Tampere University

Jhonny Villota Coral

HYBRID POWER ESTIMATION FOR TELECOMMUNICATION SOCS ON EARLY DESIGN STAGES

Master of Science Thesis
Faculty of Information Technology and Communication Sciences
Examiners: Dr. Taneli Riihonen
Dr. Joonas Säe
October 2021

ABSTRACT

Jhonny Villota Coral: Hybrid Power Estimation for Telecommunication SoCs on Early Design Stages
Master of Science thesis
Tampere University
Master's Degree Programme in Electrical Engineering
Major in Wireless Communications and RF Systems
October 2021

Modern mobile networks require high-performance and low-power baseband processing systems. Those digital systems are designed as System-on-Chip (SoC), integrated circuits comprising billions of transistors into a single chip. The baseband processing SoCs are composed of several power-hungry engines such as the Layer-1 processing subsystem. That subsystem performs essential tasks for modern multicarrier and multiantenna techniques. Each task is executed in individual Intellectual Property (IP) blocks, independently developed, and progressively integrated into the subsystem. The most power-consuming functionalities of the subsystem are IFFT/FFT for OFDM symbol generation, decimation for the Physical Random-Access Channel (PRACH) signal extraction, sub-band filtering for mixed numerology carrier support, and Physical Resource Block (PRB) compression and decompression. The convergence of such high-computing processing tasks and multiple technologies into a single chip continuously increases the SoC power dissipation. Therefore, the power consumption is a crucial parameter on SoC design and must be estimated and tracked as early as possible in the design process to mitigate the problem through power optimizations. Nonetheless, the maturity of individual IP blocks at early design stages differs and generally does not include the final intended functionalities, which leads to inaccurate power estimates.

The main objective of this thesis is to simulate and model the power consumption of a Layer-1 subsystem which is part of a Digital Front-End (DFE) SoC. The subsystem is a high-performance 4G/5G baseband processing accelerator for Layer-1 in the 3GPP base station functional stack. The subsystem power estimation for different FDD/TDD test cases is calculated using individual IP power simulations in different modes of operation. The baseline IP power simulations were carried out at the Register-transfer level (RTL) and repeated at various design stages. In addition, gate-level simulations were also used in the latest design stage to calibrate the power model. The latest gate-level simulations have more design information; thus, they are considered the closest to the real results and are used to compare the early RTL power simulations and the power model.

In the worst case, using the first-round results of RTL simulations and without calibration, the power model produced a mean absolute error of 4.1% compared to the latest gate-level results. However, results also show that error decreases with calibration and as the design maturity progresses. In addition, a simple spreadsheet-like tool was developed to quickly estimate the subsystem power consumption for different test cases and processing capacities, allowing designers, integrators, and system architects to perform estimates without requiring new power simulations for each scenario. Finally, considering the concept of reuse of Intellectual Property (IP) blocks, the power database built serves as an accurate starting point for future similar projects.

Keywords: SoC, ASIC, Low-Power Design, RTL, Baseband, DFE, Mobile Networks.

PREFACE

This thesis was done at the Baseband and SoC Development Unit of Nokia Mobile Networks in Tampere, Finland. I would like to express my gratitude to my colleagues at the L1Low team, especially to Tuomas Järvinen, for his guidance and advice to finish this project, and my line manager Sakari Patrikainen, who allowed me to combine studies, work, and personal development.

I would also like to thank Dr. Taneli Riihonen and Dr. Joonas Säe for examining this thesis and giving me valuable feedback during the writing process.

I want to express my infinite gratitude to my parents, Nelson and María Eugenia, and my sister Stephanie for always being my support despite the distance. Finally, I would like to thank W.H. for being light amid the darkness.

Thanks to all those who, at some point, were part of this.

Tampere, 14th October 2021

Jhonny Villota

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

3GPP	3rd Generation Partnership Project
5GCN	5G Core Network
ADC	Analog-to-Digital
AFE	Analog Front End
AGC	Automatic Gain Control
AI	Artificial Intelligence
AMF	Access and Mobility Management Function
AMPS	Advanced Mobile Phone System
ANN	Artificial Neural Networks
ARQ	Automatic Repeat Request
ASIC	Application-Specific Integrated Circuit
ASIP	Application-Specific Instruction-Set Processors
BBU	Baseband Unit
BDMA	Beam-Division Multiple-Access
CD	Continuous Delivery
CDMA	Code-Division Multiple Access
CI	Continuous Integration
CMOS	Complementary Metal–Oxide–Semiconductor
CN	Core Network
CU	Central Unit
C-RAN	Cloud Radio Access Network
DAC	Digital-to-Analog
DFE	Digital Front End
DL	Downlink
DPD	Digital Pre-Distortion
DSP	Digital Signal Processor
DU	Distributed Unit
EDA	Electronic Design Automation
EDGE	Enhanced Data rates for GSM Evolution
eMBB	Enhanced Mobile Broadband
EPC	Evolved Packet Core
ES	Engineering Samples
FDD	Frequency-Division Duplexing
FDMA	Frequency-Division Multiple Access
FEC	Forward Error Correction
FFT	Fast Fourier Transform
FinFET	Fin Field-Effect Transistor
FLPA	Functional-Level Power Analysis
FPGA	Field-Programmable Gate Array
FSDB	Fast Signal Data Base
FSM	Finite State Machine
GAA	Gate-All-Around
gNB	New generation Node B
GPP	General Purpose Processor
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HDL	Hardware Description Languages
HM	Hard Macro
HPA	High Power Amplifier
HSPA	High-Speed Packet Access
HW	Hardware
IC	Integrated Circuit

ICO	Current-Controlled Oscillator
IF	Intermediate Frequency
IFFT	Inverse Fast Fourier Transform
IoT	Internet of Things
IP	Intellectual Property
L1	Layer-1
L2	Layer-2
L3	Layer-3
LDPC	Low-Density Parity Check
LO	Local Oscillator
LNA	Low-Noise Amplifier
LTE	Long-Term Evolution
LUT	Lookup table
M2M	Machine-to-Machine
MAC	Medium-Access Control
MIMO	Multiple-input and Multiple-output
ML	Machine Learning
MNO	Mobile Network Operator
mMTC	Massive Machine Type Communication
NCO	Numerically Controlled Oscillator
NFV	Network Function Virtualization
NMT	Nordic Mobile Telephony
NR	New Radio
NSA	Non-Standalone
O-RAN	Open-RAN
OFDM	Orthogonal Frequency Division Multiplexing
OPEX	Operational Expenses
PAPR	Peak-to-Average Power Ratio
PDCP	Packet Data Convergence Protocol
PHY	Physical Layer
PSM	Power State Machine
PSS	Processor Subsystem
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio-Access Technology
RF	Radio Frequency
RLC	Radio-Link Control
RRC	Radio Resource Control
RRH	Remote Radio Head
RTL	Register-Transfer Level
RU	Radio Unit
SA	Standalone
SDAP	Service Data Adaptation Protocol
SDN	Software Defined Network
SHE	Self-Heating Effect
SMF	Session Management Function
SMS	Short Message Service
SNR	Signal-to-Noise Ratio
SoC	System-on-Chip
SS	Subsystem
SW	Software
TDD	Time-Division Duplexing
TDMA	Time-Division Multiple Access
TD-SCDMA	Time-Division Synchronous Code-Division Multiple Access
TLM	Transaction Level Modelling

TTM	Time-to-Market
UART	Universal Asynchronous Receiver-Transmitter
UE	User Equipment
UL	Uplink
UMTS	Universal Mobile Telecommunications System
UPF	User-Plane Function
URLLC	Ultra-Reliable Low Latency Communications
VCO	Voltage-Controlled Oscillator
VHDL	Very High-Speed Integrated Circuits Hardware Description Language
WCDMA	Wideband Code-Division Multiple Access
WiMAX	Worldwide Interoperability for Microwave Access
WLM	Wire Load Model

c_{ACT}	Time spent in active mode (normalized)
c_{HLT}	Time spent in halt mode (normalized)
c_{IDL}	Time spent in idle mode (normalized)
C_{L}	Parasitic capacitances
D_{IP-ACT_i}	IP design maturity coefficient for active mode
D_{IP-HLT_i}	IP design maturity coefficient for halt mode
D_{IP-IDL_i}	IP design maturity coefficient for idle mode
f_{clk}	Clock frequency
I_{G}	Gate-oxide tunnelling leakage current
I_{GILD}	Gate-induced drain leakage current
I_{PT}	Punchthrough leakage current
I_{RB}	Reverse-biased diode leakage current
I_{short}	Short circuit current
I_{ST}	Subthreshold leakage current
I_{static}	Static current
N_W	Number of training weights
P	Total average power
P_{dyn}	Dynamic power
$P_{dyn-ACT}$	Dynamic power in active mode
$P_{dyn-HLT}$	Dynamic power in halt mode
$P_{dyn-IDL}$	Dynamic power in idle mode
P_{IP-ACT_i}	IP power in active mode
P_{IP-AVG}	IP average power
P_{IP-HLT_k}	IP power in halt mode
P_{IP-IDL_j}	IP power in idle mode
P_{leakage}	Leakage power
P_{short}	Short circuit power
P_{SS-AVG_calib}	Calibrated subsystem average power consumption
P_{SS-MAX}	Subsystem maximum power consumption
P_{static}	Static power
$P_{\mathrm{switching}}$	Switching power
P_T	Total average power
$\mathrm{P_T}$	Random power over an interval
Q_{short}	Average charge per output transition
t_{TDD-DL}	TDD time spent in downlink mode (normalized)
t_{TDD-UL}	TDD time spent in uplink mode (normalized)
V_{dd}	Source voltage
α	Switching activity factor
β	Expected accuracy

1. INTRODUCTION

This chapter introduces the background and motivation of the thesis in Section 1.1. Then, the objectives and scope are introduced in Section 1.2. Next, a brief revision of thesis results and observations are presented in Section 1.3. Finally, the structure of the thesis is unveiled in section 1.4.

1.1 Background and Motivation

The growing demand for wireless connectivity makes mobile network base stations a key component of telecommunication system architectures of actual society. Base stations require complex digital systems to process the high and constantly growing amounts of data generated by humans and machines. Those complex systems are generally designed as System-on-Chip (SoC) and are primarily used for radio and baseband processing, as is shown in Figure 1. SoCs have different subsystems to process data depending on the transmission stage and layer. These subsystems are generally designed by integrating Intellectual Property (IP) blocks, which are designed to perform particular tasks and can be reused in other subsystems.

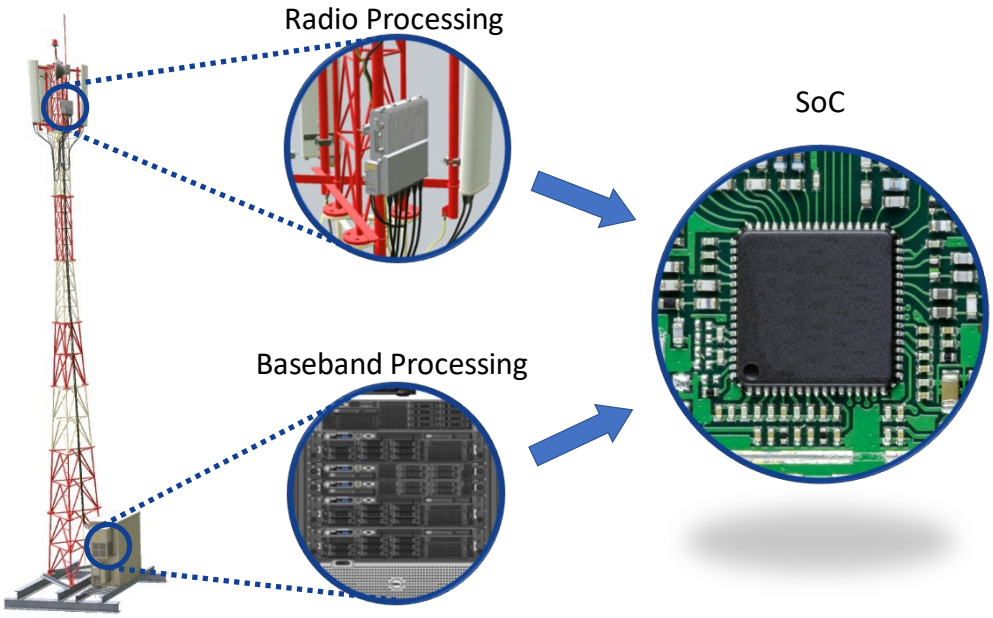


Figure 1. *System-on-Chip in a typical radio base station.*

The design of SoCs for baseband processing is part of a long and complex process that requires a considerable amount of time and resources. This chipset system design pro-

cess starts by collecting high-level requirements from different sources to define the reference and target architecture of the new system. The IP and SoC system design are continuous processes that start by studying the latest industry-standard releases like the 3rd Generation Partnership Project (3GPP) and modelling them using high-level programming languages. At the same time, the design or modification of required IPs is performed by using Hardware Description Languages (HDL). Research and development are conducted continuously, allowing technical and management teams to decide changes alongside the process. The chipset system design process can take several years, from the first concept to the final product ready to the market.

Among the various SoC design challenges, low power consumption is one of the most important, as high-power consumption may cause chip failure, performance issues and higher costs on packaging and cooling systems. In wireless systems, there is a trade-off between complexity and power consumption of transceivers and antennas [1]. Consequently, power optimization is a critical factor for advanced digital baseband systems, considering baseband processors require a significant number of power-hungry blocks that can jeopardize the low-power chipset targets.

The previous step to power optimization is power estimation, which defines how much power is expected the SoC consumes in several functionality cases. Power estimation methods are efficient substitutes to real measurements, as they do not require a characterization step, allowing a quick power exploration in the design [2]. The first power estimation is merely referential, and it is based on the power consumption of previous similar chipsets. However, realistic power estimates are available only in the later phases of the SoC design process when it is extremely difficult to change sub-blocks to get significant power savings. On the other side, the highest power optimization opportunities can be achieved by performing more accurate power estimations as early as possible during the design process. The problem with early power estimations arises when most of the IP blocks that compose the SoC are designed from scratch and do not have any prior information regarding power consumption.

Therefore, the primary motivation of this thesis is to study different methods for early power estimation, taking advantage of state-of-the-art simulation tools used by the industry chipset manufacturers at different design abstraction levels. Lastly, to compare simulation results throughout a typical subsystem design process and define a general power model using early IP simulations as a baseline to minimize percentage errors.

1.2 Objectives and Scope

As seen in Figure 2, there are several design abstraction levels at which simulations can be performed. However, Register-Transfer Level (RTL) and gate-level are the leading interest for power simulations. Gate-level power simulations are more accurate but require more running time and a higher design maturity than RTL simulations. In that sense, RTL estimations can be used at earlier design stages to shed light on power-saving opportunities at architectural levels.

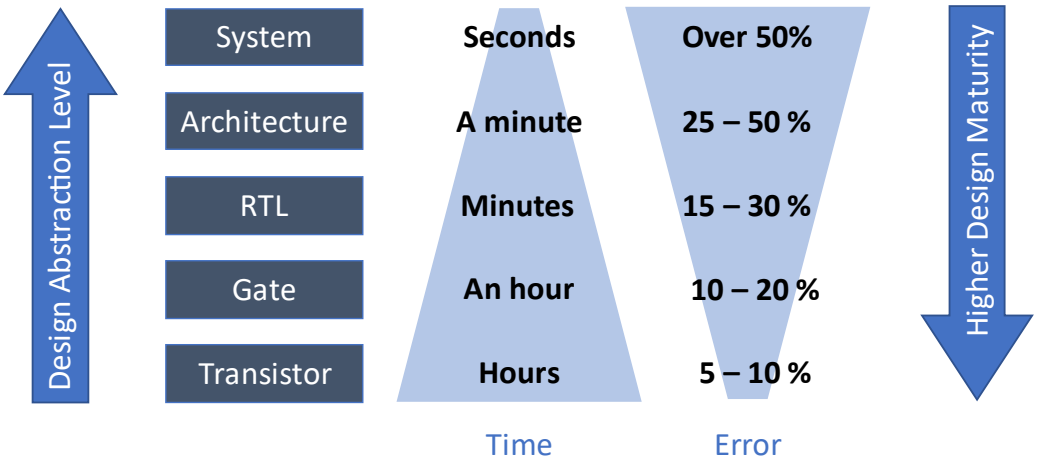


Figure 2. *Power estimation speed and error in different abstraction levels of a typical SoC design flow.*

This thesis aims to study different hybrid power estimation methods and present one high-level power model applied to a real subsystem of a telecommunication SoC. This thesis uses state-of-the-art Electronic Design Automation (EDA) tools to get RTL power estimations for IP blocks and uses those results on a hybrid high-level power model to estimate the total subsystem power.

Early power estimation results are valuable for designers to make early architectural changes on IP blocks and for SoC power management teams to take actions in the SoC power optimization flow. Furthermore, the studied blocks are designed to comply with the IP reuse concept, allowing the final hybrid power model to be an accurate reference for future systems.

The scope of this thesis is to provide a simplified yet precise hybrid power model for a specific subsystem of a baseband processor. Designers, integrators, and system architects can use the power model to quickly estimate the subsystem power dissipation for specific TDD cases and different capacities. The power model should be as simple as possible since the SoC design flow must integrate fast processes.

1.3 Results and Observations

The first outcome of this project was the implementation of a systematic RTL power estimation flow at an early design stage of the subsystem. As a result, the IP designers and verifiers got early power estimations and optimization recommendations at the expense of spending few hours generating the activity files and identifying the proper analysis time windows per each mode of operation. In addition, several rounds of power simulations were performed, allowing designers to follow the evolution of power consumption in each IP to evaluate the impact of implementing different functionalities and configurations. As expected, most IPs increased the power over the design timeline; however, the most power-hungry blocks were the main contributors to the overall subsystem power optimization.

Another contribution of this project was the possibility of deriving subsystem power estimates for different TDD cases and capacities, even without fully functional IP blocks or an integrated subsystem. That was achieved by designing a power model based on the three modes of operation of IP blocks, enabling system architects and integrators to quickly obtain power figures for the most relevant scenarios without requiring new simulations.

Power simulations were performed at different stages during the design process, and this enabled the calculation of calibration coefficients per Hard Macro (HM) through simple regression analysis. The coefficients serve to calibrate the model according to the gate-level simulations of the latest design version. However, the variability of design paces and architectural changes, such as the clock frequency increment in one IP block or the suppression of six instances in the bypass HMs, caused high variability on calibration coefficients and different trends among the evaluated IPs.

Finally, the accuracy of the power estimations cannot be truly determined until engineering samples (ES) are available for real measurements in the SoC. Nonetheless, the RTL power simulations showed an acceptable level of precision compared to gate-level simulations. Thus, they fulfilled the primary goal of building a power database for future projects where designed IPs are potentially reusable. Furthermore, a simplistic power model and low-complexity method were achieved for fast implementation into the SoC design flow.

1.4 Organization

The thesis structure consists of five chapters: the introduction of the theory in Chapter 1, the literature review in Chapter 2, the RTL simulations and power modelling in Chapter

3, analysis and comparison are presented in Chapter 4. Finally, the conclusions of the thesis are unveiled in Chapter 5.

The second chapter collects information regarding mobile networks, SoC design, and power estimation and modelling methods. Chapter 2 is divided into six sections that explain the basic concepts of radio access networks, baseband radio architectures, SoC design, and the primary power estimation and modelling techniques. The last section is a brief review of related works.

The third chapter explains the power modelling approach and the methodology used to estimate the subsystem's power based on individual IP block power simulations. The power simulations performed throughout the design process are also detailed in this chapter.

In the fourth chapter, the results of power simulations and power models are mainly presented in charts and analysed by numbers. The different rounds of RTL power simulations are compared to each other and with the gate-level power simulations. The power model results are presented in three different subsections: maximum power consumption, different capacities, and outcomes after model calibration. Calibration coefficients analysis and error sources are also discussed at the end of the chapter.

Finally, Chapter 5 includes the conclusion of the thesis, precision achieved by the model, and future improvements for the power estimation flow in the design process.

2. LITERATURE REVIEW

This chapter introduces mobile networks, 5G radio access networks, and radio architectures as a baseline for the following topics. The SoC design process is also presented to put in context the previous related work on power estimation for SoCs. Finally, power estimation and power modelling methods and their place on design flow are detailed.

2.1 Mobile Networks

Mobile networks are one of the most significant and complex systems that humanity has built up. Although smartphones are indeed a remarkable technology advance in the last decades, those devices are just the end of a vast and complex network abundant in installations, protocols, standards, and countless patents that have evolved through decades of research and development in the telecommunications industry. A summary of the evolution of mobile networks, their main features, and standards are depicted in Figure 3.

The mobile network systems started their journey around 1980 with the first generation (1G) of wireless cellular technologies. Using analogue techniques and Frequency-Division Multiple Access (FDMA), systems like the Advanced Mobile Phone System (AMPS) and Nordic Mobile Telephony (NMT) were able to establish voice communication over relatively short distances. Nonetheless, during the second generation (2G), mobile networks started a massive standardization and implementations around the world. Roughly speaking, the main contributions of 2G were the digitalization of mobile communications, efficient use of radio frequency spectrum, and the introduction of considered the first data service: Short Message Service (SMS). During this generation, the first standardization effort took place with the Global System for Mobile Communications (GSM), which uses Time-Division Multiple Access (TDMA). Subsequent technology updates brought more efficient modulation schemes for slow internet access with General Packet Radio Service (GPRS) and Enhanced Data rates for GSM Evolution (EDGE).

In the third generation (3G), the main goal was to improve data transfer speed, and the primary channel access technique was Code-Division Multiple Access (CDMA) with variants like Wideband CDMA (WCDMA) or Time-Division Synchronous CDMA (TD-SCDMA). The standardization took another big step in creating the 3rd Generation Partnership Project (3GPP). Their main functions are to develop and maintain protocols for mobile telecommunications from 2G to 5G and beyond. The dominant standard in the

third generation was Universal Mobile Telecommunications System (UMTS), although the following enhancements came with High-Speed Packet Access (HSPA) standards.

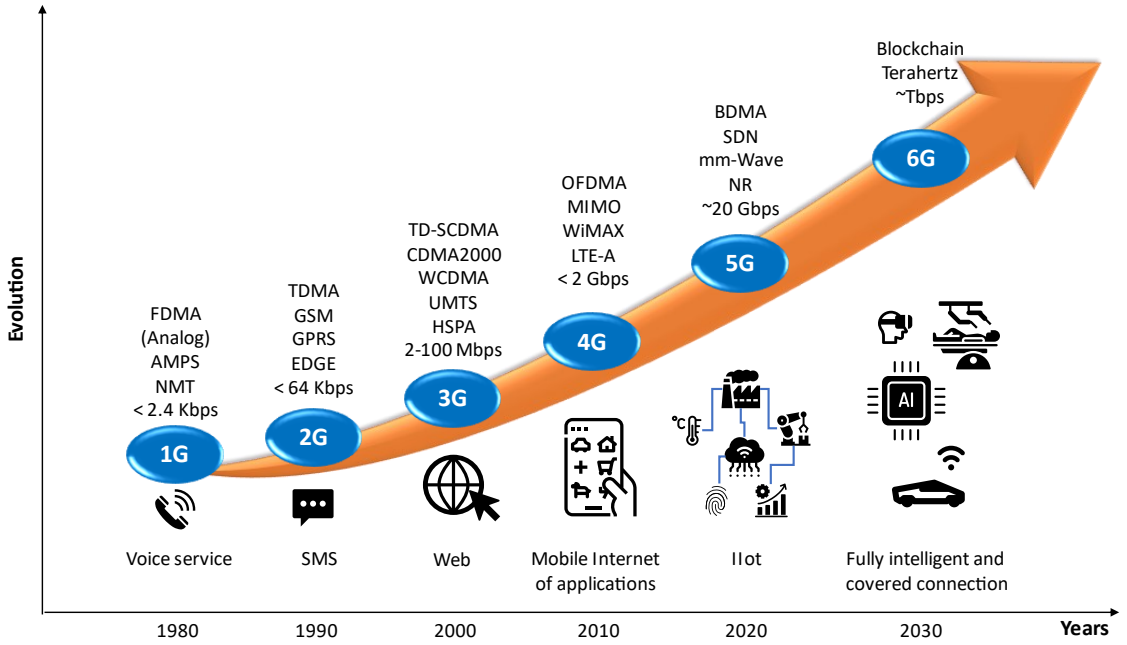


Figure 3. *Mobile networks evolution from 1G to 5G* [3].

The fourth-generation (4G), also known as Long-Term Evolution (LTE), centred its efforts to improve the capacity and speed of wireless data networks. But also on a redesign and simplification of network architecture through an Internet Protocol-based system, improving the transfer latency compared to the previous architectures. In addition, the introduction of more advanced techniques like Orthogonal Frequency-division Multiplexing (OFDM) and Multiple-input and Multiple-output (MIMO) multiplied the radio link capacity and improved the spectrum efficiency. Other standards like Worldwide Interoperability for Microwave Access (WiMAX) are also considered part of the fourth generation, although they were not widely deployed as LTE.

The most recent but not last, fifth-generation (5G), focuses its labour on new cases for human communication and an increasing number of connected machines. Those efforts are committed to improving reliability, latency, speed of data transfer, and capacity for massive machine communications. The 3GPP standard for this generation is known as 5G New Radio (NR). Its main novelty is millimetre-wave frequencies, which increases the transmission rate and allows Beam-division Multiple-access (BDMA) techniques. BDMA techniques are nothing else than the controlled generation of antenna beams towards specific receiver positions [4]. The Software Defined Networks (SDN) also started taking an essential role in the fifth generation, allowing decentralise the traditionally static network architectures [5].

Despite the complexity of new generation rollouts, mobile network stakeholders have managed to keep networks working during the relatively fast evolution of mobile networks. Excluding the first generation, all generations are still on service in most of the globe. However, some countries have already started to switch off their 2G and 3G networks to reassign those licensed frequencies to technologies with better spectral efficiency like LTE or 5G.

2.1.1 RAN Evolution

Besides mobile terminals, also known as user equipment (UE), and regardless of the generation, mobile network architectures consist of two parts: Core Network (CN) and Radio Access Network (RAN). CN serves as a connection node towards other networks like the internet, and it is responsible for critical functions such as managing subscriber profile information and authentication of services. RAN includes the elements used to provide radio communication and access between UE and CN. It is typically composed of base station equipment and antennas to provide mobile coverage in a specific area.

The RAN is responsible for all radio-related functionalities, including scheduling, radio resource management, and more specific tasks like modulation, coding, beamforming, and others [6]. Radio signal and data processing tasks performed in a RAN can be split into two main domains: radio and baseband processing. In radio processing, and from the receiver perspective, the RF signal is conditioned and converted from analog to digital domain. After that, in baseband processing, the digital signal is processed to obtain useful information bits. From the transmitter perspective, and roughly speaking, the reverse process is held with few minor changes.

The architecture of mobile RAN differs from generation to generation [7]. Traditionally, 2G and 3G base stations are decentralized because of the relatively low capacity and latency requirements. For these systems, radio and baseband processing happen directly in each cell site, very close to the antennas.

Early 4G deployments introduced centralizing the Baseband Unit (BBU) of several base stations. This innovative architecture, called Cloud Radio Access Network (C-RAN), made it possible BBUs are not placed on cell sites anymore but are in a central office, facilitating the sharing of processing resources between different base stations [8]. Nonetheless, this approach in 4G or early networks is unfeasible in some cases or too expensive to be implemented. Still, a more critical issue is that network architecture does not meet the high-performance requirements for new 5G use cases, shown in Figure 4.

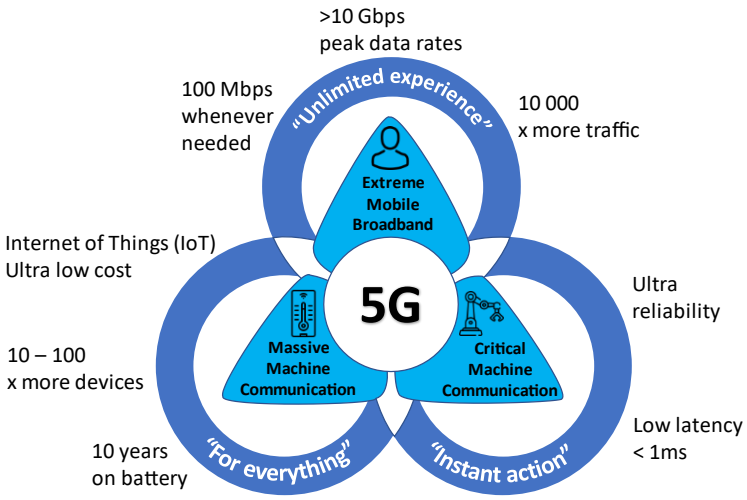


Figure 4. *5G use cases and requirements* [9].

The central services imposed by 5G New Radio (NR) targets are Massive Machine Type Communication (mMTC), Enhanced Mobile Broadband (eMBB), and Ultra-Reliable Low Latency Communications (URLLC) [6]. The mMTC use case is required due to the forecasted expansion of the Internet of Things (IoT). IoT is nothing else than everyday objects and sensors connected to the internet to improve quality of life through seamless communication networks, big data, and analytics in a hyperconnected world that require millions of devices per square kilometre connected to the mobile networks. Another use case of 5G, eMBB, refers to the increase in data transfer capacity. It is required for an enhanced user experience and to provide fixed wireless services with superior throughput than the traditional fixed-line internet connections. Finally, URLLC refers to the reduced transmission time between the base station and the UE and the extremely high reliability of the link to ensure, for example, critical machine-to-machine (M2M) communications.

The evolution towards 5G and beyond requires a flexible network architecture to process data in a centralized or decentralized manner, depending on possible scenarios. That being said, in 5G New Radio-RAN (NR-RAN), the previous functions of LTE BBU are split into three components: Central Unit (CU), Distributed Unit (DU), and Radio Unit (RU) [10]. The CU is the closer to core network entity and is responsible for non-real-time functions of higher layer-2 (L2) and layer-3 (L3). Meanwhile, the DU oversees the real-time processing of layer-1 (L1) and L2 scheduling functions. In general terms, the RU purposes are very similar to the functionalities of Remote Radio Head (RRH) inherited from previous RAN generations, which means dealing with radio-related front-end processing, parts of the physical layer (PHY *or* L1), and digital beamforming functionalities. Figure 5 shows a simplified RAN architecture evolution from 2G to 5G from the transport perspective.

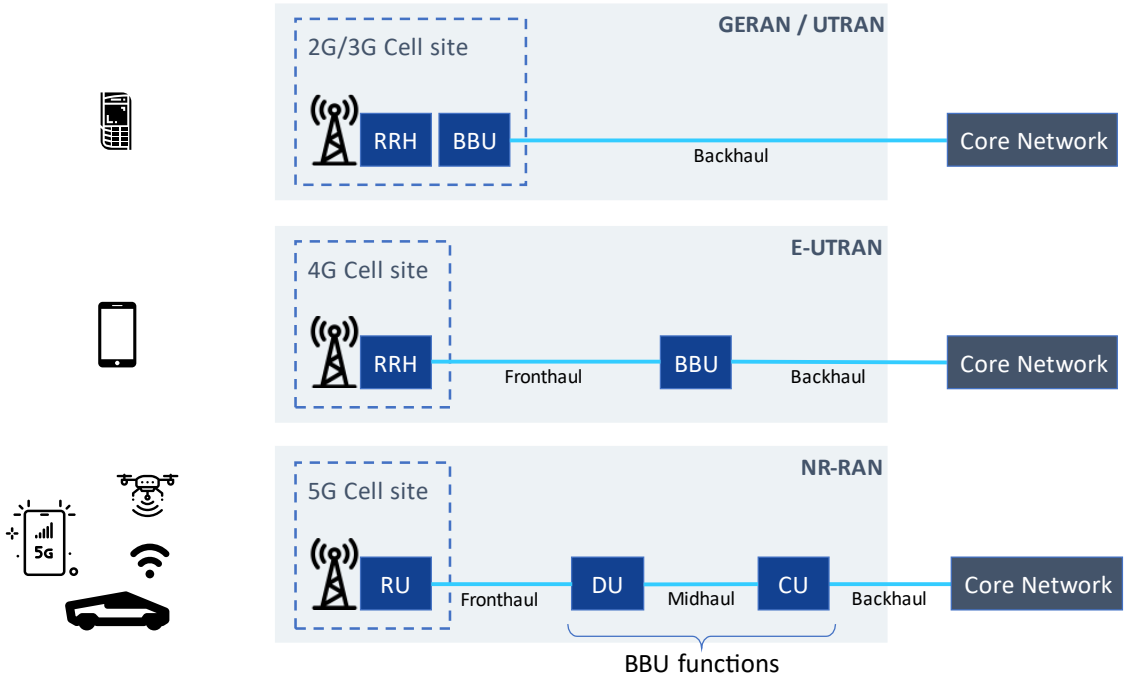


Figure 5. *Evolution of Radio Access Networks and its interfaces from the transport network perspective.*

New generation Node B (gNB) is simply the name for 5G base stations, and it is considered the merger of DU and CU [6]. It is worth mentioning that the advancement of RAN architectures also introduced new connectivity segments and the habitual backhaul used to link the RAN to the core networks. For example, the fronthaul connects the RU to the DU in 5G networks or the RRH to the BBU in LTE architectures. In turn, a new interface known as “midhaul” connects the DU and CU.

2.1.2 5G RAN

To achieve backward compatibility and smooth transition from LTE to 5G, the NR-RAN was designed so that it is possible to connect it not only to the 5G core but also to the legacy LTE core network known as Evolved Packet Core (EPC) [6]. In turn, the architecture of NR-RAN allows the connection of two types of nodes to the 5G core network (5GCN): gNB for NR devices and ng-eNB for LTE devices.

There are two types of implementation for 5G networks: Standalone (SA) and Non-standalone (NSA). Standalone refers to only one RAN technology connected to a core network, either LTE or NR. On the other side, non-standalone options mean both RAN technologies, LTE and NR, are connected to the core network. The combination of different core networks and radio access technologies (RAT) is depicted in Figure 6, where dashed lines represent control-plane interfaces and solid lines correspond to user-plane interfaces. Control-plane refers to functions related to user connection management,

Quality of Service (QoS) policies, user authentication, and others. On the other hand, the user-plane only takes care of user data traffic forwarding. LTE networks first introduced this control-plane/user-plane separation, aiming to make user-plane function independently scalable, allowing operators for more flexible deployment and dimensioning of the network.

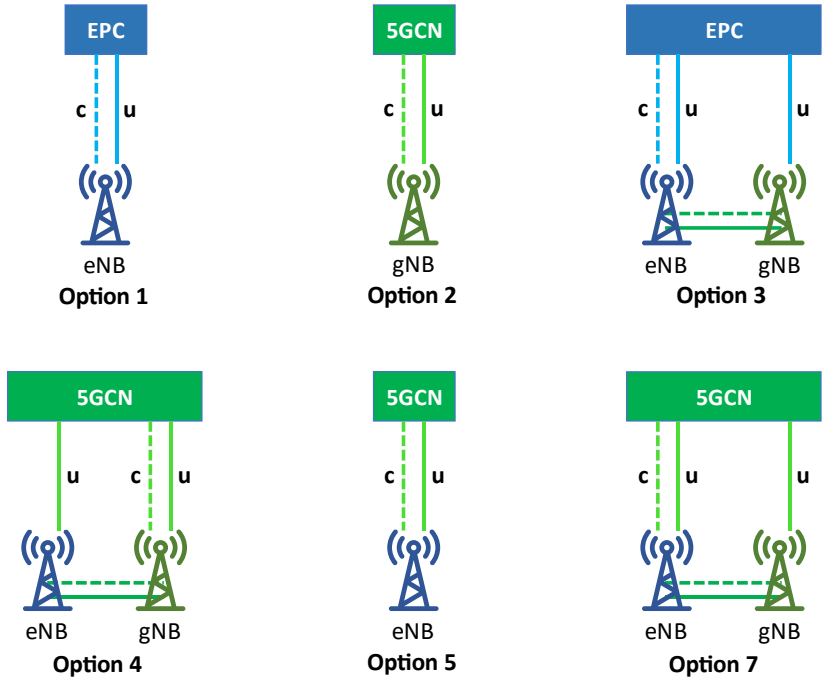


Figure 6. *Different combinations for core networks and radio-access technologies. Redrawn version of* [6].

Due to the backward capability and lower implementation costs, NSA option 3 is a natural path from LTE to early 5G deployments. Operators can leverage existing network investments in transport and core to deliver high-speed connectivity to consumers with 5G devices. However, the full benefits of 5G can be achieved only by implementing the whole 5G core and transport networks. Because of the flexibility of NR-RAN architecture, mobile network operators (MNO) can make a gradual transition according to market needs by using NSA options 4 and 7. Control-plane interface is connected to the core network through eNB in option 4 or gNB in option 7. The final step on this transition is SA option 2, which means both gNB and core are entirely 5G.

5G Core Network architecture

The 5G core network has a service-based architecture, supporting network slicing and control-plane/user-plane split. The service-based architecture means that specification focuses on functionalities provided by the core network rather than nodes as used to be in previous core networks.

One of the main concepts used to handle high-performance requirements in the network is the Quality of Service (QoS). QoS refers to the measurement of the overall performance of a service experienced by the network users. In 5G networks, QoS flow is used to identify and classify traffic priority. QoS markers are added to each packet; thus, the maximum network capacity can be assigned to high priority packets through a feature called network slicing. Network slicing is the ability to divide the network capabilities to serve several customers with different necessities under the same physical core and radio network. For example, one slice of the network can serve mobile broadband requirements from multiple users; meanwhile, the same network could provide ultra-reliable and low latency services to a customer from the automotive industry. The network response could vastly differ for each case, even though they work under the same network infrastructure.

Control-plane and user-plane are entirely independent of each other. That means that it is possible to scale capacity in case of dynamic needs. Control-plane and user-plane also have separate interfaces between gNB/eNB and core network. NG-c represents the control-plane interface, and NG-u corresponds to its user-plane equivalent.

Figure 7 shows a high-level view of the 5G core network architecture and the NR-RAN interfaces. The new core network architecture has a service-based structure, where services and functionalities are the focus. The user-plane function (UPF) is a gateway between RAN and external networks like the Internet. The control-plane function comprises several parts like the Session Management Function (SMF) and the Access and Mobility Management Function (AMF). SMF manages IP address allocation for the UE, control of policy enforcement, and general session-management functions. AMF oversees control signalling between the core network and the UE, security for user data, and authentication [6].

gNB

Functions of gNB include radio resource management, admission control, connection establishment, routing of control and user plane information, QoS flow management and other radio-related tasks. It is important to mention that gNB is rather a logical node, unlike the physical base station concept on previous RANs. In practice, that means gNB not only can manage the traditional three cell base stations but also could have one baseband unit processing data for several remote radio units not necessarily close to each other [6].

As previously mentioned in section 2.1.1, tasks of baseband unit in NR-RAN can be divided into DU and CU. These are sometimes referred to as gNB-DU and gNB-CU because the combination of both is considered the entire gNB. Figure 8 shows the NR-RAN interfaces for both unified gNB and divided gNB-DU/gNB-CU. The midhaul between DU and CU was standardized as F1 interface. The connection between different gNBs is kept by Xn-c and Xn-u interfaces. The gNBs are connected to 5GCN through NG-u and NG-c interfaces. As in previous RAN generations, the air interface between gNB or gNB-DU and the UE is the Uu interface.

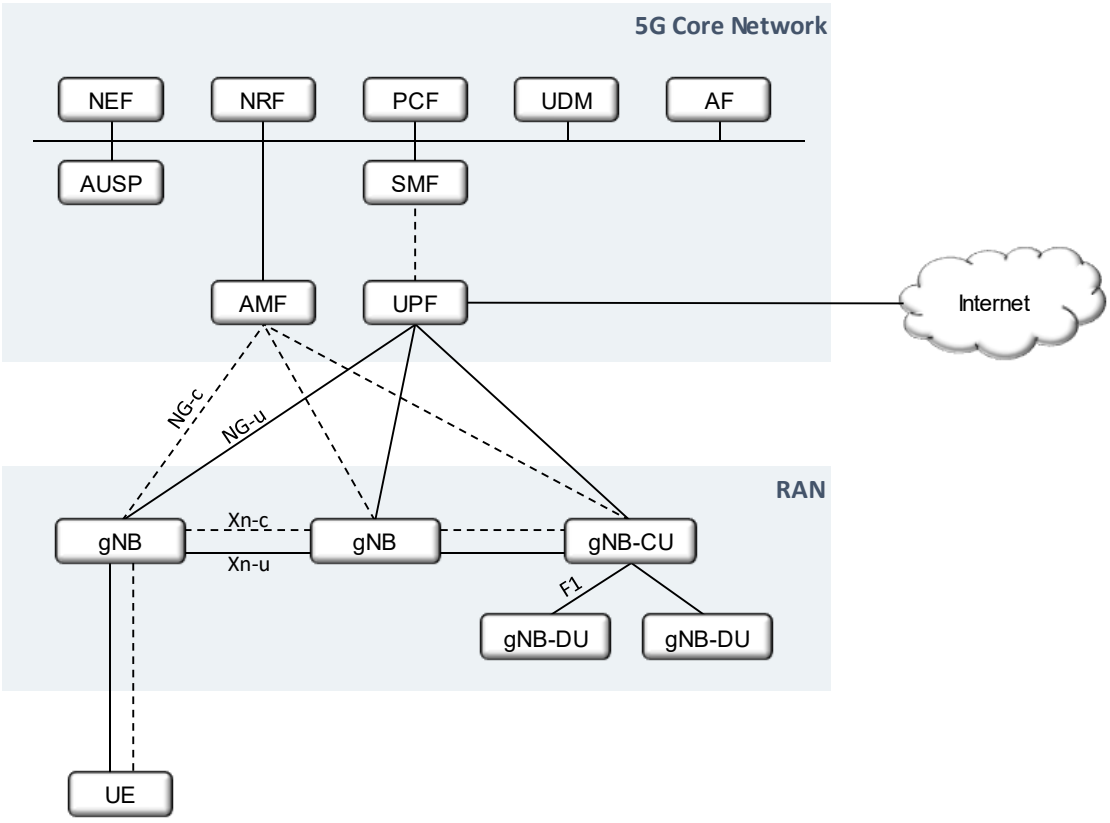


Figure 7. *High-level 5G core network architecture and NR-RAN interfaces. Based on* [6].

The main reason for splitting gNB into DU and CU is to divide the handling of different radio protocols according to different scenarios. For example, some time-critical services, like URLLC, require the processing of lower layers in DU, very close to the RU. That reduces the latency and bandwidth requirements of the traffic carried between DU and CU [11]. On the other hand, other scenarios, like eMBB, does not require baseband processing close to RU, allowing operators to centralize and virtualize those functions. Therefore, DU/CU split is not needed.

2.1.3 Radio Protocol Architecture

NR-RAN protocol architecture was built up into the legacy LTE protocol structure. However, some of the functionalities have been redefined or moved to other layers. In addition, there are new functionalities that are only available if a 5GCN is present. Entities from the 5G protocol stack for both user-plane and control-plane are shown in Figure 9 and summarized below [6]:

- **Physical layer (PHY)**, also known as Layer-1 (L1), is responsible for the coding, physical-layer hybrid-ARQ processing, modulation, multi-antenna processing, and mapping of the signal to physical time-frequency resources [6].
- **Medium-Access Control (MAC)** layer multiplexes logical channels, hybrid-ARQ retransmissions and scheduling-related functions. Scheduling functionality is in gNB for both uplink and downlink.
- **Radio-Link Control (RLC)** manages segmentation, retransmission and provides services to PDCP in the form of RLC channels. However, NR RLC does not support in-sequence delivery of data to higher protocol layers.
- **Packet Data Convergence Protocol (PDCP)** performs IP header compression, ciphering and integrity protection. It also handles retransmissions, in-sequence delivery, and duplicate removal in the case of handover.
- **Service Data Adaptation Protocol (SDAP)** is a new protocol designed to deliver Quality of Service (QoS) information required in 5G systems. The multiple QoS handling is accepted by the SDAP protocol layer only if the connection uses a 5G core.

The control-plane protocols are responsible for connection setup, mobility, and security. Most protocols are the same for both control-plane and user-plane. However, SDAP is present only in the user-plane. Meanwhile, the control-plane functionality Radio Resource Control (RRC) does not appear on the user-plane side.

In a split gNB, the protocol stack and its functions are divided as follows. RRC, PDCP and SDAP protocols reside in gNB-CU, while the remaining RLC, MAC, PHY protocols are in gNB-DU. However, 5G architecture allows selecting where to perform each protocol handling by a new functional split feature.

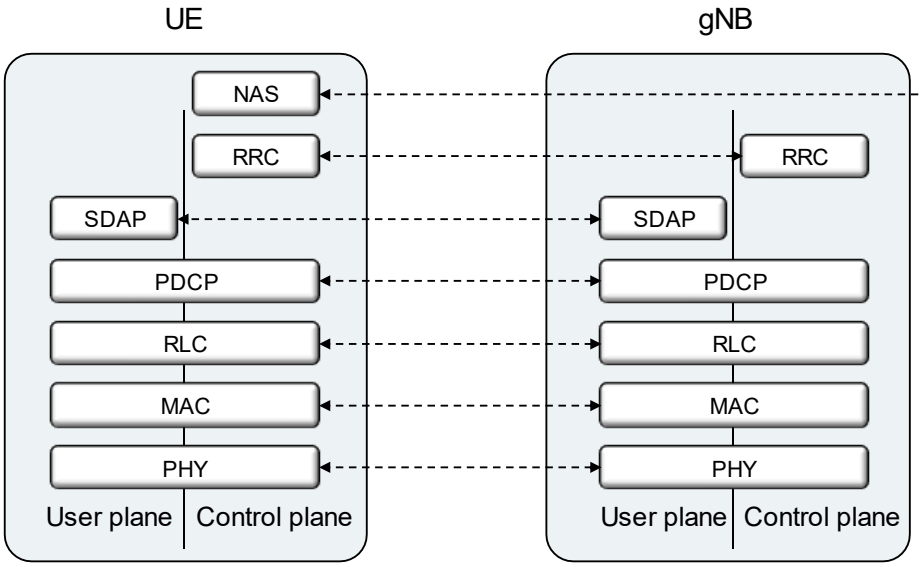


Figure 8. *5G NR protocol stack for user plane* [6].

Functional split

5G allows further decomposing of the gNB protocol stack layers, which means functional splits between DU and CU [12]. This flexible way of splitting the gNB functionalities is intended to leverage the benefits of virtualization and centralization [13]. Figure 9 shows different functional split options. As an example, option 7 is highlighted, and the background shows the CU and DU scopes.

This mouldable architecture allows operators and equipment vendors to develop solutions for different split options depending on radio network deployment scenarios, constraints and intended supported devices. Among the multiple benefits of functional split architecture, we can find flexible hardware implementations, configurable adaptations to several use cases such as variable latency on transport, and last but not least, it enables Network Function Virtualization (NFV) and Software-Defined Networking (SDN) [12].

O-RAN

Besides the 3GPP standardization body, another standardization alliance called Open-RAN (O-RAN) emerged as a carrier-led effort to make RAN open, virtualised and fully interoperable. This alliance also started studies on some of the functional split options. One of the main targets of O-RAN is, for example, to make open and compatible interfaces between RU, DU, and CU, allowing operators to use different equipment vendors in each segment of RAN architecture.

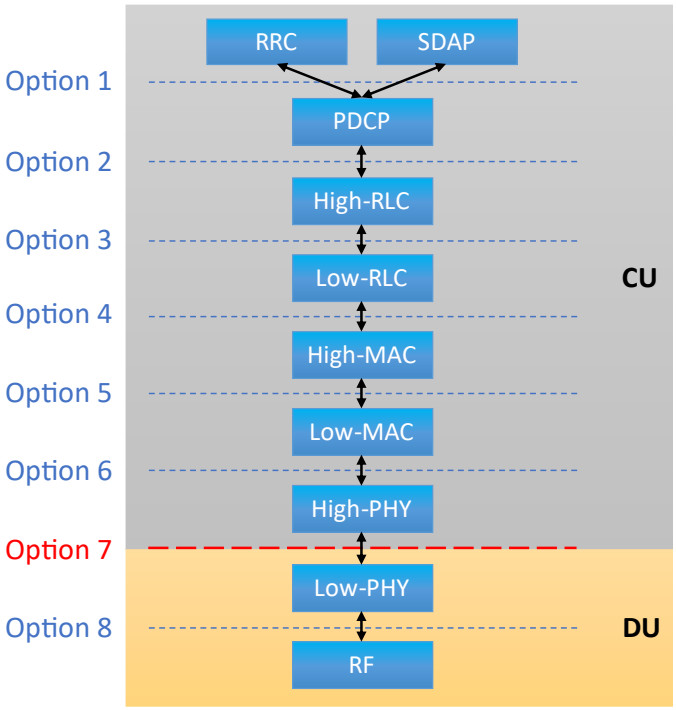


Figure 9. *5G functional split options whit emphasis on Option 7. Based on* [12].

2.2 Radio Architectures

Recent demand for multi-standard technologies, flexibility, and higher data rates, increased the necessity of simplified radio architectures [14]. The main targets for radio components design in wireless communication systems depend on the perspective. From the UE point of view, the aim is small-size, low-cost, low power consumption, multiband and multimode capabilities. On the other side, from the base station perspective, the concerns are system performance, several parallel TX/RX capabilities and similarly, although less critical, size and cost [15].

Radio transceivers are considered as an interface between digital data and electromagnetic waves [15]. The main functionalities are digital baseband/IF waveform generation, digital-to-analog conversion, frequency-translation to desired RF carrier, and power amplification on the transmitter side. Meanwhile, receiver radio modules take care of bandlimitation of incoming signals, amplification, frequency translation to IF/baseband, analog-to-digital conversion, and waveform processing to retrieve data bits [16].

Different radio architectures mean how the front-end functionalities are organized into the radio chain [15]. Several architectures have appeared throughout the receiver history; however, the most practical and thus widely used are the heterodyne and the direct-conversion architectures. Heterodyne architecture benefits are better filtering and channel selection; however, it has a complicated structure that is not easy to integrate. On

the other hand, direct-conversion architecture has a more straightforward structure with fewer filtering stages, lower power consumption, less silicon area [17], and it is a widely used solution for mobile terminals and base stations [15].

Radio architectures can be split into two main stages: front-end and baseband processing. The main blocks of a typical modern direct-conversion radio transceiver are depicted in Figure 10. Here, considering the receiver side, RF signals are processed in the Analog Front End (AFE) block and then converted to the digital domain. The Digital Front End (DFE) block is mainly intended to perform tasks like filtering unwanted frequencies, clipping voltage spikes on transmission signals, cancelling crosstalk from the receiver, or performing Digital Pre-distortion (DPD) tasks. DPD refers to the set of techniques intended to increase linearity or compensate for non-linearity in power amplifiers. Meanwhile, baseband processing blocks are designed for Layer-1 (L1) and Layer-2 (L2) signal processing tasks like beamforming, layer mapping, modulation, channel coding, equalization, and others.

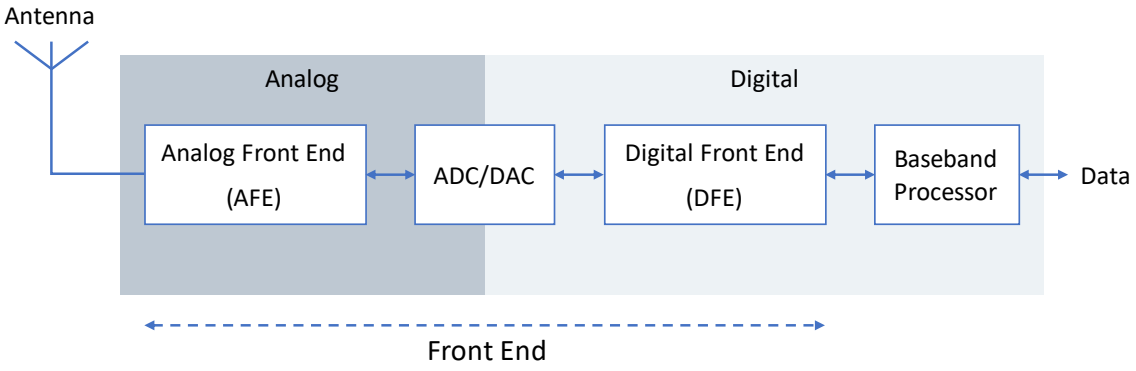


Figure 10. *Digital transceiver architecture* [14].

Challenges on the receiver side are more significant than in the transmitters because detecting weak desired signals in the presence of much stronger signals is more complicated. Therefore, it is not a surprise that most of the literature mainly focuses on receiver architectures rather than on the transmitter side.

2.2.1 Front-End

The front-end is everything between antennas and the digital baseband system [18]. That includes amplification, frequency translation and filtering stages, sampling, analog-to-digital (ADC) and digital-to-analog (DAC) conversion. In addition, the front-end has both analog and digital signal processing stages [15].

Low-noise amplifiers (LNA) with automatic gain control (AGC) are required to cope with different and weak signal levels in receivers on the first stages of the front-end. Other critical components at this stage are filters. These components need to achieve enough

selectivity to get rid of undesired neighbour frequencies signals. Meanwhile, mixers are used for frequency translation. Oscillators help by generating the local oscillator (LO) signals in a tunable manner.

Amplifiers

Depending on the function, amplifiers can be designed for high output power in transmitters or to deliver low-noise performance on the receiver side [19]. Transmitter amplifiers require high output power to the antenna, and lately, stringent power efficiency and linearity are essential in millimeter frequencies systems [20]. Therefore, most radio systems utilize high power amplifiers (HPA), which must deal with several drawbacks, such as high peak-to-average power ratio (PAPR). The PAPR is a metric to evaluate the variation of the signal envelope. It is defined as the ratio between the maximum peak and the signal's average power [21]. High PAPR mitigation techniques for multicarrier schemes like Orthogonal Frequency Division Multiplexing (OFDM) are a topic of broad and current interest due to their direct impact on power efficiency and linearity of HPAs on the transmitter side.

On the other side, receiver sensitivity is the minimum input signal strength needed to produce a good quality output signal [22]. Sensitivity mainly depends on filtering out unwanted incoming signals and boosting the desired but weak ones until acceptable signal-to-noise ratio (SNR) levels. That is achieved by designing low noise amplifiers (LNA), whose function is to provide sufficient gain and sensitivity of RF signal from the antenna [23]. Since the following stages of the receiver chain work over the retrieved and amplified signals, the LNA capacity largely determines the overall system performance. If the LNA performance is low, remaining design efforts on the circuitry of the front-end is useless [24]. Thus, it is not a surprise that vendors are increasingly investing in research and development of this area for higher 5G frequencies [24].

Filters

One of the fundamental parameters in receivers design is selectivity, which is nothing more than the ability to reject signals outside the desired band [22]. This simple task is impossible to achieve by using tuneable RF filters. Nonetheless, adequate selectivity can be achieved through fixed filters at RF/IF bands or by analog filters at relatively low bandpass center frequencies. Multi-rate digital filters also have an acceptable performance within up to a few hundred MHz range. Some specific receiver architectures require special complex filters to suppress frequency ranges from the negative part of the frequency axis [15].

Mixers and oscillators

Mixers are used to shift signals from one range frequency to another. This characteristic of frequency conversion is widely used in direct conversion and superheterodyne receiver architectures. Both architectures' basic idea is that an incoming RF signal is mixed with a carrier signal to produce a baseband signal before detection [22]. A mixer has two inputs: the RF signal and the local oscillator (LO). The LO is at a fixed offset from the desired signal to be tuned. As shown in Figure 11, the output of the mixer produces two results: the sum and the difference of input frequencies. However, the nonlinearities also emit other harmonics during the mixing process [25]. If the input frequencies are the same, the mixer could be used for direct down-conversion, producing the baseband (BB) signal instead of the intermediate frequency (IF).

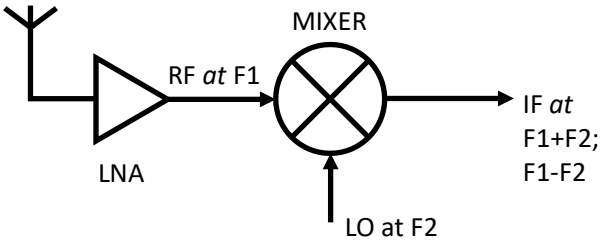


Figure 11. *Mixer structure on a basic superheterodyne receiver architecture* [25].

The main oscillators used in radio transceivers are voltage-controlled (VCO), current-controlled (ICO), and numerically controlled oscillators (NCO). In VCO, the voltage input determines the instantaneous oscillation frequency. Similarly, frequency oscillation in ICO is determined by the current provided at the input. The purpose of using ICOs is to effectively respond to feeble currents, which also leads to a very low power dissipation [26]. Finally, NCO is purely digital generated, containing no analog components or analog inputs. The advantages of NCO are the practical absence of stability problems and frequency and phase noise [27].

2.2.2 Baseband Processing

Baseband processors are hardware accelerators intended to handle physical layer (PHY) processing [28]. From the receiver perspective, the baseband processor receives the raw I/Q data stream from DFE; the data is then filtered and processed according to PHY layer requirements. The useful data is delivered to application processors to handle MAC and higher layers. Transmitter makes the reverse functionality, i.e. receives data from MAC layer and builds up the I/Q data stream for DFE. Figure 12 illustrates a vastly simplified overview of baseband processing tasks.

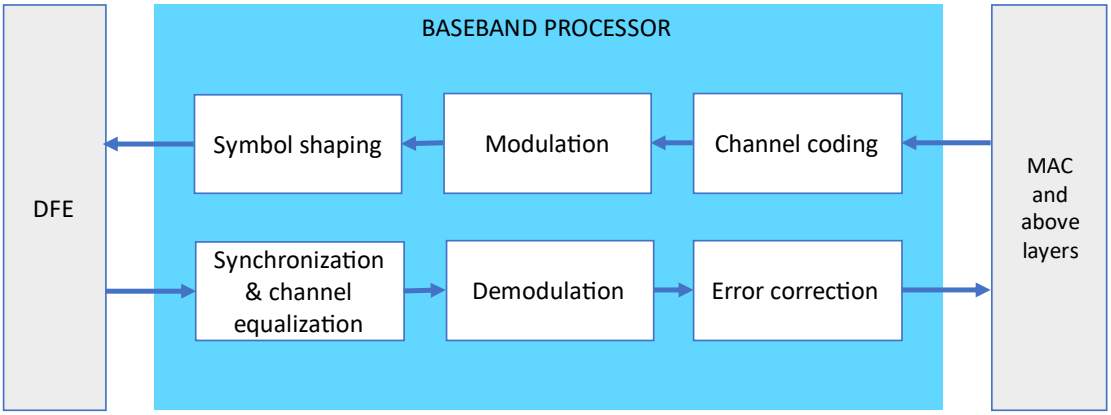


Figure 12. *Baseband processor overview. Redrawn version of* [28].

On the transmitter side, the main functions of the baseband processor consist of channel coding, digital modulation, and symbol shaping. Channel coding includes different error detection and correction methods like Reed-Solomon, low-density parity-check (LDPC), polar and convolutional codes. Digital modulation consists of mapping a bit stream into symbols generally represented by IQ complex samples. There is a second step called domain translation in most cases, especially for multicarrier and multiantenna systems like LTE and 5G. Orthogonal Frequency Division Multiplexing (OFDM) is widely used to divide and map resource elements into time and frequency domains. OFDM generation requires an inverse fast Fourier transform (IFFT), which uses computationally intensive algorithms and therefore is one of the most power-hungry blocks of baseband processors. Finally, symbol shaping consists of filtering the square wave of discrete symbols to convert them into a continuous-time signal so that signals are band-limited. IQ signals' real and imaginary parts are mixed with carrier frequency, added and sent to the DFE [28].

The baseband processing on the receiver side is essentially the opposite of the transmitter. However, the regeneration of transmitted signals is more challenging due to several distortions the signal suffers through the channel. The first step is the detection of incoming signals, which are generally supported by the transmission of known preambles or pilot sequences scattered on time and frequency. The exact timing of incoming signals is carried out by synchronization block through auto and cross-correlations. The next step on receiver flow is channel estimation and equalization, which determines the channel response over time and frequency. To cope with the varying channel response and multipath propagation, known transmitted pilots and adjustable filters are used in the receiver. Demodulation in OFDM systems uses fast Fourier Transform (FFT) and then de-maps the constellation diagram into bits. The final step is channel decoding, which

mainly consists of error detection and correction. Some error correction algorithms can also be computationally intensive and, therefore, high-power drainers.

2.2.3 Baseband SoC Complexity

In the past, the low complexity of single-carrier systems and the uniformly big cells of early RANs allowed the design of relatively simple baseband processors. However, the advent of RAN architectures evolution brought out different cell sizes and multi-carrier, multi-user, and multi-technology systems that indeed increased the complexity of baseband processors. Following this, the necessity of designing flexible SoCs started to become an essential concern for baseband equipment vendors [29].

The baseband processors have moved from a simple fixed-function pipe using several discrete chips to complex multichannel and multimode real-time systems built up over sophisticated SoCs [29]. With the arrival of more complex systems like 5G and flexible architectures like C-RAN, chipset development for baseband has become even more critical. Furthermore, baseband processors for 5G and beyond must deal with multiple challenges like different latency requirements; thus, different kinds of signal processing are needed for:

- Massive FFT processing for OFDM modulation and demodulation.
- Multiantenna channel equalization and estimation.
- 3D beamforming.
- Forward error correction (FEC) with high processing requirements.
- Medium access control acceleration.

The processes mentioned above require different processors like application-specific instruction-set processors (ASIP), high-precision digital signal processors (DSP), custom accelerators, and microcontrollers. Additionally, different processing units must share high-speed memory subsystems, and they shall be kept as local as possible to minimize power consumption [30].

The trade-off between energy efficiency and programmability in baseband processing hardware is essential for modern wireless technologies [29]. General-purpose processors (GPP) are highly programmable and appealing to build up virtualized baseband pools for multi-technology systems. However, they are not power-efficient as DSP and application-specific integrated circuits (ASIC). The drawbacks of DSP and ASIC are the limited programmability and the very high development costs. Despite this, ASIC appears as the only high-performance solution from Layer-1 to Layer-3 in baseband processing

unit development [31]. Figure 13 shows the different semiconductor integrated circuits (IC) functional families for 5G BBU systems and illustrates the market share of ASIC versus generic IC solutions.

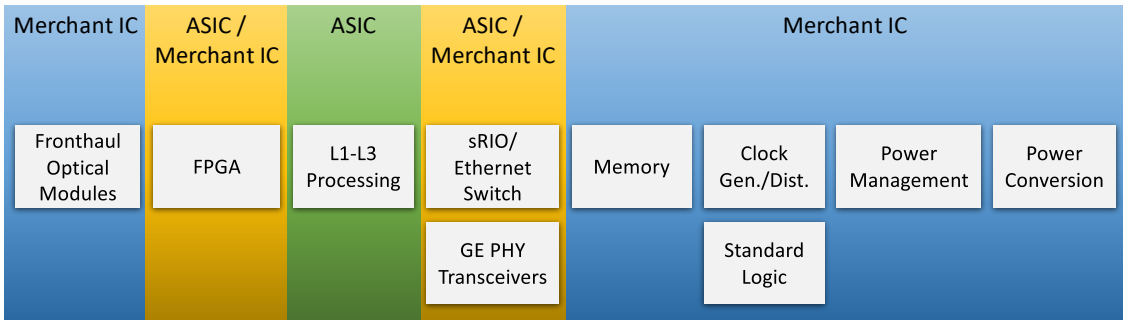


Figure 13. *BBU ASIC versus Merchant IC across functional product families. Redrawn version of* [31]

Many internal blocks that compose a typical baseband unit are designed using standard components provided by traditional IC suppliers. However, generic IC solutions do not offer specific software functionality support at the hardware level. That, in turn, considerably reduces the performance of equipment. The development of ASIC solutions enables equipment vendors to adapt the software and hardware according to the necessity of the final product, allowing them to increase the overall performance. Nevertheless, in addition to the technical complexity, the development of an ASIC also carries very high financial and strategic risks [31].

Baseband processors used to be fabricated using CMOS (complementary metal-oxide-semiconductor) or RF CMOS technology [32]. However, the semiconductor industry has recently moved its interest to FinFET (Fin Field-Effect Transistor) technology due to its impact on SoC performance and power [33]. The main advantages of FinFET are the several orders of magnitude lower device leakage and faster switching speed. However, FinFET technology drawbacks, especially for mmWave designs, are limited gain, self-heating effect (SHE) and parasitic of scaled interconnect [34]. Therefore, even though semiconductor foundry companies manage these physical processes, the related issues and challenges also reflect the overall chipset costs.

Typically, baseband signal processing resources consume the same power per sector in both macro and small cells [29]. However, due to the growing number of small cells in modern RANs, baseband power consumption has become more critical for overall network power efficiency and, therefore, a key factor to reduce operational expenses (OPEX) in mobile networks. Despite the complexity involved in designing SoCs for baseband processing, mobile network stakeholders have noted that the future opportunities

opened by 5G and beyond are promising; and baseband SoCs are vital components to achieving high-quality networks and, therefore, revenue success.

2.3 SoC Design

Silicon technology has evolved in the last decades so that today it is possible to integrate billions of transistors onto a single chip. In that sense, the prediction known as Moore's Law is still valid and stands that the number of transistors in dense IC doubles about every two years, as is shown in Figure 14. However, the way these ICs are designed has also changed and is constantly adapting to the challenges of designing chipsets at acceptable costs, with sufficient quality, and shorter times [35].

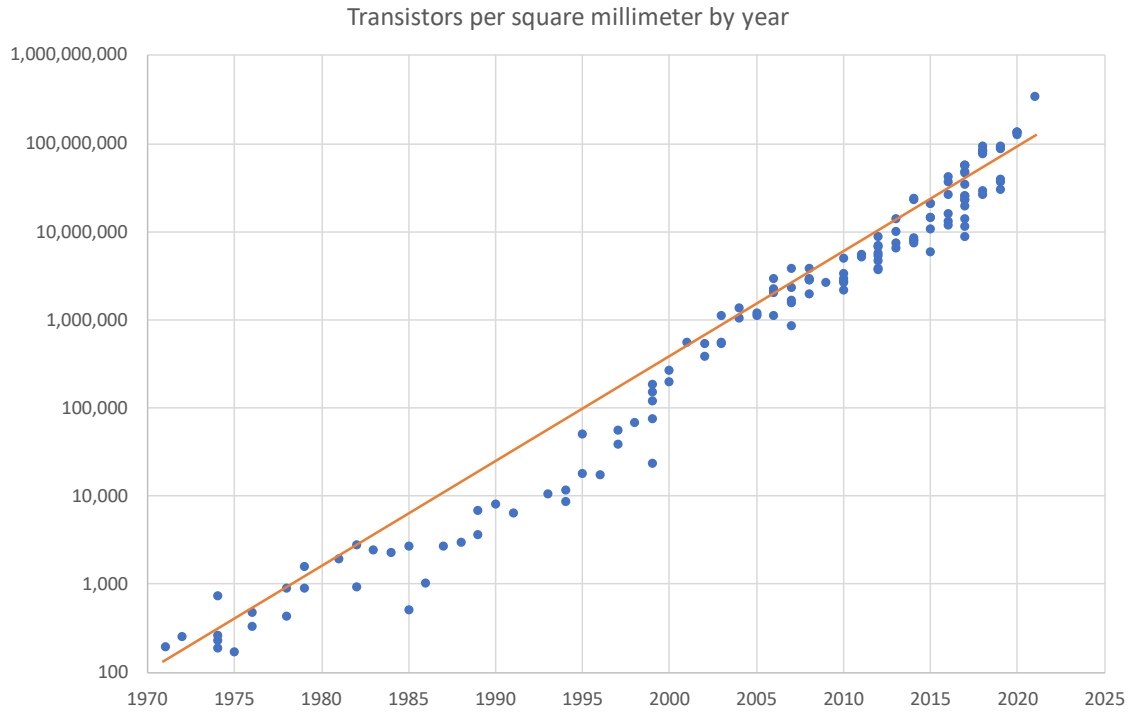


Figure 14. *Transistors per square millimetre by year, 1971-2021.*

In the past, digital systems were built using separate ICs for each function, such as data processing and storage. However, the grade of integration nowadays allows ICs to include whole systems into a single chip. Telecommunications SoCs have also evolved rapidly during the last decades; they can contain billions of transistors and complex functions built into a single chip. An example of a modern SoC is the Qualcomm Snapdragon 865 mobile chip, which integrates 10.3 billion transistors in 83.84 square millimetres. It incorporates multiple processors, memories, and blocks with specialized functions for wireless applications in smartphones and tablets. Figure 15 shows a layout of a 4G baseband card.

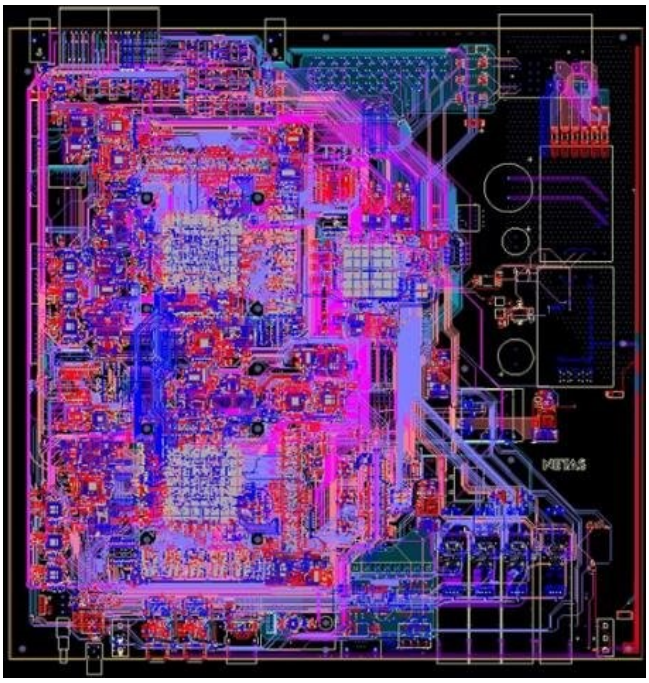


Figure 15. *Physical layout design of a 4G baseband card* [36].

Although less known and present in all places where there is a mobile network, SoCs for base stations require handling massive amounts of data generated by different types of users. Moreover, with the technological evolution from 2G to 5G and beyond, baseband SoCs must deal with increasing throughput requirements and multi-standard support. That places baseband chips among the most complex SoCs in the telecommunications industry [37].

Digital logic of SoCs can be implemented using two main approaches: through field-programmable gate array (FPGA), which allows making modifications on the design after the implementation; or by ASIC, which is a customized IC that cannot be modified after fabrication. There are trade-offs for both options. In the FPGA approach, bugs in logic design can be fixed after implementation, but the increased amount of unused logic degrades its performance and power efficiency. On the other hand, in the ASIC approach, performance and power efficiency are higher at the expense of not having possibilities to fix bugs after implementation; this boosts efforts on verification before fabrication and increases the overall design costs. However, the ASIC unit cost is lower than FPGA and decreases even more with large scale production. Finally, the development time of FPGA can be measured in months, while ASIC development can take several years until the final productization [38].

One of the most powerful ways to achieve successful SoC designs is by reusing previously designed cores. They are smaller subsystems that perform particular tasks or processes. These subsystems, generally known as IP (Intellectual Property), are featured

products in the portfolio of chip design companies. The power of IPs lies in the quick availability and flexibility, as they can be fully integrated with other cores and modules in the design faster and cheaper than designing everything from scratch.

2.3.1 Baseband SoC Structure

SoCs are generally designed to perform very specialized tasks and require different IPs and subsystems depending on the expected performance and capacity. Therefore, the structure of telecommunication SoCs also depends on the capabilities of the equipment and its architecture. Generally, modern SoCs for base station transceivers integrates Layer-1 baseband processing and DFE on a single chip. Figure 16 depicts a very simplified structure of an L1/DFE SoC.

This structure may include different types of processor subsystems (PSS) marked in yellow, different memory systems in light green and on-chip buses for both high and low speed marked in orange. Baseband and DFE subsystems are marked with a grey background. Baseband processors include sub-blocks to perform sub-band filtering, IFFT/FFT transformation, beamforming, modulation, and other tasks that are not listed on the diagram to simplify the overview. The main tasks performed in the DFE subsystem are digital pre-distortion (DPD), digital down-conversion, channel filtering, and usually also includes the analog-to-digital and digital-to-analog conversion, which are interfaces to the RF modules in the RRH.

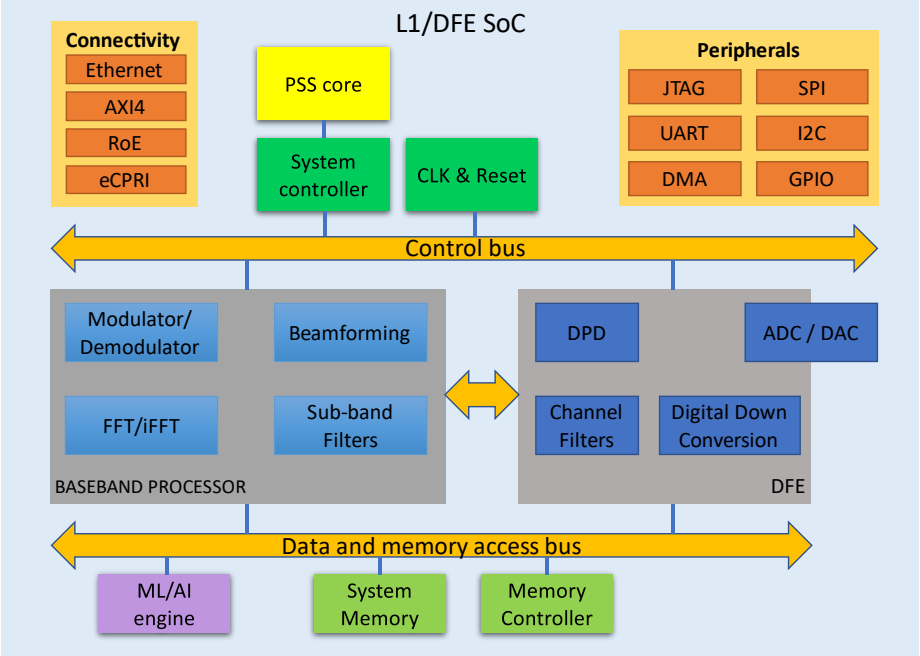


Figure 16. *Generic baseband L1/DFE SoC structure.*

Other general elements include system and memory controllers, clock and reset systems, connectivity interfaces, and peripheral interfaces like universal asynchronous receiver-transmitter (UART) or similar [35]. Recently, other blocks like artificial intelligence (AI) and machine learning (ML) engines are also included in baseband processing SoCs. These engines have potential applications in physical L1 and MAC layers, especially in recent challenging areas such as channel coding, synchronization, positioning, channel estimation, and beamforming [39].

The above representation is not an actual design, but it covers most of the critical structures and challenges in a modern baseband SoC. Real baseband SoC designs are largely more complex, typically including several sets of IP interfaces and data transformations, complex memory structures, multiple processor combinations, DSPs, and other structures that depend on final product requirements. In baseband systems, it is also common to have several data paths or pipes to support parallel processing capacities. In addition, the IP blocks are replicated several times and grouped into hard macros (HM), which are roughly a hierarchy between IP blocks and subsystems. This mixture of complexity makes it almost impossible to build everything with an entirely in-house approach; thus, there are blocks that third-party design companies provide.

2.3.2 Design Challenges

The processors, memories, peripherals, connectivity interfaces, and other components that a baseband SoC handles are non-static technologies; instead, they constantly bring innovations and increasing complexity. Therefore, it is reasonable that system design challenges also arise [35]. Excluding silicon process issues managed by semiconductor foundries, the main challenges of modern baseband system design are the flexibility to support multiple technologies, high performance at low-power consumption, and agility to keep up with the fast pace of standardization evolution. In addition, although not technical, but also a key challenge on the SoC design process is the limited time-to-market (TTM), which must be carefully followed and managed to keep competitiveness in this complex industry [35].

Designing chips with everything from scratch is not a good choice anymore since the concept of design reuse came into practice to save resources, costs and to fit better into tight schedules. That means that a design can – and must – use pre-designed and pre-verified cores [35], integrating them into single but robust and scalable systems. The challenge for designers is how to effectively adopt the reuse concept [35], modify blocks if necessary, and integrate them according to specific case requirements. That is also

applicable to baseband SoCs, where common signal processing blocks can be reused in various projects to make them cost-effective.

Design methodologies depend on chipset-specific targets and challenges. However, there are typical problems faced by everyone who is involved in the design of complex SoCs [35]:

- TTM pressures for rapid development.
- Quality in performance, power, and area (PPA).
- Increasing complexity for verification.
- Submicron issues worsen timing closure.
- Design flow, tools, and guidelines constantly change.
- SoC designs include embedded processor cores, and therefore, software components lead to different methodology, processes, and organizational challenges.

A block-based design approach helps to address mentioned problems. However, if the used blocks are not designed for reuse, there is no gain or benefit due to the effort required to integrate them. In addition, the blocks designed for reuse must provide the correct views, documentation, and functionality.

Challenges on modern baseband SoCs

In addition to the challenges mentioned above, modern baseband SoCs have other obstacles to overcome alongside the whole design process:

- Flexibility to support different technologies and a broad set of use cases. That is especially applicable for base stations since they are expected to handle different technologies from 2G to the latest and even future generations.
- Improved algorithm performance while minimizing power consumption and overall costs of production and operational costs for mobile providers.
- Rapid and flexible development of baseband SoCs, as standards are constantly evolving fast, the design flow must assure constant update of the overall architecture and design.

Finally, the integration of several components on a single chip makes it highly prone to errors at different design stages. Furthermore, the required flexibility of modern wireless systems increases the number of different cases to be evaluated, thus increasing the complexity of verification methods. These verification methods are intended to detect design integration mistakes and the expected interaction between other sub-systems.

Due to the high computation effort of conventional simulation methods, faster simulation methods with reduced computation effort are needed to verify large analog/mixed-signal systems [40].

2.3.3 Abstraction Levels

One strategy to cope with system design complexity is by modelling the digital logic at several abstraction levels. Figure 17 shows the most common abstraction levels: algorithm level, register-transfer level (RTL), gate level, and transistor level.

At the algorithm level, the system behaviour is described as a set of functions that correlate the expected system inputs and outputs without considering timing aspects. The most common design abstraction level is RTL. That is a time-aware model of digital circuits whose basic components are synchronous hardware registers. RTL also describes data flow between registers and other basic digital components such as adders and memories [41]. RTL models are implemented at the gate level through basic AND, NOT, and OR logic gates. Here, timing issues are more critical, and simulations give more realistic results than those at the RTL or algorithm level. Finally, transistor-level abstraction refers to the final layout before the physical implementation of the logic gates into transistors. At this level, the design is not digital anymore, and silicon manufacturers must deal with the analog characteristics of transistors [38].

The advantage of using several abstraction levels is that systems are parallelly modelled at very early design stages, allowing different teams to solve problems on various fronts of the SoC design process. For example, system architects can work at the highest abstraction level, giving an initially loose system specification. This specification can be explored and modified alongside the whole design process using programming languages such as SystemC, C++, System Verilog, and others. At the same time, IP designers can focus their efforts on the RTL implementation of different blocks to meet the requirements of the initial specification. The RTL design can be modified to fix bugs reported by the verification team or new requirements from the algorithm level team. After the first RTL release, a gate-level representation of the RTL design can be obtained, allowing the design team to find timing issues caused by placing and routing components into the physical layout. Gate-level simulations also give very accurate estimates of area and power [38].

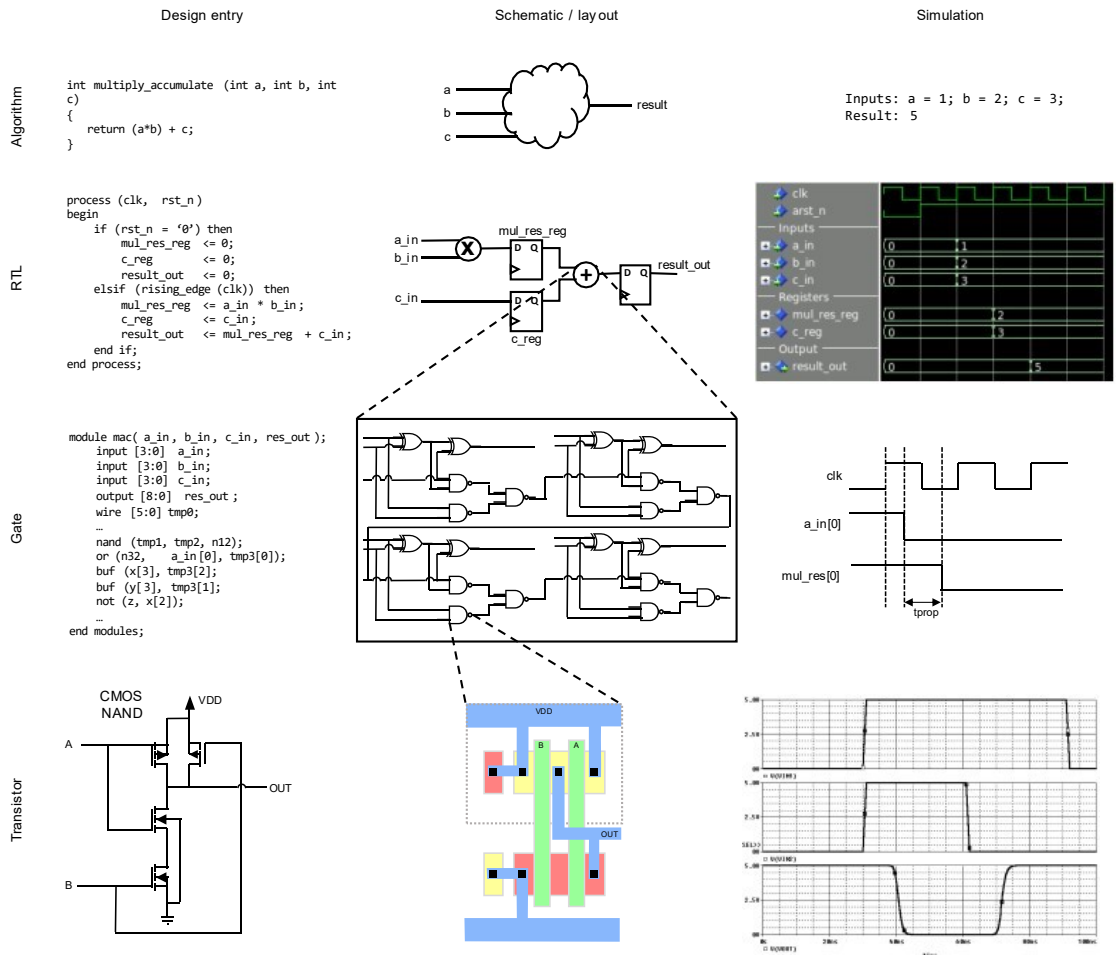


Figure 17. *Most common abstraction levels in digital design* [38].

2.3.4 Design Flow

The chipset design process begins with a product platform requirement, which is a formal technical request to develop a necessary SoC for a new product or service. Then, a roadmap for the coming necessities of chipsets is continuously developed and followed by chipset design companies. This roadmap is aligned with industry technologies evolutions and innovations. At the same time, and due to the high strategic and financial risks of ASIC and SoC design, the roadmap must follow strategic plans and studies regarding competitive intelligence based on customer feedback. Sometimes, initially planned chipsets are cancelled or re-designed due to many factors such as technological changes and financial or management movements. However, ASIC spins are expensive; thus, extensive quality processes must minimise the number of re-spins.

The sequence of steps to accomplish the design of an IC is called design flow [42]. That provides a standardized way to follow the steps and the EDA tools required to carry out those steps successfully. Very roughly speaking, this process can be divided into three

main phases: specification, design, and verification. Each of these phases overlaps alongside the whole project duration, as depicted in Figure 18 [38].

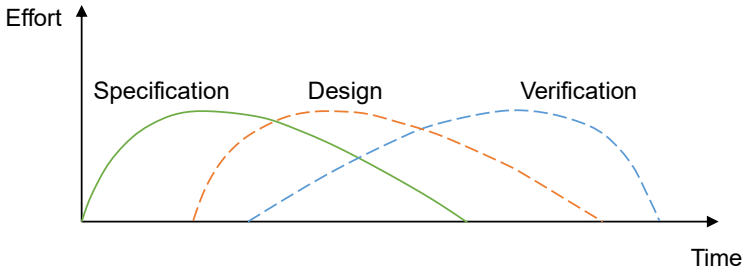


Figure 18. *Design phases and estimated effort in the SoC development timeline* [38].

The specification phase is done at the algorithm abstraction level and begins with an abstract system representation. That representation is refined to a behavioural model and eventually transformed into a manufacturable form. The functionality partition between software and hardware is also defined in this phase. In ASIC design, the target technology and the physical implementation feasibility are also expressed at an early specification phase.

The design phase consists of the implementation of the system. This phase is divided into front-end and back-end. The specification is transferred to RTL architecture using Hardware Description Languages (HDL) such as VHDL and Verilog in the front-end. Meanwhile, the back-end refers to the physical implementation steps, including mapping from RTL to gate-level (also known as synthesis), placing and routing the cells in the physical layout, and checking that the physical design meets timing, area, and power requirements.

The functional verification mainly checks that digital design behaves as is described in the specification and takes place as soon as the early RTL design versions are available. The correctness in system functionality is largely tested at several abstraction levels since a single bug that is not correctly detected in time might lead to expensive system redesigns.

SoC design lifecycle

From the management perspective, there are several processes involved in the SoC design lifecycle. This lifecycle is a continuous iteration covering exploration, planning, development, and production stages [43]. Figure 19 shows the main processes included in the modern SoC design lifecycle. These processes are mainly carried out in parallel. Design teams work simultaneously from the beginning to the end of the project, giving

continuous feedback to other groups, ensuring the design meets its performance goals [35].

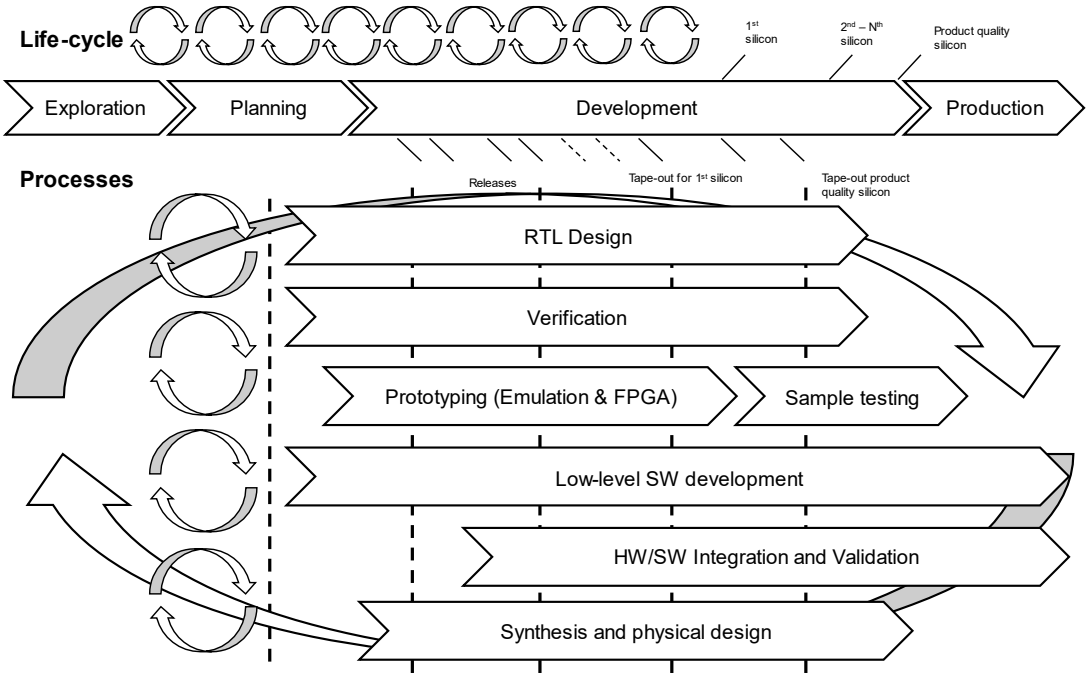


Figure 19. *SoC design lifecycle and processes* [43].

The exploration stage is intended to understand the requirements and target the ASIC technology early to recognize the limits and feasibility of system implementation. Feedback from the SoC stakeholders and previous iteration is collected. The previous iteration can be an old version of the SoC or a similar system previously designed. In case there are no previous iterations, the inputs for exploration are the requirements of the top product where the system will be integrated.

In the planning stage, decisions about system architecture are made based on high-level models. Requirements for the system and hardware architecture are discussed at this stage. Another critical factor decided at this point is whether to develop subsystems in-house or from external IP vendors.

The development stage includes most of the SoC design processes. Each of these processes is performed by specialized teams that are generally spread in several countries. Each work team is not independent of other teams, but rather they need to synchronize HW/SW development at several levels to achieve a smooth design flow.

Finally, the production stage starts when the system is ready for the next level of integration, and the final engineering samples (ES) of SoC pass extensive functional tests. Then, the new SoC or ASIC begins a mass production and is released to the market or integrated on upper-level systems.

2.4 Power Estimation Methods

Typically, processors and other high-performance IPs are dominant contributors to SoC peak power and overall chip power integrity [44]. Peak power is the maximum power the system can dissipate without failures, and power integrity refers to the ability to meet the voltage and current targets from source to destination.

It is crucial to get an early look ahead on power integrity issues and potential power savings during the SoC design process. Most significant power reductions are detected early at the architectural level, allowing the design and algorithm teams to improve the design in a more power-efficient way.

The first step regarding power analysis in an SoC design is the power estimation. Power estimation gives the expected power consumption of the chip based on a provided functionality or testcase. Typically, the peak-power scenario is considered chiefly due to the importance of not overpassing the maximum power capacity the system can handle. Average and idle test cases are also considered to analyse possible power optimization during normal operation and standby modes. Those results are taken as a guideline during the whole design process. Simulations are performed as frequently as possible to be aware of power variations because of new features or modifications on the design and their potential impact on the top-level performance.

The power estimation on an SoC can be done at several abstraction levels. The more accurate results are obtained at lower levels; however, the simulation time increases as the abstraction level decreases. Power estimation at lower abstraction levels cannot be done at the early design stages. A basic netlist design is only delivered at the middle design phases when some functionalities are already implemented. However, analysing power after the synthesis process is too late to identify power-inefficient RTL code. Therefore, making power estimation at early RTL releases is a crucial tactic in the low-power development cycle.

2.4.1 Power Dissipation in CMOS Circuits

The power consumption in CMOS digital circuits takes place in two forms: static and dynamic. Dynamic power consumption mainly occurs during normal system operation due to the high number of toggles in the logic gates. The switching activity is associated with the internal and external capacitance generated by the charge and discharge of gates' transistors. On the other hand, static power refers to the power consumed by several leakage sources, and it becomes relevant on devices during standby mode operation [45].

The simplest way to illustrate the primary sources of power consumption in CMOS circuits is by analysing the inverter circuits depicted in Figure 20. The inverter is composed of an nMOS transistor (bottom) and a pMOS transistor (top). The circuits “a” and “b” occur during switching activity. Circuit “a” illustrates the charging and de-charging of the parasitic capacitances, while circuit “b” sketches the short-circuit current that flows during very short periods on each logic toggle. Finally, circuit “c” depicts the leakage current, which is independent of switching activity and mainly depends on the physical characteristics of the target technology.

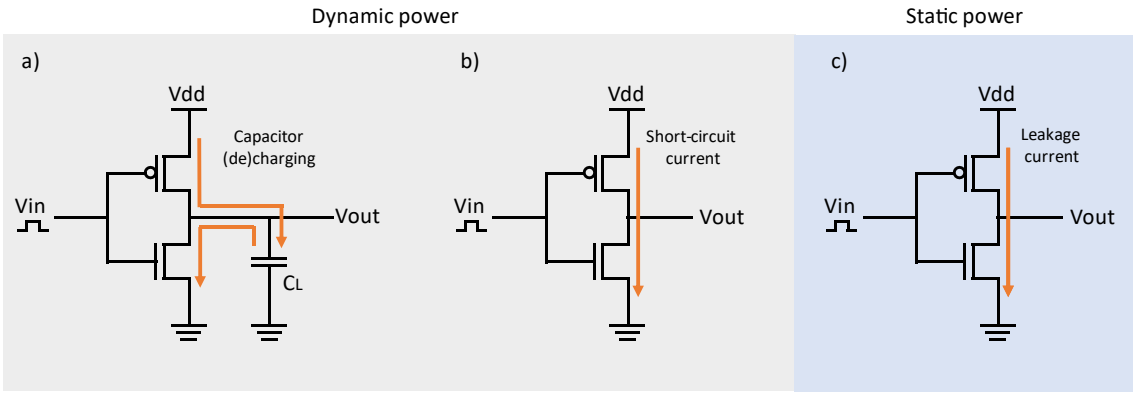


Figure 20. *Power sources in CMOS circuits. Based on* [46].

Equation (1) represents the total power consumption and comprises the sum of the two components above mentioned. Dynamic power is usually split into switching and short-circuit power. Static power is sometimes referred to as leakage power [47].

$$P = P_{\text{dyn}} + P_{\text{static}} = P_{\text{switching}} + P_{\text{short}} + P_{\text{leakage}} \tag{1}$$

Where, P_{dyn} represents the dynamic power, P_{static} the static power, $P_{\text{switching}}$ the switching power, P_{short} the short circuit power, and P_{leakage} the leakage power. The switching power, calculated from equation (2), depends on the parasitic capacitances of the transistors and the connection between them. They are known as load and wire capacitances, respectively [46].

$$P_{\text{switching}} = \frac{1}{2}\alpha C_{\text{L}} {V_{\text{dd}}}^2 f_{\text{clk}} \tag{2}$$

Where, V_{dd} depicts the source voltage; C_{L} the parasitic capacitances; f_{clk} the clock frequency; and α the switching activity factor, which refers to the number of transitions during each clock cycle [46].

The short-circuit power component is also related to the switching activity of the transistors and is subject to internal capacitances and short-circuit currents peaks formed by

the finite transition times during change of state in the transistors. It occurs because there is a short period when both pMOS and cMOS transistors are on, thus creating a short circuit current that can be calculated by estimating the average amount of charge per output transition. The final short-circuit power [46] is given by

$$P_{\text{short}} = I_{\text{short}} V_{\text{dd}} = \alpha Q_{\text{short}} V_{\text{dd}} f_{\text{clk}} \tag{3}$$

where, I_{short} is the short circuit current, and Q_{short} the average amount of charge per output transition.

The static power is not related to circuit activity and exists as soon as the circuit is powered. There are several static power sources, but all of them are typically simplified on leakage power as

$$P_{\text{static}} = P_{\text{leakage}} = I_{\text{static}} V_{\text{dd}} \tag{4}$$

where, P_{static} represents the static power, and I_{static} the sum of all the static currents. The static currents inside CMOS transistors are always present but can vary between different logical states of the circuit. The components of the static currents are expressed as

$$I_{\text{static}} = I_{\text{G}} + I_{\text{RB}} + I_{\text{ST}} + I_{\text{GILD}} + I_{\text{PT}} \tag{5}$$

where, P_{static} represents the static power, I_{static} the static currents, I_{G} the gate-oxide tunnelling leakage current, I_{RB} the reverse-biased diode leakage current, I_{ST} the sub-threshold leakage current, I_{GILD} the gate-induced drain leakage current, and I_{PT} the punch-through leakage current [47]. In traditional planar CMOS circuits, the leakage currents increase as target technology scales down. That increases the temperature of the chip and its static power dissipation. The 3D structure of FinFET technologies helps directly tackle this problem, which is especially visible in technologies under 100 nm. The multi-gate structure of FinFET provides better electrical control of the effective channel formed in the transistor, thereby significantly reducing leakage [47]. Figure 21 shows the structure of the traditional planar CMOS, the FinFET used for technologies down to 5 nm, and the projected Gate-All-Around (GAA) for smaller technologies [48].

To summarize, the power dissipation components in CMOS circuits are static and dynamic power. In power estimations, static power and short-circuit power are looked up from defined libraries based on the target circuit technology. These libraries include timing, power, area, and other characteristics of standard cells. Power related to switching activity is estimated through several methods studied in the following sub-section.

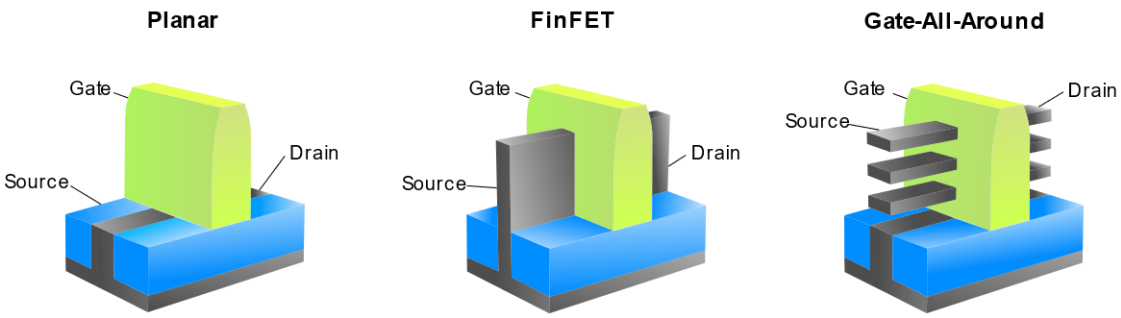


Figure 21. *Planar CMOS, FinFET, and GAA 3D structures* [49].

2.4.2 Power Estimation and Power Analysis

The low-power SoC designs require a continuous power estimation and analysis flow. Power analysis is performed when the physical layout of the design is available; generally, this happens at the late stages of the design flow. On the other hand, power estimation is done with incomplete information and must be started as soon as system design starts. The estimates and analysis are generally executed using EDA tools integrated into the design flow.

There are different types of power estimation and analysis methods, which are designed for different abstraction levels. Power estimation accuracy increases when moving down to lower abstraction levels because there is more information available for calculations. However, at the same time, that high amount of data slows down the simulation time at lower abstraction levels. This thesis focuses on low-level power estimation techniques at RTL and gate-level, which are mainly used by EDA tools in the semiconductor industry.

RTL analysis

There are HDL descriptions of functional blocks at the register-transfer level, but information about the gate, circuit, and layout level may not exist in the early design stages. Therefore, a significant challenge at RTL analysis is getting enough accuracy since there is insufficient information about the design. Thus, the power estimations must rely on power model libraries or power descriptions for each functional block. RTL's main power modelling methods can be categorized as analytical, macro-modelling and fast synthesis methods [47].

Gate-level analysis

Gate-level simulators generally can handle full power estimation on large designs. The main reason for simplifying this type of analysis is that energy consumed by each output transition does not depend on the resistive characteristics of the transistors. That allows

applying equation (2), which only requires models of gate parasitic and wire capacitance. That is only applicable to the dynamic power component, whereas, for total power consumption, the transistor-level simulations are still necessary to estimate leakage or static power [46].

Transistor-level analysis

Power estimates at the transistor level are obtained using transistor-level simulators. These simulators take user-defined inputs and compute voltage and current on each node of the circuit. At this level, complex models for circuit devices can be used, providing very accurate results for digital CMOS circuits and analog modules.

A detailed simulation requires complex models whose solutions are not feasible for large designs in this type of analysis. Similarly, the input sequences must be very short since each toggle propagated in the circuit represents a high load in the computation resources and simulation time. For these reasons, circuit-level power estimations are only performed for characterizing small circuits, where accuracy is crucial. One workaround for power estimation at this level is to simplify the models of the active devices as much as possible. Unfortunately, this simplification is possible at the expense of accuracy loss. On the other hand, although more expensive, parallel computing can help accelerate large simulations without losing accuracy [46].

Switching activity analysis

Independently of the abstraction level of analysis, dynamic power consumption is the main contributor to the total SoC power consumption. Therefore, the study of switching activity is of common interest for each of the abstraction levels. There are three classes of techniques for dynamic power estimation: simulation-based, probabilistic-based, and statistical-based methods. Those methods are also known as dynamic, static, and hybrid methods, respectively [46].

2.4.3 Simulation-based Methods

Simulation-based methods are also known as dynamic since they require switching activity information directly from circuit simulations [47]. These methods are common in lower abstraction levels, where the circuit model can be completely simulated. The dynamic estimation methods are helpful for final power evaluation and validation because of their accuracy; however, they are computationally intensive and compared with static methods, their simulation time is too long.

In simulation-based methods, highly optimized logic simulators are used; however, this leads to two main issues: the delay model used in logic gates and the number of input vectors to simulate [46].

Gate delays

The simplest way to handle the gate delays is to assume zero delays for gates and wires, which means all the transitions in the circuit occur simultaneously [46]. However, this is a non-realistic approach since gates and wires have non-zero transport delays, which might cause several toggles in response to a single input vector. To illustrate this, Figure 22 shows a simple circuit formed by two 2-input NAND gates. Assuming initial inputs of 1 and 0 for x and y, respectively, the output of z is 1. Without delays, the transition on y from 0 to 1 should change the signal w from 1 to 0 but keeping the output z in 1. Nonetheless, the processing delay in gate A keeps the signal w high during a short time Δ_A, this causes an undesired toggle in output z during the same period Δ_A. The undesired toggles are known as 'glitches' and are a source of switching power waste. Gate B also introduces a delay Δ_B, which is shown on the bottom of the timing diagram, but it keeps the 'glitch'. The glitches are orange marked in the timing diagram.

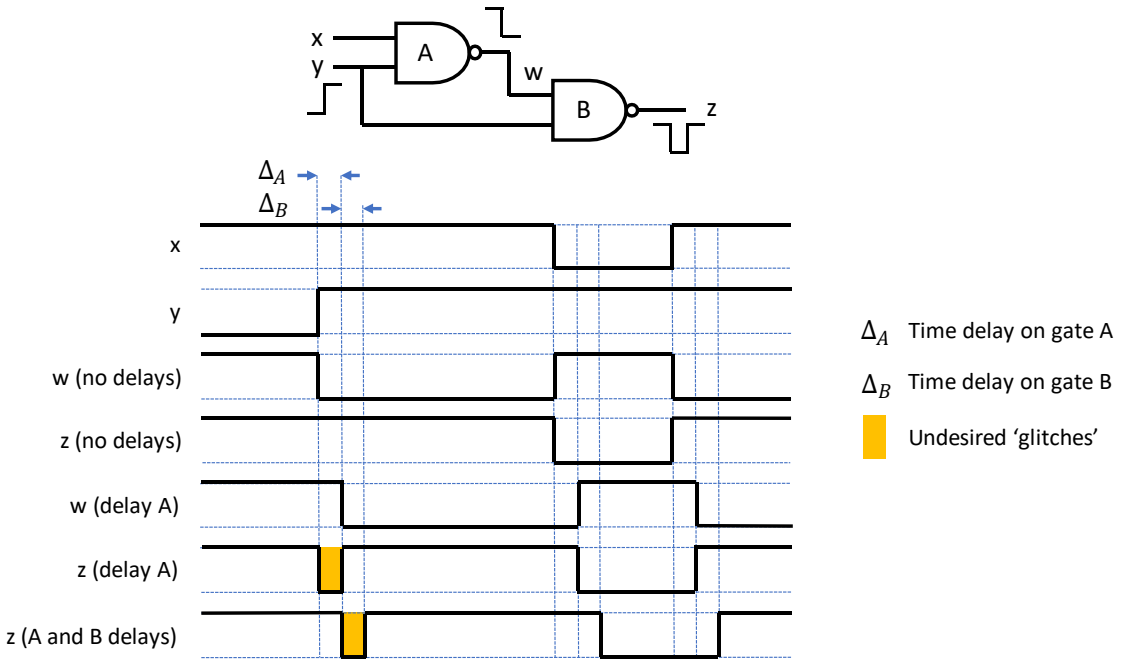


Figure 22. *Basic 2-input NANDs circuit and its time diagram. Based on* [46].

In this simple circuit, the expected number of toggles in the output of gate B would be 2; however, the 'glitch' doubles the transitions to 4. This spurious activity can significantly increment the overall switching activity; thus, it is imperative to integrate appropriate delay models in simulation tools.

Number of input vectors

The second main issue in simulative methods is determining the number of input data vectors to simulate [46]. If the user gives the input vectors, the switching activity can be computed in logic simulators; however, there still exists a trade-off between the simulation time and the accuracy of estimations, which are highly dependent on the number of input vectors [2].

There is another scenario where only one set of input statistics is provided, and the sequence of input vectors must be generated. One option is to generate a sequence based on those input statistics and perform the simulations until the average power converges, which means a value achieves a steady value defined and calibrated by the user. An alternative to the last option is to rely on the basic assumption that the power consumption during a given period has a normal distribution; then, the number of input vectors can be obtained based on the central limit theorem [46].

In general, the number of input vectors required to achieve a reasonable accuracy is relatively small -around thousands-, even for large circuits. However, low-switching activity nodes require such high accuracy, where this approach would lead to an excessively large number of input vectors. Therefore, designers might assume that low-activity nodes do not significantly impact dynamic power consumption, or they still can run simulations using parallel architectures that shorten simulation times [46].

2.4.4 Probabilistic-based Methods

Probabilistic power estimation methods, also known as static methods, are non-simulative and do not require simulation activity; therefore, they rely on data characteristics rather than real data. The idea is to generate initial inputs and propagate their statistics to obtain the switching probability on each circuit node [46].

These estimation methods are suited for higher abstraction levels, where there is not much detailed information about the design. For example, in the SoC design flow, static methods are typically used for the initial power dissipation estimates at the architectural level. As in simulative methods, probabilistic methods also must sort out a set of complex issues.

The static probability of having a HIGH logic value in a signal s can be defined as p_s and the transition probability that signal changes from LOW to HIGH or vice versa can be defined as p_s^{01} or p_s^{10}, respectively. The basic digital circuit previously shown in Figure 22 can also be used to illustrate the basics of these methods. The probability of having a HIGH logic value in both inputs x and y is 0.5: $(p_x = p_y = 0.5)$. Then, the probability of

having 1 in w is $p_w = 1 - p_x p_y = 0.5$. However, the output z is not the intuitive $p_z = 1 - p_w p_y = 0.625$, but has a more complex equation since its inputs, w and y, are not independent: $p_z = 1 - p_y + p_x p_y = 0.75$. This shows that not accounting for the spatial correlation of gates can lead to significant errors.

Another well-known issue in probabilistic methods is the temporal correlation, which occurs when an input bit depends on the previous bit of the same input, affecting the transition densities. In practice, calculating the probability of a signal to be 1 is insufficient, and we also need the transition probabilities given an actual state. To illustrate this problem, we can consider the timing diagram of signals x and y, shown in Figure 23; vertical lines depict the clock periods, i. e. there are eight potential toggles for each signal. In this example, both signals, x, and y have the same amount of 1 and 0 values in the depicted time interval; hence the probability of both signals to be 1 is $p_x = p_y = 0.5$. If we do not consider the temporal correlation, the transition probability can be calculated as: $\alpha_x = \alpha_y = p^{01} + p^{10} = p^0 p^1 + p^1 p^0 = 0.5$. Nevertheless, the probability that x change from 0 to 0 is $p_x^{00} = \frac{3}{8} = 0.375$. Similarly, the other transition probabilities can be calculated as: $p_x^{01} = \frac{1}{8} = 0.125$, $p_x^{10} = \frac{1}{8} = 0.125$, $p_x^{11} = \frac{3}{8} = 0.125$. Therefore, the actual average switching activity for x is $\alpha_x = p_x^{01} + p_x^{10} = 0.25$. On the other hand, signal y has transitions during the whole depicted time interval, maximizing its switching activity as: $\alpha_y = p_y^{01} + p_y^{10} = 1$. This simple example shows that temporal correlation must be adequately modelled; however, the computation of the average switching activity considering temporal correlation is feasible only for small circuits [46].

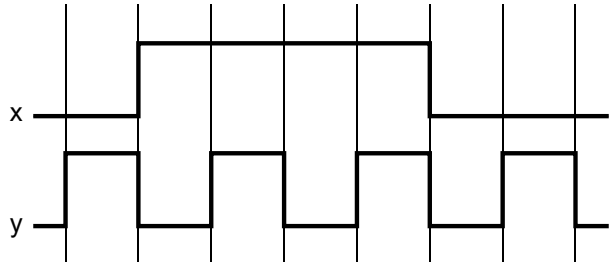


Figure 23. *Example signal to illustrate the concept of temporal correlation* [46].

Finally, the transport delay of signals is also a problem in these methods since the glitches can alter the expected transition probabilities. These errors can be propagated through the circuit, thus incrementing the error estimations. The use of simple delay models can produce fluctuations and uncertainties, which can become a source of sig-

nificant errors in power estimations. One solution for this issue is to describe the probability of signals as a continuous function of time, which offers more accurate results than fixed delay models [2].

2.4.5 Statistical-based Methods

Traditional simulation-based approaches simulate the circuits using defined functional input patterns, which is expensive and time-consuming. In probabilistic-based methods, the trade-off between accuracy and simulation speed makes this approach unacceptable and generally impractical for large circuits. A hybrid alternative that combines the precision of simulation-based techniques and the speed of probabilistic-based methods is the statistical-based approach. It generates random input patterns to the circuit and monitors the resulting power consumption through a simulator. This task is performed until an acceptable accuracy, with a specific confidence level, is achieved [2].

In statistical-based methods, there are several approaches to estimate power consumption. One of the first works regarding these techniques was presented in [50], and given the necessity of generating a finite number of patterns from an infinite set of possible input patterns, it uses the Monte Carlo method, which is dimension independent. Thus, the number of samples required to make a reasonable estimation does not depend on the problem size.

To describe this approach, it is worth using the equation (2), previously presented, and make it general for a finite number of internal nodes or gate outputs. We can consider the signal $x_i(t)$ at node i, and $\alpha_{x_i}(T)$ as the number of transitions of x_i in the time interval $\left(-\frac{T}{2}, +\frac{T}{2}\right]$. The total average power [50] dissipated in the circuit during the same interval is given by

$$P_T = \frac{{V_{\mathrm{dd}}}^2}{2} \sum_{i=1}^{m} C_i \frac{\alpha_{x_i}(T)}{T} \tag{6}$$

Based on (6), the stochastic process $X_i(t)$ can be constructed as a family of the logic signals $x_i(t+\tau)$, where τ is a random variable. For each τ, $x_i(t+\tau)$ is a shifted copy of $x_i(t)$. Observing P_T, for $x_i(t+\tau)$ corresponds to measuring the power of $x_i(t)$ centred at τ. The random power of $X_i(t)$ over the interval $\left(-\frac{T}{2}, +\frac{T}{2}\right]$ is [50]:

$$\mathrm{P_T} = \frac{{V_{\mathrm{dd}}}^2}{2} \sum_{i=1}^{m} C_i \frac{n_{X_i}(T)}{T} \tag{7}$$

where, $n_{X_i}(T)$ is now a random variable, and since $X_i(t)$ is stationary, the expected average number of transitions per second is a constant:

$$E\left[\frac{n_{X_i}(T)}{T}\right] = \lim_{T\to\infty}\left(\frac{n_{X_i}(T)}{T}\right) \tag{8}$$

where, $E[\cdot]$ is the expected value operator. As a result, the power estimation problem is reduced to a mean estimation, which is a frequent problem in statistics. The average power can be expressed as

$$P = E[\mathrm{P_T}] = E\left[\frac{{V_{\mathrm{dd}}}^2}{2}\sum_{i=1}^{m} C_i \frac{n_{X_i}(T)}{T}\right] \tag{9}$$

Some statistical-based methods compute the power consumption of individual gates, making it too expensive to simulate each of the gates with randomly generated inputs. In other approaches, the total power consumed by the circuit is monitored without considering internal components. According to [50], the time to get sufficient accuracy using the total circuit approach is significantly less than in the individual gate monitoring approach. Statistical-based methods are also time-consuming but offer faster results in comparison to purely simulation-based methods. On the other hand, probabilistic methods suffer from the trade-off between speed and accuracy, making the fast implementations inaccurate.

2.5 Power Modelling Techniques

The power modelling techniques can be used to find the relationship between the power consumption and other metrics such as voltage, frequency, input transition density, and input temporal and spatial signal correlation [2]. The power modelling approaches can be classified as analytical, table-based, polynomial-based, and neural networks. Some parameters serve as qualitative metrics for different modelling strategies. Those parameters are the modelling level, effort, granularity, and characterization technique. The modelling level refers to the abstraction level of the model. The modelling effort analyses the effort that is needed to build the model. A low modelling effort might only require few simulations to construct the power model, whereas a high modelling effort typically means multiple iterations of the design flow. The model granularity represents the information level that is used to build the model. In the fine-grain granularity, bit-level information such as transition densities and static probabilities are needed. On the other hand, the coarse-grain granularity models include more abstract information such as the

number of operations, data width, and others. Finally, power modelling methods require a characterization phase based on estimated or measured values [2].

2.5.1 Analytic Modelling

Analytic approaches try to model the power consumption through switching activity and capacitance of the design [51]. This modelling type can be divided into activity-based and complexity-based techniques. The capacitance is roughly estimated from the design architecture in the complexity-based strategies, but input patterns are not considered. Another strategy is to evaluate the number of equivalent gates of the circuit and linearly model the design's power consumption since the power dissipated by a 2-input NAND is known. The main limitation of this approach is that it does not rely on switching activity and thus do not adequately consider the dynamic power consumption.

On the other hand, activity-based methods model the power from the entropy concept, which is used to evaluate the average activity of the design. Even though most approaches have focused only on dynamic power modelling, most recent analytical approaches model area, delay, and static and dynamic power during design exploration [2]. The main drawback of analytical power modelling is the glitches in digital circuits, which cannot be adequately modelled for complex systems like baseband processors.

SRAM Power Model

Analytical models are generally suitable for functional blocks with a regular organization, such as cache memories, register files, queues, and buffers [52]. The fixed structure of these functional blocks allows to simulate small portions of the pattern, such as the static random-access memory (SRAM) individual cells, and then linearly scale up the total capacity of the design. For example, the structure of a register file formed by six-transistors (6T) SRAM cells is depicted in Figure 24.

Analytical models can be implemented for several structures using different equations per each type; however, the switching activity of more heterogeneous structures make this approach unfeasible for large and complex systems.

2.5.2 Table-based Modelling

The lookup table (LUT) or table-based modelling approach tabulates power values [2]. Those power values are obtained by an earlier power characterization of smaller components. With this approach, it is also possible to easily interpolate results to get intermediate missing power values in the table.

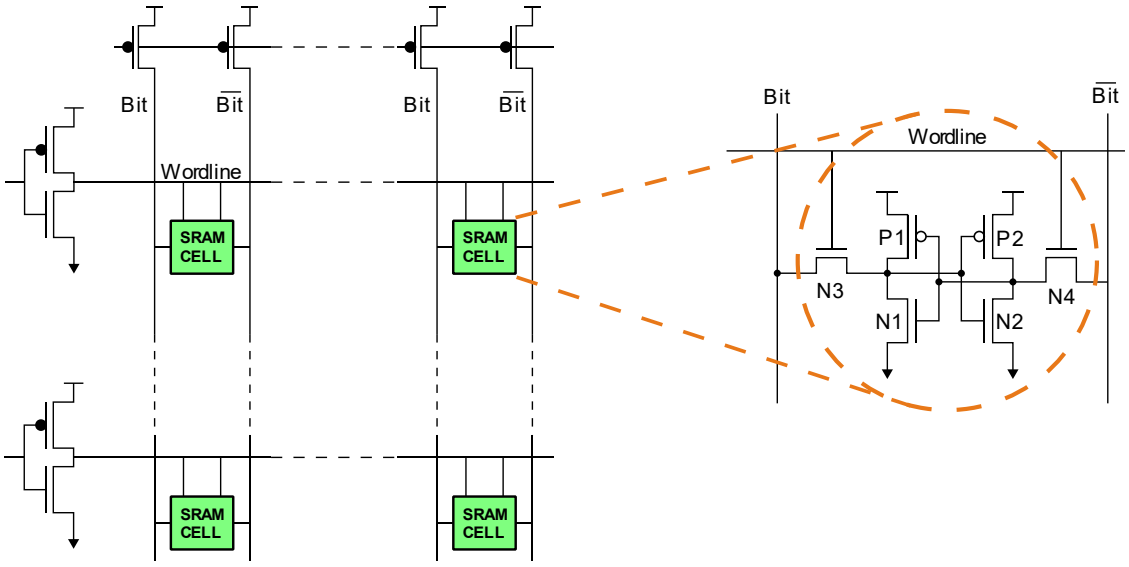


Figure 24. *Register file schematic (left) and a typical 6T SRAM cell structure (right). Based on* [52].

The table-based models can be constructed using input and output signal metrics such as the average signal probability, transition density, spatial and temporal correlation, and others. Furthermore, this modelling approach is robust since any function can be built up with the desired accuracy at the cost of increasing the table dimensions and size [53]. Therefore, the trade-off between accuracy and the computational effort to construct the LUT must be considered before applying this modelling approach.

Compared to analytical modelling, the table-based approach does not require any mathematical model and is easier to build. Therefore, these techniques have gained considerable research interest since it is possible to easily consider glitching activity, spatial and temporal correlation, increasing accuracy. However, the accuracy of these methods mainly depends on the quality of the characterization phase. Another limitation is the necessity of storing many data and the increase in computational effort as tables grow [2].

2.5.3 Polynomial-based Power Models

Polynomial or regression-based models are statistical inference methods to determine the linear relationship between power and one or more design parameters [52]. In power estimation, the dependent variable for the regressions is the observed power response. In contrast, the independent variables can be architectural variables such as clock frequency, voltage, memory configuration, or system variables like design area, mean activity rate, and others.

These approaches also need a power characterization phase generally performed at a low level. As for dynamic power, leakage power models can also be modelled using regression by considering parameters like CMOS channel length and doping, temperature, and others. Figure 25 shows offline regression-based modelling to establish model coefficients through a set of training benchmarks. The output of power simulators is used with correlation analysis to find out the model coefficients through regression analysis. Regression is generally done through the method of least squares. Once established, these coefficients are used to predict power without performing runtime simulations [52].

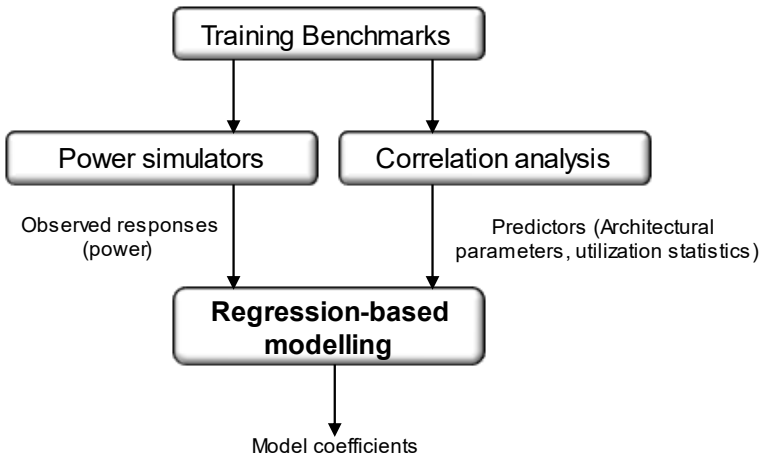


Figure 25. *Regression analysis for estimating model coefficients. Redrawn version of* [52].

Basic linear regression modelling can be too simple to estimate the power consumption of complex systems. Furthermore, polynomial-based approaches are limited by the number of input variables during modelling. They allow to find a linear relationship between few variables; however, they cannot solve non-linear problems accurately [2].

2.5.4 Neural Networks Based Techniques

Recent power modelling techniques are based on machine learning and artificial intelligence (ML/AI) approaches since they are very efficient in pattern recognition problems, prediction, control, and classification. Artificial Neural Networks (ANN) are suitable to model non-linear systems. These networks can establish complex and non-linear relationships between input and output variables. As illustrated in Figure 26, an ANN emulates connected neurons that propagate information among them. The input and output layers of the system are accessible; however, there are also hidden layers that affect the output but cannot be directly accessed. The ANN models the system behaviour using training sets to find the relationship between input/output signal statistics and the corresponding power dissipation [54].

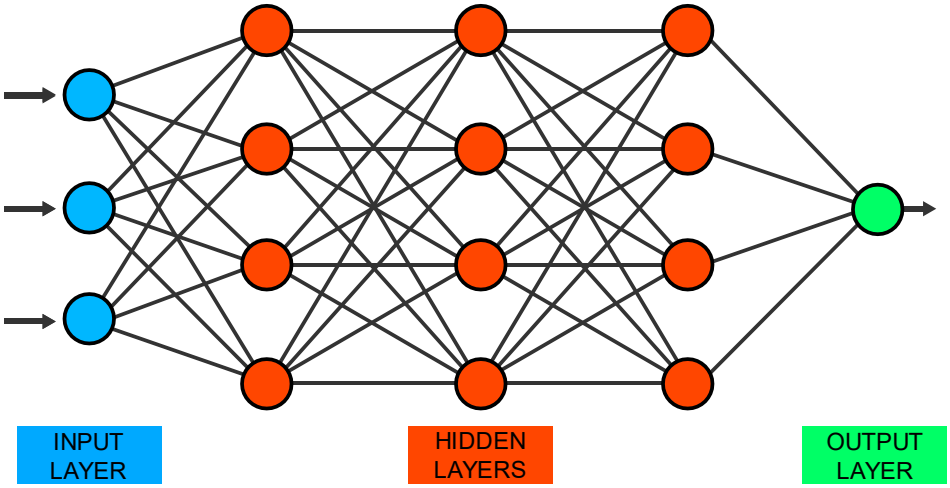


Figure 26. *Layer's representation of Artificial Neural Networks (ANN).*

Before building a neural power model, several parameters must be decided, such as the input data type, the number of neurons in each layer, and the size of the training set. The input data is generally related to input/output signal transition statistics. The number of hidden neurons typically depends on the complexity of the input/output relationship. However, there is no easy or general way to determine the optimal number of hidden neurons that must be used. Finally, the number of training samples can be calculated based on (10), where S is the number of samples, N_W is the number of weights to be trained, and β is the expected accuracy [54].

$$S > \frac{N_W}{1 - \beta} \tag{10}$$

Neural networks provide an efficient approach for developing predictive models and may achieve excellent accuracy with relatively low complexity [55]. However, the modelling effort is considerably higher than other techniques due to the training phase and large dataset [2].

2.5.5 Power State Machines

A power state machine (PSM) is a finite state machine (FSM) that is used for modelling the power consumption of a component, a subsystem, or an entire system [56]. The concept of power states is also part of the Unified Power Format (UPF) version 3.0, a standard that provides the ability for electronic systems to be designed with power as a key consideration early in the process [57].

A characteristic of PSM is that they do not influence the functionality or the timing of the simulation. The timed and functional simulations are used as inputs for PSM. They can be as simple as function calls to change from one state to another or more sophisticated

mechanisms such as observing transaction-level modelling (TLM) processes to determine the power states. The output of PSM is used to compute the power estimate, and these values can be actual power numbers or other parameters used to calculate the dynamic power such as capacitance, frequency, and voltage. Thus, it is feasible to calculate power consumption for different voltages and frequencies [56].

The PSMs are created manually by the designer of the simulation. For each state, the power consumption is obtained by simulations or actual measurements. The estimation is then calculated based on realistic scenarios. This approach falls in the linear table-based power modelling methods; however, it can use simple power states like active and idle, but it can also combine many transaction states such as read, write, hold, and others depending on the complexity of the system.

2.6 Review of Related Works

Since power consumption is a major concern in the CMOS electronic design process [58], there is a continuous interest in research for power estimation techniques. Thus, there are several studies on different power estimation methods and the different abstraction levels where they can be applied.

In [2], the authors address the key enabling concepts of power estimation and modelling techniques from RTL to transistor-level for FPGA and ASIC. Along with this, the existing works on power modelling and power characterization techniques such as analytical, table-based, polynomial-based, and neural networks-based are surveyed. Finally, they have also proposed a classification of the techniques according to defined metrics, intended to perform a fair comparison between the different approaches. This work can be used as a guideline to explore main research contributions on specific situations. The conscientious yet simplified comparison between various references makes this an important starting point for research on power estimation and modelling techniques.

Power macro models

The power macro modelling approach includes two main methods: equation-based and table-based methods. The equation-based method relies on construct equations that describe the power consumption of a circuit by analysing its inputs or outputs. However, this method is complex and less accurate since the relationship between inputs/outputs and the power consumption is difficult to model correctly. Conversely, the table-based methods extract some input/output data characteristics and the corresponding power consumption and stores those parameters in power libraries called lookup tables (LUT).

As a result, the table-based methods show better accuracy since they use actual measurements or simulations as a reference in the LUTs.

The authors of [59] have provided an RTL power macro model, which uses RTL simulations to generate power contours based on various sample inputs. The power contours are modelled using the statistical information of the inputs and the power calculated by RTL simulations. These power contours make it possible to estimate the power of any input vector by interpolating the position between the two closest contours. Finally, the authors have also provided the experimental results, demonstrating that the model improved the RMS error by 2.76% on average, compared to the best table-based method. Nonetheless, the parametrization to build up the power contours requires gate-level simulations, making it impractical for power estimations at early design stages.

State-based and functional-level approaches

A state-based approach was presented in [60]. It offers a formal model for different states in which the SoC can operate in several modes such as *Active* for full operation, *Idle* for the no-input-activity state, and *Sleep* for the fully clock-gated state. It introduces an early formal analysis and exploration for power design. The power consumption of different cores is modelled as individual power state machines (PSM). A PSM describes the power behaviour in a simplified form, as depicted in Figure 27 [61]. Finally, a symbolic simulation for all possible input combinations and specific scenarios is performed.

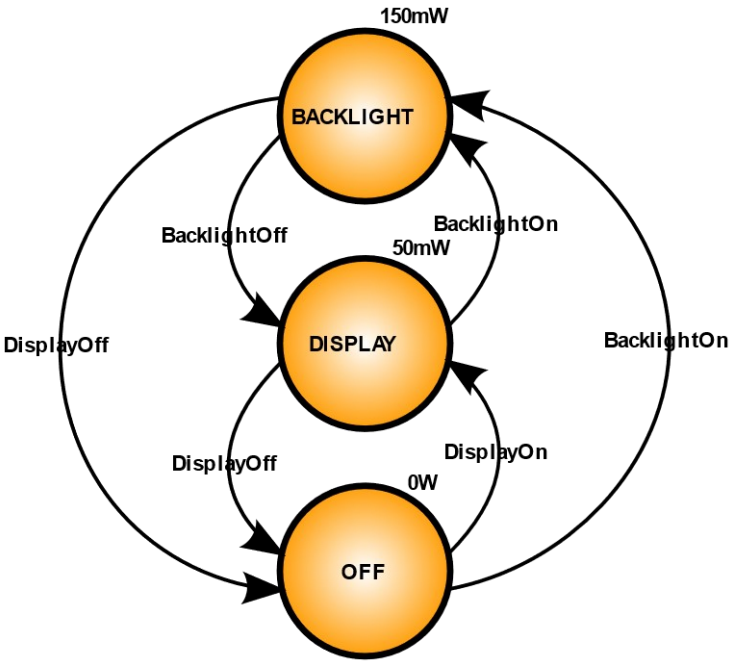


Figure 27. *Power state machine (PSM) for a simple display* [61].

Other similar to state-based works are performed in [62], [63], [64], [65], and [66]. Here, the researchers use Functional-Level Power Analysis (FLPA) techniques, which start with simple block diagrams of the architecture or the algorithms. Then, the power for each block and state is characterized by real measurements or by low-level simulations. The average error in [65], against real measurements, is about 2.4%. While in [66], the error against gate-level simulations is within 5%. Real measurements are not suitable for ASIC projects since power consumption on chipsets cannot be optimized once the real measurements can be physically performed. The gate-level characterization phase is acceptable to some extent but is insufficient to achieve sufficient accuracy in the early design stages.

A more generic but similar approach is presented in [67]. Here, the functionality of the gate-level blocks is translated into a module of the system-level language like SystemC, C++, and others. Applying a simplistic statistics-based model and not using any other power characterization make this approach wildly inaccurate. Nonetheless, its simplicity and fast running time shed light on future research on system-level power estimation.

Hybrid techniques

Hybrid power estimation techniques are presented in [65], [68], and [69]. The proposed methodologies are realized in two steps using a low-level characterization process and high-level system modelling. Different combinations of functional-, instruction-, and state-based methods are integrated with several power characterization strategies. Power figures come from low-level simulations, such as gate-level and even from real measurements.

A functional-level approach is used in [65]. The characterization is performed by reduced measurements in evaluation boards. The method is applied in different processors and using several signal processing algorithms. The results are compared with the specialized SPEC-95 benchmark, which evaluates CPU performance and power. The trade-off between accuracy and estimation time is quite acceptable and could be a good strategy for FPGA based designs. Nonetheless, as in previously presented methods, this approach is not feasible for ASIC projects due to its necessity of having real measurements for the characterization phase.

The authors of [69] present a low-level characterization process with a high-level system modelling approach. The characterization phase is performed by using internal power measurement tools of FPGAs, and the high-level modelling is based on SystemC. This work is the most relevant for our thesis since it focuses on hardware for baseband processing in the wireless communication domain. Therefore, it evaluates specific wireless

communication scenarios and presents analytical power models like the IFFT processor block for particular IP blocks. However, the research was intended for FPGA applications and the use cases analysed do not present the complexity and variety that actual base-band ASIC projects have. The most important limitation of this approach is that if the high-level model of an IP is not available in the library, designers must perform both the characterization and modelling of the IP block in SystemC. Therefore, it is an impractical approach from the easiness perspective for the designers.

3. POWER MODELLING AND SIMULATIONS

This chapter aims to model and simulate the power consumption of a Layer-1 subsystem which is part of a DFE SoC optimized for LTE MIMO and 5G. The final chipset is expected to have high flexibility since it will be used in several base station types, from macro to small cells. Therefore, performance and power consumption are expected to be state-of-the-art of their type. Similarly, the chipset design flow is devoted to maximizing the reuse of designed components in future projects. Thus, power estimation and modelling at early design stages give designers figures of power consumption and optimization opportunities and serve as a strong baseline for more accurate estimates in future projects.

The project aims to use a simplified high-level power modelling approach. The power model uses low-level simulations as a reference to estimate power consumption at different capacities and TDD patterns. The low-level simulations are mostly RTL simulations; nonetheless, those results are also taken for model calibration once it is possible to perform gate-level simulations.

The RTL simulations are executed individually for each IP component of the subsystem. As baseband subsystems must offer great flexibility, there are several modes of operation where blocks indistinctively can be either in the *active*, *idle* or *halt* mode. Those numbers are then used to estimate the total power the subsystem dissipates at different capacities. Finally, gate-level simulations are available at relatively late design stages. Those results are taken to calibrate the power model for future projects that will use the same or similar IPs. Section 3.1 describes the methodology used to simulate and build the power model. The power model construction is explained in Section 3.2, while the power simulation requirements and the different types of simulations performed in the project are presented in Section 3.3.

3.1 Methodology

The hybrid methodology starts once a functional version of the IP is ready as HDL code. The basic required input files are the RTL design files, activity stimuli vector, and standard cell libraries if available. All different operation modes should be simulated if functionality is already implemented. Otherwise, only available modes must be considered for the power model, but the limitations must be annotated.

Once all IP blocks are simulated at least once, the first round of power modelling must be carefully performed considering the actual limitations. For example, if *halt* mode is not yet implemented in some blocks, *idle* power numbers must be considered since that power consumption is closer to *halt* mode than the *active* mode numbers. The main objective is to give the first glance to designers and system integrators regarding the maximum power each IP and the subsystem dissipate.

Hereafter, average power for different capacities and TDD patterns is represented by using individual IP models. After having the model and the first power database, the designers and system integrators can obtain average power consumption for any test case with any capacity.

The RTL power simulations must be iterated several times to track the power consumption evolution as the design progresses. Those RTL simulation iterations are carried out until the first netlist for gate-level simulations is available. The model is then calibrated according to more realistic gate-level simulations. Figure 28 summarizes the methodology described above.

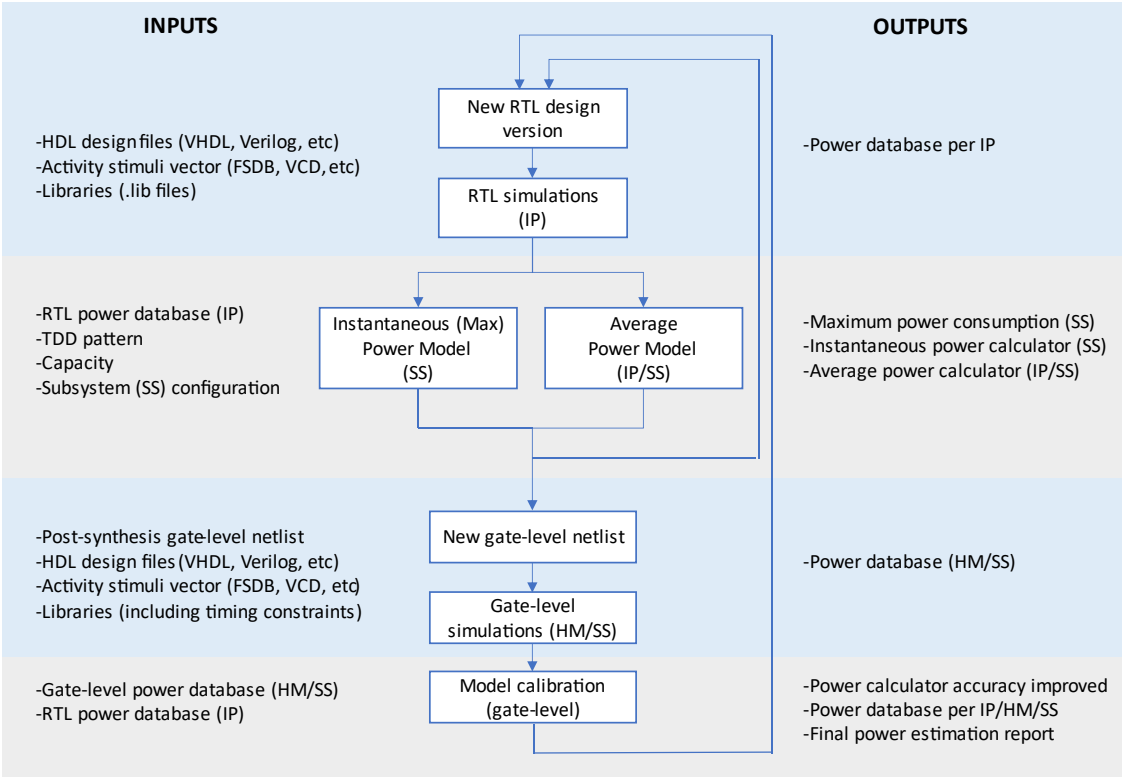


Figure 28. *Methodology for IP/HM/SS power modelling and simulations.*

The process must be repeated for each design release, preferably until the RTL code is frozen. Then, the final simulations can be performed, and the power estimation process

ends. The power analysis for the SoC continues once the chip's first engineering samples (ES1) are available for real measurements.

3.2 Power Model

Given the fast pace of the SoC design flow, a simplistic approach is used to model the power consumption in the subsystem. A combination of functional-level power analysis and high-level power modelling was chosen to estimate the power consumption quickly. For functional-level analysis, RTL simulations are performed for the three main states of the subsystem blocks: *active*, *idle* and *halt*. *Active* mode means the block is on and processing data during the whole analysis time. In the *idle* state, the block is clocked, but there is no data traffic. Finally, in *halt* mode, the block is mostly clock gated to avoid unnecessary power wastage. Clock gating is a technique to save switching power by removing the clock signal when the circuit is not in use. In addition, *halt* mode is helpful to switch off non-functional blocks during TDD operation, e.g. some receiver blocks can be clock-gated during transmission time, avoiding unnecessary switching activity.

The power is modelled for both subsystem and IP levels. The model describes the power consumption at the IP level by using the low-level simulation results and the expected capacity for a given scenario. In the subsystem model, the individual power contribution of each IP is simply added up. The model is also used to estimate power consumption on different TDD cases.

3.2.1 Subsystem Model

A high-level modelling approach is chosen because the subsystem components are designed and treated as blocks in the design flow process. Figure 29 depicts the individual blocks of the subsystem for a data path for both downlink and uplink paths. The colour represents the functional state of each block in a typical TDD operation, where most of the DL blocks are in *active* mode (green), while for UL, most of the blocks are in *halt* mode (yellow). The bypass block in DL is the only one in *idle* mode (red).

The internals of IP blocks are treated as black boxes, which means none of the internal processes is known. In this case, even the input and output data are not monitored in the high-level analysis. The only output of interest is the power consumption for each state, which is stored for a table-based power estimate.

Functional level power analysis

The activity in their building blocks causes the power consumption on a processor. Functional Level Power Analysis (FLPA) exploits this fact for high-level power estimation [56].

The power consumed by each block during different modes of operation can be estimated through low-level simulations or actual real measurements. The last option is only feasible for FPGA prototypes, where changes on design are still feasible at relatively late stages. Meanwhile, ASIC projects need low-level simulations since real measurements are feasible only after the chipset's first engineering samples (ES) are delivered. That offers impractical power optimization opportunities. After low-level simulations, the power numbers are stored in look-up tables to incorporate them into the high-level power model.

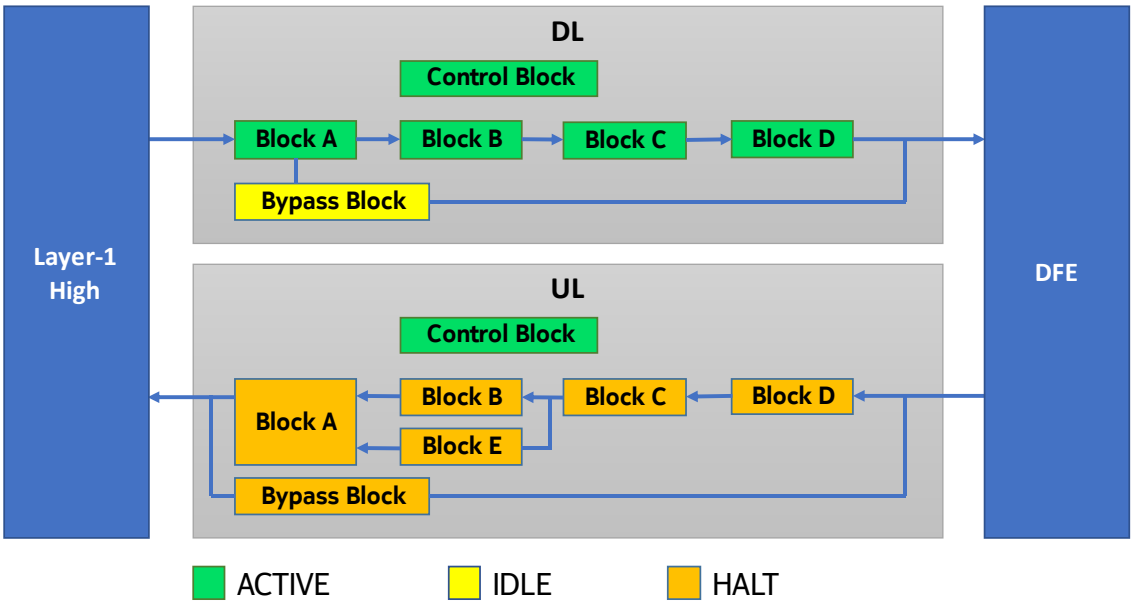


Figure 29. *Layer-1 Low subsystem building blocks during a TDD functional case.*

The next step is to model the total maximum power consumption of the subsystem. The maximum power dissipated by the subsystem, P_{SS-MAX}, is described by summing up the individual power consumption of each block component as

$$P_{SS-MAX} = \sum_{i=1}^{m} P_{IP-ACT_i} \tag{11}$$

where, P_{IP-ACT_i} is the power dissipated by block i in ACTIVE mode at maximum processing capacity. The number of blocks that compose the subsystem is denoted by m, but it can vary since there are components, like bypass blocks, that might be off during maximum capacity operation. In that case, those blocks must be suppressed for the calculation. It is important to estimate unrealistic peak power with all blocks in active mode since the subsystem might experiment with it intentionally or unintentionally during system integration. Therefore, designers and system integrators must know these numbers not to exceed the maximum power density and temperature allowed on the chipset

hotspots. It is worth it to mention that P_{IP-ACT_i} includes both static and dynamic power consumption of each IP.

By having the power consumption for each IP in the *active*, *idle*, and *halt* modes; any case of the subsystem instantaneous power can be built up as follows

$$P_{SS} = \sum_{i=1}^{m} P_{IP-ACT_i} + \sum_{j=m+1}^{n} P_{IP-IDL_j} + \sum_{k=n+1}^{p} P_{IP-HLT_k} \quad (12)$$

where, P_{IP-IDL_j} is the power dissipated by block j in IDLE mode, P_{IP-HLT_k} is the power dissipated by the block k in HALT mode. The blocks must be ordered from 1 to m for ACTIVE, from $m+1$ to n for IDLE, and from $n+1$ to p for HALT modes. This model allows getting a rough estimation of the instantaneous power consumption of the subsystem for any case. On the other hand, the average power consumption for different cases must be modelled considering the time each IP spends on each mode of operation. The individual power model for each IP is presented in the following subsection.

3.2.2 IP Model

Further action after calculating instantaneous power dissipation is to refine the model for different processing capacities and TDD patterns. Considering that low-level simulations are carried out using maximum processing capacity, we can also assume the maximum sample rate and the maximum number of active data paths as reference data. The capacity and TDD pattern are used to estimate the dynamic power consumption at different use cases linearly. For example, Figure 30 shows four TDD different scenarios with the corresponding percentage the IP would spend in each mode.

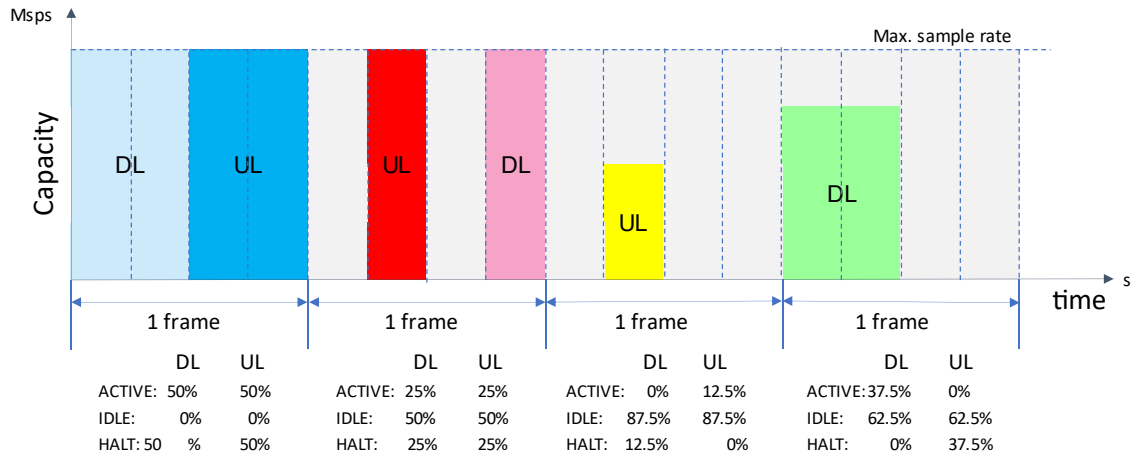


Figure 30. *TDD theoretical capacity scenarios for four frames.*

The total average power that an IP consumes in a specific TDD scenario can be sketched as the summation of static and dynamic power on different modes. The dynamic power is modelled considering the sample rate with respect to the maximum capacity and the percentage of time the IP spends in each mode as follows

$$P_{IP-AVG} = P_{static} + c_{ACT} * P_{dyn-ACT} + c_{IDL} * P_{dyn-IDL} + c_{HLT} * P_{dyn-HLT} \quad (13)$$

where, P_{static}, $P_{dyn-ACT}$, $P_{dyn-IDL}$, and $P_{dyn-HLT}$ correspond to static power, dynamic power for *active*, *idle*, and *halt* cases, respectively. These values are taken from RTL simulation results, while coefficients c_{ACT}, c_{IDL}, and c_{HLT} are the percentage of time the IP spends in each mode during a radio frame. These coefficients are calculated based on the throughput for each case, and the user can modify them accordingly to the TDD case. The following equation shows the coefficients for downlink; however, the generalization for uplink only needs to interchange DL instead of UL and vice versa. The coefficient for active mode is given by

$$c_{ACT} = \frac{throughput}{max.\ sample\ rate} * t_{TDD-DL} \quad (14)$$

where *throughput* is the amount of data the IP is expected to process, the *maximum sample rate* is the maximum capacity the IP can process, and t_{TDD-DL} is the percentage of time the IP spends in DL mode. Similarly, the c_{IDL} and c_{HLT} coefficients are easy to calculate, as is shown in the following equations.

$$c_{IDL} = \left(1 - \frac{throughput}{max.\ sample\ rate}\right) * t_{TDD-DL} \quad (15)$$

$$c_{HLT} = t_{TDD-UL} \quad (16)$$

where, t_{TDD-UL} is the percentage of time the UL chain is computing, which means the time when IPs of the DL chain are not in the *active* or *idle* modes. This does not always apply for control or bypass blocks since they can operate in *active* or *idle* mode depending on the subsystem configuration.

Design maturity coefficients

RTL simulations can be performed at early design stages; however, the functionalities in components at those stages are not fully implemented yet. Consequently, it is expected that the numbers are not realistic enough. Therefore, an SoC maturity weighting factor is added to the model to minimize the problem. This factor is adjusted according to the

gate-level simulation results, which are more realistic but come at the design flow's middle and late stages.

Gate-level simulations are less error-prone than RTL simulations; hence, the first gate-level simulations can calibrate the model through weight coefficients that depend on the maturity of the design. The basic idea is to match the gate-level power numbers with the previously calculated RTL power figures. The final calibrated model for the subsystem is derived from (12) as follows

$$P_{SS-AVG\_calib} = \sum_{i=1}^{m} D_{IP-ACT_i} P_{IP-ACT_i} + \sum_{j=m+1}^{n} D_{IP-IDL_i} P_{IP-IDL_j} + \sum_{k=n+1}^{p} D_{IP-HLT_i} P_{IP-HLT_k} \quad (17)$$

where, D_{IP-ACT_i} is the design maturity coefficient for the block i in ACTIVE mode, D_{IP-IDL_j} is the design maturity coefficient for the block j in IDLE mode, and D_{IP-HLT_k} is the design maturity coefficient for block k in HALT mode. The coefficients calculation based on RTL and gate-level simulations are given by

$$D_{IP-ACT_i} = \frac{P_{IP-ACT_i}(gate-level)}{P_{IP-ACT_i}(RTL)} \quad (18)$$

where, $P_{IP-ACT_i}(gate-level)$ is the average power the block j consumes in the active mode based on gate-level simulations, while $P_{IP-ACT_i}(RTL)$ corresponds to the RTL simulations results. D_{IP-ACT_i} can be calculated for each round of RTL simulations performed before the gate-level simulations.

3.3 Low-level Simulations

Most of the low-level power simulations in the SoC design flow are executed at RTL and gate-level. The RTL simulations are often performed at very early design stages since they do not demand timing and delay information. Instead, they only require design files, stimuli or activity vector files, and basic power libraries to estimate each type of cell's leakage and internal power dissipated. On the other hand, gate-level simulations require a post-synthesis netlist with detailed libraries regarding power, timing, and area of the cells that compose the design. Thus gate-level simulations are more reliable, but they are performed at relatively late stages of the design process.

Despite the less accurate results, RTL power simulations are important for designers to identify power-inefficient RTL code and implement architectural optimization. The most common approach is to perform individual power simulations per IP; however, it is also possible to carry out hard-macro (HM) and subsystem (SS) simulations.

3.3.1 Requirements

This thesis's RTL power estimation methodology requires an HDL description of the RTL design, power-characterized libraries, some synthesis and technology parameters, net capacitance information, and simulation activity data. RTL design files are generally presented in VHDL and Verilog file formats. Power-characterized technology and memory libraries are standardized in Liberty format (.lib). The synthesis and technology parameters include clock definitions, memory port definitions, threshold voltages, output load capacitance, signal transition time, and others. Finally, the net capacitance file is presented in wire load model (WLM) format [47].

A simulation activity file is a stimuli vector generated by designers and verifiers. This file contains the waveform of the whole signals present in the design. The Fast Signal Data Base (FSDB) file format is the most common for this activity data file. It is worth mentioning that designers and verifiers also provide the analysis time windows for the different operation modes of the design. Figure 31 depicts a general example of time window definitions, highlighting the first *idle* window. A general criterion is that the window time should be long enough to process at least two symbols; however, with 10us, we can also guarantee sufficiently accurate results. The initialisation period should never be considered for power simulations due to each block's unusual activity during switch-on.

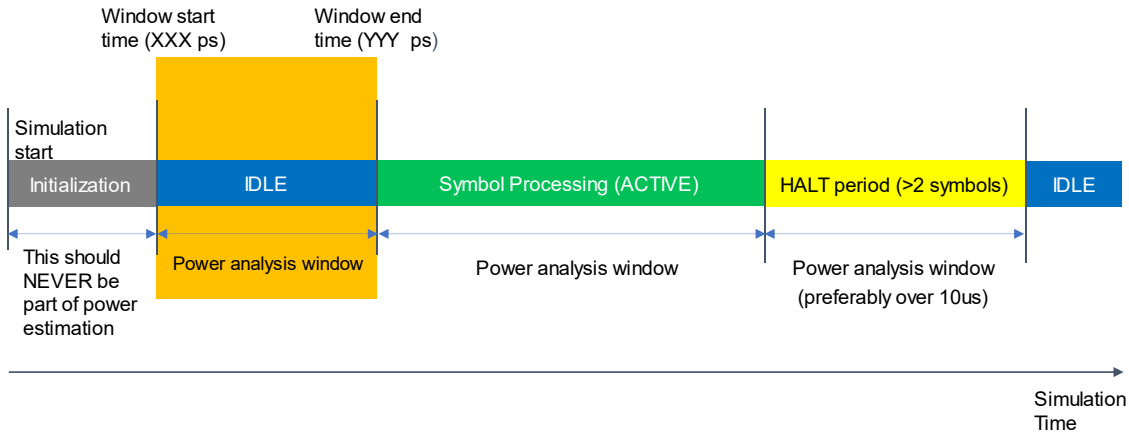


Figure 31. *Simulation time windows for active, idle, and halt cases.*

Finally, the commercial RTL and gate-level power tools are the most critical element of the simulation part of this methodology. These tools typically allow to perform average and time-based power estimation, either based on simulation activity files or without them

in the so-called “*vectorless*” approach, which uses probabilistic- and statistical-based methods as presented in section 2.4. More advanced features on power estimation commercial tools include automatic or manual RTL power reduction. That allows designers to rapidly find where the reduction opportunities are and whether they are feasible to implement or not. In addition, the outputs of commercial tools include summarized and detailed reports of power hotspots in the design, which are hierarchically ordered. Lately, those reports have been used to make the information more interactive by presenting it user-friendly through graphical user interfaces (GUI).

3.3.2 IP Block Simulations

Some IP blocks are completely new in baseband subsystems, while others can be inherited from past projects. If a block starts development from scratch, it is good to track the power evolution as frequently as possible and follow the power evolution for each new feature implemented. On the other hand, IPs with a past version may only require some modifications. In that case, the number of simulations is not so critical since the first simulation will probably give realistic numbers. Better yet, there could be a power database of the IP in previous projects.

Since each IP design has a different pace of development, the more accessible approach to estimate the power consumption at early design stages is to simulate the IP blocks individually. In this way, designers can get IP specific power reports and power reduction opportunities inside the block. Therefore, getting at least one round of power simulations for each IP is imperative before applying the power model. Once the first round of the IP simulation is done, it is up to designers to deliver new simulation files to assess the power evolution of the design.

Reference Power

Until the design code of this project was frozen for tapeout, it was possible to perform four rounds of RTL and two rounds of gate-level power simulations. The first three RTL power simulations were performed without any gate-level simulation reference. The exact numbers in watts are not included in this thesis due to confidentiality reasons. However, the total power of the subsystem is used as a reference in this thesis. The total reference power is the maximum power consumed by each IP in active mode during the first round of RTL power simulations, multiplied by the number of times each block is used in the subsystem. Table 1 shows the number of blocks that constitute the subsystem.

3.3.3 Hard-Macro Simulations

The gate-level simulations in this project were performed at the HM level. As was stated in Section 2.3.1, a hard macro (HM) is a set of one or more IPs wrapped into a higher hierarchy. Thus, several HMs constitute the subsystem, and in turn, each HM is formed by one or more IP blocks. Table 2 summarizes each HM composition in the subsystem.

RTL power simulations were also performed at the HM level to compare the results with the gate-level simulations. The same simulation files, libraries, and analysis time windows were utilized. Those results are presented in the next chapter and use the same reference power used in the IP power simulations.

Table 1. *Number of blocks that constitutes the subsystem*

	IP	Number of blocks
DL	**Block A**	4
	Block B	4
	Block C	4
	Block D	4
	Control Block	4
	Bypass Block	1
UL	**Block A**	4
	Block B	4
	Block C	4
	Block D	4
	Block E	4
	Control Block	4
	Bypass Block	1
Subsystem	**Grand Total**	46

Table 2. *Subsystem and Hard Macro composition*

	HM	Number of HMs	IP Blocks
DL	**HM1**	4	Block C
	HM2	4	Block A, Block B, Block D
	HM3	1	Control Block
	HM4	1	Bypass Block
UL	**HM5**	4	Block C
	HM6	4	Block A, Block B, Block D
	HM7	1	Control Block
	HM8	1	Bypass Block
Common	**HM9**	1	Interconnect Blocks

3.3.4 Subsystem Simulations

The last piece of work on the power simulation segment is to perform a complete simulation of the entire subsystem. At this stage, most IP functionalities are implemented, the subsystem is integrated and ready to be merged to other subsystems of the SoC. Regardless of the short simulation times that RTL offers, these simulations take several hours due to the size and complexity of the subsystem.

These simulations are not intended for optimization; instead, they confirm the power consumption with final versions of each IP, HM, and subsystem. The results can be compared with the results obtained with the power model from both RTL and gate-level simulations. Charts and analysis for all power simulation results are presented in the next chapter.

4. ANALYSIS AND COMPARISON

The first purpose of this thesis is to describe how well the early RTL power simulation figures correlate with the final and more accurate, gate-level power simulation results. The results of Chapter 3 are presented in the next sections. Section 4.1 presents the power simulation results for IPs, HMs, and the subsystem. Finally, the power model results are unveiled in Section 4.2. All data presented is given in percentages with respect to the overall power consumed by the 46 blocks that compose the subsystem at maximum capacity, that is, in the *active* mode. These numbers were obtained in the first round of RTL power simulations. They are used as a baseline for the whole analysis.

4.1 Power Simulation Results

There were four rounds of RTL power simulations at IP level and three rounds of gate-level simulations at HM level in our project. The RTL simulations were spread over time so that the power evolution of the design could be followed through them. On the other hand, gate-level simulations came at late design stages and would only reflect the evolution of the design in a relatively late stage of the design process.

In the RTL simulations, there were IP blocks where designers required more than one simulation per round. Interestingly yet expected, those more simulated blocks showed better optimization performance at the final releases. Contrarily, gate-level simulations were neither performed per IP nor used for power optimization. Due to their accuracy, gate-level simulations were intended only to confirm the entire subsystem power consumption.

4.1.1 RTL Power Simulations per IP

The results for all rounds of RTL power simulations in *active* mode are introduced in Table 3. As was previously stated, the numbers are expressed in percentages with respect to the overall subsystem power obtained in the first round.

Figure 32 summarises total power consumption by IP at different operation modes in the first round of RTL power simulations. Block C is the most power-hungry block in both DL and UL paths, with around 62% of the subsystem total power sum. These figures serve designers to identify the power hotspots inside each IP and gave them the first optimization reports to consider power reduction opportunities at the architectural level.

Table 3. *RTL simulation results per IP block. Active case.*

IP		Total power [%]			
		1st Round	2nd Round	3rd Round	4th Round
DL	Block A	2.46%	2.07%	2.93%	3.24%
	Block B	6.79%	4.78%	5.58%	3.20%
	Block C	28.83%	21.46%	19.80%	15.71%
	Block D	1.66%	2.32%	4.32%	4.38%
	Control Block	5.29%	5.16%	5.55%	5.74%
	Bypass Block	0.76%	1.33%	1.38%	1.59%
UL	Block A	1.92%	4.03%	3.91%	3.98%
	Block B	5.52%	5.77%	6.88%	3.81%
	Block C	32.78%	11.06%	17.34%	21.29%
	Block D	0.36%	0.47%	0.94%	0.89%
	Block E	6.98%	6.92%	8.58%	6.85%
	Control Block	5.29%	5.31%	5.98%	5.91%
	Bypass Block	1.36%	1.51%	1.52%	1.61%
Grand Total		100.00%	72.20%	84.70%	78.19%

Results of rounds 2 and 3 presented notable power reduction in most IPs. Those results were considered for the power model and are shown as a reference in Appendix A. The fourth and final round of RTL power simulations is summarized in Figure 33. There is a notable improvement in power consumption for Block C in both DL and UL paths. Despite introducing new functionalities in the final version, the designers were able to reduce the maximum power consumption of Block C (DL + UL) from 62% to 37%. Similarly, Block C (UL) in *idle* mode reflects an important improvement from 25% to 6%.

Finally, the evolution of each IP power consumption for the *active* case is shown in Figure 34. The *active* case is taken since it is the most power-consuming and relevant case for any IP. Results for *idle* and *halt* cases are annexed in Appendix B. The power for the *active* case shows a variety of tendencies. Seven blocks could be considered stable power contributors throughout the whole design process since their power contribution does not exceed 2% of difference in any round of simulations. Those IPs are Block A, Bypass Block, Control Block in DL, and Blocks D, E, Bypass Block and Control Block in UL.

The second tendency group correspond to two blocks in the DL path: Block B and C. They show a constant power decrease in each round of simulations. The most relevant is Block C since it is the most power-hungry block in the DL path. It showed a power reduction of 13% if comparing the first with the last round of simulations. On the other hand, Block C in the UL path showed a deep decrease of 21% in the second round. However, the power was increased in the following rounds, yet not exceeding the first-round power. The remaining blocks showed variable tendencies such as constant power increase (Block D in DL), steady growth with a final decrease (Block B in UL), and a slight rise in the second round but stable power until the final round (Block A in UL).

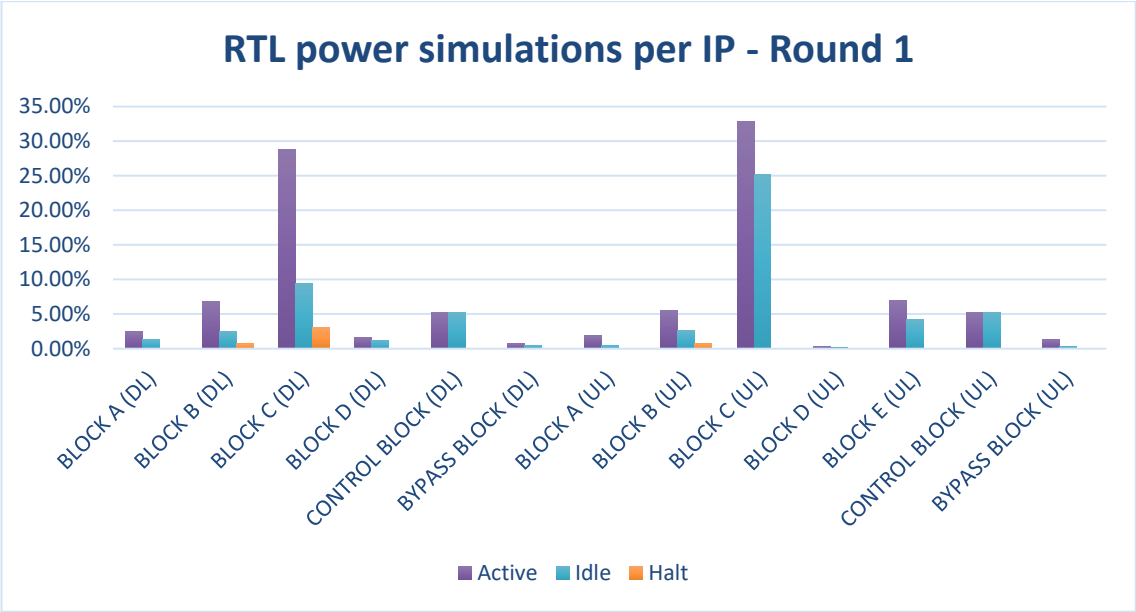


Figure 32. *Round 1 of RTL power simulations per IP and per mode of operation. 100% equals the overall subsystem reference power of Section 3.3.2.*

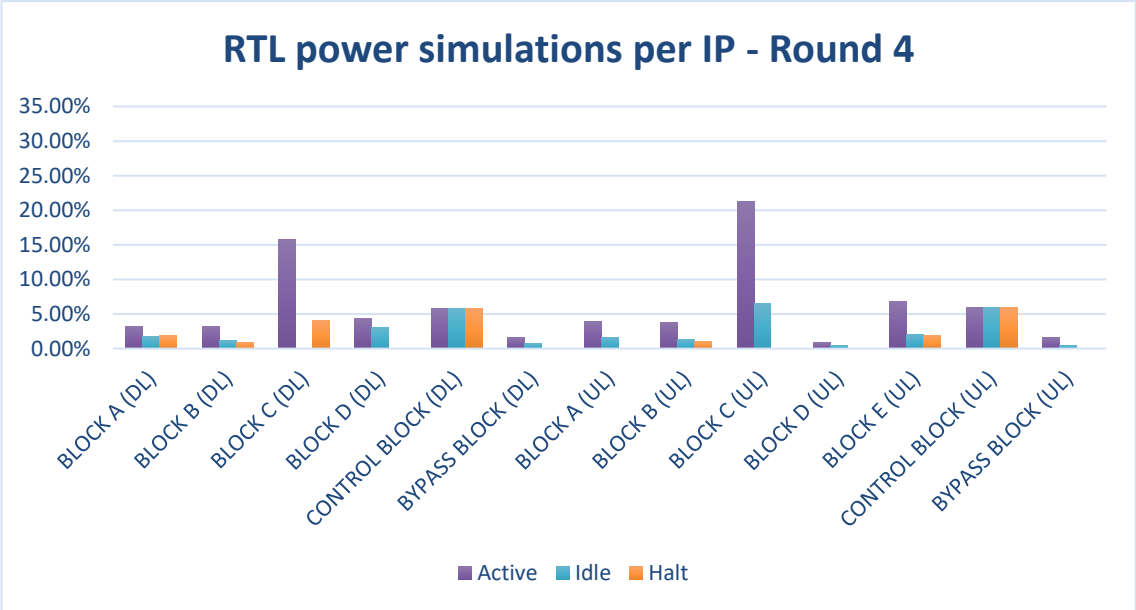


Figure 33. *Round 4 of RTL power simulations per IP and per mode of operation. 100% equals the overall subsystem reference power of Section 3.3.2.*

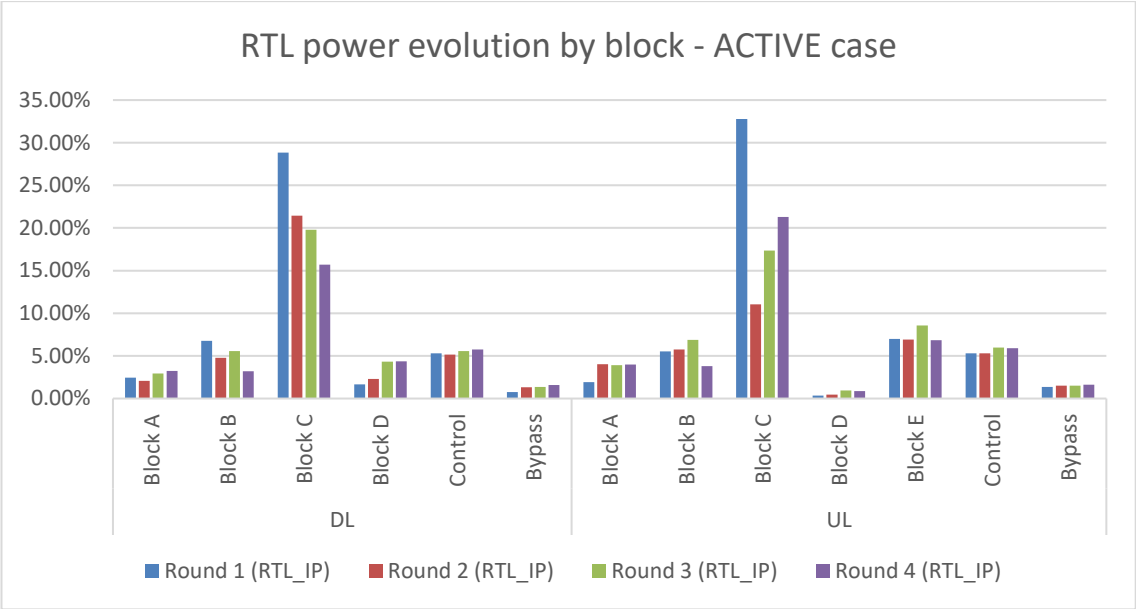


Figure 34. *RTL power evolution by block for the active case. 100% equals the overall subsystem reference power of Section 3.3.2.*

4.1.2 RTL vs Gate-level Power Simulations per HM

The gate-level power simulations were only performed for HMs. The RTL simulations were performed using the same libraries, activity files, and analysis time windows. In addition, the simulations were performed using different commercial tools labelled Tool A for RTL and Tool B for gate-level. The results for the *active* case are presented in Table 4 and Figure 35 as percentages with respect to the reference power.

Table 4. *RTL and gate-level power simulations per HM. Active case.*

	Total power [%]					
	RTL (Tool A)			**Gate-level (Tool B)**		
HM	**1st Round**	**2nd Round**	**3rd Round**	**1st Round**	**2nd Round**	**3rd Round**
HM1	27.8%	21.2%	18.8%	21.4%	18.4%	16.9%
HM2	15.6%	16.3%	12.7%	13.0%	13.8%	11.3%
HM3	1.1%	1.1%	1.3%	1.3%	1.2%	1.4%
HM4	0.7%	0.7%	0.7%	-	-	0.9%
HM5	34.2%	26.4%	24.8%	28.6%	22.8%	22.9%
HM6	19.2%	18.4%	16.7%	13.6%	12.9%	12.9%
HM7	1.4%	1.4%	1.4%	1.6%	1.6%	1.7%
HM8	0.5%	0.6%	0.6%	-	-	0.6%
HM9	0.5%	0.6%	0.5%	0.5%	0.6%	0.5%

The power for the *active* case is mainly concentrated in four Hard Macros: HM1, HM2, HM5, and HM6, which accumulated around 64% of the reference power in the last gate-level simulations. The average deviation of the RTL simulations with respect to the gate-level results observed in the first, second, and third rounds were 12.9%, 10.8%, and

1.3%, respectively. The major deviation was observed in HM6, with a final deviation of 29.3% in the last round.

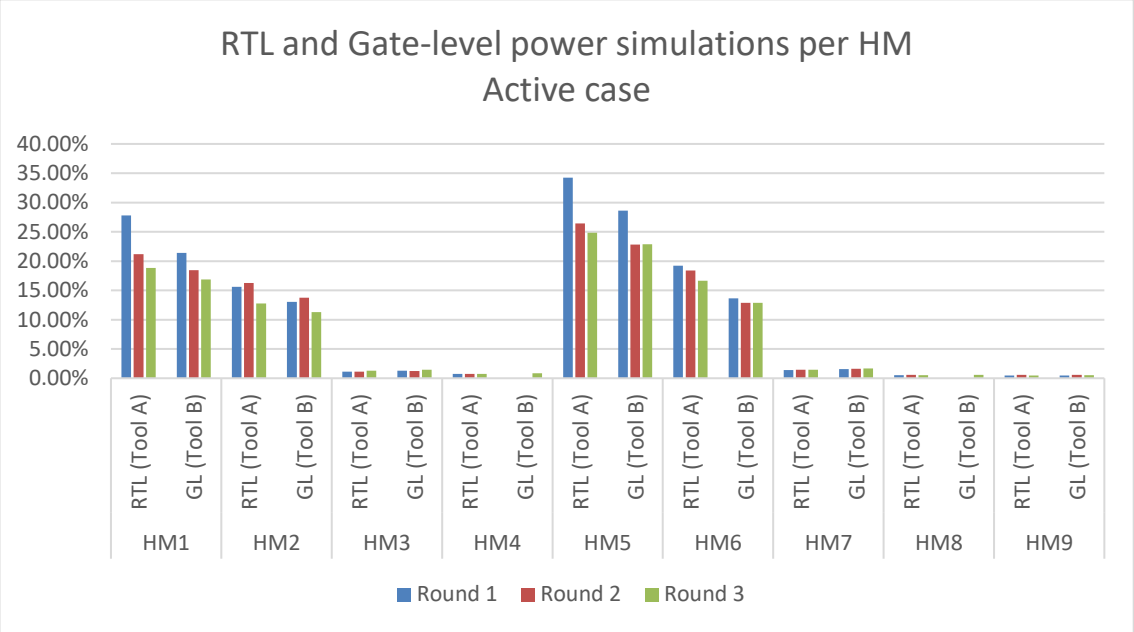


Figure 35. *HM power simulations for RTL and gate-level. Total power for Active case. 100% equals the overall subsystem reference power of Section 3.3.2.*

4.1.3 Subsystem RTL Power Simulations

At the final estimate before the last power simulation, our subsystem comprised around 100 million gates. That makes it feasible to perform only RTL power simulations due to the long-running times of gate-level simulations. The results of the three rounds completed before the tapeout are presented in Table 5 and Figure 36. Table 5 summarizes the RTL power simulation results for the three operation modes of the subsystem.

Table 5. *Subsystem RTL power simulation results*

Path	Mode	Total power [%]		
		1st Round	2nd Round	3rd Round
DL	Active	83.7%	64.3%	67.7%
	Idle	47.2%	33.7%	44.9%
	Halt	48.9%	35.5%	46.4%
UL	Active	83.6%	71.4%	66.5%
	Idle	47.3%	33.9%	35.4%
	Halt	62.4%	59.9%	58.9%

Interestingly, the latest design adjustments and optimizations show that power consumption was reduced for all three modes in DL and UL paths. Another remarkable fact in the latest round is that the UL path leads to higher power consumption than the DL path. This increase was primarily caused because the clock frequency in block C (UL) was increased from 614MHz to 800MHz. Finally, the sum of power in DL and UL paths in *active* mode results in around 134% with respect to the reference power.

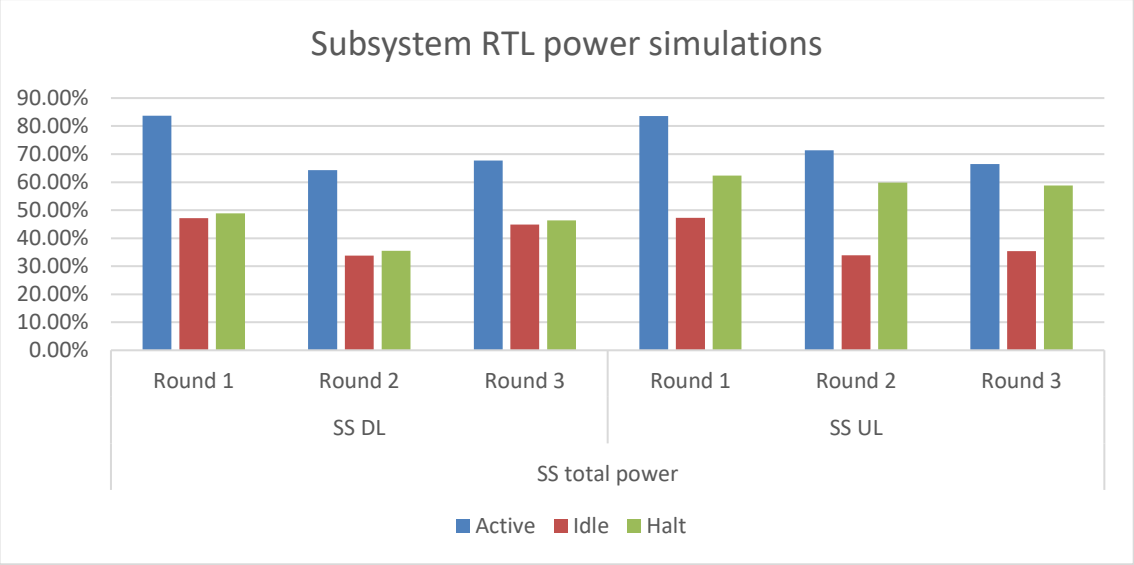


Figure 36. *Subsystem RTL power simulation results. Latest releases before tapeout. 100% equals the overall subsystem reference power of Section 3.3.2.*

4.2 Power Model Results

One of the objectives of the project was to predict power consumption in different scenarios. Therefore, a linear regression approach was used to model the power consumption at different TDD patterns and capacity cases. The generated model multiplies the resulting power consumption obtained by early RTL power simulations by the estimated percentage of time each block spends in different modes. Based on this information, a spreadsheet was generated containing the power estimates per block, hard macro, and the total power consumption of the subsystem. The table is parameterized with variables such as processing capacity and TDD pattern that the user can vary dynamically. Similarly, bypass blocks can be selected or deselected to calculate the total power consumption in each subsystem configuration.

4.2.1 Maximum Power Consumption

The maximum power consumed by the subsystem was modelled as an instantaneous power sum of each IP contribution, multiplied by the number of instances. The raw results without any adjustment and the model adjusted results are shown in Figure 37. Based on the total subsystem model, the power consumption decreased from 100% to 76%, a notable overall improvement, even though the latest version has more functionalities implemented than the first RTL version.

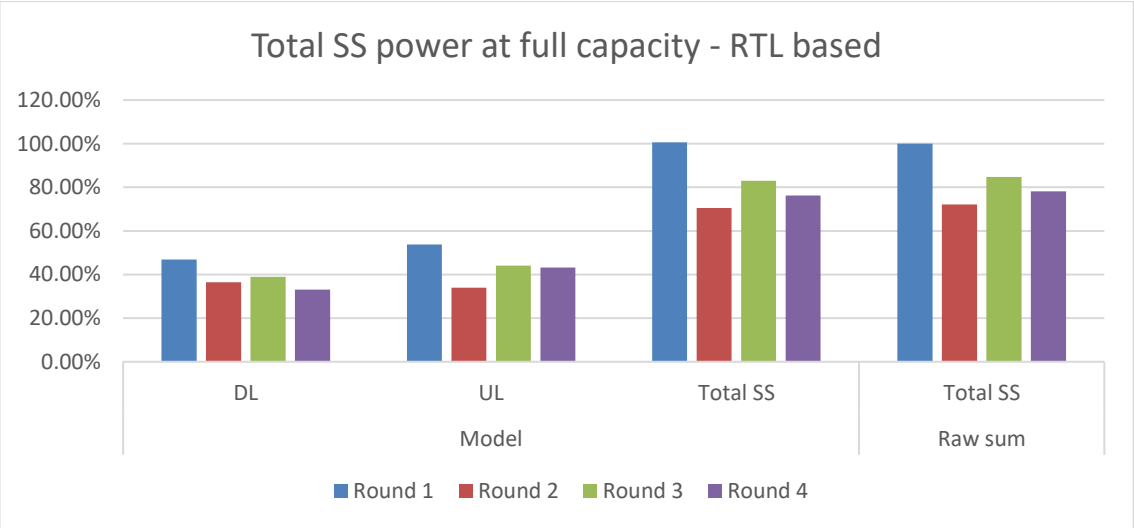


Figure 37. *Subsystem power consumption at full capacity. Based on RTL power simulations. 100% equals the overall subsystem reference power of Section 3.3.2.*

If compared to the raw sum of each IP contribution in *active* mode; the use of the model, that is, considering the power contribution of the *active*, *idle*, and *halt* modes, reflected a difference of 0.59%, 1.73%, 1.76%, and 1.9% for the first, second, third and fourth rounds respectively.

4.2.2 Power at Different Capacities

The final model can handle any capacity and TDD pattern selected by the user; however, the most relevant test cases were chosen to summarize the power estimation. Figure 38 shows those results, which include three FFD and three TDD cases. For FDD, the full capacity was considered for both DL and UL (both at 100% of capacity), as well as the case in which DL is at full capacity (100%), and UL is not active (0%), and vice versa. Although the last cases are not entirely practical, they serve as guidelines to find which path consumes the most in IDLE mode. Regarding TDD cases, the selection was simple: TDD36 case spends 25% of the time in DL and 75% in UL; TDD45 equally uses 50% for both DL and UL, and TDD55 operates 75% of the time in DL and 25% in UL.

As expected, the FDD full capacity case registers the major power consumption, but compared with the first round, the power on the final round decreased from 100.59% to 76.29%. On the other hand, all the TDD cases, which are more realistic scenarios, tended to have similar levels of power consumption, around 55% in the last round, representing a maximum difference of 4% between them.

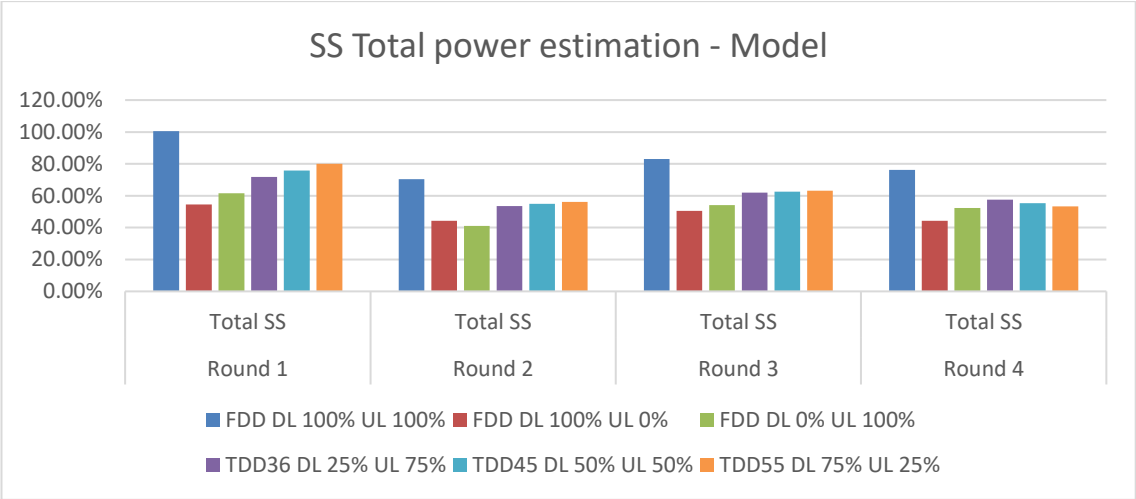


Figure 38. *Subsystem total power estimation for FDD and TDD use cases. Model adjusted using RTL power simulations per IP. 100% equals the overall subsystem reference power of Section 3.3.2.*

4.2.3 Power Model After Calibration

Finally, one of the objectives of this thesis is to present the model after it has been calibrated using the latest results of the gate-level power simulations. Figure 39 shows the results of the power model before the calibration. The results of the latest gate-level (GL) simulations are also included as a reference since they are expected to be closer to the real measurements. Expectedly, the model based on the first round of RTL power simulations gives the less precise numbers in most HMs. This is mainly because of the lack of implemented functionalities and not-yet-optimized IP blocks.

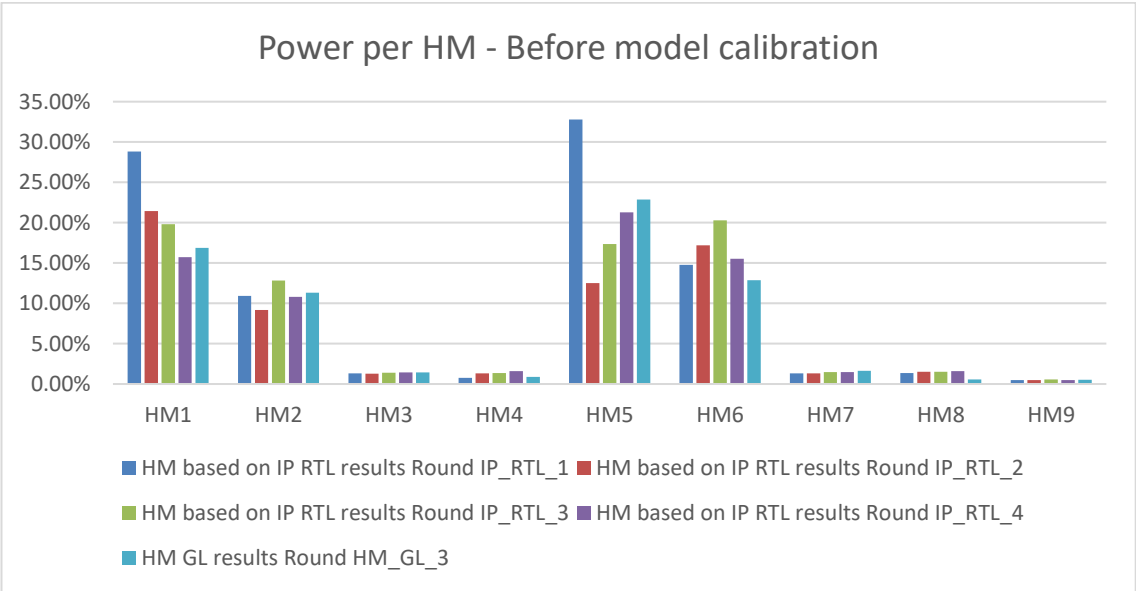


Figure 39. *Power per HM before model calibration. 100% equals the overall subsystem reference power of Section 3.3.2.*

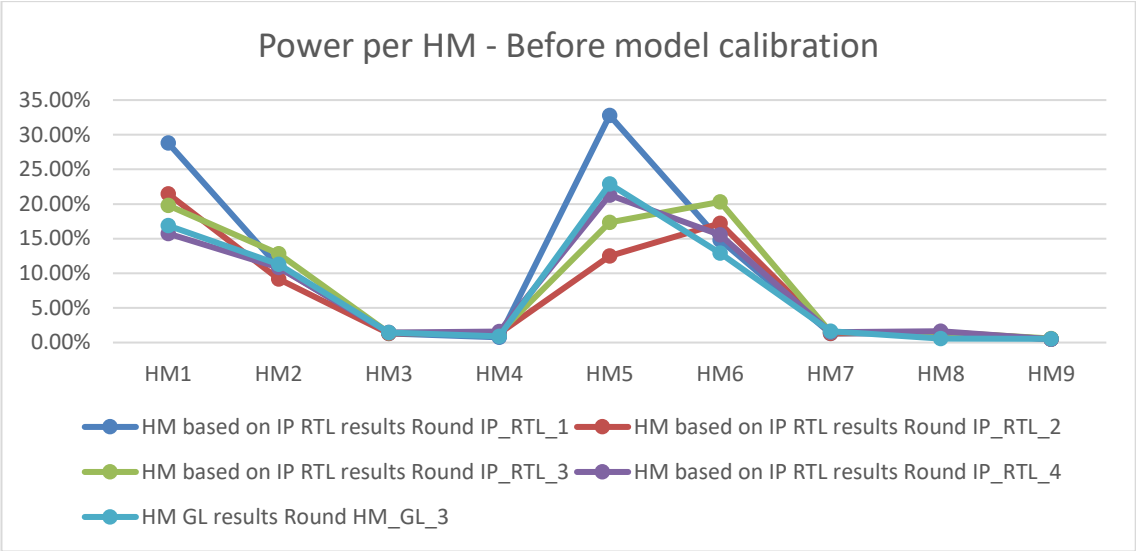


Figure 40. *Power per HM before model calibration.*

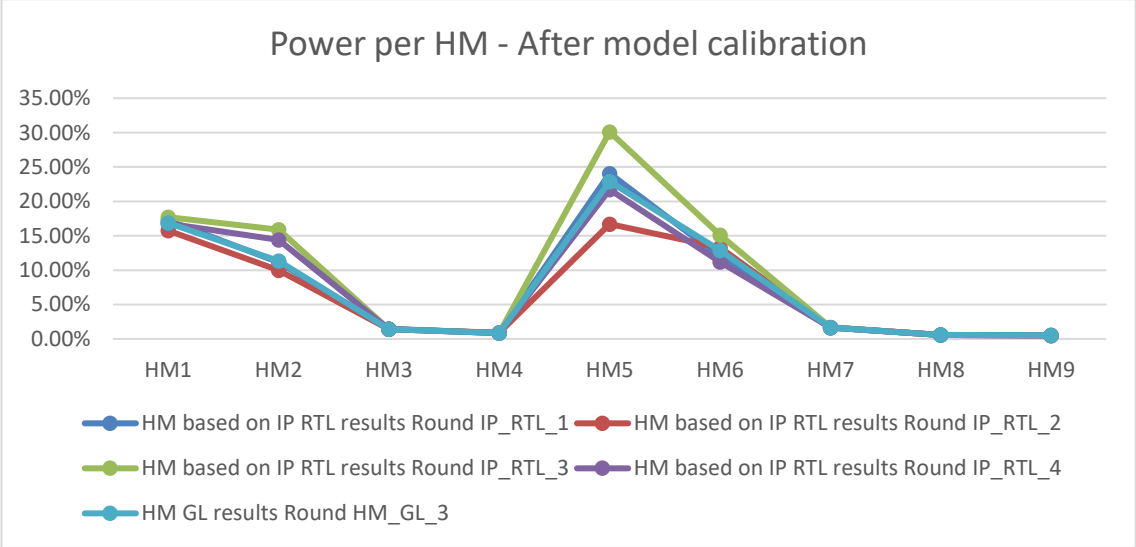


Figure 41. *Power per HM after model calibration.*

Even though the power of HMs is independent of each other, the bars of Figure 39 are converted into connected lines in Figure 40. That is merely referential to visualize in a better way how the power model improves the accuracy after model calibration. The goal is to match the lines of each round with the light blue line, which represents the most accurate results obtained in the latest gate-level simulations. As examples, HM3 and HM9 depict the smaller average gap between the model and the simulations. In those cases, the model will have the least weighted calibration coefficients. On the other hand, HM1 and HM5 show the most significant gaps, reaching a 12% of difference with respect to the gate-level simulation values.

Figure 41 shows the results after model calibration. The gaps in all HM are minimized, and the maximum difference was reduced from 12% to 0.4%. The average gap decreased from 2.11% to 0.92%.

4.2.4 Calibration Coefficients Analysis

The calibration coefficients are highly dependent on design maturity, and they showed several tendencies. First, there are blocks with power increasing trend over time, mainly due to the aggregation of new functionalities. However, few blocks have a power decreasing trend since designers performed optimization once the required functionality was achieved. Second, the two most power-hungry IPs were also the most power-optimized blocks; thus, the main contributors to the overall subsystem power reduction.

The power consumption in most IPs showed a linear tendency over the design maturity timeline. That allowed using simple linear regressions to calculate calibration coefficients. However, there was an IP block that showed a variable trend. In that case, a quadratic linear regression was used to improve the model accuracy. Figure 42 presents the final calibration coefficients per HM.

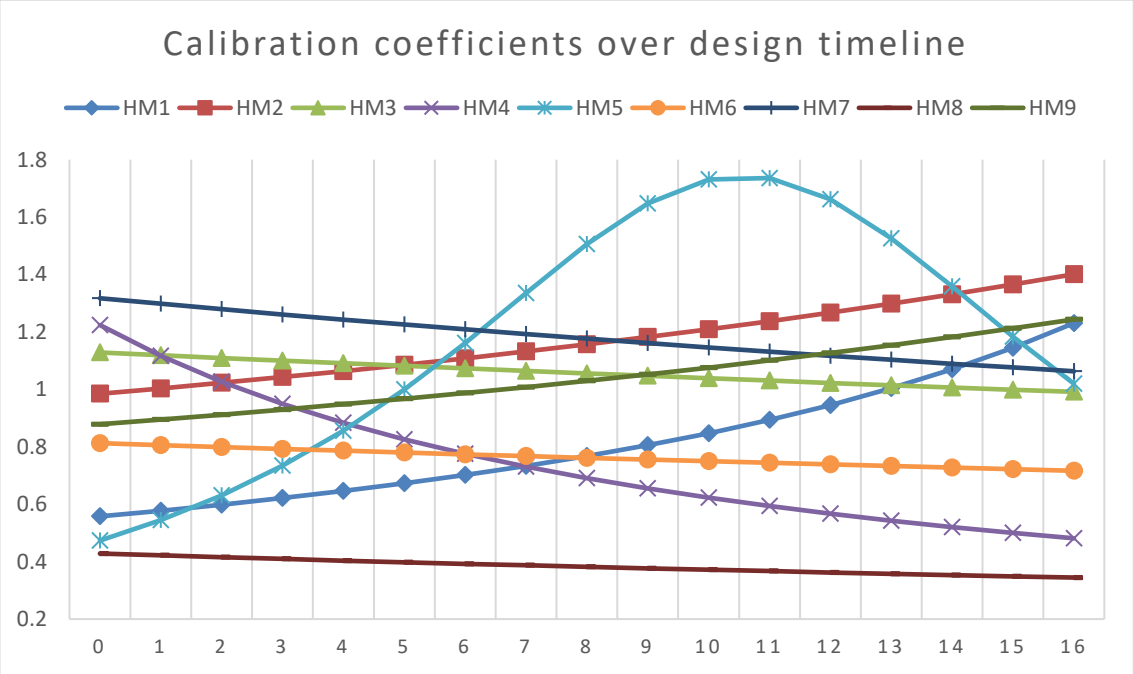


Figure 42. *HM calibration coefficients over design timeline.*

4.3 Error sources

According to [70], no physical quantity can be measured with complete certainty; however, we can reduce the uncertainty until reasonable amounts for the required application. The well-established way to describe uncertainty is by expressing errors. A premise

for calculating uncertainty is that the measured quantity must be the “same” each time [70]. Unfortunately, that is impossible in the actual project since we always measure different designs, starting from the basic functional blocks in the early design stages until the latest stable version of the fully integrated subsystem.

Furthermore, it is worth mentioning that after the first silicon tapeout order, designers continue working on new versions of the subsystem in case the first version presents problems during physical verification. Thus, the error numbers are merely referential to describe the precision of estimates concerning the gate-level simulations. The accuracy of the power models and simulations cannot be assessed until actual measurements in the engineering samples are performed. Therefore, the errors are calculated using the latest gate-level power simulations as a reference to determine the precision rather than the accuracy.

Estimating the power consumption of a complex subsystem composed of several IP blocks, designed at different paces, leads to several systematic errors. Those errors can be classified into four kinds as follows:

Instrumental:

- The use of two different EDA tools for RTL and gate-level simulations intrinsically gives different accuracies. The approach each of them uses to deal with power estimation at different abstraction levels is not mensurable unless a deep analysis is performed; however, that is out of the scope of this thesis.
- Generally, we assume that the EDA tools used for simulations are well enough calibrated. However, relatively simple parameters such as avoid declaring a clock can affect the results, even though it is expected that tools automatically recognize them.

Observational:

- The analysis time window per each IP block is left to designers and verifiers criteria because their knowledge of block behaviour can ensure choosing the correct analysis span. However, the criteria could vary between designers, especially if there are no unified requirements for the whole team.

Environmental:

- The connection between IP blocks and HMs may induce additional noise that is considered neither in RTL nor gate-level simulations.

- Structural changes in the design happened during the project. For example, the clock frequency increase on a block led to a drastic increment in power consumption, thus affecting the linear regression. Similarly, the bypass HM changed the number of IPs from 4 to 1. That enormously affected the error numbers for that HM.

Theoretical:

- In the linear regressions, the static and dynamic power are not separated to simplify calculations; this led to inaccuracies since switching activity have different impact in static and dynamic power.

In general, the RTL simulations results were higher than gate-level simulations; this approach is understandable since RTL has no timing and delay information; thus, EDA tool vendors tries to compensate that with pessimistic results rather than optimistic ones.

Finally, in this project, only three IP blocks implemented the *halt* mode for the first round of simulations, while in the last round, that number was increased to seven. When there was no *halt* mode information available, the model used *idle* power numbers for calculations, leading to pessimistic estimations.

Table 6. *Mean absolute error before model calibration*

	1st Round	2nd Round	3rd Round	4th Round
HM1	17.34%	6.66%	4.26%	1.68%
HM2	0.59%	3.09%	2.20%	0.71%
HM3	0.16%	0.21%	0.06%	0.01%
HM4	0.16%	0.66%	0.73%	1.04%
HM5	14.34%	15.04%	8.04%	2.32%
HM6	2.76%	6.24%	10.75%	3.83%
HM7	0.48%	0.47%	0.23%	0.26%
HM8	1.14%	1.37%	1.37%	1.51%
HM9	0.07%	0.07%	0.07%	0.08%

Table 7. *Mean absolute error after model calibration*

	1st Round	2nd Round	3rd Round	4th Round
HM1	0.58%	1.64%	1.21%	0.09%
HM2	0.23%	1.96%	6.62%	4.47%
HM3	0.03%	0.04%	0.00%	0.00%
HM4	0.03%	0.06%	0.02%	0.01%
HM5	1.70%	8.96%	10.45%	1.71%
HM6	1.40%	0.60%	3.22%	2.43%
HM7	0.02%	0.04%	0.05%	0.04%
HM8	0.00%	0.01%	0.01%	0.01%
HM9	0.05%	0.05%	0.03%	0.03%

Despite the limitations that errors might bring, the presented power model fulfils the project's main objective: to give designers and system architects simplified close-to-real estimates about the subsystem power consumption at different scenarios. The mean absolute error per HM without model calibration are presented in Table 6, while the results after model calibration are shown in Table 7. Since the power model tries to serve as a baseline for early power estimations, the most relevant case for our thesis is the first round. The mean absolute error for that round decreased from 4.12% without calibrations to 0.45% before model calibration.

5. CONCLUSIONS

Baseband SoC design is a complex process requiring multidisciplinary teams specialized in telecommunications standards, coding in several programming languages, electronics at the physical design level, project management, and other underlying electrical engineering and computer sciences fields. In general, this thesis presents a small yet essential part of nowadays telecommunication SoC development: the power-aware design.

Power consumption is a crucial factor in SoC development. The power estimation is the first step for designing high-performance, low-power SoCs. The designers can extend the complexity of power estimation flow as far as the SoC design complexity itself. However, designers look for simplicity since they mainly focus on functionality rather than on power-optimized designs. This thesis implements a simplified way of building up a hybrid power estimation method for IP blocks of baseband chipsets. The presented methodology combines a high-level power modelling approach with a functional-based power analysis, which consists of performing RTL power simulations at early design stages and compare the results with simulations of more mature versions of each IP. The final model is calibrated considering the latest gate-level simulation results, which are more accurate but only available at late design stages. The model leads to an accurate yet fast and computationally feasible way to obtain the power consumption for different testcases.

The architectural design of the subsystem changed during the development of the thesis. It is worth mentioning that throughout the design process, most IPs were optimized; some functionalities were added while others were discarded. Despite that, this approach made it easy to execute changes immediately without any other requirement than adjusting the spreadsheets according to the new architecture. The strength of this approach is that using accurate libraries for simulations gives more precise results; meanwhile, the easiness of having spreadsheet-based models makes it flexible and fast to determine the power consumption at an endless number of scenarios. Another merit of this approach is that the power model can be calibrated throughout the design process, increasing the precision in the late stages. Furthermore, if system architects reuse an IP block in other designs, there will be exact power estimates even for different test cases.

At maximum capacity, the overall subsystem power was reduced from 100% in the first round to 78% in the last round of RTL power simulations. As expected, the top power-hungry blocks in the subsystem are related to IFFT/FFT operations, OFDM generation, and filtering to combine the different bandwidth parts in a multi-numerology carrier.

Those blocks combined represented around 75% of the total subsystem power in the first round. However, that power dropped to 47% in the last round of RTL simulations.

A regression analysis was deployed for estimating the relationship between the power consumption as the dependant variable and the design maturity timeline as the independent variable. The results were used to calculate model calibration coefficients that will serve for future projects. By following the calibration coefficient regression, designers might analyse the power evolution through different functionality implementations in each IP block. The RTL simulations showed a variety of tendencies in the power evolution of each IP, mainly due to the nature of this type of project, where the different design paces in IP blocks might lead to power increases or reductions at different stages. Therefore, it is crucial to identify the block maturity to use the correct design maturity coefficients.

The system designers, architects, and modellers do not need to generate new simulation files for specific scenarios using the power model. Still, they can quickly and accurately calculate the expected power by selecting parameters like TDD case, throughput, parallelism, and operating mode of the bypass block. This is especially important in baseband SoC development since the number of test cases is quite extensive, and the power simulations for each scenario would lead to an additional workload for designers, verifiers, and system integrators.

A simplistic yet precise method was implemented to demonstrate its effectiveness in the fast-paced industry of SoC design for telecommunications. The initial power model based on independent IP block RTL power simulations achieved a mean absolute error of 3.06% concerning the final gate-level simulations. The calibrated model improved that accuracy, reaching a final mean absolute error of 1.3% when considering the overall power consumption and all the power simulation rounds. The IP power databases obtained, and the power model will serve as a reference for future projects that might use one or several IP blocks used in the actual subsystem. It is recommended to automate the process so that designers can get power numbers after each new IP version release, even without having activity simulation vectors. The next step for future projects is integrating the power estimation methodology in a continuous integration and delivery (CI/CD) manner to the SoC design flow.

REFERENCES

[1] E. Calvanese Strinati *et al.*, “6G: The Next Frontier: From Holographic Messaging to Artificial Intelligence Using Subterahertz and Visible Light Communication,” *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 42–50, 2019, doi: 10.1109/MVT.2019.2921162.

[2] Y. Nasser, J. Lorandel, J.-C. Prevotet, and M. Hélard, “RTL to Transistor Level Power Modelling and Estimation Techniques for FPGA and ASIC: A Survey,” *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 40, no. 3, pp. 479–493, Mar. 2021, doi: 10.1109/TCAD.2020.3003276.

[3] T. Huang, W. Yang, J. Wu, J. Ma, X. Zhang, and D. Zhang, “A Survey on Green 6G Network: Architecture and Technologies,” *IEEE Access*, vol. 7, pp. 175758–175768, 2019, doi: 10.1109/ACCESS.2019.2957648.

[4] P. K. Dalela, P. Bhave, P. Yadav, A. Yadav, and V. Tyagi, “Beam Division Multiple Access (BDMA) and modulation formats for 5G: Heir of OFDM?,” in *2018 International Conference on Information Networking (ICOIN)*, Jan. 2018, pp. 450–455, doi: 10.1109/ICOIN.2018.8343158.

[5] K. Benzekki, A. El Fergougui, and A. Elbelrhiti Elalaoui, “Software-defined networking (SDN): a survey,” *Secur. Commun. Networks*, vol. 9, no. 18, pp. 5803–5833, 2016, doi: https://doi.org/10.1002/sec.1737.

[6] E. Dahlman, S. Parkvall, and J. Skold, *5G NR: The Next Generation Wireless Access Technology*. Elsevier Science, 2020.

[7] D. T. Kiet, T. M. Hieu, N. Q. Hung, N. Van Cuong, V. T. Van, and P. N. Cuong, “Research and Implementation of eCPRI Processing Module for Fronthaul Network on FPGA in 5G – NR gNodeB Base Station,” in *2020 4th International Conference on Recent Advances in Signal Processing, Telecommunications Computing (SigTelCom)*, Aug. 2020, pp. 1–5, doi: 10.1109/SigTelCom49868.2020.9199019.

[8] P. Chanclou *et al.*, “Optical fiber solution for mobile fronthaul to achieve cloud radio access network,” in *2013 Future Network Mobile Summit*, Jul. 2013, pp. 1–11.

[9] Nokia, “5G New Radio Network: Use Cases, Spectrum, Technologies and Architecture,” 2021. [Online]. Available: https://onestore.nokia.com/asset/205407?_ga=2.162353212.1872235669.1612373348-1681341718.1603974422.

[10] S. Shew, “Transport network support of IMT-2020/5G,” 2018. [Online]. Available: https://www.itu.int/dms_pub/itu-t/opb/tut/T-TUT-HOME-2018-PDF-E.pdf.

[11] M. Klinkowski, “Latency-Aware DU/CU Placement in Convergent Packet-Based 5G Fronthaul Transport Networks,” *Appl. Sci.*, vol. 10, no. 21, p. 7429, 2020.

[12] 3GPP TR 38.801, “Technical Specification Group Radio Access Network; Study on new radio access technology: Radio access architecture and interfaces,” 2017.

[Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3056.

[13] O. Adamuz-Hinojosa, P. Munoz, J. Ordonez-Lucena, J. J. Ramos-Munoz, and J. M. Lopez-Soler, “Harmonizing 3GPP and NFV Description Models: Providing Customized RAN Slices in 5G Networks,” *IEEE Veh. Technol. Mag.*, vol. 14, no. 4, pp. 64–75, Dec. 2019, doi: 10.1109/MVT.2019.2936168.

[14] L. Anttila, “Digital Front-End Signal Processing with Widely-Linear Signal Models in Radio Devices,” Tampere University of Technology, 2011.

[15] M. Valkama and M. Renfors, “An Introduction to Radio Architectures and Signal Processing,” *Dep. Commun. Eng. Tampere Univ. Technol.*, p. 84, 2011, [Online]. Available: http://www.cs.tut.fi/kurssit/TLT-9707/presentations/Radio-Arch-SP-short-2pp.pdf.

[16] M. Valkama, A. Springer, and G. Hueber, “Digital signal processing for reducing the effects of RF imperfections in radio devices — An overview,” in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, May 2010, pp. 813–816, doi: 10.1109/ISCAS.2010.5537444.

[17] S. Kim, B. Kim, Y. Lee, S. Kim, and H. Shin, “A 28GHz Direct Conversion Receiver in 65nm CMOS for 5G mmWave Radio,” in *2019 International SoC Design Conference (ISOCC)*, Oct. 2019, pp. 29–30, doi: 10.1109/ISOCC47750.2019.9027756.

[18] C. Bowick, *RF Circuit Design*. Elsevier Science, 2011.

[19] J. Love, *RF Front-End: World Class Designs*. Elsevier Science, 2009.

[20] Anh-Vu Pham, D. P. Nguyen, and M. Darwish, “High efficiency power amplifiers for 5G wireless communications,” in *2017 10th Global Symposium on Millimeter-Waves*, May 2017, pp. 83–84, doi: 10.1109/GSMM.2017.7970327.

[21] M. M. Mowla and S. M. Hasan, “Performance improvement of PAPR reduction for OFDM signal in LTE system,” *Int. J. Wirel. Mob. Networks*, vol. 5, pp. 35–47, 2013, doi: 10.5121/ijwmn.2013.5403.

[22] L. Besser and R. Gilmore, *Practical RF Circuit Design for Modern Wireless Systems: Passive Circuits and Systems Passive Circuits and Systems, Volume 1*. Artech House, 2002.

[23] K. Lin, L. Ji, M. Wei, H. Chen, and J. Tseng, “Design and Analysis of a Low Noise Amplifier for 5G Systems,” in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, May 2018, pp. 1–2, doi: 10.1109/ICCE-China.2018.8448590.

[24] B. Schweber, “Get the Most Out of Exotic Processes for 5G LNAs,” *Digi-Key’s North American Editors*, 2018. https://www.digikey.com/en/articles/get-the-most-out-of-exotic-processes-for-5g-lnas (accessed Feb. 22, 2021).

[25] B. Schweber, “Understanding the Mixer’s Role in an RF-receiver Design,” 2013. https://www.digikey.com/en/articles/understanding-the-mixers-role-in-an-rf-receiver-design#:~:text=The mixer is a critical,to a known%2C fixed frequency (accessed Feb. 24, 2021).

[26] Chunyan Wang, M. O. Ahmad, and M. N. S. Swamy, “A CMOS current-controlled oscillator and its applications,” in *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS ’03.*, May 2003, vol. 1, pp. I–I, doi: 10.1109/ISCAS.2003.1205683.

[27] I. Janiszewski, B. Hoppe, and H. Meuth, “Precision and performance of numerically controlled oscillators with hybrid function generators,” in *Proceedings of the 2001 IEEE International Frequncy Control Symposium and PDA Exhibition (Cat. No.01CH37218)*, Jun. 2001, pp. 744–752, doi: 10.1109/FREQ.2001.956374.

[28] E. Tell, “Design of Programmable Baseband Processors,” Institutionen för systemteknik, 2005.

[29] A. Gatherer, H. Zhu, and M. Erez, “Chapter 18 - Baseband architectures to support wireless cellular infrastructure: History and future evolution,” in *Academic Press Library in Mobile and Wireless Communications*, S. K. Wilson, S. Wilson, and E. Biglieri, Eds. Oxford: Academic Press, 2016, pp. 689–705.

[30] G. P. Fettweis *et al.*, “5G-and-Beyond Scalable Machines,” in *2019 IFIP/IEEE 27th International Conference on Very Large Scale Integration (VLSI-SoC)*, Oct. 2019, pp. 105–109, doi: 10.1109/VLSI-SoC.2019.8920308.

[31] Semiconductor Industry Associtation, “5G Wireless Infrastructure Semiconductor Analysis,” 2020. [Online]. Available: https://www.semiconductors.org/wp-content/uploads/2020/07/SIA-5G-Report_2.pdf.

[32] W.-K. Chen, *The VLSI Handbook, Second Edition (Electrical Engineering Handbook)*. USA: CRC Press, Inc., 2006.

[33] S. Banna, “Scaling challenges of FinFET technology at advanced nodes and its impact on SoC design (Invited),” in *2015 IEEE Custom Integrated Circuits Conference (CICC)*, 2015, pp. 1–8, doi: 10.1109/CICC.2015.7338378.

[34] S. Callender, W. Shin, H. Lee, S. Pellerano, and C. Hull, “FinFET for mm Wave - Technology and Circuit Design Challenges,” in *2018 IEEE BiCMOS and Compound Semiconductor Integrated Circuits and Technology Symposium (BCICTS)*, Oct. 2018, pp. 168–173, doi: 10.1109/BCICTS.2018.8551125.

[35] M. Keating and P. Bricaud, *Reuse Methodology Manual: For System-on-a-Chip Designs*, 3rd Editio. Springer US, 2012.

[36] Ö. Aydin and O. Uçar, “Design for Smaller, Lighter and Faster ICT Products: Technical Expertise, Infrastructures and Processes,” *Adv. Sci. Technol. Eng. Syst. J.*, vol. 2, pp. 1114–1128, 2017, doi: 10.25046/aj0203141.

[37] Y. Mathys and A. Chatelain, “Verification strategy for integration 3G baseband SoC,” in *Proceedings 2003. Design Automation Conference (IEEE Cat. No.03CH37451)*, Jun. 2003, pp. 7–10, doi: 10.1145/775832.775835.

[38] T. Joentakanen, “Evaluation of HLS modules for ASIC backend,” Tampere University of Technology, 2017.

[39] S. Ali, S. Walid, and D. Steinbach, “White paper on machine learning in 6G wireless communication networks,” Oulu, Finland, 2020. [Online]. Available: http://urn.fi/urn:isbn:9789526226736.

[40] F. Speicher *et al.*, “Methodology for improved event-driven system-level simulation of an RF transceiver subsystem for wireless SoCs,” in *2018 13th International Conference on Design Technology of Integrated Systems In Nanoscale Era (DTIS)*, Apr. 2018, pp. 1–4, doi: 10.1109/DTIS.2018.8368550.

[41] F. Vahid, *Digital Design with RTL Design, VHDL, and Verilog*. Wiley, 2010.

[42] A. Kumar and P. R. Panda, “Front-End Design Flows for Systems on Chip: An Embedded Tutorial,” in *2010 23rd International Conference on VLSI Design*, Jan. 2010, pp. 417–422, doi: 10.1109/VLSI.Design.2010.70.

[43] A. Rautakoura, M. Käyrä, T. D. Hämäläinen, W. Ecker, E. Pekkarinen, and M. Teuho, “Kamel: IP-XACT compatible intermediate meta-model for IP generation,” in *2020 23rd Euromicro Conference on Digital System Design (DSD)*, Aug. 2020, pp. 325–331, doi: 10.1109/DSD51259.2020.00060.

[44] A. Arun, S. Stelmach, R. Venkatasubramanian, J. Flores, C. Jitlal, and F. Cano, “Perils of power prediction in early power-integrity analysis,” in *2014 IEEE Dallas Circuits and Systems Conference (DCAS)*, Oct. 2014, pp. 1–5, doi: 10.1109/DCAS.2014.6965335.

[45] M. Govind and C. Alatagi, “Estimation of Power Dissipation of CMOS and finFET based 6T SRAM Memory,” 2016.

[46] L. Lavagno, I. L. Markov, G. Martin, and L. K. Scheffer, *Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology: Circuit Design, and Process Technology, Second Edition*. CRC Press, 2016.

[47] M. Haataja, “Register-transfer level power estimation and reduction methodologies of digital system-on-chip building blocks,” University of Oulu, Faculty of Information Technology and Electrical Engineering, 2016.

[48] K. Schuegraf, M. C. Abraham, A. Brand, M. Naik, and R. Thakur, “Semiconductor Logic Technology Innovation to Achieve Sub-10 nm Manufacturing,” *IEEE J. Electron Devices Soc.*, vol. 1, no. 3, pp. 66–75, Mar. 2013, doi: 10.1109/JEDS.2013.2271582.

[49] N. Draeger, “FinFETs Give Way to Gate-All-Around,” *Lam Research Blog*, 2020. https://blog.lamresearch.com/finfets-give-way-to-gate-all-around/ (accessed Apr. 22, 2021).

[50] Burch, Najm, Yang, and Trick, “McPOWER: a Monte Carlo approach to power estimation,” in *1992 IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1992, pp. 90–97, doi: 10.1109/ICCAD.1992.279392.

[51] P. Landman, “High-level power estimation,” in *Proceedings of 1996 International Symposium on Low Power Electronics and Design*, Aug. 1996, pp. 29–35, doi: 10.1109/LPE.1996.542726.

[52] S. Reda and A. N. Nowroz, “Power Modeling and Characterization of Computing Devices: A Survey,” *Found. Trends Electron. Des. Autom.*, vol. 6, no. 2, pp. 121–216, Feb. 2012, doi: 10.1561/1000000022.

[53] M. Barocci, L. Benini, A. Bogliolo, B. Ricco, and G. De Micheli, “Lookup table power macro-models for behavioral library components,” in *Proceedings IEEE Alessandro Volta Memorial Workshop on Low-Power Design*, Mar. 1999, pp. 173–

181, doi: 10.1109/LPD.1999.750418.

[54] W. T. Hsieh, C. C. Shiue, and C.-N. Liu, “A novel approach for high-level power modeling of sequential circuits using recurrent neural networks,” *Proc. - IEEE Int. Symp. Circuits Syst.*, pp. 3591–3594, Dec. 2005, doi: 10.1109/ISCAS.2005.1465406.

[55] Y. Nasser *et al.*, “NeuPow: A CAD Methodology for High-Level Power Estimation Based on Machine Learning,” *ACM Trans. Des. Autom. Electron. Syst.*, vol. 25, no. 5, Aug. 2020, doi: 10.1145/3388141.

[56] S. Schuermans and R. Leupers, *Power Estimation on Electronic System Level using Linear Power Models*. Springer International Publishing, 2018.

[57] “IEEE Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems,” *IEEE Std 1801-2018*, pp. 1–548, Mar. 2019, doi: 10.1109/IEEESTD.2019.8686430.

[58] S. Alipour, B. Hidaji, and A. S. Pour, “Circuit level, static power, and logic level power analyses,” in *2010 IEEE International Conference on Electro/Information Technology*, May 2010, pp. 1–4, doi: 10.1109/EIT.2010.5612180.

[59] H. Kawauchi, I. Taniguchi, H. Tomiyama, and M. Fukui, “Accurate and efficient RTL power estimation based on power contour model,” *Electr. Eng. Japan*, vol. 182, no. 3, pp. 48–56, 2013, doi: https://doi.org/10.1002/eej.22340.

[60] R. A. Bergamaschi and Y. W. Jiang, “State-based power analysis for systems-on-chip,” in *Proceedings 2003. Design Automation Conference (IEEE Cat. No.03CH37451)*, Jun. 2003, pp. 638–641, doi: 10.1145/775832.775992.

[61] L. Benini, R. Hodgson, and P. Siegel, “System-level power estimation and optimization,” in *Proceedings. 1998 International Symposium on Low Power Electronics and Design (IEEE Cat. No.98TH8379)*, Aug. 1998, pp. 173–178, doi: 10.1145/280756.280881.

[62] J. Ktari and M. Abid, “System Level Power and Energy Modeling for Signal Processing Applications,” in *2007 2nd International Design and Test Workshop*, Dec. 2007, pp. 218–221, doi: 10.1109/IDT.2007.4437463.

[63] S. K. Rethinagiri, R. Ben Atitallah, S. Niar, E. Senn, and J.-L. Dekeyser, “Hybrid system level power consumption estimation for FPGA-based MPSoC,” in *2011 IEEE 29th International Conference on Computer Design (ICCD)*, Oct. 2011, pp. 239–246, doi: 10.1109/ICCD.2011.6081403.

[64] S. K. Rethinagiri, R. Ben Atitallah, S. Niar, E. Senn, and J.-L. Dekeyser, “Fast and accurate hybrid power estimation methodology for embedded systems,” in *Proceedings of the 2011 Conference on Design Architectures for Signal Image Processing (DASIP)*, Nov. 2011, pp. 1–7, doi: 10.1109/DASIP.2011.6136852.

[65] J. Laurent, N. Julien, E. Senn, and E. Martin, “Functional level power analysis: an efficient approach for modeling the power consumption of complex processors,” in *Proceedings Design, Automation and Test in Europe Conference and Exhibition*, Feb. 2004, vol. 1, pp. 666-667 Vol.1, doi: 10.1109/DATE.2004.1268921.

[66] P. Soulard and Y. Xu, “Accurate System Level Power Estimation through Fast

Gate-Level Power Characterization," *Design and Reuse*. https://www.design-reuse.com/articles/17971/system-level-power-estimation.html.

[67] J. Knödtel, W. Schwabe, T. Lieske, M. Reichenbach, and D. Fey, "A Novel Methodology for Evaluating the Energy Consumption of IP Blocks in System-Level Designs," in *2018 28th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Jul. 2018, pp. 46–53, doi: 10.1109/PATMOS.2018.8464149.

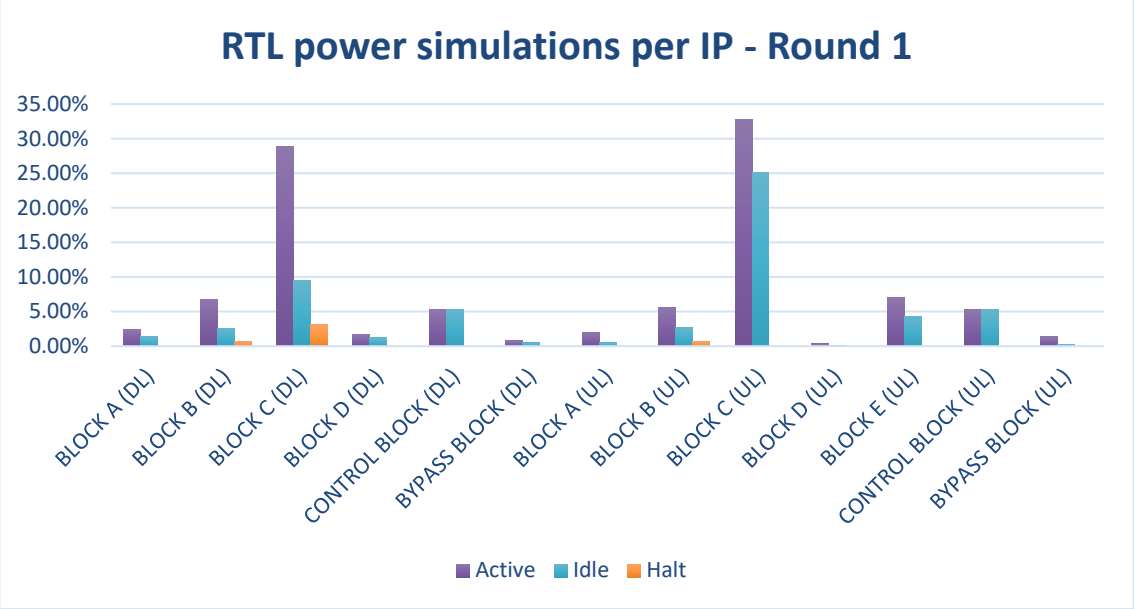
[68] S. Hesselbarth, T. Baumgart, and H. Blume, "Hardware-assisted power estimation for design-stage processors using FPGA emulation," in *2014 24th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2014, pp. 1–8, doi: 10.1109/PATMOS.2014.6951877.

[69] J. Lorandel, J.-C. Prévotet, and M. Hélard, "Fast Power and Performance Evaluation of FPGA-Based Wireless Communication Systems," *IEEE Access*, vol. 4, pp. 2005–2018, 2016, doi: 10.1109/ACCESS.2016.2559781.

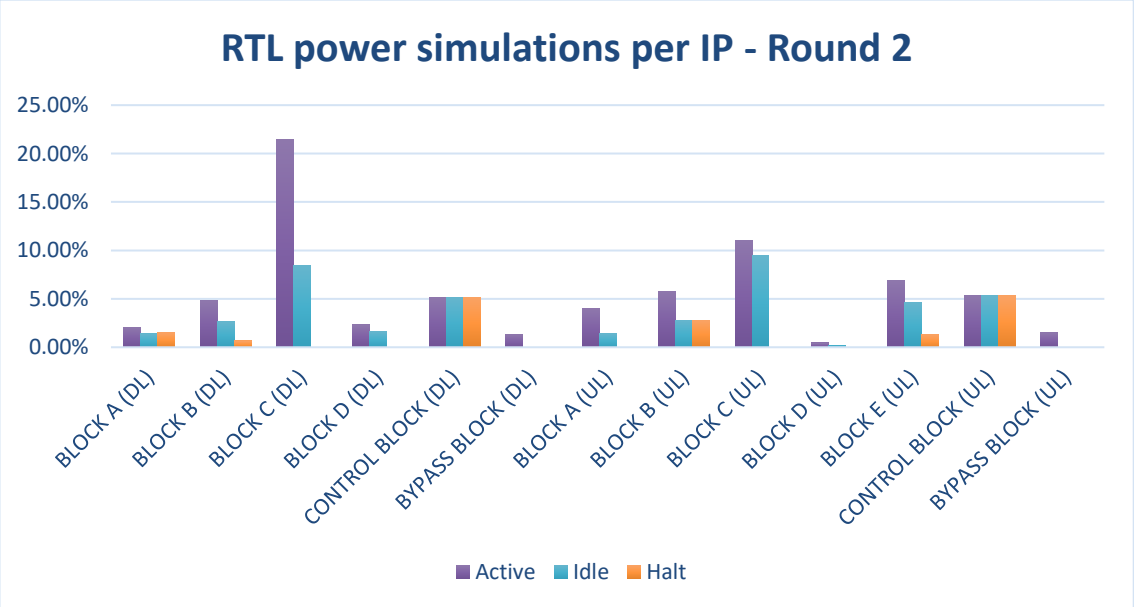
[70] J. R. Taylor, *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements*. University Science Books, 1982.

APPENDIX A: RTL POWER SIMULATIONS PER IP

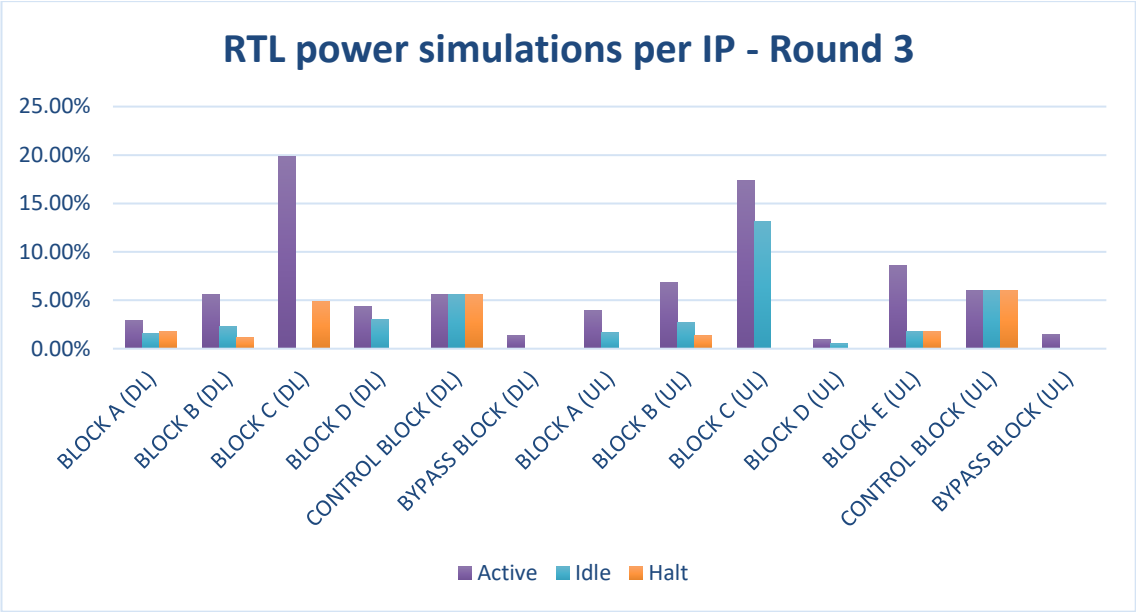
This appendix shows the four rounds of RTL power simulations per IP and per mode of operation. 100% equals the overall subsystem reference power of Section 3.3.2.



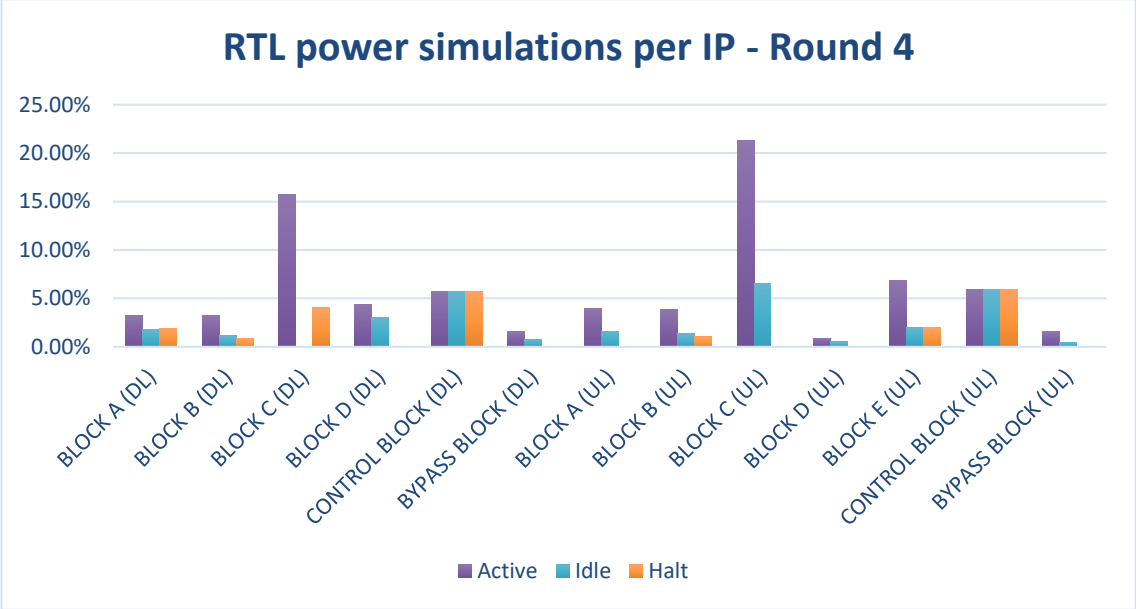
a) *Round 1 of RTL power simulations per IP and per mode of operation. 100% equals the overall sub-system reference power of Section 3.3.2.*



b) *Round 2 of RTL power simulations per IP and per mode of operation. 100% equals the overall sub-system reference power of Section 3.3.2.*

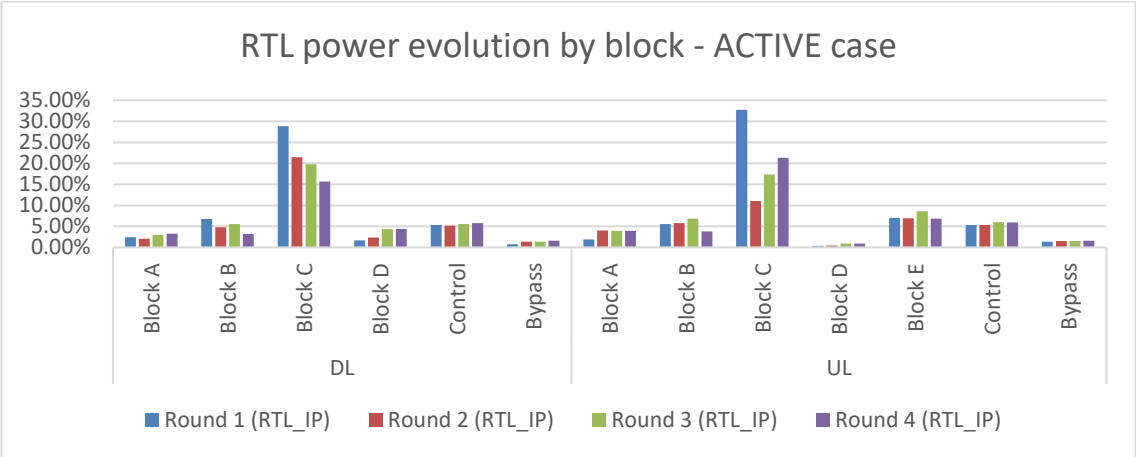


c) Round 3 of RTL power simulations per IP and per mode of operation. 100% equals the overall sub-system reference power of Section 3.3.2.

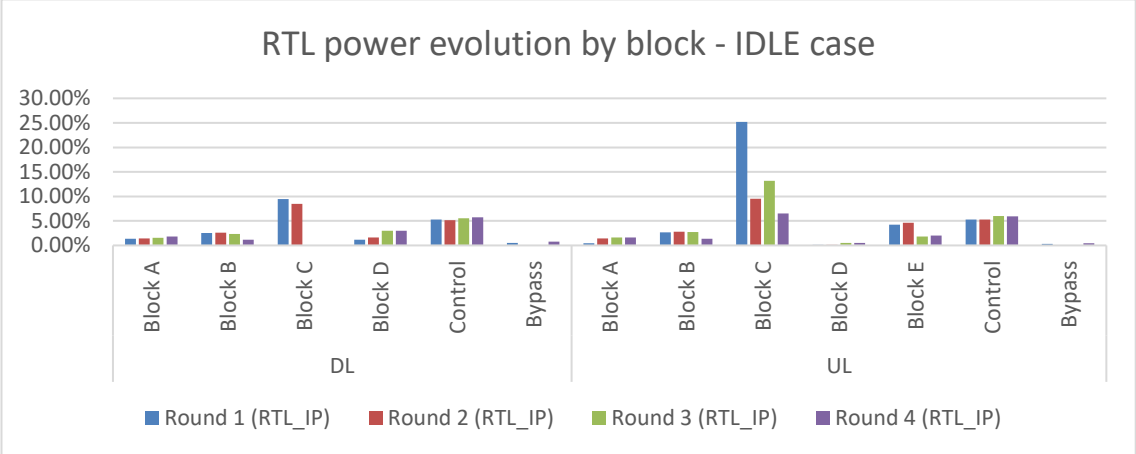


d) Round 4 of RTL power simulations per IP and per mode of operation. 100% equals the overall sub-system reference power of Section 3.3.2.

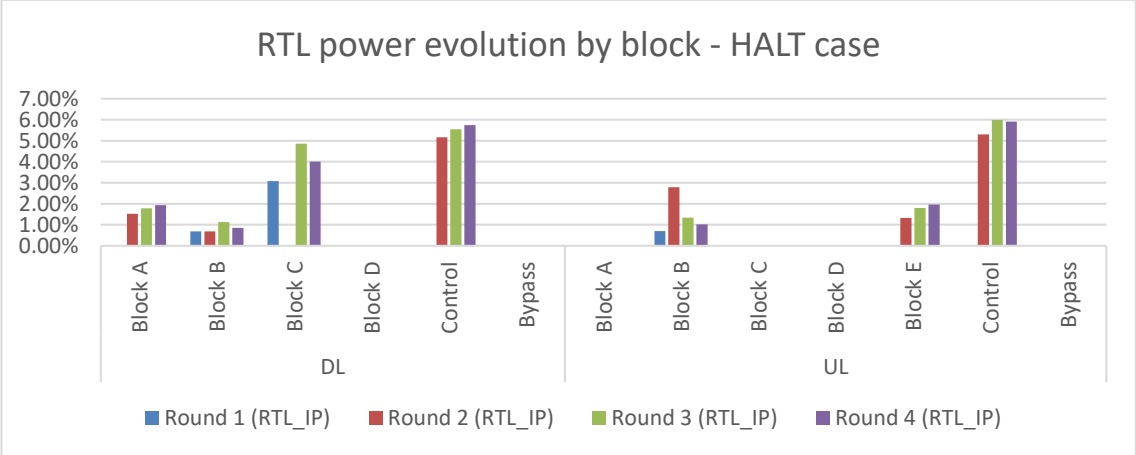
APPENDIX B: POWER EVOLUTION BY BLOCK



a) RTL power evolution by block for the ACTIVE case.



b) RTL power evolution by block for the IDLE case.



c) RTL power evolution by block for the HALT case.