial reconfiguration (v2.0),” vol. 3, December 2007.

[21] X. , Inc., “Xilinx ug702 partial reconfiguration user guide (v14.5),” April 2013.

[22] S. Kelem, “Xilinx xapp151 virtex series configuration architecture user guide (v1.7),” October 2004.

[23] P. Sedcole, B. Blodget, T. Becker, J. Anderson, and P. Lysaght, “Modular dynamic reconfiguration in virtex fpgas,” in *Computers and Digital Techniques, IEEE Proceedings-*, vol. 153, pp. 157–164, IET, 2006.

[24] P. Sedcole, B. Blodget, J. Anderson, P. Lysaghi, and T. Becker, “Modular partial reconfigurable in virtex fpgas,” in *Field Programmable Logic and Applications, 2005. International Conference on*, pp. 211–216, IEEE, 2005.

[25] D. Dye, “Xilinx wp374 partial reconfiguration of xilinx fpgas using ISE design suite(v1.2),” May 2012.

[26] S. Teig, “Going beyond the fpga with spacetime,” in *FPL2012 Keynote Oslo, Norway*, August 2012.

[27] X. , Inc., “Xilinx ug190 virtex-5 fpga user guide (v5.4),” March 2012.

[28] X. , Inc., “Xilinx ug081 microblaze processor reference guide (v17.7),” 2013.

[29] D. Lim and M. Peattie, “Xilinx xapp290 two flows for partial reconfiguration: Module based or small bit manipulations (v1.0),” 2002.

[30] D. Koch, C. Beckhoff, and J. Torresen, “Zero logic overhead integration of partially reconfigurable modules,” in *Proceedings of the 23rd symposium on Integrated circuits and system design*, pp. 103–108, ACM, 2010.

[31] “Digital cellular telecommunications system (Phase 2+); Channel coding,” *3GPP TS 45.003 version 12.1.0 Release 12*.

[32] “Universal Mobile Telecommunications System (UMTS); Multiplexing and channel coding (FDD),” *3GPP TS 25.212 version 12.0.0 Release 12*.

[33] “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding,” *3GPP TS 36.212 version 12.2.0 Release 12*.

[34] “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications,” *IEEE Computer Society LAN MAN Standards Committee and others*.

[35] “Digital cellular telecommunications system (Phase 2+); Physical layer on the radio path; General description,” *3GPP TS 45.001 version 12.1.0 Release 12*.

[36] “Digital cellular telecommunications system (Phase 2+); Multiplexing and multiple access on the radio path,” *3GPP TS 45.002 version 12.4.0 Release 12*.

[37] “Digital cellular telecommunications system (Phase 2+); Modulation,” *3GPP TS 45.004 version 12.0.0 Release 12*.

[38] “Universal Mobile Telecommunications System (UMTS); Physical layer - general description,” *3GPP TS 25.201 version 12.0.0 Release 12*.

[39] “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*.

[40] “Universal Mobile Telecommunications System (UMTS); Spreading and modulation (FDD),” *3GPP TS 25.213 version 12.0.0 Release 12*.

[41] “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*.

[42] “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); LTE physical layer; General description,” *3GPP TS 36.201 version 12.0.0 Release 12*.

[43] “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation,” *3GPP TS 36.211 version 12.5.0 Release 12*.

[44] “LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures,” *3GPP TS 36.213 version 12.5.0 Release 12*.

Appendix A

Xilinx Design Flow

A.1 Design Flow

FPGAs are Integrated Circuit (IC) chips that are programmed using Hardware Description Languages (HDL) such as (VHDL, Verilog). These languages are considered precise description for the logic circuits and its connections. Where the Computer Aided Design tools (CAD) are the tools that an engineer uses to develop the HDL, translate, verify, synthesis, simulate and program the FPGA. Recently High Level Synthesis (HLS) tool is used in the developing of the FPGA. This new tool allows software engineers to develop FPGA without the need to dig in the FPGA fabrication and clock level timing. HLS is developed using software languages like C/C++.

A.1.1 FPGA design flow

Figure A.1 describes the basic design flow of the FPGA:

Design entry: describe system requirements in HDL Language such as VHDL and Verilog. This HDL language represents the Register Transfer Level (RTL) of the circuit, which is a high level representation of the connections and blocks used in the circuit.

Functional Simulation: execute the written HDL code to ensure the output is same as required, no timing or delay calculations done at this stage.

Synthesis: convert the designed code (RTL) into logic gates and a netlist that describes the generated nets connections between the logic gates.

Place and Rout: or in other words where is the generated netlist will be placed and how routing is done on the FPGA fabric between the different LUTs.

Timing simulation: this stage takes care of the delays due to wires routing, logic gates and timing constraints.

Bitstream generation: after finishing the design, the generated bitstream file is formed that will be deployed on the FPGA chip. The format of the bitstream and how it configure the FPGA is a property of the FPGA vendors.

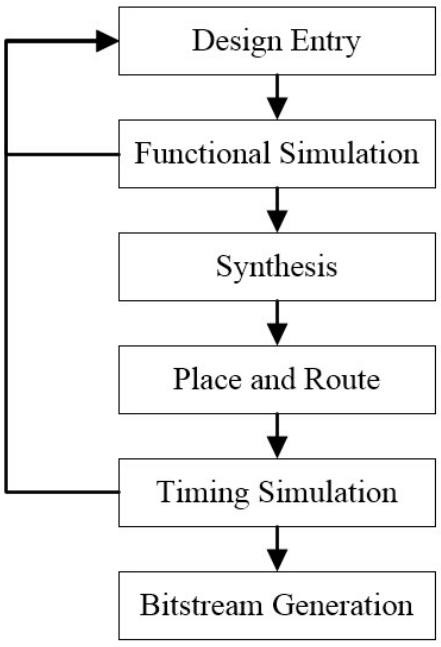


Figure A.1: FPGA design flow

A.1.2 ISE Suit for developing Xilinx FPGA

CAD tools are used for implementing and simulating electronic design. Integrated Synthesis Environment (ISE) suit is a CAD tool provided by Xilinx to facilitate the dealing with Xilinx FPGAs. Figure A.2 provide an overview of the main tools in the ISE suit and how they are mapped to the general FPGA design flow. Note that ISE latest version is 14.7, Xilinx discontinued this family of tools and upgraded them to a new set of tools which is named Vivado. Vivado is PlanAhead based software, takes the interface of PlanAhead, a powerful new tool but will not be discussed in the thesis context as it supports only the Xilinx 7 series till the written of this thesis.

Integrated Synthesis Environment (ISE): Is a design suit that controls Xilinx design flow, it facilitates accessing the design implementation files through the main project files. It provides the design entry by adding the VHDL / Verilog codes or schematics, synthesize the design through accessing Xilinx Synthesize Tool (XST), perform functional simulation and timing analysis using ISim and draw RTL diagrams.

PlanAhead: This tool provides more precise working with the FPGA resources. It helps in design placing and routing, pin assignment, refine timing constraints, CLBs selection and more. It also helps in partitioning design, hierarchical design and provides access to the Partial Reconfiguration (PR) design flow. It almost integrates with all Xilinx tools that why it became the base interface for the new Vivado tool.

Embedded Development Kit (EDK): EDK helps the engineer to design, debug and verify an entire embedded design, the two key components in the EDK tools are the Xilinx Platform Studio (XPS) and the Software Development Kit (SDK).

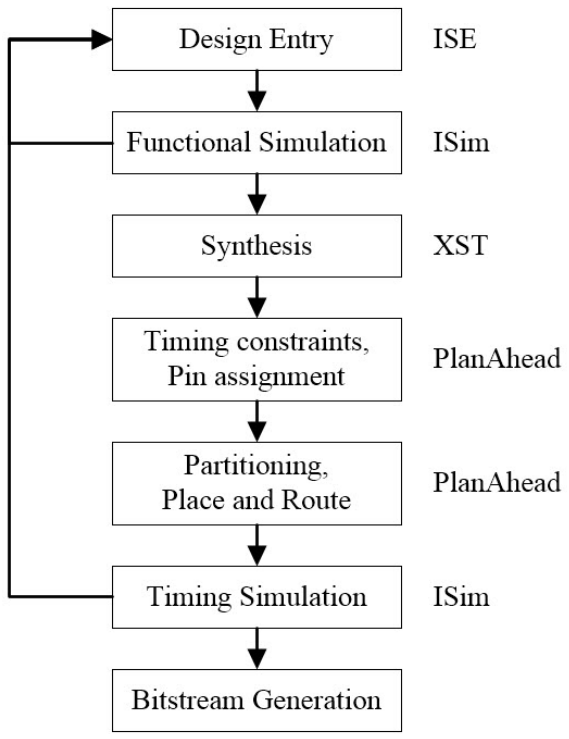


Figure A.2: Xilinx FPGA design flow

Xilinx Platform Studio (XPS): is a graphical platform for adding hardcore processors, softcore processors and different IPs for the design. It is a powerful tool included in the Embedded Development Kit (EDK) to design embedded systems. It helps engineer to build, connect and configure embedded processor-based system.

Software Development Kit (SDK): It is an eclipse-based software design environment, which includes GNU C/C++ compiler and debugger. After developing the software, Data2MEM utility is used for loading and updating bitstream with the developed software.

A.1.3 Xilinx DPR Design Flow

Figure A.3 simplifies the Xilinx DPR flow in both normal design and embedded design, where the embedded design is considered a normal design with extended System on Chip (SoC) functionality that the internal microprocessor controls the reconfiguration of the reconfigurable modules. In Xilinx DPR system, hardware modules that are partially reconfigured are called Partially Reconfigurable Modules (PRMs) these modules can be reconfigured real time during system operation. The region where a PRM module is configured is called a Partial Reconfigurable Region (PRR), whereas the other region is called Static Region that will contain the top module and related interfaces with the PRR. A bit stream file that contains the configuration data of PRM is called a Partial Bit Stream (PBS). Xilinx ISE Project Navigator manages the different modules in the DPR design, where the top module and the PRMs are designed in Verilog HDL or VHDL and are synthesized to a netlist file using Xilinx XST. In the top module, the PRMs are defined as only an external interface and designed as black box modules without internal logic. Where the internal logic will be defined in the PRMs files as a separate project.

For the DPR in embedded system, the external interface between the PRR with the embedded system is developed through the Xilinx XPS and SDK. Using XPS an embedded processor such as MicroBlaze or PowerPC is added and other needed IPs such as UART. In XPS process, the user IPs and its interface with the reconfigurable controller (processor like MicroBlaze) are designed. Using SDK, C program running on the embedded processor is implemented. The program is compiled by dedicated compilers to an Executable and Linkable Format (ELF) file.

Xilinx PlanAhead tool is a design tool for the entire FPGA design and implementation cycle. It is integrated with the Xilinx ISE and EDK, which makes it easier to export designs from these environments to the PlanAhead tool to finish implementation. It is the main part to apply the DPR concept through its flow, this flow is the same for the normal DPR design and the embedded DPR design.

In the PlanAhead design flow, the PRR is set using the tool. Then design runs are planned for the different configuration. These design runs are verified against each other to make sure of the interfaces are compatible when switching takes place. For each design run a bit stream file is generated for the full design, which includes the top module and the static part, as well as the partial bit stream files that contain the PRM.

In the embedded DPR the generated bitstream file of the full design and the generated ELF file are assembled using Data2MEM tool into a bit stream file for initializing FPGA.

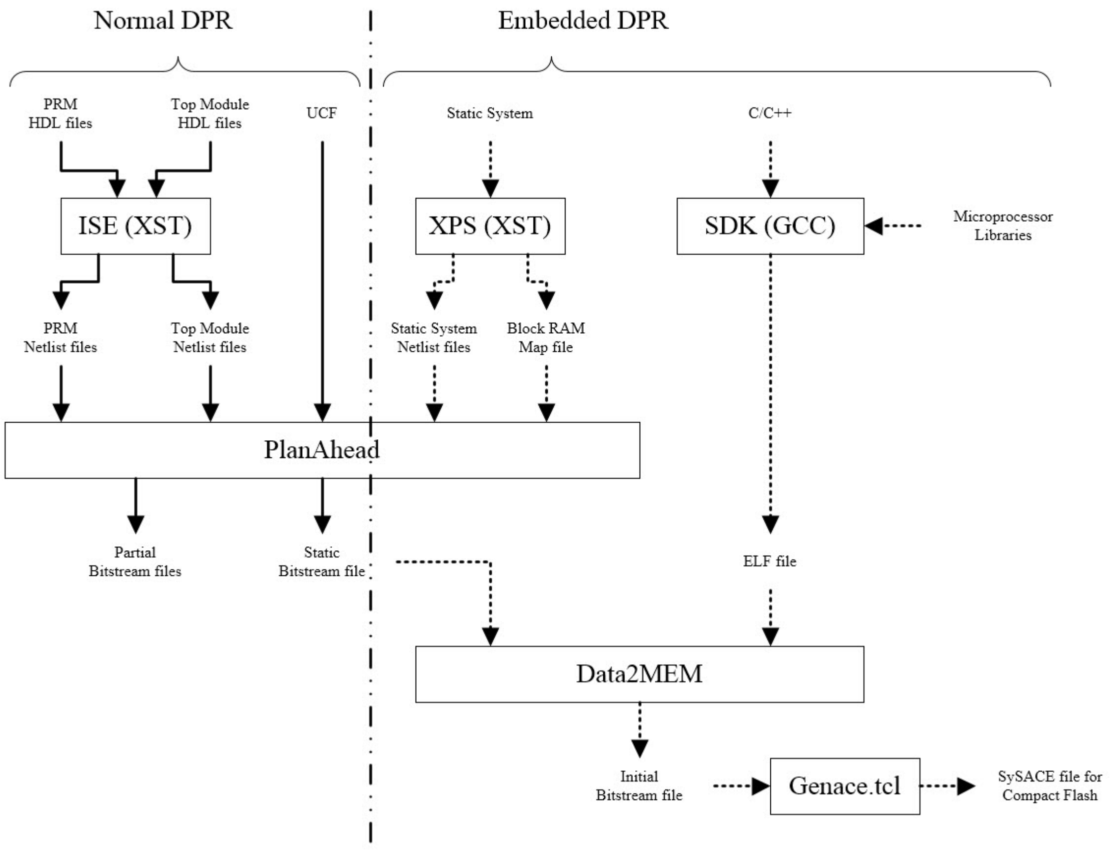


Figure A.3: Xilinx DPR Design Flow

الملخص

الراديو المعرف برمجيا (SDR) هو نظام اتصالات مصمم على أن تكون الطبقة المادية يمكن تنفيذها عن طريق وحدات محددة برمجية. الوحدات البرمجية ل(SDR) يمكن تكوينها و اعادة برمجتها بسهولة لتشكيل عدد مختلف من اجهزة الاتصالات ذات الطول الموجي. وتزداد فوائد ال(SDR) اذا تم اعادة تشكيل الجهاز على المستوى الفزيائي و يعتبر حقل مصفوفات البوابات المنطقية (FPGA) الحل الأمثل لتنفيذ إعادة التشكيل جهاز ال(SDR) في هذه الأطروحة يتم استخدام تقنية جديدة ل(FPGA) هي إعادة التشكيل الجزئي الديناميكي (DPR)، في تصميم مبتكر فى السماح بالتبديل بين معايير الاتصالات اللاسلكية المختلفة. هذا يؤدي الى تمديد عمر البطارية و تقليل المساحة المستخدمه على ال(FPGA). تم تنفيذ البحث واختباره على Xilinx Vertix-5 XUP V5-LX110T Kit.

في هذا العمل، يتم تنفيذ الراديو المعرف برمجيا باستخدام تقنية اعادة التشكيل اللحظي و الجزئي على جزأين. في الجزء الأول، يتم تنفيذ التحويل بين مختلف التلافيف التشفيرية التي تستخدم في معايير الاتصال المختلفة للجيل الثاني للمحمول (2G) و الجيل الثالث للمحمول (3G) و نظام التطور الطويل الأمد للمحمول (LTE) و تقنيات الاتصال اللاسلكي (WIFI) في المعيار (IEEE 802.11) و الأنمطة المختلفة لكل معيار. حيث تتم مقارنة اثنين من التطبيقات المختلفة، الأول هو تحقيق جميع أجهزة التشفير التلفيفي على الشريحة نفسها في نفس الوقت، و يدعى الوحدة العامة للتشفير (GEM). التنفيذ الثاني، هو تحميل نظام تلفيفي احادي (SLEM) حيث يكون هناك نظام تلفيفي واحد على ال(FPGA) و الاخرين يتم تخزينهم على وحدة تخزين خارجية و يعتمد على تقنية ال(DPR) و يتم تحميل الوحده حسب الطلب على ان يكون التحميل في الوقت الحقيقي حين أن النظام على وضعية التشغيل الكامل.

في الجزء الثاني من الأطروحة، يتم التحقق من مفهوم ال(DPR) من خلال تنفيذ السلسلة الأولية لنظم الاتصالات المختلفة للجيل الثالث للمحمول (3G) و نظام التطور الطويل الأمد للمحمول (LTE) و تقنيات الاتصال اللاسلكي (WIFI) في المعيار (IEEE 802.11). ويتم المقارنة مع هذه الانظمة المصممة بالطريقة التقليدية.

مهندس:	أحمد محمد صادق مبروك
تاريخ الميلاد:	7 \ 6 \ 1983
الجنسية:	مصري
تاريخ التسجيل:	1 \ 10 \ 2010
تاريخ المنح:	\ \
القسم:	الالكترونيات والاتصالات الكهربية
الدرجة:	ماجستير

ضع صورتك هنا

المشرفون:

أ.د. أمين نصار

د. حسن مصطفى

الممتحنون:

أ.د.	(الممتحن الخارجي)
أ.د.	(الممتحن الداخلي)
أ.د. أمين نصار	(المشرف الرئيسي)
أ.د.	(عضو)

عنوان الرسالة:

الراديو المحدد برمجيا باستخدام تقنية البرمجه الديناميكية واللحظية

الكلمات الدالة:

مصفوفات البوابات المنطقية القابلة للبرمجة ، الراديو المحدد برمجيا ، التغير اللحظي والديناميكي ، التغيير الجزئي

ملخص الرسالة:

فى هذا البحث يتم استخدام تقنية حديثة تتيح برمجة مصفوفات البوابات المنطقية برجمة ديناميكية ولحظية فى الميدان مما يؤدي الى تغير عملها كليا دون اي توقف. ويقدم هذا البحث كيفية استخدام هذه التقنية فى توحيد الطبقة الملموسة فى التقنيات المتعدده للاجهزة الاسلكية مما يؤدي الى تحسين كفاءة الراديو محدد برمجيا. ويتم تطبيق هذا عمليا من خلال توحيد الشكل العام لنظم الاتصاالات اللاسلكيه الجيل الثاني للمحمول (2G) و الجيل الثالث للمحمول (3G) و نظام التطور الطويل الأمد للمحمول (LTE) و تقنيات الاتصال اللاسلكي (WIFI) في المعيار (IEEE 802.11).

الراديو المحدد برمجيا باستخدام تقنية البرمجه الديناميكية واللحظية

إعداد

أحمد محمد صادق مبروك

رسالة مقدمة إلي
كلية الهندسة – جامعة القاهرة
كجزء من متطلبات الحصول علي درجة
الماجستير
في
الكترونيات الحاسب

كلية الهندسة – جامعة القاهرة
الجيزة – جمهورية مصر العربية
٢٠١٦

الراديو المحدد برمجيا باستخدام تقنية البرمجه الديناميكية واللحظية

إعداد

أحمد محمد صادق مبروك

رسالة مقدمة إلي
كلية الهندسة – جامعة القاهرة
كجزء من متطلبات الحصول علي درجة
الماجستير
في
الكترونيات الحاسب

تحت إشراف

أ.د. أمين نصار	د. حسن مصطفى
أستاذ دكتور	أستاذ مدرس
قسم قسم الالكترونيات والاتصالات الكهربية	قسم الالكترونيات والاتصالات الكهربية
كلية الهندسة – جامعة القاهرة	كلية الهندسة – جامعة القاهرة

كلية الهندسة – جامعة القاهرة
الجيزة – جمهورية مصر العربية
٢٠١٦

الراديو المحدد برمجيا باستخدام تقنية البرمجه الديناميكية واللحظية

إعداد

أحمد محمد صادق مبروك

رسالة مقدمة إلي
كلية الهندسة – جامعة القاهرة
كجزء من متطلبات الحصول علي درجة
الماجستير
في
الكترونيات الحاسب

كلية الهندسة – جامعة القاهرة
الجيزة – جمهورية مصر العربية
٢٠١٦