Diogo
Costa Marques

Switch P4 para Fronthaul O-RAN 5G

5G O-RAN Fronthaul P4 Switch

Universidade de Aveiro
2021

Diogo
Costa Marques

Switch P4 para Fronthaul O-RAN 5G

5G O-RAN Fronthaul P4 Switch

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Arnaldo S. R. Oliveira, Professor Auxiliar da Universidade de Aveiro, e do Doutor Pedro Filipe Vieira Rito, Investigador Auxiliar do Instituto de Telecomunicações de Aveiro
.

O presente estudo foi realizado ao abrigo do Projeto Aveiro STEAM City (UIA03-084), cofinanciado pelo Fundo Europeu de Desenvolvimento Regional, incluído no programa "Urban InnovativeActions".

O presente estudo foi também realizado ao abrigo do Projeto "Augmented Humanity" [POCI-01-0247-FEDER-046103 e LISBOA-01-0247-FEDER-046103], cofinanciado pelo Programa Operacional Competitividade e Internacionalização e pelo Programa Operacional Regional de Lisboa do PORTUGAL 2020, através do Fundo Europeu de Desenvolvimento Regional.

o júri / the jury

presidente / president

Prof. Doutor Diogo Nuno Pereira Gomes
Professor Auxiliar da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor Fernando Manuel Valente Ramos
Professor Associado da Universidade de Lisboa

Prof. Doutor Arnaldo Silva Rodrigues de Oliveira
Professor Auxiliar da Universidade de Aveiro (Orientador)

agradecimentos / acknowledgements

Agradeço à minha namorada, Joana, por me apoiar e acompanhar durante esta etapa.

Aos meus pais e irmão por acreditarem em mim e pelo suporte prestado durante todo o meu percurso académico.

A todos os meus amigos pelo apoio, pelas conversas construtivas e momentos de diversão.

Por fim, a disponiblidade e ajuda dos meus orientadores, Doutor Arnaldo Oliveira e Doutor Pedro Rito, e dos meus colegas do Instituto de Telecomunicações.

Palavras Chave

5G, Open RAN, Fronthaul, P4, switch, FPGA.

Resumo

Com o desenvolvimento das redes de comunicação móveis o 5G emergiu para satisfazer novos requisitos e novos casos de utilização. O 5G irá causar um grande impacto não só na sociedade mas também na indústria, permitindo maiores taxas de transmissão de dados, baixa latência e a conectividade necessária para ligar tudo, incluindo máquinas e dispositivos, permitindo avançar para o próximo patamar de conectividade mundial. A Rede de Acesso Rádio 5G tem se tornado um conjunto de estações base que suportam o transporte de dados, transmissão e recepção de sinais de rádio. Ao fornecer conectividade entre dispositivos com o núcleo da rede, é o elemento chave de um sistema 5G que precisa de evoluir para que se possam cumprir os requisitos atuais. Assim, a RAN viu a sua arquitectura tornar-se mais centralizada e virtualizada. Inicialmente, o aparecimento da arquitectura C-RAN propôs uma divisão da RAN em dois componentes, com uma divisão das funções fixa e uma ligação fronthaul que liga ambas as unidades. Havendo necessidade de mais flexibilidade, surge uma nova RAN com uma divisão que consiste em CU, DU e RU com várias opções na divisão de funções. A O-RAN pretende fazer evoluir a RAN para uma rede mais orientada para o software, virtualizada e flexível, apoiando a interoperação entre fornecedores.

Tendo em consideração os requisitos que uma ligação fronthaul possui, tais como janelas de tempo restritas uma vez que se trata do transporte de sinais de rádio, o objectivo desta dissertação é a concepção de um switch que permite que a ligação fronthaul de uma O-RAN seja partilhada com tráfego de uso geral. O switch com três portas de alto desempenho combina tráfego de uso geral e de alta prioridade numa ligação óptica em ambas as extremidades da ligação de fronthaul. A ferramenta P4-SDNET foi utilizada para descrever como é feito o processamento de pacotes, data plane, através da linguagem P4. O desenvolvimento foi realizado utilizando um kit de desenvolvimento baseado numa FPGA e o ambiente Xilinx SDNet que permite a programação do data plane com linguagens P4 tendo como alvo a FPGA. Foi também concebida uma plataforma de geração de tráfego, com base no mesmo kit, para testar a latência que os switchs adicionam ao fronthaul. Foram obtidos resultados positivos com latências adicionadas entre 1 e 3 microsegundos para o tráfego O-RAN. Contudo, a principal limitação do sistema é o reconhecimento do tráfego do S-Plane.

Keywords 5G, Open RAN, Fronthaul, P4, Switch, FPGA.

Abstract

The development of mobile communication networks has led to the creation of the fifth generation of cellular networks (5G) to meet new requirements and new use cases. 5G is expected to have a large impact not only on society but also in industry, enabling increased peak data rates, low latency and the connectivity needed to link everything, including machines and devices, enabling the next level of world connectivity. The 5G RAN has become a collection of base stations that support the capabilities of data transport, radio transmission and reception. By providing connectivity between devices and sensors with their core network, is the key element of a 5G system that needs to evolve to enable today's requirements. Thus, the RAN has seen its architecture become more centralized and virtualized. Initially, C-RAN architecture proposed a division of the RAN in two, with a fixed function split and a fronthaul link that links both units. There was a need for more flexibility and a new RAN arises with the split into CU, DU and RU with various options in the functional split. The O-RAN intends to evolve the RAN to a network more software-driven, virtualized and flexible, supporting interoperation between vendors.

Taking into consideration the requirements that a fronthaul link has, such as restricted time windows since it is dealing with radio signals, the goal of this dissertation is the design of a switch that allows the fronthaul link of an O-RAN to be shared by general-purpose traffic. The high performance three-port switch combines general-purpose and high-priority traffic in one optical link at both ends of the fronthaul link. The P4-SDNET tool was used to describe how the data plane processes the packets through the P4 language. The development is done by using a development kit based on FPGA and the Xilinx SDNet high-level environment that allows the design of packet-processing data plane with P4 languages that target FPGA hardware. A traffic generation platform was also designed, based on the same kit, to test the latency that the switches add to the fronthaul. Positive results were obtained with added latencies between 1 and 3 microseconds for O-RAN traffic. However, the main limitation of the system is the recognition of S-Plane traffic.

Contents

List of Figures

List of Tables

Glossary

3GPP	3rd Generation Partnership Project
5GC	5G Core
AI	Artificial intelligence
AMBA	Advanced Microcontroller Bus Architecture
AXI	Advanced eXtensible Interface
BBU	Baseband Unit
CAPEX	Capital expenditures
CDC	Clock Domain Crossing
CN	Core Network
COE	Coefficient
C-Plane	Control Plane
CPRI	Common Public Radio Interface
C-RAN	Centralized RAN
CUS-Plane	Control, User and Synchronization Plane
CU	Centralized Unit
C-V2X	Cellular Vehicle-to-Everything
DL	Downlink
DMA	Direct Memory Access
D-RAN	Distributed RAN
DU	Distributed Unit
eAxC	extended Antenna-Carrier
eCPRI	Enhanced CPRI
eMBB	Enhanced mobile broadband
EPC	Evolved Packet Core
E-UTRAN	Evolved Universal Terrestrial Radio Access Network
E-UTRA	Evolved Universal Terrestrial Radio Access
FCS	Frame check sequence
FFT	Fast Fourier transform
FIFO	First In First Out
FMC	FPGA Mezzanine Card
FPGA	Field Programmable Gate Array
GEM	Gigabit Ethernet MAC Controller
GEM	Gigabit Ethernet MAC
GMII	Gigabit media-independent interface
gNB	5G Node B
GSM	Global Systems for Mobile communications
GUI	Graphical user interface
HDL	Hardware Description language
HLS	High layer split
ICT	Information and Communications Technology

ILA	Integrated Logic Analyzer
IMT	International Mobile Telecommunications
IoT	Internet of Things
IoT	Internet of Things
IP'	Intellectual property
IP	Internet Protocol
ITU	International Telecommunication Union
JSON	JavaScript Object Notation
LLC	Logical Link Control
LLS	Lower layer splits
LTE	Long Term Evolution
LUT	Look up Table
MAC	Media Access Controller
MGT	Multi-Gigabit Transceiver
MII	Media-independent interface
MIMO	Massive Multiple Input Multiple Output
ML	Machine learning
MMS	Multimedia Messaging Service
mMTC	Massive machine-type communications
M-Plane	Management Plane
MPSoC	MultiProcessor System On Chip
NETCONF	Network Configuration Protocol
NFV	Network Functions Virtualization
NF	Network functions
ng-eNB	Next-generation Evolved Node B
NG	Next Generation
NMS	Network Management System
NR	New Radio
NSA	Non-Stand Alone
O-Cloud	O-RAN Cloud
O-CU-CP	O-RAN Central Unit – Control Plane
O-CU-UP	O-RAN Central Unit – User Plane
O-CU	O-RAN Control Unit
O-DU	O-RAN Distributed Unit
O-DU	Open DU
OPEX	Operating expense
O-RAN	Open RAN
O-RU	O-RAN Radio Unit
O-RU	Open RU
OSI	Open Systems Interconnection
P4	Programming Protocol-independent Packet Processors
PCS	Physical Coding Sublayer
PDCP	Packet data convergence protocol
PHY	Physical layer
PL	Programmable logic
PMA	Physical Medium Attachment
PMD	Physical Medium Dependent
PS	Processing system
PTP	Precision Time Protocol
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio Access Technologies

RF	Radio frequency
RGMII	Reduced gigabit media-independent interface
RIC	RAN Intelligent Controller
RLC	Radio link control
RNL	Radio Network Layer
RoE	Radio over Ethernet
RRH	Remote Radio Head
RTL	Register Transfer Level
RT	Real Time
RU	Remote Unit
RX	Receive
SA	Stand-Alone
SDN	Software Defined Networking
SFP+	enhanced small form-factor pluggable
SFP	Small form-factor pluggable
SMO	Service Management & Orchestrator
SMS	Short Message Service
SyncE	Synchronous Ethernet
TEMAC	Tri-Mode Ethernet MAC
TNL	Transport Network Layer
TSG	Technical Specification Groups
TX	Transmit
UART	Universal Asynchronous Receiver-Transmitter
UE	User equipment
UL	Uplink
U-Plane	User Plane
URLLC	Ultra-reliable and low-latency communications
VHDL	Very High-Speed Integrated Circuit Hardware Description Language
vRAN	virtualized RAN
XGMII	10-gigabit media-independent interface

CHAPTER 1

Introduction

1.1 Scope

Driven by technology developments and society's needs, a new generation of mobile wireless communications appear every 7 to 10 years. The first generation appeared in the early 1980s, this totally analog network allowed the first cell phones to be connected. The lack of security, low sound quality and reckless handoff made it a network with many disadvantages and several vulnerabilities. Around 10 years later, with completely different technologies, using digital signals this time, came the second generation. The Global Systems for Mobile communications (GSM) being the main system used for voice communications, provided new ways of communication by introducing Short Message Service (SMS) and Multimedia Messaging Service (MMS). The third generation of mobile wireless communications had 4 times the data transfer capabilities of 2G, and it has managed to combine high-speed mobile access with Internet Protocol (IP) based services. Although 3G requires more power it has brought features like global roaming and better voice quality. 4G is based on Long Term Evolution (LTE) and has launched the mobile broadband era. A 4G system improves communication networks by providing a reliable IP-based solution. Voice, data, and multimedia will be transmitted at much higher data rates, enabling services such as high-quality video streaming, fast mobile web access or HD videos. [1]

Since its adoption in 2009, 4G has just about reached its capacity in terms of data transferring speeds. With an increase in the demand of the users, the world needs a faster and reliable network [2]. Despite that, the 5th generation (5G) of mobile and wireless communications is expected to have a large impact on society and industry that will go far beyond the information and communications technology field. On one hand, it will enable significantly increased peak data rates compared to previous cellular generations, and allow for high experienced data rates almost anytime and anywhere. 5G networks are expected to offer the connectivity needed to link everything and everyone including machines and devices, enabling the next level of human connectivity. [3]

The development of 5G communication systems focuses on three fundamental issues, namely increased capacity, massive connectivity and a diverse set of services. As shown in 1.1, the International Telecommunication Union (ITU) targets three main usage scenarios with distinct connectivity requirements:

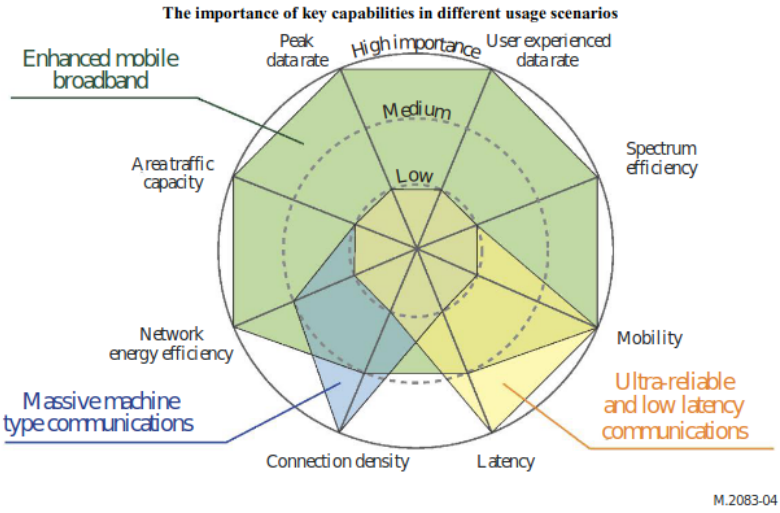


Figure 1.1: The importance of key capabilities in different usage scenarios.

- **Enhanced mobile broadband (eMBB)** - the Enhanced mobile broadband addresses a human-centric use case with an increasing demand for more data. eMBB has main data-intensive applications requiring enhanced access to multimedia content, services and data with improved performance, such as mobile high-definition video streaming or immersive gaming. These are all enabled by 5G extreme data rates, larger data capacity, and other performance improvements.
- **Ultra-reliable and low-latency communications (URLLC)** - Services like Cellular Vehicle-to-Everything (C-V2X), autonomous vehicles, intelligent factories, remote medical surgery, robots need extremely high reliability, availability and security. The reliability is defined as the probability of successful data delivery within a specified time. It is expected that URLLC services will provide the main part of the foundation of Industry 4.0 and have a substantial impact on industries far beyond the Information and Communications Technology (ICT) industry. URLLC features are designed to meet the requirements of 'no-failure' devices and optimizations must be made at every step of the Downlink (DL) and Uplink (UL) transmission process. The need to reduce data processing response times is also leading to the emergence of highly distributed edge computing strategies.
- **Massive machine-type communications (mMTC)** - This use case is characterized by a large number of connected devices, which usually need to be cheap and have a long battery life, transmitting a low volume of non-delay-sensitive data. A key example for this service type would be logistics applications, involving the tracking of tagged

objects) or agricultural applications where small, low-cost and low-power sensors are spreaded over large areas to measure ground humidity and fertility.

In order to understand more concretely the engineering challenges facing these three 5G service types, it is necessary to first identify the requirements for a 5G system. The ITU-R has recommended a set of parameters to be key minimum technical performance requirements of International Mobile Telecommunications (IMT)-2020 [4]. It should be taken into consideration that all requirements do not have to be fulfilled at the same time. In order to efficiently accommodate vertical use-cases along with increased user demands over the same network infrastructure the network will be logically sliced into different virtual networks in order to meet diversified service requirements and provide flexible support to various application scenarios.

- **Peak data rate** is the maximum achievable data rate that a user device transmits or receives under ideal conditions, in bits per second. The minimum peak data rates are 20 Gbps in the DL and 10 Gbps in the UL.
- **Peak spectral efficiency** is the maximum data rate under ideal conditions normalised by the channel bandwidth, in bps Hz. This requirement is set to 30 bps Hz in the DL and 15 bps Hz in the UL. User experienced data rate refers to the achievable data rate that is available continuously across the coverage area to a mobile user or device. The target values for the user experienced data rate are 100 Mbps in the DL and 50 Mbps in UL.
- **Latency** can be divided into user and control plane latency and is the time from when the source sends a packet to when the destination receives it and the transition time from different connection modes, in ms.
- **Connection density** is the total number of connected or accessible devices per unit area. The minimum value for connection density is 1 000 000 devices per km2.
- **Energy efficiency** refers to the capability to minimize the radio access network energy consumption concerning the traffic capacity provided, on the network side and on the device side is the capability to minimize the power consumed by the device modem.
- **Reliability** is defined as the success probability of transmitting a layer 2/3 packet within a required maximum time. The main goal is to transmit a packet of 32 bytes in less than 1ms with 99.999% probability.
- **Mobility** is given as the maximum moving speed of a user device (terminal) at which the device can provide a certain quality of communication link to a base station.
- **Bandwidth** refers to the maximum aggregated system bandwidth. Is recommended to be at least 100 MHz and up to 1 GHz in higher frequency bands, above 6 GHz.

1.2 Motivation

With 5G, new technologies are increasing based on concepts such as virtualization and flexibility to survive new requirements imposed by the evolution of industry, society and their smart cities. In order to create an open and intelligent network capable of supporting interoperation between

vendor's RAN equipment, emerges the Open RAN (O-RAN). The O-RAN architecture is an evolution of the architectures adopted by Centralized RAN (C-RAN) and Next Generation (NG)-Radio Access Network (RAN), where there is a split of functionalities into three distinct units. The process of splitting and centralization results in Centralized Unit (CU), Distributed Unit (DU) and Remote Unit (RU) and the fronthaul link. The open fronthaul is the transport layer that interconnects the RU and the DU combining the complexity of the RU with the centralization of functions. Of the various options for separation of functions between these units the O-RAN adopts the split option 7.2x. This option makes the RU simple but brings time constrained requirements due to the critical packets transported.

Due to the implementation of a 5G O-RAN based system there is a need to share the fronthaul link with traffic from other devices external to the 5G network. This scenario arises because there are general purpose communication devices that want to communicate with each other, being located in the same place in the DU and RU. With the assumption that the remaining traffic, general purpose, is not critical and the high latency does not restrict the system. The purpose is to analyse if it is possible to share the link with the critical fronthaul traffic. Always prioritizing O-RAN packets, it is expected that the requirements will be met, while avoiding the creation of a new link between locations.

1.3 Objectives

The objective of this dissertation is the design and validation of a high performance 3-port switch, which is able to combine in one optical link general-purpose traffic and high priority traffic between the RU and the DU of a 5G network.

The switch is expected to be as versatile as possible taking advantage of a reconfigurable data plane. The switch is intended to be inserted at both ends of the fronthaul link connecting the RU and the DU, while providing 3 ports. One of the 10Gbps ports is deployed to connect with the fronthaul link and the other two, one Gbps and one 10Gbps, are intended for general-purpose equipment and the 5G DU/RU, respectively. Thus, the switch in one direction must prioritize traffic coming from the DU/RU and in the other direction needs to characterize traffic coming from the fronthaul link so that it can forward ORAN packets to the 10Gbps port. In this way, it is possible to make the type of switching equipment used in the datacenter flexible while meeting strict time constraints.

In order to fulfill the main goal of creating and validating the switch, the following tasks were outlined.

- Study and familiarization with concepts such as O-RAN, fronthaul and SDN;
- Specification of the switch to be implemented, defining its internal architecture;
- Design of the proposed solution based on FPGA;
- Design of a platform for traffic generation and time measurement;
- Validation and testing of the proposed solution.

1.4 Document structure

Following this chapter the document is organised as follows:

- **Chapter 2, Fronthaul in Radio Access Networks** - Is discussed radio access networks in a 5G architecture, with a main focus on the fronthaul link of an open RAN.
- **Chapter 3, Programmable Data Plane** - The concept of SDN is discussed and how the P4 language and the SDNet framework is useful.
- **Chapter 4, Proposed Solution and Implementation** - The general architecture of the switch is presented, by first studying the scenario in which it will be integrated. It is explained in detail how the various components presented in the proposed solution were implemented.
- **Chapter 5, Test Platform and Initial Results** - The traffic generator and monitor platform used to test the switch performance is presented. Tests and validation done are described.
- **Chapter 6, Optimization and Final Results** - Some solutions are evaluated to optimize the switch, is presented how it was implemented and presented the results of the tests performed.
- **Chapter 7, Conclusion** - The main conclusions and limitations of the work are presented, and future work is discussed.

CHAPTER 2

Fronthaul in Radio Access Networks

This chapter explores the adoption of O-RAN, contextualizing the evolution of 5G RAN and architectures.

2.1 3GPP

The success of mobile communication has relied heavily on international technical specs and standards. This has enabled the deployment and interoperability of devices and infrastructure from many suppliers, as well as the operation of devices and subscriptions on a worldwide scale.

The 3rd Generation Partnership Project (3GPP) [1] gathers various telecommunications standard development organizations and provides a stable environment for their members to produce the Reports and Specifications that define 3GPP technologies. The project covers mobile telecommunications technologies, providing a detailed overview of the mobile telecommunications system. The specifications also provide hooks for interworking with non-3GPP components, for instance, non-3GPP Access Networks . It is currently developing specifications for 5G systems, and is divided into three Technical Specification Groups (TSG):

- Radio Access Networks (RAN);
- Services & Systems Aspects (SA);
- Core Network & Terminals (CT).

This document is primarily based on the specification and technical reports of the 3GPP Technical Specification Group RAN. This group ensures that systems based on 3GPP specifications are capable of rapid development and deployment with the provision of global roaming of equipment. Reducing complexity and avoiding fragmentation of technologies on offer is the goal of each 3GPP radio access technology.

[1] https://www.3gpp.org/

2.2 5G Telecommunications Systems

Conceptually, a wireless telecommunications system consists of 3 main components: the Core Network (CN), the RAN and the User equipment (UE). At the end of the system are the UEs, which are mobile phones or other wireless devices, used by an end-user to communicate. The CN is the central element of a system that is responsible for critical functions and provides services to subscribers. A RAN is a type of network infrastructure that wirelessly link the users and the core usually consists of radio base stations with large antennas. [3]

Like the system mentioned above, the mobile network functional architecture has traditionally been divided into two main components: RAN and CN. Bearing in mind that the current trend is the softwarization and decentralization of the network the future 5G networks will have a heterogeneous physical deployment, in terms of different frequency bands, different cell sizes, but also the co-existence of different Radio Access Technologies (RAT) and air interface variants. One of the objectives of 5G networks is to reduce to a minimum any dependencies between the core and access networks, allowing an independent evolution. The interface between the CN and RAN is a logical point-to-point interface and will support the separation between control and user plane. It is required to be open and future-proof, and it will be decoupled within the possible RAN deployment variants.

In the last decade the mobile operators have invested a lot of money to build 4G networks and have faced a challenge in order to support large-scale 5G deployments. So it would be very convenient if it were possible to take advantage of the current 4G infrastructure. The 5G system will support the following scenarios for connectivity between RAN consisting of Evolved Universal Terrestrial Radio Access (E-UTRA) and New Radio (NR), and a CN consisting of an 5G Core (5GC) and an Evolved Packet Core (EPC). The 3GPP proposes in the Technical Report 21.915 Release 15 [5], two types of 5G systems deployment solutions are shown in the figure 2.1:

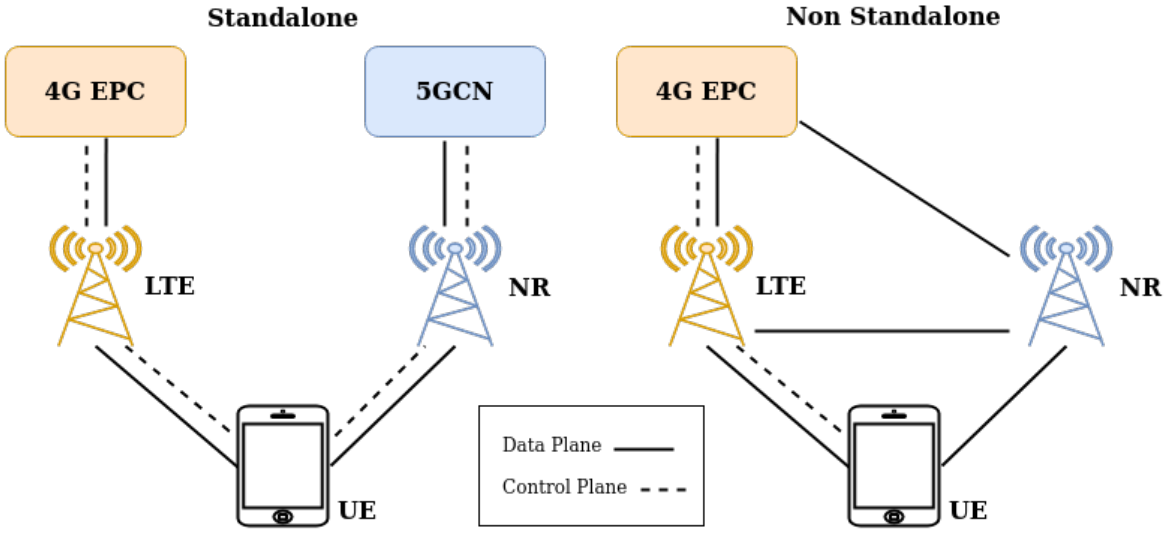


Figure 2.1: SA and NSA deployments solutions.

- **Non-Stand Alone (NSA)**: the 5G RAN and its NR interface is used in conjunction with the existing LTE and EPC infrastructure Core Network, allowing it to take advantage of NR without having to replace the network core. This architecture is considered as a simpler introduction to the handling of service requirements in eMBB services.
- **Stand-Alone (SA)**: the 5GC is introduced with several new capabilities built inherently into it, being able to support all 5G services and requirements. Being addressed as a full 5G deployment, no part of a 4G network is needed to operate. The 5G RAN is only connected to the 5GC.

2.2.1 5G Core Network

5G Core is based in cloud technologies optimised for a cloud-native, programmable, modular and service-based architecture making it a future-proof solution. The technology behind the core architecture is Network Functions Virtualization (NFV) and Software Defined Networking (SDN). With this new services and functionalities may be introduced more quickly and in accordance with agile techniques. Network functions (NF) can offer services to consumers according to their needs since they have flexibility within the network, for instance, these can be placed in specific locations in order to solve latency problems.

Like previous versions of core networks, the 5GC is able to keep track of subscription information, register UEs, establish data sessions, traffic forwarding in both UL and DL directions and ensure Quality of Service (QoS) functions.

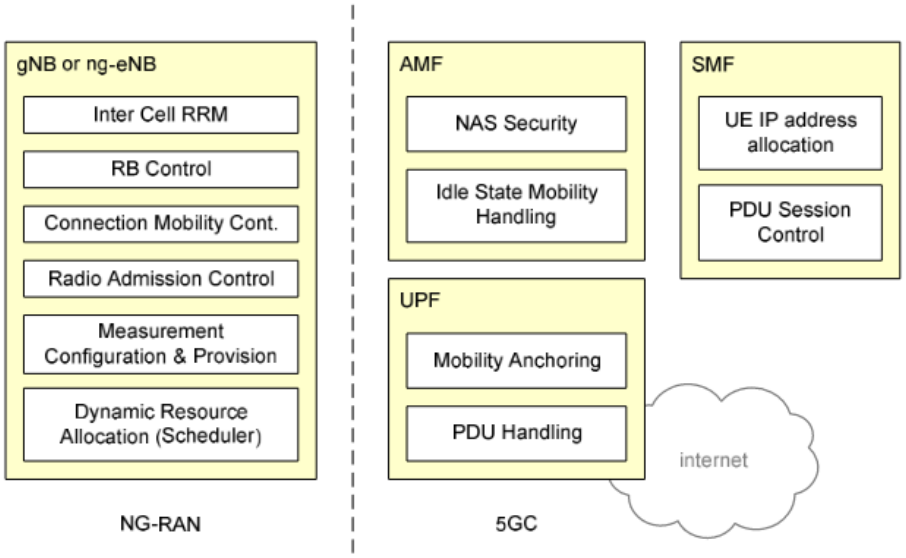


Figure 2.2: Functional Split between NG-RAN and 5GC (retrieved from [6]).

2.3 Radio Access Network

The relative mobile market share is shown in the figure 2.3, 4G is about to reach the peak of the mobile market in the next year, remaining the most dominant mobile technology, with

more than 50% of connections (excluding licensed cellular Internet of Things (IoT)). While the other generations enter the decline, 5G adoption emerges and is expected to represent 20% of the connections by 2025. In this way, it becomes clear that the current base station RANs need to guarantee the coexistence of 5G and 4G user equipment, which will still form an important part of the cellular demand.

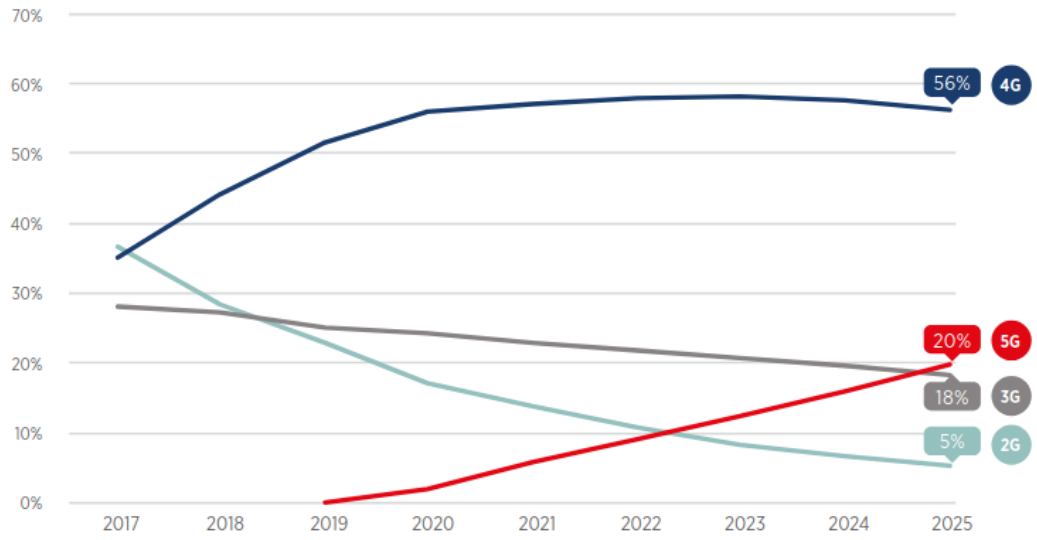


Figure 2.3: Mobile market share (retrieved from [7]).

In the latest deployments, the RAN is a collection of base stations that supports the capabilities of data transport, radio transmission and reception. In the 5G era, these capabilities shall be enhanced to accommodate the requirements and use cases discussed in the previous chapter 1. Being the edge element in a mobile network, the RAN exchanges information through the air interface with mobile devices by implementing a RAT.

2.3.1 Centralized RAN

In more traditional RAN architecture, all radio and baseband processing functions were present at the base station, at the same location. With 3G and 4G the RAN has evolved to a distributed RAN, here the radio functions are separated from the signal processing functions giving rise to two new units, Remote Radio Head (RRH) and Baseband Unit (BBU). Both units can be placed in different locations where it is more convenient and advantageous, but each RRH is connected to its own BBU. The Distributed RAN (D-RAN) is an efficient RAN solution for 3G and 4G networks, but did not meet the requirements of 5G networks, such as high bandwidth, low latency and cost efficient services. In order to meet some requirements of 5G mobile networks a new architecture has emerged, a C-RAN. [8]

As in D-RAN, C-RAN has divided the traditional base station into two parts, RRH and BBU. To fulfill the requirements, distributed RRHs continue to exist and the BBUs are clustered into a pool, a centralized site having a set of BBUs. The pool is able to support up to tens of RRHs, and is connected through a backhaul link with the core network. The link

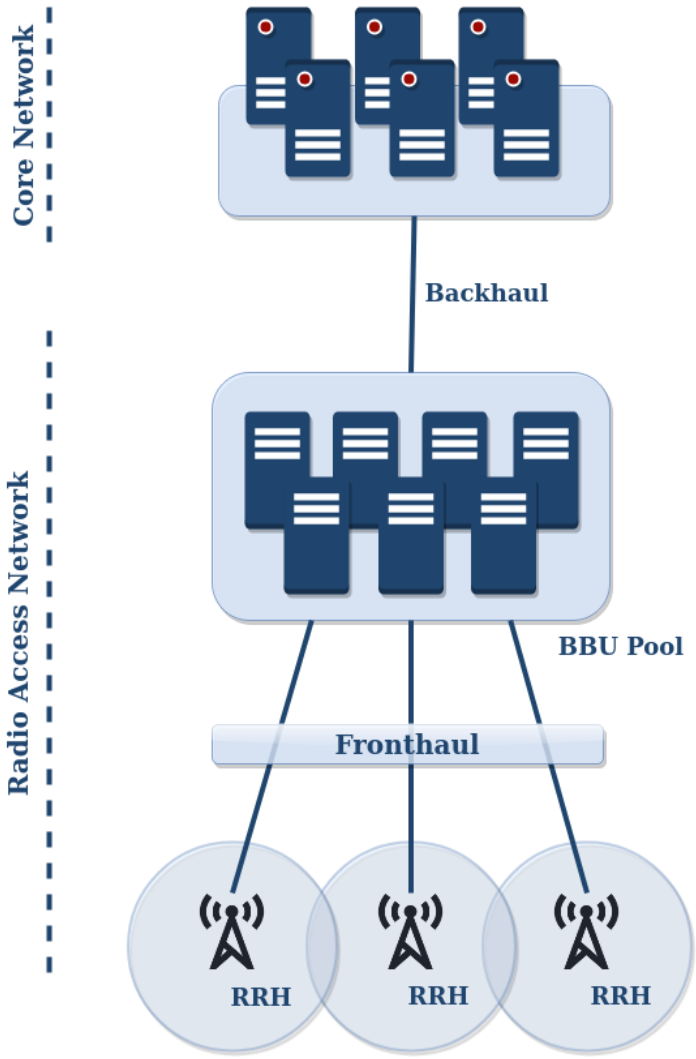


Figure 2.4: C-RAN Architecture.

that connects each RRH to its corresponding BBU pool is the fronthaul link, as shown in the figure 2.4. The C-RAN incorporates cloud computing into the 5G RAN architecture. It is initially built on two primary tenets: the centralization and the virtualization of baseband processing. The centralization of the C-RAN architecture brings several benefits compared to past wireless systems, such as:

- Capital expenditures (CAPEX) and Operating expense (OPEX) reduction of mobile network operators, as the BBUs are centralized and only the RRHs are distributed the deployment and maintenance cost of cell sites can be reduced significantly.
- Energy and spectral efficiency, the resources in the BBUs and RRHs can be utilized effectively as per service demand.
- Flexibility in supporting future radio access technologies, since the centralized BBU design can handle numerous wireless standards, they can be successfully deployed, controlled, and utilized based on the needs of the users.

If taken into account the split between BBU and RRH functions, are obtained two types

of C-RAN [8]:

- A fully centralized C-RAN, where all functions referring to Layer 1, 2 and 3 are present in the BBU. RRH is responsible for the Radio frequency (RF) functions. Although this type benefits from the advantages mentioned above it suffers from two major problems: the high bandwidth requirements and the timing requirement of transmission signals between RRH and BBU.
- A partially centralized C-RAN, where Layer 1 related functions are moved from the BBU to the RRH. This architecture requires a low transmission bandwidth between RRH and BBU compared with the fully centralized option. Even moving the baseband processing from the BBU to the RRH, the low flexibility in network upgrades and less convenience for multi-cell collaborative signal processing remains a problem.

2.3.2 Next Generation - Radio Access Network

The NG RAN represents the newly defined radio access network for 5G. Based on the study on scenarios and requirements of 5G use cases a set of prerequisites for the NG-RAN architecture and the migration of NG RATs has been established by the 3GPP RAN working groups. [9] The NG-RAN architecture must allow efficient interworking between NG RATs and LTE, connection through various transmission points, flexible deployment and functional split options, as well as network slicing and NFV.

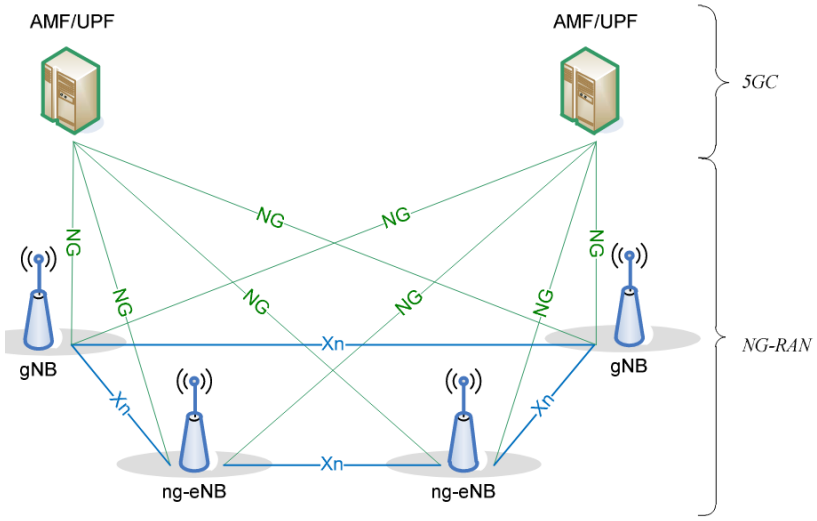


Figure 2.5: NG-RAN overall architecture (retrieved from [6]).

The NG-RAN consists of a collection of base stations, as seen in the architecture of figure 2.5 interconnected by the Xn interface and connected to the 5GC via the NG interface. The coexistence of 4G and 5G UE the need for two different types of nodes arises: 5G Node B (gNB) and Next-generation Evolved Node B (ng-eNB), provide different air interface accesses – LTE and NR, respectively. This is different compared to previous RANs generations, like Evolved Universal Terrestrial Radio Access Network (E-UTRAN), where a RAN only supports one access technology. E-UTRAN is an access network which together with EPC, form an LTE network, the main technology used in 4G. Being a no centralized controller, consists in only

base stations (eNode B) like the NG-RAN. The gNB provides 5G user and control plane protocol terminations toward the UE, while the ng-eNB provides 4G protocol terminations. Although the 4G air interface is used, the UE still needs to implement 5G protocols in order to interoperate with 5GC.

In a high-level view the major functional elements presented by the 3GPP that the NG-RAN nodes perform are the following [6]:

- Radio Resource Management: like Radio Bearer and Admission Control or Connection Mobility Control;
- Dynamic allocation of resources to UEs;
- Routing of User and Control Plane;
- Support of Network Slicing;
- Data compression, encryption, and integrity protection using IP headers;
- Scheduling and transmission of paging messages and broadcast information.

2.3.3 5G - New Radio

As seen earlier, 5G will not focus on just one RAT, instead it will be a collection of RATs that will include enhancements of current technologies with new ones. As such, in the first phase, the most economical approach would be to invest in the improvement of existing RATs, as is the case with LTE. Although this alternative solves some of the problems that 5G tries to address, it does not meet all the requirements and a new solution is needed. Key elements such as the efficient utilization of radio spectrum at higher frequency bands is fundamental component that will enable the exponential increase of connections and usage. [10]

NR is the new radio interface and radio access technology for cellular networks specified by 3GPP [11]. It will improve the speed and responsiveness of mobile broadband experiences, as well as extend mobile technology to integrate and redefine a variety of new verticals. The 5G NR specification defines how UE and NR network infrastructure, like RANs, wirelessly transmit data using electromagnetic radio waves. 5G NR will allow the use of two operating bands, from 450 MHz to 6GHz and from 24.25GHz up to 52.6 GHz. High network capacity and extreme data rates will be possible due to the high frequencies, mmWave, with the transmission bandwidths associated with that wider spectrum. Lower frequency bands, sub-6 spectrum, will continue to be useful, especially with technologies such as beamforming and Massive Multiple Input Multiple Output (MIMO) that reduce range and obstacle penetration restrictions. [12]

2.3.4 NG-RAN (gNB) Architecture

Earlier it was defined that the NG-RAN consists of a set of gNBs connected to the 5GC. The suggested structure for gNB is based on a centralized RAN architecture as discussed in section 2.3.1. The gNB is disaggregated into two parts, where lower level functions are implemented in the DU and the remaining higher layer functions are implemented in the CU. As can be seen in figure 2.6, the gNB then consists of one gNB-CU and several gNB-DUs. The maximum number of gNB-DUs that can be connected to a gNB-CU is only limited by

the implementation. The gNB-CU and associated gNB-DUs are only visible to other gNBs, 5GC does not know the internal structure of a gNB.

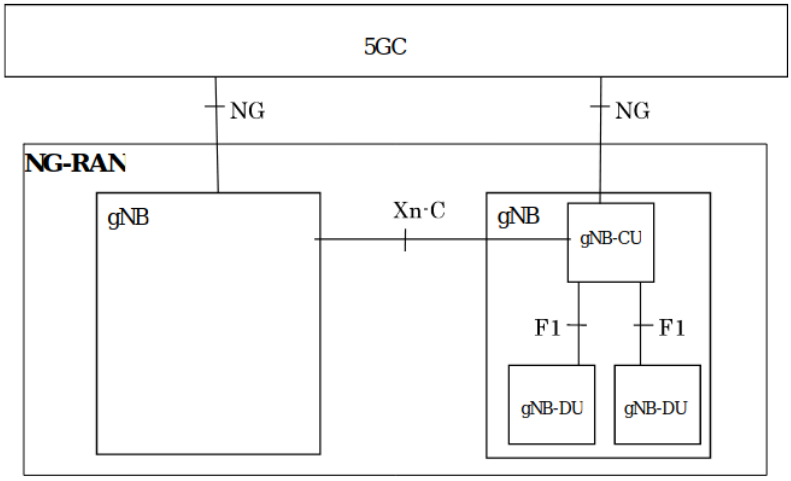


Figure 2.6: NG-RAN - gNB architecture (retrieved from [9]).

In contrast to the previous C-RAN architecture with fixed functional splits, the architecture presented in the NG-RAN specifications, in addition, allows an extended configuration of the functional split. This split enables the adaptation of the RAN to various use cases, such as variable transport latency. The interface that links the gNB-CU to the gNB-DU is called F1. The F1 is a point-to-point interface that supports data transmission and signalling exchange. F1 interface supports the separation of the control plane and user plane. The interface also separates Radio Network Layer (RNL) and Transport Network Layer (TNL). [13]

It is important to note that 3GPP in its specifications only considers the split architecture of the base station containing CU and DU. Currently, the most used split architecture is the separation into three logical nodes: CU, DU and RRH or RU [14] [15]. As shown in figure 2.7, the functions that in the C-RAN architecture belonged to the BBU in the new architecture are moved to all three units. The CU provides high-layer protocol stack functions, while the DU provides low-layer protocol stack functions. The RU provides low layer functions, for instance, function part of the Physical layer (PHY). Alongside this division are the links that connect the 5GC to the CU, the CU to the DU, and the DU to the RU, which are the backhaul, midhaul, and fronthaul, respectively.

As a follow-up to this type of architecture, 3GPP standardized a number of split options enumerated in Figure 2.8. These split options will define where functions will be centralized along with the network for both midhaul and fronthaul. Generally, Lower layer splits (LLS) offer higher gains in terms of performance and resource utilization. Unfortunately, comes at a cost of increased complexity in terms of standardization, implementation, and interoperability of the fronthaul interface. Lower splits can also significantly increase fronthaul requirements in terms of latency and throughput, to the point where some splits are no longer practical, figure 2.9. The split option 8 is the fronthaul split present in the traditional c-ran, a continuous bitrate transport is required regardless of whether user traffic is present or not. In the

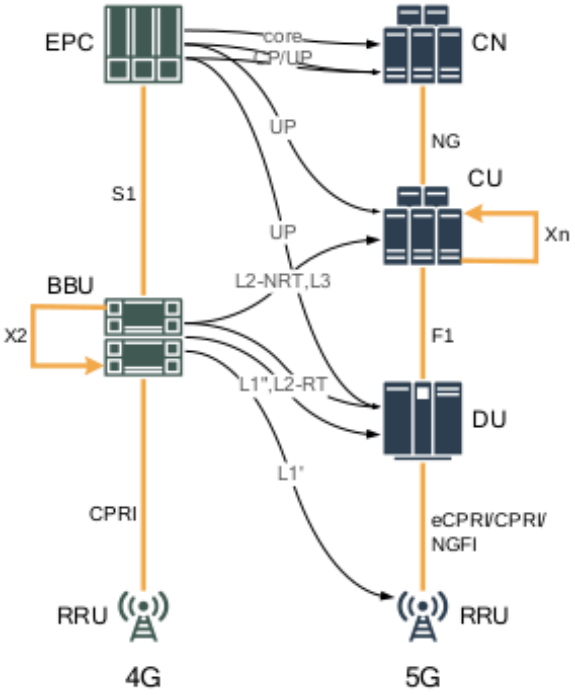


Figure 2.7: Evolution to a split function architecture from a traditional C-RAN architecture (retrieved from [14]).

remaining options, the amount of data to be transported varies with user traffic. The 3GPP have selected the option 2 , Packet data convergence protocol (PDCP) / high Radio link control (RLC), as High layer split (HLS) option for midhaul. Thus, it leaves open the choice of the split option for the fronthaul.

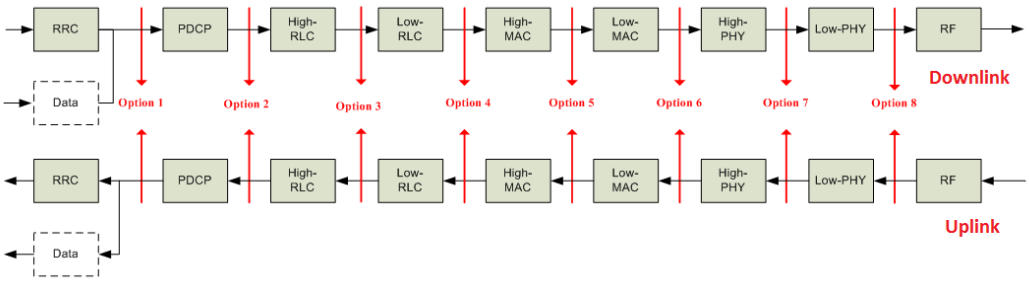


Figure 2.8: Function Split options. (retrieved from [16]).

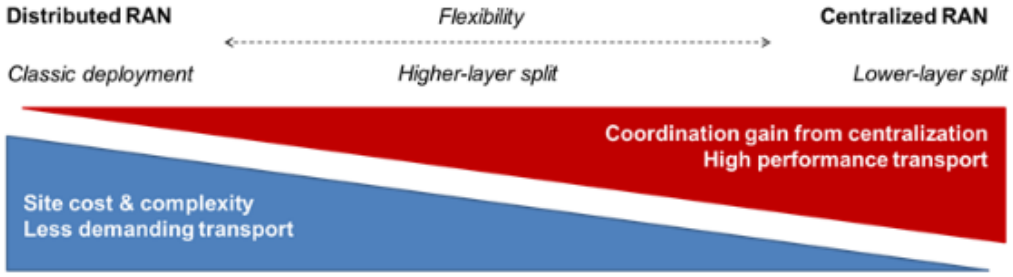


Figure 2.9: Comparison C-RAN Functional Splits (retrieved from [17]).

2.4 Open Radio Access Network

2.4.1 O-RAN Alliance

O-RAN ALLIANCE was founded in 2018 by a world-wide consortium of mobile network operators with the purpose of evolve radio access networks. Currently the community brings together vendors, and research academic institutions and aims to enable a more competitive and vibrant RAN based on more intelligent, open, virtualized O-RAN standards. O-RAN ALLIANCE is active in three main areas: Specification effort, O-RAN Software Community and Testing and integration effort. The development on the O-RAN specification has been divided into technical workgroups, the rest of this chapter will be based on WG4: The Open Fronthaul Interfaces Workgroup. The focus of this group is to provide truly open interfaces in order to enable multi-vendor DU-RU interoperability. [18]

2.4.2 O-RAN Overview

The O-RAN concept arises from the goal of having a network capable of supporting interoperation between vendor's RAN equipment. According to the O-RAN Alliance, the O-RAN intends to evolve the RAN to a network more software-driven, virtualized, flexible, and energy efficient. Its core principles are openness and intelligence. Openness is required to create a more cost-effective and agile RAN. Open interfaces both enable smaller vendors to launch their own services faster and operators customize their network according to their needs. As with software this allows multi vendor deployments, making the supplier market more competitive and dynamic. [19] [20]

The diversification of richer and more demanding applications are making 5G networks increasingly complex. As such, virtualization and the RAN split have added even more complexity. To constrain this complexity, it is necessary to find other ways to deploy, optimize and operate a network. By introducing intelligence in RAN, the O-RAN Alliance will enable deep learning techniques and automated management and control by using Artificial intelligence (AI) and Machine learning (ML).

There are three benefits that can be identified with regards to O-RAN [21]:

- **Reduce network CAPEX and OPEX** - the O-RAN is able to reduce CAPEX by introducing open interfaces and facilitating interoperability between multi-vendor equipment. This creates a more scalable and competitive market, along with open source

software and hardware reference designs that accelerate the processes. Operators can select products from any supplier and make the best choice from a cost perspective as well. An automated and virtualized RAN contributes to reducing OPEX. Intelligent components are less maintenance-intensive.

- **Enable network efficiency and performance** - It is possible to perform automated interventions as the network is constantly monitoring performance and resources. O-RAN is able to offer, even in complex networks, optimized and efficient radio resource management in order to increase network performance and upgrade user experience.
- **Upgrade with great agility** - with its native cloud infrastructures, the O-RAN makes it simple to add new network capabilities through simple software upgrades.

2.4.3 O-RAN Architecture

The appearance of split architectures, such as the NG-RAN with the split between CU and DU, have brought flexibility to the network. Such flexibility allows the hardware and software that runs on them to make networks scalable and cost-effective deployments, but for this there needs to be interoperability between components. Such interoperability is achieved with the help of the open interfaces between CU, DU and RU.

The O-RAN Reference Architecture is designed to enable NG-RAN infrastructures. In order to fulfill the requirements discussed above and maintain its principles, the goal is to be a base in order to create a virtualized RAN (vRAN) on top of open hardware. The architecture is based on well-defined, standardized open interfaces to support and complement standards developed by groups like 3GPP. [22] The high-level view of the O-RAN architecture makes it possible to divide the system into three parts: the network functions, a Service Management and Orchestration (SMO) framework to handle the network functions, and an O-RAN Cloud (O-Cloud) to host the cloud network functions. VNFs (Virtualized Network Functions) can be an example for O-RAN network functions, for instance, Virtual Machines or Containers. [22] In figure 2.10 can be analyzed the O-RAN architecture from a more logical perspective. By analyzing this new architecture, which may correspond to a gNB as seen in the section on NG-RAN, it is possible to identify the new components that compose it:

- **Service Management & Orchestrator (SMO)**: The component in charge of orchestration, management, and automation of RAN components. It supports O1, A1 and O2 interfaces.
- **Non-Real Time (RT) RAN Intelligent Controller (RIC)** : Non-RT RIC is the functionality internal to the SMO in the architecture that provides the A1 interface to the Near-RT RIC. The objective is to assist intelligent RAN optimization by giving policy-based guidance, ML model management and enrichment data to the near-RT RIC function, allowing optimization of the RAN. Performs functions in non-real-time interval, in other words, in more than 1 second.
- **Near-RT RIC**: This controller adds additional functions that take use of embedded intelligence, such as QoS management, connection management, and seamless handover control. The Near-RT RIC can be viewed as a robust, secure, and scalable platform that

enables near-real-time control and optimization of O-RAN elements and resources. It comprises SMO policy interpretation and enforcement, as well as enrichment information to enhance control function. Being a Near-RT Controller operates functions in less than 1 second.

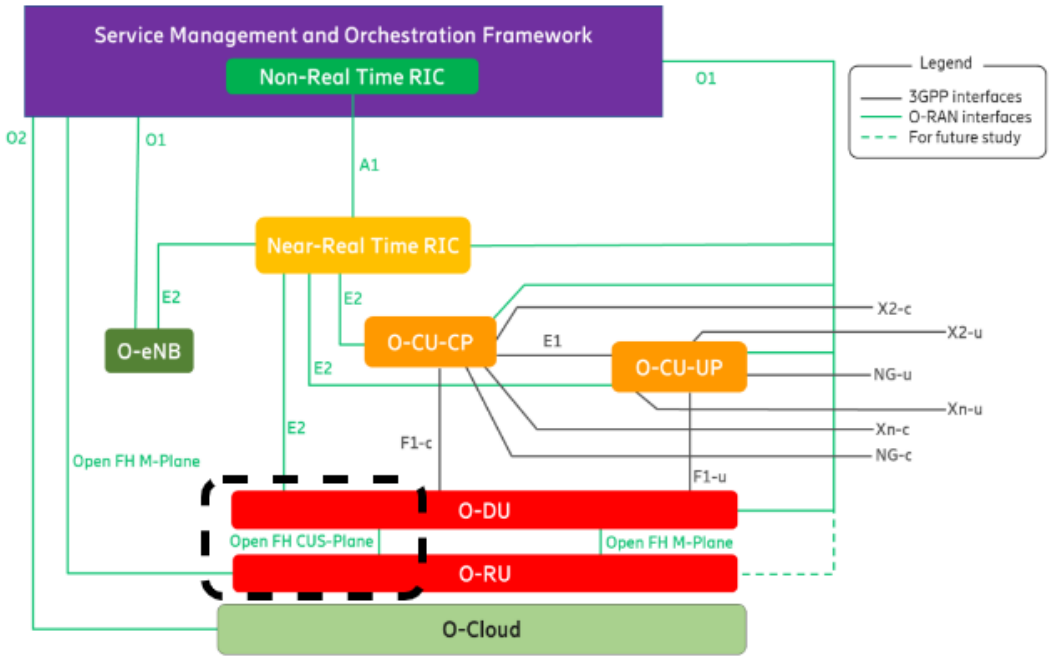


Figure 2.10: O-RAN Architecture (retrieved from [22]).

Besides these new components that are the drivers of automation, intelligence and control of RAN, the old units continue to exist with some changes. Now, instead of just one O-RAN Control Unit (O-CU), it has been vertically split and there is a separation between the control plane and the user plane. Resulting in O-RAN Central Unit – Control Plane (O-CU-CP) and O-RAN Central Unit – User Plane (O-CU-UP), responsible for high layer functions, have seen interfaces improved in order to support interoperability. The near-RT RIC module now has some control over how basic functions are implemented. Finally, there was also an opening in the Open RU (O-RU) and Open DU (O-DU) interfaces, giving rise to open fronthaul. Unlike the other architectures that were analyzed in this document, in O-RAN there is no longer just one fronthaul link, but a division among the various planes existing in the protocol stack. There is the open fronthaul M-Plane and the open fronthaul Control, User and Synchronization Plane (CUS-Plane), the latter will be analyzed in more detail since it is the key link in this dissertation.

2.4.3.1 Open Fronthaul M-Plane

The M-Plane split facilitates a variety of O-RU initialization, configuration and management functions to set parameters and support the functional split. This is a way to eliminate dependence on vendor's implementation. The O-DU and Network Management System (NMS) are used in M-Plane to manage the O-RUs, by using Network Configuration Protocol

(NETCONF). NMS and O-DU are NETCONF clients while O-RU is a NETCONF server. The O-RAN architecture supports two types of configuration models: Hierarchical and Hybrid model. In the hierarchical model one O-RU is managed by O-DUs, that handle the monitoring and control. In this model the NMS has a reduced processing load and does not require it to support NETCONF. On the other hand in the hybrid model the O-RU is managed by both NMSs and O-DUs. This architecture has the benefit of allowing NMSs to monitor/manage other network devices in addition to O-RUs, allowing for standard maintenance, monitoring, and control of all equipment. [23]

2.5 Open Fronthaul

Most 5G applications require low-latency, making the fronthaul bandwidth an important issue. As a result, a new open protocol that can deliver these data rates was introduced. The Common Public Radio Interface (CPRI) used in traditional C-RAN is no longer a viable option to meet the specifications of the fronthaul. In O-RAN deployments, there are two transport options that are available for implementation of fronthaul for DUs and RUs: Enhanced CPRI (eCPRI) and Radio over Ethernet (RoE). The CPRI Forum defined the eCPRI as a packet-based fronthaul protocol to enable efficient and flexible radio data transmission via a packet based fronthaul transport network like Ethernet. It provides an interface to transport upper layers of the protocol stack. This protocol delivers higher data rates for 5G by utilizing data compression techniques for improved fronthaul. [24] The IEEE 1914.3 working group [2] have standardized the RoE protocol. Like the eCPRI encapsulate and map radio protocols over Ethernet frames.

2.5.1 Split Option 7-2x

In 5G wider frequency bandwidths and higher number of antennas due to MIMO are causing the required fronthaul transmission bandwidth to increase. In order to solve this problem it was necessary to revisit functional splitting. According to O-RAN specifications [25], there are two points that must be taken into account:

- There is a benefit in keeping an O-RU as simple as possible because size, weight and power draw. These should be the main points to think about;
- It is advantageous to have an interface at a higher level that tends to reduce the interface throughput relative to a lower level, but the higher the interface level, the more complex the O-RU.

With this in mind, O-RAN has selected a single split point, called Split Option 7-2x that places in the radio unit some Layer 1 functions traditionally located in the baseband processing side. An overview of Split Option 7-2x and some additional tradeoffs is shown in Figure 2.11.

This functional splitting between O-DU and O-RU divides the function of the PHY Layer named as High PHY resides in O-DU and Low PHY resides in O-RU. In the DL, the functions

[2] https://sagroups.ieee.org/1914/p1914-3/

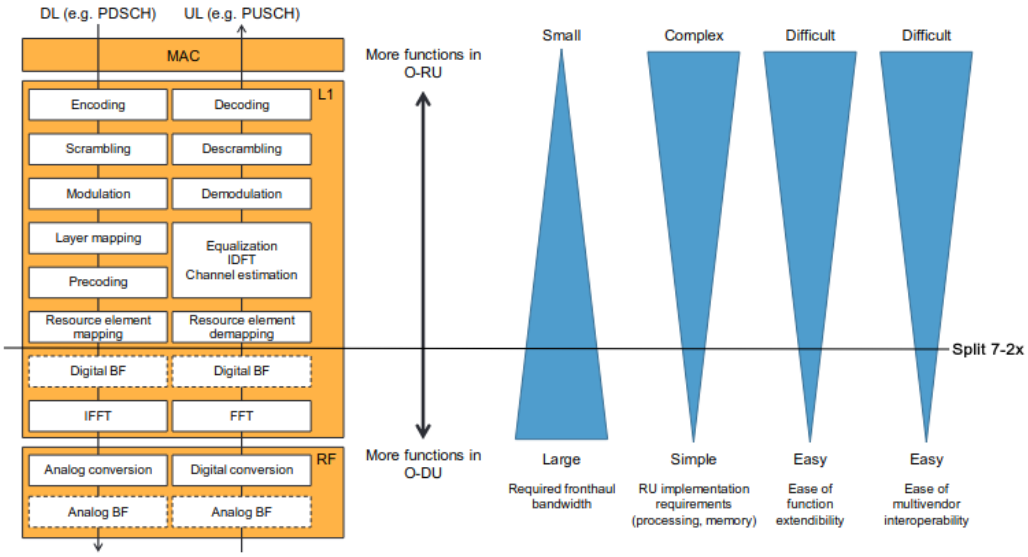


Figure 2.11: O-DU and O-RU functional split tradeoffs (adapted from [23]).

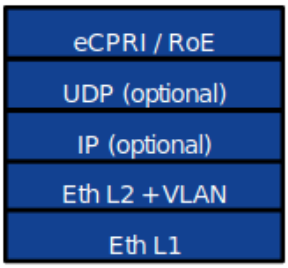
of encoding, scrambling, modulation, layer mapping, precoding and resource element mapping reside in the O-RU; these functions result in an IQ sampling sequence. The rest of low-PHY functions are performed in the O-RU. In the UL, the de-mapping, equalization, de-modulation, de-scrambling, rate de-matching and de-coding reside in the O-DU. Low-PHY functions like Fast Fourier transform (FFT) and digital beamforming are located in the O-RU. [25]

Besides the lower O-RU complexity and interoperability that the split enables. This split option benefits from the simplicity of interfaces as it simplifies the data mapping and limits that the control messages require and since the split is symmetrical the specifications are reduced. As mentioned earlier one of the benefits of O-RAN is virtualization, making it easy to introduce new features through upgrades. Placing most functions at O-DU will facilitate this by becoming a RAN future-proof. These features apply to both the UL and the DL. [25]

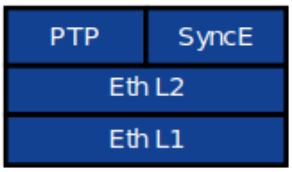
2.5.2 Protocol Stacks - User, Control and Synchronization Plane

The O-RAN fronthaul specification anticipates how an O-DU will interact with an O-RU. Besides the split option and M-Plane O-RAN fronthaul describes CUS-Plane specifications. [25] [23]

- **C-Plane** messages define the scheduling and coordination needed for transferring data, including aspects of beamforming. Due to the very strict delay constraints, it is not possible to have message acknowledgements. Both eCPRI or IEEE 1914.3 encapsulate these messages into ethernet frames. After the first encapsulation identifies fields such as the message type, there is a second layer that contains the fields required for control and synchronization. The frame format for a Control Plane (C-Plane) message is shown in figure 2.12a. Information such as the message type, payload size, message source and destination identifiers, and message sequence number are all included in the eCPRI



(a) C/U-plane protocol structure.



(b) S-plane protocol structure.

Figure 2.12: CUS-plane protocol structure (retrieved from [23]).

header. The eCPRI payload of the C-Plane message passed from the O-DU to O-RU consists of information specifying BF weights, time and frequency resource information which will be applied when transmitting and receiving signals on the radio interface. The transmission header, application header, and sections are all designed to be aligned on 4-byte boundaries and are sent in "network byte order", which means that the most significant byte of a multi-byte parameter is sent first. [25] [23]

- **U-Plane** uses messages for efficient data transfer within strict time limits and the frame format can be seen in the figure 2.12a. This frame format is used in both directions, that is, for transmission from the O-DU to O-RU and transmission from the O-RU to O-DU. As well as C-Plane messages, the User Plane (U-Plane) messages are encapsulated in eCPRI or RoE and do not support acknowledgements. The O-RAN fronthaul specifications prescribe an extended Antenna-Carrier (eAxC) as source and destination identifiers of the message. The eCPRI payload is used to transmit an IQ sample (iSample/qSample) sequence of the OFDM signal applying IQ compression. Together with this information, time/frequency resource information is transmitted. [25] [23]
- **S-Plane** messages are responsible for the timing and sync aspects 2.12b. Protocols such as Precision Time Protocol (PTP) and Synchronous Ethernet (SyncE) are supported by O-RAN fronthaul specifications to deliver high-accuracy synchronization on the O-RU side by synchronizing with the clock on the high-performance O-DU side.

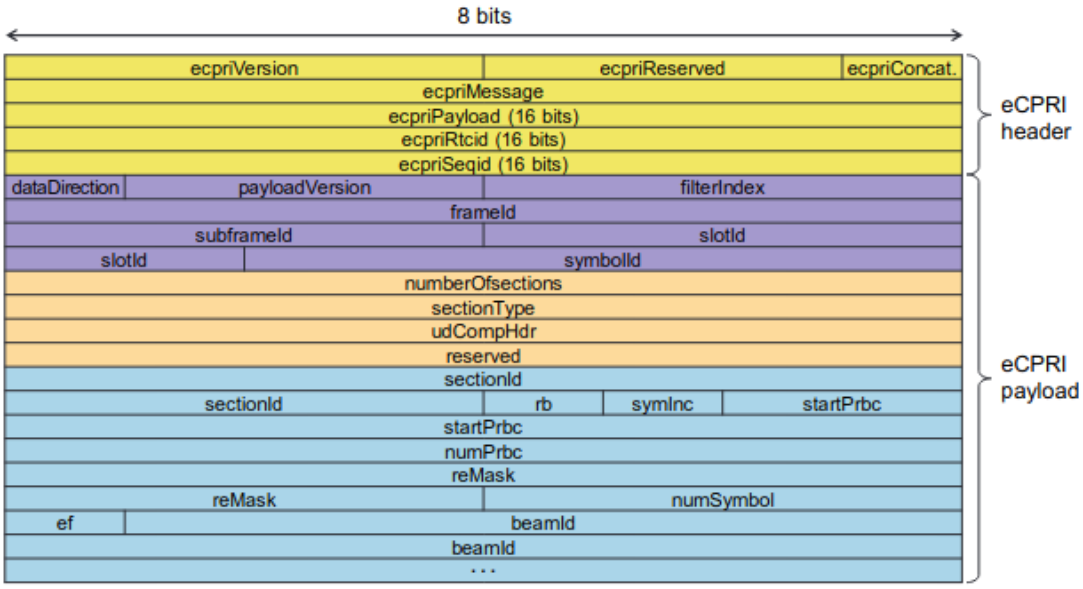


Figure 2.13: Message frame format in C-Plane (retrieved from [23]).

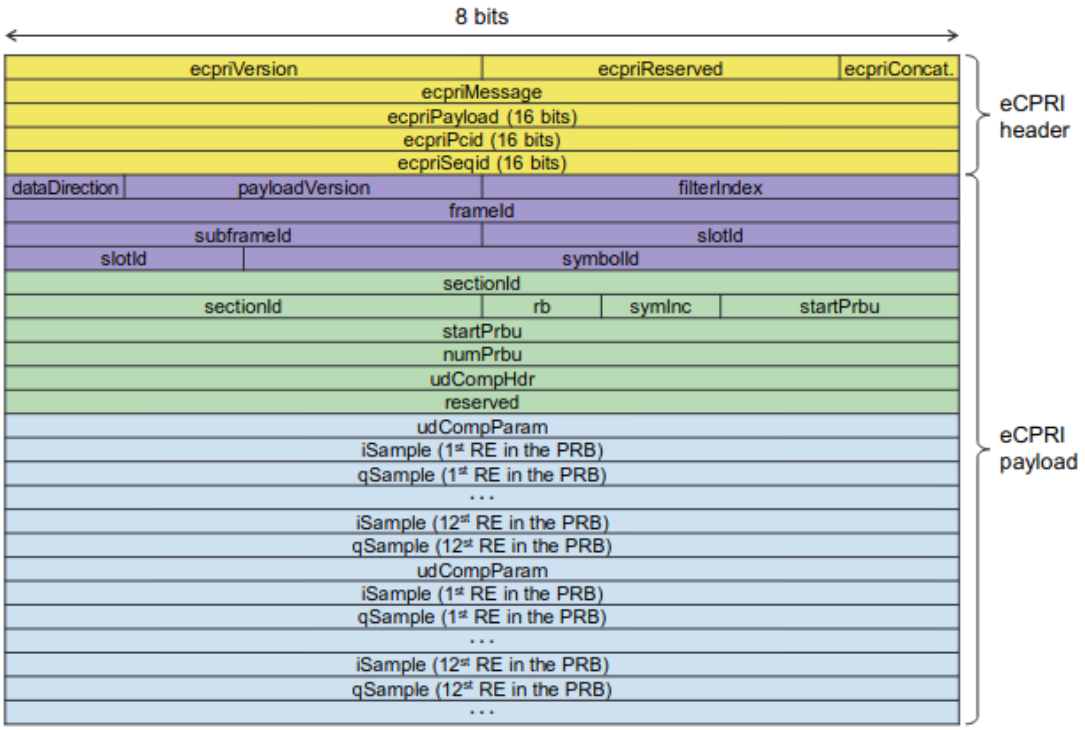


Figure 2.14: Message frame format in U-Plane (retrieved from [23]).

2.5.3 Latency Requirements

The O-RAN fronthaul specifications have characteristics of a stringent bandwidth and tight latency requirement that needs to be supported by the transport network, which may vary according to the operating environment, topology, and use cases. The latency model used is based on reference points defined by eCPRI [24] as shown in figure 2.15. The reference points are:

- **R1** - Transmit interface at O-DU;
- **R2** - Receive interface at O-RU;
- **R3** - Transmit interface at O-RU;
- **R4** - Receive interface at O-DU;
- **Ra** - Antenna interface at O-RU.

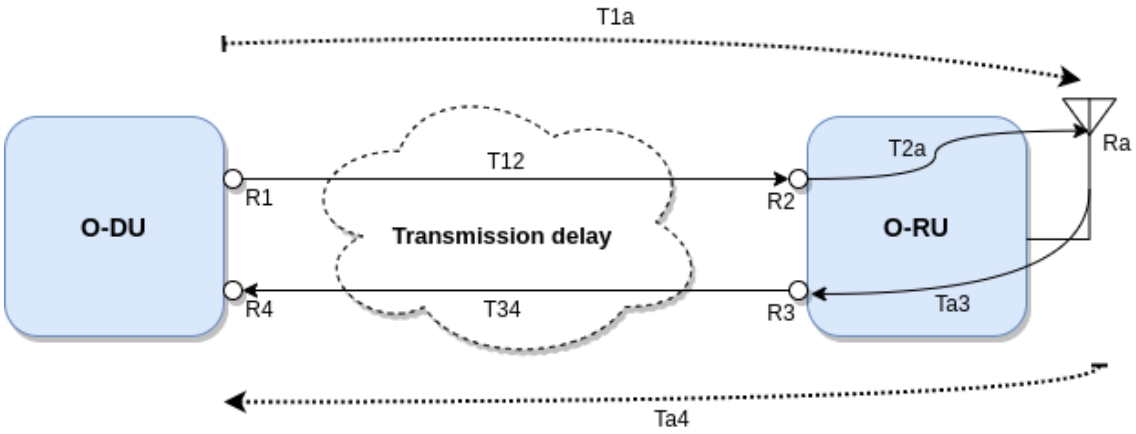


Figure 2.15: Reference points for delay management .

In the figure, T12 on the downlink and T34 on the uplink represents transmission delay between O-DU and O-RU. This transmission delay represents only the time at which a bit is transmitted from the R1 and R3 interface and is received at the other end of the fronthaul link. In the ethernet transport network this link is not constant, on the other hand, fixed time is needed at the Ra interface to use this point as a reference for the rest of the model. Both O-RU and O-DU should use the same timescale with the relative time error needing to be less than 3μs.

In order to ensure that the transmission is carried out successfully it is necessary that relationships between some of the times mentioned are met. Since transmission times are not fixed it is advantageous to look at upper and lower bounds of the transport delays. T12min represents the shortest path a packet can take and T12max the longest path. According to the specifications some factors that affect the variance of these transmission times are transport media rate, air interface bandwidth, and amount of data compression. So the transmitter has a transmission window, the time it has to transmit all the data, defined by the difference between the maximum T1a and the minimum T1a. Similar to the transmission window there

is a reception window that must always be larger than the transmission window plus the transport variation.

Therefore, delay management for O-RAN interfaces is necessary to guarantee the alignment of transmission and reception windows. The O-RAN allows the O-DU to calculate the required transmit and receive windows based on the delay and transport of the O-RU network characteristics. When adopting a computed delay approach as mentioned, O-RAN takes into account two methods: Defined Transport Method and Measured Transport method. In the first method the delay is pre-defaulted by the operator and everything is computed based on those values. In the second case the network delay is estimated based on measurements of packet latency on the downlink and uplink. In the case of the Measured Transport method, the eCPRI specifications have defined the one-way delay measurement message. The purpose of this message is to estimate the transport delay as it varies between a maximum and minimum time value.

CHAPTER 3

Programmable Data Plane

This chapter describes how networks have evolved to adopt the SDN paradigm, exploring the P4 language for data plane programming.

3.1 Software-Defined Networking

The increase of mobile devices and content, server virtualization and cloud services are becoming more popular and it becomes necessary to revisit the conventional network architecture. The traditional architecture made sense when the server-client paradigm was dominant, but a static system is beginning to be a weak point in today's networks. The virtualization allows the implementation of various servers in the same hardware and facilitates the migration to another device for load balancing or in case of machine failure. As a way of facilitating network evolution, the concept of programmable networks has been proposed. In order to simplify network management and facilitate evolution, SDN appear as a new networking paradigm [26]. Since the control plane is physically separated from the data plane, and one control plane controls multiple forwarding devices , this devices can be programmed in different ways. One controller with open interface, such as OpenFlow, control various forwarding devices from different hardware and software vendors. OpenFlow is the first standard interface defined between the control and forwarding planes of an SDN architecture and enables direct access and manipulation of network devices forwarding planes of network equipment.

According to Open Networking Foundation (ONF) [1], SDN is an emerging architecture that decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. SDN centralizes network intelligence logically in software-based controllers, and network equipment are reduced to plain packet forwarding devices that may be configured via an open interface.

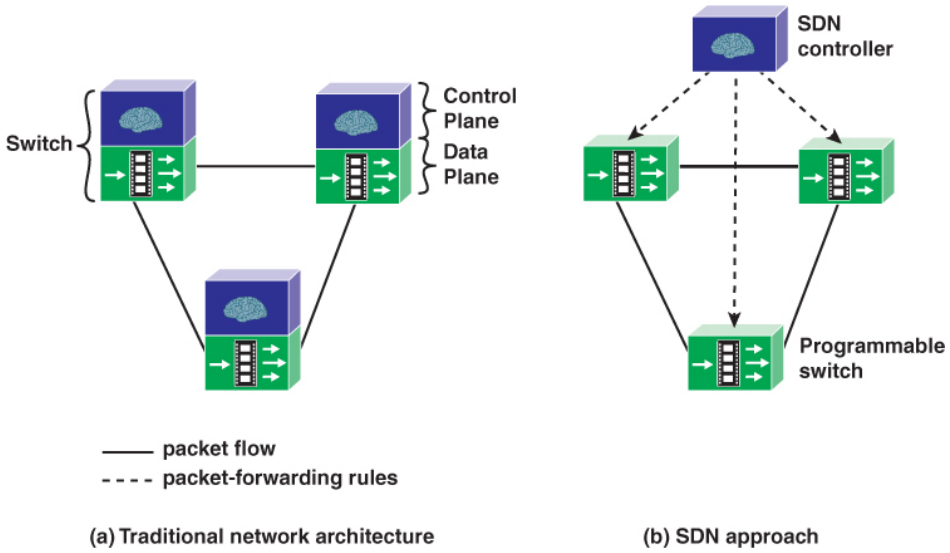


Figure 3.1: Traditional Network vs SDN approach (retrieved from [27]).

3.1.1 SDN Architecture

Traditionally the devices like switches and routers that are interconnected in a network operate as a closed system. With limited interfaces and vendor-specific, it becomes difficult to upgrade networks with this type of device. The concept of SDN emerges to mitigate this problem based on an architecture that splits the switching function between a data plane and a control plane. In figure 3.1 is visualized how the network has evolved, in the SDN the forwarding hardware is separated from the control part, making network virtualization even more affordable. As can be seen, once the control plane is separated from the data plane, only one centralized SDN controller is needed to manage the multiple switches. In the architecture presented in figure 3.2 is analysed, it is possible to see that applications that operate on top of the SDN controllers can program the network since the SDN controller presents an abstraction layer of network resources. [28] [26]

Briefly, the ONF points out five benefits of SDN architecture [28]:

- **Directly programmable** - Since it is separated from the forwarding functions the network control is programmable;
- **Agile** - Administrators can dynamically change network-wide traffic flow to accommodate current demands by abstracting control from forwarding;
- **Centrally managed** - The network becomes intelligent because the software-based SDN controller is centralized maintaining a global view of the network;
- **Programmatically Configured** - SDN enables the configuration, management, security and optimization of network resources in a way that is quick, dynamic and automated;
- **Open standards-based and interoperability between vendors** - Once is based on open standards and interfaces, the network is simplified as it allows interoperability between devices from different vendors.

[1]https://opennetworking.org/sdn-definition/

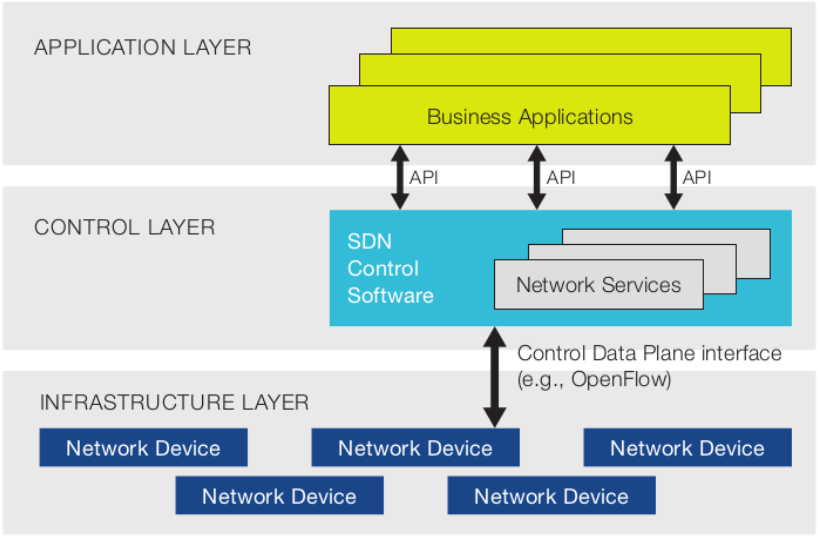


Figure 3.2: Software-Defined Network Architecture (retrieved from [28]).

3.1.1.1 Control Plane

The SDN control section translates application-layer service requests into precise instructions and directives for data plane switches, as well as providing data plane topology and activity information to applications. The Control plane is responsible for functions such as populating the routing table or forwarding table and enabling the data plane functions. The control layer is implemented as an SDN controller, which is a server or a group of servers that work together.

3.1.1.2 Data Plane

The SDN data plane is an infrastructure layer where data is transported and processed by network forwarding devices based on decisions made by the SDN controller. The device supports two types of functions, control support functions are the functions that interact with the SDN controller. The data forwarding functions accept and forward data flow from other network devices according to the rules defined by the SDN applications.

3.2 P4

Following the adoption of SDN appears Programming Protocol-independent Packet Processors (P4) [2]. P4 was first described in [29] as a high-level language that works in parallel with SDN control protocols such as OpenFlow. P4 is an open-source, domain-specific programming language to express how packets are processed by the data plane of a programmable forwarding element such as hardware or software. Initially was designed for programming switches, but now its scope has been extended to cover other network equipment. The abstract model of the language generalizes how packets are processed in different forwarding devices and by different technologies being target-specific and independent. This makes it possible for the

[2] https://p4.org/

target to be a hardware-based device such as an Field Programmable Gate Array (FPGA). Succinctly, the P4 Language Consortium [30] defines three main goals for the language:

- **Reconfigurability** - The controller should be possible to reconfigure the switch to change the way it processes packets;
- **Protocol independence** - The device should not be tied to specific packet formats, the language has no support even for basic protocols like Ethernet or IP. By describing the header formats it is possible to process any type of protocol.
- **Target independence** - P4 programs are designed to be implementation-independent, the programmer does not need to specify the target type. P4 targets can be different types of network infrastructure, such as general-purpose CPUs, FPGAs and system-on-chip.

3.2.1 Architecture Model

The architecture is formed by the blocks like parser, ingress control flow, egress control flow, deparser and data plane interfaces. In order to better understand the P4 program, it is necessary to understand some of the concepts that support it:

- **Headers** - A header describes the sequence and structure of a series of fields, providing names for referencing information. The headers can be divided into packet headers and metadata. Packet headers are extracted from the packets and the metadata contains information about the packet, such as the source port.
- **Parsers** - A parser is a state machine that is used to identify headers and valid header sequences within packets.
- **Tables** - With match+action tables the packets are processed. The extracted header fields are matched in possibly multiple lookup tables.
- **Actions** - Actions can be seen as functions that are executed when populating the table at run time.
- **Control Programs** - The sequence in which match+action tables are applied to a packet is determined by the control program.

3.3 Xilinx SDNet

The Xilinx SDNet high-level environment is a framework that allows the design of packet-processing data planes that target P4 hardware. The tool converts a P4 program into a Xilinx P4 design solution, allowing the programmers to build data planes specifying the header and packet processing. SDNet offers the following features:

- Construction of hierarchical SDNet systems, with various types of engines such as Parsing, Deparsing and Match-Action engines.
- Supports different clocks domains for packet reading, packet processing and control rate for controlling and configuration.
- System supports high-performance hardware implementations up to 200 Gb/s.
- The interface used for packets is the AXI-Stream protocol.
- System backpressure capability is automatically dealt with by the use of buffers. This allows dataflow synchronization at the engines.

3.3.1 SDNet Architecture

Xilinx's SDNet supports a Xilinx P4 architecture based on a P416 specification architecture. This architecture is a pipeline with three customizable engines: Parser, Match-Action Engine and Deparser. In figure 3.3 it is possible to visualize how the engines are interconnected. The main data interfaces provided by SDNet are an AXI-Stream interface for moving packets between engines and the "external" environment and a metadata interface that passes packet related information. The engines and consequently the SDNet IP only contain one input and output packet port. The AXI-Lite memory-mapped ports is used as the control interface.

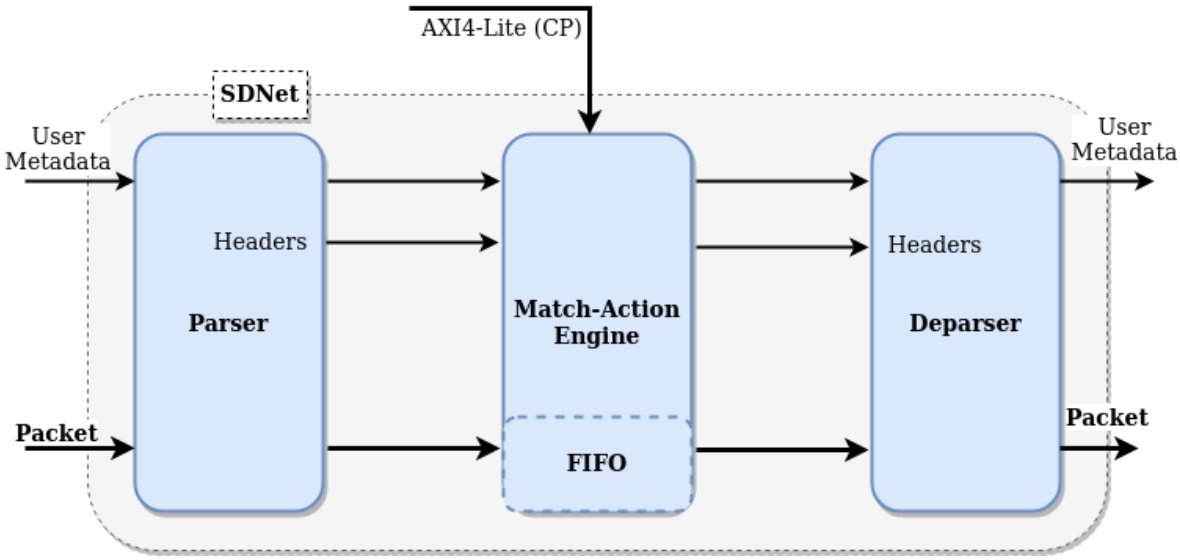


Figure 3.3: Xilinx P4 Top-Level SDNet Design (adapted from SDNet Packet Processor User Guide).

The parsing engine is the first engine in the pipeline and it is used to decode packet headers and extract the information needed. This information can be used later for classification or modification, but the parser can only read the packets and does not have the permission to modify them. Parser supports fixed and variable header sequence and length fields.

Unlike parsers, deparser are used for packet manipulation. This engine cannot read from the packet data bus, but it may insert, change, or remove data from the packet d

The architecture is formed by the blocks like parser, ingress control flow, egress control flow, deparser and data plane interfaces. In order to better understand the P4 program, it is necessary to understand some of the concepts that support it:

atapath. A typical function of the Deparser is to write metadata into packets.

Look-up engines and Actions engines are combined to form Match-action engines. The response of a Look-up engine maps or activates many operations. The Look-up engine response is used to enable only one action engine and store parameters required by actions.

3.3.1.1 SDNet Tool Flow

In order to use the P4 program in hardware from the Vivado, the P4 program is loaded on SDNet. Here SystemVerilog files are generated which are then used for simulation or synthesis

and implementation. In the case of implementation the information is added to the bitstream, used for the hardware configuration on the development board.

SDNet provides not only the Intellectual property (IP') module, but also the external P4C_SDNet Compiler. The compiler takes as input a P4 file and give as an output a JavaScript Object Notation (JSON) file for use by the P4 Behavioral Model. The Behavioral Model builds an independent clone of the operations defined in the source file to compare against expectations and the Register Transfer Level (RTL) implementation. Besides the JSON file it also gets a file about the packets and metadata to simulate.

CHAPTER 4

Proposed Solution and Implementation

In this chapter the architecture of the solution is presented along with its requirements and is described how the switch was implemented according to the proposed solution.

4.1 Scenario

The need for the switch arises from the existence of an O-RAN architecture already installed where it is desired to take advantage of the fronthaul link so that it can be used for other types of traffic. By using the existing fronthaul link it is not necessary to make changes to the infrastructure which would imply costs and construction works that are not intended to be done. As discussed in Chapter 2, the various RUs of an O-RAN architecture are geographically dispersed linked to the remaining components of a centralized RAN component such as the DU. In this way, there are already fronthaul O-RAN links connecting several locations (RUs) to a central location. In this central location, in addition to the DU, there are also servers with other purposes.

In figure 4.1, it can be visualized that in the same location as the RUs there is a gateway to other communication devices that intend to connect to the server existing at the location of the DU. One solution would be the existence of two dedicated links, so there would be no influence of one traffic over the other. The need of sharing the fronthaul link between the O-RAN traffic and the traffic of the remaining components emerges, which during this document is called general-purpose traffic. The general-purpose traffic is essentially data from IoT sensors, not critical and with low latency requirements. These characteristics make it possible to share the fronthaul link, which was initially dedicated exclusively to packets from the different O-RAN planes, with critical traffic and with restricted time windows for transmitting and receiving data. The remaining non-critical traffic can suffer a higher latency

always giving priority to O-RAN packets. The switch is meant to be inserted at both ends of the fronthaul link connecting the RU and the DU.

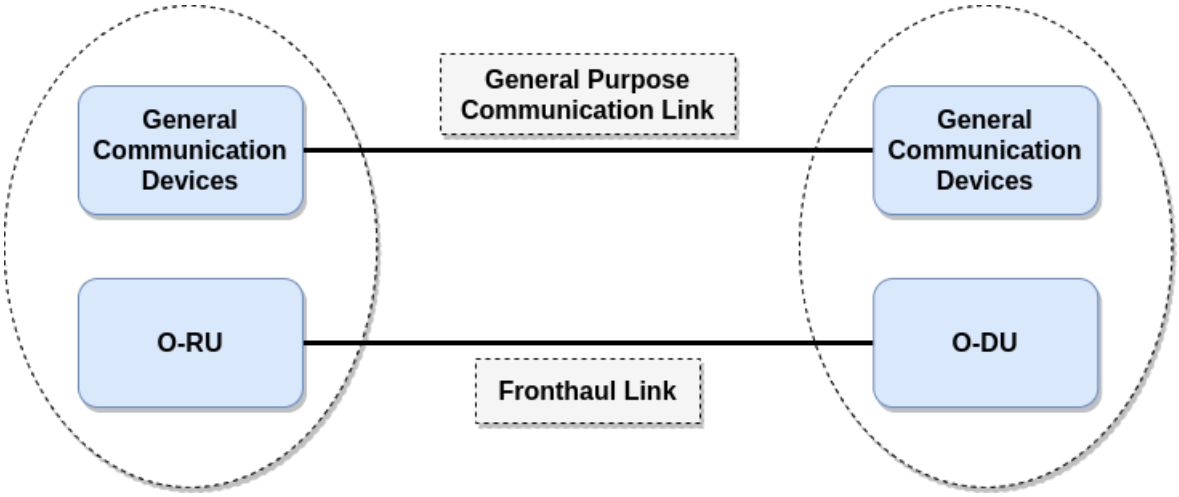


Figure 4.1: Two link scenario.

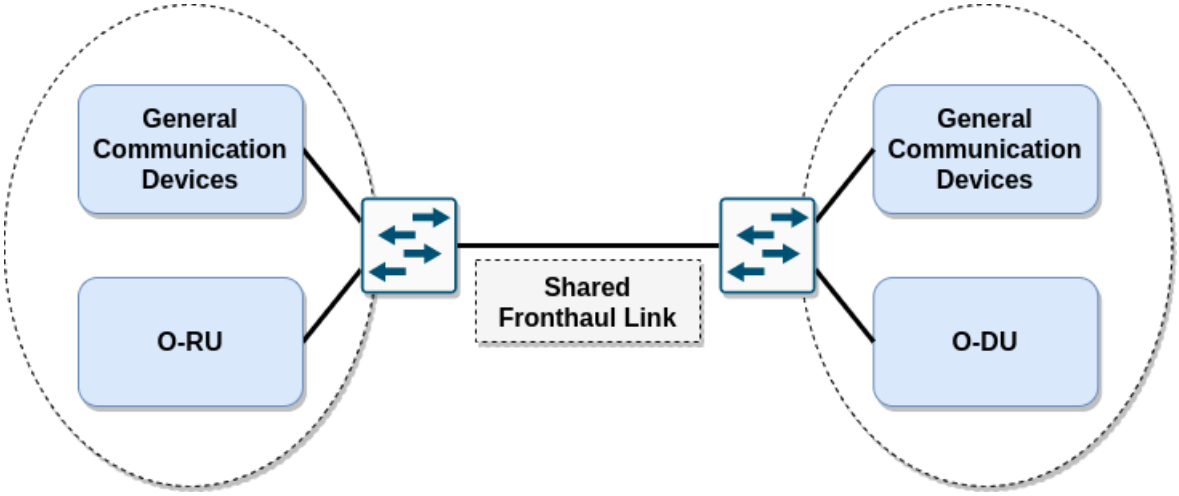


Figure 4.2: Shared fronthaul link scenario.

4.2 Requirements

Looking closely at what the switch consists of, the objective of the switch is to combine in one optical link general-purpose traffic with relaxed latency requirements and high priority traffic between the RU and the DU of O-RAN. In the other direction the switch should separate this same traffic by forwarding to each specific port. Figure 4.3 illustrates how packet forwading is done between ports. As the switch will be at both ends of the fronthaul instead of adopting concepts like UL and DL when addressing the direction of traffic it refers to direction FH when packets are received through the fronthaul link and direction O-U when they are received on the two other interfaces. It is important to note that although in this specific scenario there is no packet exchange between the general purpose interface and the interface connected

to the O-RU, the switch should be able to exchange packets between all interfaces if desired. The switch needs to meet the following requirements:

- Must provide 3 ports, two 10 Gbps and one Gbps. One of the 10 Gbps ports is design to connect with the fronthaul link, the other 10 Gbps port is connected to 5G O-DU/O-RU and at last the Gbps port is available for the general-purpose equipment;
- In the direction "O-U" of traffic the switch must be able to always give priority to O-RAN packets (CUS-Plane);
- In the direction "FH" the switch has to separate O-RAN and general purpose packets and forward them to the correct interfaces;
- Due to the latency requirements of the O-RAN, the switch latency should be the lowest in both directions.
- It should only work in Layer 2 and it is not expected to have a control plane as in a normal architecture.

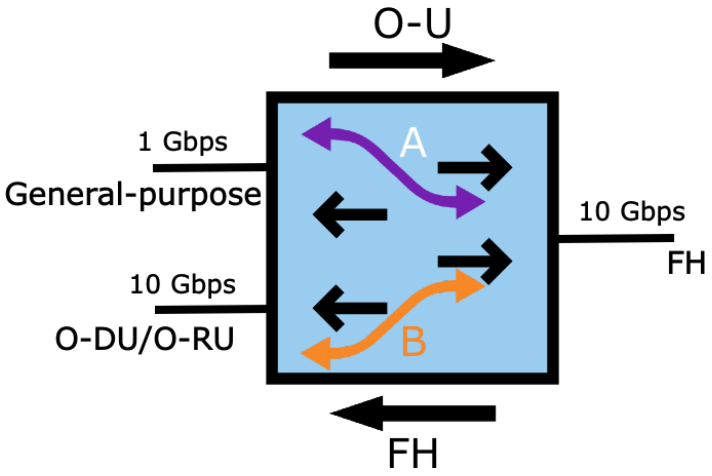


Figure 4.3: 3-port switch and traffic forwarding.

4.3 Architecture

Figure 6.2 presents a high level architecture of the proposed switch. This architecture was based on the NetFPGAReference Pipeline [1], the main blocks are the reception and transmission interfaces, the arbiters and the entity that will perform the operations on the packets. Between all blocks there has to exist a complete exchange of frames so that each component can perform its function.

The Transmit (TX) and Receive (RX) modules are the only ones that provide an interface to the switch with the rest of the network. Having ports at different speeds forces the switch to have two types of modules that are capable of receiving and transmitting packets over an optical

[1] https://github.com/NetFPGA/NetFPGA-public

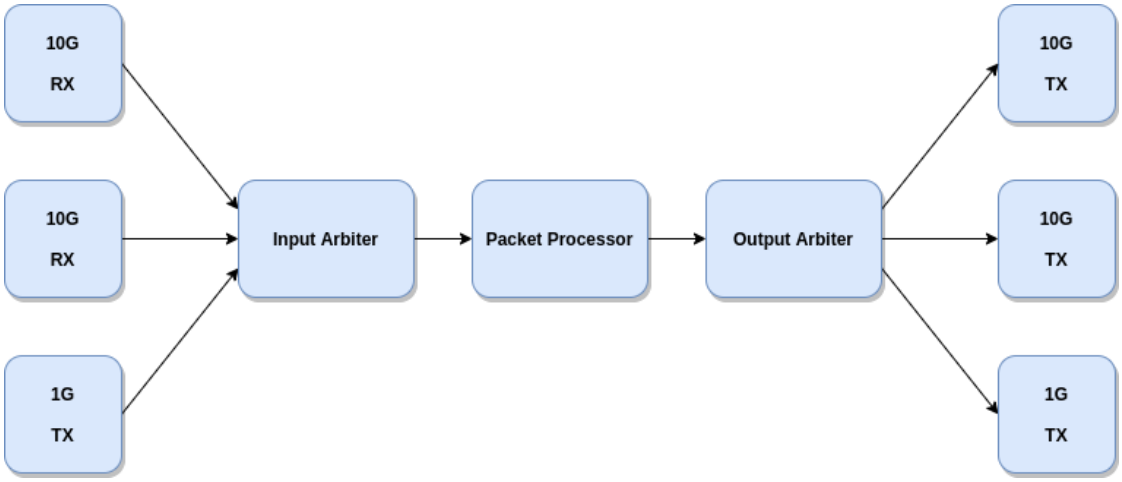


Figure 4.4: Switch high level architecture.

link at the required speed. Combined with the transceivers present in the FPGA, this block is capable of performing Layer 1 and Layer 2 functions (Ethernet Media Access Controller (MAC) and Physical Coding Sublayer (PCS) / Physical Medium Attachment (PMA)), in order to provide an Ethernet frame to the next module.

The various RX and TX modules connect to the input arbiter module. The main task of the input arbiter is the aggregation of the various packets in a single interface, in order to be forwarded to the Packet Processor. As the processing in the following module can be time-consuming, it must be ensured that O-RAN packets are handle first by the input arbiter.

The packet processor is responsible for deciding which port the packet should be forwarded to. For this, the packets must be parsed and according to the information in the headers and the source port. After the packet has been processed is handed to the output arbiter. The output arbiter has the task of receiving the traffic and the information to which port it should be transmitted and forward each packet to the correct location.

4.4 Setup and Laboratory Tools

Hardware and software tools were required for the development of the switch. The resources used were mainly from Xilinx.

4.4.1 Development Board

Taking into consideration the necessary requirements of the switch and the proposed architecture in section 4.3, a Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit [4] was used. The kit is based on Zynq UltraScale+ MPSoC, including as main elements for this work fourenhanced small form-factor pluggable (SFP+) interfaces for Ethernet, two FPGA Mezzanine Card (FMC) interfaces for function expansion, six 16.3Gb/s GTH transceivers and an RJ45 Ethernet connector. In addition to these advantages, having an FPGA with a considerable amount of programmable resources makes the system design more flexible. Since the GTH transceiver supports line rates from 500Mbps up to 16.375 Gbps, it satisfies switch

requirements such as having interfaces with a data rate of 1 Gbps and 10 Gbps. In figure 4.5 it is possible to visualize the kit with some of its components highlighted.

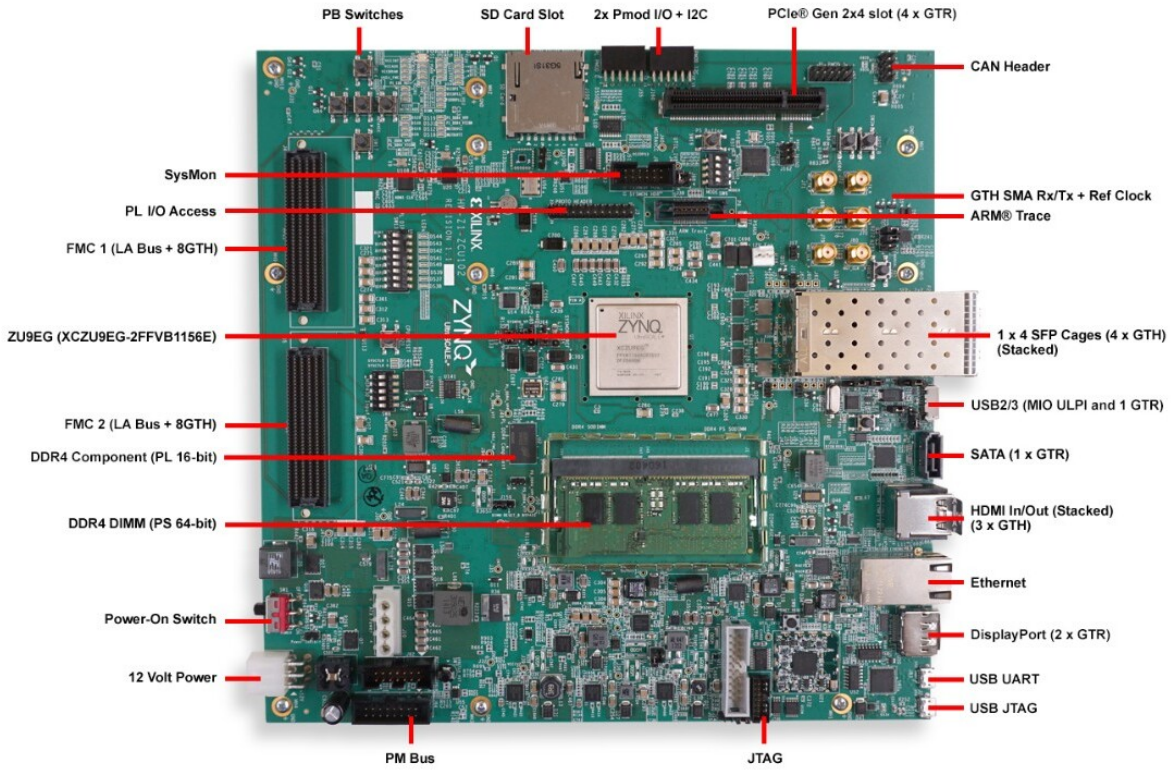
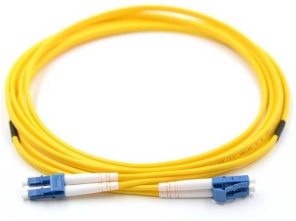


Figure 4.5: Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit. (retrieved from [31])

In order to interconnect the FPGA with the other elements of a network it was used Single Mode Duplex Fiber Cables 4.6a connected to an SFP+ 4.6b, in order to have 10 Gbps port. In regard to the Gigabit port a Category 6 Ethernet cable 4.6c was used. Small form-factor pluggable (SFP) is a hot-pluggable optical module transceiver design to provide high speed and physical compactness. Typically used in telecommunication and data communications applications, allows electro-optical or fiber optic networks to be repaired and upgraded with only one module change. The single mode duplex fiber cable enables reliable, high-speed data transmission over long distances. A duplex fiber cable has two fibers that can transmit data in both directions at the same time. Category 6 cable has a lower bandwidth and transfer rates than fiber cable and is designed for Gigabit Ethernet. It is compatible with fast Ethernet standards 10BASE-T and has RJ45 connectors at both ends.



(a) Single Mode Duplex Fiber Cable.

(b) SFP+.

(c) Category 6 Ethernet cable.

Figure 4.6: Network equipment.

4.4.2 Software Tools

Besides the hardware presented, the project was developed in Xilix's development environments - Vivado Design Suite and Vitis unified software platform. The Vivado Design Suite allows hardware development based on a block design consisting of IP colors or Hardware Description languages (HDLs) such as Verilog and Very High-Speed Integrated Circuit Hardware Description Language (VHDL). Vivado implementation incorporates the steps necessary to place and route the netlist onto device resources, within the logical, physical, and timing constraints of the design. The Vitis platform supports embedded software development for MultiProcessor System On Chip (MPSoC) and soft-core microprocessors.

In addition to these platforms, it was also necessary to use the Xilinx System Controller Graphical user interface (GUI), which allows the reading and configuration of voltage, power, and frequency clocks. In this way, it is possible to make the necessary calibrations so that the actual board parameters match the desired ones.

4.4.2.1 Integrated Logic Analyzer

Besides the IP cores supplied by Xilinx and the modules created in VHDL, the Integrated Logic Analyzer (ILA) module was essential in the validation and debugging of the developed system [32]. The ILA is logic analyzer that can be used to monitor a design's internal signals. The module can have multiple probe ports, which can be combined into a single trigger condition. After a trigger condition occurs, the module buffers are loaded and information about the desired signals are sent through an auto-instantiated debug core hub that connects to the JTAG interface of the FPGA. Thus, it is possible to view the data using Vivado's waveform window. ILAs were constantly added and removed from the system in order to analyse various interfaces in error discovery.

4.5 Implementation

Figure 4.7 shows how the architecture proposed in section 4.3 was implemented. Abstracting the control path and focusing only on the datapath it is possible visualize through the block diagram the traffic flow through the switch. The links between blocks represent the exchange of Ethernet frames, and the most commonly used interface is Advanced Microcontroller Bus

Architecture (AMBA) Advanced eXtensible Interface (AXI)4-Stream. The datapath can be divided according to the different clock domains, one for each type of port and one for packet processing. The various clock domains exist because the switch has ports at different data rates and packet processing is done at the highest possible frequency, in order to minimize latency.

The input and output arbiters that were presented in the proposed architecture in section 4.3 , when it comes to implementation, not only do they perform the mentioned functions of aggregation and forwarding of the packets, but also play a central role in Clock Domain Crossing (CDC) from an implementation point of view. In the 1G clock domain is essentially composed of Zynq UltraScale+ MPSoC which corresponds to the Processing system (PS) side of the ZCU102 board. The PS-side has a Gigabit Ethernet MAC (GEM) responsible for receiving and transmitting Ethernet frames. Besides this main component, there are also auxiliary blocks in this clock domain to facilitate and allow the exchange of frames between IP' cores, which will be discussed in more detail in section 4.5.2.

However, the 10G clock domain is based on the same concept of having a module that sends and receives Ethernet frames as in the 1G domain. Since the desired data rate is different and instead of making use of the RJ45 port it is intended to use the interfaces for SFP+, the 10G/25G High Speed Ethernet Subsystem is used. Unlike GEM, which is on the PS-side, this Ethernet subsystem is implemented on the FPGA Programmable logic (PL) side.

SDNet platform that will be used to process the packets. All packets arriving through the various interfaces and thus into the two respective clock domains are forwarded to the SDNet module and after being processed are passed on to the transmission modules again.

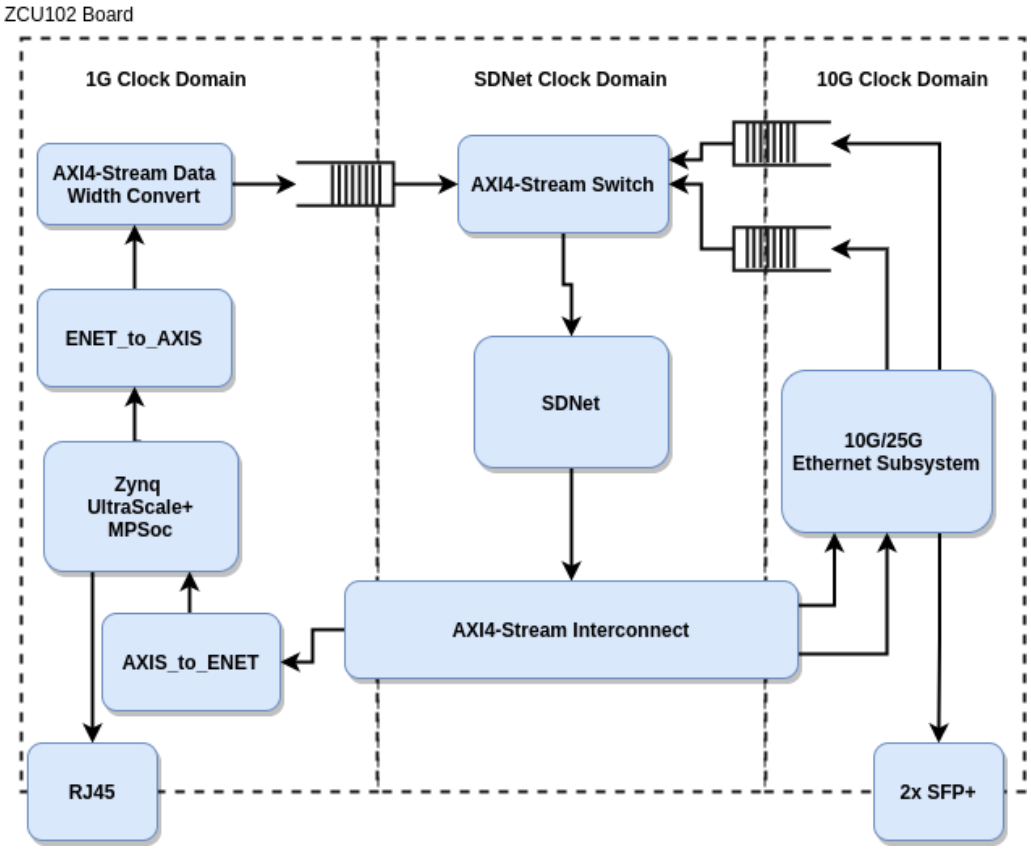


Figure 4.7: Switch datapath block diagram.

4.5.1 Ethernet Systems

To be able to send and receive Ethernet frames is needed modules that perform the MAC and PHY functions. Since the ports running at different speeds, two different approaches were needed.

4.5.1.1 Ethernet Frame Structure

An Ethernet frame is a protocol data unit defined by the data link layer used to carry data over the system. The network hardware exists to move Ethernet frames between endpoints, enabling device communication. Figures 4.8 and 4.9 show how the bits in an Ethernet frame are organized into fields. These bits are organized differently depending on the Ethernet standard and may contain more or fewer data fields. In order to keep it simple, this document focuses on two types of formats: the Ethernet II frame and the Ethernet IEEE 802.3 frame.

Preamble	Destination Address	Source Address	Type	Data	FCS
8 Bytes	6 Bytes	6 Bytes	2 Bytes	46 - 1500 Bytes	4 Bytes

Figure 4.8: Ethernet II frame format.

An Ethernet frame needs to be at least 64 bytes for collision detection to work and starts with the preamble. Initially, the preamble was used so the hardware had the ability to

Preamble	Destination Address	Source Address	Length	DSAP	SSAP	Control	Data	FCS
8 Bytes	6 Bytes	6 Bytes	2 Bytes	1 Byte	1 Byte	1 Byte	43 - 1500 Bytes	4 Bytes

Figure 4.9: Ethernet 802.3 frame format.

understand when a frame would start to be transmitted. Nowadays it is no longer necessary, but it is still transmitted so that no changes need to be made to the frame format. After the preamble is the destination and source addresses. The next field corresponds to the Type, which defines the network protocol, in the case of Ethernet II and the frame size if the format is IEEE 802.3. For Ethernet II, the data unit follows the Ethernet Type field, the size is restricted to 46 to 1500 bytes if it is not met, padding must be added to meet the minimum size. In the IEEE 802.3 format after the size field is the Logical Link Control (LLC) header used to manage and ensure the integrity of data transmissions. Finally, at the end of the frame, there is an 8 Bytes Frame check sequence (FCS) field, used to check the integrity of the data in the entire frame. Ethernet II is the most popular format because it allows more data to be sent since it does not have the LLC header.

4.5.1.2 Ethernet Controller Overview

In order to have an Ethernet controller, that controls Ethernet communications transmitting and receiving Ethernet frames, it is necessary to implement the first two layers of the Open Systems Interconnection (OSI) model. The OSI model describes computer functions into a common set of rules and standards, to facilitate interoperability across devices and applications. The traditional architecture of an Ethernet system can be visualized in figure 4.7, being the two first layers the Physical and Data Link Layer.

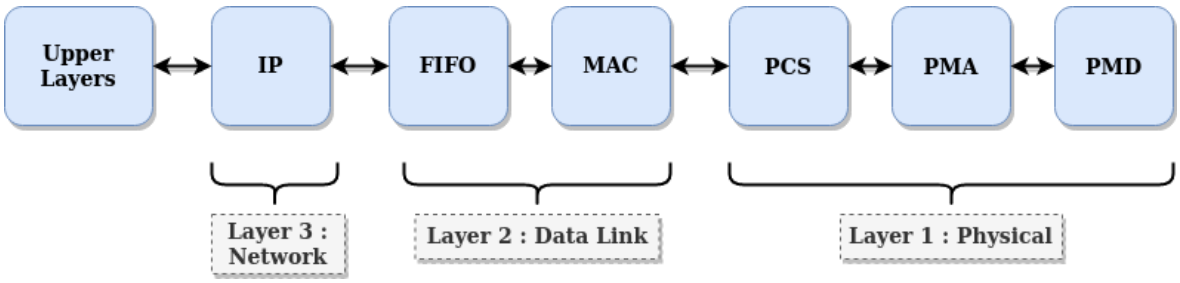


Figure 4.10: Ethernet System Architecture.

The PHY is the lowest layer that allows raw unstructured data bits to be sent over a network from the physical layer of the sending device to the physical layer of the receiving device through electrical or optical mechanisms. The PHY is a transceiver that acts as a link between the digital and the analog world and can be divided into three sublayers: Physical Coding Sublayer, Physical Medium Attachment and Physical Medium Dependent. The PCS is responsible for encoding and decoding data, scrambling, descrambling, serializing and deserializing. The PMA maps transmit and receive code-bits between the PCS and Physical Medium Dependent (PMD), and perform synchronization and detection. The last sublayer

describes the details of individual bit transmission and reception on a physical medium, consisting of a transceiver.

The PHY and MAC are connected via the Media-independent interface (MII), allowing any MAC to be used with any PHY. To improve and support higher speeds with fewer signals, some variants were developed, such as Gigabit media-independent interface (GMII), Reduced gigabit media-independent interface (RGMII) and 10-gigabit media-independent interface (XGMII). The second layer is composed of the MAC and First In First Outs (FIFOs) associated with transmitting and receiving. The MAC provides flow control, multiplexing and encapsulation for the transmission, also working as an abstraction layer for the upper layers.

4.5.1.3 10G/25G High Speed Ethernet Subsystem

The solution for the implementation of an Ethernet controller supporting 10Gbps was developed based on the Xilinx 10G/25G High Speed Ethernet Subsystem [33]. The 10G/25G High Speed Ethernet Subsystem implements the Ethernet MAC, PCS and PMA functions for 25Gbps and 10Gbps, providing the ability to receive and transmit via the AXI4-Stream interface. For the switch context, only the Ethernet subsystem functions for 10Gbps were used.

Figure 4.11 shows the 10G core block diagram, the IP' core can be configured to have up to 4 cores. It provides 64-bit or 32-bit AXI4-Stream user data interfaces and for control or status, it can be used vectors or the AXI4-Lite interface. The PCS module is not shown in the figure, as the cores are connected to it via board Multi-Gigabit Transceivers (MGTs). It is possible to use the IP' core as Optional Standalone Version MAC or PCS, both with XGMII Interfaces.

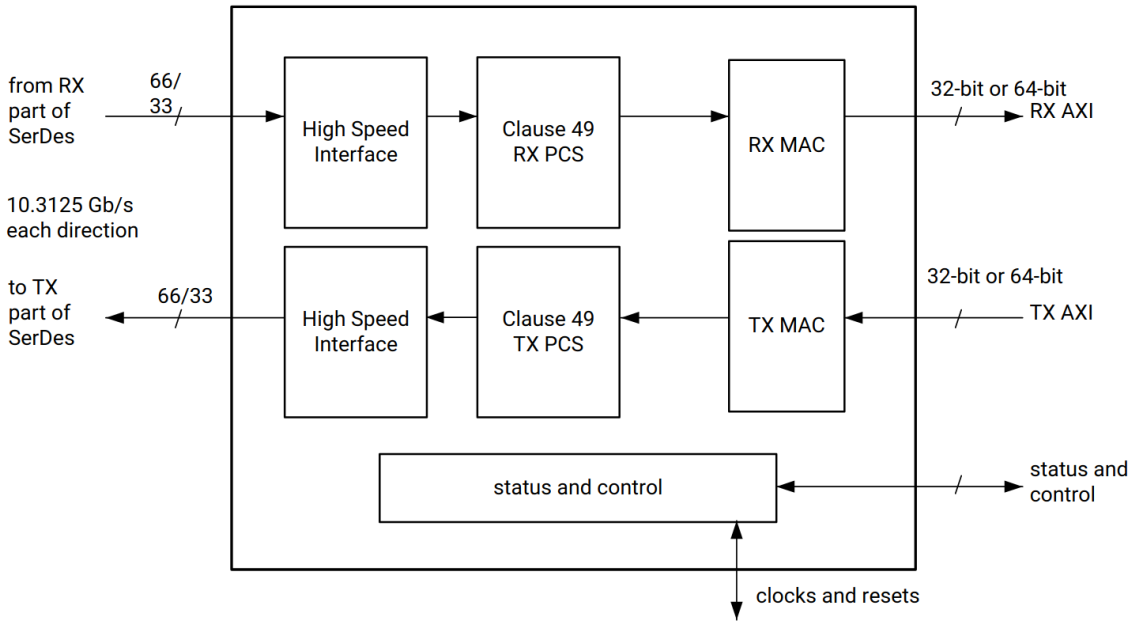


Figure 4.11: 10 Gbps Core Block Diagram (retrieved from [34]).

Once the Ethernet Subsystem is placed in the block design there are several parameters that need to be configured in order for the IP' core to work as intended. The first important

step is to choose the IP' core speed since it can be configured with 10G or 25G. In addition, the IP' core is configured with two cores as two systems will be required for the two switch ports, thus having separate interfaces for the two 10G ports. The cores are configured as Ethernet MAC+PCS/PMA with 64-bit interfaces, leaving only the PMD function located on the transceiver. Regarding PHY functions, it is configured with BASE-R standard, featuring 64B/66B signal encoding and supporting transmission over optical fiber medium. Besides the core of figure 4.11, the IP' core also has a GT subcore and some shared logic. The shared logic consists mainly of clock buffers and reset logic. The IP' core configuration window also allows the definition of the frequency clocks related to the GT and the choice of the transceiver associated to each core, specifying the quad and the lane of the transceiver on the FPGA. Each transceiver chosen corresponds to the SFP interface that is intended to operate as the switch port.

The timing of a normal 64-bit frame transfer is shown in figure 4.12. To transmit a frame, the first step is to assert the *tx_axis_tvalid*, thus indicating that the bus data is valid. So in the same clock cycle place the data and control information in *tx_axis_tdata* and *tx_axis_tkeep*, respectively. The data has only been accepted by the ethernet core when the *s_axis_tx_tready* signal is asserted, only here is it desired to provide the next data word in the next clock cycle. The *tx_axis_tkeep* flag indicates which of the bytes present in *tx_axis_tdata* are valid, because the signal width is 64-bits but there may only be 32 left, for example. Lastly, the core knows that the packet ends when the *tx_axis_tlast* signal is asserted for one cycle. Extra care is needed in the order in which the bytes that correspond to the source and destination addresses are sent.

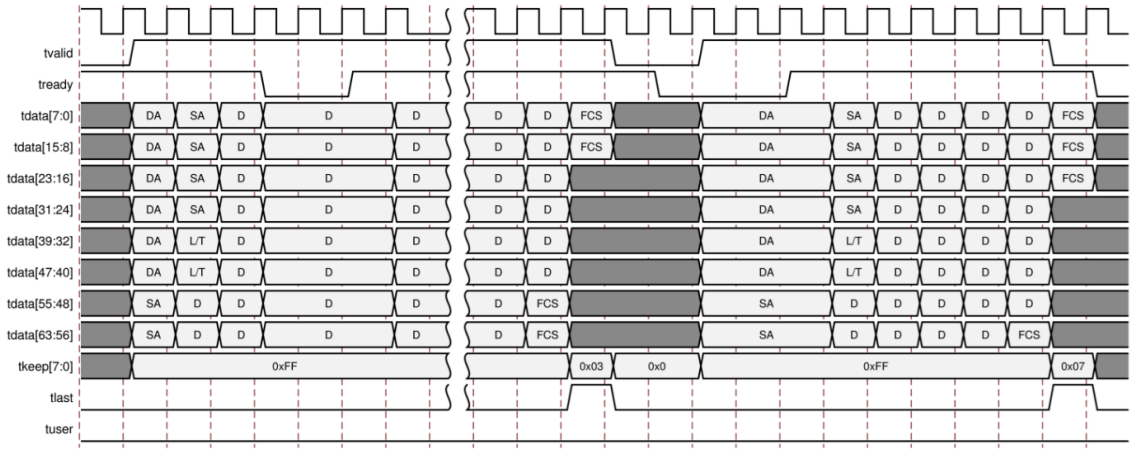


Figure 4.12: Normal 10G frame transfer (retrieved from [34]).

4.5.1.4 PS - Gigabit Ethernet Controller

Unlike the 10G controller, to take advantage of the hardware already on the board, to implement the 1G Ethernet solution the GEM in the board was used. The board is equipped with four controllers that are on the PS-side and can implement Ethernet communications at 10/100/1000Mbps Ethernet interface. Each controller can be configured independently but

only GEM3 was used [35]. As figure 4.14 demonstrates the GEM3 is directly connected to a TI DP83867IRPAP Ethernet RGMII PHY before being routed to an RJ45 Ethernet connector. The DP83867IRPAP device [36] is Ethernet PHY Transceiver robust and low power, which implements all the functions referred to in section 4.5.1.2. This system only supports RGMII and Management Data Input/Output (MDIO), which is used to manage the PHY module.

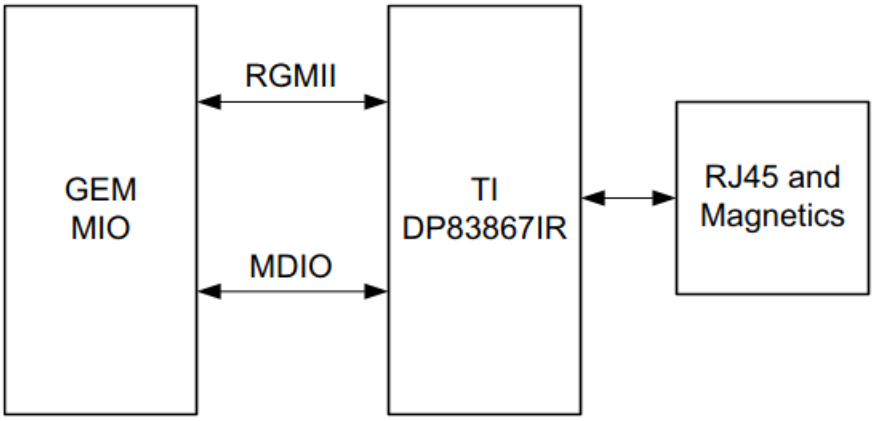


Figure 4.13: PS-side Ethernet Block Diagram (retrieved from [35]).

As can be visualized in figure 4.14 which demonstrates the block diagram of the GEM Ethernet controller there are two ways to access the packets received at the PS via the PL. Since the data is transmitted and received via the GEM RXFIFO and TXFIFO can be accessed through a GEM DMA controller or an external FIFO interface. Since the packet processing module is on the PL side, it was decided to use the slave interface via the external FIFO interface with an 8-bit data access width.

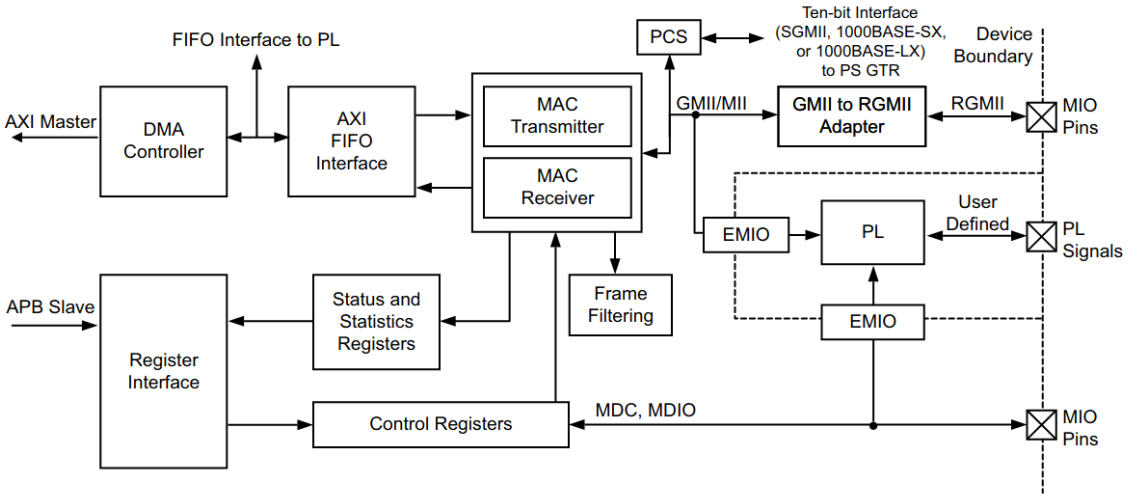


Figure 4.14: GEM Block Diagram (retrieved from [35]).

In order to have access to PS-side, and consequently GEM3, the Zynq UltraScale+ MPSoc was added to the project block design enabling the External FIFO interface. The interface available by GEM for communication between PS and PL was FIFO_ENET, since throughout

the project the AXI4-Stream protocol is used as a standard interface to connect components that need to exchange data it would be advantageous to convert this interface to AXIS. Therefore, two VHDL modules were created that would do this conversion resulting in the block diagram in figure 4.15 for the 1G solution.

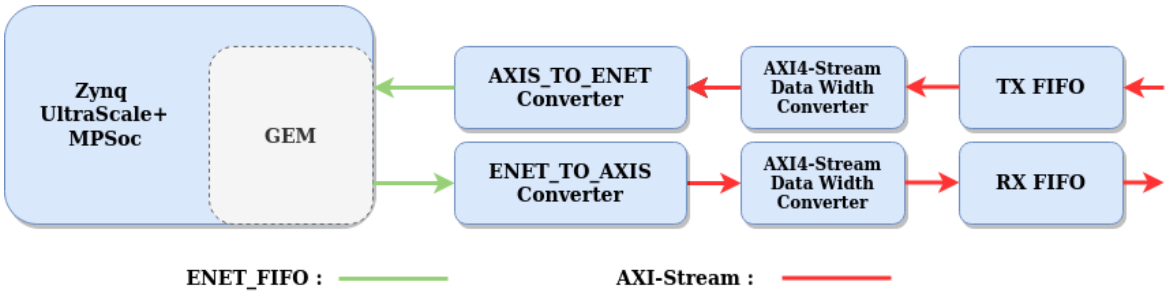


Figure 4.15: Blocks implemented for 1G solution.

The module "enet_to_axis" which converts the ENET interface into the AXI Stream interface keeping the 8 bits of data width is essentially done forwading signals. This is possible since the interfaces are very similar as can be observed in figure 4.16, where the timing diagram of the transmission of a packet through the GEM FIFO interface is represented. The main difference between these two interfaces is the existence of a signal, *rx_w_sop* that indicates when a packet starts to be received. The module "axis_to_enet" is responsible for the opposite conversion, based on state machines that generate the desired interface signals by using similar signals from the AXI Stream. As specified in figure 4.17, the ENET interface has an offset of one clock cycle between the signals indicating the validity of the data and the signal indicating that the GEM is available to receive data. This implies the need for a state machine and the addition of a clock cycle delay in the transmission.

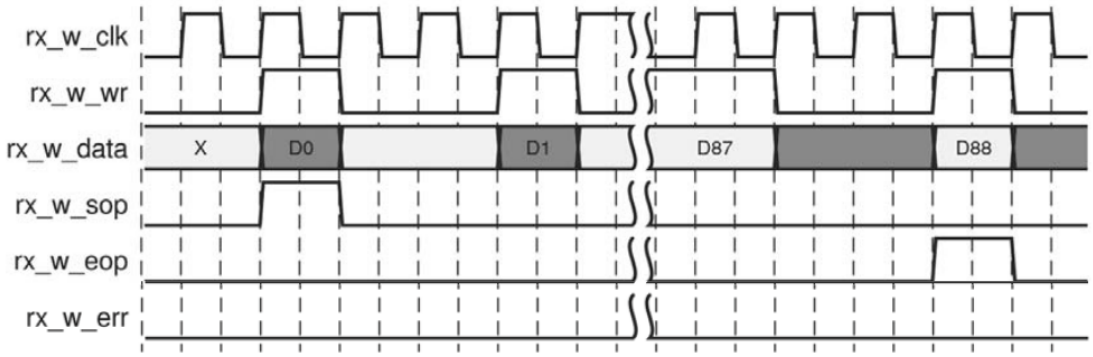


Figure 4.16: Frame reception on the FIFO interface (retrieved from [37]).

Since both conversion modules keep the data width it is necessary to convert the interfaces from 8 to 64 bits, since this is the data width used in the rest of the system's data path. To meet this requirement the Xilinx AXI4-Stream Data Width Converter IP is used [38]. The IP is able to increase and decrease the width of the TDATA signal by combining a series of AXI4-Stream transfers into one larger transfer or splitting into a series of smaller transfers.

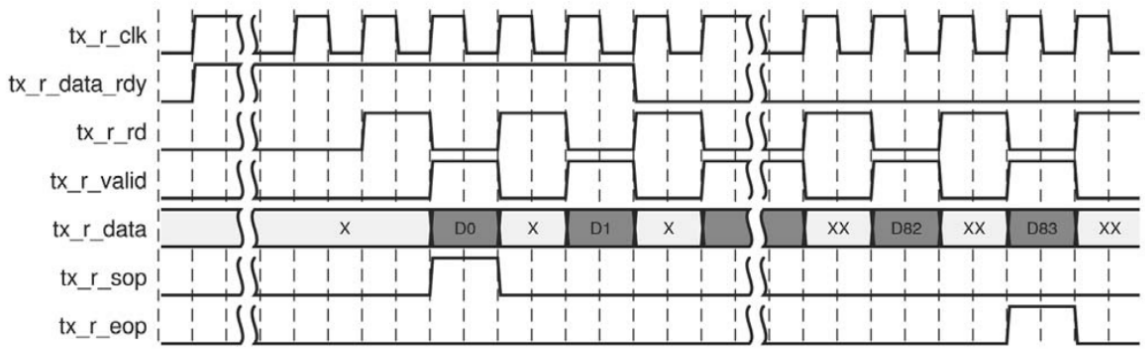


Figure 4.17: Frame transmission on the FIFO interface (retrieved from [37]).

On the PS-side the Xilinx provides a driver which works as a standalone Ethernet driver that handles transmission and reception of Ethernet frames, as well as configuration and control. The driver is the XEMACPS driver [39], which supports full duplex operations at 1000Mbps, automatic padding and FCS, address checking and flow control, among other features. The driver was used as an example, but once it only supports DMA transfers, being unable to support the external FIFO mode it was necessary to make some changes. The example provided together with the driver, "xemacps_example_intr_dma", was used as a support. The advantage was taken of sections such as GEM and PHY configuration, discarding the DMA-related part and added configurations such as:

- Select the GEM3_FIFO clock;
- Enable jumbo frames and full duplex;
- Enable PL FIFO mode along with TX and RX;
- In addition to the existing PHY settings, loopback was disabled, 1G speed was configured and auto-negotiation with the link partner was added.

4.5.2 Input and Output Arbiters

As a way to implement the output arbiter AXI4-Stream Interconnect was used. This IP core routes connections from one or more AXI4-Stream master channels to one or more AXI4-Stream slave channels. In the case of this project the goal would be to route a slave interface, which comes from the SDNet, to three master interfaces that are connected to the 3 ports of the switch. The module besides being able to perform data switching/routing, by using the TDEST signal to route transfers to different slaves, has in its architecture components such as FIFOs, data width converters and clock converters. Which can be integrated or removed in order to enrich their capabilities. In this case the interconnect is embedded with FIFOs with packet mode active, to ensure that a packet when it starts being transmitted to the Ethernet subsystems is already fully in the FIFO. This is guaranteed by activating the TVALID signal only when a TLAST signal is received.

Regarding the input arbiter, an integration of the AXI4-Stream Interconnect was attempted with the aim of implementing it in the same way as the output arbiter, since it had all the desired characteristics. Due to external problems it was not possible to use the interconnect, so a solution was developed that takes advantage of the same IP cores used by it. In addition

to the AXI4-Stream Switch, that in this case was used to connect three masters to one slave interface, AXI4-Stream Data Width Convert was used to increases the width of the TDATA signal, that comes from the "enet_to_axis" module mentioned in the subsection, from 8-bit to 64-bits. No AXI4-Stream Clock Converter was needed since AXI4-Stream Data FIFOs that were used to ensure that the system can always receive packets, already provides cross clock domain.

The switching module prioritises the 10Gbps ports over the 1Gbps port to ensure that ORAN traffic is not delayed. This priority is achieved by setting the interface scheduling algorithm to fixed priorities. In order to guarantee the correct transfer of a packet it is necessary to ensure that it only starts being forwarded to the destination interface once it is already available from the SDNet module. If this is not guaranteed, a transfer may include the malformation of a packet that comes from more than one interface. The AXI-Stream Switch with this configuration can perform a maximum number of transfers before it is forced to switch interfaces in order to avoid starvation. In this case it should not be avoided as the 10G ports should always have priority over the 1G ports.

4.5.3 P4 SDNet

Packets arrive at the SDNet module through a single AXI4-Stream interface. Therefore, it is necessary that they are identified with the source port. The module supports meta-data input in order to provide more information along with the packets, either as input or output. In order to facilitate its implementation, the module allows signals such as TID to be added to the interface, this signal being used to identify the source port and TDEST to identify the destination port. Since the SDNet output is designed for 100G rates, it can easily handle the aggregate 21G rate.

The module is configured with a P4 program that describes how the packet processing should be done. As discussed, ORAN packets can be encapsulated in several different transport layers. In the P4 program, several headers are created, like eCPRI, RoE, VLAN, for specific protocols and each packet is parsed in the parser engine, the first component of a P4 architecture. On figure 4.18 is possible to visualize the parser structure.The P4 parser describes a finite state machine where the first state is always named start and in the end a packet can be accepted or rejected.

The headers are passed to the match-action engine where data can be changed and is checked if the packets are ORAN according to the fields in the headers. It is at this stage that the meta-data, i.e. the source port, is analysed and the destination port is processed. Several fields are checked for valid values, for example the ecpriMessage field can only have the following values: 0x0 for U-Plane data, 0x2 for C-Plane data or 0x5 for network delay measurement messages, according to the O-RAN fronthaul specifications. Finally, only the inverse of the parsing, deparsing, remains to be done, in order to have a final packet ready to be sent.

The SDNet module calculates the worst-case latency automatically, in this case 96.26 ns. The latency variance may depend on the back-pressure applied in the AXI Stream interface,

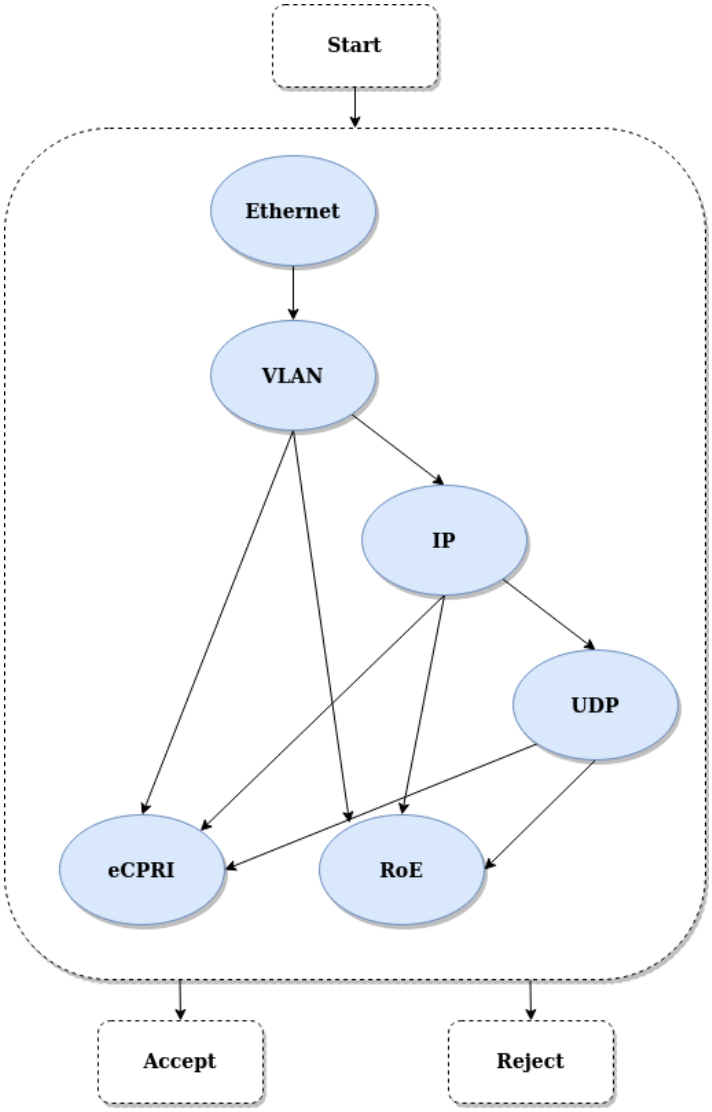


Figure 4.18: Parser finite state machine structure.

the size and number of packet headers. If a packet has a small number of headers to be parsed, can have a lower latency.

4.5.4 Embedded Application

In order to have access to the PS-side a Zynq UltraScale+ MPSoC is instanced on the system. The processing system features two Arm processors, a Cortex-A53 64-bit quad-core processor and Cortex-R5 dual-core real-time processor. Besides the instructions necessary for the use and control of GEM3 mentioned in section 4.5.1.4, the processor Cortex-A53 is still used for control and management of the IPs using a C embedded application on the Vitis platform. In the application, the IP configuration is done mainly by writing values in specific registers and reading status values. In the application, the IP configuration is done mainly by writing and reading values in addresses from the Configuration and Status Register Map of the IPs.

This writing and reading are done through the AXI4-Lite interfaces with the support of AXI Interconnect, which allows the master interface from the Zynq module to be connected

to the desired IP cores. The address space of the PS-side is thus segmented so that each slave device has an address range of the size it needs.

4.5.5 Clocking

So far the clock has been omitted during the implementation in order to simplify its explanation. Nevertheless, the clock is a crucial element in the system. Figure 4.19 represents the system clock tree, where it is possible to visualize the origins of the various clocks used. In the figure the FIFOs and reset modules are omitted, although these also require clocks. Cross clock domain is achieved in the switch implementation through FIFOs, since a large amount of data is always to be transferred between domains.

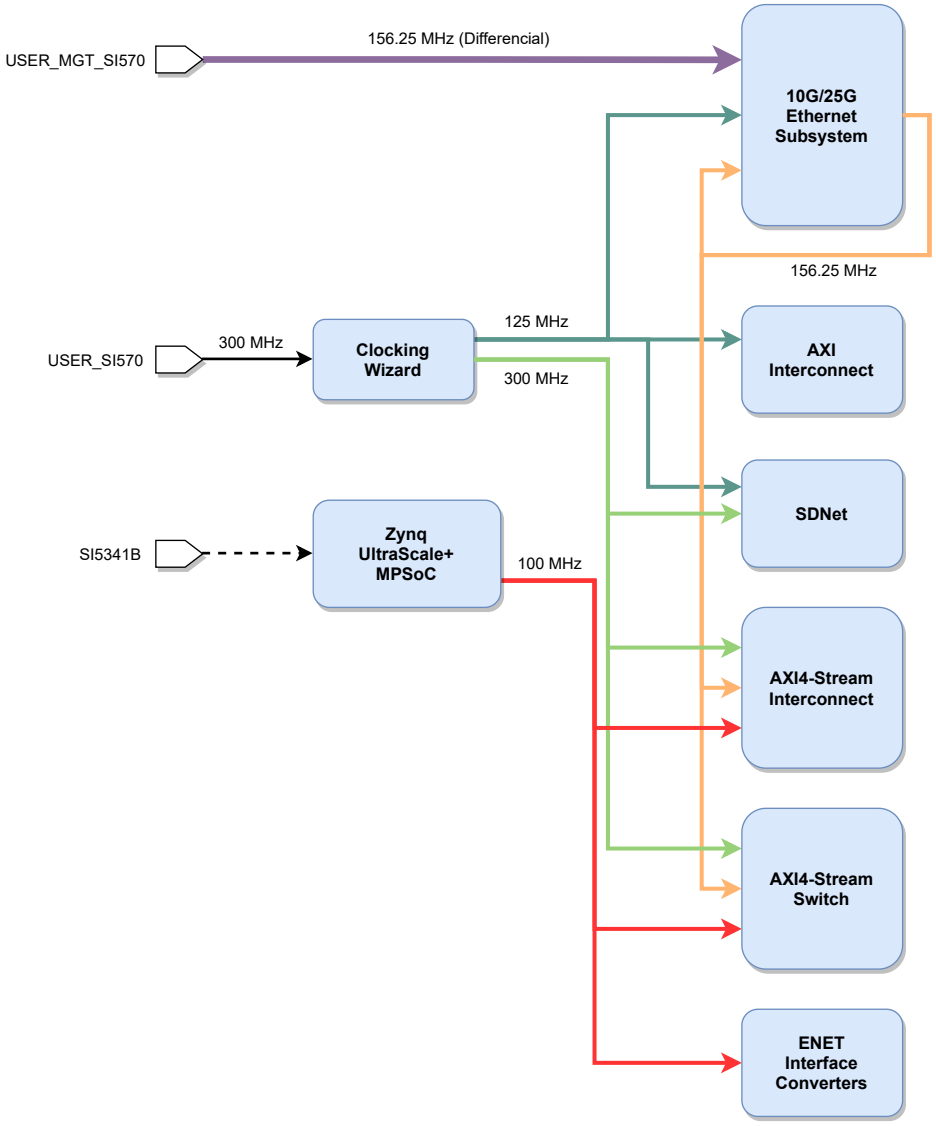


Figure 4.19: Switch clock tree.

The 10G/25G Ethernet Subsystem block needs multiple clocks since it has three clock domains in the internal datapath. Since the 64-bit interface for 10G is being used, the GT block needs a 156.25 MHz reference clock. The module receives as input a pair of differential

clocks and a buffer transforms this clock to a single-ended signal. The clock used is the USER_MGT_SI570 that is generated from a programmable low-jitter 3.3V LVDS SI570 differential oscillator connected to a SI53340 clock driver. The clock USER_MGT_SI570_CLOCK1 was the one used since it is in the same bank as the GTs and SFPs thus not violating localization constraints. The single-ended clock mentioned is the clock associated with the AXI-Stream data transmission and control interfaces. As can be seen in the figure, this clock is again driven into the system to be used as an input clock for the RX core. When the system is connected in this way, all the frame reception and transmission logic are in the same clock domain. Finally, the system still has the dclk signal that should be a convenient stable clock. It is used as a reference frequency for the GT help blocks that start the GT itself, in this case having a frequency of 125 MHz.

Regarding the GEM used on the PS side, it uses as an input to the Tx and Rx clock of the module an internal clock from the PS. As clocks of the transmit and receive interfaces, two 100Mhz clocks are available, represented in the figure as a Zynq block output. The 100Mhz signal is used in the clock domain of the part of the system that operates at 1G.

In order to generate the remaining clocks, the Xilinx Clocking Wizard was used. This IP core accepts as input one or two reference clocks and from these is capable of generating up to eight clocks with the frequencies desired by the user. As input, the programmable clock USER_SI570 was used, which is generated in the same way as the clock for the MGTs, but this time it is directly connected to the FPGA. From this, two clocks were generated, one with a frequency of 125 MHz and the other of 300 MHz. The 300 MHz clock is used by the packet processing module as the AXI4-Stream clock. The signal frequency is this as it is the highest frequency supported by the SDNet module, thus ensuring maximum processing efficiency. The 125 Mhz signal is used as the clock for all the AXI4-Lite interfaces, which are the interfaces used to configure the various modules by writing and reading internal registers.

At this point of development, the switch is ready to be tested both functionally and in terms of the latency it introduces to the fronthaul link. In the next chapter it will be described how this kind of tests were performed.

CHAPTER 5

Test Platform and Initial Results

In this chapter the objective is to present the platform developed and validate the implementation deployed.

5.1 Functional Validation

During the development of the switch it was necessary to validate the developed P4 program. The P4 Behavioral Model was used before the implementation to test the P4 program. This model was available together with the SDNet IP core and produces an independent replica of the operations provided in the source file to compare against expectations and the RTL implementation. The model receives a JSON file that is provided by the P4 compiler, along with Packet Capture (PCAP) files with the packets information and a file with the corresponding metadata. Tests were carried out with eCPRI, RoE, TCP and UDP packets on the various ports and the results were always positive, Once packets have been captured on the correct interface, thus validating the proper functioning of the P4 program.

As a way of testing the various stages of development as well as the final version, the laboratory setup shown in figure 5.1 was used. A computer with a network interface card with two SFP slots was used as a link partner. This way the three ports used by the switch can be connected to the computer as it already had an RJ45 port. Therefore, through the computer, it is possible to generate and capture packets through any interface.

Initially, the tools used for packet generation and capture were Packeth [40] and Wireshark [40], respectively. The Packeth is a packet generator tool for Ethernet that allows the creation and sending of packets or sequences of packets on the Ethernet interface. The Wireshark is a tool that captures packets from a network connection and a network protocol analyzer. By generating, injecting and capturing various types of traffic on the various interfaces, it was possible to analyse the behaviour of the switch, validating it functionally. Besides varying the type of packets, the size of the packets and the number of packets sent was also varied, and the switch matched the expectation.

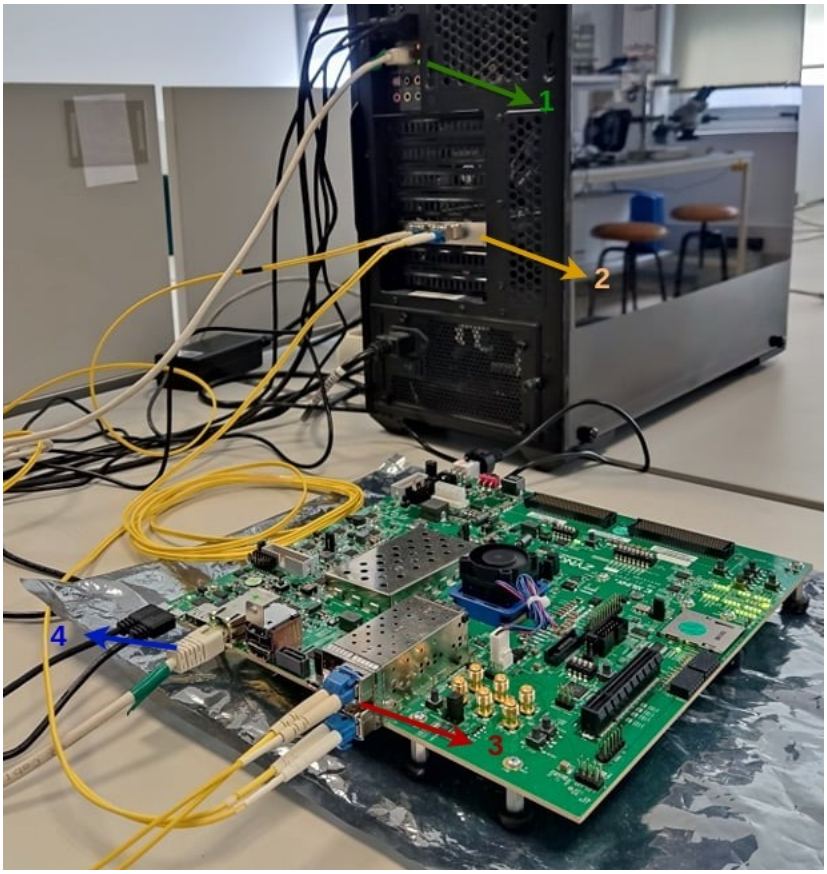


Figure 5.1: Laboratory Functional Setup. 1 and 4) RJ45 connecting the 1G ports of the computer and board via an Ethernet cable. 2 and 3) SFP+ connecting the 10G ports of the computer and board via an fiber cable.

As the use of the two tools as a form of validation was not very practical, other solutions were considered. A C program based on the socket and pthread libraries was developed in order to automate the functional validation. The program sends and captures the packets from an interface and verifies if the traffic sent was the same as the traffic received.

5.1.1 FPGA Resource Utilization

In figure 5.2 it is possible to verify the FPGAs resource usage. On the right is the distribution of resources and the area they occupy and on the left is a graph representing the percentage used of each type. With plenty of resources left, there is room for improvement in some features, such as increasing the size of FIFOs to store a larger number of packets.

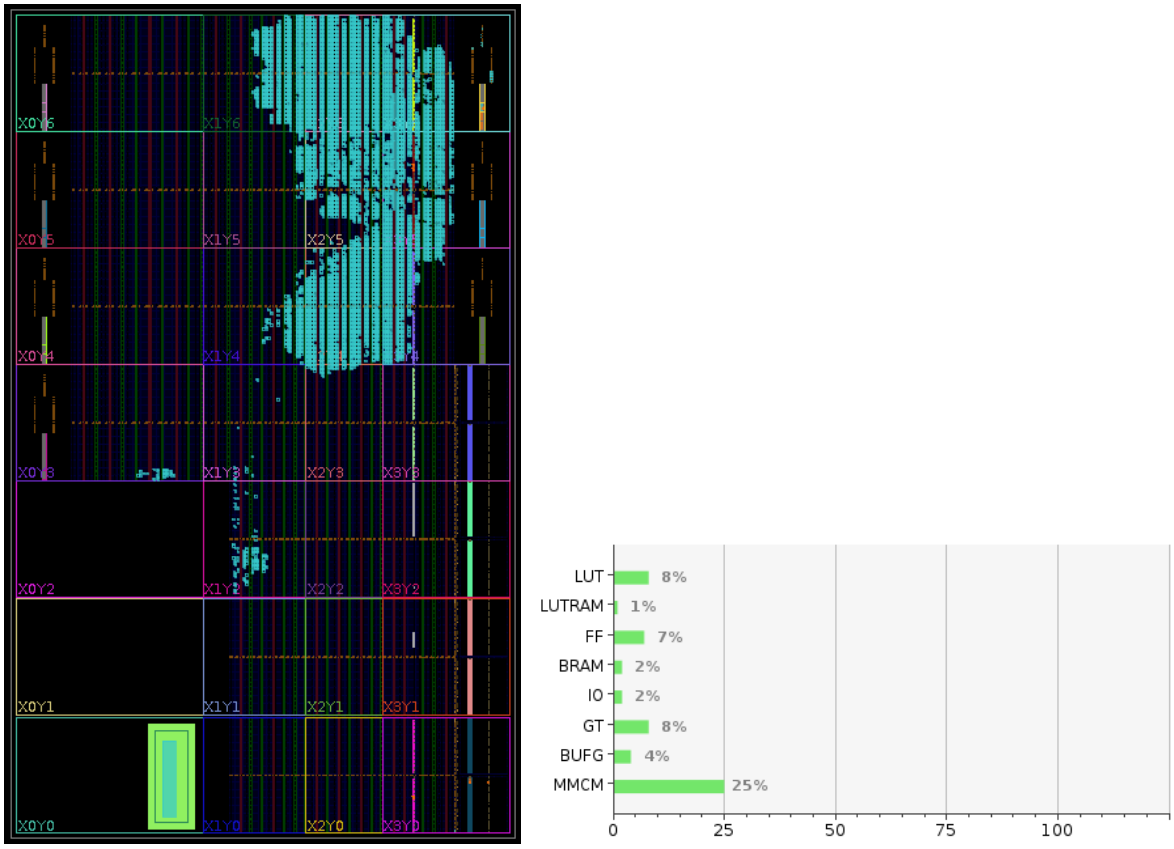


Figure 5.2: ZCU102 FPGA Resource Utilization.

5.2 Platform Implementation

In order to test the switch from a performance point of view, a platform that generates and monitors the traffic was developed. The platform must have two ports, one of 1Gbps and the other of 10Gbps, through which traffic will be injected with packets previously loaded on the platform memory. The packets sent must be received on the same interfaces. The platform measures the time it takes to send and receive packets to calculate latency, in this case the latency that the switch adds to the network. The same development board was used to measure times on a small time scale. Unlike a computer, the FPGA is able to achieve an accuracy of nanoseconds. The platform communicates with a partner, usually a PC, via a Universal Asynchronous Receiver-Transmitter (UART). The latency times already processed by the ZCU102 processor are sent via the UART.

Figure 5.3 illustrates the architecture of the performance test platform. The orange blocks represent the VHDL modules developed from scratch and the blue blocks represent the IP cores available from Xilinx, some of which have already been covered previously. As can be seen, use has been taken on the PL and PS side, leaving tasks such as time processing and communication via UART to the processor on the development board. On the PL side, there is a symmetry in the implemented system, the blocks that compose the 1G solution are the same with different configurations except for the Ethernet Subsystems. The 10G/25G Ethernet Subsystem was used with the same configurations as in the switch, but in this case,

only one core was required as there was only one 10G port. In the implemented version it was used with two cores in order to have a port that helps in debugging and validation of the packets sent, this is not represented in the figure. All control blocks related to reset system and clocks are omitted from the figure in order to facilitate their interpretation.

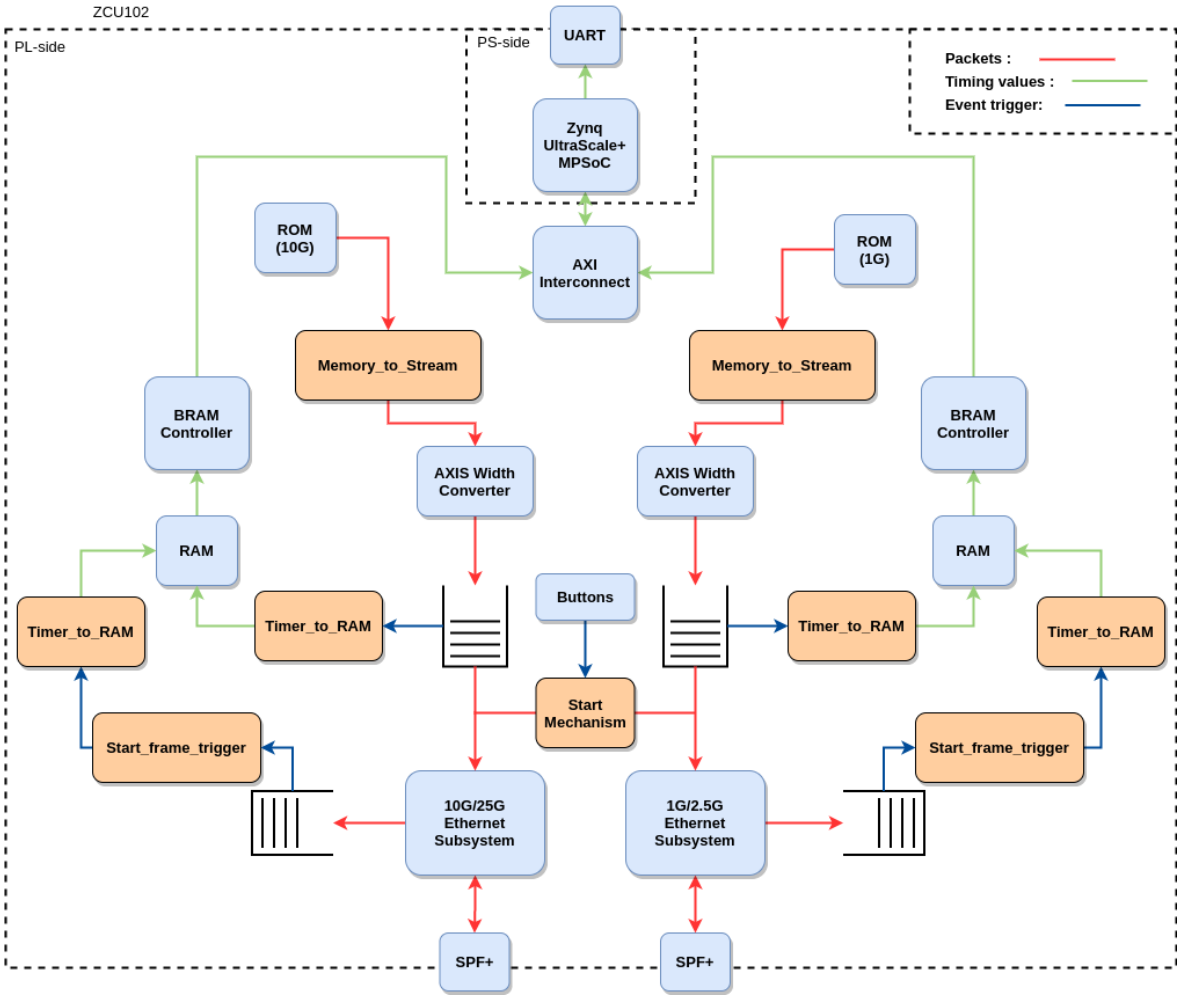


Figure 5.3: Traffic generator and monitor platform block diagram.

5.2.1 Ethernet Subsystems

The initial goal was to make it possible to implement the test platform and the switch on the same development board, so a new approach to the 1G Ethernet solution was needed. Since the RJ45 port on the PS-side was already being used by the switch it was only left with the interfaces provided by the board to use a 1G SFP+ RJ45, figure 5.4. The AXI 1G/2.5G Ethernet Subsystem was used, which allows the implementation of Layer 2 and Layer 1 functions, thus enabling the reception and transmission of Ethernet frames. As can be seen in figure 5.5, the components of the Ethernet Subsystem are the Xilinx Tri-Mode Ethernet MAC (TEMAC) and 1G/2.5G Ethernet PCS/PMA. It is possible to add the AXI Ethernet Buffer core to have more functions, but it was not used in this application. The subsystem uses

an AXI4-Lite interface for configurations by writing to registers and AXI4-Stream interfaces for receiving and transmitting Ethernet data to and from the subsystem.

Figure 5.4: 1G SFP+ RJ45.

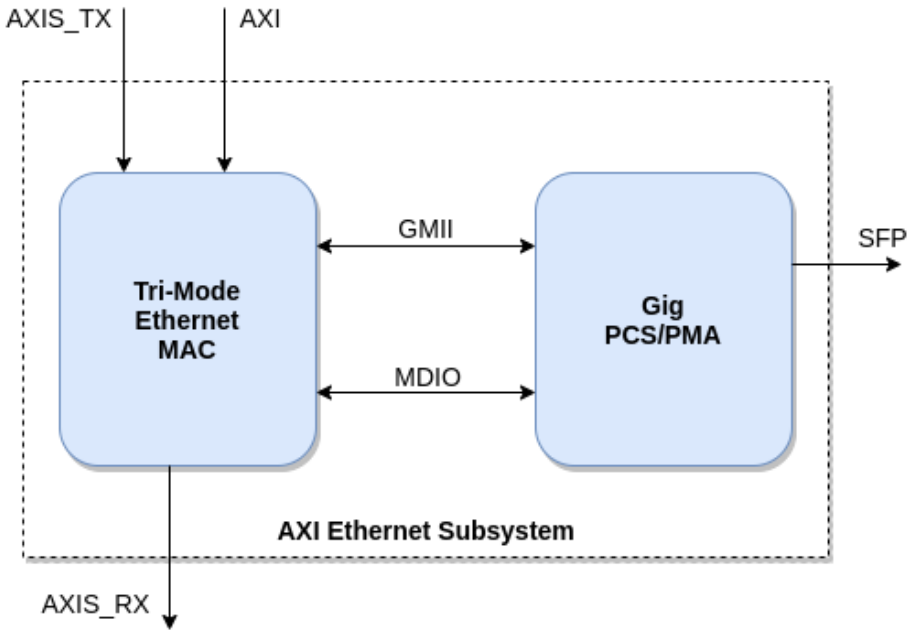


Figure 5.5: Block diagram of the AXI Ethernet Subsystem.

The major difficulty in implementing Ethernet subsystems is related to the differential clock for the serial transceiver. This problem arises because the SFP cage is located on bank 230 of the ZCU102 board and on Ultrascale+ boards the reference clocks required for transceiver operations can only be routed up to two banks up or two banks down from the bank where the transceivers are located. As shown in figure 5.6, the reference clock needs to exist on banks 228, 229 or 230. Existing the differential clock MGT_REFCLK on bank 230, it is used for the 10G/25G Ethernet Subsystem but it is not possible to use the same clock in the other subsystem. This happens since the buffers, IBUFDS_GTE4, that convert this clock to a single-ended signal are inside the subsystems and is impossible for the differential IO clock ports to be derived for multiple buffers. One solution would be to extract the buffer from the subsystems, using only one for the two and providing already the single-ended clock for the IP cores. It turns out that the way to extract the buffer is to stop including Shared Logic in the Core, it no longer exists in the block design of the project and it is necessary to

import the VHDL modules through IP Example Design. In this way, it is possible to extract the buffer and use the same one for both subsystems but the shared logic is composed of more components like PLLs and other buffers. It was noticed that the extraction of the shared logic brought an added complexity to the design as many more control signals were placed at the user's level.

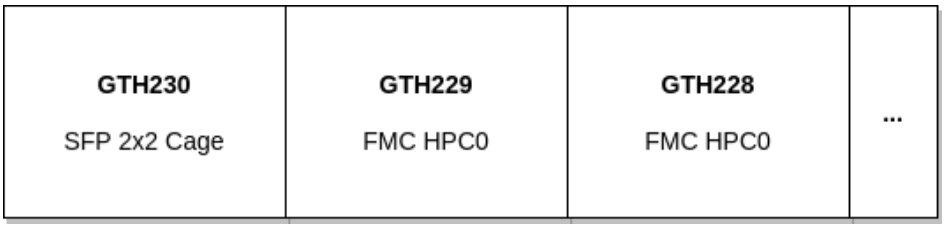


Figure 5.6: ZCU102 Evaluation Board Block Diagram.

So there remains the option of using a clock from the neighbouring banks, but it turns out that banks 228 and 229 provide an FMC connector. The FMC is a standard that specifies a standard mezzanine card form factor, connections, and a modular interface to an FPGA on a base board. Allowing expanding the functionalities of an FPGA there are several types of FMC. In this case, the FMC would only serve to provide one more clock to the system and must have a frequency supported by the subsystems. This it was used a FM-S14 Quad SFP/SFP+ transceiver FMC [41], represented in figure 5.7, which provides up to four SFP/SFP+ module interfaces and two programmable reference clocks. Fulfilling the requirements, the SFP interfaces were not used as the four of ZCU102 board were sufficient, being used one of the four default frequencies (312.5 MHz) that can be selected using switches on the FMC module.

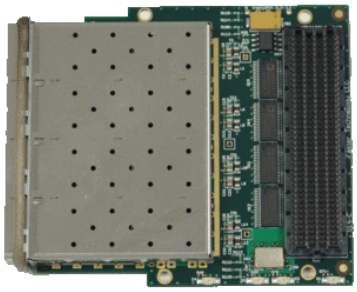


Figure 5.7: FM-S14 Quad SFP/SFP+ transceiver FMC.

5.2.2 Packet Generation

The packets that are sent are initially loaded into memory. For this purpose, is used the Block Memory Generator core [42] which allows the construction of performance-optimized memories using embedded block RAM resources in the FPGA. For this purpose, the memories were configured as Single-port ROM since packets are loaded into the system before synthesis and implementation. The initialization of the memory is done from a Coefficient (COE) file loaded during the block configuration. The COE file allows the specification of the initial contents of each memory location.

To ensure that the ethernet subsystems have the frames ready to be transmitted, a module was developed that can be seen as a simplified version of a Direct Memory Access (DMA). The module is the Memory_to_Stream which provides two interfaces, one master AXI-Stream and one BRAM interface. The block is configured with address and data width, the frame size and the number of frames that need to be transmitted. With this information when the system is booted the module reads from memory the data of the packets through the BRAM interface and sends it through the AXI-Stream interface in order to fill the FIFO. The transfer from one interface to the other is done through a state machine, according to the reception availability of the FIFO interface and asserting the TLAST signal in the last word of the packet. The main limitation of the module is that it does not support the transfer of packets of different sizes, so the loaded COE file can have different data but with the same size.

After packets have been loaded into the FIFO the platform has a mechanism that allows the beginning of the transmission of information from the FIFOs to the subsystems. The transmission starts after the user pushes a button on the board. The module via Flip-Flops and logic gates controls the TVALID and TREADY signals of the streams interfaces to ensure that no packets are transmitted. The user has the option of sending packets through both interfaces simultaneously or only through one of their choice.

5.2.3 Timing Mechanism

Once the packets are being transmitted and received, the second objective of the platform is to measure the transmission and receiving times in order to calculate the network latency. For this, the module developed has an internal timer and when the trigger signal is activated it saves the current timer value in memory. So, besides the trigger input and the clock that sets the timer frequency, the module has the same BRAM interface that the previous module to write to specific memory addresses.

In this case, it was used again the block memory generator as a way to instantiate a block of memory in the FPGA. Since it is intended that time values are to be written to and read from memory, it was configured as True Dual Port RAM without any initialization of the memory. In the case of the Memory_to_RAM block, it only writes to memory so that the values can be read with the help of the processor later. Since the RAM has two ports the other is connected indirectly to the PS-side. The PS-side that has an AXI interface with the PL accesses the memory through two blocks, an AXI Interconnect and the BRAM controller. The AXI Interconnect is used to connect one AXI memory-mapped master device to multiple memory-mapped slave devices. The slave devices are the BRAM controllers which allow the connection between AXI Interconnect and the RAM blocks allowing operations over the memory. The reading of the times is only performed after the transmission and reception of the packets is finished.

The timer modules write a value to memory when the trigger input signal assumes an active value ('1'), since it is planned to capture times in two different periods in the execution of the system the signal used also differs. Firstly, the TLAST signal from the AXI-Stream interface connecting the TX FIFO to the Ethernet subsystem is used as a trigger in order

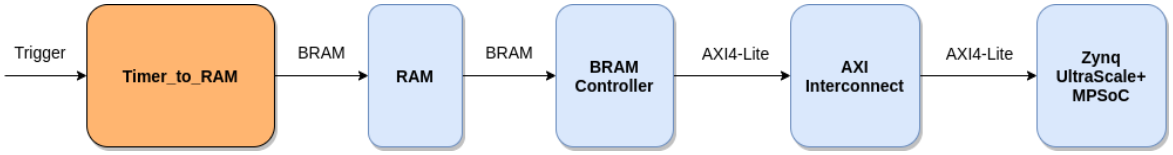


Figure 5.8: Timing mechanism with the interfaces between blocks.

to obtain the time at which the packet starts to be transmitted. In the case of reception, it is intended that the time is captured when at the instant the packet starts to be received. Since the AXI-Stream interface does not have any signal indicating the start of the packet, a module was developed in VHDL that receives the TLAST, TREADY and TVALID signals and detects when a new packet is received. When this event occurs it generates the trigger signal to capture a time value.

5.2.4 Embedded Application

On the PS side, the Embedded application not only configures the Ethernet Subsystems but also processes the time data and sends it to the PC. The times are read from the various RAMs of the system and the latencies, minimum, maximum and average values are calculated. After that, all the information is sent through the PS UART.

5.2.5 FPGA Resource Utilization

In figure 5.9 it is possible to verify the FPGAs resource regarding the implementation of the platform for testing. Comparing the platform to the switch there is a higher resource utilisation, this can be explained by the use of one more Ethernet Subsystem. While the switch used some features that were situated on the PS side, such as the solution for the transmission of packets at 1G, now everything is implemented on the FPGA resources.

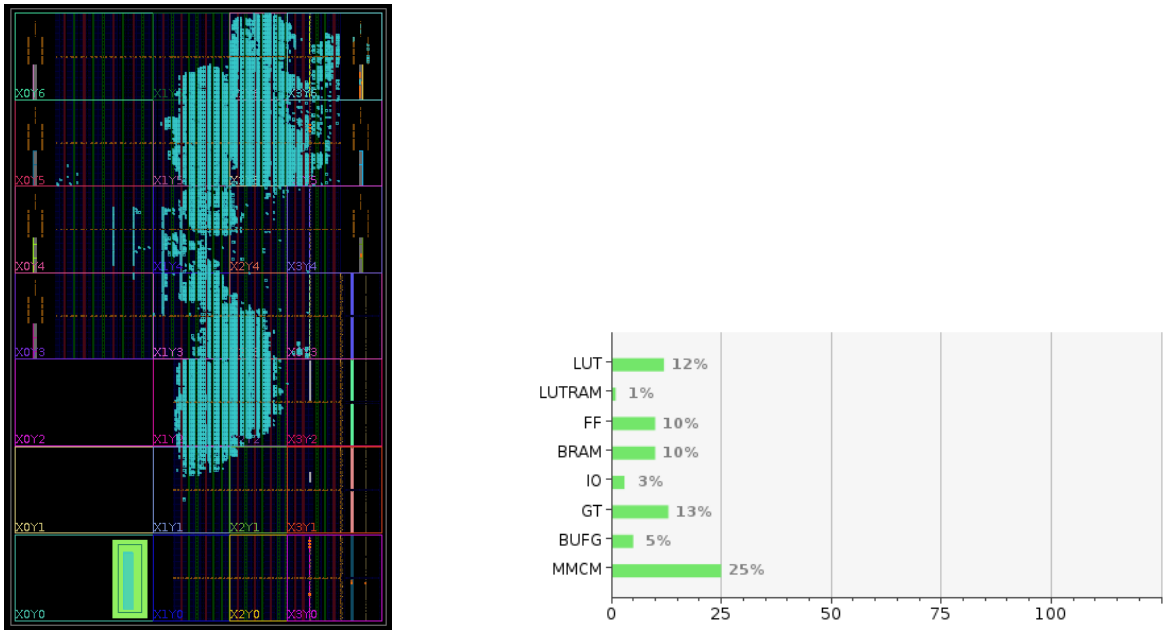


Figure 5.9: ZCU102 FPGA resource utilization of the test platform.

5.3 Performance Tests

With the platform completed, it was validated using ILAs and Wireshark. Traffic generation was tested with the platform connected to the computer as in section 5.1 and capturing packets on both interfaces. Since the packets stored in memory are known previously, it was possible to confirm the packet information and quantity. Regarding the validation of the times, the waveforms of the data captured by ILAs were analysed with the obtained times, concluding that the times are the expected ones with the addition of a delay cycle due to the time measurement mechanism used.

With the switch and the test platform functionally validated, it is possible to measure the latency that will be added to the fronthaul link with two switches. Consequently, with these times it is also possible to analyze the latency added to the general-purpose traffic due to the lowest priority. Therefore, the laboratory setup demonstrated in figure 5.10 was assembled with two ZCU102 boards and a computer. In the setup, the test platform is connected to the computer via UART and connected to the switch through the 1G and 10G interfaces. On the other 10G port that corresponds to the interface connected to the fronthaul, traffic exits a loopback using a 10G SFP+ Loopback module. Figure 5.11 shows the two boards, the material used and the connections between them.

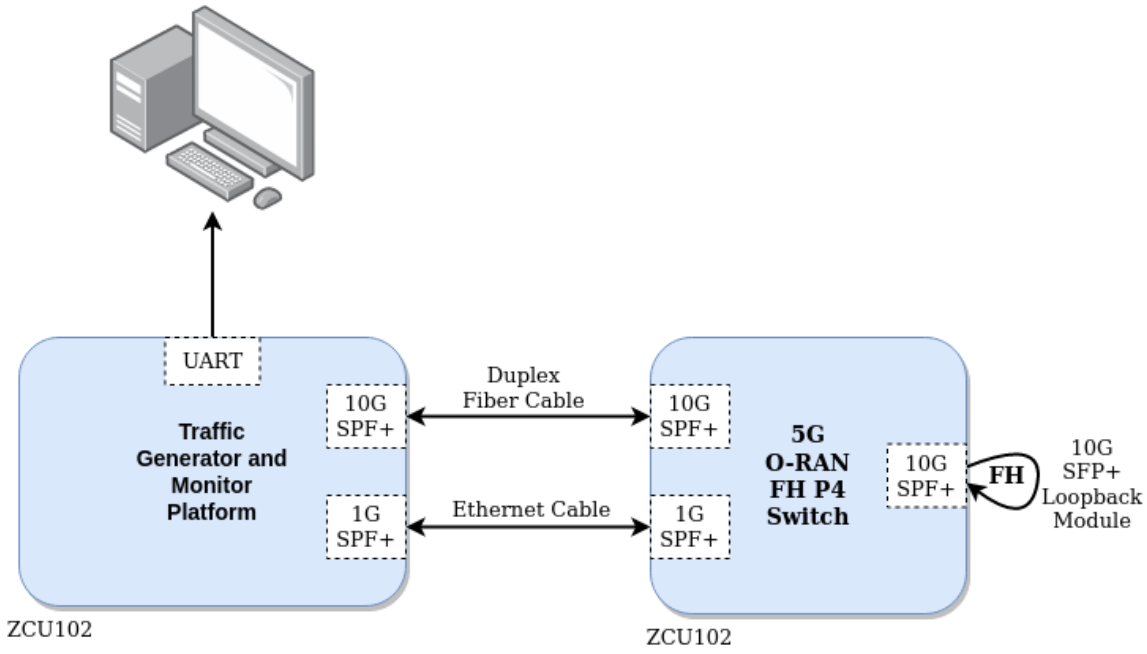


Figure 5.10: Laboratory performance setup block diagram.

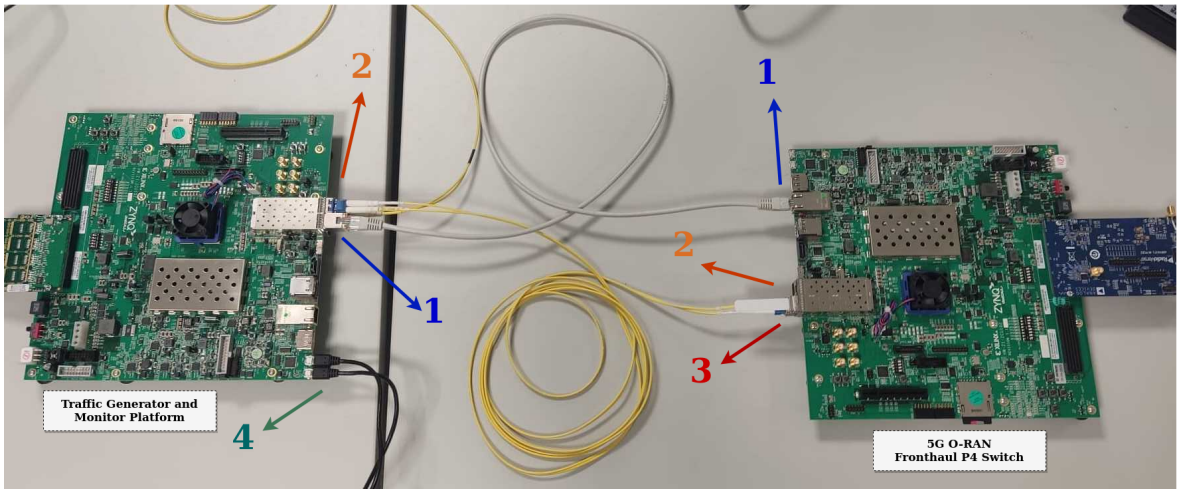


Figure 5.11: Laboratory performance setup. 1) RJ45 connecting the 1G ports of both boards via an Ethernet cable 2) SFP+ connecting the 10G ports of thecomputer and board via an fiber cable. 3) 10G SFP+ Loopback Module 4) USB connecting the board and the computer (UART).

With this setup it is possible to replicate the path of a packet on the fronthaul link with a switch at each end. This happens since each packet will be processed by the switch once in each direction, the loopback represents the physical fronthaul link. With the switch implemented on a board the steps after the platform implementation from the generation of a packet to its reception are as follows:

1. Packets in memory are transferred to the TX FIFO;
2. The user through push buttons allows packets to start being transmitted;
3. Packets are sent through the Ethernet subsystem. Simulating that O-RU/O-DU and general-purpose devices generate the traffic;
4. Whenever a packet is sent, the current timer value is written to memory;
5. The packet is received on the switch via the corresponding interface;
6. Packets from the 10G interface have priority and are processed first;
7. The traffic is sent through the 10G fronthaul interface and as there is a loopback it is received again by the switch. Simulating the transmission from one switch to the other, crossing the fronthaul link;
8. In this direction the traffic is categorized during the processing phase and is sent through the correct port;
9. When the packet arrives back at the platform through the same interface that sent it, a new time is stored in memory. Simulating the reception of information on the other side of the link;
10. After transmission and reception of all packets the times are read, processed and sent to the computer.

5.4 First Results

After performing a test of sending 100 packets of 64 Bytes on each interface, the values shown in graphic 5.12 were obtained. Analysing the graphic it can be seen that the O-RAN packets

are constantly increasing the latency concerning the previous, about 70.4 ns. It is expected that this traffic would have a constant latency since it has priority over 1G traffic. On the other hand, the general-purpose packets have high but constant latency, proving that this interface suffers from starvation because it has the lowest priority. Only the first packet has a lower latency since it was processed in an initial phase where no O-RAN packet was ready to be processed. Performing the test where only O-RAN packets are sent the result obtained is the same. Despite the unexpected latency behaviour, all sent packets were received.

After some analysis and debugging with ILA modules, it was concluded that the fact that there was only one processing module for all interfaces and a fixed priority algorithm in the input arbiter led to this behaviour. This way the switch is only able to process one packet at a time, giving priority to traffic coming from the fronthaul. In this specific case, the O-RAN packets sent from the platform were causing the delay in the following O-RAN packets since after the loopback they had priority in processing. If packets are being received destined for O-RU or O-DU, the switch does not process packets that these units may be sending, making it impossible to install this version of the switch in an O-RAN architecture.

Since these results were not positive for the integration of the switch in the open fronthaul, a study on possible solutions for optimisation, new implementation and testing was done in the next chapter.

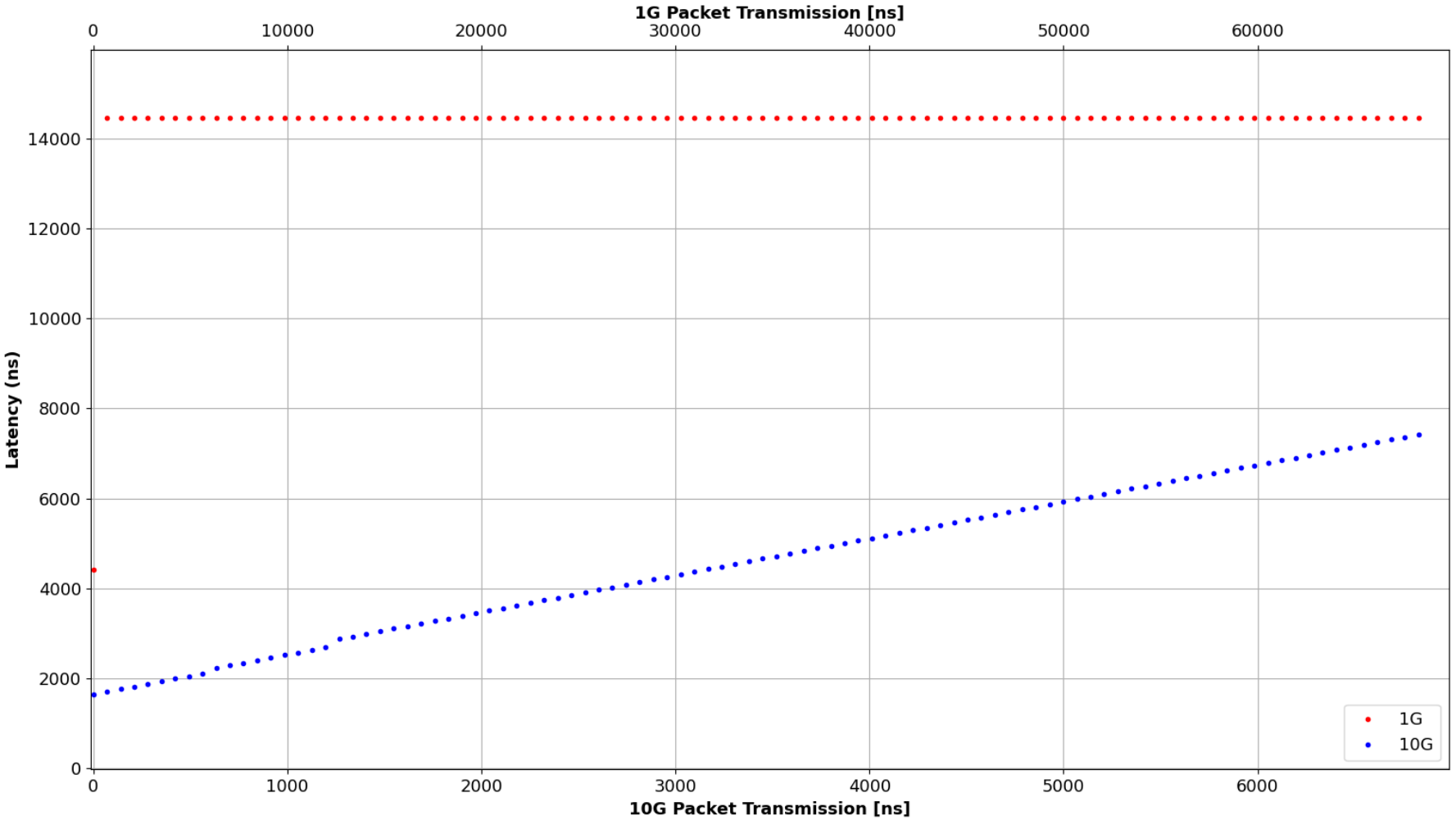


Figure 5.12: Latency of 200 packets with 64 bytes, first version.

CHAPTER 6

Optimization and Final Results

In this chapter are evaluated some solutions to optimize the switch and presented the results of the tests performed.

6.1 Proposed Optimization Solutions

To make the switch installation on the open fronthaul viable with the results obtained in the previous chapter, it was necessary to make changes in the switch architecture and implementation in order to optimize the switch. The changes need to ensure that none of the 10G ports, which receive and send critically and priority O-RAN packets, never experience starvation and ensure that latency is as low as possible. With these objectives three possible approaches were studied:

- Since the problem was related to the scheduling algorithm used, the first solution was to modify the module responsible for this task. Thus, the input arbiter should use a scheduling algorithm that applies a fairness policy between the 10G interfaces while ensuring lower priority for the 1G interface. The algorithm for the 10G interfaces could be Round-Robin or some version of it, and continue with fixed priority between the output of this algorithm with the 1G interface. The architecture of this solution would be the same as the one presented in chapter 4, changing only the implementation of one of the modules.
- Analyzing the problem in another way, the existence of the scheduling algorithm emerges from the need to forward three interfaces to a module that processes packets. Since the switch has no MAC address table, it is possible to increase the number of packet processing modules. With three modules it is possible to dedicate one to each interface and there is no need for an input arbiter. With these characteristics, the proposed architecture would be the one presented in figure 6.1. For a fixed priority algorithm not to be a problem the output arbiter must function as a crossbar switch, allowing for parallel data exchange between interfaces. This solution would mainly take advantage of

the parallelization of processing, one of the advantages of using FPGAs. The latency is reduced and only in the transmission interface to the fronthaul there is the segregation of two flows.

- Based on the previous architecture a new solution emerges restricting the switch even more to the scenario of chapter 4. Since the traffic coming from the 10G and 1G interfaces that are connected to the O-RU/O-DU and the general purpose equipment has to be sent only to the interface connected to the fronthaul. There only needs to be packet processing in the opposite direction to the fronthaul. A new architecture emerges, illustrated in figure 6.2. The switch directions are separated in the architecture, the receive interfaces only have access to the transmit interface required. In theory, this will be the solution with the lowest latency since packets on some interfaces do not need to be processed.

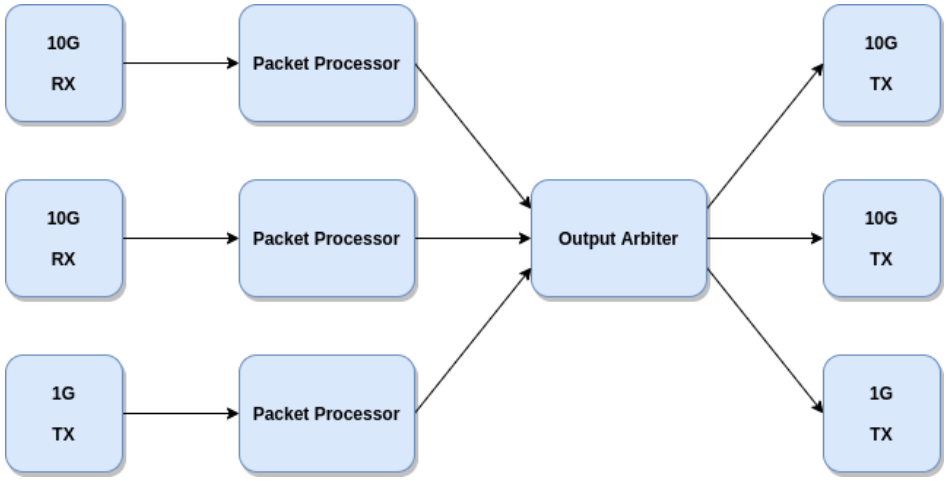


Figure 6.1: Second option architecture.

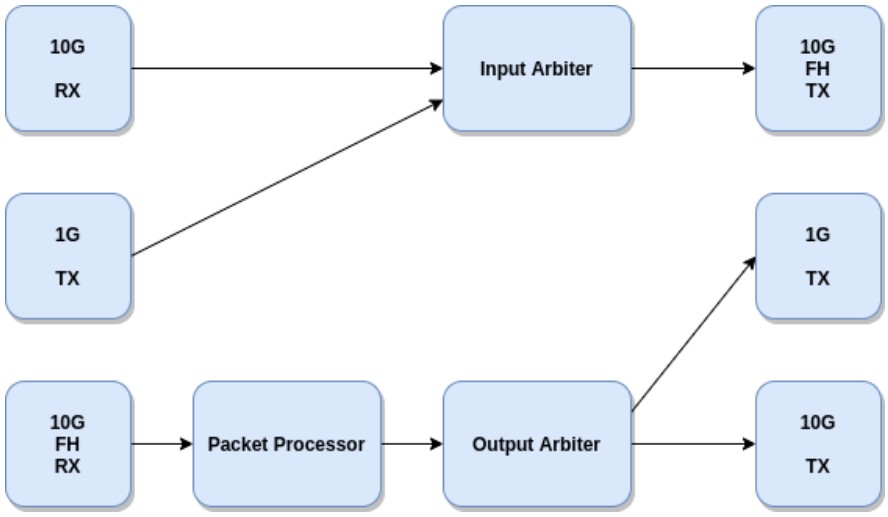


Figure 6.2: Third option architecture.

With the various solutions presented, solution two was chosen since it allowed a reduction in latency because it allows parallel packet processing and does not have a buffer that would add delay to the input of the system. At the same time prevent packet starvation while maintaining the symmetry of the system. It was valued that the traffic arriving at the ports was processed in the same way, maintaining a connection between all of them, allowing the switch to have new functionalities in the future with just the change of the implemented P4 program. Taking advantage of the resources made available in the FPGA, there is now the parallel processing of packets, being able to process packets that are being transmitted in opposite directions, which did not happen before or in the first solution presented.

A new architecture consequently led to a new implementation. The only changes made were in the SDNet clock domain as presented in figure 6.3. In this section of the system, the AXI-Stream Switch that corresponded to the Input Arbiter has been removed. Now all the logic from each RX interface was connected directly to an SDNet module via the 64-bit AXI-Stream interface. Since there are three SDNet modules all loaded with the same P4 program used in the previous version. The last change was made in the AXI-Stream Interconnect, having now three input interfaces required the configuration of the fixed priority algorithm, as in chapter 4.

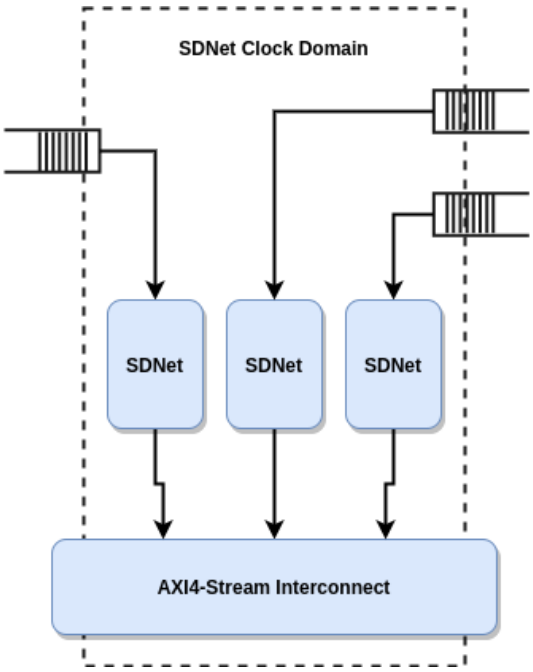


Figure 6.3: Switch high level architecture.

With the new switch implemented, the same functional validation tests were performed as in the previous version, described in chapter 5. The various tests were successful, thus guaranteeing the correct functionality of the new version.

6.2.1 FPGA Resource Utilization

In figure 6.4 it is possible to verify the FPGAs resource usage now of the optimised version. Compared to the previous one there is a greater use of Look up Tables (LUTs) and BRAMs. The difference is not significant and there are still many resources that are not used.

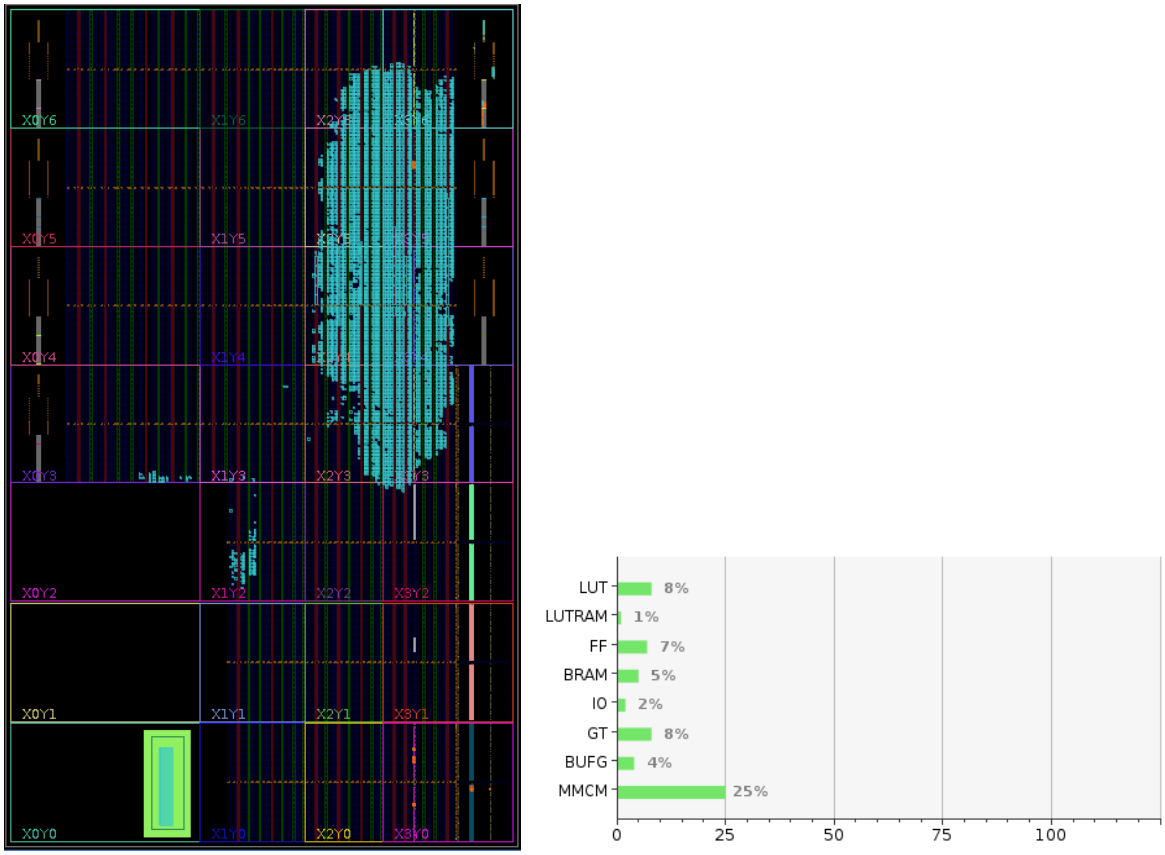


Figure 6.4: ZCU102 FPGA Resource Utilization.

6.3 Tests and Final Results

With the same laboratory setup, new tests were performed with the packet generation platform, obtaining times with different amounts and sizes of packets. Tests were also performed with packet injection in only one of the three interfaces in order to obtain results how the switch behaves without the interference of other packets.

Initially a test was performed where 100 eCPRI packets with 64 bytes were sent over the platform's 10G interface and 100 TCP packets with 64 bytes were sent over the platform's 1G interface. There are some considerations that should be reviewed before analysing the results in relation to the latencies obtained. The RX and TX latency of the core of the Ethernet Subsystem and the GT was not ignored, so it is presented in the results. However, since this latency is fixed during the whole test it does not affect the interpretation of the switch behavior.

The latencies obtained are shown in plot 6.5. Here are the latencies obtained for the 200 packets sent. There are two horizontal axis, the upper one corresponding to the time when

the 1G packets were transmitted (red colour) and the lower one corresponding to the times when the 10G packets were transmitted (blue colour). Since the packet size is equal, the difference between packet transmission times over the 1G link corresponds to values ten times higher than the transmission times over the 10G link. Compared to the previous version of the switch it is possible to verify that the behaviour of the latency, during the transmission of the various packets, has changed. There is no longer a continuous and fixed increase in latency because the 10G port links to RU and DU no longer suffer from starvation, but the latency is constant with some periodic disturbances. Before analysing these disturbances, by analysing the latencies of 1G traffic it is possible to verify that the latency is more than twice that of 10G. Initially, there is a phase where the first 8 packets transmitted are increasing the latency in relation to the previous ones until it remains constant for the rest of the traffic.

Returning to the perturbations in the latency of 10G packets it is possible to verify that the number of occurrences is equal to the number of 1G packets whose latency increases. This relationship between packet and latency increases is illustrated by the arrows in the figure. These disturbances seem to indicate that traffic arriving through the switch's 1G interface has a higher priority than traffic arriving at the 10G interface since they cause increased delay. This is not the case, this happens because the AXI-Stream Switch forwarding packets only allow a transfer between interfaces as they are completely available in order to guarantee that there are no unwanted breaks in the Ethernet Subsystem. As seen previously, this verification is done through the TLAST signal. Therefore, there is a moment in the AXI-Stream Switch input interfaces where a packet coming from the 1G interface has already been fully processed by the SDNet module and the packet referring to the 10G has not yet been. This way it starts to be transferred and even if a packet is ready on the other port with the same destination interface, even with a higher priority, it will have to wait for the current packet to finish. This pause adds latency to the entire switch pipeline and is reflected in every packet that follows. So, the added latency value then corresponds to the transfer between AXI-Stream Switch interfaces of eight 64-bits words. From the first occurrence on, there is a repetition every 10 packets of 10G, which means every 1 packet of 1G.

Further tests were carried out where the same packets were sent but the amount was varied. In figure 6.6 and 6.7 it is possible to visualize the results obtained for sending 30 and 300 packets, respectively. It is then verified that the behaviour obtained previously is replicated independently of the number of packets transmitted. The same perturbations occur more or less often depending on the number of packets sent. However, although the delay for the first packet sent at 1G is the same in all tests, the average and maximum latencies increase depending on the number of packets sent at 10G. These values can be seen in table 6.1.

The latencies represented in the previous plots and obtained from the platform include the latency related to the logic of the Ethernet Subsystems for transmission and reception and two passages through the switch, in different directions. In order to have a more approximate notion of the switch latency, the transmission and reception times of the platform with loopback module in both interfaces were measured first. Thus, for the 10G and 1G interfaces

Type Interface	Traffic	60 packets	200 packets	600 packets
eCPRI 10G	**Minimum**	1280 ns	1280 ns	1280 ns
	Average	1341 ns	1587 ns	2288 ns
	Maximum	1382 ns	1888 ns	3296 ns
TCP 1G	**Minimum**	2936 ns	2936 ns	2936 ns
	Average	3012 ns	3485 ns	4831 ns
	Maximum	3016 ns	3512 ns	4936 ns

Table 6.1: Impact of number of packets on latency.

with packets of 64 Bytes, it obtained times of 262.4 ns and 368 ns, respectively. Subtracting these values from the minimum value obtained for the various tests, which was always the same, the same when sending packets only on one interface, is obtained the times of 1017.6 ns and 2568 ns. Since there were two passages in the switches, the approximate average switch latency will be 508 ns if the packet is received by a 10G interface or 1284 ns if it is received by a 1G interface.

Finally, more tests were performed varying the packet size. These tests were intended to measure the latency that the switches add to the fronthaul without interfering with other packets. Table X shows the results obtained with the transmission and reception logic of the test platform times already subtracted. With some values, it was possible to find the linear regression with the best fit to predict the latency for any size of packets. The X plot represents this latency variance. Considering the O-RAN traffic, i.e. 10G, its latency can vary between 1 μs and 3 μs for the limits of Ethernet frame sizes. The latency of general purpose traffic varies between 2.6μs and 15μs.

Latency	1G	10G
64 Bytes	2.620 μs	1.056 μs
256 Bytes	4.137 μs	1.317 μs
512 Bytes	6.448 μs	1.644 μs
1024 Bytes	10.840 μs	2.463 μs
1518 Bytes	15.088 μs	3.166 μs

Table 6.2: Packet Size Impact on Latency.

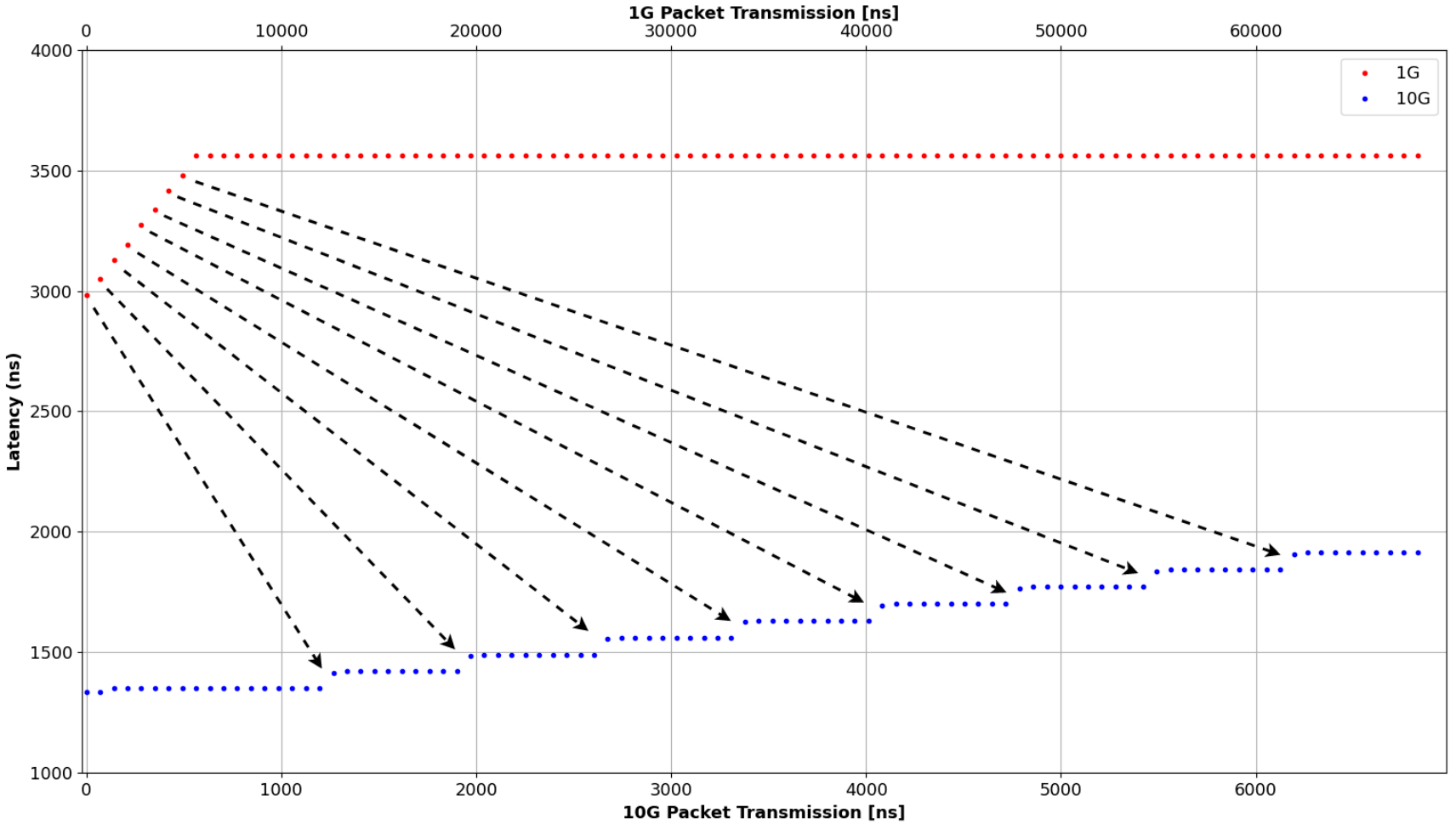


Figure 6.5: Latency of 200 packets with 64 bytes.

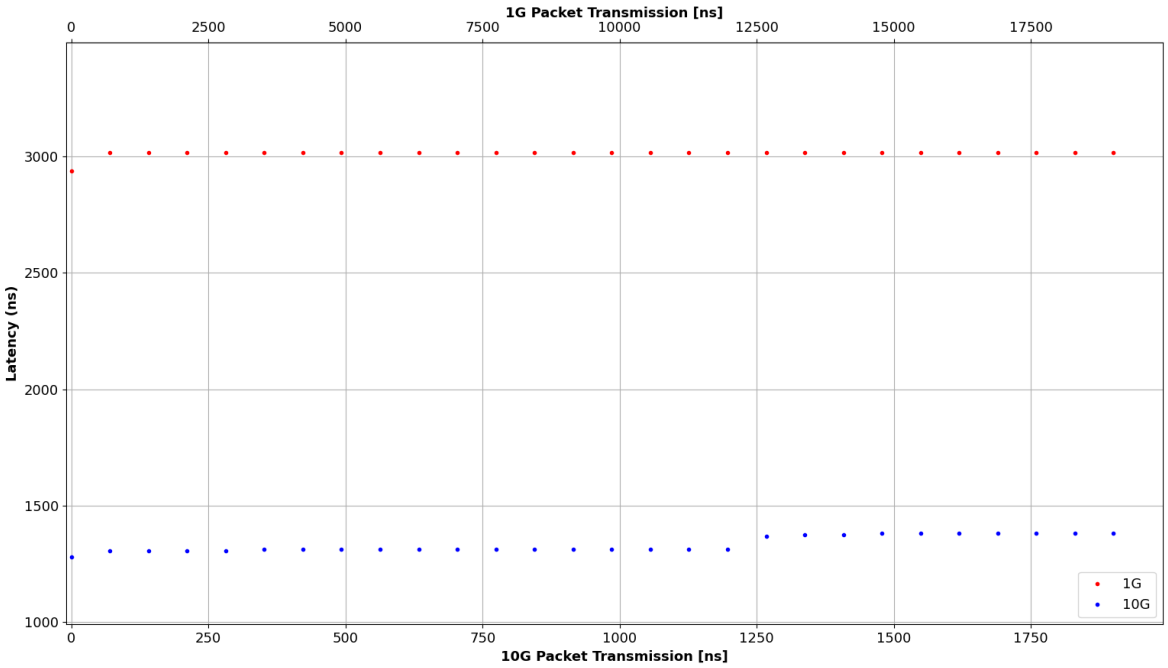


Figure 6.6: Latency of 60 packets with 64 bytes.

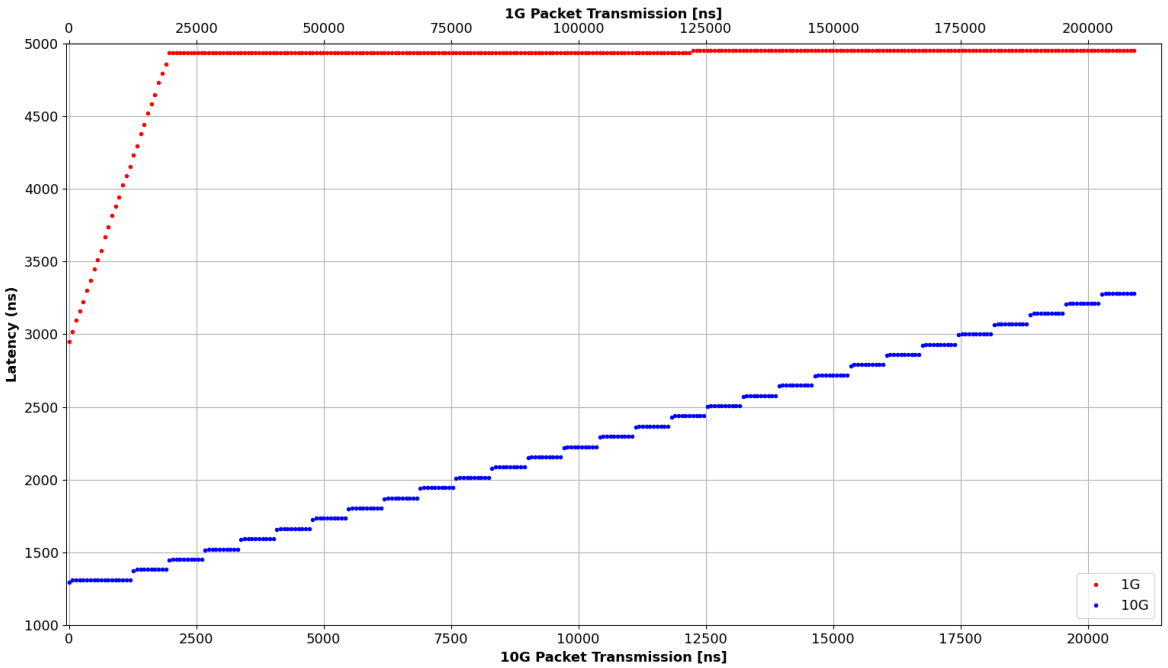


Figure 6.7: Latency of 600 packets with 64 bytes.

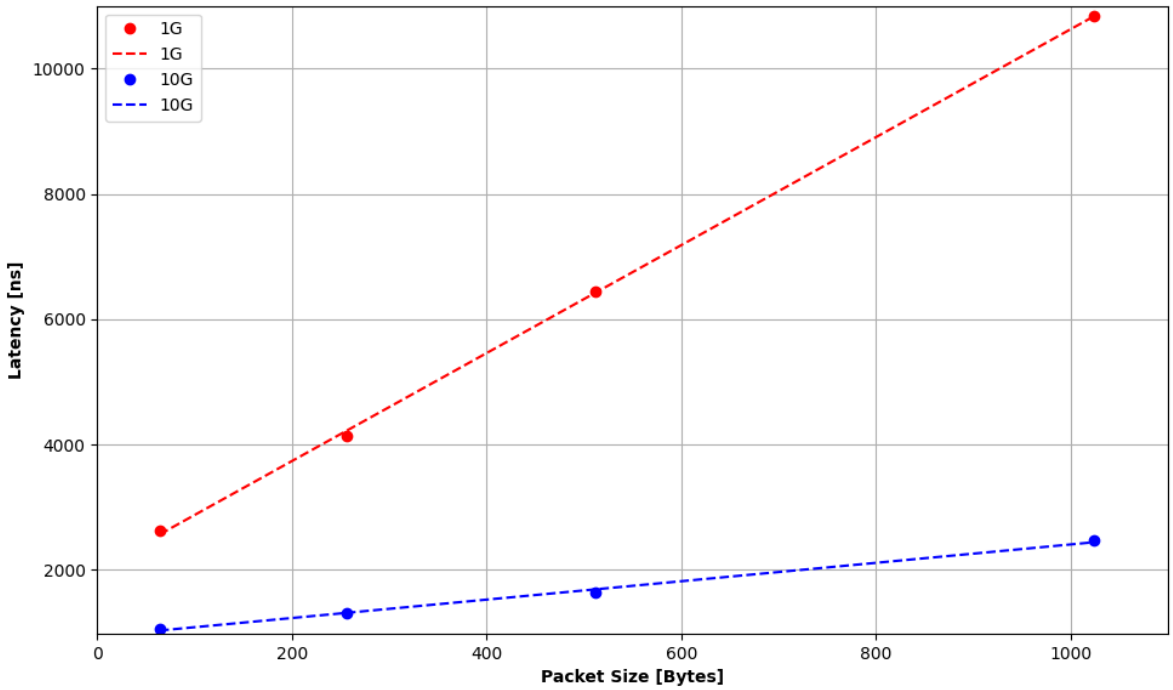


Figure 6.8: Latency varying plot size.

CHAPTER 7

Conclusion and Future Work

In this chapter the conclusions are highlighted and a survey is made for future work.

7.1 Conclusion

The main goal of this dissertation was the design of a 3-port switch capable of combining, in an optical link, general purpose and high priority traffic. Destined to be inserted in both ends of an open fronthaul link the high priority traffic corresponds to the protocol stack of the CUS-Planes. In this context, a proof of concept of a switch that is able to prioritize traffic in one direction and categorize and separate traffic in the other direction was successfully achieved. The biggest limitation of the developed switch is that the functionalities are only restricted to Control and User plane traffic. Regarding Sync Plane, since it uses general protocols such as PTP and SyncE it was not possible to ensure that they correspond to packets destined for O-RAN units without having a MAC address table. Since the switch does not have the ability to learn new addresses automatically, it was developed with the goal of being placed in the fronthaul knowing only the type of traffic beforehand.

The initial phases of the work involved a study and familiarization of the requirements and characteristics of O-RAN, more concretely on the open fronthaul. Thus, a study was carried out on the evolution of access networks until reaching the current 5G paradigm. Since O-RAN is based on principles of virtualization, openness and standardisation, it was decided to use the P4 language for programming the switch data plane. Using an SDN controller alongside the switch was never an initial goal.

The switch was developed on FPGA-based development kits from Xilinx based on IP cores and VHDL custom modules. The main components are the Ethernet solutions used and the packet processing module. Besides the functional validation, a platform for traffic generation and monitoring was developed to test the latency introduced in the fronthaul. This platform was developed in the same development kit as the switch.

The first version of the switch successfully passed the functional tests, but during the latency tests, an undesired latency increase was obtained, which put at risk the timing requirements of the open fronthaul. The constant increase in packet latency over the previous ones arose because there was only one processing unit and the input arbiter was configured with a fixed priority scheduling algorithm. Thus, if the switch is processing packets arriving from the fronthaul, the packets coming from the O-RU/O-DU suffer from starvation.

Therefore, an optimization was needed in the switch that led to a new architecture. The system now has a processing module for each port, solving the previous problem. Tests on the new version confirmed the correct operation of the switch, where only the 1G port suffers from starvation. However, when a packet is being forwarded between interfaces by the switch, if a packet is being transferred from the 1G port, the other packets will have to wait for it to finish even though they have a higher priority. This particular case adds latency to the switch pipeline for the 10G ports. That said, the latencies obtained during the tests seem favourable to the installation in the fronthaul, but since the timing requirements are not fixed, everything depends on the units used in the RAN and on the network architecture. Considering the O-RAN traffic, which is critical and has to guarantee low latency, the added latency varies between 1 μs and 3 μs depending on the packet size. These values are very encouraging since the relative error between RU and DU should be less than 3 μs and the latency of the fronthaul should not exceed 100 μs. However more exhaustive testing should be done and tested in a real environment.

7.2 Future Work

From the study and the results obtained, implementation possibilities emerged to solve known problems and optimize the current solution. Thus, several points emerge as possible lines of future work:

- Integrate the functionalities of the S-Plane traffic switch. Implement and test a solution where the PTP and SyncE packets arriving through the fronthaul link are replicated, being sent to both ports;
- Implementation and validation of the remaining solutions presented to optimize the initial version of the switch. Analyse the various possibilities, already existing or developing a custom one, of packet scheduling algorithms that meet the necessary requirements of the input arbiter;
- Perform more intensive tests with different sizes and numbers of packets. Test and analyse the behaviour of latencies when link occupancy is varied, such as when the delay is added between bursts of packets or when the delay is added between packets;
- Installation of the switch on a testbed for testing in a real environment.

APPENDIX A

10G/25G Ethernet Subsystem Configuration

The following figures contain the configurations used in the generation and customization of the 10G/25G Ethernet Subsystem. The Configuration, MAC Options and GT Selection and Configuration tab provides the basic core configuration options, additional core configuration options and enables the configuration of the serial transceiver features of the core, respectively. For further information please consult the documentation provided on the 10G/25G High Speed Ethernet Subsystem Product Guide [34].

In the GT Selection and Configuration Tab, the values in the Core to GT Association section vary according to the board used and the SFP modules location. In the case of the ZCU102 Evaluation Board the GT associated with the SFP cage is in Quad X1Y3 and the association of each module to the lane is the following:

- Top Right - X1Y12;
- Bottom Right - X1Y13;
- Top Left - X1Y14;
- Bottom Right - X1Y15;

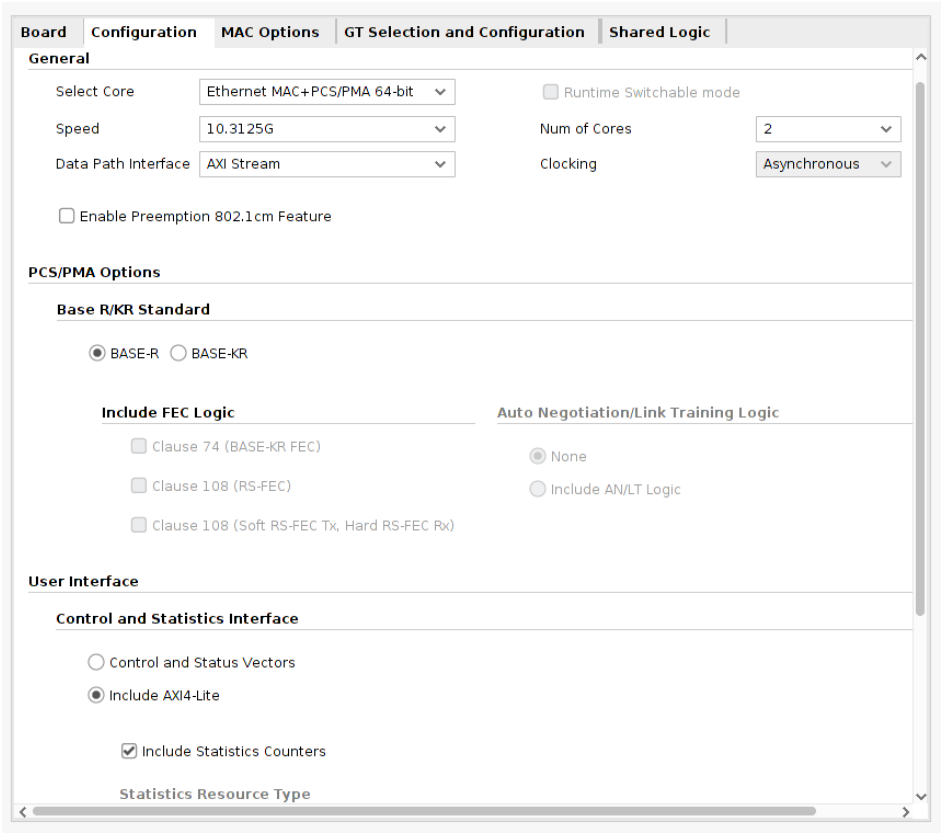


Figure A.1: 10G/25G Ethernet Subsystem - Configuration Tab.

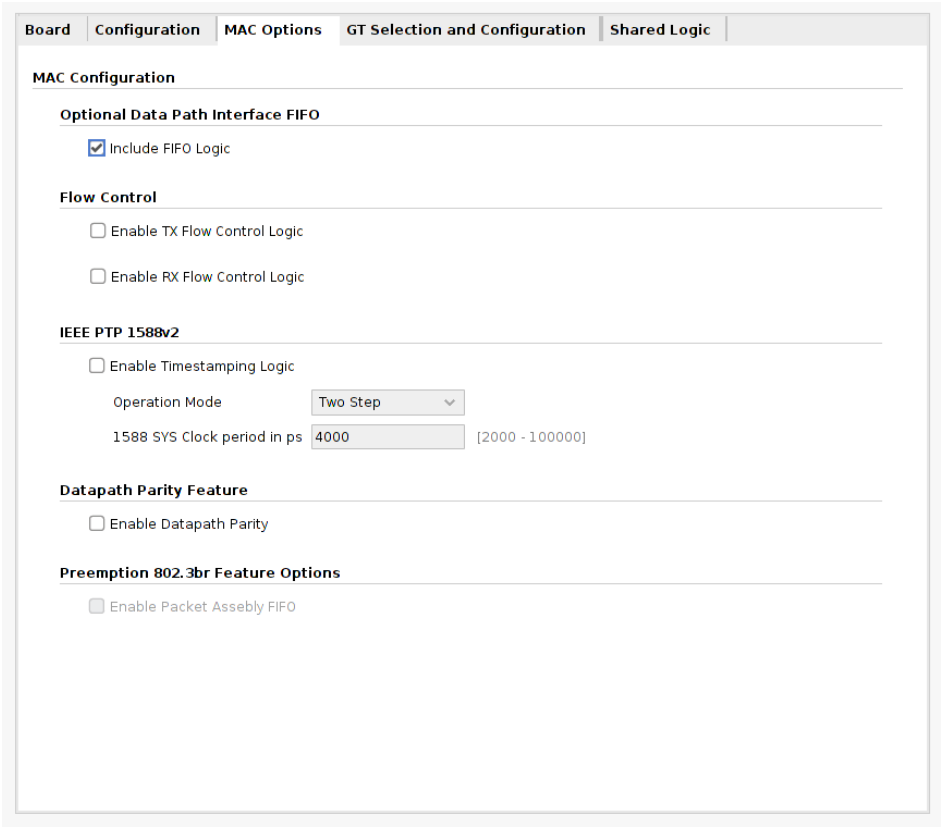


Figure A.2: 10G/25G Ethernet Subsystem - MAC Options Tab.

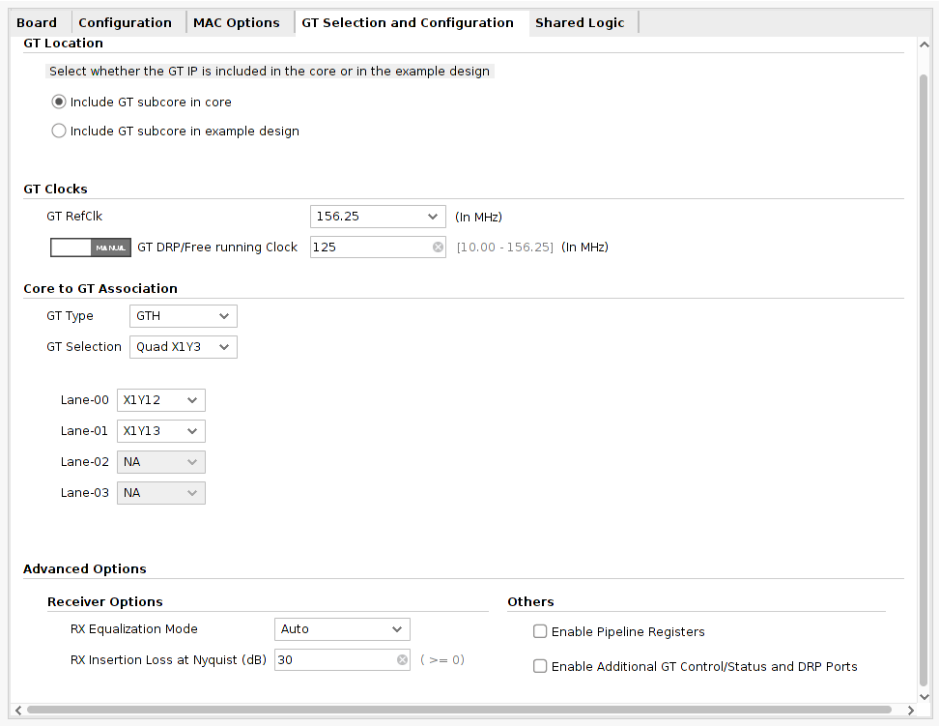


Figure A.3: 10G/25G Ethernet Subsystem - GT Selection and Configuration Tab.

Figure A.4: 10G/25G Ethernet Subsystem High Level Block.

APPENDIX B

AXI4-Stream Switch Configuration

The next figure contain the configurations used in the generation and customization of the AXI4-Stream Switch. With the use of this module it was pretended to use fixed priority as the scheduling algorithm, where the transaction level arbitration must respect the TLAST as the packet boundary. It was not intended to have a transfer limit, thus allowing the starvation of interfaces.

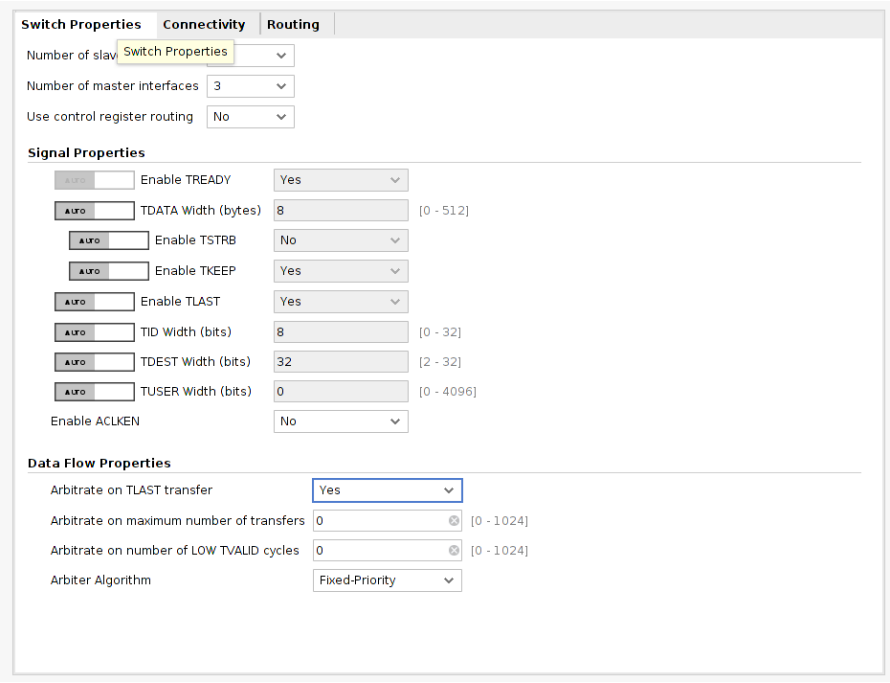


Figure B.1: AXI4-Stream Switch Configuration.

References

[1] A. Gupta and R. K. Jha, «A Survey of 5G Network: Architecture and Emerging Technologies», *IEEE Access*, vol. 3, pp. 1206–1232, 2015. DOI: 10.1109/ACCESS.2015.2461602.

[2] Richard Galazzo, *Timeline From 1G to 5G: A Brief History on Cell Phones*, 2021. [Online]. Available: https://www.cengn.ca/information-centre/innovation/timeline-from-1g-to-5g-a-brief-history-on-cell-phones/ (visited on 10/27/2021).

[3] Patrick Marsch, Omer Bulakci, Olav Queseth, and Mauro Boldi, *5G System Design: Architectural and Functional Considerations and Long Term Research*. 2018, pp. 1–551, ISBN: 9781119425120. [Online]. Available: https://ieeexplore.ieee.org/book/8367991 (visited on 10/27/2021).

[4] ITU-R, «Minimum requirements related to technical performance for IMT-2020 radio interface(s)», *Working Party 5D*, vol. November, no. Report ITU-R M.2410-0, pp. 1–11, 2017. [Online]. Available: http://www.itu.int/ITU-R/go/patents/en.

[5] 3. G. P. P. (3GPP), «Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G; Release description; Release 15», *3rd Generation Partnership Project (3GPP), Tech. Rep. 21.915 version 15.0.0*, pp. 1–120, 2019. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3389.

[6] ——, «TS 138 300 - V16.2.0 - 5G; NR; NR and NG-RAN Overall description; Stage-2 (3GPP TS 38.300 version 16.2.0 Release 16)», 2020.

[7] GSMA, «The Mobile Economy 2020», Tech. Rep., 2020. [Online]. Available: https://www.gsma.com/mobileeconomy/.

[8] M. A. Habibi, M. Nasimi, B. Han, and H. D. Schotten, «A Comprehensive Survey of RAN Architectures Toward 5G Mobile Communication System», *IEEE Access*, vol. 7, pp. 70 371–70 421, 2019. DOI: 10.1109/ACCESS.2019.2919657.

[9] 3GPP, «TS 138 401 - V15.3.0 - 5G; NG-RAN; Architecture description (3GPP TS 38.401 version 15.3.0 Release 15)», pp. 0–39, 2018. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3219.

[10] E. Dahlman, S. Parkvall, and J. Sköld, *5G NR: The Next generation wireless Access technology*. Elsevier, Aug. 2018, pp. 1–466, ISBN: 9780128143247. DOI: 10.1016/C20170013472.

[11] T. Report, «TR 138 912 - V14.0.0 - 5G; Study on New Radio (NR) access technology (3GPP TR 38.912 version 14.0.0 Release 14)», pp. 0–76, 2017. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3059.

[12] S. Ahmadi, *5G NR: Architecture, technology, implementation, and operation of 3GPP new radio standards*. Elsevier, Jan. 2019, pp. 1–960, ISBN: 9780081022672. DOI: 10.1016/C2016-0-04944-6.

[13] 3GPP, «3GPP TS 38.470 V16.0.0 NG-RAN; F1 general aspects and principles Release 16», Tech. Rep., 2019. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3257.

[14] ITU-T and S. Stephen, «Transport network support of IMT-2020/5G», *ITU-T Technical Report*, no. February 2018, 2020. [Online]. Available: https://www.itu.int/en/myitu/Publications/2020/02/28/12/47/Transport-network-support-of-IMT-2020-5G---FEB.

[15] S. Sirotkin, *5G Radio Access Network Architecture: The Dark Side of 5G*. Wiley, 2021, ISBN: 978-1-119-55088-4. DOI: 10.1002/9781119550921.

[16] 3GPP, «Specification 38.801 - Study on new radio access technology: Radio access architecture and interfaces (Release 14)», Tech. Rep. Release 14, 2017, p. 91. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3056.

[17] O. Mocerino, G. S. Architect, and F. N. Communications, «5G Backhaul/Fronthaul Opportunities and Challenges - NCTA Technical Papers», Tech. Rep., 2019. [Online]. Available: https://www.nctatechnicalpapers.com/Paper/2019/2019-5g-backhaul-fronthaul-opportunities-and-challenges.

[18] O-RAN Alliance, *About O-RAN ALLIANCE — O-RAN ALLIANCE*, 2018. [Online]. Available: https://www.o-ran.org/about (visited on 11/15/2021).

[19] ——, «O-RAN: Towards an Open and Smart RAN», *O-RAN Alliance*, no. October, p. 19, 2018.

[20] NTT DOCOMO, «5G Open RAN Ecosystem Whitepaper», 2021. [Online]. Available: https://www.nttdocomo.co.jp/binary/pdf/corporate/technology/whitepaper_5g_open_ran/OREC_WP.pdf.

[21] ORAN Alliance, «O-RAN use cases and deployment scenarios: Towards open and smart RAN», Tech. Rep., 2020, pp. 1–21. [Online]. Available: https://www.o-ran.org/resources.

[22] O-RAN ALLIANCE, «O-RAN Minimum Viable Plan and Acceleration towards Commercialization», Tech. Rep., 2021. [Online]. Available: https://static1.squarespace.com/static/5ad774cce74940d7115044b0/t/60f9b144abdc902712f43475/1626976585796/O-RAN+Minimum+Viable+Plan+and+Acceleration+towards+Commercialization+White+Paper+29+June+2021.pdf.

[23] A. Umesh, T. Yajima, T. Uchino, and S. Okuyama, «Overview of O-RAN Fronthaul Specifications», *NTT DOCOMO Technical Journal*, vol. 21, no. 1, pp. 46–59, 2019, ISSN: 09240136.

[24] Ericsson, Huawei, Nokia, and N. C. And, «eCPRI Specification V1.0 Common Public Radio Interface: eCPRI Interface Specification», pp. 1–62, 2017. [Online]. Available: http://www.cpri.info/spec.html%20http://www.cpri.info/spec.html%5C%0Ahttp://www.cpri.info/downloads/eCPRI_v_1_0_2017_08_22.pdf.

[25] O-RAN Alliance, «O-RAN Fronthaul Working Group: Control , User and Synchronization Plane Specification v4», pp. 1–253, 2019.

[26] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, «A survey of software-defined networking: Past, present, and future of programmable networks», *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014, ISSN: 1553877X. DOI: 10.1109/SURV.2014.012214.00180.

[27] W. Stallings, *5G Wireless: A Comprehensive Introduction*. 2021, ISBN: 9780136767145. [Online]. Available: www.pearson.com/permissions/..

[28] Open Networking Foundation, «Software-Defined Networking: The New Norm for Networks [white paper]», *ONF White Paper*, pp. 1–12, 2012.

[29] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, «P4: Programming protocol-independent packet processors», *Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014, ISSN: 19435819. DOI: 10.1145/2656877.2656890. arXiv: 1312.1719.

[30] The P4 Language Consortium, «P4.16 Language Specification», Tech. Rep., 2020. [Online]. Available: https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html%20https://p4.org/p4-spec/docs/P4-16-v1.2.1.pdf.

[31] Xilinx, «Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit», 2020. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/ek-u1-zcu102-g.html (visited on 11/15/2021).

[32] Xilinx Inc., «Integrated Logic Analyzer v6.2», *Xilinx*, vol. PG172, p. 31, 2016. [Online]. Available: www.xilinx.com%20https://www.xilinx.com/support/documentation/ip_documentation/ila/v6_2/pg172-ila.pdf.

[33] ——, «10G/25G High Speed Ethernet Subsystem v2.5 Product Guide Vivado Design Suite», Tech. Rep., 2020. [Online]. Available: www.xilinx.com.

[34] ——, «10G/25G High Speed Ethernet Subsystem v3.3 Product Guide», 2020. [Online]. Available: www.xilinx.com.

[35] Xilinx Corp., *VCU108 Evaluation Board - User Guide*, 2015. [Online]. Available: www.xilinx.com.

[36] T. Instruments, *DP83867IR Industrial temperature, robust gigabit Ethernet PHY transceiver*. [Online]. Available: https://www.ti.com/product/DP83867IR#tech-docs (visited on 11/09/2021).

[37] Xilinx Inc., «Zynq UltraScale + Device - Technical Reference Manual», *Ug1085*, vol. 1085, pp. 1–1181, 2018. [Online]. Available: www.xilinx.com.

[38] ——, *AXI4-Stream Infrastructure IP Suite v3.0 LogiCORE IP Product Guide*, 2018. [Online]. Available: www.xilinx.com.

[39] Xilinx, *Standalone Ethernet Driver - Xilinx Wiki - Confluence*. [Online]. Available: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842012/Standalone+Ethernet+Driver (visited on 12/02/2021).

[40] Packeth, *Packeth*. [Online]. Available: http://packeth.sourceforge.net/packeth/Home.html (visited on 12/02/2021).

[41] F. Technology, «FM-S14 Quad SFP/SFP+ transceiver FMC», [Online]. Available: www.fastertechnology.com.

[42] Xilinx, *Block Memory Generator*, 2009. [Online]. Available: www.xilinx.com (visited on 11/15/2021).